GoPxL for Gocator 2D Smart Cameras

USER MANUAL

Gocator 1050 & 1120 2D Smart Cameras

Firmware version: 1.4.x.x

Document revision: K

Copyright

Copyright © 2026 by LMI Technologies, Inc. All rights reserved.

Proprietary

This document, submitted in confidence, contains proprietary information which shall not be reproduced or transferred to other documents or disclosed to others or used for manufacturing or any other purpose without prior written permission of LMI Technologies Inc.

No part of this publication may be copied, photocopied, reproduced, transmitted, transcribed, or reduced to any electronic medium or machine readable form without prior written consent of LMI Technologies, Inc.

Trademarks and Restrictions

Gocator™ is a registered trademark of LMI Technologies, Inc. Any other company or product names mentioned herein may be trademarks of their respective owners.

Information contained within this manual is subject to change.

Contact Information

LMI Technologies, Inc.
9200 Glenlyon Parkway
Burnaby BC V5J 5J8
Canada

Telephone: +1 604-636-1011
Fax: +1 604-516-8368

www.lmi3D.com

Table of Contents

Copyright **2**
Table of Contents **3**
Introduction **10**
Read Before Use **11**
EN - English 11
Safety 11
Intended Use 11
DA - Dansk 12
Sikkerhed 12
Tilsigtet brug 12
DE - Deutsch 13
Sicherheit 13
Bestimmungsgemäßer Gebrauch 13
ES - Español 14
Seguridad 14
Uso previsto 14
FI - Suomi 15
Turvallisuus 15
Käyttötarkoitus 15
FR - Français 16
Sécurité 16
Utilisation prévue 16
HE - עברית 17
בטיחות 17
שימוש מיועד 17
IT - Italiano 18
Sicurezza 18
Uso previsto 18
JA - 日本語 19
安全性 19
用途 19
NL - Nederlands 20
Veiligheid 20
Beoogd gebruik 20
NO - Norsk 21
Sikkerhet 21
Tiltenkt bruk 21
SV - Svenska 22
Säkerhet 22
Avsedd användning 22
ZH - 简体中文版 23
安全需知 23
预期用途 23
Notes **24**
Compliance, Safety, and Intended Use **25**
Compliance Notifications 25
For customers in the US 25
Supplier Declaration of Conformity 25
Party issuing Supplier's Declaration of Conformity 26
Responsible party - US contact information 26
For customers in Canada 26
Pour utilisateurs au Canada 26
Avoid electromagnetic interferences 26
Intended Use 27
Safety 27
Handling Lens Mounts 27
Handling Hot Gocator 2D Smart Cameras 27
Gocator 2D Smart Camera Mounting 28
Cybersecurity 28
Avoiding Product Damage 28
Electrical Connections 29
ESD 29
Power 29
Ground Loops 29
Cable Connections 30
No Hot-plugging for All Connections 30
Cable signal quality 30
Optical Components 30
Sensor 30
Possible damage to image sensors due to electrostatic discharge 31
Sensor cleanliness 31
Lenses 31
Lens Dimensions 31
Mechanical Components 32
Mounting Lens Tubes 32
Camera Light 32
General Information 32
Smart Vision RMX 140 Mini Ring Light 33
Thermal Management 33
Providing Optimal Heat Dissipation 33

Using the Manual **35**
Quick Start - Installation **36**
Gocator 2D Smart Camera - Hardware Overview ... 36
Touching Hot Smart Cameras 37
Sensor Damage by ESD 37
Cables and Levels 38
Mounting the Lens 38
Connecting Power and I/O Cordset 40
Connecting the Cordset to the Device 40
Connecting Open Wire Leads to the Power Supply 41
Connecting Open Wires to Peripherals 42
Connecting Network Cordset 42
Preparing Power Supply 43
Adjusting Focus and Aperture 44
Mounting Lens Cover 46
Mounting Ring Light Adapter Plate 47
Mounting Ring Light to the Camera 49
If You Are Using the Lens Cover... 49
If You Are Not Using the Lens Cover... 49
Connecting Ring Light to the Camera 50
Wiring Trigger Input 51
Wiring Digital and Light Trigger Outputs 52
Mounting Camera 53
Specifications **55**
Gocator 2D Cameras 56
Gocator 1050-M / 1050-C / 1120-M / 1120-C 2D Smart Cameras 57
Gocator 1050-M / 1050-C / 1120-M / 1120-C 2D Mounting Plate 59
RMX140 Mounting Plates 60
IP Class 61
Maximum dimensions for lenses 61
Shock and Vibration 62
IEC 60068-2-6: Sinusoidal vibration 63
IEC 60068-2-27: Non-repetitive shock 63
IEC 60068-2-27: Repetitive shock 64
IEC 60068-2-64: Random vibration 64
Sensor Connectors 65
Gocator Power / IO Connector 65
Opto-isolated Input Specifications 66
Opto-isolated Output Specifications 67
Gocator Ethernet Connection 67
Gocator 2D Smart Camera Light Connector 68
Required Ports **70**
Key Concepts **72**
Measurement and Anchoring 72
Complex Defect Detection 72
Arrays, Batching, and Aggregation 72
Output and Control 73
User Interface Overview **75**
Status Bar 78
System Messages 78
Metrics Area 79
Keyboard and Mouse Shortcuts **80**
File Formats **83**
Data Viewer **85**
Exposure Mode 85
Displaying and Pinning Outputs 87
Using Multiple Data Viewers 91
Histogram 92
Recording and Working with Scan Data **94**
Starting, Stopping, and Recording 95
Playing Back Recorded Data 97
Using the Replay Editor 98
Creating, Saving and Loading Jobs **101**
Pro Tools: Installing CodeMeter Runtime **102**
Configuring GoPxL **106**
Connect - Networking 106
Connect - Creating a System & Networking 106
Network Settings 106
Acquire - Calibration and Image Acquisition 107
Calibrating Smart Cameras 107
Triggers 110
Trigger Settings 111
General - Exposure, HDR, Gamma, Lighting 112
Active Area 113
Color 114
Advanced Settings 115
Inspect - Measurement and Processing 117
Adding a Tool 118
Tool Connection Logic 118
Removing a Tool 119
Duplicating a Tool 119
Renaming a Tool 119

Working with Tool Chains 119
Changing Tool Display Options 120
Reordering Tools 121
Data Types 121
Understanding the Data Flow in Tool Chains 121
Connecting Tools 123
Disconnecting Tools 124
Tool Configuration 125
Inputs and Outputs 126
Arrays, Batching, and Aggregation 126
Batching 127
Aggregating 129
Batch and Array Indicators in GoPxL 130
Regions 132
Standard Regions 132
Flexible Regions 133
Working with Circular and Elliptical Regions 134
Region Rotation 136
Feature Points 136
Geometric Features 137
Fit Lines 137
Decisions 138
Measurement Anchoring 138
Array Tools 140
Array Create 141
Inputs 141
Parameters 141
Outputs 141
Array Decision 142
Inputs 142
Parameters 142
Outputs 142
Array Index 144
Inputs 144
Parameters 144
Outputs 144
Conditional Logic Tool 146
Inputs 146
Parameters 146
Outputs 147
Feature Tools 148
Feature Circle Create 149
Inputs 149
Parameters 149
Output Types 149
Constant Circle 149
Circle from Points 150
Circle from point and plane 151
Circle tangent to two lines 151
Outputs 152
Feature Dimension 154
Inputs 154
Parameters 154
Outputs 154
Feature Intersect 156
Inputs 156
Parameters 156
Outputs 157
Feature Line Create 158
Inputs 158
Parameters 158
Output Types 158
Line 159
Constant Line 159
Line from Offset 159
Perpendicular or Parallel Line from Point and Line 160
Perpendicular Line from Point to Plane 161
Intersect Line of Two Planes 162
Projected Line on Plane 162
Projected Center Line from Two Lines and a Plane 163
Outputs 163
Feature Plane Create 164
Feature Point Create 165
Inputs 165
Parameters 165
Output Types 165
Point 165
Constant Point 166
Point from Offset 166
Point from Three Planes 167
Point from Line and Circle 167

Point from Line and Line 168
Point from Line and Plane 168
Projected Point on Line 169
Projected Point on Plane 169
Outputs 170
Feature Robot Pose 171
Image Tools 172
Image Anomaly Detector 172
Inputs 174
Parameters 174
Development Workflow 175
Collect dataset 176
Implement a Preprocessing Toolchain 176
Train Model 176
Add Image Anomaly Detector Tool 176
Create New or Open Project 177
Label Frames and Choose Set 179
Configure Training 180
Test Model Performance 185
Deploy in production 186
Anomaly Tool Setup 186
Outputs 187
Store production data 188
Train using additional data 188
Training Archive 191
Inputs 191
Parameters 191
Outputs 192
Image Arithmetic 194
Inputs 194
Parameters 194
Outputs 196
Image Barcode 196
Inputs 196
Parameters 197
Outputs 199
Image Blob 202
Inputs 202
Parameters 202
Outputs 205
Image Circle 207
Inputs 207
Parameters 207
Outputs 209
Image Classifier 211
Inputs 211
Development Workflow 211
Collect dataset 211
Implement a Preprocessing Toolchain 212
Train Model 212
Add the Tool 212
Create New or Open Project 212
Add Classes 214
Label Frames 214
Configure Training 216
Model Performance Testing 219
Parameters 219
Outputs 220
Image Converter 224
Inputs 224
Parameters 224
Outputs 224
Image Feature Mask 225
Inputs 225
Parameters 225
Outputs 226
Image Filter 227
Inputs 227
Parameters 227
Outputs 229
Image Intensity 230
Inputs 230
Parameters 230
Outputs 231
Image Line 232
Inputs 232
Parameters 232
Outputs 233
Image Matrix Code 235
Inputs 235
Parameters 236
Outputs 237
Image OCR 240
Inputs 241

Parameters 241
Outputs 243
Image Pattern 244
Inputs 244
Creating a Pattern 245
Editing a Pattern 247
Parameters 248
Pattern Parameters 248
Tool (Matching) Parameters 251
Outputs 254
Image Pixel Count 256
Inputs 256
Parameters 256
Outputs 256
Image QR Code 258
Inputs 258
Parameters 259
Outputs 261
Image Track 264
Inputs 265
Parameters 265
Outputs 267
Image Transform 268
Inputs 268
Parameters 268
Outputs 269
Measurement Formula Tool 270
Inputs 270
Parameters 270
Outputs 271
Script Tool 272
Included Python Modules 272
Inputs 272
Parameters 273
Outputs 274
Functions and Classes 274
Getting Input 274
Drawing Graphics 276
Sending Output 277
Logging 280
Memory 280
Data Structures 280
Examples 287
User Data Input Tool 308
Inputs 308
Parameters 308
Outputs 309
Communicate - PLC, Digital, GDP, HMI 309
GoPxL Data Protocol 312
GoPxL Data Protocol 315
Modbus TCP Protocol 315
Modbus TCP Protocol Format - PLC Information 319
Messages 319
Connections Map 320
Supported Block Types 321
Control Input 321
Control Output 322
System State 322
Sensor Group State 323
Stamp 323
Measurement 324
String 324
EtherNet/IP Protocol 324
EtherNet/IP Protocol Format - PLC Information 331
Supported EtherNet/IP Objects 331
Object and Register Assembly Mapping 332
Supported Block Types 333
Identity Object (Class 0x01) 334
TCP/IP Object (Class 0xF5) 334
Ethernet Link Object (Class 0xF6) 335
Assembly Object (Class 0x04) 335
Explicit Messaging Command Input 336
Input Structure 336
Explicit Messaging Command Output 337
Output Structure 337
Explicit Messaging Sensor Group State 337
Output Structure 338
Explicit Messaging System State 338
Output Structure 338
Explicit Messaging Sample State 339
Stamp Block 340
Measurement Value Block 340

String Block 340
Explicit Messaging User Data Input 340
User Data Input Block 341
Implicit Messaging Input 341
Input Structure 341
Implicit Messaging Input Large 342
Input Structure 342
Implicit Messaging Output 343
Stamp Block 344
Measurement Value Block 344
String Block 344
Ethernet ASCII Protocol 345
Ethernet ASCII Protocol Format - PLC Information 350
Commands and Formats 351
Ethernet ASCII client commands 351
Data Output Format Mode 357
PROFINET Protocol 359
Using PROFINET on a PC 363
PROFINET Protocol Format - PLC Information 364
Connections Map 364
Assembly Objects 364
Supported Block Types 365
Control Input 365
Control Output 366
System State 367
Sensor Group State 367
Stamp 368
Measurement 368
String 368
Digital Output 369
HMI 373
Data Storage Service 378
Date and Time Service 384
Monitor - Measurement Stats, Health & Performance 387
Measurements 387
Health 388
Performance 391
Manage - Sensor Management and Maintenance 393
Settings 393
Autostart 394
Administrator Login 394
Jobs 396
Characters Allowed in Job Names 398
Maintenance 399
Software Upgrade 400
Backup and Restore 400
Factory Restore 401
Support 401
Running GoPxL on a Windows PC 403
System Requirements 404
Launching GoPxL on a Windows PC 404
Running GoPxL Manager in Administrator Mode 407
Closing GoPxL Manager and Stopping All PC Instances 408
Application Settings 408
Installing Licensing Software 410
Instance Settings 411
Creating a Diagnostic File 414
Accelerating a Sensor 414
Stopping Acceleration 418
Loading Scan Data 419
GoPxL Emulator (Beta Feature) 422
GoHMI and GoHMI Designer 425
Overview of Managers and Settings Pages 429
Enabling and Configuring GoHMI in GoPxL 430
Launching GoHMI Designer 435
Opening, Editing, and Saving a Project 439
Publishing Changes or Exiting 440
Changing Mapped Outputs in GoPxL 441
Importing and Exporting an HMI 441
Using a Data Viewer in an HMI 443
Limiting Flash Memory Write Operations 448
GoPxL SDK and REST API 449
Installation Guide 449
Location of GoPxL SDK and Documentation 449
Visual Studio Solution Files and Linux Makefile 449
Building the SDK 449
Running SDK Applications 450
System Requirements 450
Accessing Documentation 450
Sample Projects 450
Header Files 450

Getting Started with the SDK 451
Connecting a Device 451
GoSystem Class Member Functions 452
Discover Devices 452
Configure Device Settings 453
Accessing Sensor Resources 454
Accessing Scanner Resources 454
Accessing an Array Element 455
Receive Device Data 455
Receive Health and Metrics Information 457
Accelerated Devices 458
GDP Data Types 458
Value Types 458
Output Types 458
GoDataSet 459
Measurement Values and Decisions 460
Limiting Flash Memory Write Operations 461
Python GDK 462
Requirements and Installation 462
Developing Custom Tools 463
Enabling the Local Python Tool Server 466
Debugging Custom Tools 467
Deploying Your Custom Tools 469
Integrations 472
Protocols (PLCs and other hardware) 472
Utilities 473
GoPxL Discovery Tool 473
Replay Utility Tool 476
CSV File Format 477
Info 478
DeviceInfo 478
RecordingFilter 479
Profile 479
RawProfile 480
Part 481
SurfacePointCloud 481
GoPxL Add-on Tool Manager 482
Adding Beta Tools 484
Removing Beta Tools 487
Pattern Editor 489
Launching the Pattern Editor 490
Overview of the Editor 492
Models 493
Adding and Removing Features Manually 495
Setting Required and Locating Features 500
Model Creation Settings and Rebuilding 501
Coarseness Levels 502
Thresholds 503
Tracking Inertia 503
Feature Selection 504
Saving and Discarding Changes 505
Miscellaneous 506
Accessories 507
Troubleshooting 508
Return Policy 509
Software Licenses 510
Support 531
Contact 532

Introduction

This documentation describes how to connect, configure, and use a Gocator 2D smart camera using the GoPxL software.

Before using the Gocator 2D Smart Camera, see *Read Before Use* on page 11. For quick start installation information, see *Quick Start - Installation* on page 36.

For a breakdown of this documentation, see *Using the Manual* on page 35.

The documentation applies to the following:

- Gocator 1000 and 1100 smart cameras
- X64-based PC (Intel/AMD) running Windows 10 (for running GoPxL on a PC)

⚠	GoPxL does not support the original, pre-NX version of GoMax.

🗋	If you need to use the Anomaly Detector tool, the Image tools, or any other Pro Tool in GoPxL on PC, you must purchase a Pro Tools license and dongle, and download the "GoPxL Pro Utilities" package (14658-x.x.x.x_SOFTWARE_GoPxL_Pro_Utilities.exe).

Notational Conventions
This documentation uses the following notational conventions:

Read Before Use

EN - English

Safety

Before using the product, read these safety instructions. Observe the warnings at all times. Use the product only as stated in *Intended Use* on page 27.

⚠ WARNING

This product can expose you to chemicals including Lead and Lead Compounds, which is known to the State of California to cause cancer and birth defects or other reproductive harm. For more information go to: www.P65Warnings.ca.gov.

⚠ CAUTION

Risk of burns

The product in operation can reach temperature levels which could cause burns.

⚠ CAUTION

Injury by a falling product

The falling product can cause injury.

⚠ CAUTION

Risk of cuts by sharp edges

The product can have sharp edges.

Intended Use

Intended use of Allied Vision product is the integration into vision systems by professionals. All Allied Vision product is sold in a B2B setting.

DA - Dansk

Sikkerhed

Læs sikkerhedsanvisningerne, før produkt bruges. Overhold alle advarsler. Brug kun produkt som anført i *Intended Use* på side 27.

⚠ FORSIGTIG

Forbrændingsfare

Når produkt bruges, kan det blive meget varmt og forårsage forbrændinger.

⚠ FORSIGTIG

Kvæstelser, hvis produkt falder ned

Falder produkt ned, kan dette forårsage kvæstelser.

⚠ FORSIGTIG

Fare for skarpe kanter

Produktet kan have skarpe kanter.

Tilsigtet brug

Allied Vision produktets tilsigtede brug er en indbygning i et visionssystem, udført af fagfolk. Alle Allied Vision produkter sælges i B2B.

DE - Deutsch

Sicherheit

Bevor Sie das Produkt benutzen, lesen Sie diese Sicherheitshinweise. Beachten Sie diese Hinweise immer. Verwenden Sie das Produkt nur wie beschrieben in *Intended Use* auf Seite 27.

⚠ VORSICHT

Gefahr von Verbrennungen

Im Betrieb kann das Produkt Temperaturen erreichen, die zu Verbrennungen führen.

⚠ VORSICHT

Verletzung durch das fallende Produkt

Das fallende Produkt kann Verletzungen verursachen.

⚠ VORSICHT

Schnitte durch scharfe Kanten

Das Produkt kann scharfe Kanten haben.

Bestimmungsgemäßer Gebrauch

Allied Vision Produkte sind bestimmt für die Integration in Bildverarbeitungssysteme durch Fachpersonal. Alle Allied Vision Produkte werden in einer B2B-Umgebung verkauft.

ES - Español

Seguridad

Antes de utilizar el producto lea estas instrucciones de seguridad. Observe las advertencias en todo momento. Utilice el producto solo tal y como se estipula en el *Intended Use* en la página 27.

⚠ ADVERTENCIA

Este producto puede exponerle a químicos incluyendo plomo y compuestos de plomo, que son conocidos por el Estado de California como causantes de cáncer y defectos de nacimiento u otros daños reproductivos. Para mayor información, visite www.P65Warnings.ca.gov.

⚠ ATENCIÓN

Riesgo de quemaduras

El producto en funcionamiento puede alcanzar temperaturas que podrían provocar quemaduras.

⚠ ATENCIÓN

Lesiones en caso de que el producto se cae

Si el producto se cae puede provocar lesiones.

⚠ ATENCIÓN

Riesgo de cortes por bordes afilados

El producto puede tener bordes afilados.

Uso previsto

El uso previsto del producto Allied Vision es la integración en el sistema de visión por parte de profesionales. Todos los productos Allied Vision se venden dentro de una relación B2B.

FI - Suomi

Turvallisuus

Lue nämä turvallisuusohjeet ennen tuotteen käyttöä. Noudata tuotetta joka hetki. Käytä tuotteen ainoastaan kohdassa *Intended Use* sivulla 27 kuvatulla tavalla.

⚠ HUOMIO

Palovammojen vaara

Käytössä olevan tuotteen saavuttamat lämpötilatasot voivat aiheuttaa palovammoja.

⚠ HUOMIO

Putoavan tuotteen aiheuttamat vammat

Putoava tuote voi aiheuttaa vammoja.

⚠ HUOMIO

Terävien reunojen aiheuttama viiltovaara

Tuotteessa voi olla teräviä reunoja.

Käyttötarkoitus

Allied Vision -tuotteen käyttötarkoitus on integrointi kuvajärjestelmiin ammattilaisten toimesta. Kaikki Allied Vision -tuotteet myydään B2B-ympäristössä.

FR - Français

Sécurité

Veuillez lire ces consignes de sécurité avant d'utiliser le produit. Respectez continuellement les avertissements. Utilisez le produit uniquement comme indiqué sous *Intended Use* page 27.

⚠ ATTENTION

Risque de brûlures

Le produit en service peut atteindre des niveaux de température susceptibles d'entraîner des brûlures.

⚠ ATTENTION

Blessures en cas de chute du produit

La chute de la produit peut entraîner des blessures.

⚠ ATTENTION

Risque de coupures sur des bords tranchants

Le produit peut présenter des bords tranchants.

Utilisation prévue

L'utilisation prévue du produit Allied Vision est son intégration dans des systèmes de vision par le soin de professionnels. Tout produit Allied Vision est vendu dans un cadre B2B.

HE - עברית

בטיחות

לפני השימוש במוצר, עליך לקרוא את הוראות הביטחון האלו. עליך לממש הוראות ביטחון אלו תמיד.

השימוש במצלמה הוא רק לפי מה שכתוב ב"כוונת השימוש" ((*Intended Use*) בעמוד 27)

⚠ זהירות

סכנת כוויה

בזמן הפעלת המוצר עלולות לעלות טמפרטורות גבוהות, שיכולות לגרום לכוויות.

⚠ זהירות

פציעה מנפילת המוצר

נפילת המוצר עלולה לגרום לפציעה.

⚠ זהירות

סכנת חתכים על ידי קצוות חדים

למוצר יכולים להיות קצוות חדים.

שימוש מיועד

מוצרי AlliedVision מיועדים לשילוב במערכות ממוחשבת לעיבוד צילומים ע"י אנשי מקצוע.

כל מוצרי AlliedVision נמכרים לשימוש בסביבת B2B.

IT - Italiano

Sicurezza

Leggere queste istruzioni per la sicurezza prima di utilizzare il prodotto. Osservare sempre tutte le avvertenze. Utilizzare il prodotto come descritto alla sezione *Intended Use* a pagina 27.

⚠ ATTENZIONE

Pericolo di ustioni

Durante il funzionamento, il prodotto può raggiungere temperature elevate che possono essere causa di ustioni.

⚠ ATTENZIONE

Lesioni dovute alla caduta del prodotto

Il prodotto può causare delle lesioni.

⚠ ATTENZIONE

Pericolo di tagliarsi sui bordi affilati

I bordi del prodotto lente possono essere affilati.

Uso previsto

Il prodotto Allied Vision è concepito per essere integrato in sistemi di monitoraggio in campo professionale. Tutti i prodotti Allied Vision sono venduti in uno scenario B2B.

JA - 日本語

安全性

本製品を使用する前に、この安全ガイドをお読みください。警告を必ず守ってください。必ず27ページのIntended Useに従って使用してください。

⚠ 注意

やけどの危険性

作動中のカメラは、やけどを引き起こす温度まで熱くなる恐れがあります。

⚠ 注意

製品の落下によるケガ

本製品が落下すると、けがをするおそれがあります。

⚠ 注意

な端部で切り傷の危険性

本製品には鋭利な部分がある場合があります。

用途

Allied Vision製品は、専門家が視覚装置に統合することを意図したものです。すべてのAllied Vision製品は、企業間取り引き用に販売されています。

NL - Nederlands

Veiligheid

Lees deze veiligheidsinstructies voordat u het product gebruikt. Neem deze waarschuwingen altijd in acht. Gebruik het product uitsluitend, zoals aangegeven in het *Intended Use* op pagina 27.

⚠ VOORZICHTIG

Risico van verbranding

Het gebruikte product, kan temperatuurwaarden bereiken die brandwonden kunnen veroorzaken.

⚠ VOORZICHTIG

Skade ved det fallende produktet

Det fallende produktet kan forårsake skade.

⚠ VOORZICHTIG

Risiko for kutt fra skarpe kanter

Het product kan scherpe randen hebben.

Beoogd gebruik

Het beoogde gebruik van het Allied Vision-product is de integratie in optische systemen door professionals. Alle Allied Vision-producten worden verkocht in de B2B-markt.

NO - Norsk

Sikkerhet

Les disse sikkerhetsinstruksene før du bruker produkt. Følg advarslene til en hver tid. Bruk kun produkt i samsvar med *Intended Use* på side 27.

⚠ FORSIKTIG

Risiko for brannskader

Produktet i bruk kan nå temperaturnivåer som kan forårsake brannskader.

⚠ FORSIKTIG

Skade ved det fallende produktet

Det fallende produktet kan forårsake skade.

⚠ FORSIKTIG

Risiko for kutt fra skarpe kanter

Produktet kan ha skarpe kanter.

Tiltenkt bruk

Den tiltenkte bruken av Allied Vision-produktet er integrering i visjonssystemer av profesjonelle. Alle Allied Vision-produkter selges i en forretning til forretning-situasjon.

SV - Svenska

Säkerhet

Läs igenom säkerhetsinstruktionerna innan du använder produkten. Var hela tiden särskilt uppmärksam på varningarna. Använd enbart produkten på det sätt som anges i *Intended Use* på sida 27.

⚠ VARNING

Risk för brännskada

Produkten i drift kan komma upp i temperaturer som kan orsaka brännskador.

⚠ VARNING

Risk för skador från fallande produkter

Fallande produkter kan förorsaka skador.

⚠ VARNING

Risk för skärsår från vassa kanter

Produkten kan ha vassa kanter.

Avsedd användning

Den avsedda användningen av Allied Vision-produkter är integrering i visionssystem av fackmän. Samtliga Allied Vision-produkter säljs i en B2B-miljö.

ZH - 简体中文版

安全需知

在使用产品之前，请阅读这些安全说明。请务必遵守相关警告 和 Intended Use 于第27 页

⚠ 注意事项

烫伤风险

在产品运行过程中，温度可能会上升，导致烧伤的危险。

⚠ 注意事项

由坠落的产品造成的伤害

产品可能会坠落并造成伤害。

⚠ 注意事项

锋利边缘割伤的风险

产品可能有锋利的边缘。

预期用途

Allied Vision 产品的预期用途是由专业人士整合到视觉系统中。所有Allied Vision 的产品均通过B2B 渠道销售。

Notes

Read this document carefully

Learn to avoid damage to your Gocator 2D Smart Camera and use it in the most safe and efficient way.

NOTICE

Damage to Gocator 2D Smart Camera and connected hardware by improper handling

Setup and operation for Gocator 2D Smart Camera systems is different than for typical machine vision systems. Components can be damaged by improper handling or use.

- Observe the safety notes and warning messages.
- Follow the instructions in *Quick Start - Installation* on page 36.

Compliance, Safety, and Intended Use

Compliance Notifications

National regulations on disposal must be followed.

For customers in the US

FCC Class B digital device

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

Supplier Declaration of Conformity

Gocator 2D Smart Camera systems comply with Part 15 of the FCC Rules. Operation is subject to the following two conditions:

1. This device may not cause harmful interference, and
2. This device must accept any interference received, including interference that may cause undesired operation.

Party issuing Supplier's Declaration of Conformity

Allied Vision Technologies GmbH
Taschenweg 2a
07646 Stadtroda, Germany
T// +49 (36428) 677-106
quality@alliedvision.com

Responsible party - US contact information

LMI Technologies Inc.
9200 Glenlyon Parkway
Burnaby, BC V5J 5J8 Canada
T: +1 833 GOC-ATOR (+1 833 462 2867)

Changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.

For customers in Canada

This apparatus complies with the Class B limits for radio noise emissions set out in the Radio Interference Regulations.

CAN ICES-3 (B) / NMB-3 (B)

Pour utilisateurs au Canada

Cet appareil est conforme aux normes classe B pour bruits radioélectriques, spécifiées dans le Règlement sur le brouillage radioélectrique.

CAN ICES-3 (B) / NMB-3 (B)

Avoid electromagnetic interferences

All cable connections are sensitive to electromagnetic interference.

- Use shielded cables only.
- We recommend using cables offered by Allied Vision.
- Avoid coiling.
- We recommend using I/Os only in environments with low electromagnetic interference.
- Moreover, avoid unnecessary bending to prevent damaging the cables.

Intended Use

Allied Vision's objective is the development, design, production, maintenance, servicing and distribution of digital cameras and components for image processing. We are offering standard products as well as customized solutions.

Intended use of Allied Vision product is the integration into Vision systems by professionals. All Allied Vision product is sold in a B2B setting.

This product is intended to be used in internal networks that are separated from public networks. For any camera operation, the customer is responsible to enable suitable protection measures. Observe the notes in *Cybersecurity* on the next page.

Unless expressly agreed otherwise, we design, manufacture, and supply in accordance with the standards of the machine vision industry.

In the event of requirements going beyond this, the customer must:

- Notify us of the special use for each model before the first order is placed so that the models in question can be separated out from the standard processes using their own part numbers, and
- Conclude a quality assurance agreement with us prior to purchasing, to define its requirements in a legally secure manner.

This may require a surcharge, as our prices are very tightly tailored to standard requirements.

Safety

This section informs about issues related to your personal safety. Descriptions explain how to avoid hazards and operate Gocator 2D Smart Camera systems safely.

Handling Lens Mounts

The lens mount thread can have sharp edges. Be careful these edges do not cut your skin when mounting or unmounting lenses.

- Avoid touching these edges with your fingers.
- Wear protective gloves for handling the lens mount of Gocator 2D Smart Camera systems.
- Follow the instructions in *Mounting the Lens* on page 38.

Handling Hot Gocator 2D Smart Cameras

A Gocator 2D Smart Camera system is throttled or powered off to avoid damage by overtemperature. However, if you hold the smart camera in your hands during operation, your skin may get hurt. If you touch the smart camera when it is heated up, we recommend wearing protective gloves.

For more information on specified temperature values and on using software features to output temperature values, see *Providing Optimal Heat Dissipation* on page 33.

For general information on thermal management, see *Thermal Management* on page 33.

Gocator 2D Smart Camera Mounting

Gocator 2D Smart Camera must be mounted using the mounting threads. If vibration is higher than specified, the smart camera system can disconnect from the mounting base. Falling Gocator 2D Smart Camera can hurt you. To avoid personal injury:

- Mount the Gocator 2D Smart Camera system according to the instructions in *Quick Start - Installation* on page 36.
- Ensure, shock and vibration do not exceed the specified range (see *Shock and Vibration* on page 62).

Cybersecurity

Before operating Gocator 2D Smart Camera in final applications and production environments, especially in open networks, take suitable measures for cybersecurity.

NOTICE

Hazard by cyber attacks

To protect your computing system from unauthorized access by criminals or other hostile parties:

- Only use your own images based on the current NVIDIA Jetson Linux Developer Guide.
- Update your images frequently when new security updates are available.
- Activate integrated protection technologies, such as: Secure Boot, OP-TEE, Disk Encryption, and Firmware TPM.

Information on protection technologies

For integrated protection technologies, such as: Secure Boot, OP-TEE, Disk Encryption, and Firmware TPM, see the Security - NVIDIA Jetson Linux Developer Guide at https://docs.nvidia.com/jetson/archives/r36.4.3/DeveloperGuide/SD/Security.

Avoiding Product Damage

To prevent material damage, read the following and understand how to safely handle and operate the Gocator 2D Smart Camera system.

Electrical Connections

The camera is connected to the network by an 8-pin M12 X-coded connector following the IEC 61076-2-109 standard for Gigabit Ethernet applications. The other electrical connections are not defined by common standards. Read specifications carefully.

LMI accessories help to avoid damage to the Gocator 2D Smart Camera system and connected components. For more information, contact your LMI Sales representative.

Gocator 2D Smart Camera can be damaged by wrong connections or voltage levels.

- For specifications, see *Sensor Connectors* on page 65.
- For instructions to avoid electronics damage, see *Quick Start - Installation* on page 36.

ESD

Follow these instructions to avoid damage to the Gocator 2D Smart Camera embedded camera system, including possible damage to the sensor, see *Possible damage to image sensors due to electrostatic discharge* on page 31.

ESD is dangerous for electronic devices, especially when tools or hands get in contact with connectors and electronic components. We recommend you to take suitable measures to avoid damage by ESD.

Power

Operating Gocator 2D Smart Camera systems beyond the specified range damages the device. The smart camera is powered using the M12 A Coded 12-pin connector with an input of 24 VDC ±10% and minimum 1.5 A without optional light or minimum 7 A with optional light, using a limited power source (LPS), according to IEC 62368-1. The smart camera system is not intended to be connected to a DC distribution network.

Only use power supplies that meet the insulation requirement according to PELV or SELV. For details, please refer to IEC 61140.

The smart camera can be damaged if they are powered with reverse polarity.

- Observe polarity in *Gocator Power / IO Connector* on page 65.

Ground Loops

Unsuitable connections can lead to different potentials between the Gocator 2D Smart Camera GND and the environmental shield/chassis GND caused by ground loops. This can damage the smart camera system and the connected devices or cause malfunctions.

- Avoid potential differences between the smart camera system’s housing and GND.
- All wiring must be done by authorized personnel, according to the corresponding technical standards.
- You may mount the smart camera system electrically isolated.

Cable Connections

Gocator 2D Smart Camera systems are ingress protected according to IP67 only when all connectors are covered by caps or when connected to cables with screw locks tightened completely. See also *Mounting Lens Cover* on page 46.

- Use cables provided by Allied Vision for best compatibility and system performance.
- Align smart camera sockets and cable plugs and tighten locking screws at 1.2 to 1.5 Nm maximum torque.

Observe general notes to avoid damage:

- Allied Vision cables were used for the IP tests as well as for the shock and vibration tests.
- Only connect cables with suitably coded connectors.
- Ensure proper orientation of cables.
- Fit connectors without strength.
- For bending of cables and minimum bending radius, see cable specifications.
- Provide strain relief to avoid damage, such as short circuits and malfunctions.

No Hot-plugging for All Connections

Gocator 2D Smart Camera systems do not support hot-plugging because it can destroy the smart and connected hardware by high inrush current. Disconnect power supplies before connecting to a smart camera system.

Cable signal quality

Noise and electromagnetic interference can disable functions of Gocator 2D Smart Camera systems.

- Avoid contact to metal surfaces, causing electromagnetic interference.
- We recommend using cables by Allied Vision.

Optical Components

Provide the following conditions to keep dirt and droplets out of the optical system of the Gocator 2D Smart Camera system, including the lens:

- Dust-free environment
- Low relative humidity
- No condensation.

When the Gocator 2D Smart Camera system or lens are stored:

- Cover the Gocator 2D Smart Camera system's lens mount with a protection foil or cap.
- Cover front and back lens with caps.
- Protect the lens tube's front glass from dirt and damage.

Sensor

Sensors are sensitive to excessive radiation: focused sunlight, UV light, lasers, and X-rays can damage the sensor. Dirt and scratches can damage the sensor as well.

Gocator 2D Smart Camera does not need additional cleaning. The included cameras are cleaned before shipping. Incorrect cleaning can damage the sensor or the filter. Therefore, never clean the sensor or the filter.

Protect the camera filter and the sensor from dirt, because dirt becomes more visible the closer it gets to the sensor. In addition, keep the back lens clean. At delivery, the camera's lens mount is protected by a circular foil. Remove the circular foil only when you are about to mount the lens.

Hold the smart camera with the camera's lens mount facing the ground to keep dirt out of the lens mount. When no lens is mounted, protect the sensor and filter reusing the circular foil or by a dust cap.

Possible damage to image sensors due to electrostatic discharge

In some rare cases, electrostatic discharge occurring on the surface of the image sensor may cause damages to particular pixel groups, which may become visible as bubbles or blobs in the image generated by the sensor. Therefore, it is very important to comply with ESD protection measures in accordance with technical standards. We recommend you to take suitable measures to avoid damage by ESD.

Sensor cleanliness

Definitions for Gocator 2D Smart Camera systems are shown below. The incident beam is not related to the lens' angle of view.

Sensor family	Aperture*	Incident beam	Visible particles quantity
Sony	f1/8.0	≥7°	0

Gocator 2D Smart Cameras are manufactured to match the requirements of typical machine vision applications. This enables a clean image for typical monitor view. Particles may become visible when the image is viewed critically, such as with contrast enhancement or edge detection.

Optical cleaning at Allied Vision

Before being shipped, each Gocator 2D Smart Camera is tested for cleanliness in order to meet the requirements of machine vision applications. For more information, see the Optical Cleaning for Allied Vision Cameras competence showcase document at www.alliedvision.com/en/support/alecs-documentation.

Advanced sensor cleanliness

If definitions for Gocator 2D Smart Camera devices do not fulfill the requirements of your application, please contact your LMI Sales representative.

Lenses

Lens Dimensions

If lenses are mounted that exceed the specified dimensions, the lens or the Gocator 2D Smart Camera system can be damaged.

- To avoid damage to the lens and to the lens tube's front glass, use only lenses that suit the dimensions of the lens tube.
- To avoid damage mainly to the sensor and filter of the camera, use only lenses that suit the dimensions of the lens mount.
- See *Maximum dimensions for lenses* on page 61.

Mechanical Components

Mounting Lens Tubes

Gocator 2D Smart Camera systems are ingress protected according to IP67 only when a lens tube is mounted. See also *Cable Connections* on page 30.

Observe for mounting lens tubes:

- Tighten screws at 1.0 Nm maximum torque.
- Follow the instructions in *Mounting Lens Cover* on page 46.

Lens dimensions

- Maximum dimensions for lenses apply only with the lens tube. Without the lens tube, there are no limitations imposed by the Gocator 2D Smart Camera design.
- Filters and extension tubes, fastening screws for adjustment rings, or other lens accessories increase the dimensions of the lens.
- For some lenses, the length increases by focusing or when the focal length is adjusted.
- Lens tubes can cause vignetting, especially with wide angle lenses.

Lens tubes for larger lenses

Current lens tubes can be used with lenses up to 70 mm length and 46 mm width.

If you want to use lenses exceeding these dimensions, please contact your LMI Sales representative.

Camera Light

General Information

We recommend you use the third-party camera light we have tested with Gocator 2D Smart Camera; for more information, see *Smart Vision RMX 140 Mini Ring Light* on the next page.

Please observe for any connected camera light: If the power output is used beyond the specified limits, Gocator 2D Smart Camera can be damaged.

- Observe the specifications in *Gocator 2D Smart Camera Light Connector* on page 68.
- This includes keeping the maximum output for **VCC-Light** below 5.9 A.
- Observe that above 700 mA, Strobe-PNP is limited to 50 ms and 10% duty cycle.

Connections for external lighting

LMI offers compatible Y-cables at different lengths allowing to separate electrical lines for lighting control by Gocator 2D Smart Camera from lighting power by external power supplies.

- To ensure proper and safe operation of the strobe output, use the preinstalled leds-rmx140 driver. This driver is used automatically if the dedicated V4L2 controls and Vimba X features are used.

Smart Vision RMX 140 Mini Ring Light

You can easily use Gocator 2D Smart Camera with Smart Vision's RMX 140 ring light. It can be mounted using the mounting plate by Allied Vision, and power consumption is exactly in the range of the lighting output specification of the smart camera. See the manufacturer data sheet for technical data and the safety notes for eye safety.

RMX 140 documentation

See the RMX 140 mini ring light data sheet at https://smartvisionlights.com.

Thermal Management

Although Gocator 2D Smart Camera systems have a heat dissipative housing, the cooling may not be sufficient for some applications. It depends on the environmental conditions if additional thermal management is required. Keep the housing temperature in the specified range of -20 °C to +65 °C; see model specifications *Gocator 2D Cameras* on page 56.

In GoPxL, you can see the current internal temperature of the smart camera on the **Monitor** > **Health** page.

NVIDIA Jetson Orin SoM temperature

The system is throttled or shut down when the specified maximum temperature is exceeded. For more information, see the NVIDIA documentation at https://developer.nvidia.com/embedded/downloads.

Providing Optimal Heat Dissipation

To prevent Gocator 2D Smart Camera from shutting down due to overtemperature, be sure to do the following:

- Operate the smart camera mounted to a base with a high thermal conductivity and large surface area.
- Use heat sinks to maximize heat dissipation.
- Provide ventilation or other active cooling of the smart camera and mounted hardware.

Connections for external lighting

LMI offers compatible Y-cables at different lengths allowing to separate electrical lines for lighting control by the Gocator 2D Smart Camera from lighting power by external power supplies.

Using the Manual

Use the following to help decide which part of this manual you need.

> The tool topics from this manual are available in GoPxL, in the **Tool Help** panel at the bottom of the interface.

Manual breakdown

Category	Description
Safety and Maintenance	Important safety and maintenance information (see *Manage - Sensor Management and Maintenance* on page 393).
Getting Started / Installation	Hardware overview and installation information (see *Quick Start - Installation* on page 36).
User Interface Overview	General overview of the GoPxL user interface (see *User Interface Overview* on page 75).
File Formats	Lists the formats used in GoPxL (see *File Formats* on page 83).
Working with Scan Data	Describes how to record scan data and how to play back recordings (see *Recording and Working with Scan Data* on page 94).
Creating, Saving and Loading Jobs	Describes how to work with jobs (see *Creating, Saving and Loading Jobs* on page 101).
Configuring Systems	Contains subsections describing how to do the following: • Perform general sensor maintenance • Create a sensor system • Align sensors • Configure acquisition • Configure inspection and processing tools • Configure control (including protocol and GoHMI) • Configure reporting For more information, see *Configuring GoPxL* on page 106.
Development Kits and API	Provides an overview of the GoPxL SDK and the REST API and describes installation of the SDK (see *GoPxL SDK and REST API* on page 449).
GoHMI	Describes how to configure GoHMI in a PC instance of GoPxL. Also describes the GoHMI Designer (see *GoHMI and GoHMI Designer* on page 425).
Integrations	Describes the industrial protocols you can use in GoPxL (see See *Integrations* on page 472).
Specifications	Provides sensor specifications and drawings (see *Specifications* on page 55).

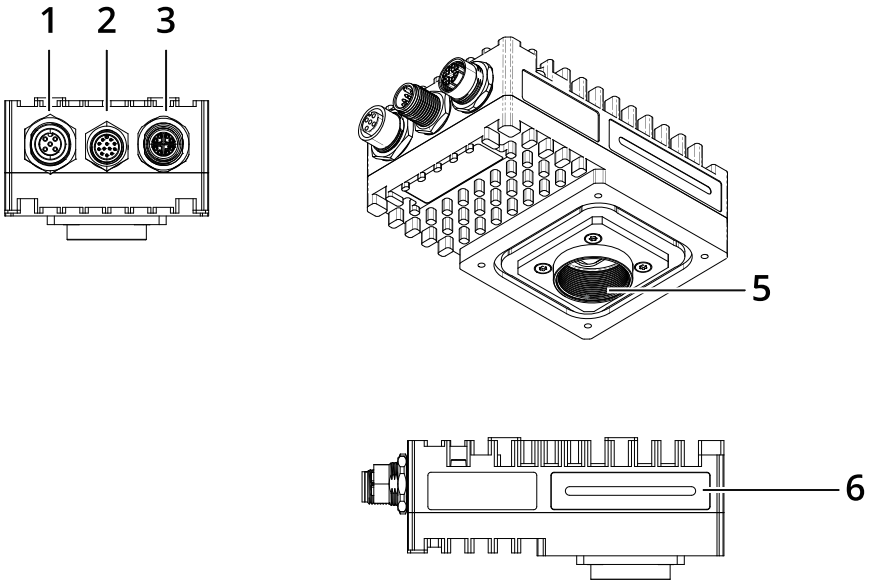
Quick Start - Installation

Before installing the Gocator 2D Smart Camera, please note the following:

- The camera is powered using the provided 24 VDC power supply. This power supply can output 10 amps, which is sufficient to power the optional RMX140 ring light (which requires 7 amps).
- The camera cannot be connected to a Master.
- The camera does not accept an encoder signal but can be triggered using a hardware signal from another Gocator 3D sensor (digital output part detection) or an inductive, capacitive, or proximity sensor typically used to detect parts on a production line
- The Gocator 2D Smart Camera cannot be accelerated from a PC; however, the camera incorporates a GoMax ORIN accelerator.

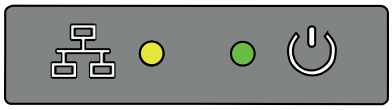
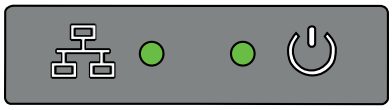
For complete specifications, see *Gocator 2D Cameras* on page 56. For cordset specifications, see *Gocator Power / IO Connector* on page 65.

Gocator 2D Smart Camera - Hardware Overview



Gocator 2D Smart Camera

	Item	Description
1	Camera light	Powers an RMX140 Smart Vision Lights ring light. Do not use this connector for powering anything else.
2	Power and I/O	Provides power to the smart camera and accepts input and output signals. Optionally, you can use this to power a camera light other than

	Item	Description
		the RMX140 ring light.
3	Network	Accepts power and connects to 1000 Mbit/s Ethernet network.
4	Indicators	Two LEDs indicate whether the camera is powered up (green indicator on right) and the network speed (left). Yellow indicator = 100 Mbps  Green indicator = 1 Gbps 
5	Lens mount	The lens mount threads. For information on mounting a lens, see *Mounting the Lens* on the next page.
6	Status light	While the camera is powering up, the status light strip is blue. After powering up, the status light turns off. You can configure the status light to show green or red based on various digital triggers, such as measurement pass/fail, system state (running or not), and so on. For information, see *Digital Output* on page 369.

Touching Hot Smart Cameras

⚠ CAUTION

Risk of burns

In operation, Gocator 2D Smart Camera systems can reach temperature levels that could cause burns.

- Wear protective gloves when you touch the smarter camera when it is hot.
- Ensure proper cooling of the smart camera.

Sensor Damage by ESD

NOTICE

Possible damage to the sensor

In Gocator 2D Smart Camera systems, sensors can be damaged by ESD. In the output image, this may become visible as bubbles or blobs, for example. For details, see

NOTICE

Possible damage to image sensors due to electrostatic discharge on page 31

Provide ESD protection measures in accordance with technical standards.

Follow the instructions in *ESD* on page 29.

Cables and Levels

NOTICE

Damage to Gocator 2D Smart Camera by wrong connections

If connected using wrong pinning or voltage levels, Gocator 2D Smart Camera can be damaged.

- Use cables by LMI Technologies.
- Observe the specifications in *Gocator Power / IO Connector* on page 65.

I/O cables maximum length

The maximum length for I/O cables must not exceed 30 m.

Default signal levels at Gocator 2D Smart Camera startup

Signals for opto-isolated Digital_Out_1, Digital_Out_2, and Trigger_Out (strobe) are low.

Mounting the Lens

⚠ CAUTION

Injury by a falling lens

Especially if you are going to operate Gocator 2D Smart Camera without a lens tube: Screw lenses into the lens mount until you feel resistance.

⚠ CAUTION

Risk of cuts by sharp edges of lens mounts

The threads of the lens mount can have sharp edges.

Be careful when mounting or unmounting lenses.

NOTICE

Damage to cameras by unsuitable lenses

The lens and the camera's sensor, filter or electronics can be damaged if a lens exceeding maximum protrusion is mounted to the camera.

- Use lenses with less than 13.6 mm maximum protrusion.

NOTICE

Damage to lenses or lens tubes

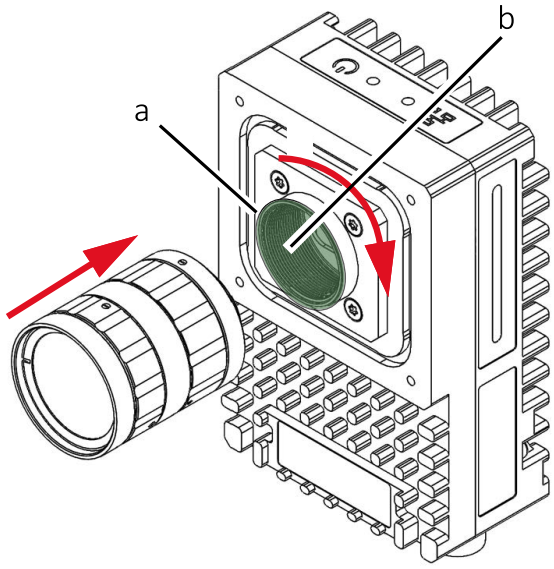
Lenses or the lens tube's front glass can be damaged if lenses are used that exceed the dimensions of the lens tube.

- Use only lenses within the dimensions of the lens tube.
- Observe that some lenses change in length when focus or zoom are used.

1. Remove the rear lens dust cap from the lens.
2. Put the lens face down on your work surface and place the dust cap on the rear of the lens.
3. Remove the plastic film (a) covering the camera's C mount with your fingers.

   Keep the plastic film in case you need to remove the lens in the future.
4. Remove the rear lens dust cap.
5. Fit the lens thread into the camera's C-mount thread (b) and screw the lens clockwise until tight.

   Be careful not to cross-thread the lens.



Connecting Power and I/O Cordset

For full pinout information of the cordset's connector, see *Gocator Power / IO Connector* on page 65.

NOTICE

Damage by reverse polarity

If powered with reverse polarity, Gocator 2D Smart Camera can be damaged.

Power the smart camera according to the specifications described in *Gocator Power / IO Connector* on page 65.

NOTICE

Damage by unsuitable signal levels

If unsuitable signal levels are used, Gocator 2D Smart Camera can be damaged.

Observe the specifications in *Gocator Power / IO Connector* on page 65.

NOTICE

Damage by unsuitable cables

If cable plugs do not fit tightly, Gocator 2D Smart Camera can be damaged.

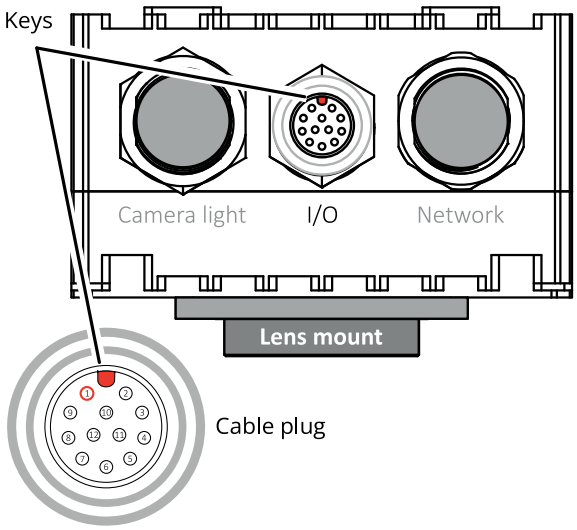
- For IP67 compliance, use only cables by LMI.
- Before using other cables, apply tests for IP67 compliance.

I/O cables maximum length

The maximum length for I/O cables must not exceed 30 m.

Connecting the Cordset to the Device

1. Remove the IP67 protective cap from the device's connector and store it for future use.
2. Rotate the cordset's connector until its key aligns with the key of the camera's connector, and push the cordset connector onto the camera connector.



3. Rotate the locking sleeve clockwise to tighten the connectors.

   Tighten to a maximum torque of 1.2 to 1.5 N m.

Connecting Open Wire Leads to the Power Supply

To power the camera, you must wire the 18-gauge red and black leads of the Power and I/O cordset to the power supply.

⚠ CAUTION

Shock hazard

Make sure the AC cord is not plugged in when wiring the power supply to the AC cord.

NOTICE

Damage to Device

Be careful not to reverse polarity when wiring the AC cord and the Power and I/O cordset to the power supply. This can damage the camera.

Sensor power and ground

Function	Pin	Lead	Description
DC_ 24V	7	Red (#18)	24 VDC (±10%)
GND_24V	8	Black (#18)	0 VDC

1. Wire the 18-gauge black lead to a V– terminal on the power supply.
2. Wire the 18-gauge red lead to a V+ terminal on the power supply.

Connecting Open Wires to Peripherals

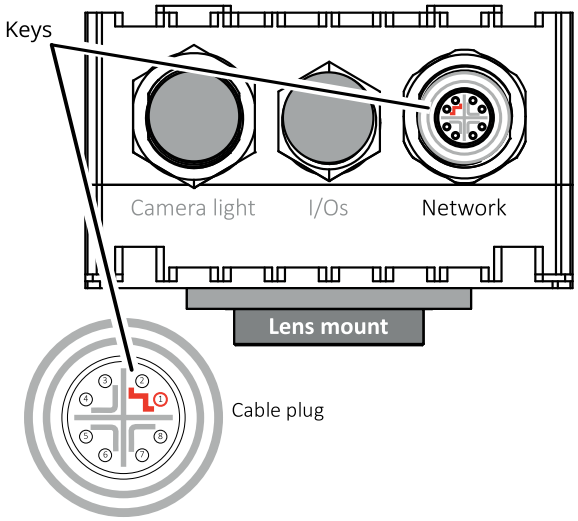
Depending on your application, you may need to wire other Power and IO cordset leads:

- Digital triggering of the camera: see *Wiring Trigger Input* on page 51.
- Software triggering of the camera using the REST API or a PLC (done via Ethernet over a switch or direct connection).
- Triggering external devices or an externally powered light source: see *Wiring Digital and Light Trigger Outputs* on page 52.

Connecting Network Cordset

Device Ethernet connector: Standard 8-pin M12 X-coded Kinsun 4003Q80000

1. Remove the IP67 protective cap from the device's LAN (POE) connector and store it for future use.
2. Align the keys of the device and cordset connectors, and push the cordset connector onto the device connector.



3. Rotate the locking sleeve clockwise to tighten the connectors.

   Tighten to a maximum torque of 1.2 to 1.5 N m.

4. Connect the RJ45 plug to a client computer or switch.

Preparing Power Supply

Gocator 2D Smart Camera is supplied with a DIN-rail mountable power supply from Mean Well. The supply outputs 24V @ 10A and supports the RMX140 ring light.

⚠ CAUTION

Shock hazard

Make sure the AC cord is not plugged in when wiring the power supply to the AC cord.

NOTICE

Damage to Device

Be careful not to reverse polarity when wiring the AC cord and the Power and I/O cordset to the power supply. This can damage the camera.

A three-prong AC cord is sold separately; you must wire the AC cord to the power supply. Use the color coding in the following table to identify Line, Neutral, and Ground for your region.

Conductor Color Coding

Conductors	International	North American	Japanese
	3 Conductors	3 Conductors	3 Conductors
Line	Brown	Black or Brown	Black
Neutral	Blue	White or Blue	White
Ground	Green/Yellow	Green or Green/Yellow	Green

1. Wire the AC cord's ground lead to the power supply's ground (⏚) terminal.
2. Wire the AC cord's neutral lead to the power supply's neutral (N) terminal.
3. Wire the AC cord's neutral lead to the power supply's line (L) terminal.

Adjusting Focus and Aperture

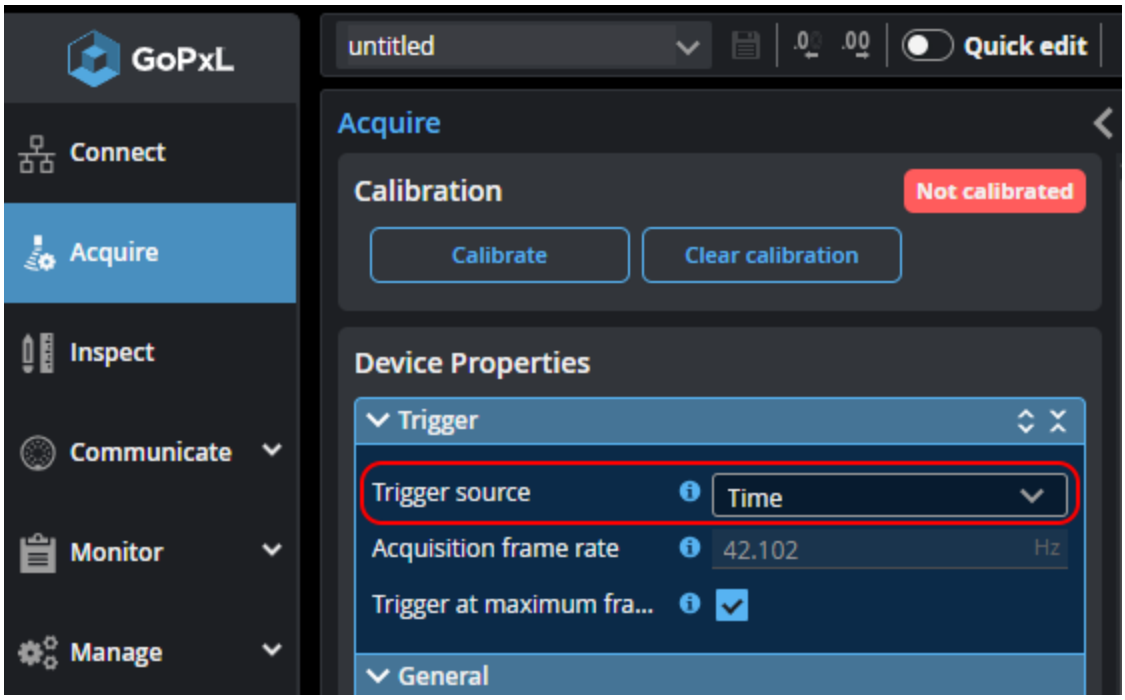
Before attaching the lens cover and ring light, you acquire a series of images so that you can adjust the lens focus and aperture.

1. Make sure the camera's network cordset is connected to a switch or directly to the client computer.
2. Apply power to the camera.
3. On the client PC, change the IP address of the network adapter to which the camera is connected as follows:

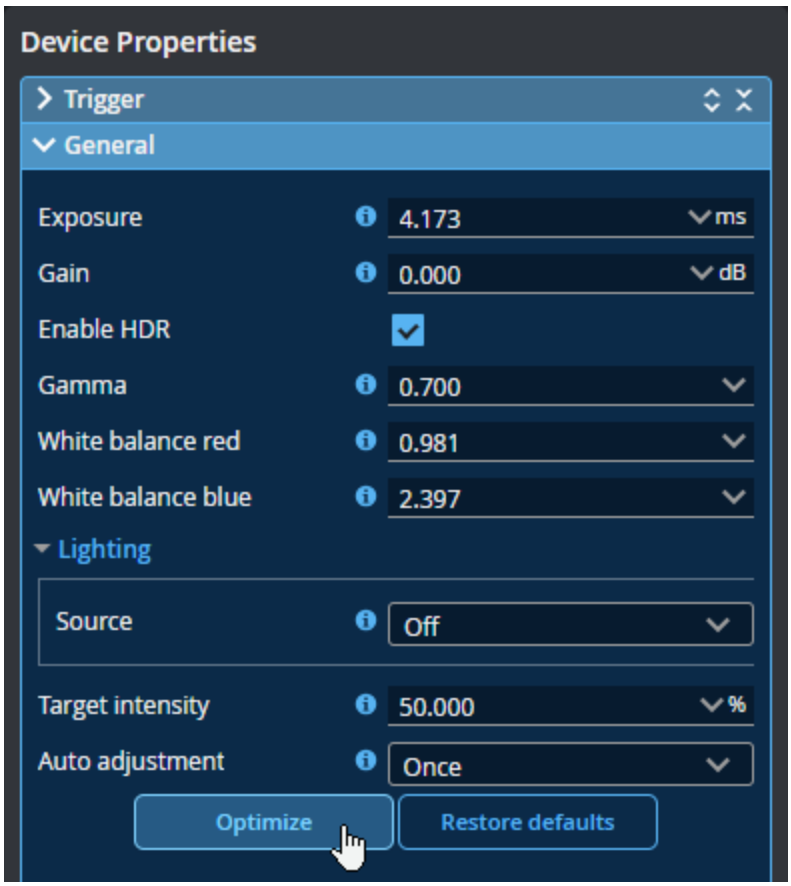
   IPv4 address: 192.168.x.x (except 192.168.1.10 or the IP address of any other devices on that network)

   IPv4 mask: 255.255.x.x
4. In web browser such as Google Chrome, go to 192.168.1.10.

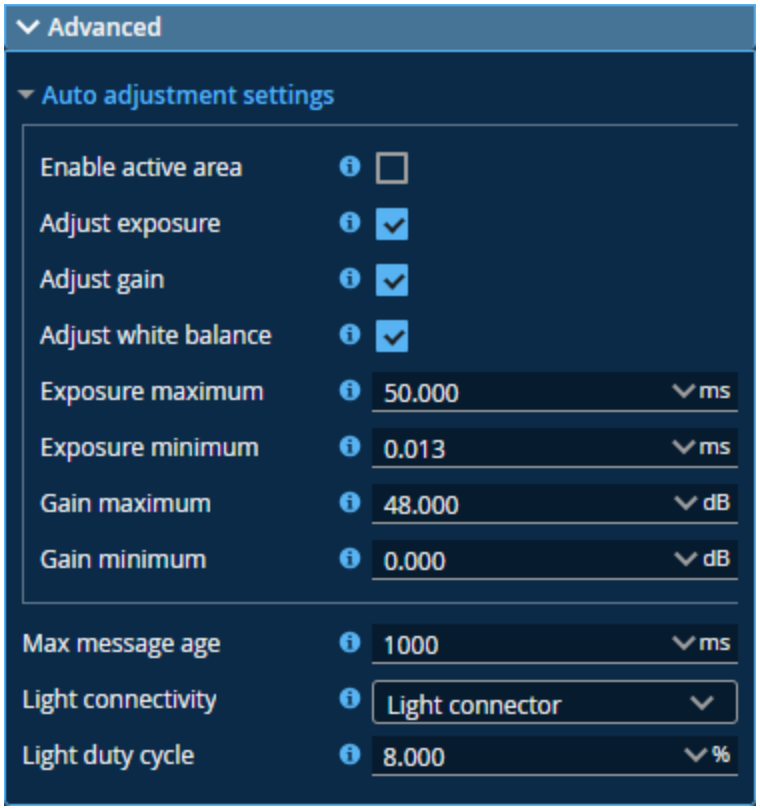
   The camera's default IP address is 192.168.1.10.
5. Go to the **Acquire** page and make sure **Trigger source** is set to Time.



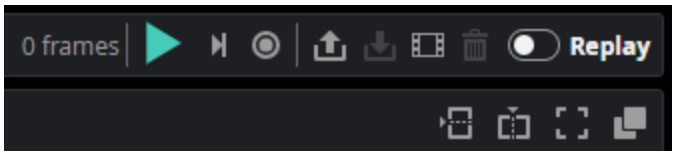
6. Remove the lens cap.
7. With a white sheet of paper under the camera, in the **General** section, click the **Optimize** button.



You can control what settings are adjusted when you optimize in **Auto adjustment settings** in the **Advanced** panel.



8. Place a target under the camera.
9. At the upper right of the user interface, click the Start button to start capturing images.



10. Adjust the aperture and focus on the lens until the image is sharp and depth of field (if appropriate) is adequate.

Mounting Lens Cover

⚠ CAUTION

Injury by falling lens covers

Tighten screws at 1.0 Nm maximum torque.

Follow the instructions below.

NOTICE

Damage to lenses or lens covers

Lenses or the lens cover's front glass can be damaged if lenses are used that exceed the dimensions of the lens tube.

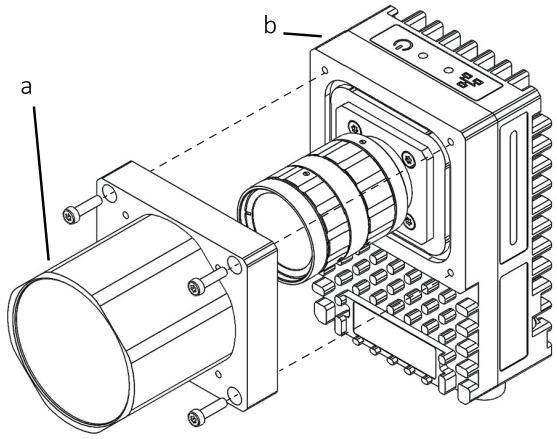
- Use only lenses within the dimensions of the lens cover.
- Observe that some lenses change in length when focus or zoom are used.

The optional lens cover protects the camera lens, and helps prevent accidental adjustment of focus and aperture.

1. Align the lens tube (a) and camera housing (b) mounting holes.
2. Using the provided M3-12↓ screws, attach the tube lens to the housing.

   Tighten the screws to a maximum torque of 1.0 Nm.

   For final assembly, LMI recommends using screw locking varnish to prevent the screws from coming loose.



Mounting Ring Light Adapter Plate

⚠ CAUTION

Injury by a falling camera ring lights

- Tighten screws at 1.0 Nm maximum torque.
- Follow the instructions below.

If you are using the RMX140 ring light, you must first install an adapter plate.

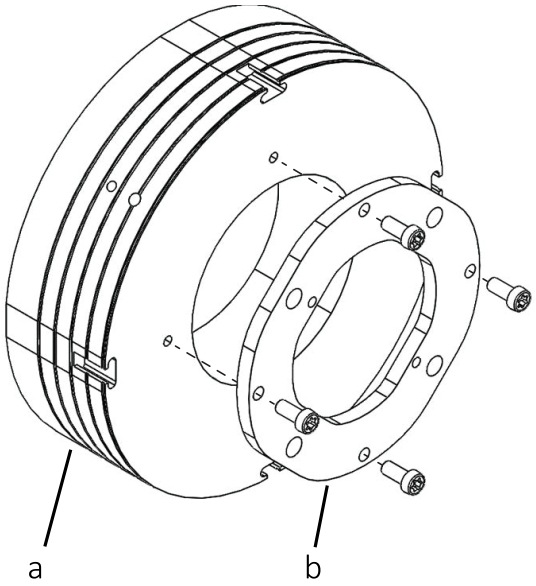
If you are using the lens cover, use the mounting plate with part number 301355. Follow these steps:

1. Align the ring light (a) and adapter (b) mounting holes.

2. Using the provided M4-10↓ screws, attach the adapter to the ring light.

   Tighten the screws to a maximum torque of 1.0 Nm.

   For final assembly, LMI recommends using screw locking varnish to prevent the screws from coming loose.

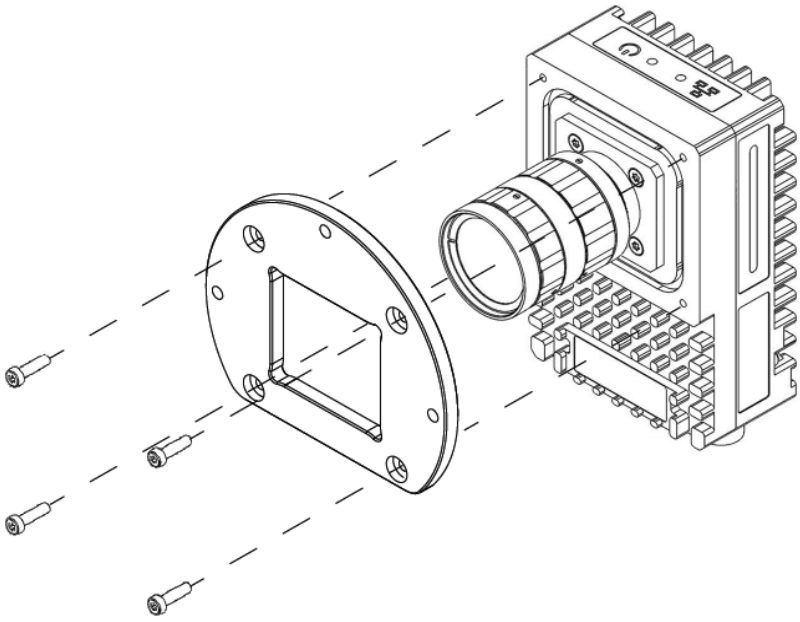


If you are *not* using the lens cover, use the mounting plate with part number 301419. Follow these steps:

1. Align the camera and adapter mounting holes.
2. Using the provided M3-12↓ screws, attach the adapter to the ring light.

   Tighten the screws to a maximum torque of 1.0 Nm.

   For final assembly, LMI recommends using screw locking varnish to prevent the screws from coming loose.



Mounting Ring Light to the Camera

⚠ CAUTION

Injury by a falling camera ring lights

- Tighten screws at 1.0 Nm maximum torque.
- Follow the instructions below.

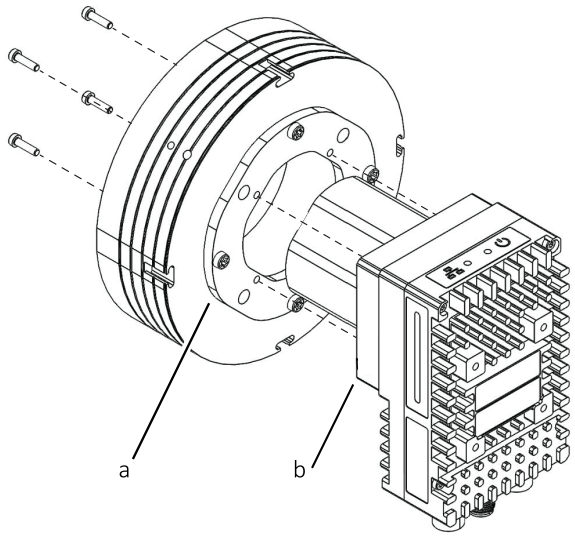
If You Are Using the Lens Cover...

After installing the adapter to the ring light, install the assembly to the camera.

1. Align the adapter plate (a) and lens tube (b) mounting holes.
2. Using the provided M3-12↓ screws, attach the ring light to the lens tube.

   Tighten the screws to a maximum torque of 1.0 Nm.

   For final assembly, LMI recommends using screw locking varnish to prevent the screws from coming loose.



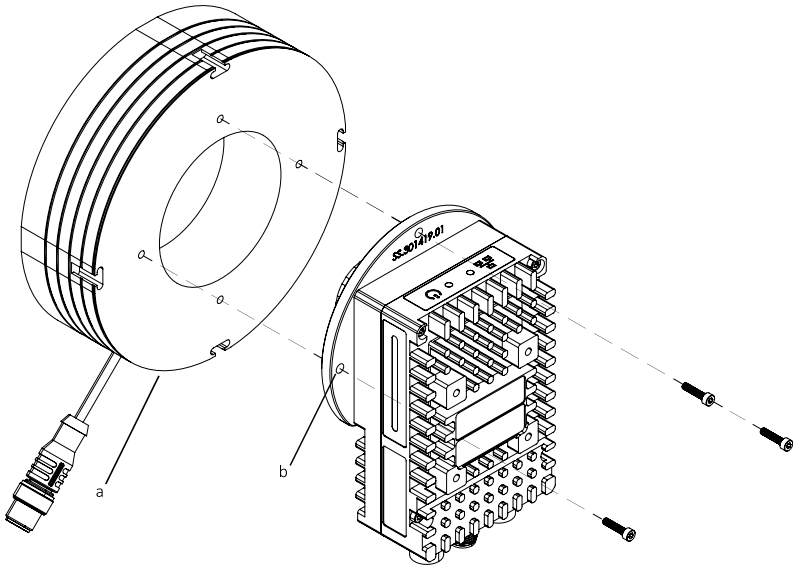
If You Are Not Using the Lens Cover...

Install the ring light to the adapter plate (301419) you mounted on the camera.

1. Rotate the ring light (a) until the holes on the back of the light align with the holes on the adapter plate (b).
2. Using the provided screws, attach the ring light to the adapter plate.

   Tighten the screws to a maximum torque of 1.0 Nm.

   For final assembly, LMI recommends using screw locking varnish to prevent the screws from coming loose.



Connecting Ring Light to the Camera

NOTICE

Damage to Gocator 2D Smart Camera electronics and connected peripherals

If the power output is used beyond the specified limits, Gocator 2D Smart Camera can be damaged.

- Keep the maximum output for VCC-Light below 5.9 A.
- Observe that above 700 mA, Strobe-PNP is limited to 50 ms and 10% duty cycle.
- Observe the specifications in *Gocator 2D Smart Camera Light Connector* on page 68.

Connections for external lighting

LMI offers compatible Y-cables at different lengths to separate electrical lines for lighting control by Gocator 2D Smart Camera from lighting power by external power supplies; contact LMI for more information.

NOTICE

Damage by unsuitable cables

If cable plugs do not fit tightly, the Gocator 2D Smart Cameracan be damaged.

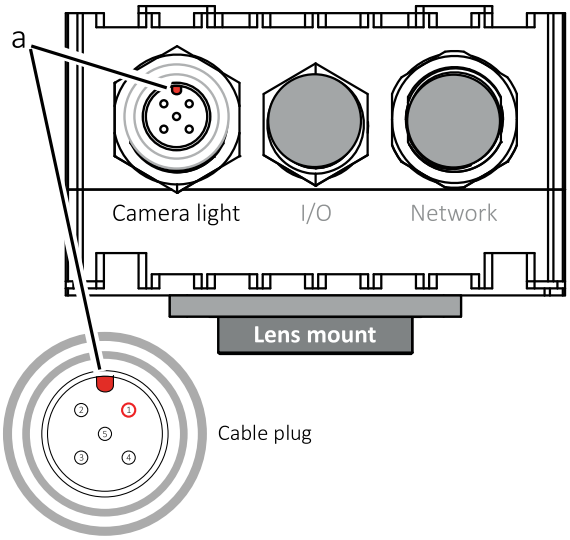
- For IP67 compliance, use only cables by LMI Technologies.
- Before using other cables, apply tests for IP67 compliance.

After mounting the ring light to the camera, you must connect the ring light to the camera using the 5-pin cordset.

1. Remove the IP67 protective cap from the camera's connector.

   Store the cap safely for future use.

2. Rotate the cordset's connector until its key aligns with the key of the camera's connector, and push the cordset connector onto the camera connector.

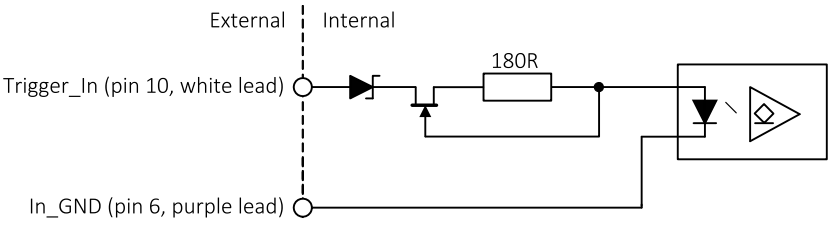


3. Rotate the locking sleeve clockwise to tighten the connectors.

   Tighten to a maximum torque of 1.2 to 1.5 N m.

Wiring Trigger Input

When triggering the camera externally with voltages up to 24 VDC, you can connect the input directly to the camera: an external resistor is not necessary.

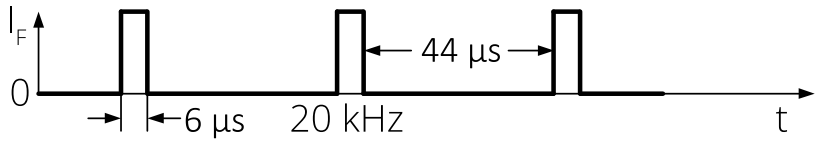


Sensor trigger input pinout

Function	Pin	Lead	Description
In_GND	6	Purple	Optically-isolated input ground. 0 VDC
Trigger_In	10	White	Optically-isolated trigger input. $V_{in\ (low)}$ = 0 to 1.0 VDC $V_{in\ (high)}$= 3 to 24 VDC

Function	Pin	Lead	Description
			Current (constant-current source) = 3 to 4 mA For > 24.0 VDC to 36 VDC, connect a 3.3 kΩ external resistor in series.

The minimum pulse width is as follows:



For full pinout information on this connector, see *Gocator Power / IO Connector* on page 65

Wiring Digital and Light Trigger Outputs

To trigger digital outputs (pin 4 or 11) or trigger an externally powered light source (pin 12), you must provide and connect an external power supply to drive these outputs, on pin 9.

NOTICE

Damage to Gocator 2D Smart Camera electronics and connected peripherals

To prevent damage to the camera, add a resistor in series to ensure the following conditions for the output pins:

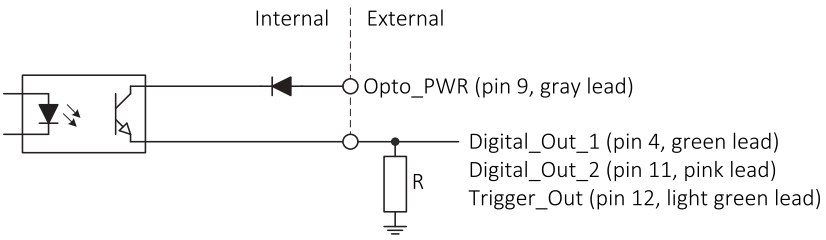
- Maximum output voltage is below 24 VDC
- Maximum output current is below 10 mA

For recommended resistor values, see *Opto_PWR voltage and external resistor values* below.

If digital outputs (pins 4 or 11) or light trigger output (pin 12) are connected to devices with a high impedance (< 5 mA draw), you must add an external resistor in series.

Opto_PWR voltage and external resistor values

Opto_PWR voltage	Resistor value	
5 VDC	1.0 kΩ	minimum 5 mA current draw required
12 VDC	2.4 kΩ	
24 VDC	4.7 kΩ	



Digital outputs and light trigger pinout

Function	Pin	Lead	Description
Digital_Out_1	4	Green	Optically-isolated output. Open emitter, maximum 10 mA.
Opto_PWR	9	Gray	Power for optically-isolated output. (User-supplied power for driving digital and trigger outputs.) Max. 30 VDC
Digital_Out_2	11	Pink	Optically-isolated output.
Trigger_Out (strobe)	12	Light Green	Optically-isolated output. Open emitter, maximum 10 mA.

For full pinout information on this connector, see *Gocator Power / IO Connector* on page 65

Mounting Camera

⚠ CAUTION

Injury by a falling Gocator 2D Smart Camera embedded camera systems

Ensure proper mounting of Gocator 2D Smart Camera, especially for dynamic applications.

- Mount the smart camera as described in the instructions.
- Always make sure the mounting threads are intact.
- Fasten screws with maximum torque, using the specified thread engagement.
- For less thread engagement, calculate torque values correspondingly.

You should mount the Gocator 2D Smart Camera at an appropriate distance from your target. Determine the appropriate distance by looking at the camera's specifications, that is, its minimum object distance and its different working distances, taking into account the size of your target.

The camera should be mounted using the provided mounting hardware (part number 301409) to a stable surface, using M4 screws. You can optionally use a 1/4" tripod mount with the mounting plate. For complete information on mounting points, see *Gocator 2D Cameras* on page 56.

NOTICE

Damage to mounting holes

The maximum torque of the M4 screws is 1.2 Nm for a minimum thread engagement of 3.0 mm between screws and mounting threads.

Be careful not to damage the internal threads by cross-threading or improper insertion of screws.

Specifications

The following sections describe the specifications of Gocator sensors and connectors, as well as Master hubs.

Gocator 2D Cameras **56**
- Gocator 1050-M / 1050-C / 1120-M / 1120-C 2D Smart Cameras 57
- Gocator 1050-M / 1050-C / 1120-M / 1120-C 2D Mounting Plate 59
- RMX140 Mounting Plates 60
- IP Class 61
- Maximum dimensions for lenses 61
- Shock and Vibration 62
  - IEC 60068-2-6: Sinusoidal vibration 63
  - IEC 60068-2-27: Non-repetitive shock 63
  - IEC 60068-2-27: Repetitive shock 64
  - IEC 60068-2-64: Random vibration 64

Sensor Connectors **65**
- Gocator Power / IO Connector 65
  - Opto-isolated Input Specifications 66
  - Opto-isolated Output Specifications 67
- Gocator Ethernet Connection 67
- Gocator 2D Smart Camera Light Connector 68

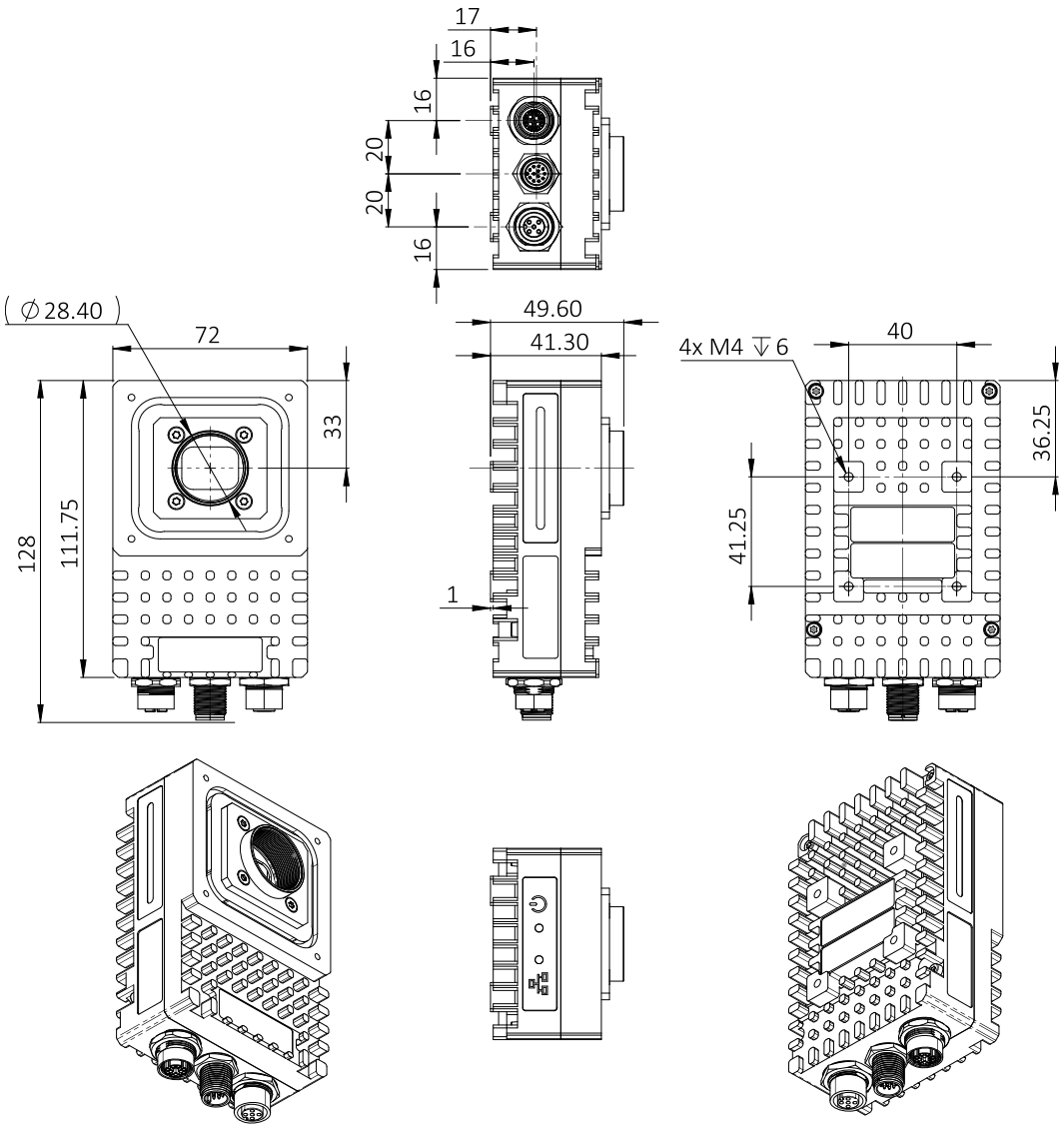
Gocator 2D Cameras

The Gocator 2D Smart Camera series includes the following models:

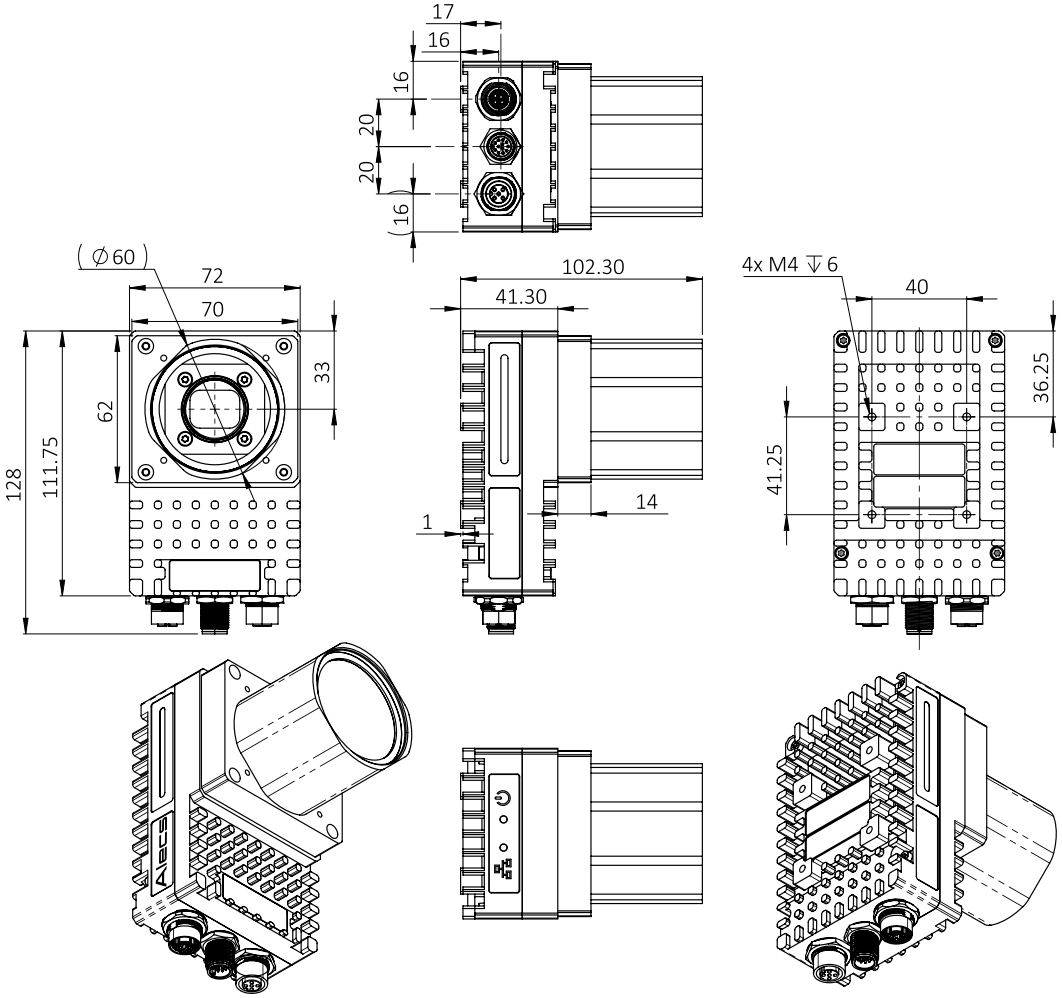
MODEL	1050-M	1050-C	1120-M	1120-C
Image Type	Mono	Color	Mono	Color
Resolution (pixels)	5 MP (2464 x 2064)		12 MP (4128 x 3008)	
Frames per Second (Burst, no processing)	84 fps		41 fps	
Sensor type	1/1.8" CMOS, global shutter		1/1.1" CMOS, global shutter	
Sensor Properties	8.9 mm diagonal, 2.74 x 2.74 µm		14.0 mm diagonal, 2.74 x 2.74 µm	
Lens mount	C mount			
Non-volatile memory	88 GB			
Application memory	15 GB			
Indicator LEDs	Power, network, system/measurement status			
Light control	External lights using IO connector			
Network	Gigabit Ethernet (10/100/1000 Mbps)			
Alignment and status IO	2 x input, 2 x output (opto-isolated)			
Trigger and Strobe	1 x Trigger, 1 x Strobe			
Industrial M12 Connectors	Power, IO, Ethernet			
Protection	IP67 with C-mount lens cover			
Power	+24 VDC (camera only, 48 W; with RMX140 light, 168 W)			
Dimensions (L x W x H)	49.3 × 72 × 111.75 mm			
Dimensions with lens tube (L x W x H)	102 × 72 × 111.75 mm		122.4 × 72 × 111.75 mm	
Camera Weight (no lens or cover)	650 g		650 g	
Lens Cover Weight	160 g		202 g	

Gocator 1050-M / 1050-C / 1120-M / 1120-C 2D Smart Cameras

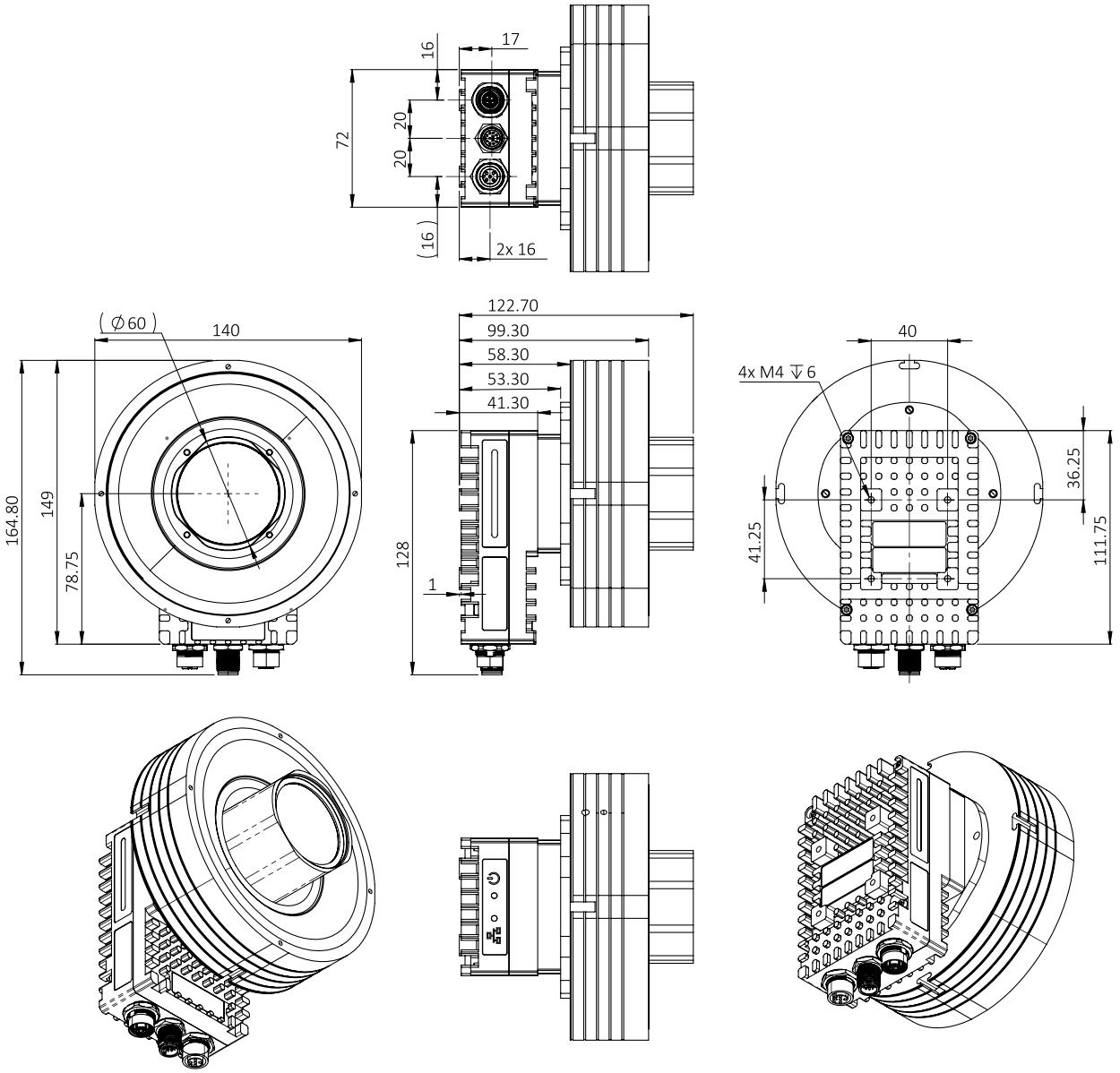
2D Smart Camera



2D Smart Camera with lens tube

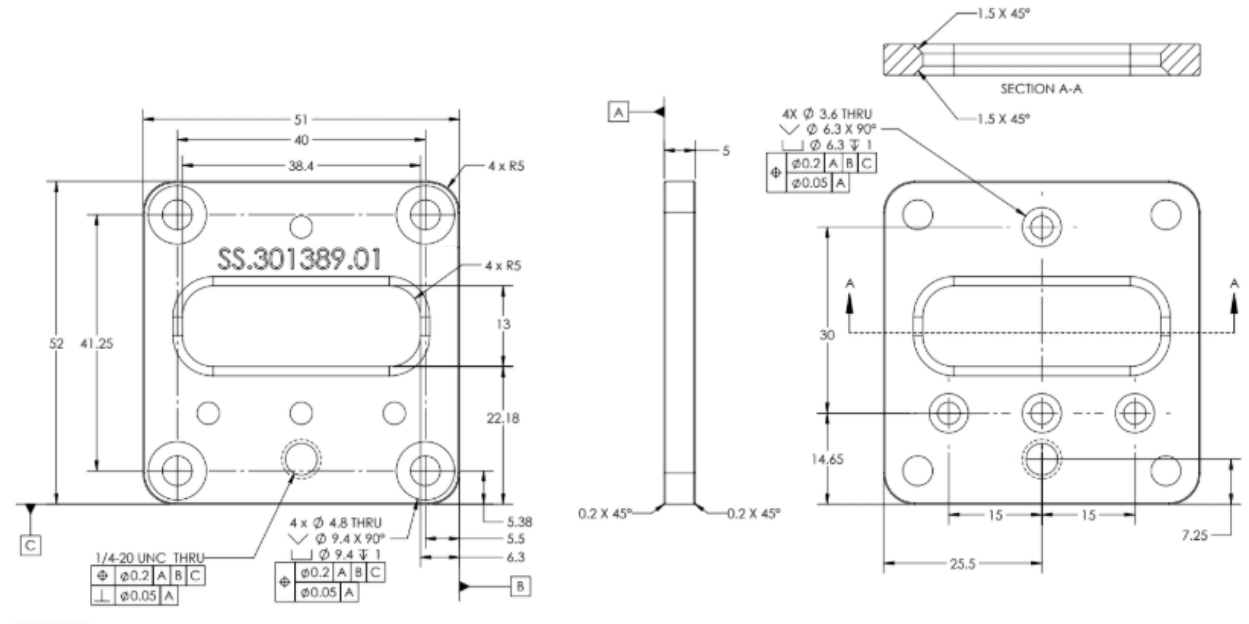


2D Smart Camera with RMX140 light ring



Gocator 1050-M / 1050-C / 1120-M / 1120-C 2D Mounting Plate

Use this plate to mount the camera to a solid surface.



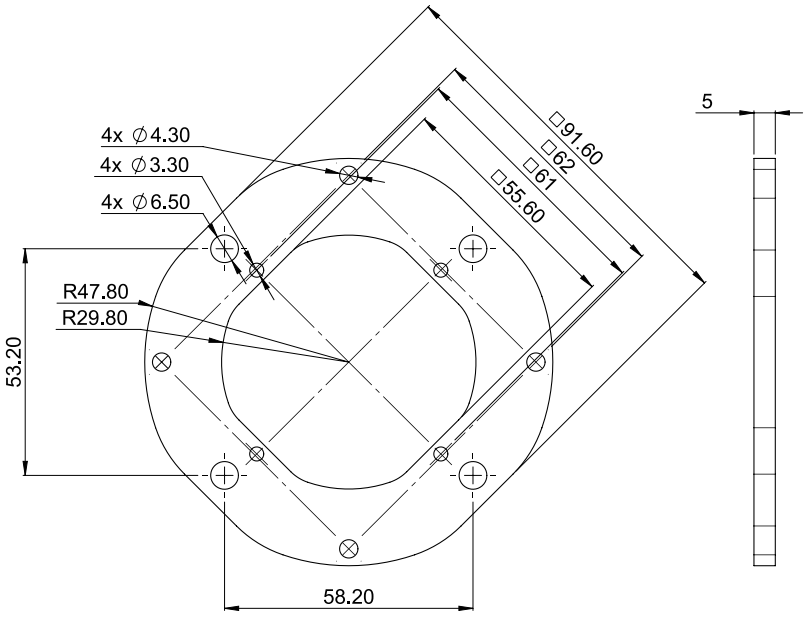
RMX140 Mounting Plates

If you are using the RMX140 ring light, use one of the following plates to mount the ring light.

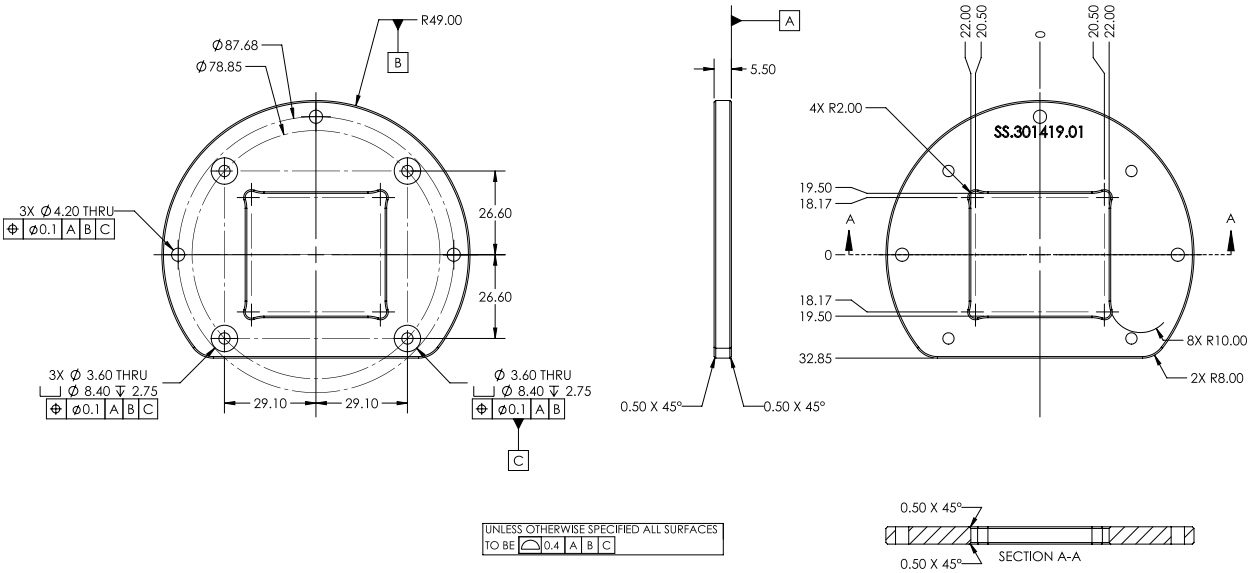
If you are using the provided lens cover, use mounting plate 301355.

If you are not using the lens cover, use mounting plate 301419.

Ring light mounting plate (301355), with lens cover



Ring light mounting plate (301419), without lens cover



IP Class

Ingress protection for Gocator 2D Smart Camera complies with IP67 class according to IEC 60529 when used as intended, with:

- Lens tube mounted properly, see *Mounting the Lens* on page 38.
- Connector caps or cables connected properly. Tests for the support of IP67 have been applied with cables by Allied Vision.

NOTICE

Damage to Gocator 2D Smart Camera

...if cable connections are not sealed against water or dust.

- Attach connector caps or connect cables with locking sleeves tightened at 1.2 to 1.5 Nm maximum torque.
- Before using third-party cables, apply tests for IP67 compliance.

Maximum dimensions for lenses

NOTICE

Damage to cameras by unsuitable lenses

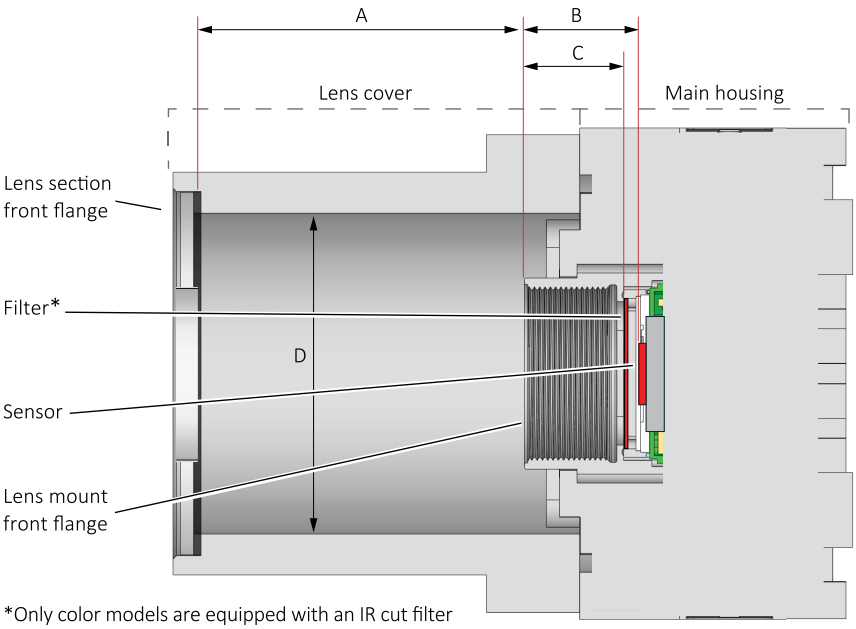
The lens and the camera's sensor, filter, or electronics can be damaged if a lens exceeding maximum protrusion is mounted to the camera.

- Use lenses with less than 13.6 mm maximum protrusion.
- See *Mounting the Lens* on page 38.

NOTICE

Damage to lenses or the lens tube

- Lenses or the lens tube's front glass can be damaged if lenses are used that exceed the dimensions of the lens tube.
- Use only lenses within the dimensions of the lens tube.
- Observe that some lenses change in length when focus or zoom are used.



Dimension	Description	Value
A[1]	Lens tube's front glass to lens mount front flange	49 mm or 70 mm
B[2]	Lens mount front flange to sensor (optical)	17.526 mm
C[2]	Lens mount front flange to filter	13.6 mm
D	Lens tube inside diameter	46 mm

1. Depending on the selected lens tube.
2. This value is known as maximum protrusion.

Shock and Vibration

Gocator 2D Smart Camera systems were tested according to the following standards:

- IEC 60068-2-6, sinusoidal vibration testing
- IEC 60068-2-27, shock testing
- IEC 60068-2-64, random vibration testing

The smart camera was inspected before and after the tests. All tests were passed successfully:

Condition Passed	Passed
Mechanics	The housing showed no deformations. The connections between the smart camera and connected peripherals did not come loose. The sensor position was within the specified tolerances of a new smart Gocator 2D Smart Camera.
2D Smart Camera behavior	Functionalities were not affected, no deviations occurred.
Image streaming	Images were streamed without errors

The conditions for included cameras and lenses were the same for all tests. Solid aluminum tubes were used to represent real lenses:

Parameter	Value
Lens dummy length	38 mm
Lens dummy mass	140 g
Center of gravity (CoG) (For camera and lens dummy assemblies, measured from the lens mount front flange)	20 mm

IEC 60068-2-6: Sinusoidal vibration

Frequency, acceleration, and displacement for IEC 60068-2-6 tests

Frequency	Acceleration	Displacement
10 Hz to 55 Hz	Not applicable	1.5 mm
55 Hz to 500 Hz	10 g (gravity)	Not applicable

Other parameters for IEC 60068-2-6 tests

Parameter	Value
Axis	x, y, z
Sweep rate	1 oct/min
Number of sweeps	10

IEC 60068-2-27: Non-repetitive shock

Parameters for IEC 60068-2-27, tests part 1

Parameter	Value
Axis	x, y, z

Parameter	Value
Acceleration	50 g (gravity of Earth)
Number of shocks per axis	10
Duration per axis	11 ms
Waveform	Half sine

IEC 60068-2-27: Repetitive shock

Parameters for IEC 60068-2-27, tests part 2

Parameter	Value
Axis	x, y, z
Acceleration	25 g (gravity of Earth)
Number of shocks per axis	10
Duration per axis	6 ms
Waveform	Half sine

IEC 60068-2-64: Random vibration

Frequency and acceleration for IEC 60068-2-64 tests

Frequency	Acceleration
5 Hz to 500 Hz	0.05 g^2/Hz (gravity of Earth^2 / Herz)

Other parameters for IEC 60068-2-64 tests

Parameter	Value
Axis	x, y, z
Acceleration RMS (Sigma)	4.9 g (gravity of Earth)
Acceleration RMS (peak)	14.8 g (gravity of Earth)

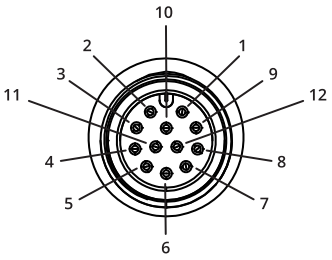
Sensor Connectors

The following sections provide the specifications of the connectors on Gocator sensors.

Gocator Power / IO Connector

The Power / IO connector is a 12-pin, M12 style connector that provides power (camera, optional RMX140 light, and digital output). All inputs and outputs are optically isolated.

> This connector is rated IP67 only when a cable is connected or when the protective cap is in place.



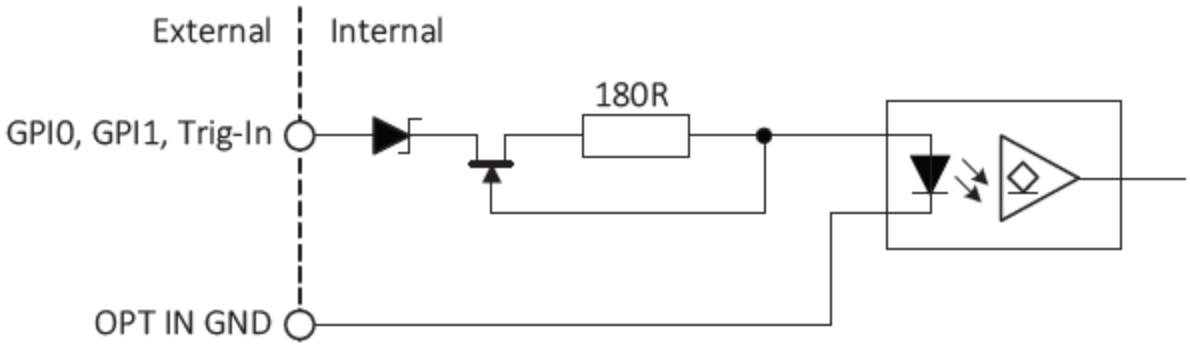
*View: Looking into the connector **on** the camera.*

Gocator Power and I/O connector pins

Function	Pin	Lead	Description
Not used	1	Brown	-
Not used	2	Orange	-
Not used	3	Yellow	-
Digital_Out_1	4	Green	Optically-isolated output. Open emitter, maximum 10 mA.
Not used	5	Blue	-
In_GND	6	Purple	Optically-isolated input ground. 0 VDC
DC_ 24V	7	Red (#18)	24 VDC (±10%)
GND_24V	8	Black (#18)	0 VDC
Opto_PWR	9	Gray	Power for optically-isolated output. (User-supplied power for driving digital and trigger outputs.) Max. 30 VDC
Trigger_In	10	White	Optically-isolated trigger input. $V_{in\ (low)}$ = 0 to 1.0 VDC $V_{in\ (high)}$= 3 to 24 VDC Current (constant-current source) = 3 to 4 mA For > 24.0 VDC to 36 VDC, connect a 3.3 kΩ external resistor in

Function	Pin	Lead	Description
			series.
Digital_Out_2	11	Pink	Optically-isolated output.
Trigger_Out (strobe)	12	Light Green	Optically-isolated output. Open emitter, maximum 10 mA.
Drain	shell		

Opto-isolated Input Specifications

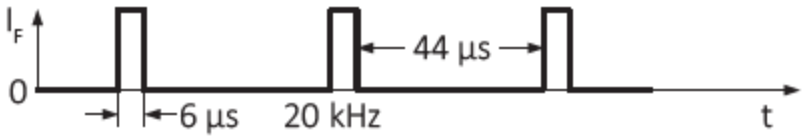


The input can be connected directly to the system for voltages up to 24 VDC. An external resistor is not necessary.

Observe the following levels:

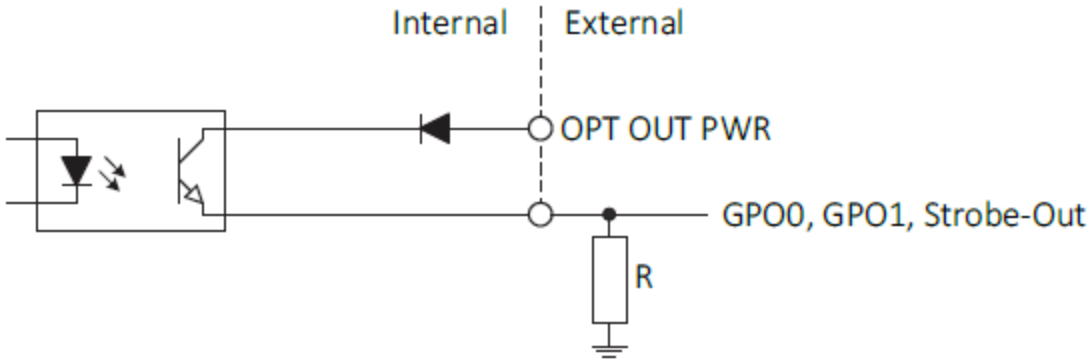
Parameter	Value
V_{in} (low)	0 to 1.0 VDC
V_{in} (high)	3 to 24 VDC
Current (constant-current source)	3 to 4 mA

The following shows the minimum pulse width:



The input signal was driven with 3.3 VDC and no external additional series resistor.

Opto-isolated Output Specifications



NOTICE

Damage to Gocator 2D Smart Camera by high output current or voltage

Exceeding the maximum output voltage or current can damage Gocator 2D Smart Camera.

Keep maximum output voltage below 24 VDC and output current below 10 mA.

Possible low output voltage

Output voltage may drop by approximately 2.5 VDC under full load.

Observe the following levels:

Opto-isolated out power	Resistor value (at ~5 mA minimum required current draw)[1]
5 VDC	1.0 kΩ
V_{in} (high)	2.4 kΩ
Current (constant-current source)	4.7 kΩ

1. A resistor is required when Digital_Out_1, Digital_Out_2, or Trigger_Out (strobe) are connected to devices with a high impedance < 5 mA draw.

Gocator Ethernet Connection

For the network connection, observe the standard for 8-pin M12 X-coded cables.

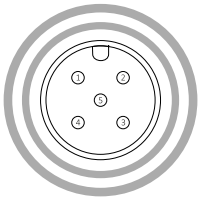
NOTICE

Attacks to your network

In open networks, use Gocator 2D Smart Camera with further protection. See *Cybersecurity* on page 28.

Gocator 2D Smart Camera Light Connector

The light connector is a 5-pin connector that provides power to a compatible light source and controls light strobing and intensity.



*View: Looking into the connector **on** the camera*

NOTICE

Damage to Gocator 2D Smart Camera electronics and connected peripherals

If the power output is used beyond the specified limits, Gocator 2D Smart Camera can be damaged.

- Keep the maximum output for VCC-Light below 5.9 A.
- Observe: Above 700 mA, Strobe-PNP is limited to 50 ms and 10% duty cycle.

Output signal polarity

The Strobe-PNP Out signal is inverted. Use LineInverter feature to trigger strobe lights.

The opto-coupled Strobe-Out signal on Pin 12 of the 12-Pin I/O connector is not inverted.

It can be used for triggering without the LineInverter feature.

Connections for external lighting

LMI offers compatible Y-cables at different lengths allowing to separate electrical lines for lighting control by Gocator 2D Smart Camera from lighting power by external power supplies.

Camera light connector pins

Function	Pin	Lead Color on Cordset	Level	Description
VCC-Light	1	Red	Max. PWR-IN	Power for connected camera light
Not used	2	N/A		-
GND-Light	3	Black	0 VDC	Supply ground for connected camera light
Strobe-PNP	4	N/A	Max. PWR-IN	Trigger signal for connected camera light
Intensity	5	N/A	0 to 10 VDC, max. 50 mA	Intensity signal for connected camera light

Required Ports

The table below lists the ports used by the following:

- sensors
- the Ethernet-based protocols
- the SDK and the REST API
- PC instances of GoPxL

Use this information to decide whether you or your network administrator needs to open network ports and to understand the traffic that a sensor system will produce over a network.

Ports used by GoPxL

Port	Data Packet Protocol	Description
80	TCP	GoPxL web port for GoPxL GUI. **Sensors and GoMax NX**: All devices use this port. **For GoPxL running on a PC**: Use port 8100; for more information, see below.
502	TCP	Used by the GoPxL Modbus protocol as the listener port.
2222	UDP	Used by the EtherNet/IP protocol for implicit messaging.
3192	TCP	FireSync physical sensor upgrade port.
3194	TCP	FireSync physical sensor health port.
3320	UDP	GoPxL discovery protocol server listener port
3500	UDP	GoPxL Remote Procedure Call port.
3600	TCP	GoPxL control port. **Physical sensors and GoMax NX**: All devices use this port. **GoPxL running on a PC**: By default, the first PC instance starts at this port number, and subsequent instances are offset (for example, 3600, 3620, and so on). You can set custom values in GoPxL Manager. For more information, see *Running GoPxL on a Windows PC* on page 403.
3601	TCP	Gocator Data Protocol (GDP) port. **Physical sensors and GoMax NX**: All devices use this port. **GoPxL running on a PC**: This port is offset from the control port by +1 (for example, 3601, 3621, and so on).
8080	TCP	Default accelerated sensor web server port number.
8100		GoPxL web port for GoPxL GUI for PC instances of GoPxL. (Sensors and GoMax NX units use port 80.) By default, the first PC instance starts at this port number, and subsequent instances are offset (for example, 8100, 8120, and so

Port	Data Packet Protocol	Description
		on). You can set custom values in GoPxL Manager. For more information, see *Running GoPxL on a Windows PC* on page 403.
8190	TCP	Used by GoPxL Ethernet for default ASCII control, data, and health port. Actual ports can be configured by the user.
44818	TCP and UDP	Used by the GoPxL EtherNet/IP protocol as a UDP and TCP listener port for explicit messaging.

Key Concepts

The following sections provide an overview of how devices acquire and produce data, detect and measure parts, and control devices such as PLCs. Some of these concepts are important for understanding how you should mount sensors and configure settings such as active area.

You can use the PC-based version of GoPxL to speed up the processing of data.

Measurement and Anchoring

After GoPxL scans a target and, optionally, further processes the data, the sensor is ready to take measurements on the scan data.

GoPxL provides dozens of measurement tools, each of which provides a set of individual measurements, giving you dozens of measurements ideal for a wide variety of applications to choose from. The configured measurements start returning pass/fail decisions, as well as the actual measured values, which are then sent over the enabled outputs to control devices such as PLCs, which can in turn control ejection or sorting mechanisms. (For more information on measurements and configuring measurements, see *Inspect - Measurement and Processing* on page 117.

The calculated offset between the two ensures that the anchored measurement will always be properly positioned on different parts. For more information, see *Measurement Anchoring* on page 138.

Complex Defect Detection

Many applications rely on more easily quantified differences part to part, such as a simple pass / fail of the height of a feature on a part. In other cases, flaws or defects are much more random and difficult to characterize so simply: a part or material can be damaged or malformed anywhere on its surface. For example, the following images show scan data of a good part, and two defective parts:

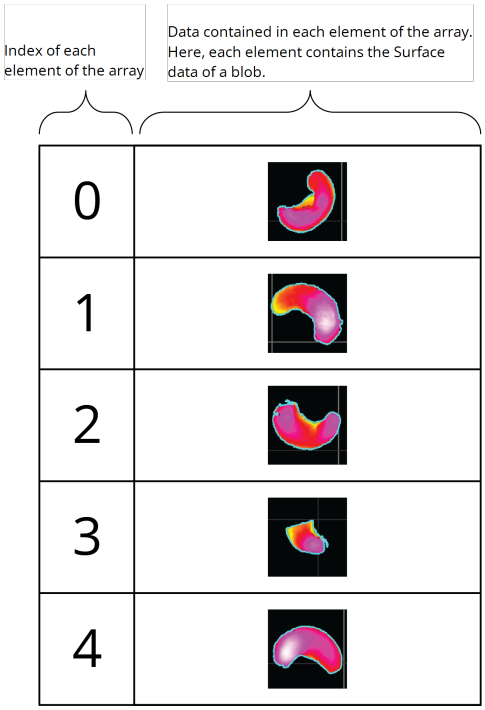
GoPxL Anomaly Detector, an AI-based toolset, lets you train a model on representative good ("OK") and bad ("NG") frames of acquired data. After, when the device scans parts in production, the tool uses the trained model in its inference engine to classify the new scan data as either OK or NG, resulting in flexible defect detection. It is also possible to refine the model with other representative frames later on, and even produce simulated variations and flaws / defects to further improve the model's accuracy. For more information, see .

Arrays, Batching, and Aggregation

GoPxL supports data structures called arrays to help simplify otherwise complicated applications.

An array is a group of data items bundled into a single structure, such as multiple profiles or surfaces, multiple measurements, or multiple features.

For example, the individual parts in a frame of scan data can be isolated and stored in an array. Each part's scan data occupies on element of the array. Note that arrays start at 0. That is, they use zero-based indexing.



Each item, or element, in the array can be processed by other tools in one of two ways:

- Batching: Tools in batching mode process each array element in an array input separately.
- Aggregating: Aggregating tools combine each element in an array output and process it as a single piece of data.

For more information, see *Arrays, Batching, and Aggregation* on page 126.

Output and Control

After the sensor has scanned and measured parts, the last step in the operation flow is to output the results.

One of the main functions of GoPxL is to produce pass/fail decisions, and then control something based on that decision. Typically, this involves rejecting a part through an eject gate, but it can also involve making decisions on good, but different, parts. Currently, GoPxL can communicate using the following over Ethernet:

- With PLCs (programmable logic controllers) using industry-standard protocols (EtherNet/IP, Modbus, PROFINET, and ASCII)
- With digital devices (such as ejector chutes or stack lights)
- With the GoPxL SDK and REST API, using the Gocator Communication Protocol

An important concept is digital output tracking. Production lines can place an ejection or sorting mechanism at different distances from where the sensor scans the target. For this reason, GoPxL lets you schedule a delayed decision over the digital interfaces. Because the conveyor system on a typical production line will use an encoder or have a known, constant speed, targets can effectively be “tracked” or "tagged." The sensor will know when a defective part has traveled far enough and trigger a PLC, for example, to activate an ejection/sorting mechanism at the correct moment.

For more information on configuring control, see *Communicate - PLC, Digital, GDP, HMI* on page 309.

For more information on the GoPxL SDK and REST API, see *GoPxL SDK and REST API* on page 449.

User Interface Overview

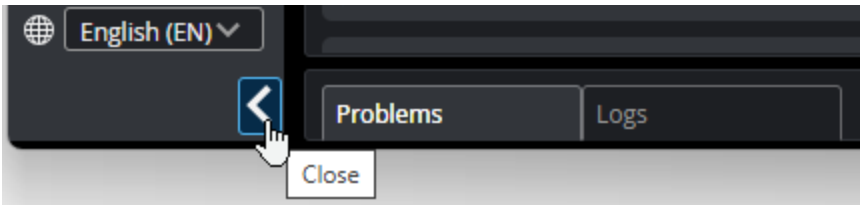
You configure sensors using GoPxL by connecting to an IP address with a web browser.

If you are running GoPxL on-sensor, in the browser, you connect to the IP address of the sensor. By default, the IP address of Gocator sensors is 192.168.1.10.

GoPxL provides categories in a vertical navigation bar on the left of the interface that group similar functions together in pages. Each page contains one or more panels that further categorize functionality. Generally speaking, the order of the categories and the pages reflects the workflow involved in configuring systems for an application for the first time, from the top down.

> Some UI screenshots in this manual may not depict the latest version of GoPxL. These screenshots will be updated soon.

	Element	Description
1	Navigation bar	The navigation bar (sometimes called the "nav" bar) along the far left of

	Element	Description
		the user interface displays categories (for example, from the top, **Connect**, **Acquire**, **Inspect**, and so on) that contain related settings. Some categories contain sub-pages (for example, the **Manage** category contains the **Settings** and **Jobs** pages, among others). Each page contains one or more panels. The category and page currently selected in the navigation bar are highlighted. By default, the navigation bar is open and shows the category, as well as the page names and icons. To close the navigation bar, click the Close button at the lower left of the interface.  For information on the categories and what you can do with the pages they contain, see *What you can do in the categories* on the next page.
2	Job-related functions	These controls let you work with jobs, for example, creating and saving jobs. For more information on jobs, see *Jobs* on page 396.
3	Decimal places	Use the decimal places buttons to control how values are displayed throughout the interface. The values displayed in the interface are rounded to the chosen number of data places, but internally all decimal places are used. Click .0 to decrease the number of decimal places, down to zero decimal places. Click .00 to increase precision, up to six decimal places.
4	Configuration area	This area contains the configurations of the selected page and panel. When configuring tools, an additional **Tools Configuration** panel lets you configure a tool's parameters.
5	Quick Edit	When working with a very large number of measurement tools (for example, a few dozen) or a very complex tool (built-in or user-created GDK tool), you can enable Quick Edit mode to make configuration faster, as the data viewer is not refreshed after each setting change. You should enable this setting if changing parameters feels sluggish. It is important to disable this mode after you have finished making parameter changes to save the changes.
6	Global controls	These controls let you start and stop acquisition, and also work with replay data. For information on starting and stopping acquisition, see *Starting, Stopping, and Recording* on page 95. For information on working with replay data, including adding and

	Element	Description
		removing frames of replay data, see *Playing Back Recorded Data* on page 97.
7	Data viewer	The data viewer (sometimes called the data visualizer) displays scan data, providing several tools in its toolbar to control how data is visualized (such as top view, front view, or perspective view). The data viewer is available on all pages. You can split it horizontally and vertically, and create pop-out windows. When GoPxL is in Replay mode, a blue border is displayed around the data viewer. For more information, see *Data Viewer* on page 85.
8	Displayed outputs pane	This area lets you show, hide, and pin outputs such as measurements. You can also control how outputs are displayed. For more information, see *Displaying and Pinning Outputs* on page 87.
9	Metrics area	This area provides CPU usage and frame rate, including the calculated maximum frame rate. For more information, see *Metrics Area* on page 79.
10	System messages	This area provides log messages, error alerts and warnings. For more information, see *System Messages* on the next page.
11	Interface language	Lets you change the language of the GoPxL interface. To apply the change, GoPxL must restart.

The following lists the categories available in GoPxL and where you can learn how to work with them.

What you can do in the categories

Category	Description
Connect	Lets you configure network options. For more information, see *Connect - Networking* on page 106.
Acquire	Lets you configure acquisition. Also lets you perform calibration. For more information, see *Acquire - Calibration and Image Acquisition* on page 107.
Inspect	Lets you add and configure measurement and processing tools. For more information on tools, see *Inspect - Measurement and Processing* on page 117.
Communicate	Lets you enable and configure the GoPxL Data Protocol and industrial protocols. Also lets you configure GoHMI. For more information, see *Communicate - PLC, Digital, GDP, HMI* on page 309.
Monitor	Lets you monitor measurements, as well as sensor health and performance. For more information, see *Monitor - Measurement Stats, Health & Performance* on page 387.
Manage	Lets you manage jobs, upgrade sensors, perform a factory restore, and so on. For more information, see *Manage - Sensor Management and Maintenance* on page 393.

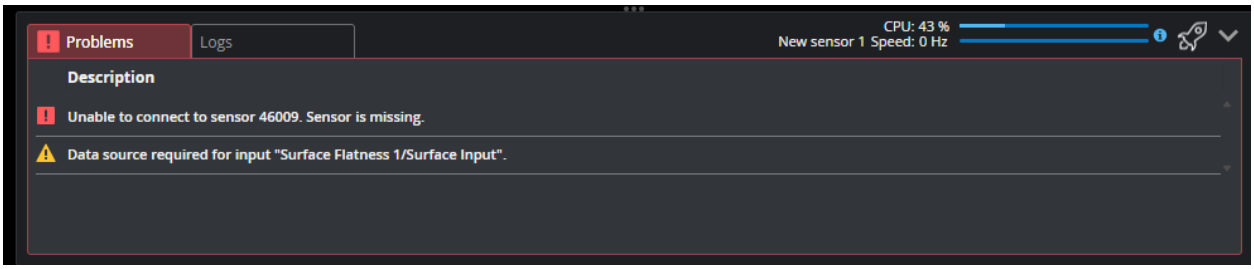
Status Bar

The status bar lets you do the following:

- Access system messages in the in the **System Messages** panel. For details, see *System Messages* below.
- See important system metrics. For details, see *Metrics Area* on the next page.

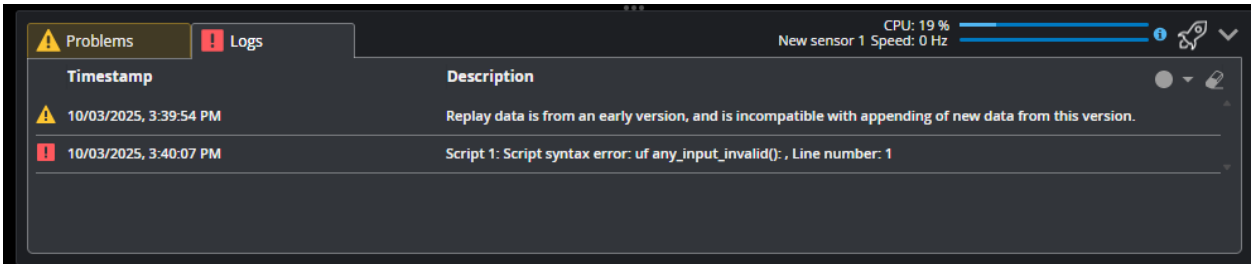
System Messages

GoPxL displays current problems and a system log containing errors, warnings, and general information, in two tabs at the bottom of the user interface. The **Problems** tab lists the *current* problem. Once you fix a problem, it is removed from the **Problems** tab.

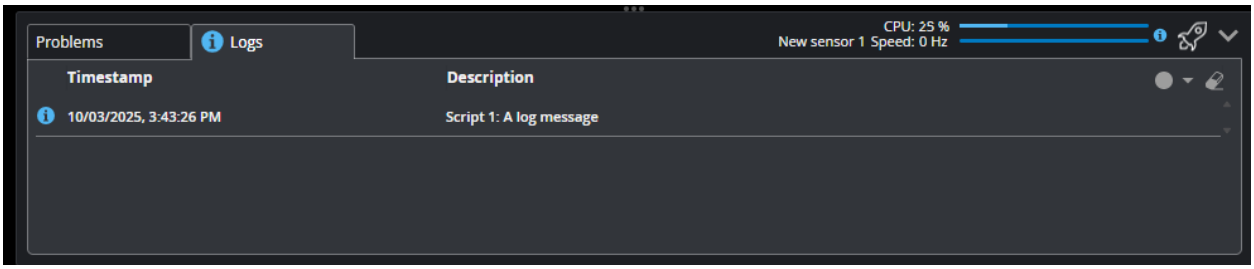


Problems tab showing a current problem with the system. The number of problems is indicated in parentheses.

The **Logs** tab displays a complete list of system-level messages. You can filter the log entry types using the filter drop-down to the right of the **Logs** tab, and clear the entries using the eraser button.



Problems that prevent the system from running (they are blocking problems) are indicated with a square red icon ( ). Problems that are less serious (they are potential problems) are indicated with a yellow triangle icon ( ). Informational messages are displayed with a blue circle icon ( ).



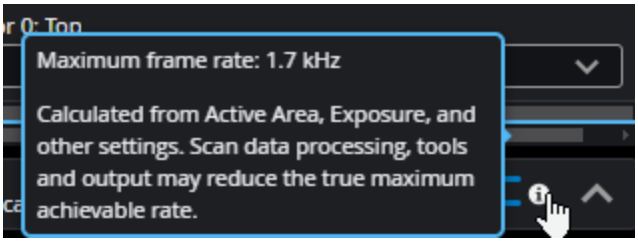
Metrics Area

The Metrics area on the right side of the status bar displays two important performance metrics: CPU load and sensor speed (current frame rate).

The **CPU** bar in the Metrics area displays how much of the CPU is being used.

The **Speed** bar displays the frame rate of the sensor.

You can see the maximum calculated frame rate by clicking the info icon to the right of the metrics area.



Keyboard and Mouse Shortcuts

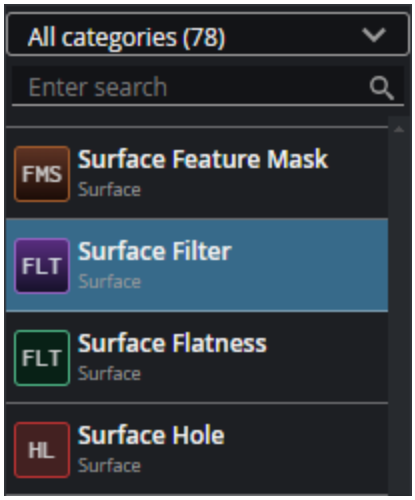
GoPxL supports various keyboard and mouse shortcuts.

Multiple tools are selected in the Tool Diagram panel (2); to the left is the Tool List (1).

Left-clicking a block in the Tool Diagram panel selects that tool. You can also select multiple tool blocks to perform operations on all of the selected tools. GoPxL supports the following shortcuts for selecting tools and performing operations on them:

Tool list shortcuts

Keyboard / mouse shortcut	Description
Drag and drop	Moves the selected blocks to a new position.
Shift + mouse click	Clicking a tool block and then pressing Shift while clicking a second block selects a range of blocks.
Ctrl + mouse click	Clicking a tool block while pressing Ctrl adds that tool to the selection; you can select multiple tools this way, including non-contiguously.
Ctrl + A	Selects all tool blocks (except for the sensor block).

Keyboard / mouse shortcut	Description
Delete (keyboard)	Removes the selected blocks.
Up or Down arrow	In the Tool Diagram, using these moves the selected tool blocks up or down in the tool list.
D key Duplicate (UI context menu)	Duplicates the currently selected tools.
Tab key	When a tool is selected in the Tool List, pressing Tab moves the focus to the next tool down. At the bottom of the list, the focus switches to the first tool in the Tool Diagram. Additional tabbing switches the focus to the next added tool. When focus reaches the bottom of the Tool Diagram, the tool selector drop-down in the **Tool Configuration** panel gets focus. 
Enter	**Tool List** Adds the tool to the Tool Diagram. Focus switches to the next tool in the Tool List. **Tool Diagram** Expands or collapses the tool that has focus. **Tool Configuration** Expands or collapses the selected sub-section, such as a measurement under **Outputs**. You must click on the sub-section first. After that, toggling works.
Double-click	**Tool List** Adds the selected tool to the Tool Diagram. **Tool Diagram** Expands or collapses the tool that has focus.

Note that the last selected tool gets focus (its configuration is displayed in the **Tool Configuration** panel and its outputs are displayed in the data viewer). Selected tool blocks are outlined in blue; the last selected tool is highlighted with a darker background.

In the data viewer, GoPxL supports the following shortcuts:

Data Viewer shortcuts

Name	Keyboard / mouse shortcut	Description
Force pan mode	Shift + Mouse drag	When the data viewer is in orbit or zoom mode, holding Shift and dragging scan data in the data viewer with the mouse switches the data viewer to pan mode. This shortcut is only active when the mouse pointer is over the data viewer window.
Ignore regions	Alt + Mouse drag	Causes the mouse pointer to ignore regions so that you can pan or orbit over a region. This shortcut is only active when the mouse pointer is over the data viewer window.
Temporary pan	Middle mouse button	When the data viewer is in Zoom mode, holding the middle mouse button while moving the mouse temporarily switches to Pan mode.
Center camera on clicked point	Double click surface in Perspective view	Double-clicking on a point in Perspective view centers that point in the data viewer. (Click above the data viewer to choose Perspective view.)
Z scale	Shift + mouse wheel	Modifies the Z scale value. For more information on Z scaling, see *Data Viewer* on page 85.
Toggle demo mode	Alt + Shift + D	Toggles demo mode (automatic camera orbiting) on or off when the data viewer is in Perspective mode.
Increase demo speed	Alt + Shift + "+" or Alt + Shift + "="	Increases the demo speed.
Decrease demo speed	Alt + Shift + "-" or Alt + Shift + "_"	Decreases the demo speed.
Undo region modification	Ctrl + Z	Undoes changes to the last modified region (size or location).
Redo region modification	Ctrl + Shift + Z Ctrl + Y	Redoes a change to the last modified region (size or location).

File Formats

The following formats are used with GoPxL, or can be converted using the replay utility tool.

File formats

Format	GoPxL extension	Gocator firmware 6.x and earlier	Content	Description
Job	.gpjob	.job	• Scan settings • Tool settings • Control and output settings	Quickly change a set of settings by switching from one job to another. For more information, see *Jobs* on page 396.
Recording	.gprec	.rec	• Current job settings[1] • Recorded scan data Note that sensor settings are not necessarily visible after loading a recording.	Quickly save and load recorded data for development and diagnostic purposes. For more information on recording and on loading recorded data, see *Recording and Working with Scan Data* on page 94. You can use the Replay Utils tool, provided in the utilities package, to convert .gprec files to and from various formats. For more information, see *Replay Utility Tool* on page 476.
Backup	.gpbak	.bak	• All jobs • Current job settings[1] • Sensor system layout and transformations • Files generated by tools such as Surface Pattern Matching • Global settings	Allows restoring a sensor to a previous state in case undesired changes were made. You back up and restore sensors on the **Maintenance** page. For more information, see *Backup and Restore* on page 400.

Format	GoPxL extension	Gocator firmware 6.x and earlier	Content	Description
Support	.gpsup	.gs	• Everything the backup file contains • Additional diagnostic information	For supporting development and diagnosing issues. You download and upload support files on the **Support** page. For more information, see *Support* on page 401.

1. See Job format details above. This also includes unsaved changes.

Format	GoPxL extension	Gocator firmware 6.x and earlier	Content	Description
CSV	n/a	.csv	Scan data in CSV format	First download replay data (.gprec) in GoPxL, and then use the Replay Utils tool to convert to the CSV format, for use with third-party tools.
.pcd / .sur / .srf / .pro	n/a	n/a	Scan data in various formats	First download replay data in GoPxL, and then use the Replay Utils tool to convert to the desired format, for use with third-party tools.

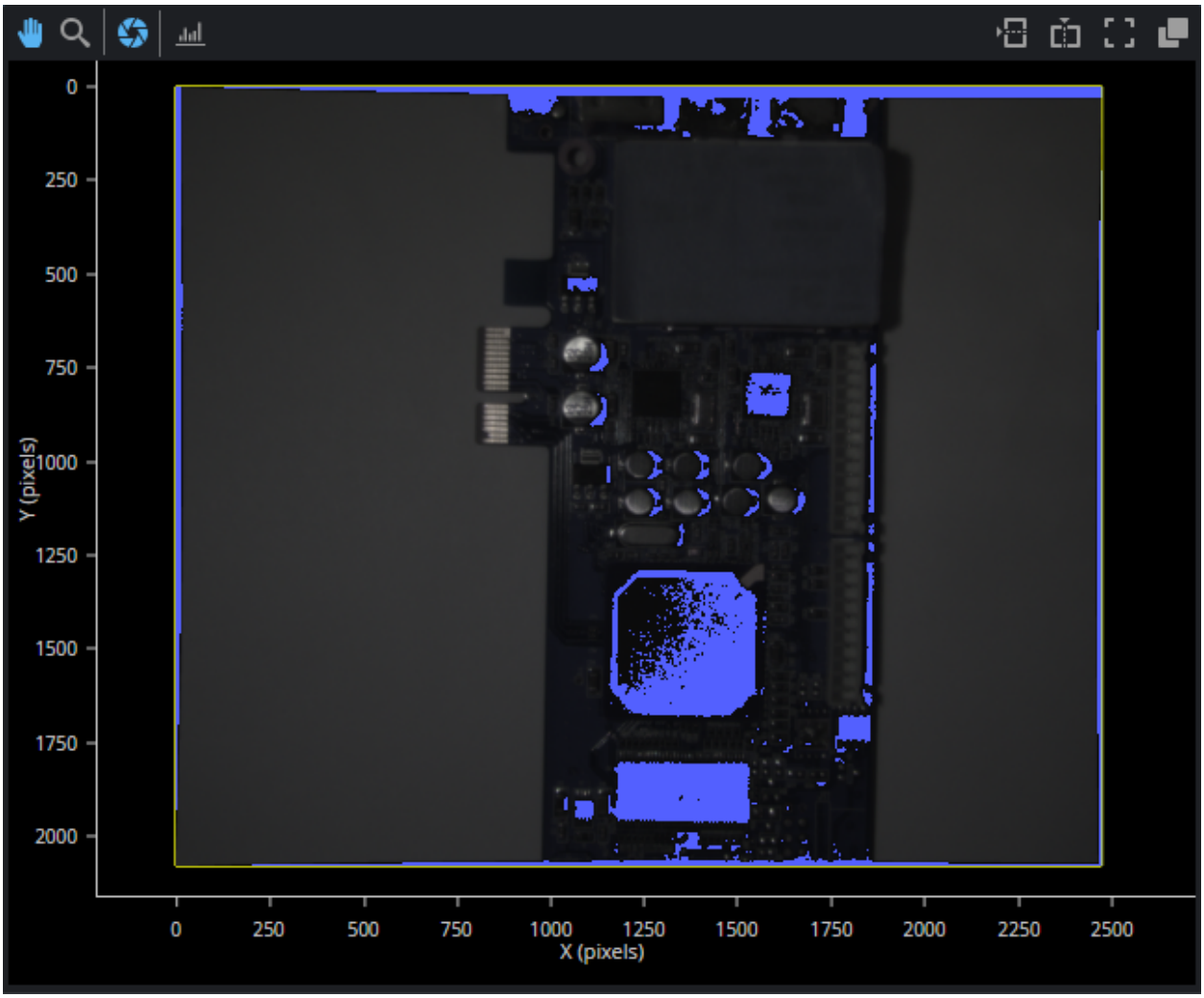
Data Viewer

You use the data viewer to observe Image data coming from the camera. You can also use it to configure measurement tools (see *Inspect - Measurement and Processing* on page 117) and a sensor's active area (see *Active Area* on page 113). GoPxL supports multiple data viewers (via both splitting and pop-out buttons).

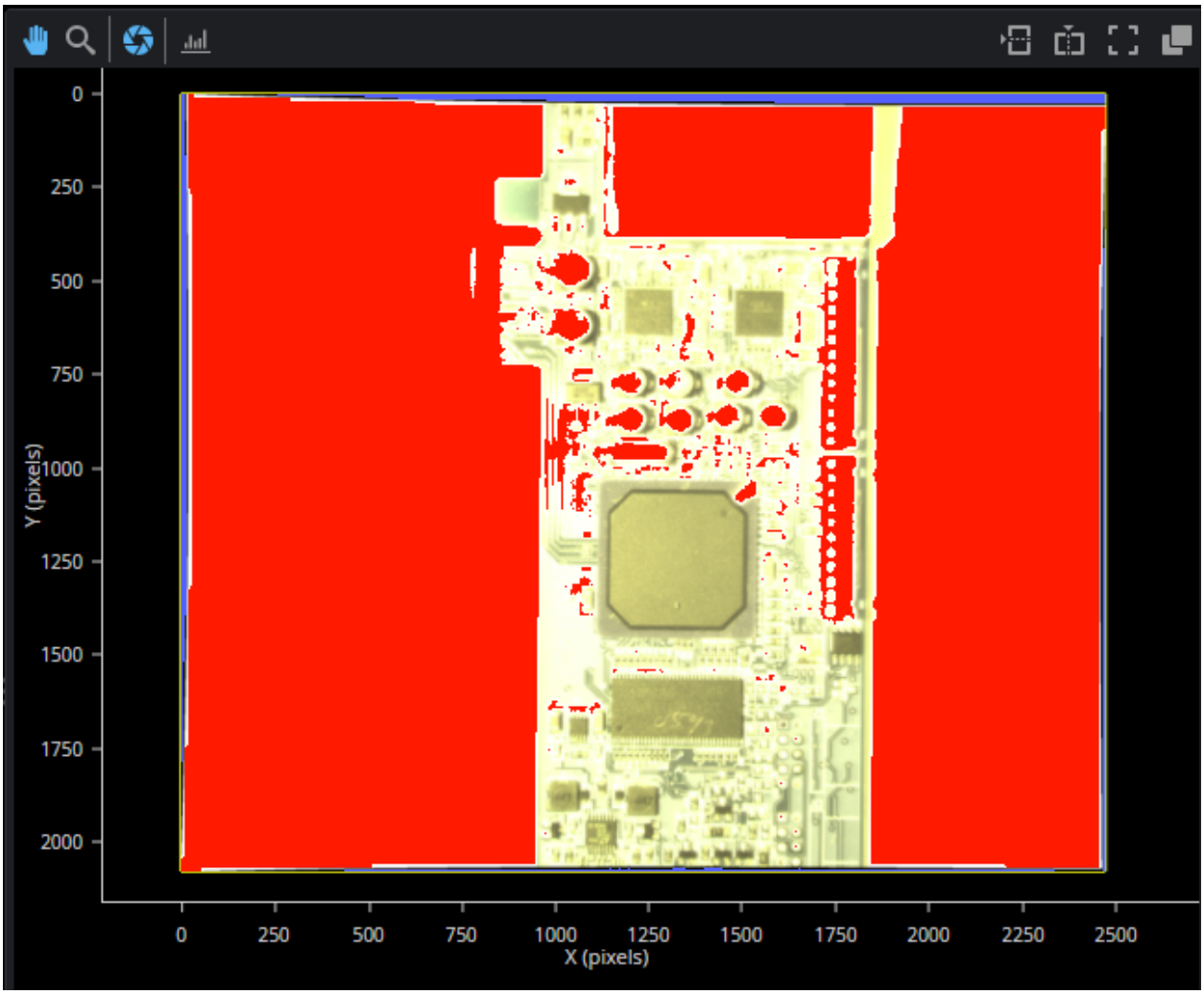
Exposure Mode

Gocator 2D Smart Cameras let you evaluate whether image pixels are underexposed or overexposed. To display exposure information, click the Exposure button ( ) at the upper left of the data viewer. You should limit underexposed or overexposed pixels, especially regions of interest, to acquire the best quality images. To do this, adjust your lighting or the device's exposure. For more information, see *General - Exposure, HDR, Gamma, Lighting* on page 112.

Underexposed pixels are displayed in blue.



Overexposed pixels are displayed in red.

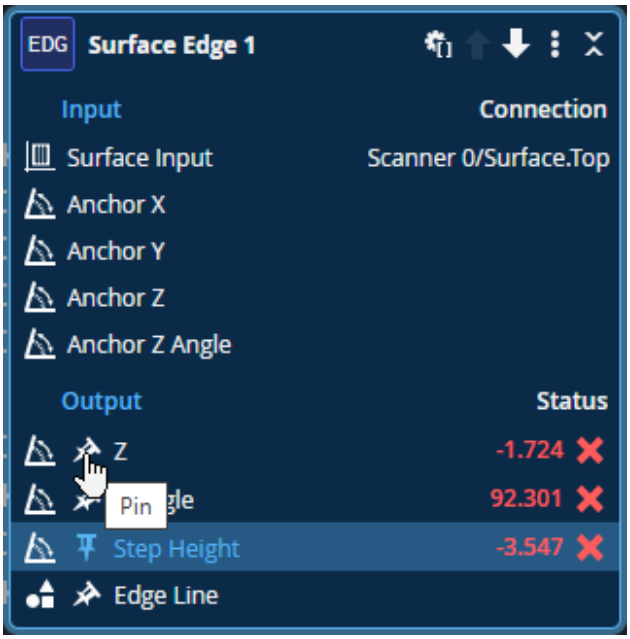


Displaying and Pinning Outputs

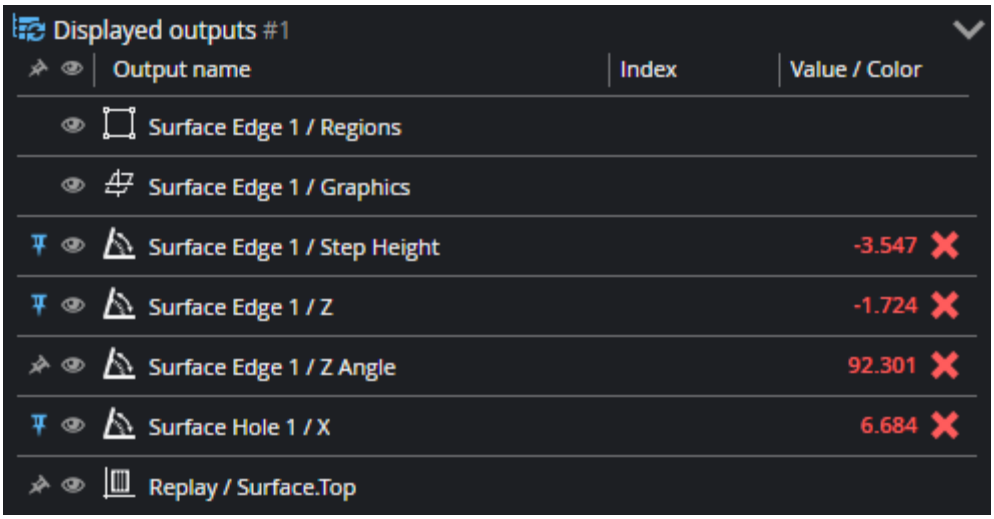
You can interact with the inputs, outputs, and other elements listed in the **Displayed Outputs** panel to control what you see in the data viewer window, by pinning or hiding outputs. This makes it easier to configure and monitor your application. When you select a block (a tool or the sensor, device, or source at the top) in the Tool Diagram panel, the **Displayed outputs** panel lists items related to the currently selected tool. By default, the panel lists *all* of the selected tool's inputs, the currently selected output of that tool, any pinned outputs, and tool-related graphics. In addition to selectively pinning or hiding outputs, you can right-click on an output in the panel to access "show / hide all" options.

You can “pin” items in the **Displayed Outputs** panel so that it they are always displayed, even if you select a different tool, or another input or output in a tool block. Pinning items is useful for example if you want to monitor multiple, independent measurements while GoPxL is running in production. Pinning is also useful when setting up tools: you can change the parameters of a tool (such as a filter) earlier in a tool chain and immediately see the impact that change has on another tool (whose output is pinned) later in the chain. This minimizes toggling and clicking between tools and measurements.

You pin items in the **Displayed Outputs** panel, in the tool's block in the Tool Diagram, or in the tool's Outputs section in the **Tool Configuration** panel. Pinned outputs are identified in GoPxL by a vertical blue "thumbtack" icon. Unpinned outputs are identified by a grey thumbtack icon. (See below.)

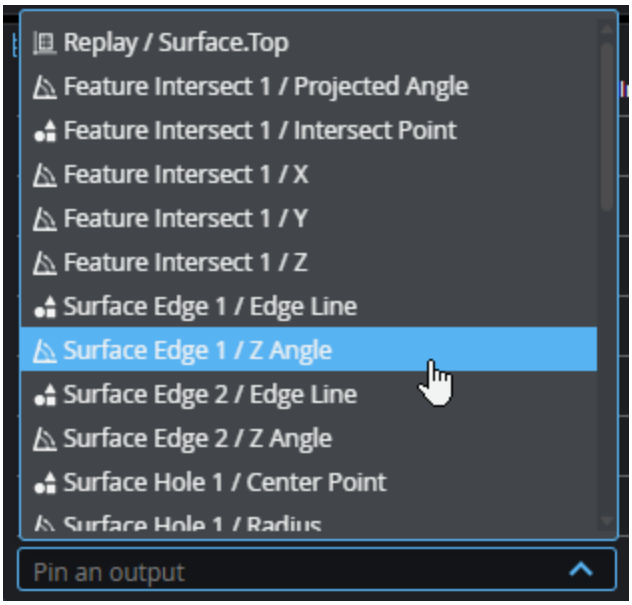


Pinning an output in the Tool Diagram.



Pinning an output in the Displayed Outputs panel.

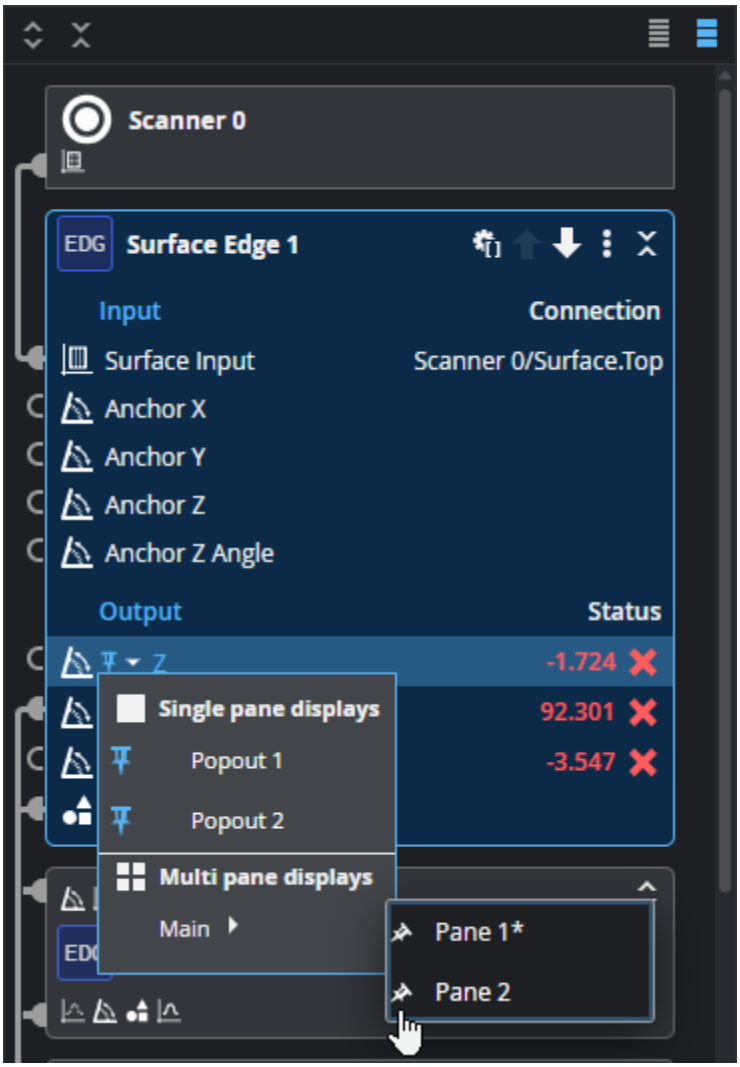
You can also choose an item from the "Pin an output" menu at the bottom of the panel.



If you have created additional data viewers, either by splitting a data viewer in the main browser window, or by popping out a data viewer in a new window, you can choose which data viewer to pin an output to in the Tool Diagram panel. (For information on creating and using new data viewers, see *Using Multiple Data Viewers* on page 91.)

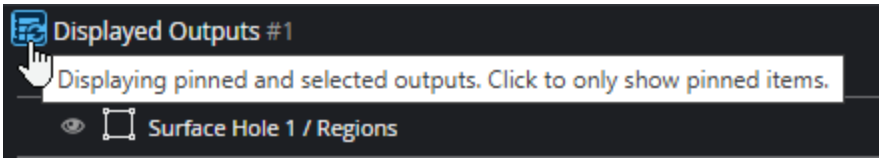
"Single pane displays" are data viewers you've created using the "Pop out" button ( ), which display in a new browser window.

"Multi pane displays" are data viewers you've created using one of the "split" buttons (or). In the popped-out data viewer, The pane identified by an asterisk is the original data viewer, the one from which you created others.



In both cases, the input of the currently selected tool and the currently selected tool output are added to the new data viewer and are pinned. You can add more outputs from the "Pin an output" menu.

You can control whether a data viewer displays both pinned and selected inputs / outputs, or instead *only* pinned outputs. Displaying only pinned outputs lets you focus on those inputs and outputs in the data viewer while configuring or monitoring your application. You toggle between these modes by clicking the Synchronize pane button ( ) at the top of the **Displayed Outputs** panel.



By default, this is enabled in the main data viewer, and disabled in any pop-out or split data viewers you create. Note that if there are no pinned outputs in any of the tools in your job, and you disable

pane synchronization, the **Displayed Outputs** panel will be empty and nothing will be displayed in the data viewer.

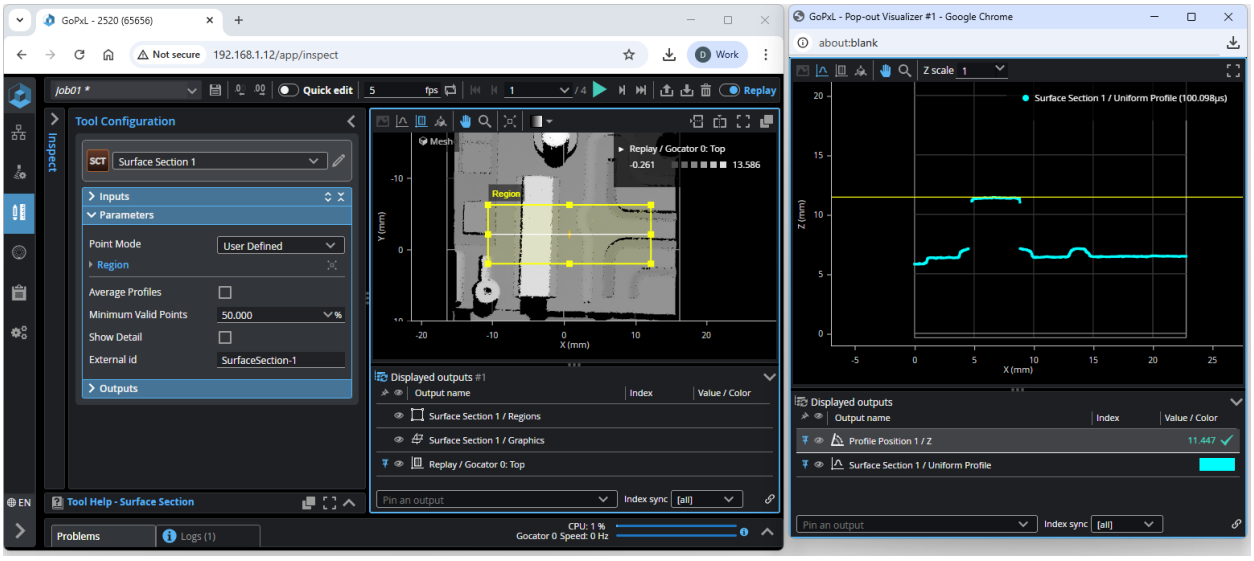
When you are working with arrayed outputs, you can select an array element for display across all arrayed outputs listed in the **Displayed Outputs** panel.

If you want to focus on a single element, you can choose or type that element's index in the **Index sync** drop-down.

Note that the default is to display all elements.

Using Multiple Data Viewers

You can more easily monitor or set up complex applications by popping out multiple data viewers outside of the main browser window, or by splitting a data viewer into two data viewers in the same browser window. You can configure each data viewer to different views, different display modes, and different sets of pinned outputs. This gives you complete flexibility.



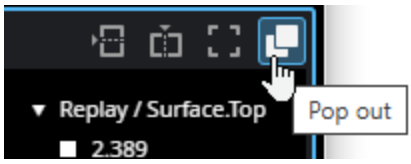
Main view in original browser window showing surface data and a defined section, and a second window showing a Profile tool running on the section.

External data viewer windows provide the same functions as the main data viewer via the toolbar above the viewer (except for the ability to open a new window). External windows also include a Displayed Outputs panel at the bottom and support the pinning of outputs; pinning in external windows is independent from the main view data viewer and other external windows. For more information on pinning outputs, see *Displaying and Pinning Outputs* on page 87.

To create a new data viewer window

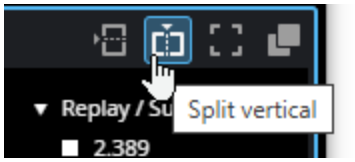
Do one of the following:

- In the toolbar of the main view data viewer, click the Pop out button ( ).



A new browser window opens containing a separate data viewer.

- In the toolbar of any data viewer, click one of the "split" buttons (or).



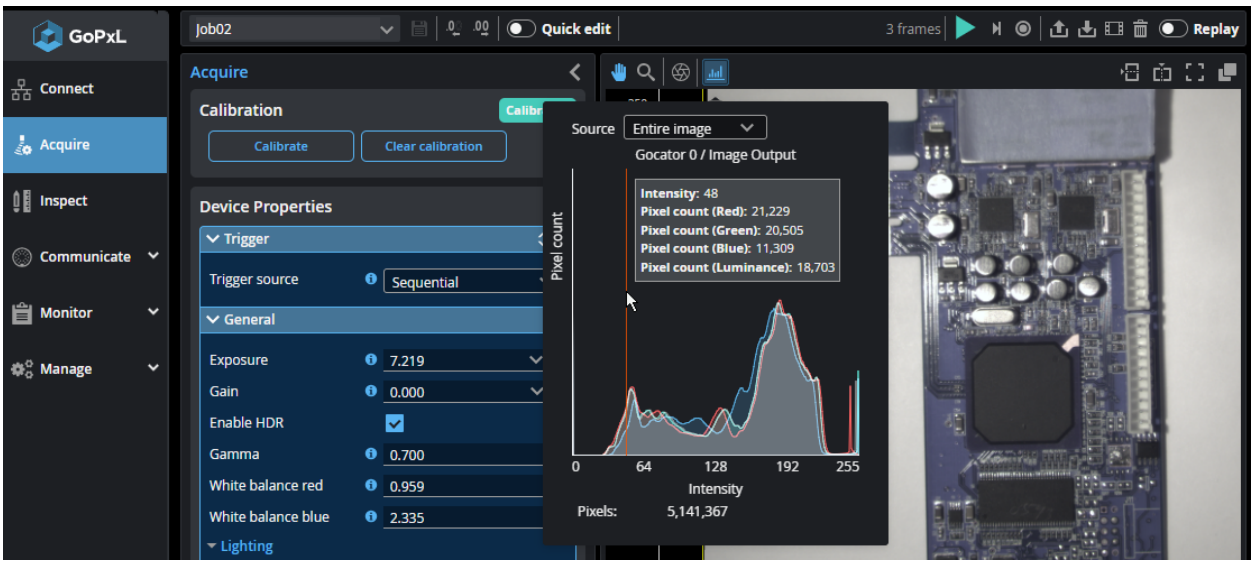
A new data viewer appears in the same browser window.

Use the tool bar at the top of the new data viewer to choose and modify the view. For more information, see *Data Viewer* on page 85.

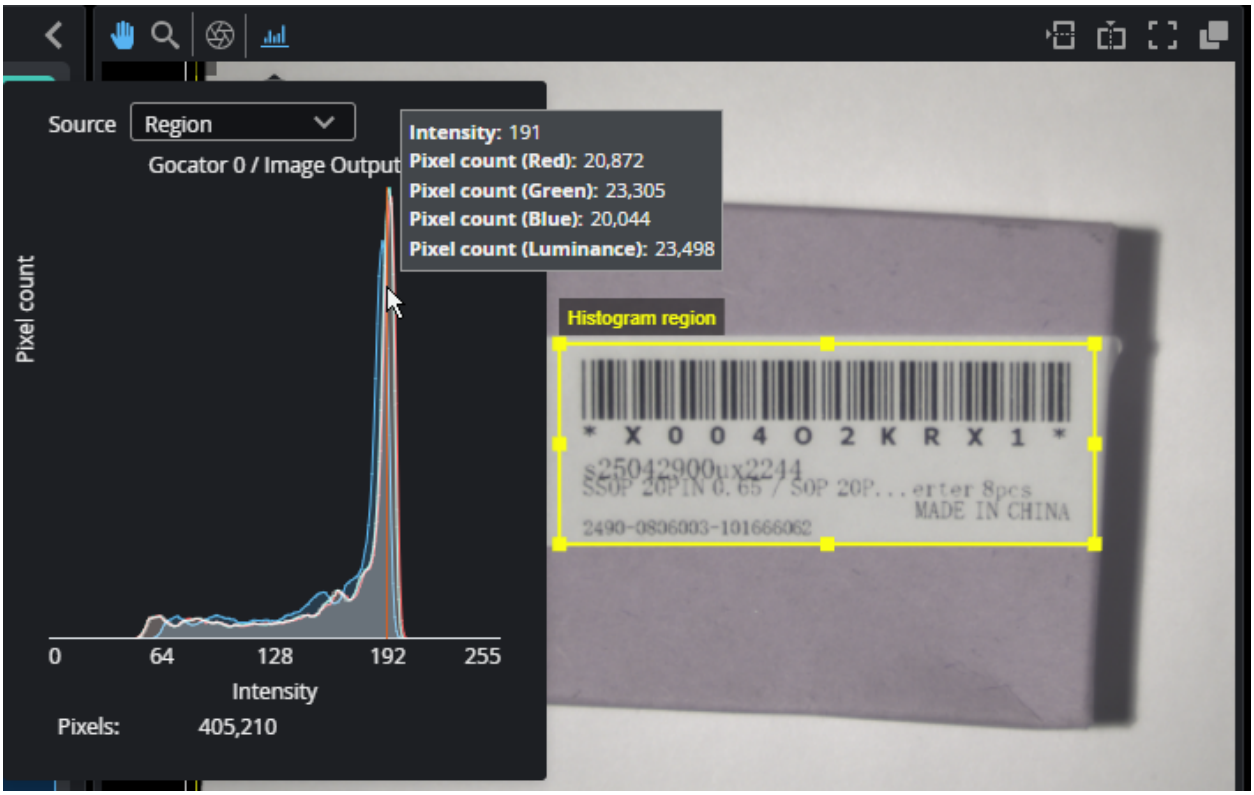
Pin outputs to the new data viewer as in the main view data viewer. For more information, see Displaying Multiple Measurements and Geometric Features (Pinning). Any outputs pinned in the main view when you open a new data viewer window appear already pinned in the new window, but pinning in data viewers is otherwise independent.

Histogram

Gocator 2D Smart Cameras let you evaluate image characteristics and quality by viewing a histogram of an image. To open the histogram viewer, click the histogram button ( ) at the upper left of the data viewer. Hover the mouse over the histogram to see pixel counts and other information for each intensity level.



You can display the histogram of the entire image, or of a region of the image to focus on an important feature.

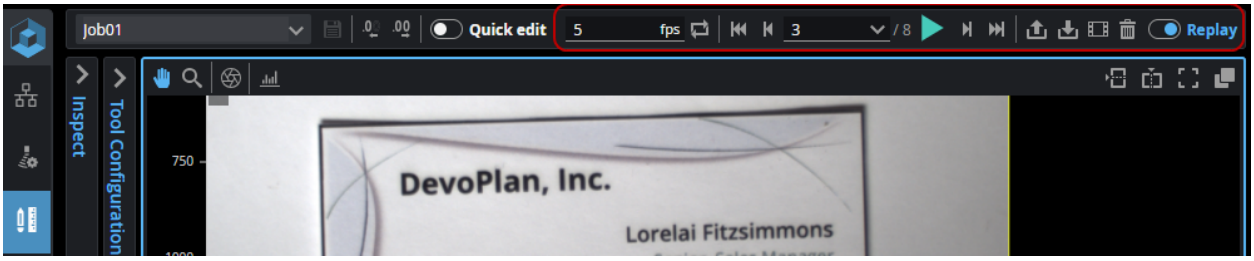

Source
Region
Gocator 0 / Image Output
Intensity: 191
Pixel count (Red): 20,872
Pixel count (Green): 23,305
Pixel count (Blue): 20,044
Pixel count (Luminance): 23,498
Pixel count
Histogram region
0
64
128
192
255
Intensity
Pixels: 405,210
* X 0 0 4 O 2 K R X 1 *
MADE IN CHINA

Recording and Working with Scan Data

You use the buttons in the right half of the global toolbar, at the top of the interface, to do the following:

- start and stop acquisition
- record replay data
- enable replay mode and play back replay data
- upload, download, and clear replay data
- use the Replay Editor

> When switching between scan modes, if you have previously recorded data, you must clear it before recording replay data of a different type.

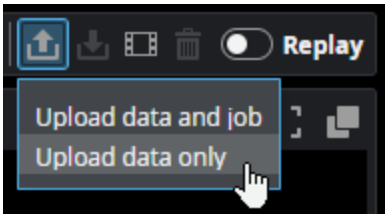


Depending on whether Replay mode (4, below) is disabled or enabled, the control buttons (2, below) change. When Replay is off, the controls let you start and stop data acquisition. When Replay is on, the controls let you play back recorded data.

The global toolbar with Replay disabled (top) and enabled (bottom)

Note that when GoPxL is in Replay mode, the data viewer is displayed with a blue border to help

	Element	Description
1	Frames per second and Repeat frames (looping)	These controls let you set the number of frames per second (fps) during playback, and also whether GoPxL repeats frames, by toggling the Repeat frames button ( ). For more information, see *Playing Back Recorded Data* on page 97.
2	Data playback and	These controls let you start and stop scanning, start recording and also

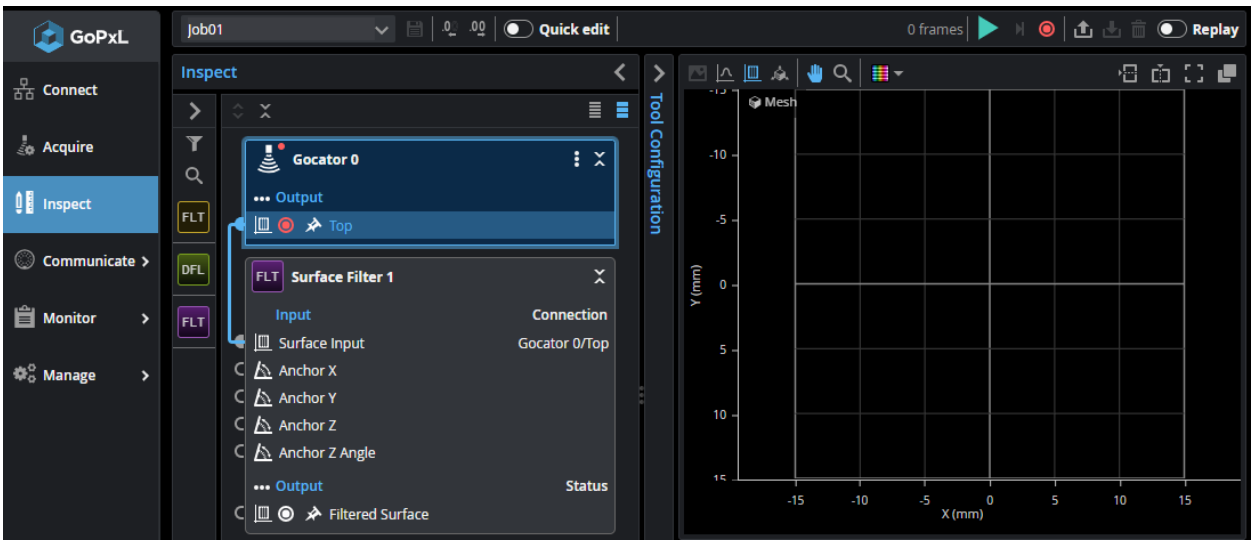
	Element	Description
	recording controls	play back replay data. The number of frames is also indicated. The controls that are available depends on whether Replay mode is disabled or enabled. For information on starting and stopping scans, see *Starting, Stopping, and Recording* below. For information on data playback, see *Playing Back Recorded Data* on page 97.
3	Replay actions	Lets you upload, download, and clear recorded data. You can also add and remove frames with the Replay Editor. For information on the Replay Editor, see *Using the Replay Editor* on page 98. When uploading data, you can decide whether to only upload data, or to upload both scan data and the associated configuration (tools and other GoPxL settings).  The file type of replay data is .gprec.
4	Replay toggle	Toggles between data acquisition (toggle is to the left) and data playback (toggle is to the right).

Starting, Stopping, and Recording

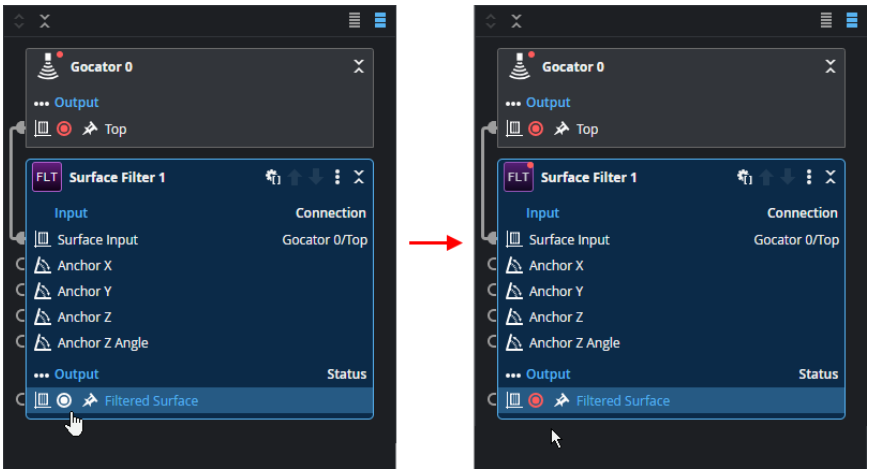
You start and stop data acquisition (and toggle recording) using the data control buttons when Replay is disabled.

	Element	Description
1	Start or stop scanning	The sensor starts scanning continually when you click the Start button ( ), and stops when you click the Stop button ( ).
2	Single frame (Snapshot)	The sensor acquires a single frame of data. For line profile sensors, you can't acquire a single frame of data if the sensor is in Surface mode, only in Profile or Image mode.
3	Record data	Toggles recording scan data as it is acquired. The number of frames recorded is displayed next to the Start button. When recording is enabled, the Record data button turns red ( ).

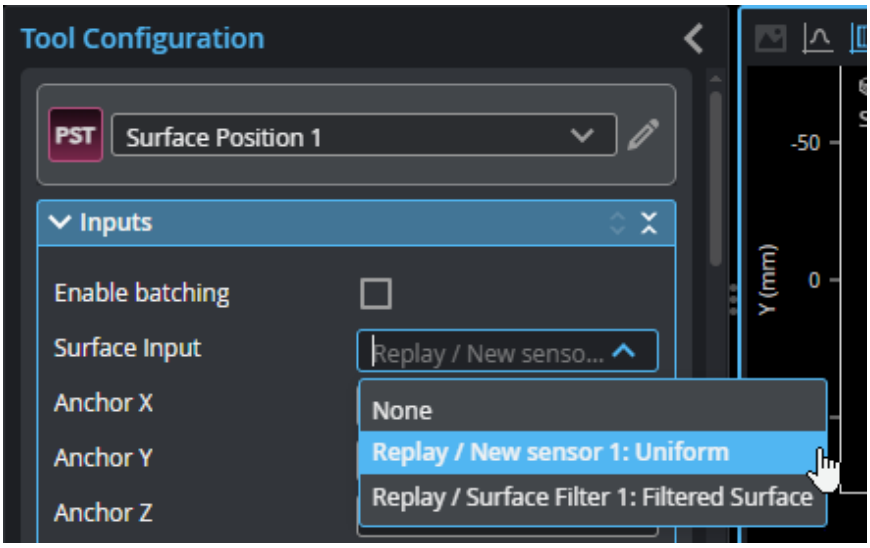
Normally, when you have enabled recording using the Record data button above the data viewer, GoPxL records scan data coming directly from the sensor or group of sensors (in the Tool Diagram, this is called the "sensor block").



You can also record a tool's output. An example of this is recording the Filtered Surface output of a Surface Filter tool. To do this, first enable recording above the data viewer as you would normally do. Then, expand a tool in the Tool Diagram, and click the white Record button next to an output.



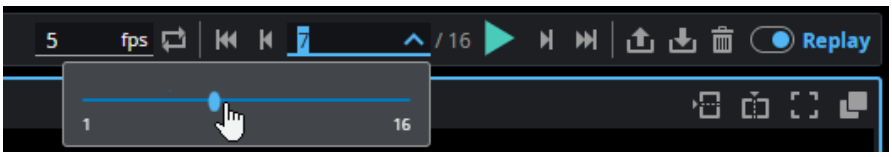
When you record data with output recording enabled for any of the tool's outputs, the resulting replay data will contain the scan data coming from the sensors, but also the tool's output data. Once you've downloaded this replay data and uploaded it to a sensor or PC instance of GoPxL, both sources are available in a tool's input.



You can choose to disable recording of the data coming from the sensor block to only record a tool's output. This can be useful when you have a long chain of pre-processing tools that performs filtering, transformations, or masking to isolate scan data, and so on. By recording the output of the last tool in the pre-processing chain, uploading that to a PC instance of GoPxL, you can then develop the decision-making part of your application separately, without the overhead of the pre-processing tool chain.

Playing Back Recorded Data

When Replay mode is enabled, you play recorded data back using the data control buttons.

	Element	Description
1	Playback rate and repeating frames	When "Repeat frames" ( ) is enabled, scan data is played back in a loop. This is useful when inspecting data or configuring tools using recorded data, and also for demos.
2	Go to previous / first frame	Displays the previous frame or jumps to the first frame of the recorded data and displays it.
3	Frame position	Lets you go to a specific frame, either by entering a value in the field, or by using a slider. 
4	Start playback	Starts playing back frames of the recorded data, at the specified number of frames per second.

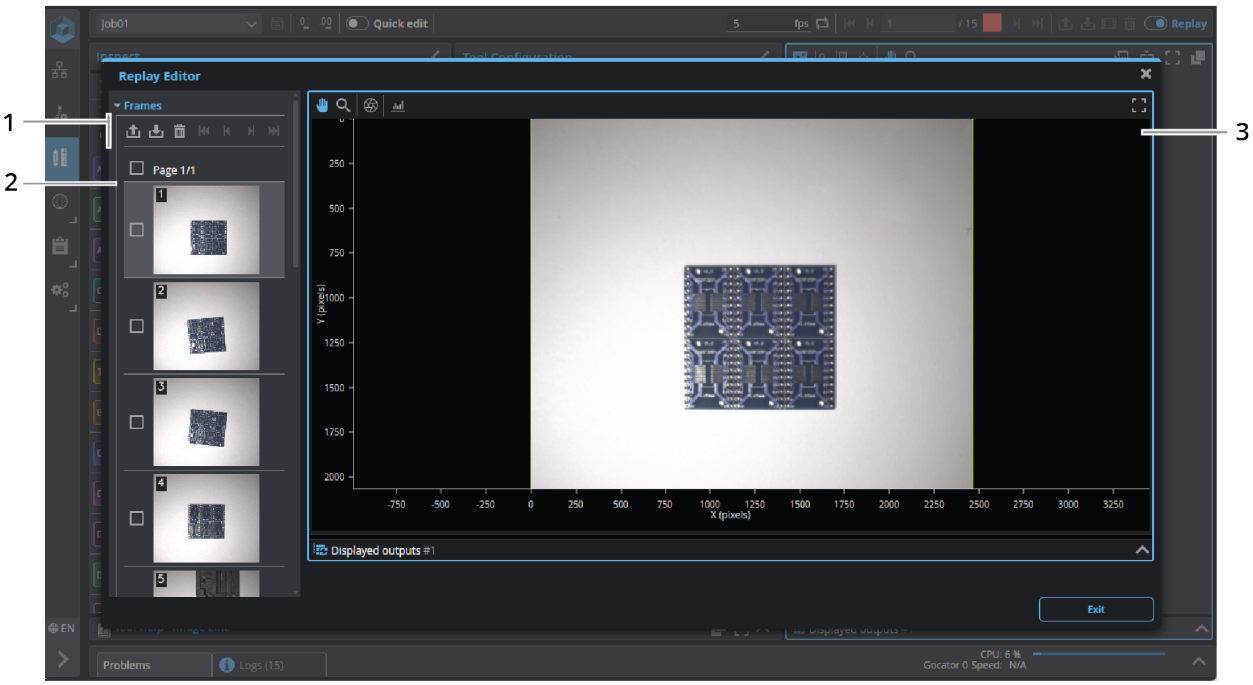
	Element	Description
5	Go to next / last frame	Displays the next frame or jumps to the last frame of the recorded data.

Using the Replay Editor

The replay editor lets you add or remove frames of recorded data in the sensor's buffer.

Some examples of using the Replay Editor:

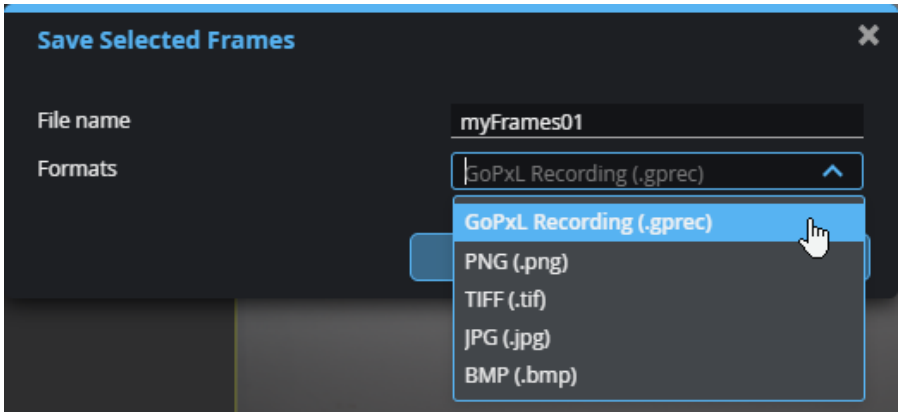
- Removing unwanted frames to produce a set frames of representative good parts (sometimes called "golden parts") or bad parts for evaluation, or for training with tools like GoPxL Anomaly Detector.
- Adding new frames.
- Preparing a test, training, or demo dataset.



	Element	Description
1	Editor tools	The replay editor tool bar lets perform operations on the frames of data in the sensor's buffer.

Download - Lets you download the currently selected frame or frames. You can select multiple frames by clicking the checkbox next to each frame you want to download.

When you click the Download button, you must provide a file name and a format for the download. If you choose GoPxL Recording, a .gprec file containing the selected frames is downloaded. If you choose one of the image formats, an image file or a .zip file containing multiple image files is downloaded.



	Element	Description
		Upload - Lets you upload either a .gprec file or individual image files. Frames are added to the end of the list of frames in the replay editor. **Delete frames** - Deletes the currently selected frames. Page buttons - The replay editor displays a maximum of 50 frames. Use the page buttons to move through pages when the device contains more than 50 frames.
2	Frame list	The list of frames in the camera's buffer. You can collapse the list to make more room in the interface.
3	Data viewer	The currently selected frame is displayed in the data viewer. For more information on the data viewer, see *Data Viewer* on page 85.

Before using the replay editor, you must either acquire some images, or upload previously acquired images.

Creating, Saving and Loading Jobs

Jobs contain sensor, tool, and control configurations for a particular inspection or quality control application. (For more information, see *Configuring GoPxL* on page 106.)

A sensor can store many jobs, subject to sensor storage limitations. Running GoPxL on a PC or using GoMax NX provides more storage. Being able to switch between jobs is useful when a sensor is used with different constraints during separate production runs. For example, width decision minimum and maximum values might allow greater variation during one production run of a part, but might allow less variation during another production run, depending on the desired grade of the part.

Most of the settings that you can change in GoPxL's web interface are temporary until saved in a job file. If there is a job file that is set as the default, it will be loaded automatically when GoPxL starts.

GoPxL lets you perform several job-related operations from the global toolbar at the top of the interface:

Other operations are available on the **Manage** > **Jobs** page. For more information on jobs and working with them, see *Jobs* on page 396.

Pro Tools: Installing CodeMeter Runtime

> You do not need to purchase a Pro Tools license and dongle, and install CodeMeter, to use a Gocator 2D Smart Camera. This is only required if you need to run GoPxL with Pro Tools *on a PC*or *on a GoMax unit*, for example for offline configuration or evaluation of acquired images.

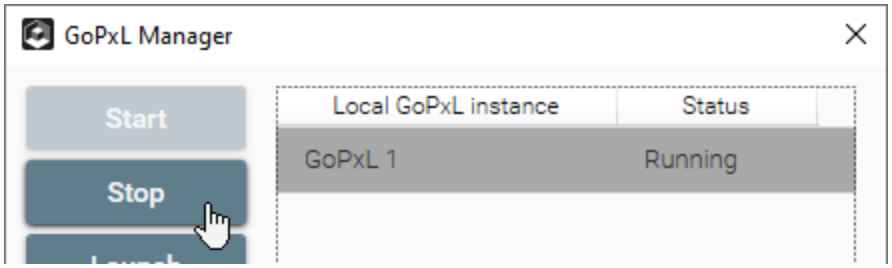
If you are using a Pro Tools tool on Windows, such as the Anomaly Detector toolset, while accelerating a sensor, you must install the CodeMeter runtime, which is required by the ProTools license and dongle. You install the runtime from GoPxL Manager.

To install the CodeMeter runtime

1. If GoPxL it is not already running, start GoPxL Manager.

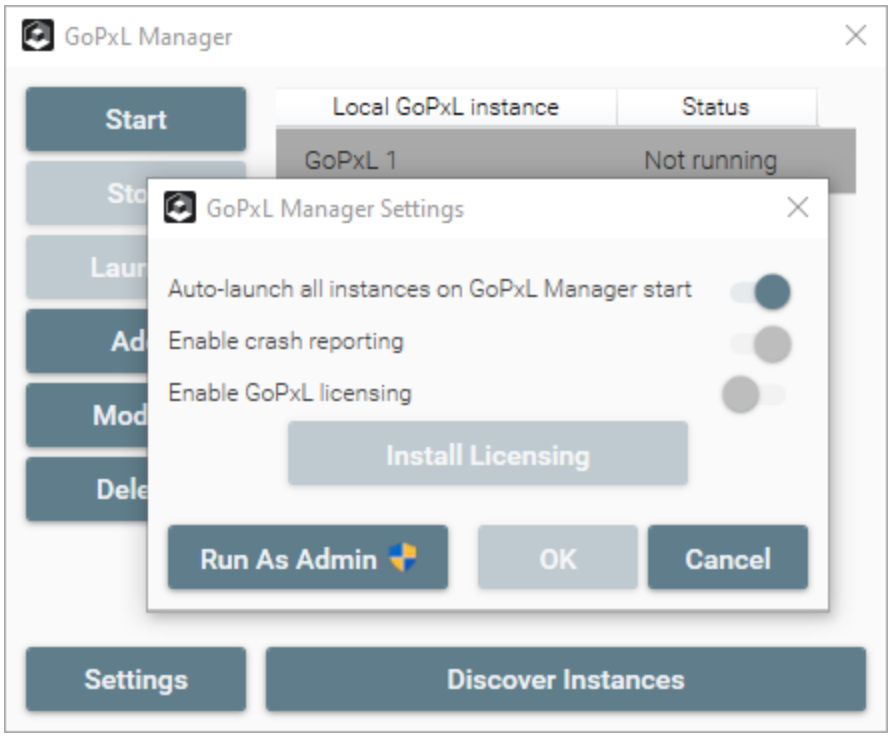
   For more information, see *Running GoPxL on a Windows PC* on page 403.

2. Stop any running instances of GoPxL.

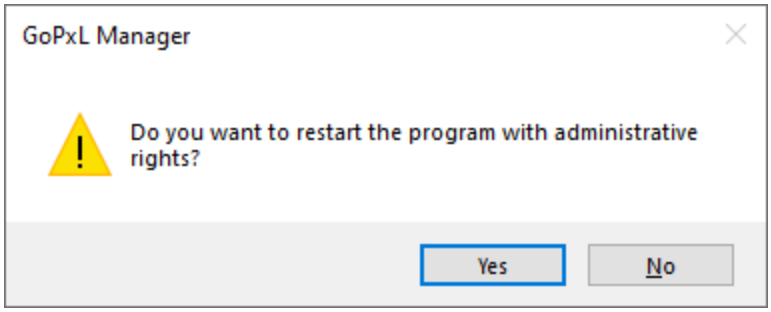


   When an instance stops, its status becomes "Not running."

3. Click **Settings**, and in the **GoPxL Manager Settings** dialog, click **Run As Admin**.



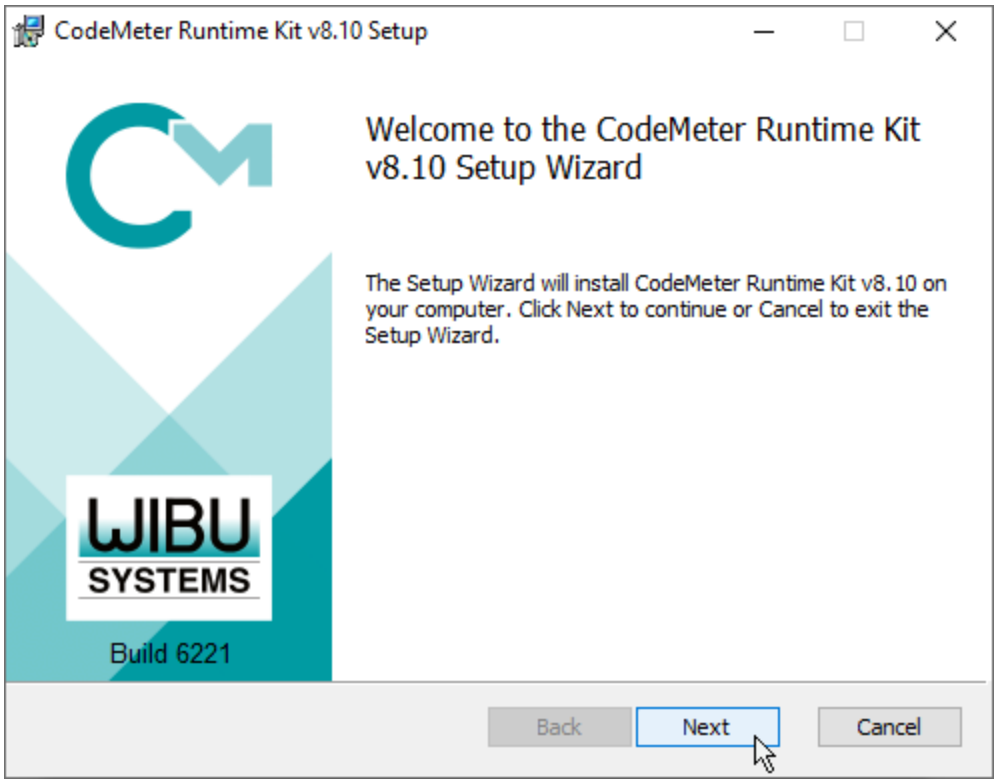
4. Answer Yes in the dialog that appears.



You will need to click Yes in the User Account Control dialog.

5. Click **Settings** again, and in the dialog, toggle on **Enable GoPxL licensing** and click **Install Licensing**.

The CodeMeter Runtime setup wizard launches.

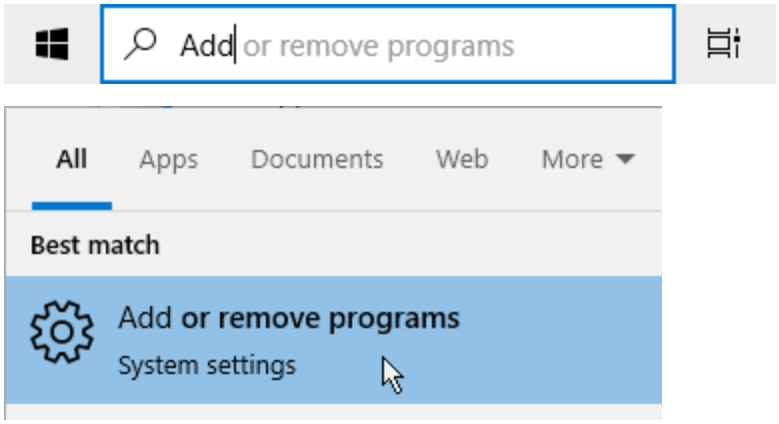


6. Go through the steps in the wizard to install the runtime.

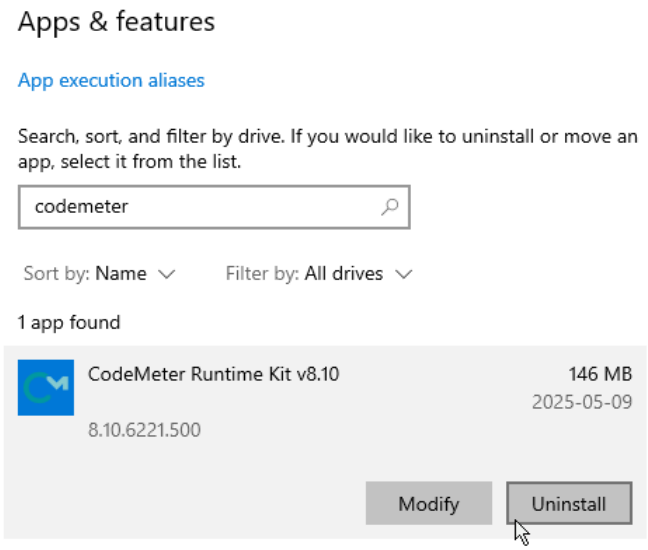
   You will need to accept the license agreement before continuing with the installation, and choose whether CodeMeter will be installed for all users on the PC. Leave the defaults in the Custom Setup step.

If you no longer need to use Anomaly Detector on your PC, remove the licensing software:

1. On your PC desktop, in the Search field, type "Add" and click "Add or remove programs" in the "Best match" list.



2. In the "Search this list" field, type "codemeter", click CodeMeter Runtime Kit to expand it, click **Uninstall**, and follow the instructions.


Apps & features
App execution aliases
Search, sort, and filter by drive. If you would like to uninstall or move an app, select it from the list.
codemeter
Sort by: Name
Filter by: All drives
1 app found
CodeMeter Runtime Kit v8.10
146 MB
2025-05-09
8.10.6221.500
Modify
Uninstall

Configuring GoPxL

The following sections describe how to configure GoPxL using its web interface.

To configure GoPxL, you connect to the camera's IP address (by default, 192.168.1.10). If you can't connect to GoPxL, use the GoPxL Discovery tool to find it and launch the GoPxL interface (for more information, see *GoPxL Discovery Tool* on page 473).

For information on configuring GoPxL programmatically, see *GoPxL SDK and REST API* on page 449.

Connect - Networking

Connect - Creating a System & Networking

The **Connect** page displays information related to your smart camera, and lets you configure network settings.

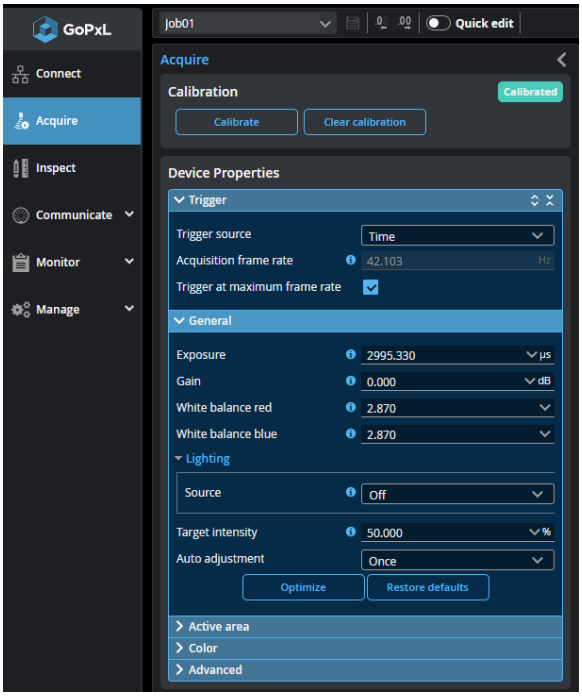
Network Settings

To configure the network settings:

1. Go to the **Connect** page.
2. Specify the type (Manual or DHCP), IP, subnet mask, and gateway settings.
   The sensor can be configured to use DHCP or assigned a static IP address by selecting the appropriate option in the **Type** drop-down.
3. Click **Save Changes...**.
   You will be prompted to confirm your selection.
   The sensor restarts, using the *new* IP address. It may take several seconds for the sensor to become available.

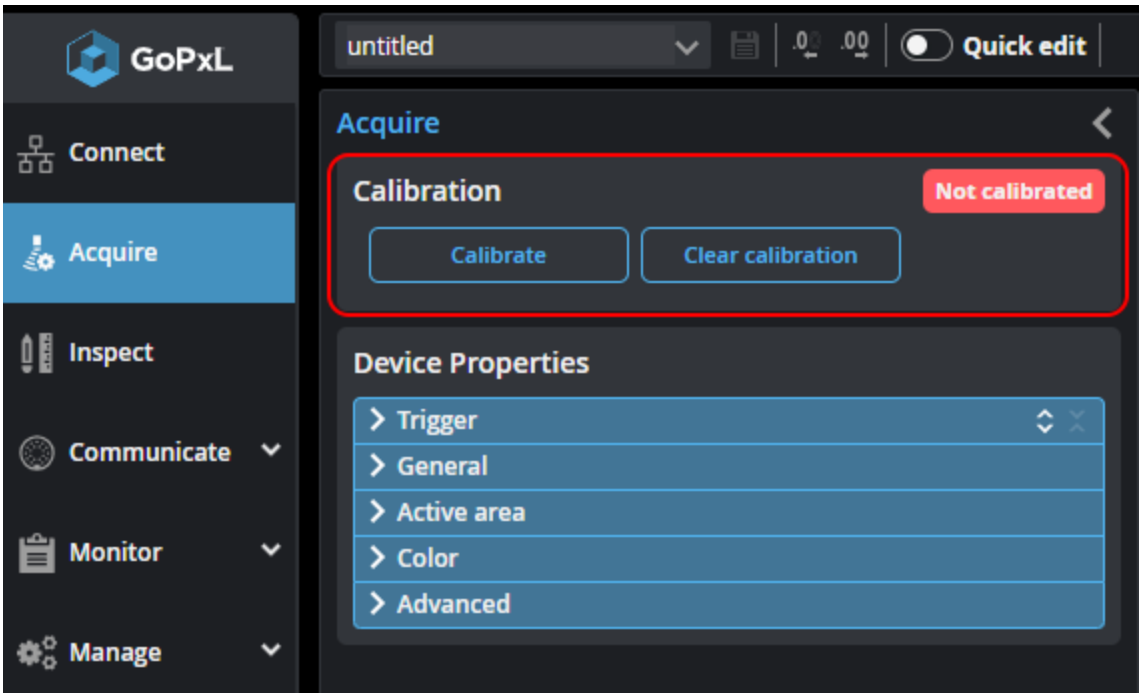
Acquire - Calibration and Image Acquisition

You calibrate smart cameras and configure acquisition in the **Acquire** page.



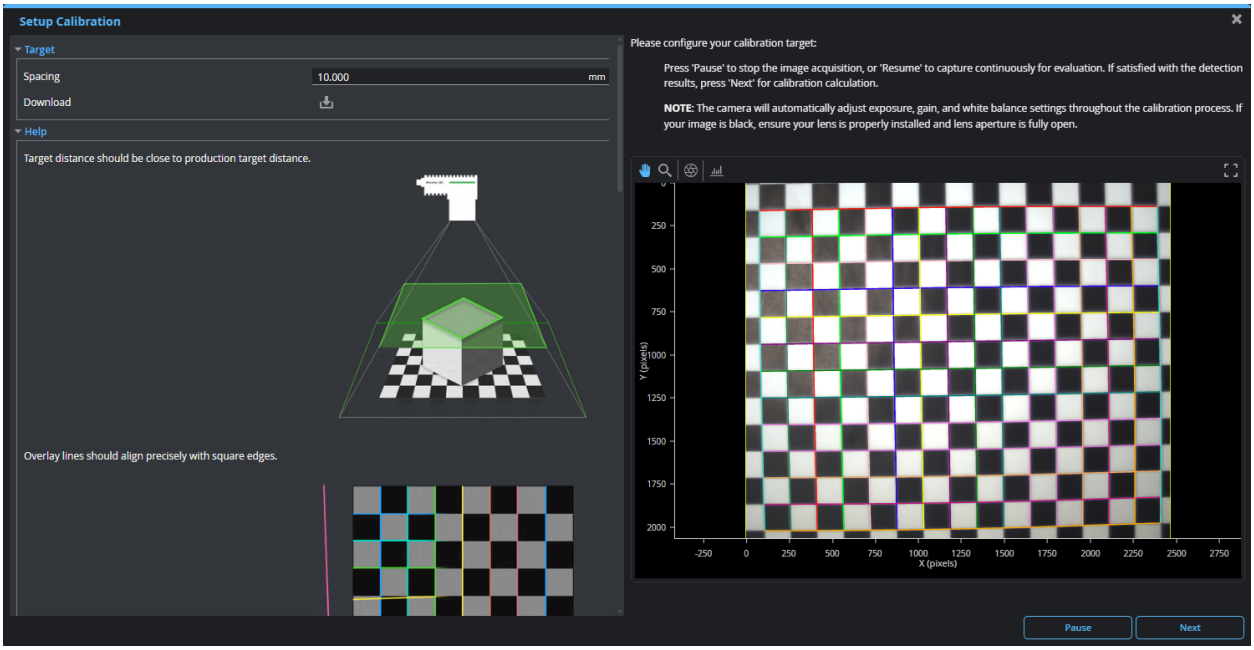
Calibrating Smart Cameras

GoPxL provides a wizard-based routine for calibrating a Gocator 2D smart camera, using a checkerboard target you can download from within GoPxL. After camera calibration, your smart camera reports measurements in real-world units (mm), and any lens-related distortions (for example toward the edges of the field of view) will be minimized.



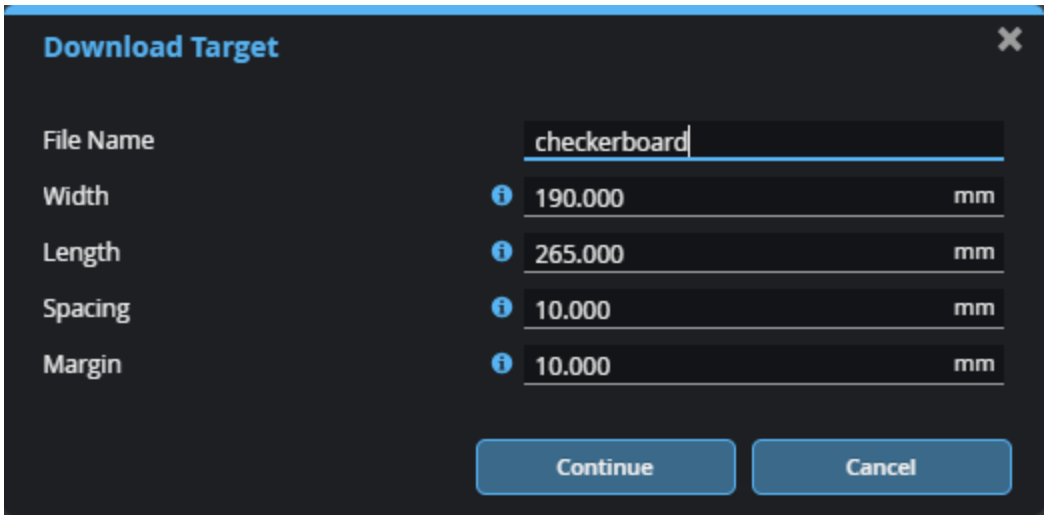
The smart camera's calibration status is indicated in the **Calibration** area on the **Acquire** page.

The calibration wizard provides animated images in the Help area to provide guidelines for the process. GoPxL displays what the camera is currently acquiring in the data viewer on the right of the wizard.



To calibrate the camera

1. Under **Target**, click the Download button ( ).
2. In **Download Target**, provide a file name, as well as the checkerboard characteristics.



The calibration target should fill the camera's field of view, so set the dimensions of the target accordingly. If you are not sure what characteristics to use, leave the default values for now.

3. Click **Continue** to download the checkerboard PDF.
4. Go to the download folder and print the target.

   If possible, print using heavier paper that will not curl as easily, for example, in humidity.
5. Center the target under the camera at the expected distance of what you will be inspecting.

   For example, if you are inspecting labels on the top of a box, the calibration target should be positioned at that distance.

   Make sure the calibration target is flat. Ensure lighting on the target is even.
6. Open the aperture fully (lowest f-stop value on the lens).
7. Focus the lens until the edges of the squares are as sharp as possible in the wizard's data viewer.

   Pause between adjustments to ensure what you see in the data viewer is up to date.

   Zooming in the data viewer may help.
8. Verify the following in the data viewer:

   There should be at least 10 squares along the longest side of the camera's field of view.

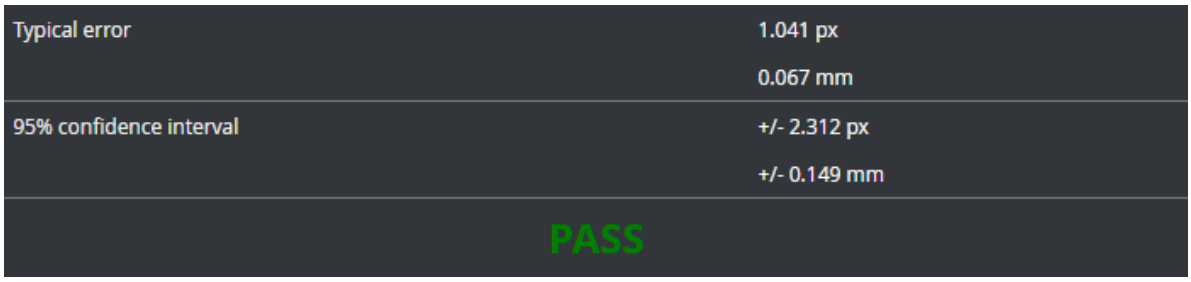
   Each square should be represented by at least 50x50 pixels. If there are to few pixels, increase their size .

   Color overlay lines should align with the squares of the target.
9. If necessary, click the Download button ( ) again, adjust the pattern characteristics in the **Download Target** dialog, reprint the target, and verify it in the data viewer.

   You may need to repeat this process a few times.
10. Measure the actual size of a square on the calibration target with a ruler or caliper.

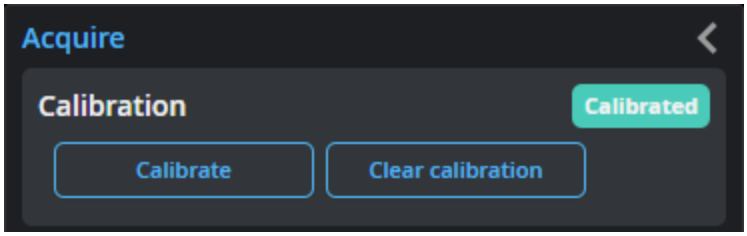
    This is important due to printer variations.
11. Under **Target** > **Spacing**, type the actual dimension of the square.
12. Click **Next**.

    GoPxL performs the calibration and displays results above the data viewer.



13. If the results are satisfactory, click **Exit**.

    The smart camera is now calibrated. Acquired images will include real-world measurements (millimeters). Measurement tools will also report real-world values, rather than pixels.

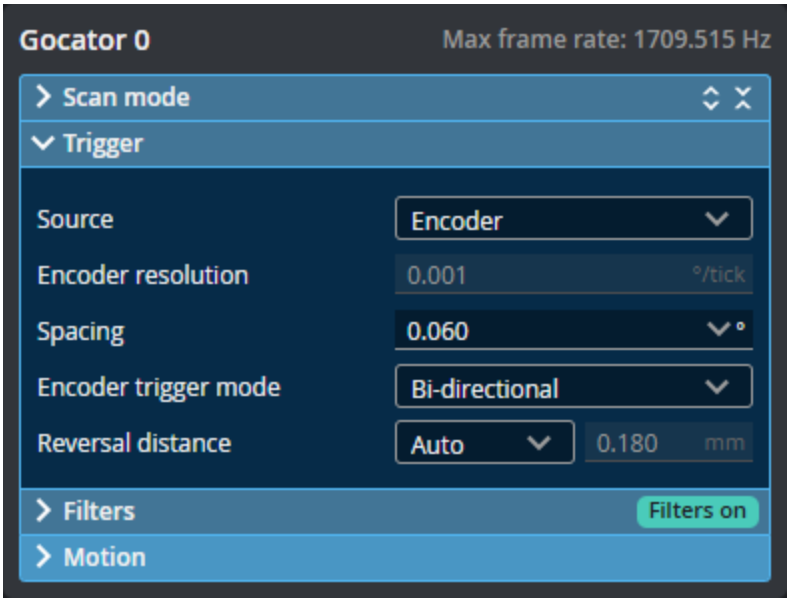


Triggers

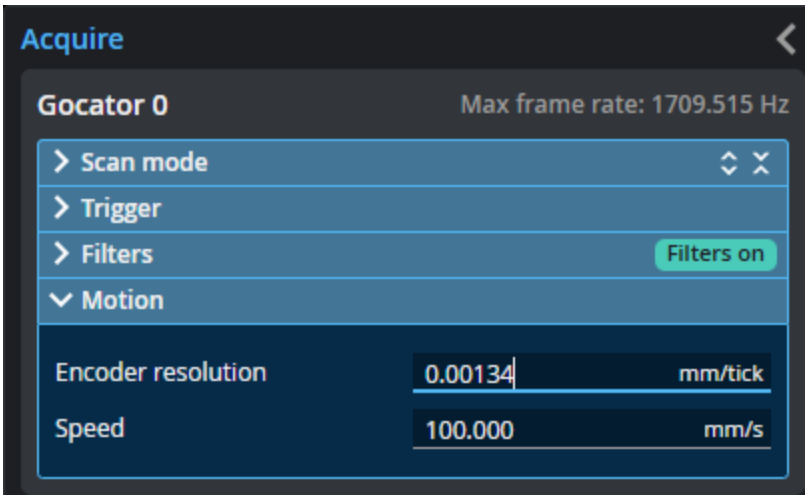
A trigger is an event that causes a sensor to acquire a single frame or image. You configure triggers on the **Acquire** > page, in the **Trigger** panel.

The sensor can be triggered by one of the sources described below.

When using encoder triggering (available with line profilers), the current encoder resolution is displayed in the **Encoder resolution** drop-down as a reference when setting trigger spacing.



To set the encoder resolution, expand the **Motion** section and set the value in **Encoder resolution**.



Trigger source descriptions

Trigger Source	Description
Time	Sensors have an internal clock that can be used to generate fixed-frequency triggers. The external input can be used to enable or disable the time triggers.
External Input	A digital input can provide triggers in response to external events (for example, a photocell). The external input triggers on the rising edge of the signal. When triggers are received at a frequency higher than the maximum frame rate, some triggers may not be accepted. Use the **Trigger Drops** indicator in the **Health** panel on the **Reports** page to check for this condition; for more information, see *Monitor - Measurement Stats, Health & Performance* on page 387.
Software	A network command can be used to send a software trigger. For more information, *Protocols (PLCs and other hardware)* on page 472.

Depending on the setup and measurement tools used, the CPU utilization may exceed 100%, which reduces the overall acquisition speed.

Trigger Settings

After you choose the trigger source, configure the other parameters in **Acquire** page > **Trigger** section.

Parameter	Trigger Source	Description
Trigger source	n/a	
Acquisition frame rate		The frame rate of image acquisition. The maximum frame rate is limited by the exposure, active area, and light duty cycle settings. When **Trigger at maximum frame rate** is enabled, this is read-only.
Trigger at maximum frame rate	Time	When this is enabled, frames are acquired at the maximum possible frame rate.
Trigger Delay	External Input	Specifies the amount of time or the distance the sensor waits before producing a frame after the external input is activated.

Parameter	Trigger Source	Description
		This is used to compensate for the positional difference between the source of the external input trigger (for example, a photocell) and the sensor.

General - Exposure, HDR, Gamma, Lighting

You configure the main 2D image acquisition settings, such as exposure and gain, in the **General** panel. You also enable and configure lighting, and run the automatic optimization in this panel.

Parameters

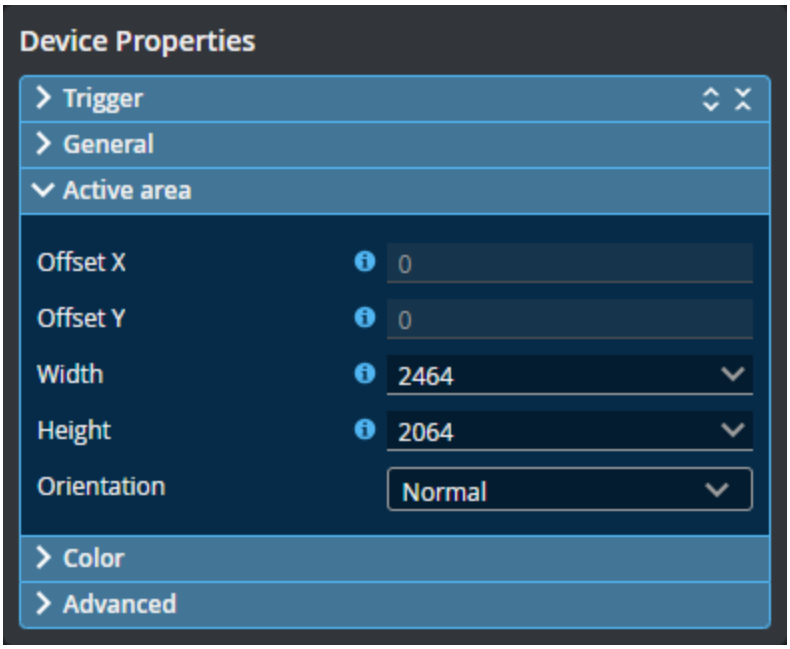
Parameter	Description
Exposure	The exposure time (time during which the shutter is open) in microseconds (µs). Longer exposures can help to detect light on dark or distant surfaces, but increasing exposure time decreases the maximum frame rate. Different target surfaces may require different exposures for optimal results.
Gain	Amplifies the analog video signal by the provided gain (dB). Increasing gain can let you decrease exposure time, as long as your application is not affected by the increased noise, or you can otherwise compensate for this noise.
White balance red White balance blue	The ratio of the red and blue color components, respectively, to the green color component.
Lighting	Expandable section containing lighting-related settings. Depending on the source you select, additional lighting settings may be available. **Source** Sets the internal acquisition or I/O source signal that is sent. One of the following: Off: Lighting is off. Continuous: Lighting is always on. Strobe: Strobes the light during exposure, and enables overdrive mode for supported smart vision lights. Readout active: After exposure has finished and frame readout (ADC conversion) is occurring. Trigger input: On trigger input. **Intensity** The intensity of the light source. **Invert** Whether to invert the selected source signal.
Target intensity	The intensity the camera will target by adjusting exposure and gain.

Parameter	Description
	When **Auto adjustment** is set to Always, the settings are adjusted for each frame. When set to Once, the settings are adjusted when you click **Optimize**.
Auto adjustment	Performs auto adjustment of the enabled auto adjust settings in **Advanced** > **Auto adjustment settings**. One of the following: Once or Always.
Optimize button	When set to Once, click **Optimize** to optimize once, when you click the **Optimize** button. When set to Always, the camera adjusts the settings for each frame.

Active Area

By default, the active area covers the sensor's entire scan area. By reducing the active area, the sensor can operate at higher frame rates. You can also reduce the active area to exclude areas that are affected by ambient light, or to avoid supports or parts of a transport system (such as a conveyor).

You set the active area in the **Acquire** page, in the **Active area** section under **Sensor Properties**.



To reset the active area

- Click the **Reset** button.

Active area parameters

Name	Description
Offset X Offset Y	The horizontal and vertical offset of the active area in pixels from the origin, respectively.
Width	The width and height of the active area (and therefore of the image the camera

Name	Description
Height	produces).
Orientation	The orientation of the produced image. One of the following: Normal Flipped vertically Mirrored horizontally Rotated 180 degrees

Color

If your smart camera supports color, you configure the smart camera's color in this section. Adjusting the color may help improve contrast and therefore feature or flaw detection, or improve color accuracy when your application needs it.

Parameters

Parameter	Description
Hue	The hue correction of the color tone.
Saturation	Amplification of the chrominance (chroma) signal in the color space.
Color correction	Color correction applied to the RGB pixels in an image using the following 3x3 transformation matrix. $$\begin{bmatrix} R_{out} \\ G_{out} \\ B_{out} \end{bmatrix} = \begin{bmatrix} Gain00 & Gain01 & Gain02 \\ Gain10 & Gain11 & Gain12 \\ Gain20 & Gain21 & Gain22 \end{bmatrix} \times \begin{bmatrix} R_{in} \\ G_{in} \\ B_{in} \end{bmatrix}$$ where: • Gain00 is the red contribution to the red pixel • Gain01 is the green contribution to the red pixel, • Gain02 is the blue contribution to the red pixel, • Gain10 is the red contribution to the green pixel, • and so on. You set **Color correction** to one of the following: **Off** - No color correction is applied. **Indoor** - Color correction for indoor lighting is applied using the following matrix. $$\begin{bmatrix} 2.0 & 0.0 & 0.0 \\ 0.0 & 2.0 & 0.0 \\ 0.0 & 0.0 & 2.0 \end{bmatrix}$$

Parameter	Description
	Outdoor - Color correction for outdoor lighting is applied using the following matrix. $\begin{bmatrix} 3.0 & 0.0 & 0.0 \\ 0.0 & 3.0 & 0.0 \\ 0.0 & 0.0 & 3.0 \end{bmatrix}$ **White LED** - Color correction for a white LED light source is applied. $\begin{bmatrix} 4.0 & 0.0 & 0.0 \\ 0.0 & 4.0 & 0.0 \\ 0.0 & 0.0 & 4.0 \end{bmatrix}$ **Custom** When the color correction is set to Custom, you provide the matrix (gain) coefficients manually in the **Correction matrix** section.
Chroma	On color cameras, lets you choose between color and grayscale.
Accelerate color processing	

Advanced Settings

The settings in the **Advanced** panel on the **Acquire** page let you configure more advanced settings, including the auto adjust settings that apply when you click **Optimize** in the **General** panel (when **Auto adjustment** is set to Once), or when Auto adjustment is set to Always; for more information on optimizing the camera, see *General - Exposure, HDR, Gamma, Lighting* on page 112.

Parameters

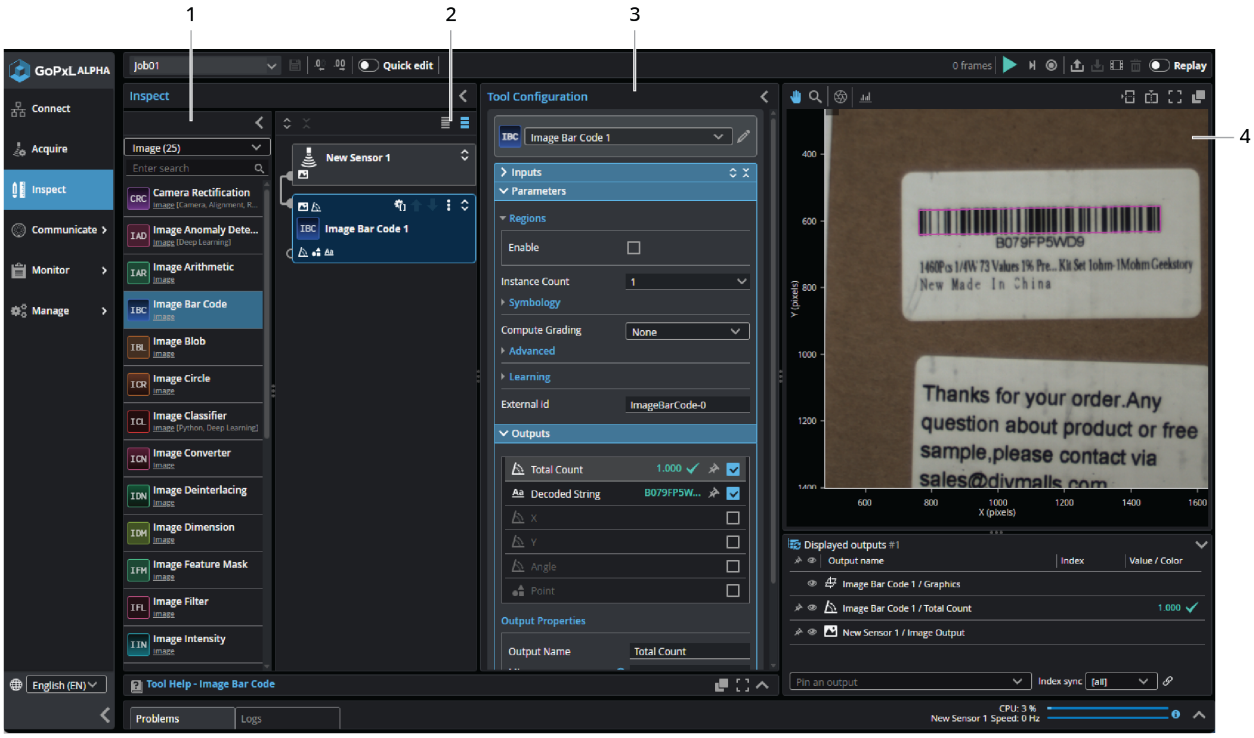
Parameter	Description
Auto adjustment settings	Expand this section to access and modify settings related to the automatic adjustment of exposure, gain, and white balance. For more information, see *Parameters - Auto adjustment settings* below.
Max message age	TODO
Light connectivity	The connector to which the light source is connected. One of the following: **Light connector** - Use this option if you are using the RMX140 ring light, which must be connected to the camera's 5-pin Camera light connector. **Power IO** - Use this option if you are using a light other than the RMX140. In this case, you must provide power to the light from an external source.
Manage all features	Enabling this setting displays many advanced parameters; it also results in slower parameter changes and saving or loading of job files. Only enable this if told to do so by LMI Support.

Parameters - Auto adjustment settings

Parameter	Description
Offset X / Offset Y Width / Height	Set the offset, width and length to limit the region GoPxL uses during auto adjustment. Otherwise, GoPxL uses the entire scan image for auto adjustment.
Adjust exposure Exposure maximum Exposure minimum	When enabled, set the range of exposures the camera is allowed to use when performing auto adjustment.
Adjust gain Gain maximum Gain minimum	When enabled, set the range of gain the camera is allowed to use when performing auto adjustment.
Adjust white balance	When enabled, the camera adjusts the white balance when performing auto adjustment. You may want to adjust the white balance to a white reference target without your production target in the FOV; if you do this, perform optimization once with **Adjust white balance** enabled, and then disable it.

Inspect - Measurement and Processing

You add and configure measurement and processing tools on the **Inspect** page.



	Element	Description
1	Tool list	Lists all of the available tools. To add a tool, double-click it or drag and drop it in the Tool Diagram panel to the right of the list. You can filter tools using the filter at the top of the Tool list. You can also search for tools by typing in the search field. Tools are listed as you type.
2	Tool Diagram	Lists the tools that you have added. The Tool Diagram lets you better visualize the workflow in complex tools chains. For more information, see *Working with Tool Chains* on page 119.
3	Tool configuration	You configure a tool using the **Inputs**, **Parameters**, and **Outputs** expandable sections in the **Tool Configuration** panel. Typically, you configure tools from top to bottom, that is, by setting and configuring the inputs first, then the parameters, and finally the outputs, all in the order they are presented in these sections. You can select which tool you want to configure from the tool selector drop-down, or by selecting the tool in the Tool Diagram to the left. For information on parameters common to most tools, see *Tool Configuration* on page 125. For information on the different types of tools, see their topics in the

	Element	Description
		following sections. Feature tools: See *Feature Tools* on page 148. Array tools: See *Array Tools* on page 140. Measurement Formula tool: See *Measurement Formula Tool* on page 270. Script tool: See *Script Tool* on page 272.
4	Data viewer	When configuring tools, you can graphically configure regions of interest. The data viewer's toolbar lets you change the display mode (for more information, see *Data Viewer* on page 85) and split a data viewer or pop out a data viewer in a new browser window.

Adding a Tool

You can add a tool from the tool list in two ways: by dragging and dropping the tool onto the Tool Diagram panel or by double-clicking the tool in the Tool List panel.

GoPxL automatically sets the added tool's input to an appropriate output when possible. The output GoPxL selects is either directly from a sensor or sensor group, or from the output of another tool, depending on various factors (see *Tool Connection Logic* below).

When dragging and dropping a tool, a line in the Tool Diagram shows where the tool will be added when you drop it.

When double-clicking a tool in the tool list, the new tool is added after the currently selected tool in the Tool Diagram.

Tool Connection Logic

When you add a tool, GoPxL looks for compatible outputs among the other tools to set the added tool's inputs, starting with the tool or sensor directly "above" the added tool in the list of added tools (in the Tool Diagram panel). By "compatible," we mean, for example, data type (Profile vs. Surface, or uniform vs. non-uniform or point cloud data). GoPxL ignores tools "below" the added tool for this initial search. GoPxL checks the compatibility of all the outputs in a tool before moving up to the next tool.

When a preceding tool has multiple enabled outputs, GoPxL goes through them in the order they are defined in *the tool* (that is, from the top down).

If no outputs from the preceding tools are compatible with the tool's inputs, GoPxL searches through the sensor or sensor group outputs.

For information on other operations you can perform in the Tool Diagram, see *Working with Tool Chains* on the next page.

Removing a Tool

If you no longer need a tool, you can remove it in the Tool Diagram. Removing an unused tool can increase performance, as it no longer takes up processing time.

To remove a tool, select it in the Tool Diagram and press Delete on your keyboard. You can also open the tool's Action menu ( ) and choose **Delete**. You can also press the Delete key on your keyboard. If multiple tools are selected, a delete operation will affect all of them.

For information on other operations you can perform in the Tool Diagram, see *Working with Tool Chains* below.

Duplicating a Tool

You can quickly create a copy of a tool you have previously added in GoPxL in the Tool Diagram panel. The tool inputs, parameter settings, and enabled outputs of the original are copied. This is useful, for example, when you need almost identical tools with only minor variations, such as different maximum and minimum decision values, or regions of interest shifted from one component to another.

In the Tool Diagram panel, you duplicate a tool by opening the tool's Action menu ( ) and choosing **Duplicate**. You can also press the D key on your keyboard. If multiple tools are selected, a duplication operation will affect all of them.

For information on other operations you can perform in the Tool Diagram, see *Working with Tool Chains* below.

Renaming a Tool

In the Tool Diagram panel, you rename a tool using the action menu of an individual tool.

To rename a tool in the Tool Diagram panel

1. Click the action menu icon ( ).

   A context menu containing various actions appears.

2. In the context menu, choose **Rename**.
3. In the tool name field, rename to the tool.
4. Press Enter on the keyboard or click outside the text field.

For information on other operations you can perform in the Tool Diagram, see *Working with Tool Chains* below.

Working with Tool Chains

The Tool Diagram panel between the tool list and Tool Configuration panel provides a visual representation of the data flow in a system (the output from a sensor group, and the input and output of tools). It lets you create and view complex tool chains with drag-and-drop, mouse, and keyboard operations, letting you quickly and easily implement and maintain applications requiring multiple, interconnected tools. Tools are represented by "blocks" in the tool chain.

All data types (if available) and their relationships between tools are displayed:

- Measurements (for use as anchors)
- Geometric features
- Tool data (some data outputs are intended to be consumed only by SDK applications and can't be used as part of a tool chain)

You can perform many operations on multiple tools at once. For more information, see *Keyboard and Mouse Shortcuts* on page 80.

For details on how the Tool Diagram panel displays information, see *Understanding the Data Flow in Tool Chains* on the next page.

For details on how to connect and disconnect, see *Connecting Tools* on page 123 and *Disconnecting Tools* on page 124.

The Tool Diagram panel is open by default. The inputs, parameters, and outputs of the tool selected in the Tool Diagram panel are displayed to the right of the Tool Diagram panel.

Tool Diagram panel showing sensor group, tools, outputs/inputs, and data flow connections.

At the top of a tool, an action menu provides functions to rename, duplicate, and delete the current tool.

Action menu (collapsed tool)

Changing Tool Display Options

The buttons at the top of the Tool Diagram panel let you control how the panel displays sensor groups, tools, and the data flow (tool chain). Buttons at the top of individual tools let you organize the tools in the list, as well as name, duplicate, and delete them.

The following describes the toolbar's functions:

1. Open all: Expands the sensor or sensor group block and all tools in the Tool Diagram panel, displaying a list of available inputs and enabled outputs for each one.
2. Close all: Collapses all items in the Tool Diagram panel.
3. Compact view: Hides the list of small input and output icons that indicate the types of the inputs and outputs the sensor or a tool has.

4. Standard view: Shows small icons that indicate the types of the inputs and outputs the sensor group or a tool has. For a list of inputs and outputs, see *Data Types* below.

Reordering Tools

You can reorder tools in the Tool Diagram to organize them more logically or to better represent the data flow. For example, you could group tools that output geometric features with the tools that use them, or group tools you use as anchors with the tools that use those anchors. Note that the order of tools in the Tool Diagram panel does not affect the flow of data.

You can reorder tools you have added by dragging and dropping them to the position you want in the list of tools. You can also click the up and down arrows in a selected tool block, or a block the mouse is hovering over. You can use your keyboard's up and down arrow keys to move the selected tool up or down in the tool chain. If multiple tools are selected, a move operation will affect all of them.

Moving the Surface Filter tool to the top of the tool chain.

Data Types

GoPxL represents the data types of inputs and outputs with icons. When an input or an output

Icon	Description
	Measurement
	Geometric feature
	Multiple input types

Data type icons, as well as tool icons, can be modified with a "[ ]" badge to communicate information related to arrays and batching. For more information, see *Arrays, Batching, and Aggregation* on page 126.

Understanding the Data Flow in Tool Chains

The rectangular blocks displayed in the Tool Diagram represent a sensor or group of sensors at the top, and any tools you have added below that. Sensor groups display output connection nodes, whereas tools display both input and output connection nodes.

The appearance of nodes changes depending on whether they are connected and whether they are selected. Connections that are used are solid. Connections that are not used are empty. When a sensor or tool is expanded, you can see which specific inputs or outputs are used as part of the tool chain.

When a tool is collapsed, however, you only know that at least one input or output is used (or none at all). For example, looking at the collapsed Feature Dimension tool at the bottom, we know that at

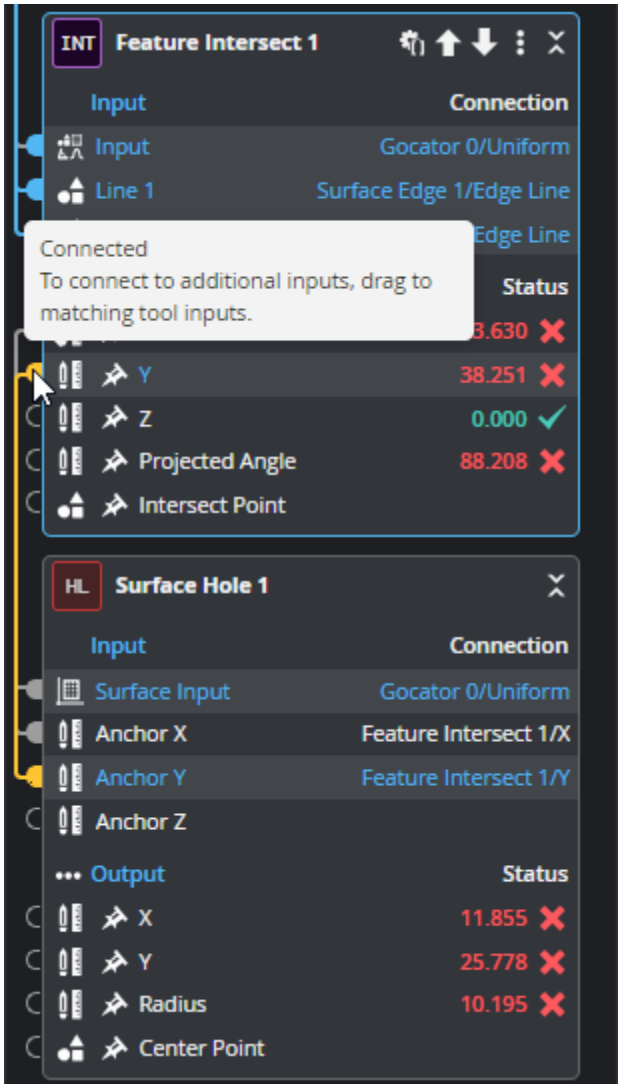
least one input (the connection node at the top) is used, and that none of the tool's outputs are used. Also, we know that inputs and outputs of the three collapsed tools at the top are used, but not exactly which ones.

In both cases (collapsed or expanded), the data flow of the selected tool or sensor block is indicated by blue connection lines. For more information, see below.

By default, sensors and tools are collapsed, but you can expand them individually by clicking the expand / collapse button at the top right of a tool to display the complete list of available inputs and outputs. Note that for an output to be listed in the Outputs section, it must be enabled in the tool's configuration: in the tool's Output list, only enabled outputs are listed.

When a tool is collapsed, you can see which inputs are connected and to what source, and also see which outputs are enabled and their values, by clicking one of the rows of small icons at the top or bottom of the block. You can also pin outputs from here; for more information on pinning, see *Displaying and Pinning Outputs* on page 87.

If you hover the mouse pointer over a blue connected node, a part of the blue connection line is highlighted in yellow to indicate what it is connected to. In the image below, you can see that by hovering over an output (the Y measurement of the Feature Intersect tool) it is used as an input (the Y anchor) of the Surface Hole tool at the bottom.



If a tool's input is missing, that input is displayed in red in the Tool Diagram panel to show that you must connect it to something.

Collapsed tool and expanded tool with a missing input

You can also consult the **Problems** tab in the **System Messages** panel at the bottom of the interface to see which tools are missing inputs.

In the Tool Configuration panel, a tool whose input is no longer available (for example, you have removed the preceding tool in the tool chain) displays "(missing)" in the input parameter.

Connecting Tools

The Tool Diagram panel lets you quickly connect tools using drag-and-drop operations.

In the following, we connect a measurement output from one tool to an anchor input of another tool. However, the same procedure applies when connecting other kinds of outputs to inputs, such as when connecting a geometric feature output from one tool to the input of another tool, or when connecting Surface output (such as the output from the Surface Filter tool) to the input of another tool.

To connect a tool's output to another tool's input

1. Make sure you have added at least two tools and that you have configured the tools higher in the tool chain.

   The output you want to connect must be enabled in the first tool. You can enable a tool's output in the tool's **Outputs** section in the **Tool Configuration** panel, or by using the **Output** menu in the tool's block in the Tool Diagram panel.

   For information on adding tools, see *Understanding the Data Flow in Tool Chains* on page 121.

2. Locate the tool whose output you want to use (the "source" tool).
3. Do one of the following:

   From an expanded tool

   a. Click and hold the output you want to connect to the other tool's input and drag it to the input.

   From a collapsed tool

   a. Click the small row of icons at the bottom of the tool to expand the list of
4. Drop the output on the desired input.

   A new connection appears between the first tool's output and the second tool's input.

Disconnecting Tools

You can quickly disconnect an input in the Tool Diagram panel, but only if the tool containing the input is expanded.

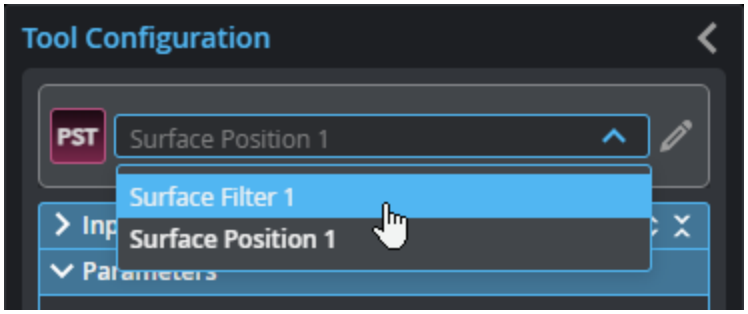
To disconnect an input in a tool:

1. If the tool isn't expanded, click the Expand button at the top of the tool.
2. In the expanded tool, move the mouse pointer over the input you want to disconnect and move it to the right until the pointer is over the Disconnect icon.
3. Click the Disconnect icon.

   The input is disconnected from the other tool's output.

Tool Configuration

You configure a tool's inputs, parameters, and outputs in the tool configuration area to the right of the Tool Diagram.

	Element	Description
1	Tool Diagram	Lists the tools that you have added. The Tool Diagram lets you better visualize the workflow in complex tools chains. For more information, see *Working with Tool Chains* on page 119.
2	Tool configuration	Lets you configure the selected tool using the **Inputs**, **Parameters**, and **Outputs** sections in this area. You can select which tool you want to configure from the tool selector drop-down, in addition to selecting the tool in the Tool Diagram. 

The following sections describe the parameters common to most tools.

Inputs and Outputs **126**
Arrays, Batching, and Aggregation **126**
Batching 127
Aggregating 129
Batch and Array Indicators in GoPxL 130
Regions **132**
Standard Regions 132
Flexible Regions 133
Working with Circular and Elliptical Regions 134
Region Rotation 136
Feature Points **136**
Geometric Features **137**
Fit Lines **137**
Decisions **138**
Measurement Anchoring **138**

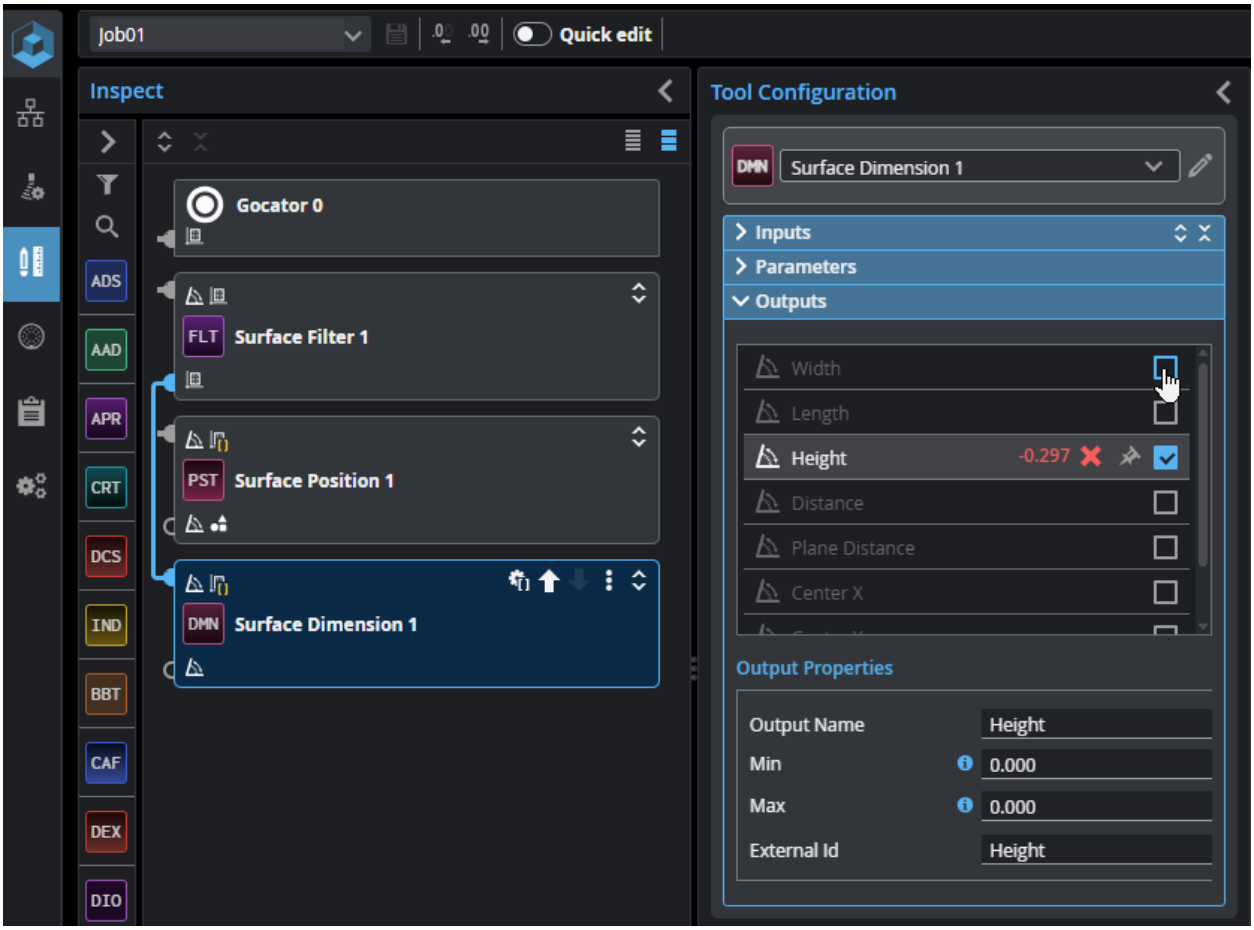
Inputs and Outputs

All tools take one or more inputs, either directly from a sensor or sensor group, or from another tool. All tools produce output, such as measurements or modified scan data.

The tool performs measurements on the input or processes it, and produces outputs.

Most tools can also take "arrays" as inputs. For more information on arrays and how tools work with them, see *Arrays, Batching, and Aggregation* below.

All of the outputs (measurements, features, and scan data such as Surface or Profile data) available in a tool are listed in the **Outputs** section of the tool. To enable an output, check its checkbox in the tool's **Outputs** panel.



Enabling the Width measurement. Only the Height measurement is currently enabled, which is displayed in the tool's output section in the Tool Diagram.

Arrays, Batching, and Aggregation

GoPxL supports data structures called arrays to help simplify otherwise complicated applications.

When working with arrays, you can select an array element (or all of the elements) across all arrayed tool outputs to make it easier to focus on a single array element. For more information, see *Displaying and Pinning Outputs* on page 87.

An array is a group of data items bundled into a single structure.

The items, or elements, in the array can be processed by other tools in one of two ways:

- Batching: Tools in batching mode process each array element *separately*. In other words, the tools "loop" through the elements of the array. Typically, tools in batching mode will output an array of outputs, where each element corresponds to the elements in the input. You enable batching by checking the **Enable batching** parameter in a tool's **Inputs** section. For more information, see *Batching* below.
- Aggregating: Aggregating tools combine each element in the array and process it as a *single* piece of data. For more information, see *Aggregating* on page 129.

Most tools support the batching of array elements. Some tools support both batching and aggregating.

Array data can come from various sources:

- Outputs from tools like Surface Blob, Surface Segmentation, and Surface Pattern Matching. For more information, see *Arrays, Batching, and Aggregation* on the previous page.
- G4 and G5 (single- and multi-sensor) and multi-layer output. For more information, see *Arrays, Batching, and Aggregation* on the previous page.
- G2 multi-sensor output. For more information, see *Arrays, Batching, and Aggregation* on the previous page.
- The Script tool. For more information, see *Script Tool* on page 272.
- The Array Create tool. For more information, see *Array Create* on page 141.

For information on working with arrays in the Displayed Outputs panel, see *Displaying and Pinning Outputs* on page 87.

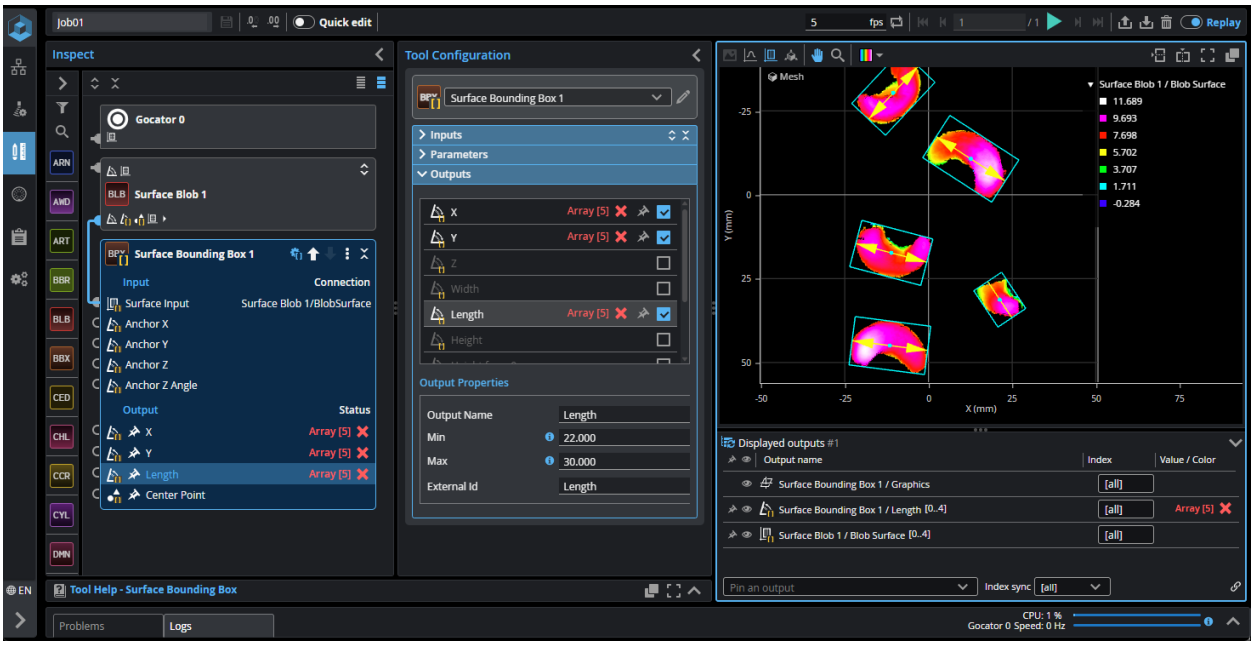
Batching

When a tool is in *batching* mode, it processes each item in the input array *individually*. A tool in batching mode also outputs one or more arrays (such as measurements, Profile or Surface data, geometric features, and so on), where each element of the output arrays corresponds to an element in the input array.

When a tool is in batching mode, there is no limit to the number of elements in the input array (other than any CPU limitations).

Batching is enabled in tools by checking **Enable batching** in the tool's **Inputs** section, or by clicking the Batching icon ( ) in a tool's block in the Tool Diagram panel (see below).

The following shows the Surface Bounding Box tool in batching mode (taking the Surface Blob tool's array of outputs as input). The tool calculates the bounding box for each element in the input array, and returns the enabled measurements as arrays. That is, it returns an X, Y, and Z measurement for each object.



Hovering over batched outputs in the Tool Diagram or in the **Tool Configuration** panel displays a list of the elements in the array.

When at least one result in the array is a fail, the output is considered a fail and the "Array[n]" label is red. Otherwise, it's green. Outputs within the pop-up list are displayed in green or red, depending on whether they represent pass or fail. You can copy the values of the measurement into the clipboard by clicking the copy icon ( ). The format of the copied values is as follows:

```
[24.604,22.805,22.018,21.765,13.099]
```

Aggregating

Some tools can take arrays as an input and combine (or "aggregate") the data in the array elements and treat it as a single input (see below for an example). These tools are called "aggregating tools"; see *Aggregating Tools* below for a list of the aggregating tools. In order for a tool to aggregate an array input, its **Enable Batching** parameter must be disabled. Otherwise, it will process the array elements individually. Note that aggregating tools also take non-array input (such as a single scan of Profile or Surface data), and simply process the data as usual.

Some aggregating tools are limited to a maximum of a 2-element array in their input. Other tools can accept an arbitrary number. See *Aggregating Tools* below to determine the limits of a tool. Note that when a tool is in *batching* mode, there is no limit to the number of elements in the input array (other than any CPU limitations), regardless of the array size limitations when the tools is aggregating data.

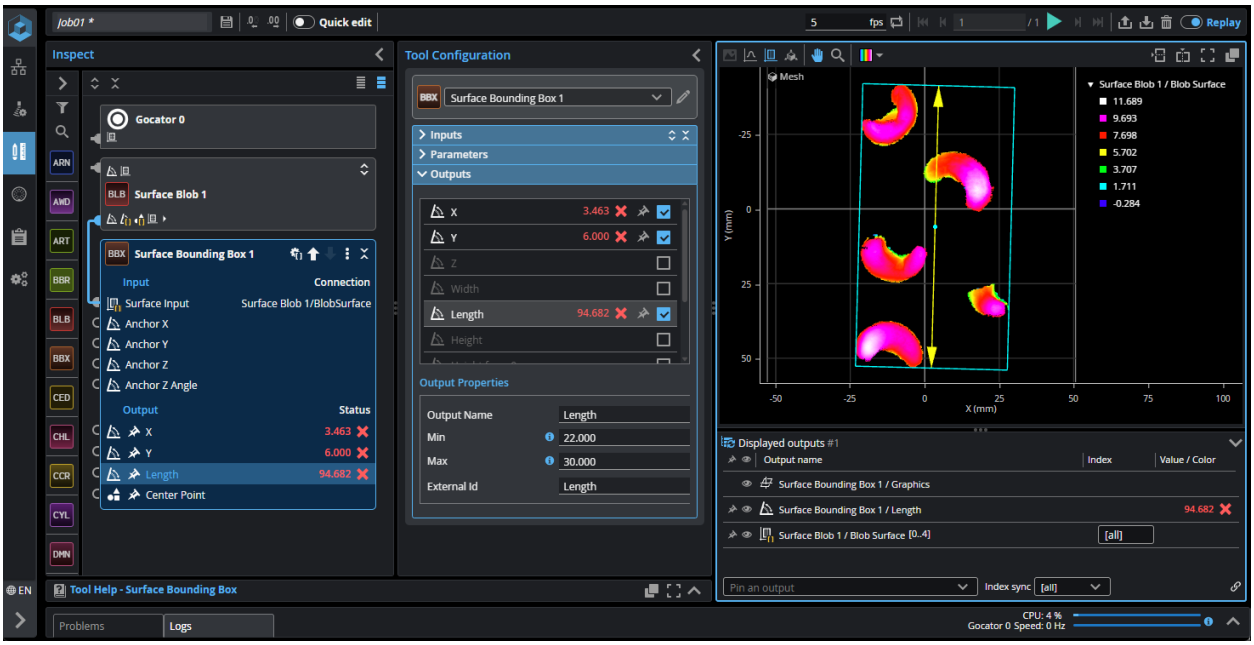
Remember to uncheck **Enable Batching** if you want a tool to aggregate array inputs.

Aggregating Tools

	Number of array elements supported for aggregation	
Tool	**2 elements maximum**	**arbitrary (2 or more elements)**
Profile Circle Radii		✓
Profile Closed Area		✓
Profile Bounding Box		✓
Profile Bridge Value		✓
Profile Circle		✓
Profile Dimension	✓	
Profile Ellipse		✓
Profile Groove	✓	
Profile Intersect	✓	
Profile Line		✓
Profile Line Intersect		✓
Profile Panel	✓	
Profile Position	✓	
Profile Round Corner	✓	
Profile Strip		✓
Surface Bounding Box		✓

	Number of array elements supported for aggregation
Surface Dimension	✓
Surface Ellipse	✓
Surface Flatness	✓
Surface Plane	✓
Surface Position	✓
Surface Section	✓
Surface Sphere	✓
Surface Stick	✓

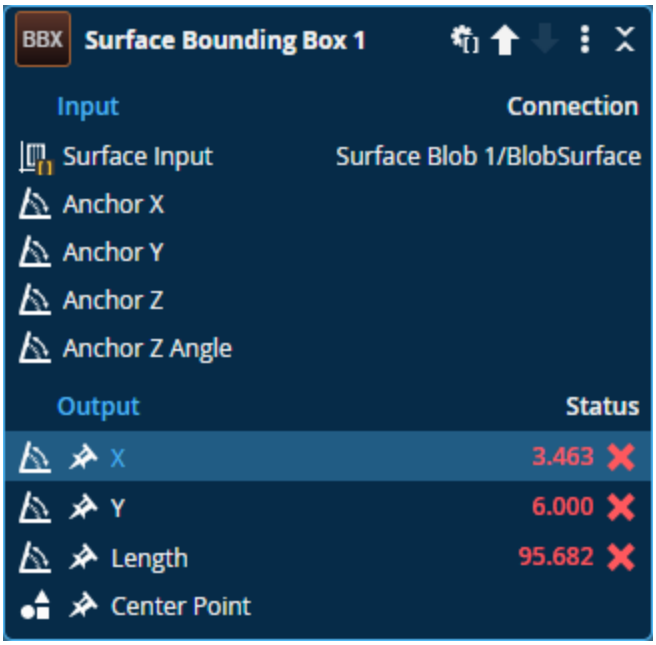
Taking the example from above, if batching in the Surface Bounding Box tool is disabled, it will aggregate the scan data in the input array, and fit the bounding around the scan data of all the elements together.



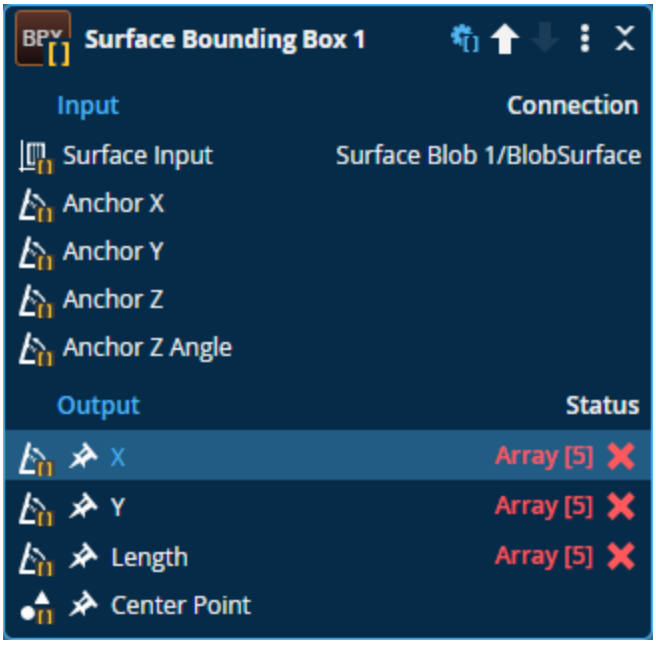
Batch and Array Indicators in GoPxL

GoPxL communicates information related to batching and arrays by adding a "square bracket" badge ("[ ]") to input, output, and tool icons in the web interface.

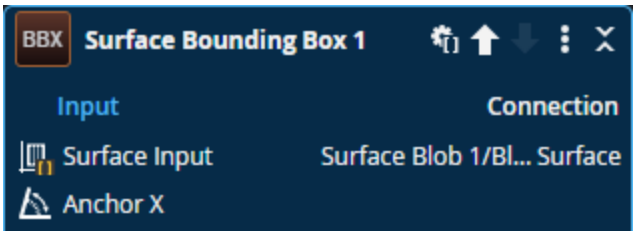
In the Tool Diagram panel, for tools that support batching, a Batching icon ( ) is displayed at the top of the tool block. Use this to toggle batching for that tool. If the tool is not currently selected (that is, you have clicked on it), indicated by a blue border, you must hover the mouse pointer over the top area of the tool block to see whether it supports batching.



When you enable batching on a tool, the Batching toggle icon turns blue, and GoPxL places yellow batching badges on various elements in the web interface. Below, the tool is in batching mode, and its inputs and outputs are therefore all arrays.



For tools that can take array inputs when batching is not enabled, the scan data input (Surface Input) shows a batching badge. Tools like this aggregate array inputs in this state; for more information, see *Aggregating* on page 129.



Regions

Most measurement tools use user-defined regions, or regions of interest, to limit the area in which measurements occur. Unlike reducing the active area, reducing the region of interest does not increase the maximum frame rate of the sensor.

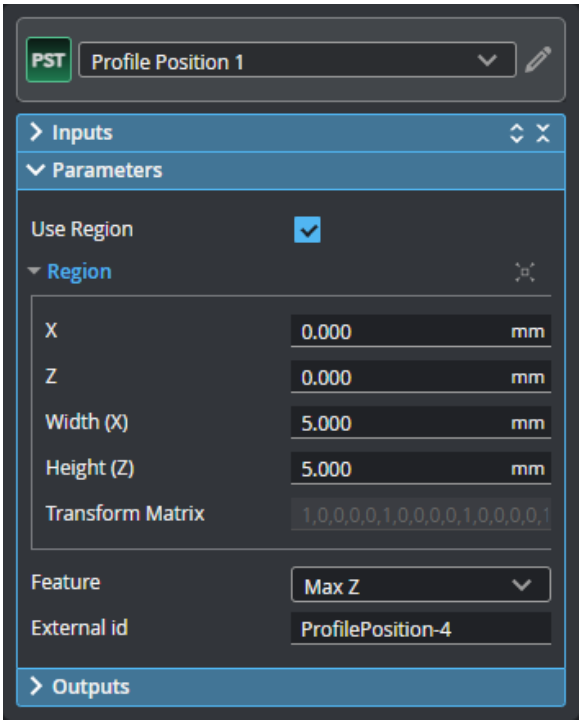
You can use Ctrl-Z and Ctrl-Shift-Z to undo and redo changes to the last changed region's size and position.

Some tools provide "flexible" regions, which in addition to rectangular regions let you create circular and elliptical regions (which can optionally be annular).For information on setting flexible regions, see *Flexible Regions* on the next page.

Other tools are currently limited to standard, rectangular regions.

Standard Regions

Some tools provide regions limited to boxes/rectangles.



The **Transform Matrix** setting is not currently used.

Flexible Regions

Many tools provide "flexible" regions, which let you choose region types such as circle and ellipse, in addition to rectangle regions. For a complete list and information on the related settings, see *Flexible Region Parameters* below.

The following parameters are available in tools that support flexible regions.

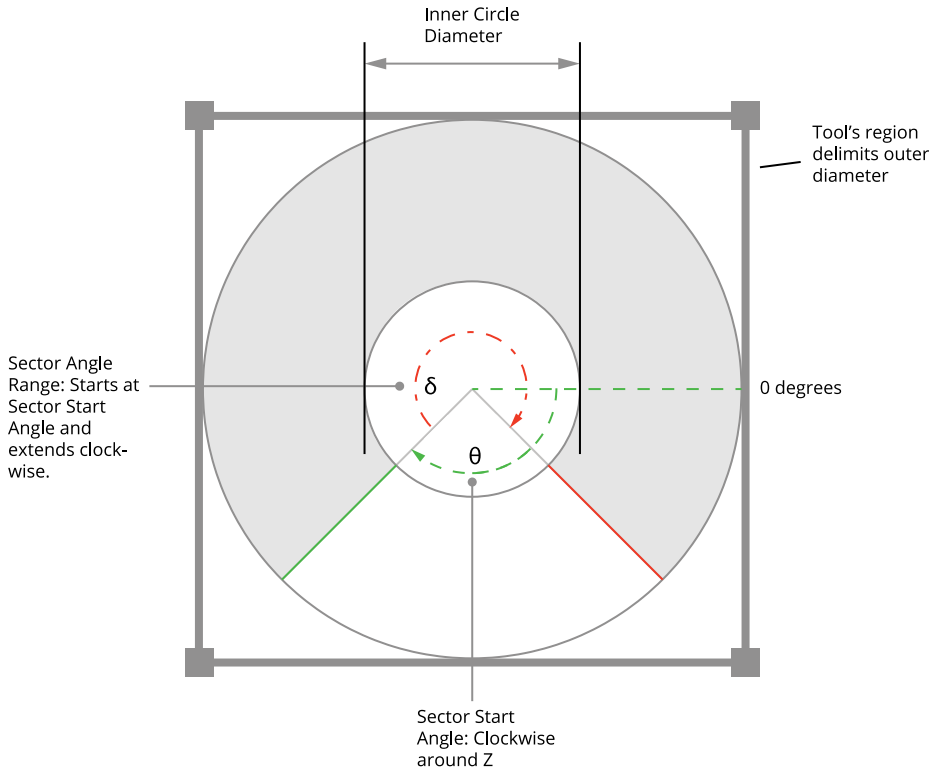
Flexible Region Parameters

Parameter	Description
Mask Mode	One of the following: **Include Data in Region**: The tool uses only the data inside the region or regions. **Exclude Data in Region**: The tool ignores the data inside the region or regions, using only the data outside.
Number of Regions	The number of regions or masks the tool uses to extract data. You can define up to 64 regions. This parameter is not available in some tools. To set a number of regions greater than 16, choose "Custom" in this parameter and set **Custom Region Count**. When you specify more than one region, the regions are initially stacked on top of one another, in the same location.
Custom Region Count	The number of regions (from 17 to 64) when **Number of Regions** is set to "Custom".
Region {n}	An expandable section containing region definitions for each region.
Region Type {n}	The type of each region. Regions can overlap. (For more information on the settings you use with the Circle and Ellipse types, see *Working with Circular and Elliptical Regions* on the next page.) One of the following: **Rectangle** Extracts a rectangular region from the acquired data. **Circle** Extracts a circular region from the acquired data, constrained by a square region. Set the region's inner circle using the **Inner Diameter** parameter to 0 to extract circular data. Set it to a non-zero to extract annular data. Use the **Sector Start Angle** and **Sector Angle Range** parameters to extract a partial circular or elliptical region. (See below.) **Ellipse** Extracts an elliptical region from the acquired data, constrained by a square or rectangular region.

Parameter	Description
	Set the region's inner ellipse (inner cyan ellipse below) using the **Inner Major Axis** and **Inner Minor Axis** parameters to extract annular data. Use the **Sector Start Angle** and **Sector Angle Range** settings to extract a partial circular or elliptical region. (See below.) **Image** TBD
X, Y, Z Width (X), Length (Y), Height (Z) Z Angle	These parameters define the size, location (center of the region), and orientation of the region. Not available when **Region Type {n}** is set to the following: **Image**.
Inner Diameter	Only available when **Region Type {n}** is set to **Circle**. Defines the diameter of the inner circle. Set this parameter to a value greater than 0 to extract a ring of data. Set this parameter to 0 to extract a circle of data.
Inner Major Axis Inner Minor Axis	Only available when **Region Type {n}** is set to **Ellipse**. These parameters define the major and minor axes of the inner ellipse, respectively Set this parameter to a value greater than 0 to extract a ring of data. Set this parameter to 0 to extract an elliptical disk of data.
Sector Start Angle Sector Angle Range	Only available when **Region Type {n}** is set to **Circle** or **Ellipse** Use these parameters together to extract a partial ring of data. **Sector Start Angle** controls the starting angle of the data, whereas **Sector Angle Range** controls the length of the arc. Note that the angles and ranges in these parameters are measured clockwise around Z, where 0 degrees is along the positive X axis. For more information on how these settings work together, see *Working with Circular and Elliptical Regions* below.
High Threshold Low Threshold	The high and low thresholds the tool uses in combination with the intensity mask. Only available when **Region Type {n}** is set to the following: Image.

Working with Circular and Elliptical Regions

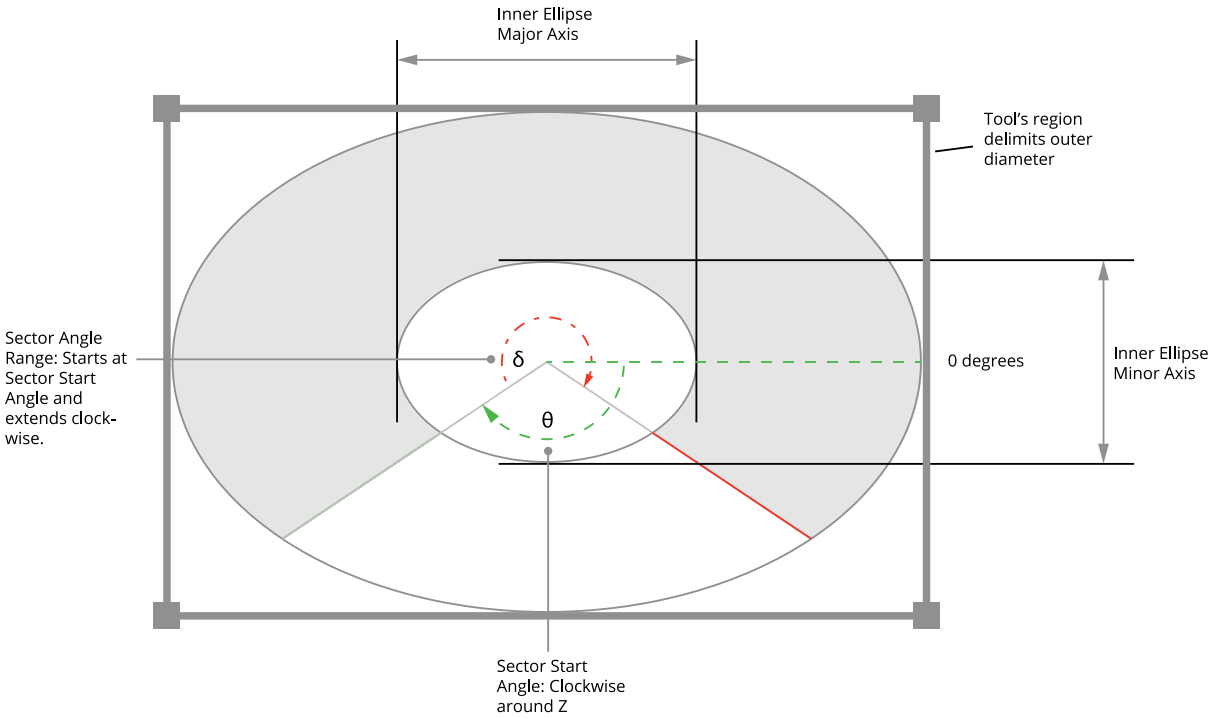
When you set a region's type to Circle or Ellipse, the tool displays several additional settings that work together to define the region. **Sector Start Angle** and **Sector Angle Range** work together to define the start and end of a partial circular/elliptical region (solid or annular). A region will be annular if **Inner Circle Diameter** is non-zero. Note that the "length" of the partial region extends *from* the start angle. In the following illustration, the start angle (θ) is 135 degrees relative to the 0-degree point indicated below, and the region extends 270 degrees (δ) from that, clockwise around Z.



Sector Start Angle starts at the 0-degree point around Z.

Note that the angles defining a partial circular/elliptical region are relative to the *region*, and not the sensor's coordinate system. So a region rotated 30 degrees using its **Z Angle** setting rotates the start angle and angle range by 30 degrees.

When you set a region type to Ellipse, instead of the inner circle diameter, you must set the major and minor axes of the inner ellipse.



Region Rotation

The regions of some tools can be rotated by setting the region's **Z Angle** (or **Angle** for some models) to better accommodate angled features on a target. By rotating the measurement region, data not related to the feature can often be excluded, improving accuracy of measurements. You can modify the size and position of a rotated region.

You can also rotate a region using a mouse in the data viewer. To do this, hover the mouse over the region's grabber and click and hold the left mouse button. Then rotate the region as needed. Finally, resize the region.

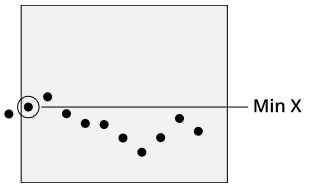
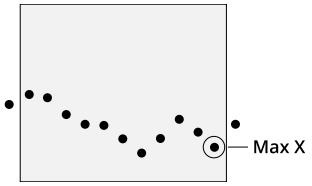
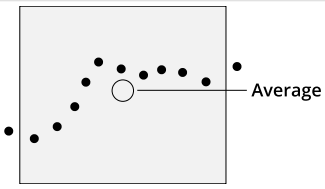
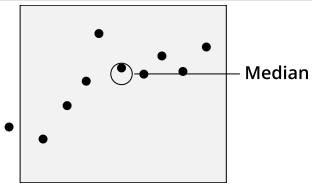
Feature Points

Dimensional and positional measurements detect *feature points* found within the defined region (whether it's a measurement region, feature region, reference region, and so on) and then compare measurement values taken at the selected point with minimum and maximum thresholds to produce a pass / fail decision. Feature points are selected in one or more **Feature** parameters in a tool's **Parameters** section, and are used for all of the tool's measurements.

It is important to note that the feature points described here are distinct from the geometric features generated by some tools, and which can be taken as inputs by other tools. Feature points are simply the location in the data at which a measurement tool performs a measurement. Geometric features are a data type produced as output or taken as input by a tool. (For more information on geometric features, see *Geometric Features* on the next page.)

The following types of points can be identified in a measurement region. The availability of a feature point type is tool-dependent.

	The following are represented as profiles for clarity. Many Surface tools use feature points.

Point Type	Examples
Min X Finds the point with the minimum X value in the region of interest.	
Max X Finds the point with the maximum X value in the region of interest.	
Average Determines the average location of points in the region of interest.	
Median Determines the median location of points in the region of interest.	
Max Intensity **Min Intensity** Determines the location of the pixel with the maximum and minimum intensity in the region of interest.	

Geometric Features

These features are called *geometric features*. Feature tools use these entities to produce measurements based on more complex geometry. (For more information on Feature tools, see *Feature Tools* on page 148.)

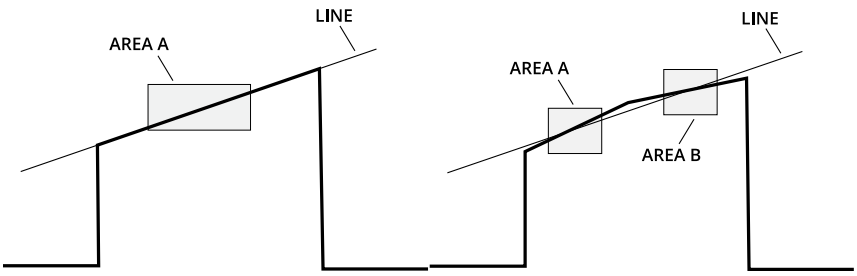
The measurement tools can currently generate the following kinds of geometric features:

Points: A 2D or 3D point (device dependent). Can be used for point-to-point or point-to-line measurements.

Lines: A straight line that is infinitely long. Useful for locating the orientation of an enclosure or part, or to intersect with another line to form a reference point that another tool can take as input.

Fit Lines

Some measurements involve estimating lines in order to measure angles or intersection points. A fit line can be calculated using data from either one or two fit areas.



A line can be defined using one or two areas. Two areas can be used to bypass discontinuity in a line segment.

Decisions

Results from a measurement can be compared against minimum and maximum thresholds you set in a tool's **Outputs** panel under **Output Properties** to generate *pass* / *fail* decisions. The decision state is *pass* if a measurement value is within the minimum / maximum range; the range is inclusive (a measurement value equal to a Min or Max threshold value is a pass). Output values that pass are green in the interface. Output values that fail are red in the interface.

Note that measurement values are output with six decimal places. However, you can configure the GoPxL interface to hide some decimal places, using the "Decrease precision" and "Increase precision" buttons (and , respectively), from zero to six decimal places. Keep this in mind when configuring thresholds.

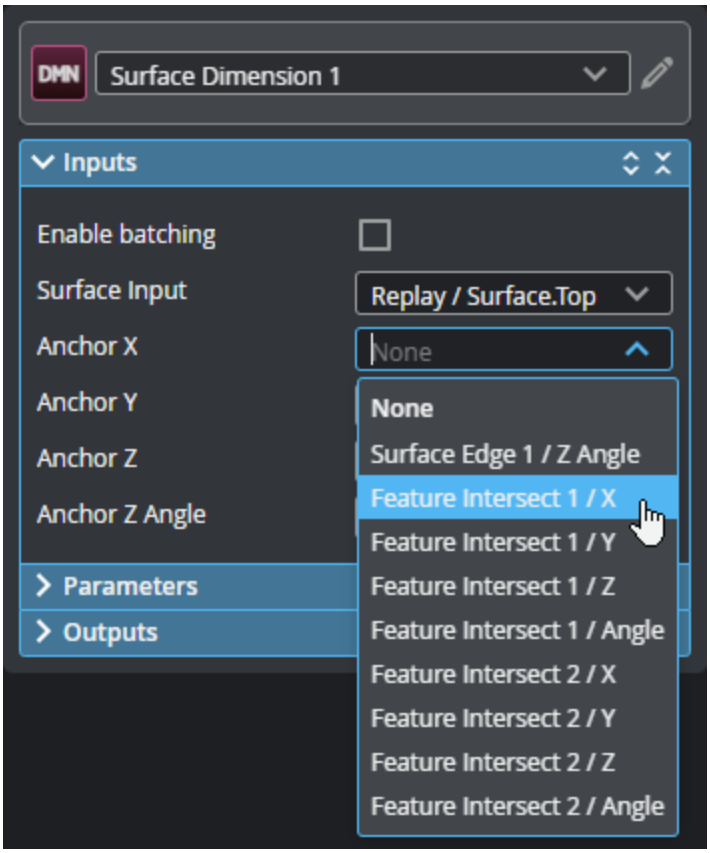
Decisions (and measurements) can be sent to external programs and devices. For more information on sending values and decisions as output, see *Communicate - PLC, Digital, GDP, HMI* on page 309.

Measurement Anchoring

The position of targets can change from part to part in one or both of the following ways. When the position and angle variation between parts is minor, you can anchor one tool to one or more measurements from one or more other tools to compensate for these minor shifts. As a result, GoPxL can correctly place the anchored tool's measurement regions on each part. This increases the repeatability and accuracy of measurements.

> For cases where movement from part to part is more drastic, you can often use the Image Pattern Matching tool to compensate. For more information, see *Image Pattern* on page 244.

You set a tool's anchors in its **Inputs** section, by setting the desired anchor (such as **Anchor X** or **Anchor Y**) to the appropriate measurement from another tool:



When you set any of a tool's anchor sources, an offset is calculated between the anchored tool and the anchor source. This offset is used for each frame of scanned data: the anchored tool's measurement region is placed in relation to the anchor source, at the calculated offset.

You can combine the positional anchors with an angle anchor (such as a Z Angle measurement) for optimum measurement placement.

	If angle anchoring is used with both X and Y anchoring, the X and Y anchors should come from the same tool.

	If angle anchoring is used without X or Y anchoring, the tool's measurement region rotates around its *center*. If only one of X or Y is used, the region is rotated around its center and then shifted by the X or Y offset.

Array Tools

Array tools let you create and work with arrays, or extract individual elements from an array.

> Arrays can contain any type of data supported in GoPxL (such as Surface, Profile, or Image data, or measurements). For more information on this and on arrays in general, see *Arrays, Batching, and Aggregation* on page 126.

The following Array tools are available:

- Array Decision: Takes an *array* of *measurements* and lets you apply different minimum/maximum thresholds to each measurement in the array.

  The tool is typically used in electronics pin inspection applications.

  For more information on this tool, see *Array Decision* on page 142.

- Array Index: Takes an *array* of data as input and extracts one or more pieces of data at the user-provided index. You can then perform measurements on the individual outputs, or perform measurements using more than one piece of data, such as measuring the thickness represented by two profiles or surfaces.

  Usage examples include separating the multiple layers produced by a confocal sensor into individual profiles or surfaces.

  For more information on this tool, see *Array Index* on page 144.

- Array Create: Takes *individual* inputs and creates an array containing those pieces of data as elements. For more information on this tool, see *Array Create* on the next page.

Array Create

The Array Create tool combines individual pieces of data (Profile, Surface, measurements, and so on) into an array. For example, a number of Surface scans could be combined into an array made up of those pieces of Surface data. Compatible measurement and processing tools can then take the array as input and perform batch operations on the elements in the array, such as measurements on each element's data or processing each element's data. For more information, see *Arrays, Batching, and Aggregation* on page 126.

For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Inputs

Inputs

Name	Description
Input {n}	An input that the tool will add to the output array. The number of inputs depends on the **Item Count** parameter. All inputs must be of the same type.

Parameters

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Item Count	The number of items expected in the array.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

The type of the output depends on the type of the inputs.

Array Decision

The Array Decision tool takes an array of measurements as input and applies individual minimum / maximum thresholds to one or more of the array elements. The tool returns results related to fails (number of fails, index and value of max/min fail, as well as arrays of failed measurements and their indices).

A common use case of Array Decision is to make pass/fail decisions for electronics pin inspection applications.

For general information on the Array tools, see *Array Tools* on page 140.

For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Inputs

Inputs

Name	Description
Measurement	The array of measurements the tool takes as input.

Parameters

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Measurement Count	The maximum number of array elements expected in the input array. The tool displays Min{n} and Max{n} fields for each element.
Min {n} Max {n}	The minimum and maximum threshold for measurement element {n} in the input array of measurements.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

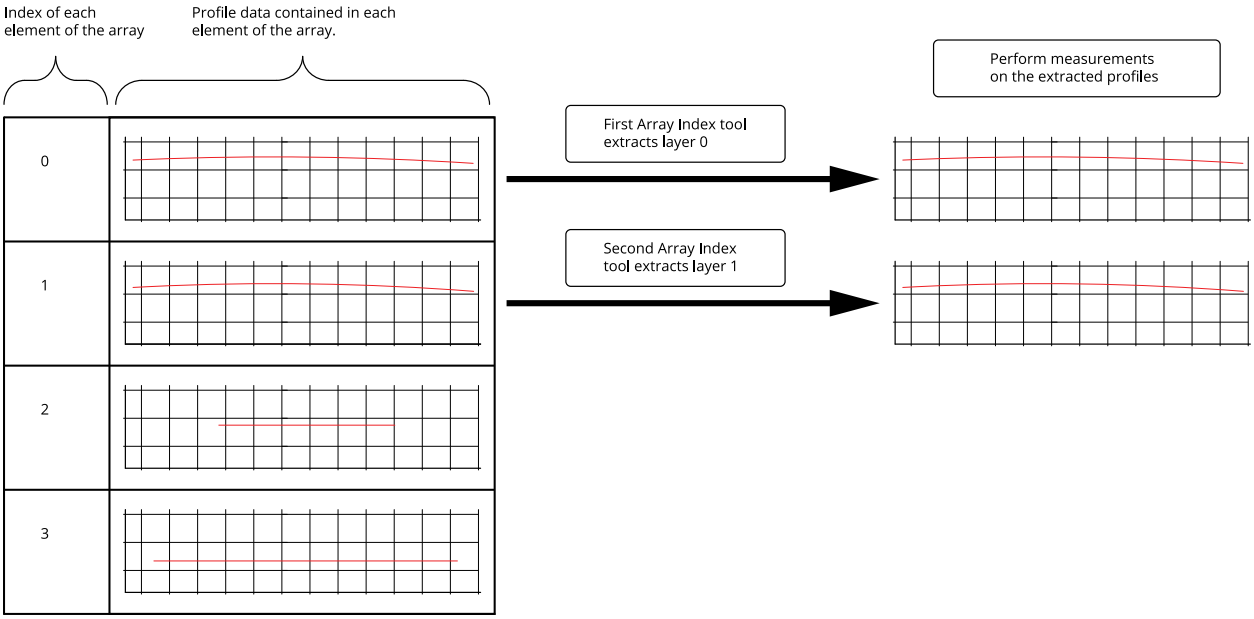
Measurements

Measurement
Fail Count The number of array values that are outside their Min/Max thresholds.
Min Fail Index The index of the array element with the greatest difference with respect to its Min threshold.
Min Fail Value The value of the array element at Min Fail Index.
Max Fail Index The index of the array element with the greatest difference with respect to its Max threshold.

Measurement
Max Fail Value The value of the array element at Max Fail Index.
Failed Indices An array of the indices of the failed measurements.
Failed Measurements An array of the failed measurement values.

Array Index

The Array Index tool extracts a one or more pieces of data (such as a profile or a surface) from the array it takes as input starting from the position (the index) you specify.



For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Inputs

Inputs

Name	Description
Array Input	The data the tool applies measurements to or processes.

Parameters

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Index	The position (the index) in the array at which the tool extracts and outputs. If **Count** is set to a value greater than 1, the index is the starting point for array element outputs.
Count	The number of individual outputs the tool produces, starting at the provided index (see above). This setting makes sending multiple outputs over protocols easier.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

If **Count** is set to a value greater than 1, the tool produces multiple outputs.

The type of the output depends on the type of the input array.

Conditional Logic Tool

The Conditional Logic tool sends input data to different outputs based on a conditional measurement value. The tool can take multiple inputs. This lets you create complex processing and inspection tool chains that depend on the conditions you set. By limiting the tool's output to a subset of the input it receives, for example, job runtime can be reduced.

For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Inputs

You configure the tool's inputs in the expandable **Inputs** section.

Inputs

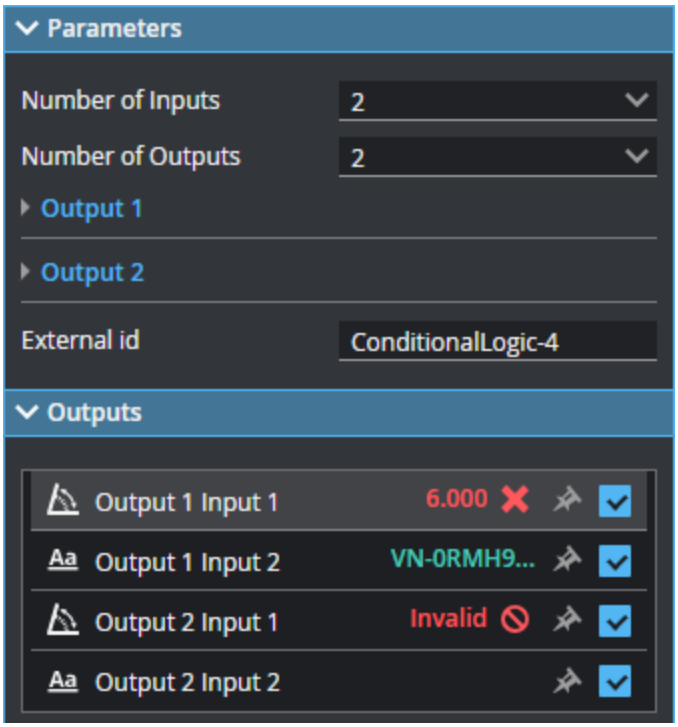
Name	Description
Input {n}	An input the tool routes to an output, based on the conditions (an operator and a value) set in the **Parameters** section. This tool can optionally take an array as input. For more information, see Arrays, Batching, and Aggregation.
Conditional Measurement	The measurement whose value the tool compares to a value set in **Parameters** > **Output {n}**, using the operator selected in the **Output {n}** section.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Name	Description
Number of Inputs	The number of data inputs the tool accepts. Set the inputs in the **Inputs** section. If the tool takes more than one input, the tool outputs all possible combinations.

Name	Description
	
Number of Outputs	The number of data outputs the tool potentially produces. For each output, configure Operator and Value in the expandable **Output {n}** section.
Output {n}	The relational operator (<, >, ≤, ≥, =, ≠) and value the tool uses to compare with the value of the conditional measurement. Set **Operator** and **Value**.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

Data

Type	Description
Output {n} Output {n} Input {n}	Data satisfying the comparison configured in **Output {n}** in the **Parameters** section. If the tool takes more than one input, the tool outputs all possible combinations, and names them using "Output {n} Input {n}" as the convention.

Feature Tools

The following sections describe the Feature tools.

Feature tools generate or measure geometric features; for more information on geometric features, see *Geometric Features* on page 137. The tools let you work with more complex geometry, letting you implement applications more quickly by reducing dependence on scripts and GDK tools to perform these kinds of measurements.

The "Create" tools (such as Feature Circle Create) generate geometric features from other tools or using values you provide.

The Feature Dimension tool lets you take measurements using geometric features.

The Feature Robot Pose tool is for use with robots.

Feature Circle Create

The Feature Circle Create tool lets you generate a Circle geometric features from geometric features generated by other tools. You can also generate a circle from an arbitrary position and radius. The following outputs are available (for more information, see *Output Types* below):

- Constant circle
- Circle from points
- Circle from point and plane
- Circle tangent to two lines

For more information on geometric features, see *Geometric Features* on page 137.

Inputs

You configure the tool's inputs in the expandable **Inputs** section.

This tool can take an array as input. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.

The tool's geometric feature inputs depend on the output type you choose in the **Output** parameter. For information on the required inputs and parameters of each output type, see *Output Types* below.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Parameter	Description
Output	The type of output the tool generates. Switching between the options changes the input types displayed in Input section and the parameters displayed in the Parameters section. For information on the output types and the required inputs and related parameters, see *Output Types* below.
Project To 2D	If checked, generates a circle in the XZ plane for Profile mode and in the XY plane for Surface mode. Otherwise, generates a 3D circle. Not shown when **Output** is set to “Circle from point and plane”.
Show Detail	Toggles the display of the input geometric features in the data viewer.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Output Types

The following sections describe the output types available in the tool's **Output** parameter, in addition to the required inputs and parameters.

Constant Circle

The **Constant circle** output type is useful if scan data from frame to frame is reliably fixed and you want to measure from an arbitrary circle to another feature. When you choose this output type, the tool displays the following parameters you can manually fill in to create a constant circle.

Disable **Project To 2D** to input an arbitrary normal vector (using the **Normal** parameters) to get a tilted 3D circle.

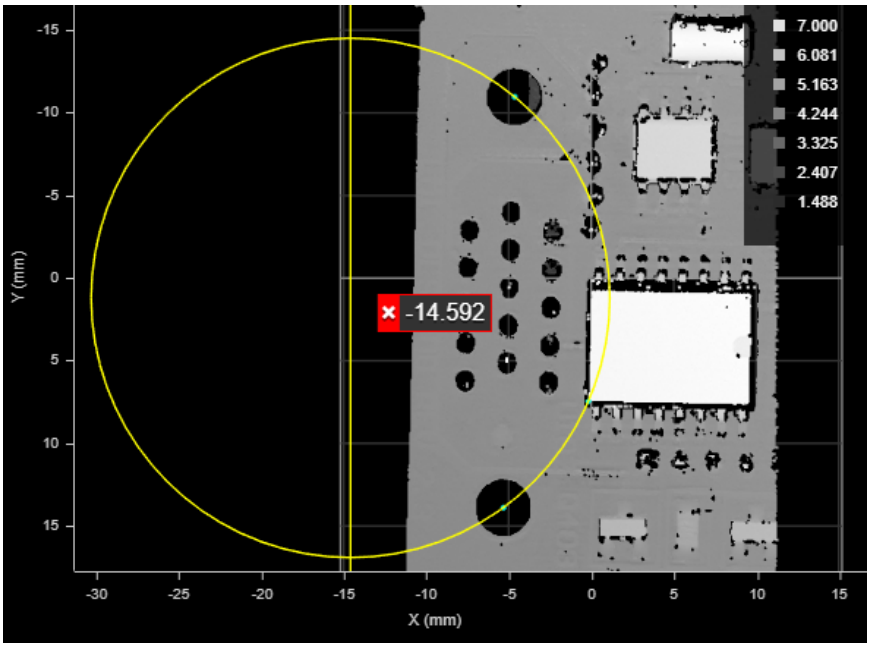
Constant Circle Output Type - Inputs and Parameters

Inputs	Type-Related Parameters
Input The Surface or Profile input.	**Center X**, **Center Y**, **Center Z** The X, Y, and Z position, respectively, of the center of the constant circle.
Anchor X, **Anchor Y**, **Anchor Z** The X, Y, and Z offsets the tool will apply to the generated circle.	**Normal X**, **Normal Y**, **Normal Z** The X, Y, and Z for the normal vector. Only displayed when **Project To 2D** is disabled.
	Radius The radius of the constant circle.

Circle from Points

The **Circle from points** output type takes three point geometric features generated by another tool and fits a circle to those points. The circle is always on the XY plane.

When **Project To 2D** is enabled, the circle is projected to the XY plane. When **Project To 2D** is disabled, circle is tilted to fit the plane the three point inputs create.



Circle generated from the center points of the two holes and the corner of the chip (cyan points).
(The corner is the intersect point resulting from the Feature Intersect tool,
taking the left vertical and lower horizontal line edges of the chip as input.)

The X, Y, and Z measurements return the center of the circle. The X, Y, and Z Angle measurements return arbitrary values.

Circle from points Output Type - Inputs and Parameters

Inputs	Type-Related Parameters
Input The Surface or Profile input.	None.
Anchor X, **Anchor Y**, **Anchor Z** The X, Y, and Z offsets the tool will apply to the generated circle.	
Point 1, **Point 2**, **Point 3** The three Point geometric features the tool uses as the center of the circle.	

Circle from point and plane

The **Circle from point and plane** output type generates a circle using Point and Plane geometric features and a user-defined radius. The circle is on the Plane received as input.

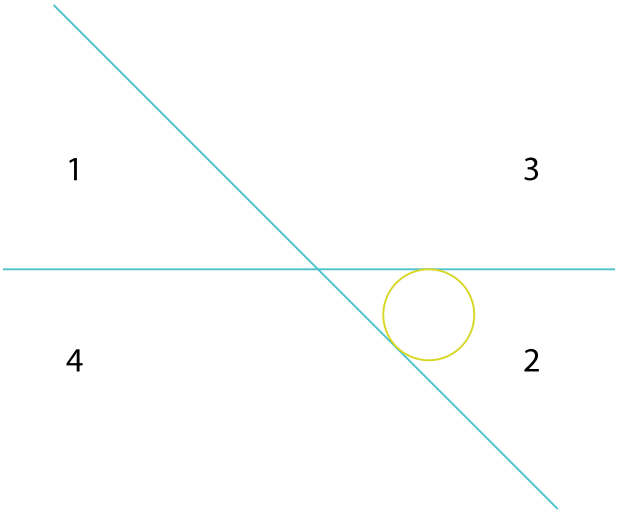
Circle from point and plane Output Type - Inputs and Parameters

Inputs	Type-Related Parameters
Input The Surface or Profile input.	**Radius** The radius of the circle.
Anchor X, **Anchor Y**, **Anchor Z** The X, Y, and Z offsets the tool will apply to the generated circle.	
Point The Point geometric features the tool uses as the center of the circle.	
Plane The Point geometric features the tool uses as the center of the circle.	

Circle tangent to two lines

The **Circle tangent to two lines** takes two Line geometric features and an optional Plane geometric feature to create a circle that is tangent to the two provided lines. The circle is on the Plane received as input.

Circle from point and plane Output Type - Inputs and Parameters

Inputs	Type-Related Parameters
Input The Surface or Profile input.	**From Acute Angle** Toggles between using the acute angle between the lines (enabled) or the obtuse angle (disabled).
Anchor X, **Anchor Y**, **Anchor Z** The X, Y, and Z offsets the tool will apply to the generated circle.	**Circle Selection** Selects whether the tool fits the circle between the upper or lower angle.
Point The Point geometric features the tool uses as the center of the circle.	**Circle Radius** The radius of the circle.
Plane The Point geometric features the tool uses as the center of the circle.	The tool uses **From Acute Angle** and **Circle Selection** together to determine where to fit the circle. For example, given these two lines, the tool fits the circle between the line segments as described below:  1: **From Acute Angle** enabled, "Upper circle" selection. 2: **From Acute Angle** enabled, "Lower circle" selection. (Circle fitted here.) 3: **From Acute Angle** disabled (uses obtuse), "Upper circle" selection. 4: **From Acute Angle** disabled (uses obtuse), "Lower circle" selection.

Outputs

Most tools provide measurements, geometric features, or data as outputs.

Outputs section with a measurement expanded to show user-configurable decision min/max fields and an external ID

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

For information specific to the different output types, see *Output Types* on page 149.

Measurements

Measurement
Center X, **Center Y**, **Center Z** The X, Y, and Z positions of the center of the circle.
Radius The radius of the circle.
Z Offset The Z position of the intersection between the circle plane and the Z axis.
Distance The distance from the origin to the circle plane.
Normal X, **Normal Y**, **Normal Z** The X, Y, and Z normal vectores of the circle.

Features

Type	**Description**
Circle	The generated Circle geometric feature. For more information, see *Output Types* on page 149.

Feature Dimension

See *Adding a Tool* on page 118 for instructions on how to add measurement tools.

> In the following measurement descriptions, the first geometric feature is set in the **Point** drop-down. The second geometric feature is set in the **Reference Feature** drop-down.

> When **Reference Feature** is set to a feature other than a point, such as a circle or a line, measurements are between the point in **Point** and the *nearest point* on the reference feature (for example, the nearest point on a circle).

Inputs

You configure the tool's inputs in the expandable **Inputs** section.

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Input	The data the tool applies measurements to.
Point	A point geometric feature generated by another tool.
Reference Feature	A geometric feature generated by another tool. Dimensional measurements are calculated *from* the reference feature *to* the point in the **Point** input.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Parameter	Description
Width Absolute Length Absolute Height Absolute	The tools returns measurements as absolute values.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

Most tools provide measurements, geometric features, or data as outputs.

Outputs section with a measurement expanded to show user-configurable decision min/max fields and an external ID

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

Measurement
Width *Point-point*: The difference on the X axis between the points. *Point-line*: The difference on the X axis between the point and a point on the line. For profiles, the point on the line is at the same Z position as the first point.
Length *Point-point*: The difference on the Y axis between the points. *Point-line*: The difference on the Y axis between the point and, for profiles, the nearest point on the line; currently, always zero.
Height *Point-point*: The difference on the Z axis between the points. *Point-line*: The difference on the Z axis between the point and, for profiles, a point on the line at the same X position as the first point.
Distance *Point-point*: The direct, Euclidean distance between two point geometric features. *Point-line*: The direct, Euclidean distance between a point and the nearest point on the line.
Plane Distance *Point-point*: The distance between two point geometric features. For profile data, the points are projected onto the XZ plane (always the same as the Distance measurement). *Point-line*: The distance between a point and a line. For profile data, projected onto the XZ plane (always the same as the Distance measurement).

Feature Intersect

The Feature Intersect tool returns the intersection of a line or plane geometric feature and a second line or plane geometric feature. For line-line intersections, the lines are projected onto the Z = reference Z line plane for features extracted from a surface, and the intersection of the lines projected onto the Y = 0 plane for features extracted from a profile. The angle measurement between the two lines is also returned.

The Feature Intersect tool can also generate a point geometric feature representing the point of intersection of the lines that the Feature Dimension tool can use in measurements.

See *Adding a Tool* on page 118 for instructions on how to add measurement tools.

Inputs

You configure the tool's inputs in the expandable **Inputs** section.

The tool's inputs depend on the intersection type you choose in the **Intersection Type** parameter. For information on the required inputs and parameters of each output type, see *Feature Intersect* above.

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Input	The data the tool applies measurements to or processes.
Line, Circle, or Plane	The geometric features the tool uses to calculate an intersect. A combination of Line, Circle, or Plane geometric features, depending on the **Intersection Type** parameter.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Parameter	Description
Intersection Type	One of the following: • Line and Line • Line and Plane • Plane and Plane • Line and Circle The intersection type determines which inputs are available in the **Inputs**.
Intersection Option	When **Intersection Type** is Line and Circle, provide the location of the option (intersection on the circle with the lowest versus the highest X position, and so on).
Project to 2D	Only shows when Intersection Type is “Line and Line”. If checked, the intersection angle is projected to the XZ plane for Profile mode and XY plane for Surface mode.

Parameter	Description
	Otherwise, outputs the 3D angle.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

Outputs section with a measurement expanded to show user-configurable decision min/max fields and an external ID

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

Measurements

Measurement
X *Line-Line*: The X position of the intersect point between the lines. *Line-Plane*: The X position of the intersect point between the line and the plane. *Plane-Plane*: The X position of the center of the line intersecting the planes.
Y *Line-Line*: The Y position of the intersect point between the lines. *Line-Plane*: The Y position of the intersect point between the line and the plane. *Plane-Plane*: The Y position of the center of the line intersecting the planes.
Z *Line-Line*: The Z position of the intersect point between the lines. *Line-Plane*: The Z position of the intersect point between the line and the plane. *Plane-Plane*: The Z position of the center of the line intersecting the planes.
Angle or **Projected Angle** (*Line-Line*: The angle between the lines, as measured from the line selected in **Reference Feature** to the line selected in **Line**. If **Project To 2D** is enabled, the output is Projected Angle. *Line-Plane*:The angle between the line and the perpendicular projection of the line onto the plane, as measured from the plane geometric feature selected in **Reference Feature** to the line selected in **Line**. *Plane-Plane*:The angle between the two planes, as measured from the plane geometric features selected in **Feature 1** and **Feature 2**.

Features

Type	Description
Intersect Point	The intersect point of the two features.
Intersect Line	The line representing the intersection of two planes.

Feature Line Create

The Feature Line Create tool lets you generate a Line geometric features from geometric features generated by other tools. You can also generate a line from arbitrary positions you provide. The following outputs are available (for more information, see *Output Types* below):

- Line
- Constant line
- Line from two points
- Perpendicular line from point and line
- Perpendicular line from point and plane
- Parallel line from point and line
- Intersect line of two planes
- Projected line on plane
- Bisect projected lines on plane
- Rotate line around point

For more information on geometric features, see *Geometric Features* on page 137.

Inputs

You configure the tool's inputs in the expandable **Inputs** section.

This tool can take an array as input. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.

The tool's inputs depend on the output type you choose in the **Output** parameter. For information on the required inputs and parameters of each output type, see *Output Types* below.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Parameter	Description
Output	The type of output the tool generates. Switching between the options changes the input types displayed in Input section and the parameters displayed in the Parameters section. For information on the output types and the related parameters, see *Output Types* below.
Show Detail	Toggles the display of the input geometric features in the data viewer.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Output Types

The following sections describe the output types available in the tool's **Output** parameter, in addition to the required inputs and parameters.

Line

The **Line** output type takes a Line geometric feature as input. This output is useful if the tool takes a feature generated by another Feature Create tool as input, on which you want to perform measurements in a second Feature Create tool. This tool can also be useful if you have developed GDK tools that only generate geometric features (no measurements): you can use this tool to extract those measurements.

The X, Y, and Z measurements return the midpoint of the line. The Z Angle measurement returns the angle of the line around the Z axis. The X angle and Y angle measurements return arbitrary values.

Line Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input.	None.
Line The Line geometric feature the tool uses to create another Line geometric feature.	

Constant Line

The **Constant line** output type displays parameters you can manually fill in to create a geometric feature. This output type is useful if the scan data from frame to frame is reliably fixed and you want to measure from an arbitrary line to another feature.

Line Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input.	**Point 1 X**, **Point 1 Y**, **Point 1 Z** The X, Y, and Z position, respectively, of the first end point of the constant line. **Point 2 X**, **Point 2 Y**, **Point 2 Z** The X, Y, and Z position, respectively, of the second end point of the constant line.

Line from Offset

The **Line from offset** output type takes a Line geometric features as input.

The X, Y, and Z measurements return the midpoint of the line.

Line from offset Output Type - Inputs and Parameters

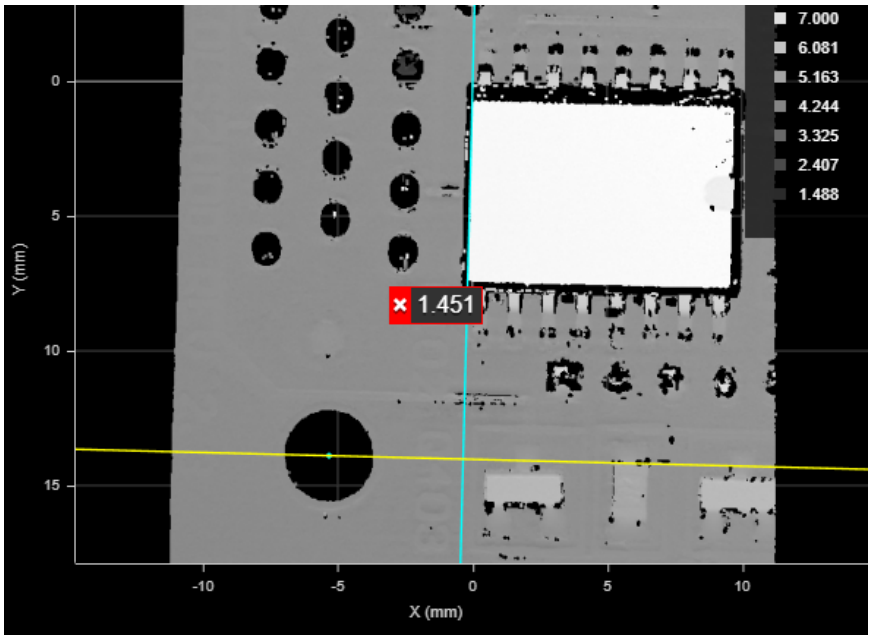
Inputs	Parameters
Input The Surface or Profile input.	For Surface data: **X Distance** / **Y Distance**

Inputs	Parameters
Line The Line geometric feature the tool uses to create another Line geometric feature.	Offsets the input line by the provided X and Y distance. For Profile data: **Offset Direction** The offset direction. Either Positive X or Positive Z. **Offset Distance** The distance along the axis specified in **Offset Direction**.

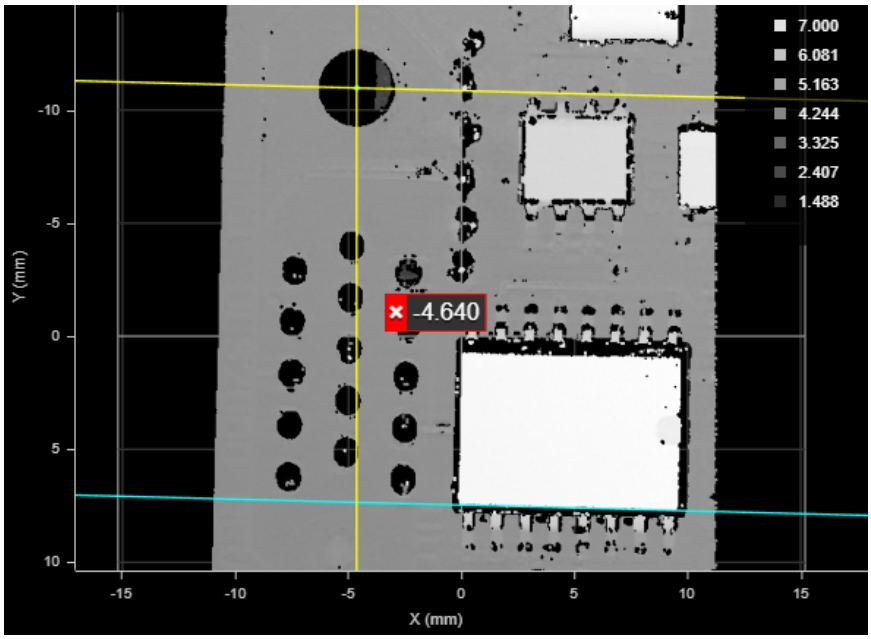
Perpendicular or Parallel Line from Point and Line

The **Perpendicular line from point and line** and **Parallel line from point and line** output types take a Point and a Line geometric feature as input to create another line.

In the following, the tool generates a roughly vertical line (yellow) perpendicular to the input line (cyan line along the left edge of the large integrated circuit), passing through the input point (cyan dot at the center of the hole).



In the following, the tool generates a roughly horizontal line (yellow) parallel to the input line (cyan line along the bottom edge of the large integrated circuit), passing through the input point (cyan dot at the center of the hole).



For both of these types of line output, the X, Y, and Z measurements return the position of the point.

For perpendicular line output, the X, Y, and Z angle measurements return the angles of the line.

For parallel line output, the Z angle measurement returns the angle of the line; the X and Y angle measurements both return arbitrary values.

Line from Point and Line Output Types - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input.	None.
Point The Point geometric feature the tool uses to create the line.	
Line The Line geometric feature the tool uses to create the line.	

Perpendicular Line from Point to Plane

The **Perpendicular line from point to plane** output type generates a perpendicular line from a point up to a plane.

Perpendicular line from point to plane Output Types - Inputs and Parameters

Inputs	Parameters
Input	None.

Inputs	Parameters
The Surface or Profile input. **Point** The Point geometric feature the tool uses to create the line. **Plane** The Plane geometric feature the tool uses to create the line.	

Intersect Line of Two Planes

The **Intersect line of two planes** output type takes two plane geometric features as input and creates a line at their intersection.

The X, Y, and Z measurements return the midpoint. The X, Y, and Z Angle measurements return the angle of the line.

Intersect line of two planes Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input. **Plane 1**, **Plane 2** The two Plane geometric feature the tool uses to create the line.	None.

Projected Line on Plane

The **Projected line on a plane** output type takes a Line and a Plane geometric feature to create a line projected onto a plane.

Projected line on a plane Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input. **Line** The Line geometric feature the tool uses to create the line. **Plane** The Plane geometric feature the tool uses to create the line.	**Projection Direction** One of the following: **Perpendicular to Plane** or **Along Z Axis**.

Projected Center Line from Two Lines and a Plane

The **Projected Center Line from Two Lines and a Plane** output type takes creates a two Line and a Plane geometric feature as input and creates a line.

Bisect projected lines on plane Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input.	None.
Line The Line geometric feature the tool uses to create the line.	
Plane The Plane geometric feature the tool uses to create the line.	

Outputs

Most tools provide measurements, geometric features, or data as outputs.

Outputs section with a measurement expanded to show user-configurable decision min/max fields and an external ID

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

For information specific to the different output types, see *Output Types* on page 158.

Measurements

Measurement
X, **Y**, **Z** The X, Y, and Z positions of the midpoint of the line described by the Point geometric features. For more information, see the sections above.
X Angle, **Y Angle**, **Z Angle** The X, Y, and Z angles of the line. For more information, see the sections above.

Features

Type	Description
Line	The generated Line geometric feature.

Feature Plane Create

This tool is not intended to be used with Gocator 2D Smart Cameras.

Feature Point Create

The Feature Point Create tool lets you generate a Point geometric features from geometric features generated by other tools. You can also generate a point from an arbitrary position you provide. The following outputs are available (for more information, see *Output Types* below):

- Point
- Constant point
- Point from offset
- Point from three planes
- Point from line and circle
- Point from line and line
- Point from line and plane
- Projected point on line
- Projected point on plane

For more information on geometric features, see *Geometric Features* on page 137.

Inputs

You configure the tool's inputs in the expandable **Inputs** section.

This tool can take an array as input. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.

The tool's inputs depend on the output type you choose in the **Output** parameter. For information on the required inputs and parameters of each output type, see *Output Types* below.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Parameter	Description
Output	The type of output the tool generates. Switching between the options changes the input types displayed in Input section and the parameters displayed in the Parameters section. For information on the output types and the related parameters, see *Output Types* below.
Show Detail	Toggles the display of the input geometric features in the data viewer.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Output Types

The following sections describe the output types available in the tool's **Output** parameter, in addition to the required inputs and parameters.

Point

The **Point** output type takes a Point geometric feature as input. This output is useful if the tool takes a feature generated by another Feature Create tool as input, on which you want to perform

measurements in a second Feature Create tool. This tool can be useful if you have developed GDK tools that only generate geometric features (no measurements): you can use this tool to extract those measurements.

Point Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input.	None.
Anchor X, **Anchor Y**, **Anchor Z** The X, Y, and Z position measurements the tool uses as anchors.	
Point The Point geometric feature the tool uses to create another Point geometric feature.	

Constant Point

The **Constant point** output type displays parameters you can manually fill in to create a geometric feature. This output type is useful if the scan data from frame to frame is reliably fixed and you want to measure from an arbitrary point to another feature.

Constant Point Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input.	**Point X**, **Point Y**, **Point Z** The X, Y, and Z position, respectively, of the first end point of the constant line.
Anchor X, **Anchor Y**, **Anchor Z** The X, Y, and Z position measurements the tool uses as anchors.	

Point from Offset

The **Point from offset** output type takes two Point geometric feature as input, and generates a point on the line connecting the two input Point features. You specify the offset direction and the distance along the line.

Point from Offset Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input.	**Offset Direction** The direction along the line running through the two input Point geometric features that the point is offset.
Anchor X, **Anchor Y**, **Anchor Z** The X, Y, and Z position measurements the tool uses	One of two values: Forward or Backward. A Forward offset direction offsets the point along the line moving

Inputs	Parameters
as anchors. **Point 1**, **Point 2** The Point geometric features the tool uses to create the line along which the offset Point geometric feature is placed.	from Point 1 to Point 2. A Backward offset direction offsets the point along the line moving from Point 2 to Point 1. **Offset Distance** The distance along the line running through the Point geometric features that the point is offset.

Point from Three Planes

The **Point from three planes** output type takes three Plane geometric features as input and generates a Point geometric feature at the intersection of those planes.

Point from Three Planes Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input. **Anchor X**, **Anchor Y**, **Anchor Z** The X, Y, and Z position measurements the tool uses as anchors. **Plane 1**, **Plane 2**, **Plane 3** The Plane geometric features the tool uses to generate the Point geometric feature. The point is at the intersection of the planes.	None.

Point from Line and Circle

The **Point from line and circle** output type takes a Line and a Circle geometric feature as input and generates a Point geometric feature where the line intersects the circle. The tool uses an intersection option to decide which intersection point to use.

Point from Line and Circle Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input. **Anchor X**, **Anchor Y**, **Anchor Z** The X, Y, and Z position measurements the tool uses as anchors. **Line**	**Intersection Option** The intersection between the line and the circle that the tool uses to generate the Point geometric feature. One of the following: Upper, Lower, or Middle. **Upper**: The uppermost intersection point in the data viewer.

Inputs	Parameters
The Line geometric feature the tool uses to generate the Point geometric feature.	**Lower**: The lowermost intersection point in the data viewer.
Circle The Circle geometric features the tool uses to generate the Point geometric feature.	**Middle**: The midpoint between the two intersection points.

Point from Line and Line

When the **Output** parameter is set to **Point from line and line**, the tool takes two Line geometric feature as input and generates a Point geometric feature where the lines intersect. The tool also returns an angle measurement for the angle between the lines.

Point from Line and Line Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input.	**Angle Range** The range returned by the XY Angle measurement. One of the following: "-180 to 180" or "0 to 360".
Anchor X, **Anchor Y**, **Anchor Z** The X, Y, and Z position measurements the tool uses as anchors.	
Line 1, **Line 2** The Line geometric features the tool uses to generate the Point geometric feature.	

Point from Line and Plane

When the **Output** parameter is set to **Point from line and plane**, the tool takes a Line and a Plane geometric feature as input and generates a Point geometric feature where the lines intersect. The tool also returns an angle measurement for the angle between the lines.

Point from Line and Plane Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input.	**Angle Range** The range returned by the Angle measurement. One of the following: "-180 to 180" or "0 to 360".
Anchor X, **Anchor Y**, **Anchor Z** The X, Y, and Z position measurements the tool uses as anchors.	
Line The Line geometric features the tool uses to generate	

Inputs	Parameters
the Point geometric feature. **Plane** The Plane geometric features the tool uses to generate the Point geometric feature.	

Projected Point on Line

The **Projected point on line** output type takes a Point and a Line geometric feature as input and generates a Point geometric feature.

Projected Point on Line Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input. **Anchor X**, **Anchor Y**, **Anchor Z** The X, Y, and Z position measurements the tool uses as anchors. **Point** The Point geometric features the tool uses to generate the Point geometric feature. **Line** The Line geometric features the tool uses to generate the Point geometric feature.	None.

Projected Point on Plane

The **Projected point on plane** output type takes a Point and a Plane geometric feature as input and generates a Point geometric feature.

Projected Point on Plane Output Type - Inputs and Parameters

Inputs	Parameters
Input The Surface or Profile input. **Anchor X**, **Anchor Y**, **Anchor Z** The X, Y, and Z position measurements the tool uses as anchors.	**Projection Direction** The direction of the projection. One of the following: Perpendicular to Plane or Along Z Axis.

Inputs	Parameters
Point The Point geometric features the tool uses to generate the Point geometric feature.	
Plane The Plane geometric features the tool uses to generate the Point geometric feature.	

Outputs

Most tools provide measurements, geometric features, or data as outputs.

Outputs section with a measurement expanded to show user-configurable decision min/max fields and an external ID

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

For information specific to the different output types, see *Output Types* on page 165.

Measurements

Measurement
X, **Y**, **Z** The X, Y, and Z positions of the Point geometric feature. For more information, see the sections above.
XY Angle The angle measurement.

Features

Type	Description
Point	The generated Point geometric feature.

Feature Robot Pose

Image Tools

The following sections describe the Image tools.

> All Image tools convert RGB color image pixels to monochrome using the following formula for processing and measurement, with the exception of the Image Intensity tool:
>
> `pixel = 0.299R + 0.587G + 0.114B`

Image Anomaly Detector

> This tool requires a special dongle and license to function with live sensor data. (Available for purchase from LMI.) You must also install the dongle's supporting CodeMeter runtime to license the tool, which you do in GoPxL Manager. For more information, see *Installing Licensing Software* on page 410.

The GoPxL Anomaly Detector toolset, which is part of LMI's GoPxL Pro Tools offering, lets you train and deploy AI-powered anomaly detection in features and parts in image data acquired by a compatible Gocator 2D Smart Camera.

You can download the Pro Tools package for PC, which contains the toolset (`14658-x.x.x.x_SOFTWARE_GoPxL_Pro_Utilities.exe`), from the downloads area of the LMI Technologies website: https://lmi3d.com/product-downloads/. Run the installer, and provide a location where the content of the package will be unzipped.

The toolset includes the following:

- The Image Anomaly Detector tool: This tool provides an AI-based model training routine and an inference engine that uses a trained model to predict whether something in production image data is an anomaly. The training routine steps you through the process of training on example images you must upload. To train a model, the tool requires a set of "good" (OK) images, roughly 15 to 20 images typically. Optionally, you can include "bad" (NG) example frames to help the tool set a good threshold in the model. The tool can also simulate part orientation and acquisition variations, as well as synthetic NG frames to improve model performance.
- The Training Archive tool: An auxiliary tool that conditionally saves frames to a project created in the Anomaly Detector tool, letting you improve the model on scan data acquired at a later time. For information on the Training Archive tool, see *Training Archive* on page 191.

Sample datasets are available in a package on LMI's Scenario Download page (https://lmi3d.com/scenario-downloads/).

If you are using this tool on a PC, you must install the CodeMeter runtime. For more information, see *Pro Tools: Installing CodeMeter Runtime* on page 102.

The following hardware and software is required to use the GoPxL Anomaly Detector in production:

1. A compatible Gocator 2D Smart Camera.
2. A GoPxL Anomaly Detector license and dongle, inserted into a USB port on the GoMax unit or PC.

A license and dongle are required for running the tool on live scan data. Without the dongle, you can only evaluate the tool on previously acquired images. The tool will also display an error message without a dongle.

Plugging in the license dongle will license the tool. If the camera system is running when you insert the dongle (that is, acquiring images), you must stop the system before it will license the tool.

3. Optionally, a GoMax NX, GoMax ORIN, or GoMax ORIN+ accelerator, depending on scan resolution, cycle time and pre- or post-processing requirements.

4. If you are running GoPxL Anomaly Detector on a PC, a version of the CodeMeter runtime that is compatible with the dongle and license is required.

   You install the runtime in GoPxL Manager; for more information, see *Running GoPxL on a Windows PC* on page 403.

   If you choose to run GoPxL Anomaly Detector on a PC instance of GoPxL, the PC must have a graphics card with a recent Nvidia GPU that supports Nvidia GeForce Game Ready Driver version 528.33 or higher, and a compute capability of 7.5 or higher. To verify the driver version of your video card, launch the NVIDIA Control Panel, and in the main panel of the application, make note of the driver version. To exceed the capabilities of GoMax ORIN in terms of training times and inference, an RTX 4070 or higher is recommended.

   Note the following general requirements for running GoPxL on a PC:

   - Processor: x86-64 (Core i3 or equivalent)
   - RAM: 8 GB

- Hard drive: 64 GB
- Operating system: Windows 10 or 11 (64-bit only)
- Display: 1366 x 768

Inputs

You configure the tool's inputs in the expandable **Inputs** section.

> To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

Name	Description
Enable Batching	Batching (when a tool takes an array as input and processes each element in the array individually) is currently not fully supported by this tool. If you choose to use batching, keep the following limitations in mind: • The Label page only shows the first element of the input array. • For anomaly detection, blobs representing the anomalies are not displayed. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Parameter	Description
Model	The trained model the tool uses to detect anomalies during production.
Anomaly threshold auto	When this setting is enabled, the tool uses a threshold calculated during training. When it is disabled, provide a threshold in Anomaly
Apply anomaly segmentation	When you set **Model** to a previously trained model, this parameter is available. When you enable this parameter, configure the parameters in **Segmentation parameters** (see below).
Segmentation parameters	The parameters that control anomaly output. **Open kernel X** **Open kernel Y**

Parameter	Description
	The X and Y kernel size, respectively, for morphological opening to remove small areas of data. Use these settings, for example, to remove bridges between areas to properly isolate them or to remove small areas entirely (perhaps caused by noise). Use different values of X and Y to use a non-rectangular filter to adapt the kernel to the kinds of unwanted data you see in the scan data. **Close kernel X** **Close kernel Y** The X and Y kernel size, respectively, for morphological closing to fill in holes smaller than the specified kernel size. Use these settings, for example, to fill small areas within potential anomalies that may be caused by drop-outs. Use different values of X and Y to use a non-rectangular filter to adapt the kernel to the kinds of holes you see in the scan data. **Use area filter** If **Use Area Filter** is enabled, the tool applies an area filter to anomalies using the values you provide in **Max area** and **Min area**. Use the filter to ignore small features such as debris or dust or large irrelevant regions. **Ordering** Orders the measurements, features, and surface data of the individual anomalies output by the tool. One of the following: • Position - X increasing • Position - X decreasing • Position - Y increasing • Position - Y decreasing • Area - Large to small • Area - Small to large
Train Model	Starts the model training routine, which steps you through process. For more information, see *Train Model* on the next page.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Development Workflow

The high level workflow is as follows:

1. Collect representative images of the features or parts you need to inspect in order to train an Anomaly Detector model. In addition to "good" parts, you can scan "bad" parts if these are available, that is, ones with typical defects, to help set a good threshold.
2. Create a pre-processing toolchain that excludes unimportant background and train an Anomaly Detector model.
3. Deploy the Anomaly Detector model and connect post-processing tools.

4. Optionally, collect new production data for model retraining.

Collect dataset

Unless you are using a previously created dataset, you must create the dataset that the GoPxL Anomaly Detector will use for training a model before it can detect anomalies in new production data. To do this, perform the following steps:

1. Record representative scans of 20 or more OK targets (targets with no flaws) and 20 or more NG targets (targets with flaws).

   You can record the scans to a single .gprec file or to multiple files. If you record scans to a single file, grouping each type (OK and NG) together will make it easier to label them later in the wizard. Make sure there are no empty frames: each frame should contain either an OK or an NG target. Do not rely on .gpsup or .rec for this as the Anomaly Detector wizard requires .gprec files when importing from PC.

2. Save a separate .gpsup file with the sensor configuration settings such as exposure time and related imaging parameters.

   Ensure the same parameters are used on the production system. Otherwise, you will need to collect new training data.

Implement a Preprocessing Toolchain

The preprocessing tool chain limits variation in the image data to the region of interest (with and without anomalies). With less variation in the image data, the total number of frames needed for training is reduced.

The preprocessing tool chain must not change between training and inference. Otherwise, the system will not function properly. You must save your preprocessing toolchain in the .gpsup file created previously or using a separate .gpjob file.

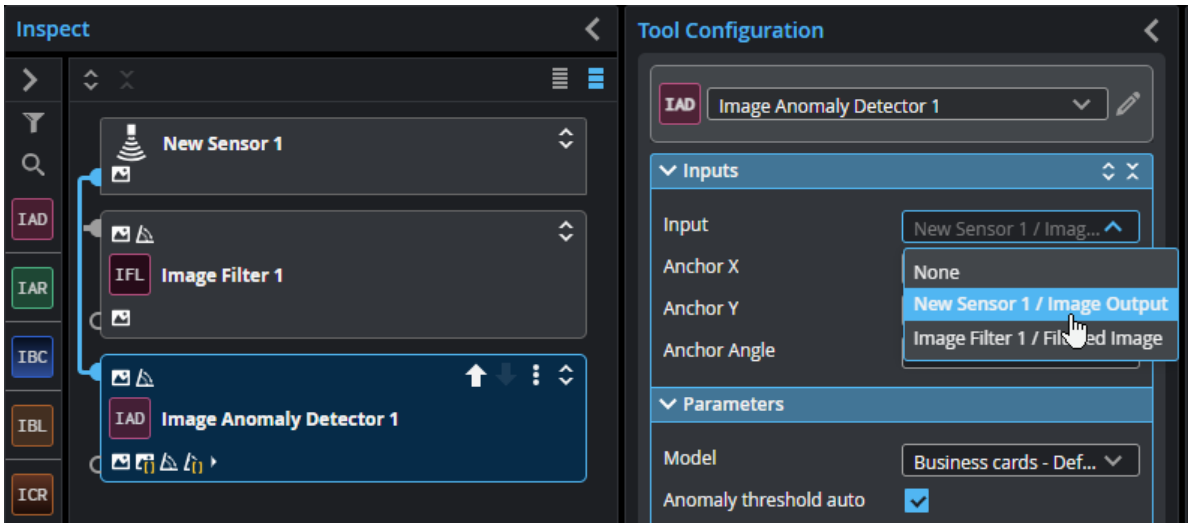
> You *must* use the same preprocessing toolchain for training and inference to ensure proper detection of anomalies.

Train Model

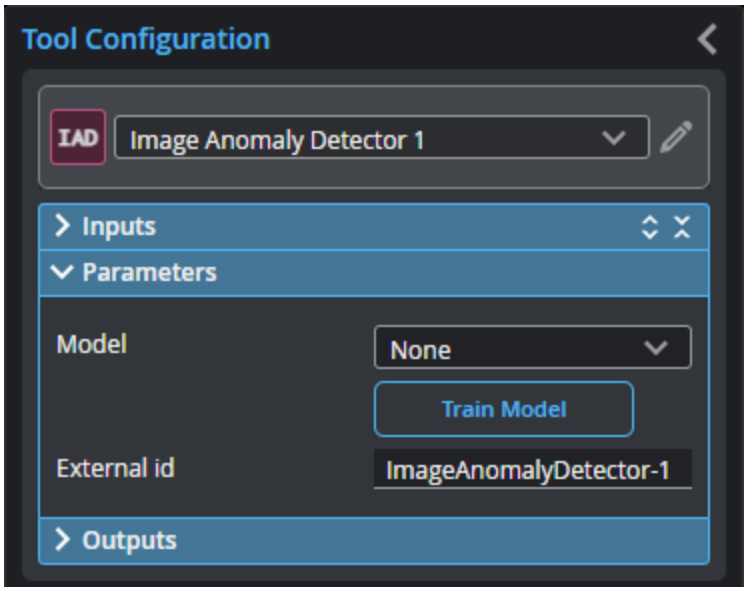
Add Image Anomaly Detector Tool

1. Add and connect a Surface Anomaly Detector tool to the output of the preprocessing tool chain.

   In the following the first four tools in the Tool Diagram on the left are the preprocessing tool chain. The last tool in the chain, Surface Filter, is used as input for the Anomaly Detector tool.



2. In tool's **Parameters** section, click the **Train Model** button.

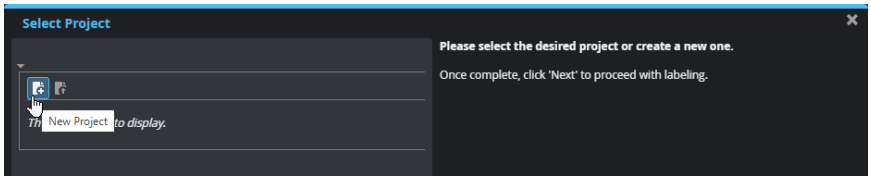


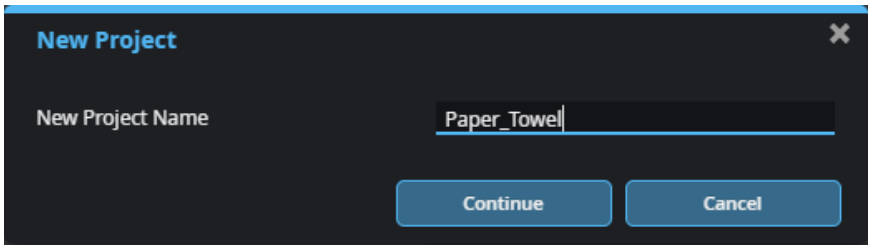
The routine starts and the **Select Project** page opens. You will create a new project in this page, or open a previously created project.

Create New or Open Project

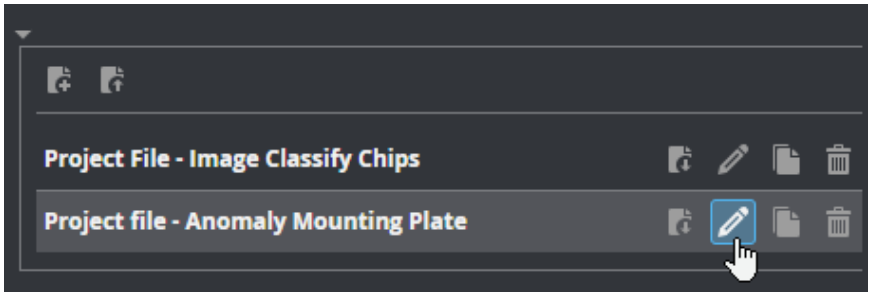
A project is a collection of frames used for training, labels you apply to those frames (OK and NG), and the models trained on those frames. You should choose a meaningful project name, such as the part name, especially when working with multiple projects.

1. On the **Select Project** page, click the New Project icon, provide a project name, click **Continue**, and then click **Next**.

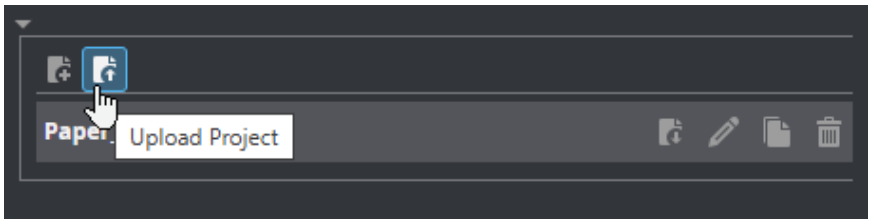




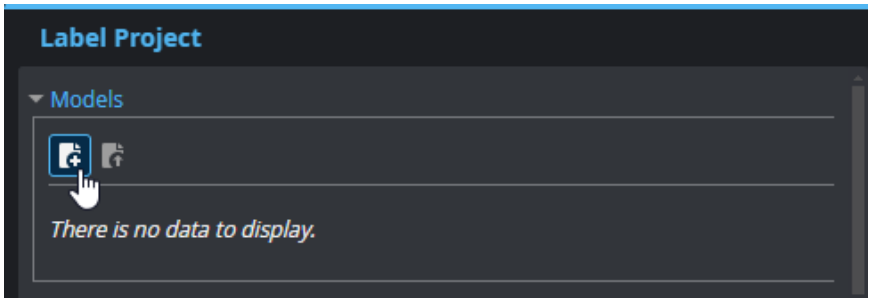
You can use the download icon on a project's row to download this to your PC and upload to another device.

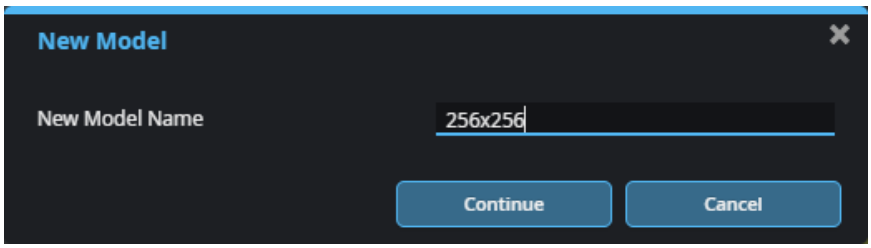


You can use the upload icon to import a previously downloaded project.

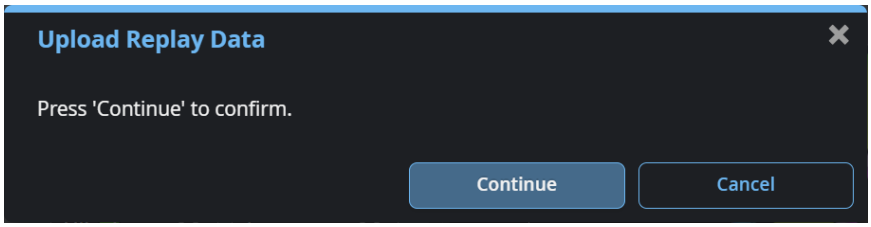


2. In the **Label Project** page, in the **Models** section, create a new model and click **Continue**.





3. In the **Frames** section, do one of the following
   - To upload currently loaded replay data, click the Upload Replay Data button ( ), and then click **Continue**.



- To upload the data from a .gprec file on the local PC, click the Upload From Disk button ( ), choose a file in the Open dialog a, click Open.

The tool loads the frames into the model.

> ⚠ Loading replay data into the project can take several minutes. If a "Delayed Response" message displays, click "Wait." In this case, add a Decimation Filter to the pipeline.

Note that you can also use the Replay Editor to upload image files from a PC to create a dataset; for more information, see *Using the Replay Editor* on page 98.

Label Frames and Choose Set

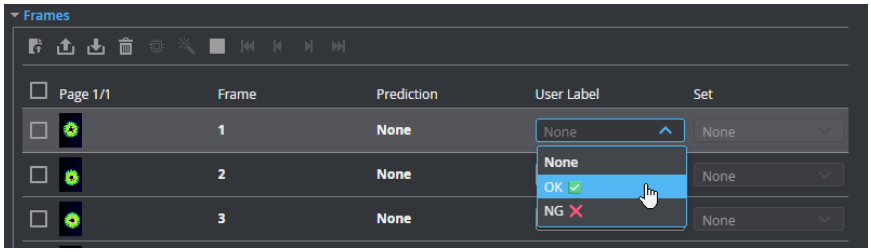
You must now label each frame as "OK" or "NG" (under the **User Label** column). You must also choose the set each frame belongs to (Train or Test, under the **Set** column). By default, the tool automatically places frames in the Train or Test set, using the proportions listed below. You can disable auto-split, but you should use the same proportions.

80% of OK is in Train

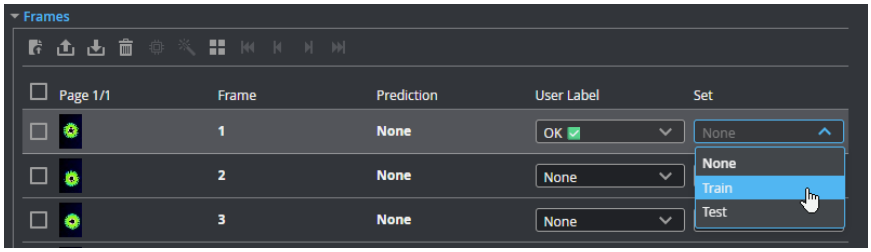
20% of OK is in Test

100% of NG is in Test

1. In the row of each frame, under **User Label**, choose the correct label.



If you have not enabled auto-split, also choose the frame's set under **Set**. Make sure you use the proportions described in step .



To quickly choose the label and set for multiple frames, do one of the following to first select the frames:

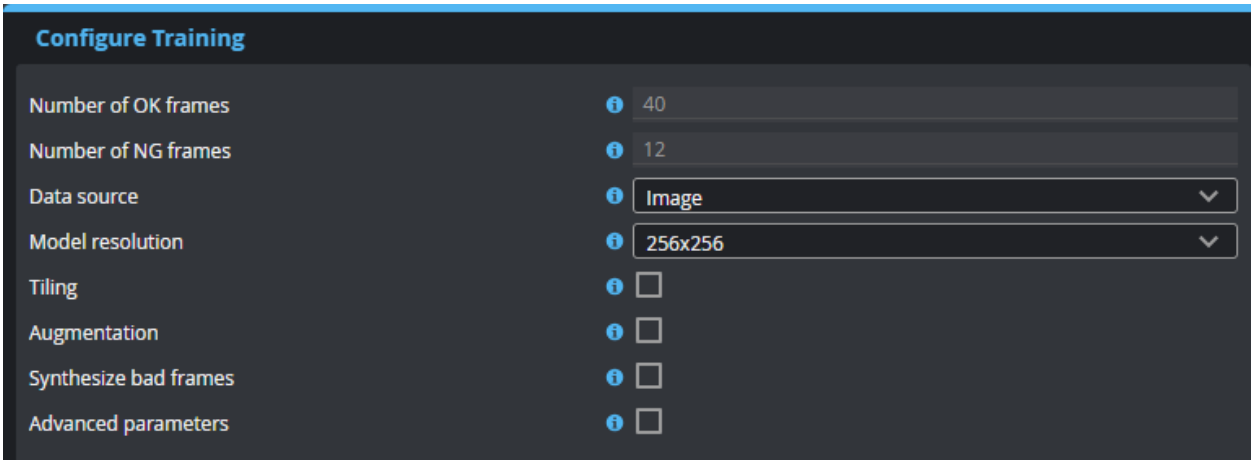
- To select a range of frames, click the checkbox of the first frame, press Shift, and click the checkbox of the last frame of the range.
- To select multiple frames, click the checkbox of the first frame, press Ctrl, and click the checkbox for each frame you want to select.
- To select non-contiguous ranges, alternate between clicking Ctrl and Shift.

After selecting the frames, choose the label and the set as described above. To deselect all frames, click anywhere between the checkbox and the User Label and Set drop-down menus.

2. Click **Train Model** to configure training.

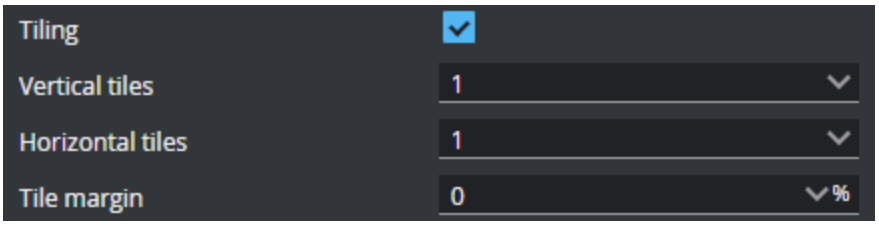
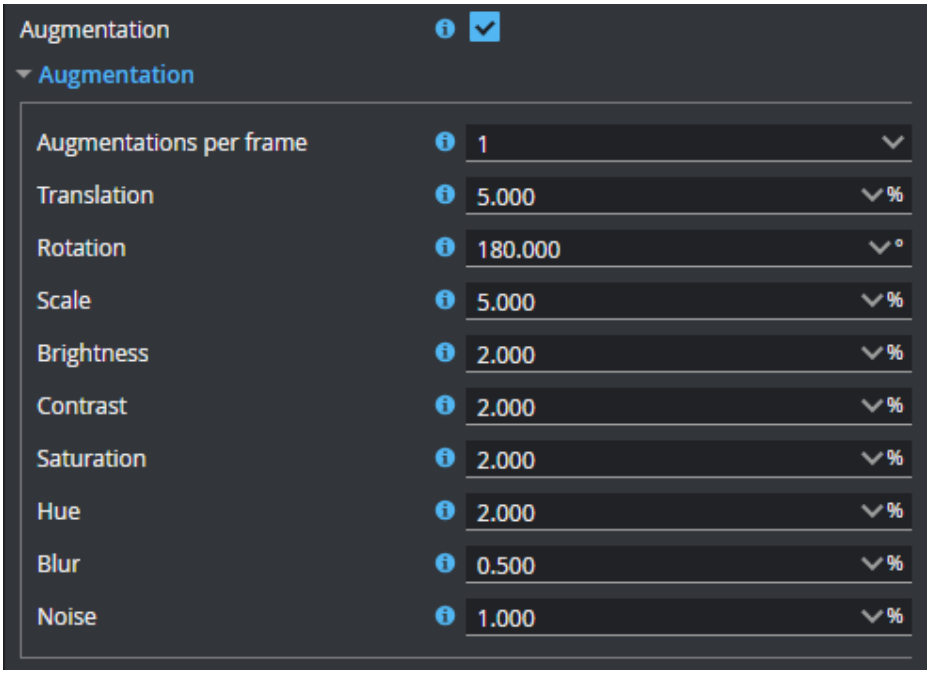
Configure Training

After labeling the frames and choosing the set they belong to, you must configure training. For descriptions of the settings, see below.



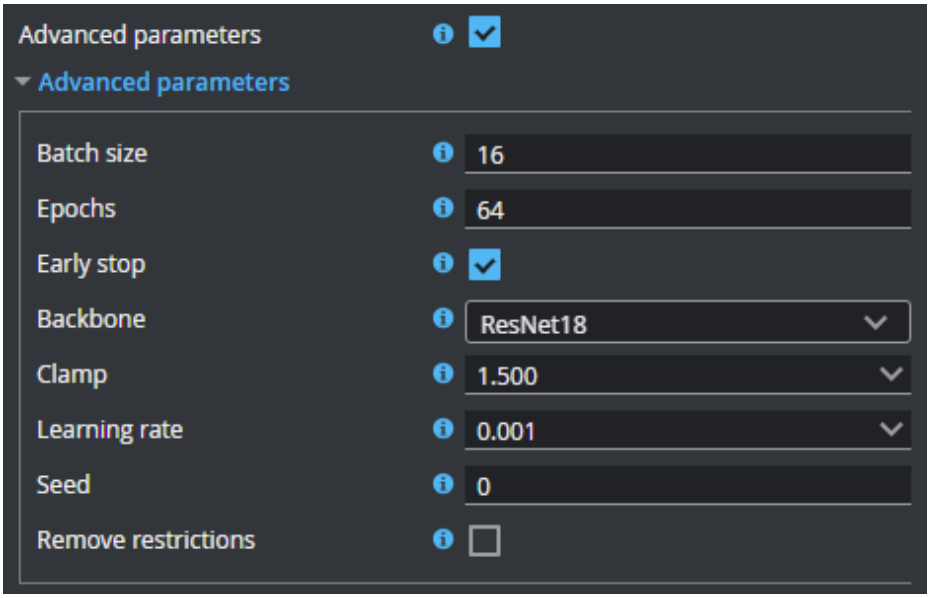
Training Parameters

Parameter	Description
Number of OK frames Number of NG frames	Read-only fields that display the number of frames you have labeled OK and NG, respectively.
Data source	The source the tool uses during training.
Model resolution	Decimates the input data to the specified resolution for processing. A higher resolution improves anomaly location accuracy but increases processing time for training. For most applications, leave this at 256x256.
Tiling	Enables tiling. Tiling divides the frames of training data into sections. The tool uses the same model for each tile, which increases the resolution of the model. This lets the tool potentially find smaller defects or increase detection accuracy. You should only use tiling with homogenous targets (for example, paper towel, wood decking, and so on) or symmetric targets. For discrete parts with a complex or asymmetric shape, disable this setting. After enabling tiling, set the number of vertical and horizontal tiles. You can also set tiles to overlap in **Tile margin**. Use this to detect anomalies that might fall on a boundary between tiles. Generally, you should use a value between

Parameter	Description
	12.5% and 25%. Note that using tiling and tiling margins increases the memory requirements for training considerably. For example, with 5x5 tiling, the tool uses 25 times more memory. 
Augmentation	When **Augmentation** is enabled, the tool generates random variations to simulate the kinds of variations encountered in production use. This improves the training data, which in turn improves the tool's ability to detect anomalies in production. The rotation, translation, and scale values are used to create a range. So for example, a rotation value of 1 degree results in a range between -1 and 1, from which the tool chooses a random value to generate data. Choose augmentations that are representative of the variations you encounter during production. Augmentations that aren't representative may lead to poor results. You can visualize example augmentations on the Model Performance page, which is available after training is complete; for more information, see *Test Model Performance* on page 185.  **Augmentations per frame**

Parameter	Description
	The number of augmentations generated for each frame in the project (OK and NG). Augmentations are not added to the total number of frames in the project itself. **Translation** Limits movement of simulated data on the XY plane, as a percentage of the pixels in the frame. **Rotation** Limits rotation of simulated data around the Z axis. **Scale** Limits the scaling of simulated data. **Brightness** Varies the brightness in frames to simulate different lighting conditions. **Contrast** Varies the contrast of the simulated data. **Saturation** Varies the saturation of the simulated data to represent a more vibrant or faded appearance of targets. **Hue** Varies the color palette of the simulated data to represent different lighting conditions or variations in the appearance of targets. **Blur** Varies the blur of the simulated data to represent device noise or poor focus. **Noise** Varies the noise of the simulated data to represent device noise or poor image quality.
Synthesize bad frames	When this setting is enabled, the tool synthesizes bad (NG) frames from the good (OK) frames in the project. The frames are used internally during training and are not added to the frames in the project or shown in the tool. Enable this setting if your project contains very few NG frames. The setting is

Parameter	Description
	automatically enabled if the project contains *no* NG frames.
Advanced parameters	If the training results are generally good but the tool is reporting a small number of false positives or false negatives, you can modify these parameters to further improve results.



Batch size

The number of frames in a batch (a subset of the entire dataset passed to the training algorithm for an iteration).

If you are using a model with a 512x512 resolution or tiling and the tool has memory issues during training, decrease this value.

Epochs

The number of times the entire dataset passes through the training algorithm. To reduce false positives or negatives, increase this value. Note that training will take longer.

Early stop

Leave enabled for the initial training runs. To help model performance, disable to improve model performance; note that training will take longer.

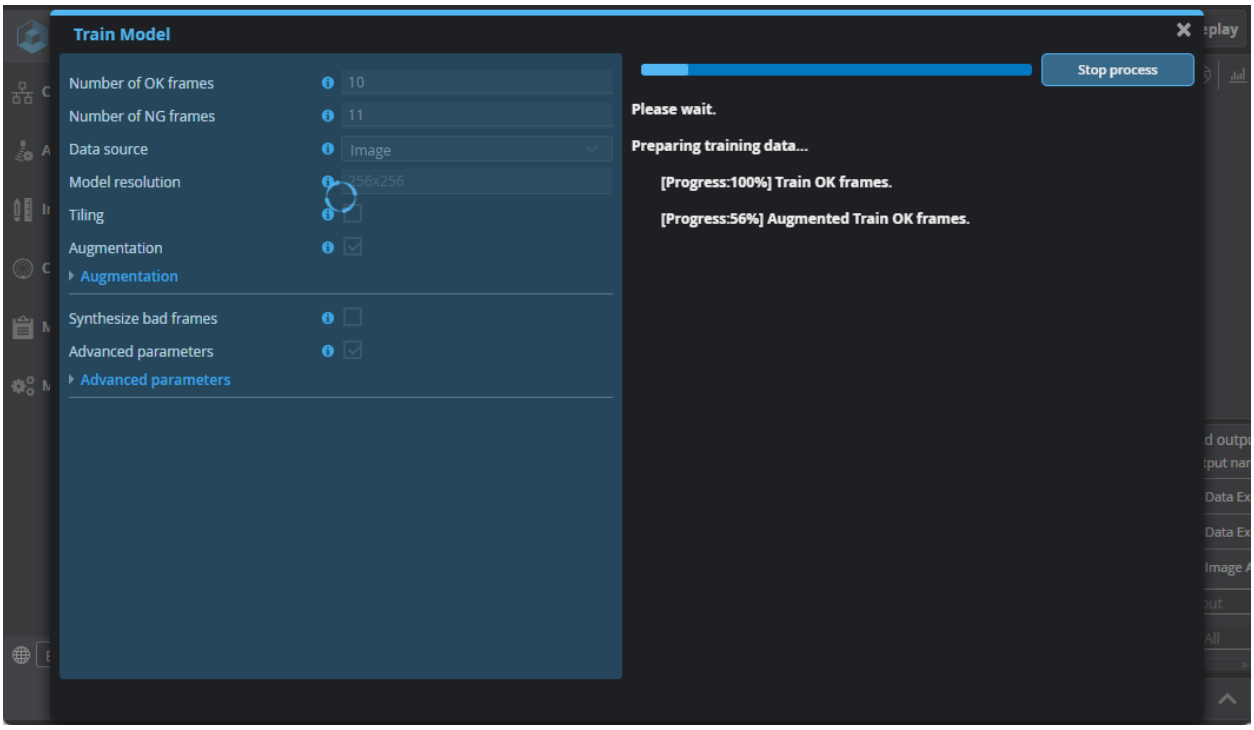
Backbone

The feature extractor used by the tool. One of the following:

ResNet18: A lighter model that is fast and a good starting point.

Parameter	Description
	Wide ResNet50-2: Slow and larger memory, with better for complex textures. **Clamp** The range within which the data values should be clamped. Reduce this value to help reduce false positives or false negatives. **Learning rate** Set this to a higher value to reduce training time. Note that a value that is too high can cause the model to miss the point where the difference between predicted and actual OK / NG is at its smallest (that is, overshooting it). A lower value causes the model to learn more slowly, but potentially more accurately; if the value is too low, the model can get stuck. **Seed** Set this to a non-zero number to ensure your results are consistent and reproducible between training runs. Set the value to 0 use a random seed for each training run. **Remove restrictions** By default, this option is disabled, and the following limitations are in place to ensure the system trains successfully and does not run out of memory: • The maximum number of tiles under the **Tiling** parameter is limited to 6. • The highest resolution in **Model resolution** is 512x512.

After you configure training, click **Start Training** to train the model. The tool will start by preparing the training data, providing progress feedback.



After the data is prepared, the tool will start generating Loss and AUROC graphs.

Loss: This graph represents the cumulative error for each frame in the Test set, determined after each training run iteration or epoch. A decreasing number (the graph's line is moving down) implies the model is improving.

AUROC: This graph defines how well the model can distinguish between the OK and NG user labels. A value of 1 (the graph's line is moving up) represents the best possible result.

As the tool trains the model on the labeled frames, the graphs should converge, that is, Loss will decrease, and AUROC will increase, approaching 1 (the best possible result). After the training has completed, click **Next** to test the model. Note that training can take several minutes.

Test Model Performance

After the tool has finished training the model, it will use the model to make predictions on the frames to determine the anomaly score for each frame, as well as the anomaly threshold. The Model Performance page displays the following results:

Performance Results

Name	Description
Model	The trained model the tool used to detect anomalies during the model performance phase.
True Positive Rate	The percentage of frames correctly predicted as OK.
False Positive Rate	The percentage of frames incorrectly predicted as OK.
False Negative Rate	The percentage of frames incorrectly predicted as NG.

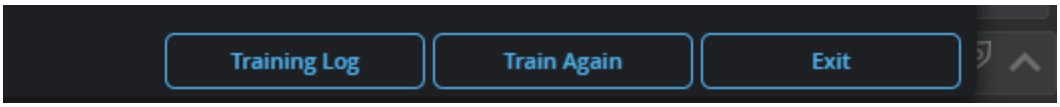
Name	Description
Processing Time	The average time required for inference. Includes the pre-processing tool chain.
Anomaly Threshold	The threshold to which a frame's Anomaly Score measurment is compared to determine whether the frame is an anomaly. An anomaly score above this value is an anomaly.

You can use the **Anomaly visualization** drop-down to view different visualizations of the selected frame:

Anomaly visualizations

Name	Description
Anomaly Outline	Displays the Image data of the frame with anomalies outlined, if present.

The Model Performance page also lets you download an HTML file containing reports and other diagnostics. LMI recommends checking the visual results in this file.



You can also retrain the model with different training configuration if you wish.

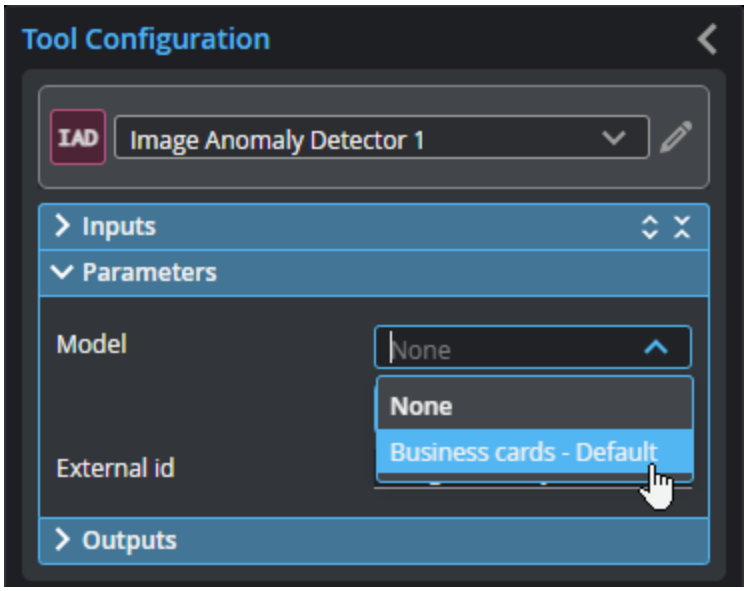
You are now ready to use the trained model on new scan data in production.

Deploy in production

Anomaly Tool Setup

1. Connect to the device using a PC on the same network.
2. Upload a .gpsup or .gprec file created during the dataset collection.

   Ensure the recording contains the same preprocessing pipeline used during training.
3. Discover the Gocator sensor and create a new sensor group.
4. Insert the GoPxL Anomaly Detector USB dongle to a USB port on the GoMax.
5. In the Image Anomaly Detector tool, in **Model**, choose the model you trained and want to use in production.



6. Configure the outputs as required, and implement post-processing to isolate and prepare anomalies for measurement. For more information, see *Outputs* below and *Image Anomaly Detector* on page 172.

Outputs

Although the tool has several outputs, the "Is Anomaly" measurement and "Anomaly Surface" output are the main outputs intended for processing by tools later in the tool chain.

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

Some outputs are only displayed if **Apply anomaly segmentation** is enabled in the **Parameters** section.

Measurements

Measurement
Anomaly Result A string representing the result: OK or NG, corresponding to 0 and 1, respectively, in the "Is Anomaly?" measurement.
Anomaly Count The number of anomalies detected in the frame.
Is Anomaly? Indicates whether the tool has detected an anomaly in the current frame of scan data.
Anomaly Score Indicates how strongly the tool believes the current frame contains an anomaly. When the value is higher than the Anomaly Threshold measurement, this measurement is set to 1.0.
Anomaly Threshold

Measurement
Threshold value (0..1) of the 'Anomaly Score' above which the model considers an input image anomalous. This value is determined during training to achieve the best possible detection rate.
Segmentation Threshold
A grayscale value in the Image data above which the model considers input intensity data anomalous. The value is constant for a trained model.
Intensity Threshold
Threshold value (0..1) of the 'Anomaly Score' above which the model considers an input image anomalous. This value is determined during training to achieve the best possible detection rate.
Center X
Center Y
Arrays containing the X and Y positions of the centers of the anomaly segments.
Width
Height
Arrays containing the width and height values of the anomaly segments.
Area
Array containing the areas of the anomaly segments.

Data

Type	Description
Heatmap	Diagnostic Image output overlaying the Anomaly Map output on the input scan data. This output can make it easier to see whether the brighter areas in the anomaly map correspond with flaws in the target.
Anomaly Image	Contains the Image data of the frame, with the anomaly map (see Anomaly Map below) added as intensity data. The intensity data represents a per-pixel score of how anomalous that pixel is.

Store production data

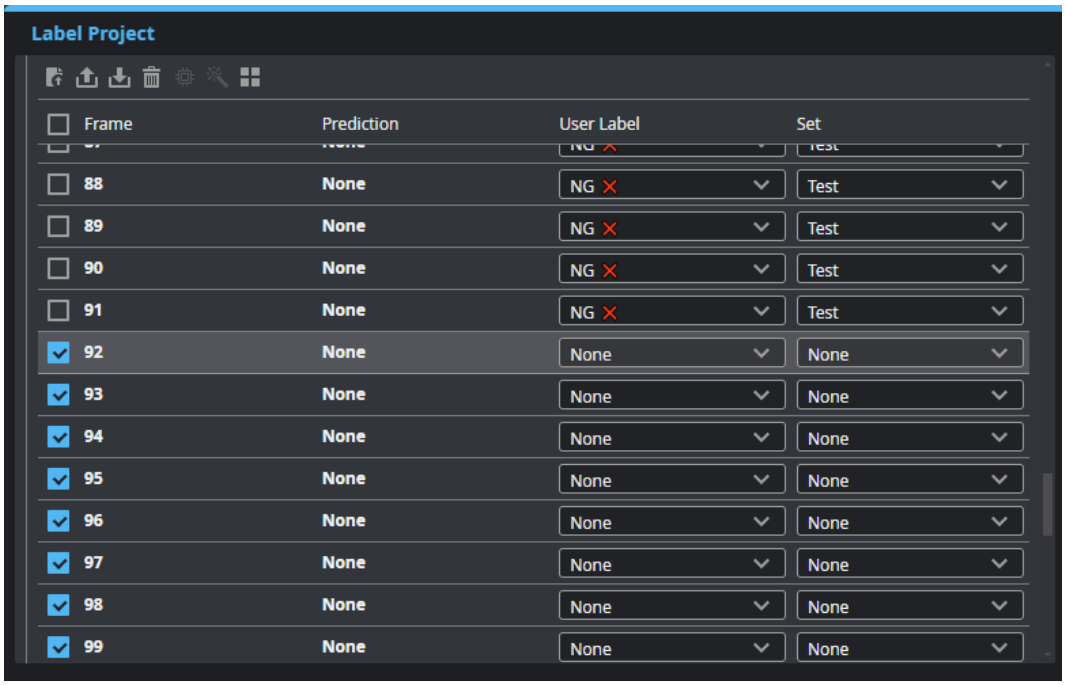
You can use the Training Archive tool to conditionally add frames to a Surface Anomaly Detector project. For more information, see *Training Archive* on page 191.

Train using additional data

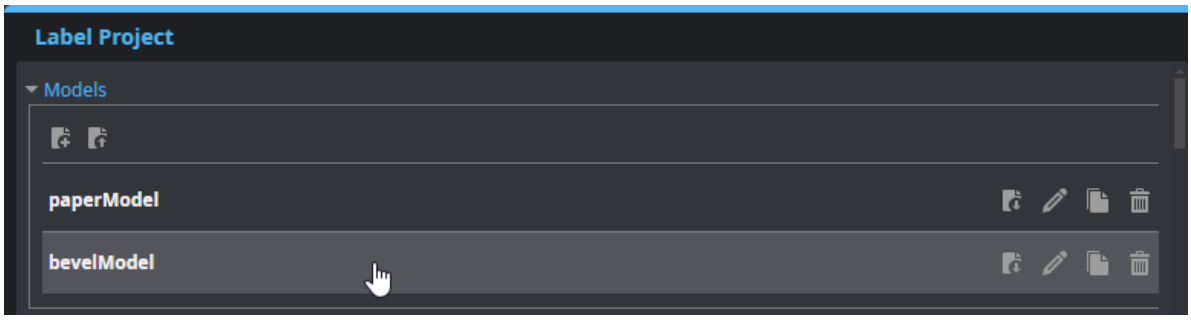
After adding new frames to a project using the Training Archive tool (see *Training Archive* on page 191) or uploading additional frames manually (from the replay buffer or from a .gprec file), you can further train the model with new frames, taking advantage of the trained model's prediction capabilities to automatically label frames as OK or NG.

To train using new data:

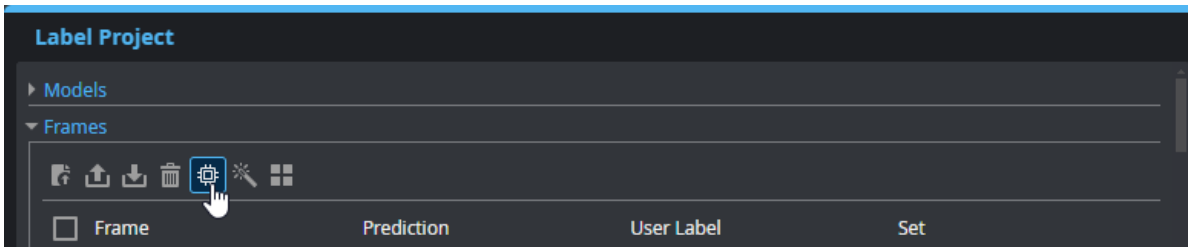
1. After adding frames to a project, go to the Label Project page.
2. Select the new frames.



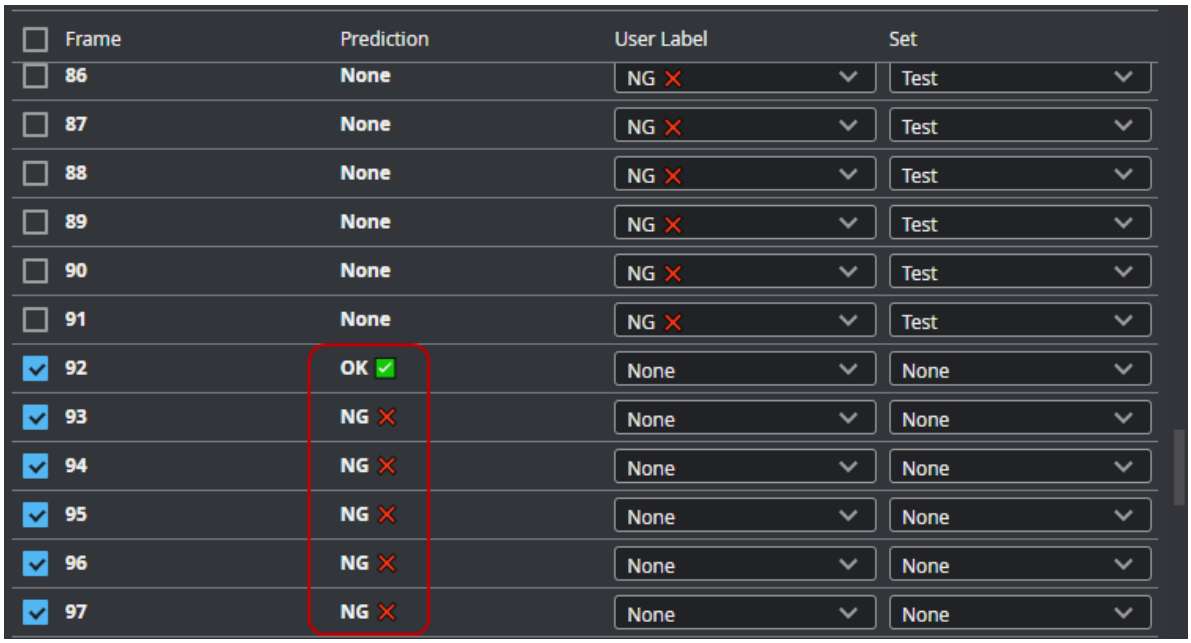
3. Choose the correct model you want to use for prediction.



4. At the top of the Frames section, click the Generate Prediction button.

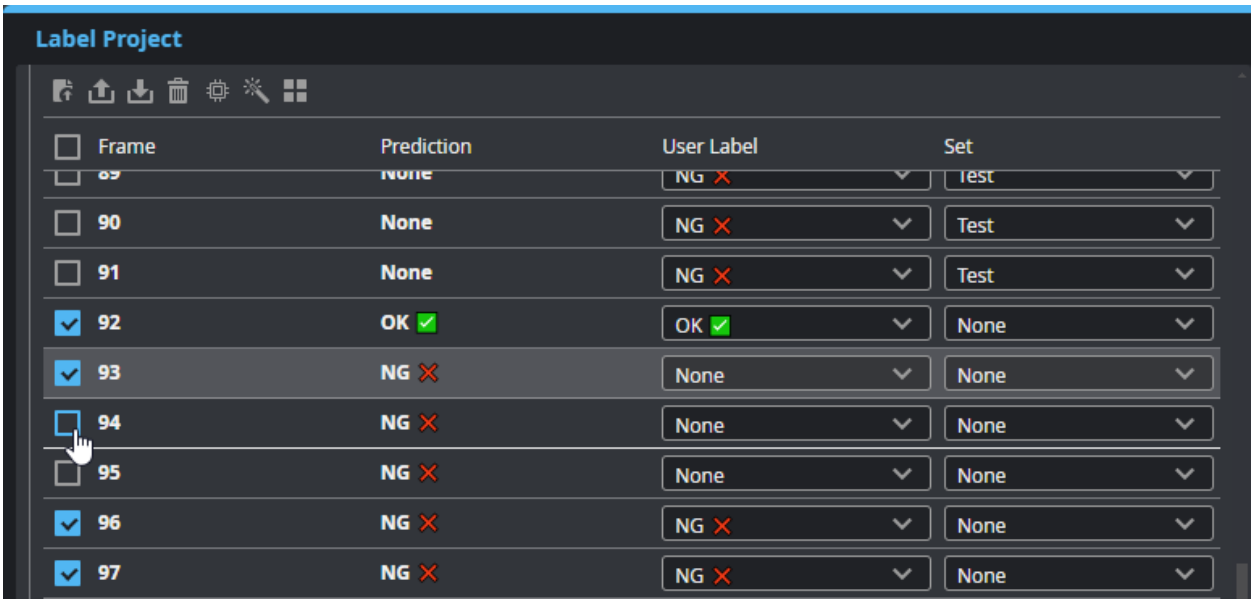


The tool uses the selected model to generate predictions (OK or NG) for the selected, unlabeled frames. Note that the user labels of the frames are unchanged at this point.

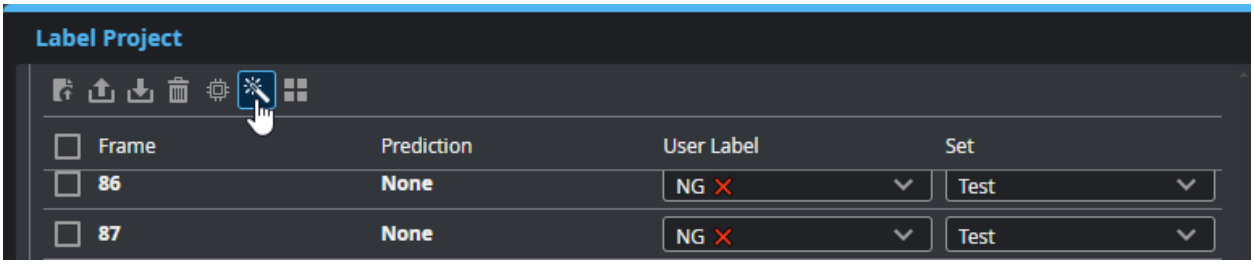


5. Review the predictions and do the following:

   a. Uncheck any invalid predictions.



   b. Click the Apply Prediction button.



c. Toggle Auto-Split to On to automatically set the Set of the selected frames.

d. Manually set the user label of any incorrectly predicted frames.

The predictions of the selected frames are copied to the user label drop-down.

6. Retrain the model with the newly added frames.

For more information, see *Train Model* on page 176.

Training Archive

The Training Archive tool lets you conditionally add frames to an Anomaly Detector project. (For more information on Image Anomaly Detector, see *Image Anomaly Detector* on page 172.)

Inputs

You configure the tool's inputs in the expandable **Inputs** section.

> To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array, other than the processing limitations of the sensor. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Input	The data the tool applies measurements to or processes. This tool can optionally take an array as input. For more information, see Arrays, Batching, and Aggregation.
Conditional Measurement	The measurement the tool uses as a condition to add frames to a project. Some examples of measurements typically used are the "Is Anomaly" output of the Surface Anomaly Detector tool, the Result measurement of the Measurement Formula tool, or an "Output" output from a Script tool.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Parameter	Description
Enable Export	When this setting is checked, if the conditions are met (**Condition** and **Condition Threshold** parameters) the tool will add the current frame to the selected project.
Condition	Displayed if the **Conditional Measurement** input is set to a measurement. Set this to the type of condition (with respect to the measurement set in **Conditional Measurement**) the tool should use to determine whether to add a frame to the selected project. The frame is added based on the following options:

Parameter	Description
	Fail or Invalid: If the selected measurement returns a Fail or Invalid result. **Invalid**: If the selected measurement returns an Invalid result. **Pass**: If the selected measurement returns a Pass result. **Less or Equal to**: If the selected measurement less than or equal to the value provided in **Condition Threshold**. **Greater or Equal to**: If the selected measurement greater than or equal to the value provided in **Condition Threshold**.
Condition Threshold	The value the tool compares with the measurement set in **Conditional Measurement** to determine whether to add a frame to the selected project. This parameter is only displayed when **Condition** is set to "Less or Equal to" or "Greater or Equal to".
Project	The project the tool will add frames to if the selected conditions are met.
Storage Threshold	The minimum amount of storage allowed on the GoMax unit or the PC. If the storage goes below this value, Training Archive will stop adding frames to the selected project, and the Storage Limit Reached output will be set to 1. Use the Available Storage output to see the current amount of storage on the device.
Manage Projects	When you enable **Manage Projects**, in the **Action** drop-down, you can delete the project currently selected in **Project**, or delete *all* of the projects on the device.
Action	⚠ You are not prompted to confirm whether to delete projects!
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

Most tools provide measurements, geometric features, or data as outputs.

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

Measurements

Measurement
Captured The number of frames added to the project selected in **Project** so far.

Measurement
Available Storage The amount of storage available on the GoMax or PC.
Storage Limit Reached When the available storage goes below the value in the **Storage Threshold** parameter, this output returns 1.

Image Arithmetic

All Image tools convert RGB color image pixels to monochrome using the following formula for processing and measurement, with the exception of the Image Intensity tool:

```
pixel = 0.299R + 0.587G + 0.114B
```

The Image Arithmetic tool lets you perform various operations on a pair of images, such as using performing dynamic masking from frame to frame. The tool performs bitwise operations (AND or OR) on the corresponding pixels in the source data.

Inputs

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array, other than the processing limitations of the sensor. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
First Image Input	The data the tool applies measurements to or processes.
Second Image Input	The secondary input the tool uses to perform arithmetic. Some operations only require a single image.

Parameters

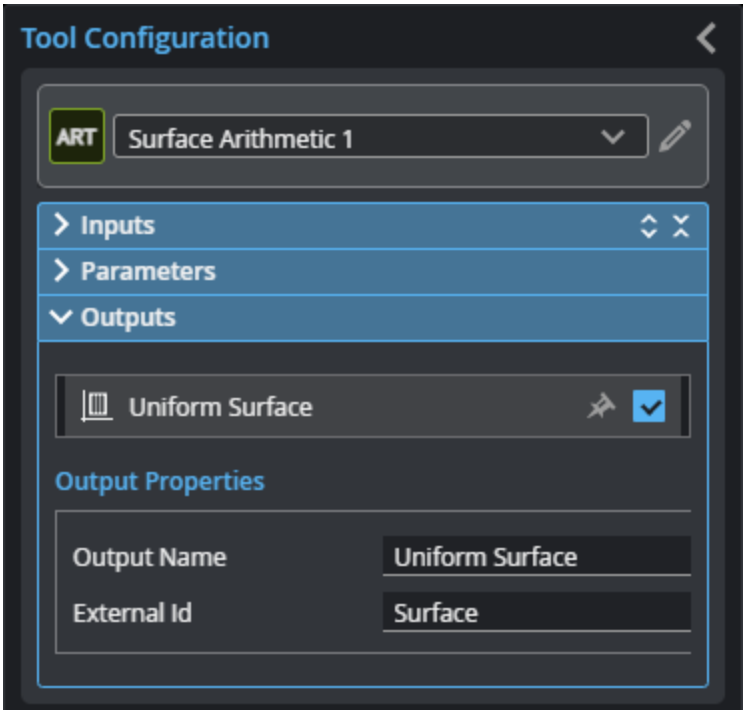
The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Operator	One of the following: **Add** - Add the pixel values at the corresponding positions in the images. If the result is an overflow, the maximum value of the image data type is used. **Subtract** - Subtract the pixel values at the corresponding positions in the images. If the result is less than 0, zero is used. **Multiply** - Multiply the pixel values at the corresponding positions in the images. If the result is an overflow, the maximum value of the image data type is used. **Divide** - Divide the pixel values at the corresponding positions in the images. **Average** - Calculate the average of the pixel values at the corresponding positions in the images. **Max** - Determine the maximum of the pixel values at the corresponding positions in the images. **Min** - Determine the minimum of the pixel values at the corresponding positions in the images. **Scale and Offset** - Perform a linear transformation for the input image (Ax+B). **>** - Calculate whether the pixel value from the first input image is greater than the

Parameter	Description
	one from the second input image for the same position. **>=** - Calculate whether the pixel value from the first input image is not less than the one from the second input image for the same position. **<** - Calculate whether the pixel value from the first input image is less than the one from the second input image for the same position. **<=** - Calculate whether the pixel value from the first input image is not greater than the one from the second input image for the same position. **==** - Calculate whether the pixel value from the first input image is equal to the one from the second input image for the same position. **!=** - Calculate whether the pixel value from the first input image is not equal to the one from the second input image for the same position. **Bitwise And** - Calculate the bitwise AND result of the pixel values at the corresponding position in the two images. **Bitwise Or** - Calculate the bitwise OR result of the pixel values at the corresponding position in the two images **Bitwise Xor** - Calculate the bitwise XOR result of the pixel values at the corresponding position in the two images **Bitwise Not** - Calculate the bitwise NOT result of the every pixel value in the input image. **Logical And** - Calculate the logical AND result of the pixel values at the corresponding position in the two images. **Logical Or** - Calculate the logical OR result of the pixel values at the corresponding position in the two images. **Logical Xor** - Calculate the logical XOR result of the pixel values at the corresponding position in the two images. **Logical Not** - Calculate the logical NOT result of the every pixel value in the input image
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs



All outputs provide an external ID (available by expanding the output in the Outputs panel) for optional use in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Data

Type	Description
Result	The processed data.

Image Barcode

The Barcode tool lets you decode 1D (linear) barcode data encoded in Image data in grayscale or color.

The tool returns the decoded string, as well as the position (center) and the angle of the code.

The tool also provides measurements that can help you assess the print quality of the code symbols. The tool uses the ISO 15416 grading standard. Only Code 128, Gs1 128, and EAN 13 are supported.

Inputs

To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array,

Name	Description
	other than the processing limitations of the sensor. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Parameter	Description
Regions	When expanded, displays the region- and mask-related settings.
Enable	Enables regions and displays the region- and mask-related settings (see below).
Region {n}	When you enable regions (see above), the tool displays additional settings related to the region type. For details on the regions supported by this tool and their settings, see *Flexible Regions* on page 133. For general information on regions and the difference between standard and "flexible" regions, see *Regions* on page 132.
Instance Count	The maximum number of bar codes to decode in a frame.
Symbology	Three categories of bar codes are provided under expandable sections: Default, Optional, and Unsafe. Use the **Operation** drop-down to select or deselect all types in the category. **Default** The following types are enabled by default. Code 128 / EAN 8 / EAN 13 GS1-128 / Code 39 / Code 39 Extended Code 39 Reduced / Code 93 / Code 93 Extended GS1 DataBar Omnidirectional / GS1 DataBar Limited / GS1 DataBar Expanded **Optional** The following types are disabled by default. Codebar / Code 25 Interleaved / MSI UPC A / UPC E / ADS Anker BC 412 / Code 11 / Code 13 Code 2/5 Datalogic / Code 2/5 Matrix / Code 2/5 IATA

Parameter	Description
	Code 2/5 Industry, Code 2/5 Compressed, Code 2/5 Inverted Code 32, Code BCD Matrix, Code C.I.P. IBM Delta Distance A, Plessey, Telepen RSS14, RSS14 Limited, RSS14 Expanded **Unsafe** Code STK, Binary Code, Pharma Code One Track
Compute Grading	Whether to assess the quality of the barcode using the ISO 15416 grading standard. Only Code 128, Gs1 128, and EAN 13 are supported. If **Compute Grading** is set to a value other than None, measurement are added in the **Outputs** section.
Advanced	Expandable section containing advanced settings. **Display Checksum** Whether to display the checksum of the decoded result. **Human Readable Gs1** Whether to convert the decoded result to human readable result. **Reading Orientation** For most symbologies, a pattern identifies whether the codes must be read from left to right or right to left. For the Code STK, Binary Code and Pharmacode symbologies, select the orientation. **Enable Permissive Decoding** Whether to decode codes containing partially incorrect information. When enabled, the tool assumes characters might be badly printed and uses the checksum to correct that character. In rare cases, this option may lead to reading false positives codes. Only applies to symbologies for which a checksum is not mandatory. **Timeout** The maximum time the tool is allowed to take. **Validate** Mandatory Checksum: Whether to validate symbologies for which a checksum is mandatory.

Parameter	Description
	Optional Checksum: Whether to validate symbologies for which a checksum is optional . Pharmacode: Whether to validate the PharmacodeOneTrack symbology.
Learning	Expandable section containing learning-related settings and operations. Enable or disable settings, place a representative barcode under the camera, and click **Learn image** to determine the best options to use to read bar codes. Click **Reset** to clear. Learning improves robustness and speed on "similar images," meaning repeated production images with comparable optics, distance, and code layout. Avoid it for highly variable scenarios (different cameras, large variation in code size and distance) where one learned configuration will not generalize well. **Keep Default Scales** Whether to keep the scales used by default even if they are useless for reading the barcode. **Add All Scales** Whether to add all of the scales at which a code is detected. Otherwise, the smallest number of scales needed to read the barcode is added.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

Most tools provide measurements, geometric features, or data as outputs.

Outputs section with a measurement expanded to show user-configurable decision min/max fields and an external ID

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

When **Compute Grading** is set to "Letter and Number", the tool outputs both "Number Grade" measurements containing a value between 4 and 0 and "Grade" outputs that contain A, B, C, D, or F. If the setting is either "Letter" or "Number", the tool only returns "Grade" outputs, which contain the selected type.

If you set **Instance Count** to a value greater than 1, measurements other than Total Count are returned as arrays.

Measurements

Measurement
Total Count

Measurement
The number of bar codes decoded.
X **Y** The X and Y position of the center of the barcode.
Angle The angle of the barcode.
Decoded String The decoded string.
Decode Grade **Decode Number Grade** Whether the barcode was decoded.
Symbol Contrast Grade **Symbol Contrast Number Grade** The fraction of the total image contrast used by the barcode. For the "Number" grade, between 0 (<= 51 gray values) and 4 (>= 180 gray values).
Minimum Reflectance Grade **Minimum Reflectance Number Grade** The ratio between the smallest and largest gray values in the barcode. For the "Number" grade, the value is 4 if the smallest gray value of the barcode is <= half of its highest value, and 0 otherwise.
Minimum Edge Contrast Grade **Minimum Edge Contrast Number Grade** The smallest contrast between two adjacent bar and space, including quiet zones (blank areas to each side of the barcode that are needed for an accurate reading). For the "Number" grade, the value is 4 if the minimum edge contrast is >= 15%, and 0 otherwise.
Modulation Grade **Modulation Number Grade** The importance of the minimum edge contrast compared to the symbol contrast. For the "Number" grade, the value is between 4 (>= 70%) and 0 (< 40%).
Defects Grade **Defects Number Grade** Degree of irregularities found in elements and quiet zones. For the "Number" grade, the value is between 4 (small irregularities) and 0 (large irregularities).
Decodability Grade **Decodability Number Grade** Degree of difference between the measured and expected distances. For the "Number" grade, the value is

Measurement
between 4 (measured distances are close to expected) and 0 (measured distances are far from those expected).
Additional Requirements Grade
Additional Requirements Number Grade
A symbology-specific requirement. Code 128, Gs1 128, and EAN 13 grade the size of the quiet zone of the barcode. For the "Number" grade, the value is 4 (quiet zone is large enough) or 0 (quiet zone is too small).
Global Grade
Global Number Grade
The lowest of all the other grades. If two scan lines yield different decoded strings, Global Grade is set to 0. For the "Number" grade, the value is between 4 (best) and 0 (worst).

Features

Type	Description
Point	The center point of the barcode.

For more information on geometric features, see *Geometric Features* on page 137.

Image Blob

The Image Blob tool lets you detect surface defects, such as uneven or excess material, gouges, or blemishes, on a relatively uniform or flat background. It can also extract targets from the image. The tool optionally lets you set its intensity threshold relative to a user-defined reference region.

The tool first filters image data based on an intensity threshold (above or below it), and then uses configurable morphological operations to better isolate parts of the image. Finally, the tool uses various size- and shape-based filters that let you exclude or include the expected defects or the targets you need (potential blobs).

The tool lets you configure the maximum number of blobs to output and returns the total blob count. For each blob, the X and Y center, the width and length, and the area. The center point of each blob is available as a geometric feature.

Note that knowing the rough size and shape of the kinds of defects you expect is important when you are configuring the open and close kernels and the tool's filters.

When **Enable Array Output** is checked (under the **Parameter** > **Advanced Parameters** section), the tool outputs the blobs in an array called Blob Image. Be *sure* to enable Blob Image in the **Outputs** section after enabling this parameter.

For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Inputs

To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

Name	Description
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Parameters

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Measurement Regions	When expanded, displays the region- and mask-related settings. Enable and configure regions to limit blob detection to specific areas.
Enable	Enables regions and displays the region- and mask-related settings (see below).

Parameter	Description
Mask Mode Number of Regions Region {n}	When you enable regions (see above), the tool displays additional settings related to the region type. For details on the regions supported by this tool and their settings, see *Flexible Regions* on page 133. For general information on regions and the difference between standard and "flexible" regions, see *Regions* on page 132.
Threshold Method	The thresholding method the tool uses to extract blobs. One of the following: **Single Threshold** The tool uses a single threshold, comparing pixel intensities to the value of **Threshold**. Use **Threshold Direction** to choose whether pixels above or below the threshold are included. **Dual Threshold** The tool uses a range of pixel intensity thresholds to extract blobs. Set **Upper Threshold** and **Lower Threshold**. **Otsu** The tool uses the OTSU algorithm to choose the optimal threshold. Use **Threshold Direction** to choose whether pixels above or below the threshold are included. **Local** The tool uses an adaptive threshold in the defined window (**Threshold Window**) using the specified threshold value (**Threshold**) in the input image.
Threshold	The intensity threshold above or below which a pixel is considered for being a blob. Use the **Threshold Direction** setting to determine whether pixel intensities above or below the threshold are considered.
Threshold Direction	Determines whether pixel intensity above or below the threshold is considered as being part of a blob. **Below**: The **Threshold** value is the maximum intensity that will be considered as part of a blob. **Above**: The **Threshold** value is the minimum intensity that will be considered as part of a blob.
Hierarchy	Provides options to let you find either external objects only or all objects. **External Objects** Use this option to ignore smaller objects in larger objects. Only the outermost objects are returned. **All objects** Use this option to include smaller objects in larger objects. **Holes**

Parameter	Description
	Use this option to get holes inside objects. **All Contours** Use this option to get all detected contours.
Use Area Filter	If **Use Area Filter** is enabled, the tool applies an area filter to potential blobs using the values in **Max Area** and **Min Area**.
Use Aspect Filter	If **Use Aspect Filter** is enabled, the tool applies an aspect filter (ratio of length and width) to the rotated bounding box that would encapsulate the area, using the values in **Max Aspect** and **Min Aspect**.
Use Circularity Filter	If **Use Circularity Filter** is enabled, the tool applies a circularity filter to potential blobs to measure how close to a circle the blob is, using the values in **Max Circularity** and **Min Circularity** . Circularity is determined from area within the contour of the blob and the perimeter of its contour. With increasing perimeter for the same area, circularity is reduced.
Use Convexity Filter	If **Use Convexity Filter** is enabled, the tool applies a convexity filter to potential blobs, using the values in **Max Convexity** and **Min Convexity** . Convexity is defined as the (Area of the Blob / Area of its convex hull), and "convex hull" of a shape is the tightest convex shape that completely encloses the shape.
Ordering	Orders the measurements, features, and image data of the individual blobs output by the tool. One of the following: • Position - X increasing • Position - X decreasing • Position - Y increasing • Position - Y decreasing • Area - Large to small • Area - Small to large
Enable Array Output	When **Enable Array Output** is unchecked, the tool outputs each blob as an individual Image output. When **Enable Array Output** is checked, the tool outputs the blobs in an array called Blob Image. Be *sure* to enable Blob Image in the **Outputs** section after enabling this parameter. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Number of Blob Outputs	Determines the number of blobs the tool outputs as measurements, features (center points of blobs), and image data. Currently limited to 200 blobs. If **Enable Array Output** is checked, this parameter is hidden.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Advanced Parameters

Parameter	Description
Open Kernel X Open Kernel Y	The X and Y kernel size, respectively, for morphological opening to remove small areas of data. Use these settings, for example, to remove bridges between areas to

Parameter	Description
	properly isolate them or to remove small areas entirely (perhaps caused by noise). Use different values of X and Y to use a non-rectangular filter to adapt the kernel to the kinds of unwanted data you see in the scan data.
Close Kernel X Close Kernel Y	The X and Y kernel size, respectively, for morphological closing to fill in holes smaller than the specified kernel size. Use these settings, for example, to fill small areas within potential blobs that may be caused by drop-outs. Use different values of X and Y to use a non-rectangular filter to adapt the kernel to the kinds of holes you see in the scan data.
Enable Array Output	When **Enable Array Output** is unchecked, the tool outputs each blob as an individual Image output. When **Enable Array Output** is checked, the tool outputs the blobs in an array called Blob Image. Be *sure* to enable Blob Image in the **Outputs** section after enabling this parameter. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Number of Blob Outputs	Determines the number of blobs the tool outputs as measurements, features (center points of blobs), and image data. Currently limited to 200 blobs. If **Enable Array Output** is checked, this parameter is hidden.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

Most tools provide measurements, geometric features, or data as outputs.

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

All outputs provide an external ID (available by expanding the output in the Outputs panel) for optional use in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

If **Enable Array Output** is checked, the tool outputs arrays for the center position, width, height, and area measurements, as well as the Center Point and Blob Image outputs.

If **Enable Array Output** is unchecked, the tool outputs individual, numbered measurements. Use the **Number of Blob Outputs** setting to determine the number of measurements listed in the **Measurements** tab.

Measurements

Measurement
Total Count
Returns the total number of blobs identified, based on the tool's parameters.
Center X {n} / Center X

Measurement

Center Y {n} / Center Y

The X and Y positions of the center of mass of a blob extracted from the image.

Width {n} / Width

Length {n} / Length

The length and width of the rotated bounding box that encapsulates the blob extracted from the image. These are always the major and minor axis of a blob, respectively.

Area {n} / Area

The area of a blob.

Features

Type	Description
Center Point {n} / Center Point	The point representing the center of a blob.

For more information on geometric features, see *Geometric Features* on page 137.

Data

Type	Description
Diagnostic Image	Image data you can use to evaluate the impact of the tool's parameters, after filters are applied, to properly separate the areas corresponding to the defects or targets you need to detect.
Image {n} / Blob Image	Image data corresponding to each blob. When the tool is configured to output arrays, the blob images are in Blob Image.

Image Circle

The Image Circle tool calculates the best-fit circle for a circular edge in the image and returns measurements of various characteristics of the circle, such as its radius and position. The tool searches along radial paths, whose spacing you define, in the image for changes in intensity values to find the circle.

Inputs

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array, other than the processing limitations of the sensor. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Parameters

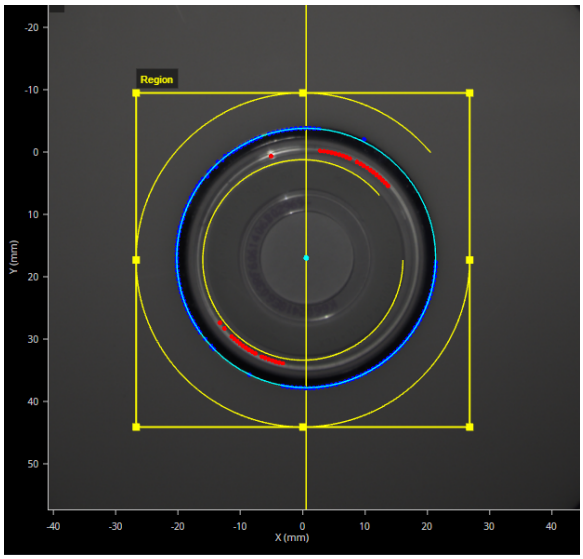
The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Use Region	When enabled, displays Region parameters (see below). When disabled, the tool uses all data.
Region	The region to which the tool's measurements will apply. For more information, see *Regions* on page 132.
Threshold	The threshold intensity value that will be used to determine an edge.
Search Direction	The search direction of the edge search. Either Inward (toward the center of the region of interest) or Outward.
Step Direction	Determines whether the expected step in intensity rises or falls. Note that this setting depends on the **Edge Search Direction** setting for its interpretation of what "rises" and "falls." One of the following: **Rising & Falling**: Searches for rising or falling intensity values. **Rising**: Searches for rising intensity values. **Falling**: Searches for falling intensity values.

Parameter	Description
Edge Selection Type	Determines which step in intensity the tool uses when there are multiple steps. **Best**: Selects the greatest change in intensity in the search direction. **First**: Selects the first change in intensity in the search direction. **Last**: Selects the last change in intensity in the search direction.
Search Range	TBD
Outlier Fraction	The percentage of outlier pixels to exclude. Setting this to a small value can help the tool fit the circle better to the edge of the circle.
Show Advanced Parameters	When enabled, displays advanced settings. Note that most of these settings are applied even when they are hidden. For information on these settings, see *Advanced Parameters* below.
Show Detail	When disabled, hides the calipers and caliper path, as well as the edge points.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Advanced Parameters

Parameter	Description
Angle Start Angle Span	These settings work together to let you set a partial path and exclude part of the image.  **Angle Start** is the starting angle for a circular path along which the tool searches for the edge of the circle. Setting this to 0 aligns the start angle with the positive direction of the X axis. **Angle Span** is the length of the circular path.
Minimum relative change	The minimum change with respect to the value in **Threshold**.

Parameter	Description
Path Spacing	Sets the spacing between paths along which the tool searches for intensity changes. A higher number of paths results in a higher number of edge points, which makes the fitting of the circle more accurate. However, a higher number of edge points results in a greater tool execution time.
Step Smoothing	The size of the window along the radial path used to calculate an average for each pixel along the path. The setting is useful for averaging out noise. If **Step Smoothing** is set to 0, no averaging is performed (only the data point under the path is used).
Mask Regions	Lets you enable up to five regions that you can use to mask data you want the tool to ignore. You can resize and reposition the mask regions using the mouse in the data viewer, or by configuring values manually in the **Mask Region** sections the tool displays in the tool settings for each region. You can only set the rotation of the mask regions manually by modifying the region's **Angle** parameter. By default, when you add multiple mask regions, they are initially placed in the same position, one on top of the other.
Reference Plane	Uses the output of a Surface Plane tool as a reference plane. Useful to correct the scan data if the target is slightly tilted. When **Show Advanced Parameters** is unchecked and **Reference Plane** is set to a plane, the plane is *ignored*.
Fit Type	The method the tool uses to calculate the roundness of the feature. One of the following: **Least Square Circle (LSC)** **Minimum Zone Circle (MZC)**: If you choose this method, set the circle the tool uses with the **Which Circle** parameter. **Maximum Inscribed Circle (MIC)**: Typically used to measure the inner edge of a circular feature, such as a hole. **Minimum Circumscribed Circle (MCC)**: Typically used to measure the outer edge of a circular feature.
Which Circle	Tells the tool which circle (Inner or Outer) to use when Minimum Zone Circle is the fit method in **Fit Type**.

Outputs

All outputs provide an external ID (available by expanding the output in the Outputs panel) for optional use in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Measurements

Measurement
X **Y** The circle center X and Y positions.
Radius

Measurement
Measures the radius of the circle.
Roundness
Returns the roundness of the fitted circle.
Min Error **Max Error**
The minimum and maximum error among the pixels with respect to the fitted circle.

Features

Type	**Description**
Center Point	The center of the fitted circle.
Circle	The fitted circle.
Min Error Point	The point of minimum error which is furthest inside of the circle.
Max Error Point	The point of maximum error which is furthest outside of the circle.

Image Classifier

The Image Classifier tool lets you train a model using classes that you define. For example, for a food processing application, you could define classes such as drumstick, leg, and so on. When scanning targets and using the tool's inference engine, it can determine what type of cut it's seeing.

Inputs

> To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

Name	Description
Enable Batching	Batching (when a tool takes an array as input and processes each element in the array individually) is currently not fully supported by this tool. If you choose to use batching, keep the following limitations in mind: • The Label page only shows the first element of the input array. • For anomaly detection, blobs representing the anomalies are not displayed. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Development Workflow

The high level workflow is as follows:

1. Collect representative scans of the targets you need to inspect in order to train a model. In addition to "good" parts, you need to scan "bad" parts, that is, ones with typical defects.
2. Save the scans to a .gprec file.
3. Create a pre-processing toolchain and train a project's model.
4. Deploy the model and connect post-processing tools.
5. Optionally, collect new production data for model retraining.

Collect dataset

Unless you are using a previously created dataset, you must create the dataset that the tool will train a model on before it can process new production data. To do this, perform the following steps:

1. Acquire 20 or images that represent each of the classes you intend to add to the project.

   You can acquire the data in a single .gprec file or to multiple files. If you record scans to a single file, grouping each type together will make it easier to label them later in the wizard. Make sure there are no

empty frames. Do not rely on .gpsup files for this as the Anomaly Detector wizard requires .gprec files when importing from PC.

2. Save a separate .gpsup file with the device configuration settings such as exposure time and related imaging parameters.

   Ensure the same parameters are used on the production system. Otherwise, you will need to collect new training data.

Implement a Preprocessing Toolchain

You typically need to create a preprocessing tool chain for your application. The preprocessing tool chain limits variation in the scan data to the region or surface of interest. With less variation in the scan data, the total number of frames needed for training is reduced.

The preprocessing tool chain must not change between training and inference. Otherwise, the system will not function properly. You must save your preprocessing toolchain in the .gpsup file created previously or using a separate .gpjob file.

> You *must* use the same preprocessing toolchain for training and inference to ensure proper detection of anomalies.

Train Model

Add the Tool

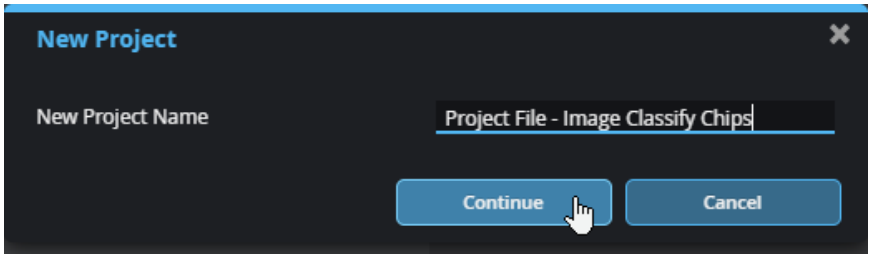
1. Add and connect the tool to the output of the preprocessing tool chain.
2. In tool's **Parameters** section, click the **Train Model** button.

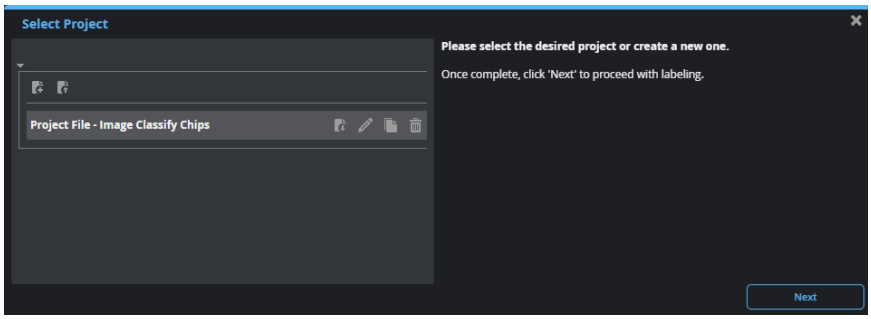
   The routine starts and the **Select Project** page opens. You will create a new project in this page, or open a previously created project.

Create New or Open Project

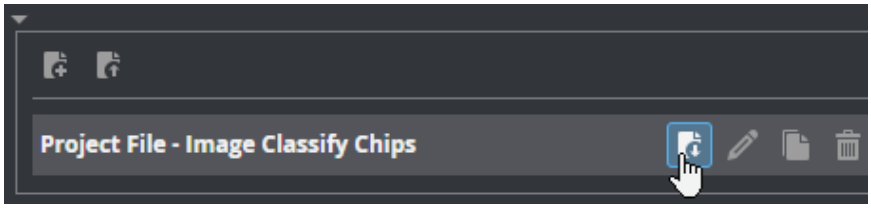
A project is a collection of frames used for training, classes and labels you apply to those frames, and the models trained on those frames. You should choose a meaningful project name, such as the part name, especially when working with multiple projects.

1. On the **Select Project** page, click the New Project icon, provide a project name, click **Continue**, and then click **Next**.

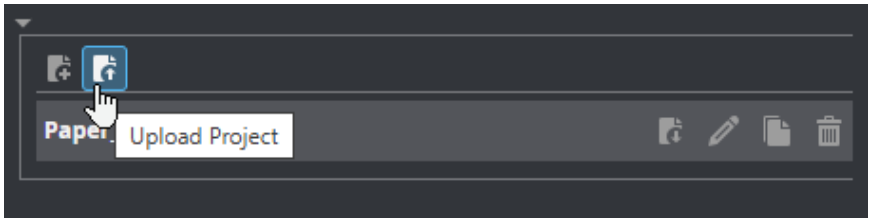




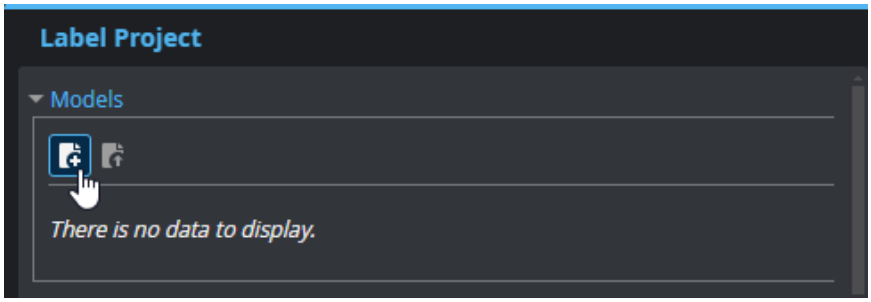
Use the download icon on a project's row to download this to your PC and upload to another device.

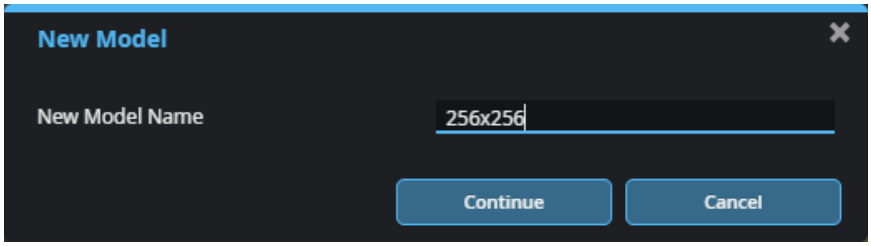


Use the upload icon to import a previously downloaded project.

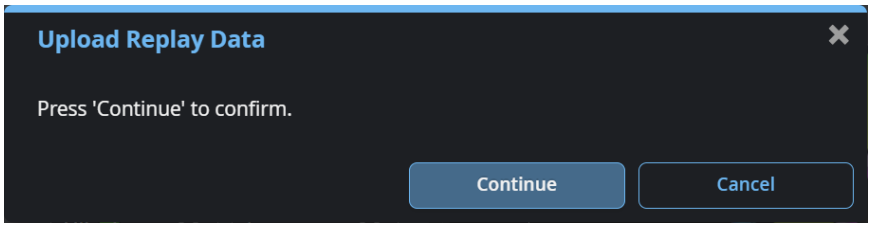


2. In the **Label Project** page, in the **Models** section, create a new model and click **Continue**.





3. In the **Frames** section, do one of the following
   - To upload currently loaded replay data, click the Upload Replay Data button ( ) , and then click **Continue**.



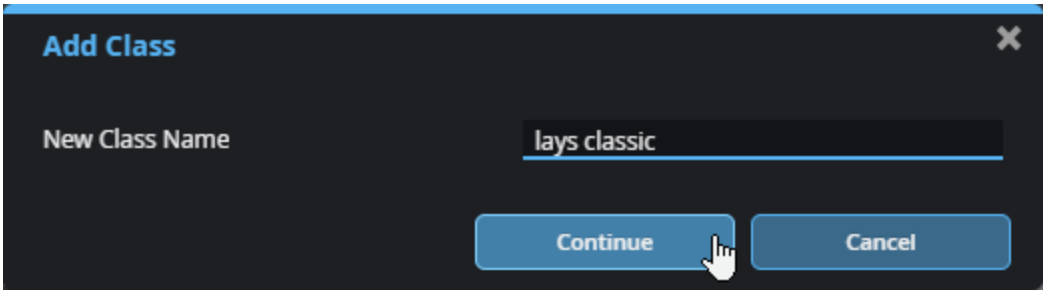
- To upload the data from a .gprec file on the local PC, click the Upload From Disk button ( ), choose a file in the Open dialog a, click Open.

The tool loads the frames into the model.

⚠	Loading replay data into the project can take several minutes. If a “Delayed Response” message displays, click “Wait.” In this case, add a Decimation Filter to the pipeline.

Add Classes

You must now add classes to the project. For example, in a snack food processing line, you might create classes for "lays classic", "cheetos", "doritos", and "ruffles". For each class, click the Add Class button ( ), provide the name of the class, and click **Continue**.



Label Frames

You must now assign a class label to each frame.

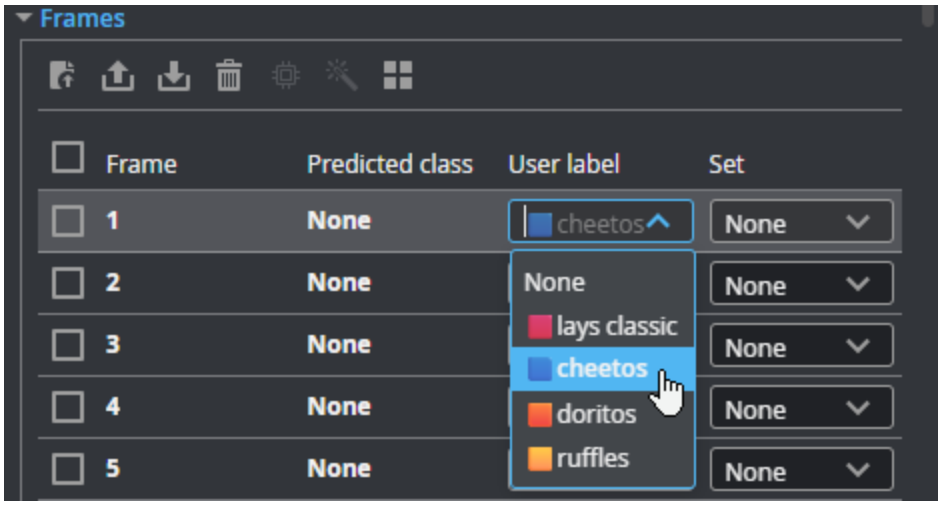
By default, the tool chooses the set each frame belongs to (Train or Test, under the **Set** column), based on the following proportions:

70% of a class in Train

30% of a class in Test

If you choose to disable Auto-split and assign frames to sets manually, use the recommended proportions.

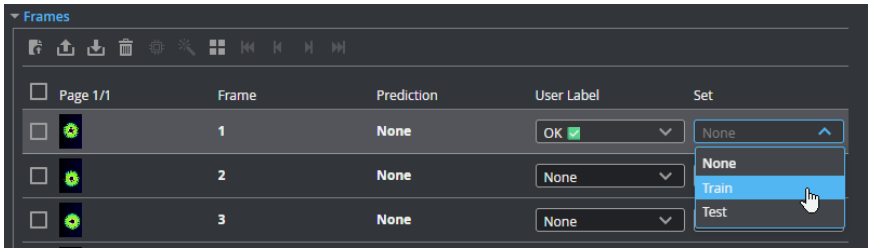
1. In the row of each frame, click **User Label** and choose the correct label for the frame.

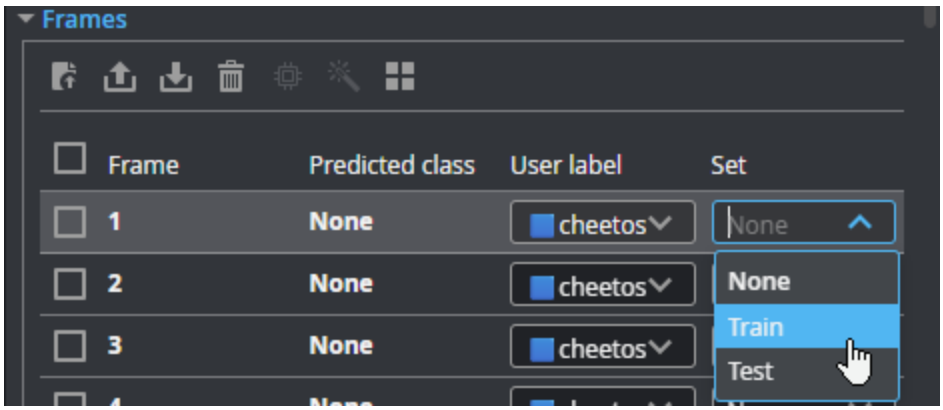


You can label multiple instances of a class in a frame, and apply labels for more than one class.

The tool adds a label instance and displays a bounding box for that instance.

If you have disabled auto-split, also choose the frame's set under **Set**. Make sure you use the proportions described in step .

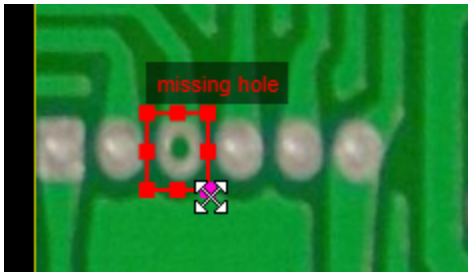




To quickly choose the label and set for multiple frames, do one of the following to first select the frames:

- To select a range of frames, click the checkbox of the first frame, press Shift, and click the checkbox of the last frame of the range.
- To select multiple frames, click the checkbox of the first frame, press Ctrl, and click the checkbox for each frame you want to select.
- To select non-contiguous ranges, alternate between clicking Ctrl and Shift.

After selecting the frames, choose the label and the set as described above. To deselect all frames, click anywhere between the checkbox and the User Label and Set drop-down menus.

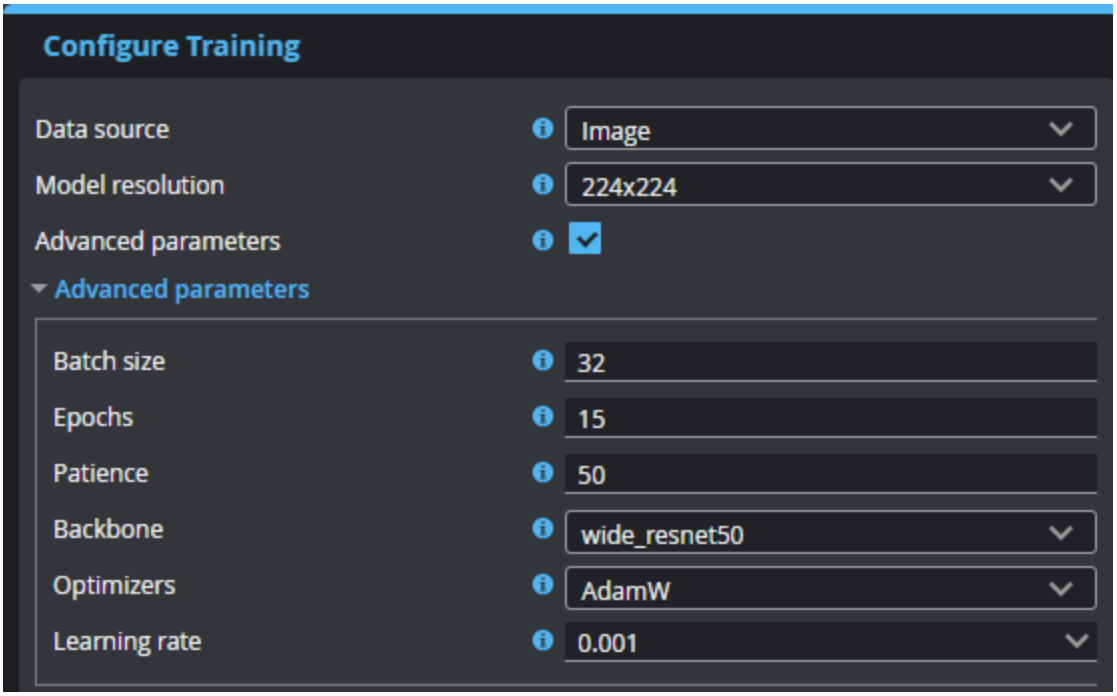


2. Click **Train Model** to configure the training parameters.

   For information on configuring training, see *Configure Training* below.

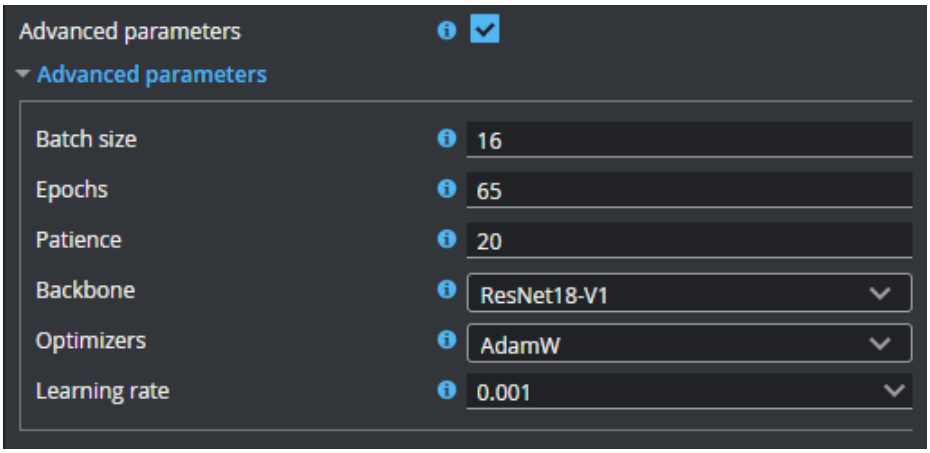
Configure Training

After labeling the frames and choosing the set they belong to, you must configure training. For descriptions of the settings, see below.



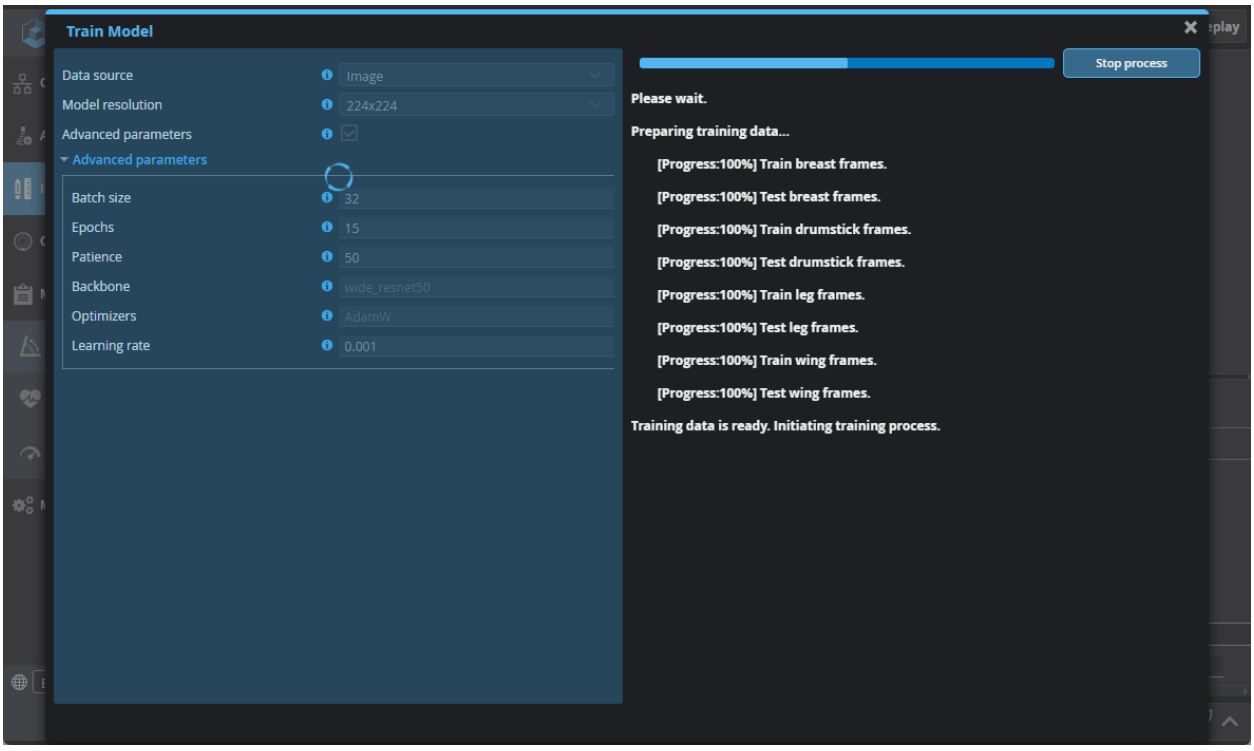
Training Parameters

Parameter	Description
Data source	The source the tool uses during training.
Model resolution	Decimates the input data to the specified resolution for processing. A higher resolution improves anomaly location accuracy but increases processing time for training.For most applications, leave this at 224x224.
Advanced parameters	If the training results are generally good but the tool is reporting a small number of false positives or false negatives, you can modify these parameters to further

Parameter	Description
	improve results.  **Batch size** The number of frames in a batch (a subset of the entire dataset passed to the training algorithm for an iteration). **Epochs** The number of times the entire dataset passes through the training algorithm. To reduce incorrect classifications, increase this value. Note that training will take longer. **Patience** The number of epochs to wait without improvement in validation metrics before stopping the training early to prevent overfitting. As a guideline, start with a value around 10-20% of total epochs. **Backbone** The feature extractor backbone used by the tool. Select pre-trained model weights for training. One of the following: ResNet18-V1: A lighter model that is fast and a good starting point. ResNet18-V2: A medium model with more capacity than V1. ResNet50: The best balance for most tasks. Wide ResNet50-2: Slow and larger memory, with better for complex textures. **Optimizers** Algorithms that adjust the model's weights and biases to minimize the loss (error)

Parameter	Description
	and improve its accuracy. One of the following: Adam: Best for most cases. AdamW: Slightly slower than Adam but better at preventing overfitting. Use this when the model performs well during training but fails on new images. SGD: Maximum accuracy, requires large datasets; use this when you need that extra 1%-2% accuracy and have time to tune. **Learning rate** Set this to a higher value to reduce training time. Note that a value that is too high can cause the model to miss the point where the difference between predicted and actual OK / NG is at its smallest (that is, overshooting it). A lower value causes the model to learn more slowly, but potentially more accurately; if the value is too low, the model can get stuck.

After you configure training, click **Start Training** to train the model. The tool will start by preparing the training data, providing progress feedback:



After the data is prepared, the tool will start training and generating graphs of metrics to chart the progress.

Loss: This metric represents the cumulative error for each frame in the Test set, determined after each training run iteration or epoch. To put it another way, it's the difference between the model's

current prediction and the actual label you provided (sometimes called the "ground truth"). A decreasing number (the graph's line is moving down) means the model is improving.

Accuracy: This metric displays the portion of total correct predictions (value between 0 and 1).

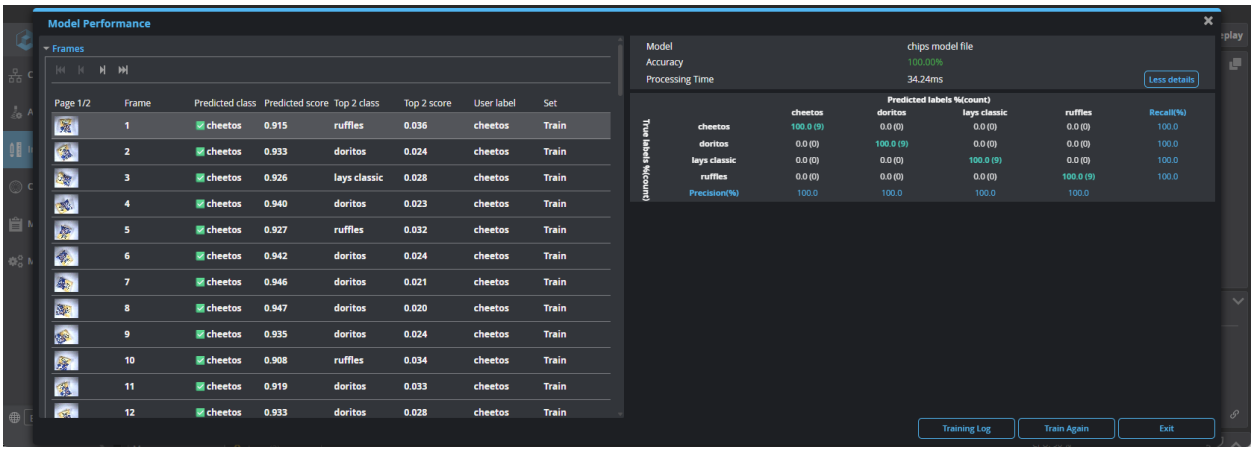
After the training has completed, click **Next** to test the model. Note that training can take several minutes.

Model Performance Testing

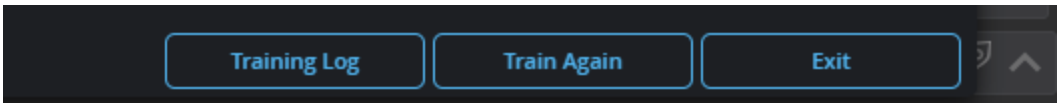
After the tool has finished training the model, it will use the model to make predictions on the frames to determine a score for each frame. The Model Performance page displays the following results:

Performance Results

Name	Description
Model	The trained model the tool used during the model performance phase.
Processing Time	The average time required for inference. Includes the pre-processing tool chain.
Accuracy	A percentage representing the overall accuracy of the model's class predictions.



The Model Performance page also lets you download an HTML file containing reports and other diagnostics. LMI recommends checking the visual results in this file.



You can also retrain the model with different training configurations if you wish.

You are now ready to use the trained model on new scan data in production.

Parameters

In the **Parameters** section of the tool, you can launch the Train Model wizard, in which you can create a project, as well as the models, classes, and frames that the project includes. After you have created the project, the Train Model wizard lets you train the model on the frames. Once training is

complete, you can use the project as an inference engine to classify new frames of date, that is, on the production line. For information on creating a project and training its model, see XYZ.

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Model	The trained model the tool uses to classify images during production.
Classes	A read-only list of the classes defined in the selected model.
Additional predictions	When this parameter is enabled, the tool also outputs arrays containing lists of the project's classes and IDs, as well as the scores associated with the classes / IDs, for the current frame. The array items are ordered from highest to lowest score. For more information, see *Measurements* below.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

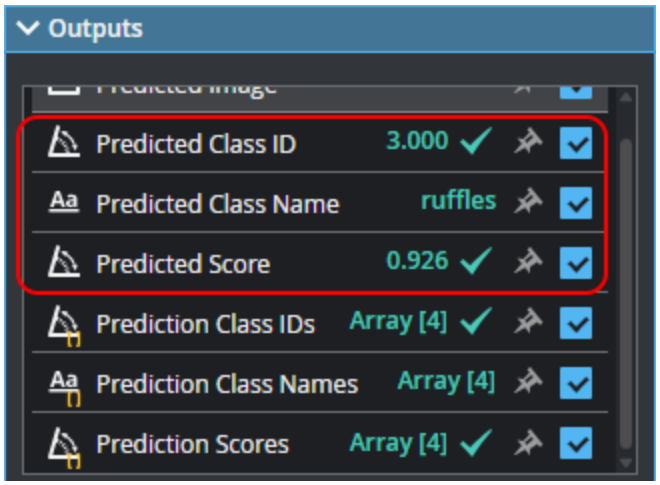
All outputs provide an external ID (available by expanding the output in the Outputs panel) for optional use in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Note that some measurements return IDs, whereas others return class names. The IDs correspond to the class names added in the project. So for example, given the classes "lays classic", "cheetos", "doritos", and "ruffles", the corresponding IDs are 0, 1, 2, and 3. The ID values are ordered based on the order in which the classes were added in the project.

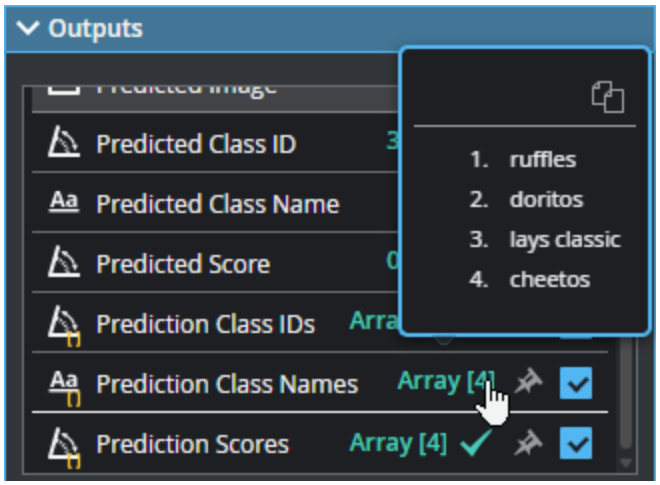
Measurements

Measurement
Predicted Class Id The ID of the highest scoring predicted class of the image.
Predicted Score The score associated with the predicted class.
Prediction Class IDs
Prediction Class Names
Prediction Scores These measurements and outputs are only displayed if **Additional predictions** is enabled. The Prediction Class IDs and Prediction Class Names outputs return arrays containing the project's class IDs and class names, ordered based on their score in decreasing order. The Prediction Scores array returns an array containing the scores of the image, ordered from highest to lowest. For example, let's say the tool has classified an image of a bag of snack food as "ruffles" (ID = 3), rather than any of the other classes ("lays classic", "cheetos", or "doritos"), with a score of 0.894.

Measurement

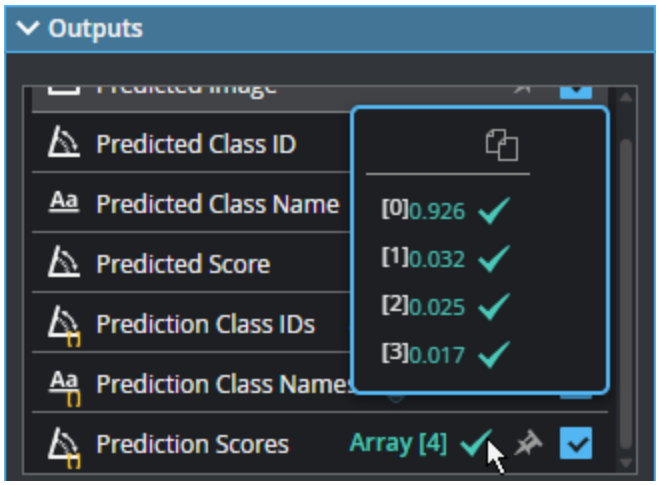


The Prediction Class Names output contains the names of classes in the project ordered from the highest to lowest score the tool assigned to the current frame. In the following, "wing", the predicted class name for the frame's image, is the first item in the array.



The scores corresponding to those ordered classes are available in the Prediction Scores output. The following shows the corresponding scores. The score for "ruffles" (0.926) is the first item. The next value, 0.032, indicates prediction score for "doritos", and so on.

Measurement



Use these alternate candidate predictions to determine if the model needs specific additional training to better classify images.

Data

Type	Description
Predicted Image	The frame's image with an overlay displaying the class names and their corresponding prediction scores.

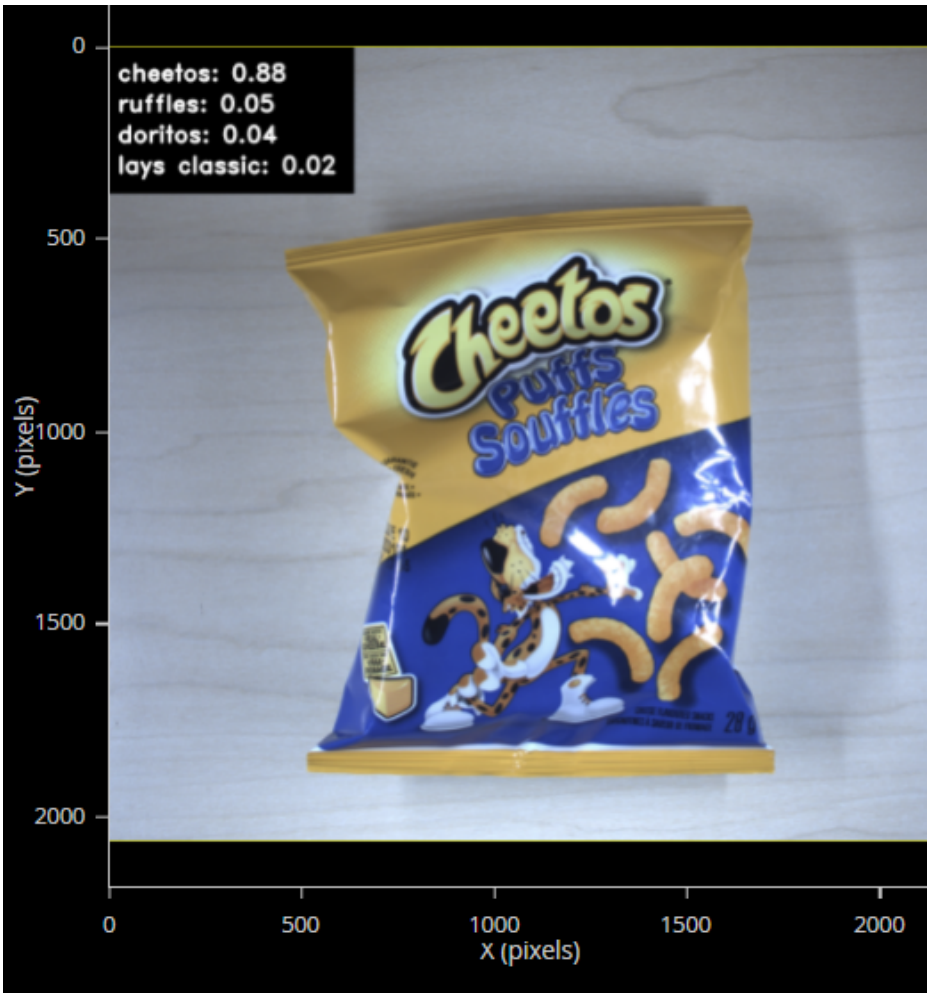
Type	Description
	
Predicted Class Name Prediction Class Names	The Predicted Class Name output contains the text of the frame's predicted class name. You can output this to a Script tool or to a PLC to control other devices based on the class name. The Prediction Class Names output contains an array of the project's class names, ordered by their score for the current frame.

Image Converter

The Image Converter tool lets you convert 8-bit RGB images to grayscale images, and from grayscale to RGB. Note that 16-bit images are not supported.

Inputs

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array, other than the processing limitations of the sensor. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Image Input	The data the tool applies measurements to or processes.

Parameters

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Converter Type	The type of conversion. One of the following: RGB to Gray, Gray to RGB. The tool only supports 8-bit images.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

All outputs provide an external ID (available by expanding the output in the Outputs panel) for optional use in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Data

Type	Description
Converted Image	The converted image.

Image Feature Mask

The Image Feature Mask tool dynamically defines a polygon or circle mask using Point geometric features produced by other tools to crop an image and output it.

When only two Point geometric features are passed as input to the tool, it creates a circular mask, where one point is the circle's center point, and the other is a point on the circle.

For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Inputs

You configure the tool's inputs in the expandable **Inputs** section.

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array, other than the processing limitations of the sensor. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Point {n}	The nth Point geometric feature. Only displayed if you set **Point Count** to 3 or higher.
Center Point	The center Point geometric feature of the circle. Only displayed if you set **Type** to Circle and **Point Count** to 2.
Edge Point	The Point geometric feature on the circle. Only displayed if you set **Type** to Circle and **Point Count** to 2.
Image Input	The data the tool applies measurements to or processes.

For more information on geometric features, see *Geometric Features* on page 137.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Parameter	Description
Type	The shape type of the mask. It supports two options: **Polygon** - The tool needs at least 3 Point geometric features. **Circle** - The tool needs 2 or 3 point geometric features.
Point Count	The count of the input point features. For the Polygon **Type** option, the tool supports integers in the range of 3 to 10. For the Circle **Type** option, the tool supports integers of 2 or 3. After setting this parameter, set the point inputs in the Inputs section to the appropriate Point geometric features.

Parameter	Description
Padding Distance	The offset from the mask. When the value is greater than 0, the offset is outward. When the value is less than 0, the offset is inward.
Show Detail	Toggles whether to display the graphics of the mask region.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

Most tools provide measurements, geometric features, or data as outputs.

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

All outputs provide an external ID (available by expanding the output in the Outputs panel) for optional use in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Data

Type	Description
Segmented Image	The cropped Image data.

Image Filter

The Image Filter tool provides several common vision processing filters that you can apply to the camera images, as well as two "cropping" filters that output a part of the image, letting you pre-process images to get more repeatable measurements. Any Image tool can use the resulting filtered image as input.

For a list of the filters, see *Filters* on the next page.

The Filter tool provides no measurements or decisions: it only outputs processed image data.

For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Inputs

You configure the tool's inputs in the expandable **Inputs** section.

> To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array, other than the processing limitations of the sensor. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Parameter	Description
Regions	When expanded, displays the region- and mask-related settings. Check Enable to configure regions.
Mask Mode Number of Regions Region Type {n} Region {n}	When you enable regions (see above), the tool displays additional settings related to the region type. For details on the regions supported by this tool and their settings, see *Flexible Regions* on page 133. For general information on regions and the difference between standard and "flexible" regions, see *Regions* on page 132.

Parameter	Description
Filter Type	The type of the filter. For information on the available filters, see *Filters* below.
Threshold	The threshold that the filter uses. (Not available on all filters.)
X Window Size Y Window Size	SThese parameters enable filtering along the X and the Y axis, respectively. Set the corresponding window sizes in **X Window Size** and **Y Window Size**.
Preserve Data Outside Region	Enable this setting when **Mask Mode** is set to **Include Data in Region** to include data outside the region in the Filtered Surface data output. Otherwise, the Filtered Surface data output only includes the data in the region. This setting is only displayed if regions are enabled.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

The following filters are available in the Filter tool.

Filters

Name	Description
Averaging	Substitutes a pixel's value with the mean value of that pixel and its nearest neighbors within the window or windows set in **X Window Size** or **Y Window Size**. . . If both X and Y smoothing are enabled, the data is smoothed along X axis first, then along the Y axis. Missing pixels will not be filled with the mean value calculated from pixels in the neighborhood.
Gaussian	A Gaussian filter. Set the **Sigma X** and **Sigma Y** parameters as required. Sigma X: Set to 0 to let the filter automatically calculate sigma based on the kernel size, using sigma = 0.3 * ((ksize-1) * 0.5 - 1) + 0.8 Sigma Y: Set to 0 to let Sigma Y be equal to Sigma X .
Median	Substitutes the value of a pixel with the median calculated within the window or windows set in **X Window Size** or **Y Window Size** around the pixel. Missing data points will not be filled with the median value calculated from data points in the neighborhood. With an odd-numbered window size, the output is at the center of the window. With an even window size, the output is 0.5 pixels to the right of the center (that is, using window / 2-1 values from the left, and window / 2 from the right.
Decimation	Decimation reduces the resolution along the X or Y axis, keeping pixels starting at the left edge of the specified window. When the Image data is acquired with a non-calibrated camera, set **X Window Size** and **Y Window Size**. On a calibrated camera, set **Factor** to the desired value; for example, a value of 4 keeps every fourth pixel.
Erosion	Applies an erosion filter.

Name	Description
Dilation	Applies a dilation filter.
Opening	Erosion followed by dilation.
Closing	Dilation followed by erosion.
Morph Gradient	Applies a morphological gradient. The difference between dilation and erosion.
Sobel Magnitude	Applies a Sobel magnitude filter.
Laplacian	Applies a Laplacian filter. Useful for detecting areas of distinct edges. Set Kernel Size to an odd number less than 31.
Negative	Inverts the pixel values in the image.
Equalize	Improves contrast of the image.
Binarize	Converts the image to a binary image by comparing pixels with the threshold.
Percentile	Limits the scan data to pixels between the values you set in **High Percentile** and **Low Percentile**.
Relative Threshold	Crops scan data relative to user-specified intensity thresholds (High Threshold and Low Threshold). Pixels between the thresholds are included in Filtered Image.
Crop only	Crops the scan data to the user-defined region.

Outputs

Most tools provide measurements, geometric features, or data as outputs.

Data

Type	Description
Filtered Image	The filtered image, available for use as input in the **Input** drop-down in other tools.

Image Intensity

The Image Intensity tool returns average, minimum, maximum, and median intensity values, as well as the contrast. The tool lets you choose whether to combine pixel colors using the formula described above, to output a single red, green, or blue channel, or an array. Optionally, you can output an array containing a histogram.

Note that when the tool returns array results, the order of the color channels is BGR.

Inputs

> To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

Name	Description
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Parameters

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Regions	When expanded, displays the region- and mask-related settings.
Enable	Enables regions and displays the region- and mask-related settings (see below).
Mask Mode Number of Regions Region Type {n} Region {n}	When you enable regions (see above), the tool displays additional settings related to the region type. For details on the regions supported by this tool and their settings, see *Flexible Regions* on page 133. For general information on regions and the difference between standard and "flexible" regions, see *Regions* on page 132.
Color operation	One of the following: Combine, Red, Green, Blue, Array. **Combine** Converts RGB pixel colors to grayscale using the formula described above. **Red** **Green** **Blue** Returns the selected color channel only.

Parameter	Description
	Array Returns an array containing the values of the color channels in BGR order.
Min percentile Max percentile	Limits calculations to the pixels whose intensities fall between Min percentile and Max percentile.
Output histogram	When enabled, outputs a histogram of the image. Set the number of bins in
Histogram bin count	The number of bins into which the pixel intensities are combined in the histogram.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

All outputs provide an external ID (available by expanding the output in the Outputs panel) for optional use in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

The **Color operation** parameter determines which pixels the tool uses for the outputs.

Measurements

Measurement
Average The average pixel intensity.
Minimum **Maximum** The minimum and maximum pixel intensity values.
Contrast The contrast of the pixels, based on the mean and standard deviation of the pixels.
Median The median of the pixel intensities.
Histogram When the **Output histogram** parameter is enabled, the tool outputs an array representing the histogram of the image, using the number of bins set in **Histogram bin count**.

Image Line

The Image Line tool fits a line to a step in an image, that is, rising or falling intensity transitions greater than the provided threshold.

Inputs

> To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array, other than the processing limitations of the sensor. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Parameters

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Number of Regions	The number of regions the tool uses. You can use multiple regions to fit a line to non-contiguous features.
Region {n}	When expanded, displays the region-related settings.
Threshold	The intensity threshold above or below which a pixel will be considered a step point. On color devices, the RGB values are converted to an 8-bit monochrome image.
Search Direction	The search direction for steps. Can be 0, 90, 180, or 270 degrees. Choose a value that is roughly perpendicular to the edge on the target. The direction is indicated by a light blue arrow in the data viewer.
Step Direction	Whether the tool looks for increasing intensity values (Black to White option) decreasing values (White to Black option) intensity values, or both.
Selection Type	One of the following: Best, First, or Last. Determines whether the best transition (greatest intensity difference), or the first or last transition exceeding the value of **Threshold** is used as a step point.
Show Detail	Displays or hides step points and the direction arrow.

Parameter	Description
Advanced Parameters	When expanded, displays advanced parameters. **Outlier Fraction** The percentage of outlier points to exclude. Setting this to a small value can help the tool fit the line better. **Minimum relative change** The minimum change with respect to the value in **Threshold**. **Path Spacing** The spacing between sample paths perpendicular to the region's center line, along which the tool searches for steps. Smaller spacing increases processing time. **Step Smoothing** The size of the 1D window in which the tool applies an average to smooth pixel intensity values. The setting is useful for averaging out noise.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

All outputs provide an external ID (available by expanding the output in the Outputs panel) for optional use in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Measurements

Measurement
X **Y** The X and Y position of the intersection between the fitted line and the center line of the region.
Angle The angle of the line.
Step Intensity The average of the differences in intensity in the steps to which the tool fits a line.
Point Count The number of valid points used to fit the line.
Min Error **Max Error** The minimum and maximum error among the valid data points with respect to the fitted line (the minimum and maximum distances).

Data

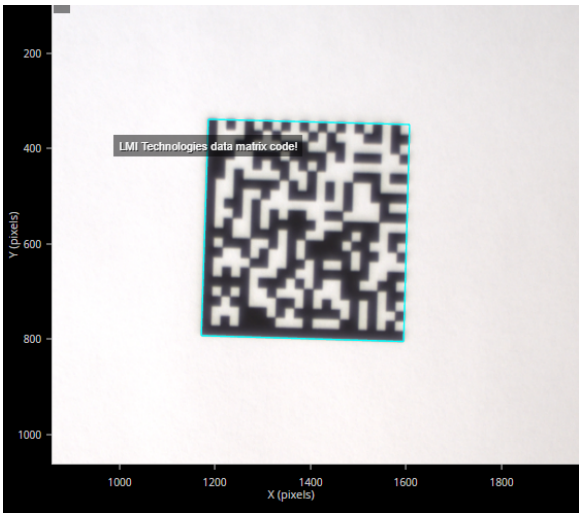
Type	Description
Edge Line	A Line geometric feature representing the fitted line.
Center Point	A Point geometric feature.

Image Matrix Code

The Matrix Code tool lets you decode data matrix codes in Image data. The tool only supports grayscale images.

The tool returns the decoded string, as well as other measures such as its dimensions and the ECC family to which it belongs. The tool also provides measurements that can help you assess the print quality of the code symbols. Indicators from the following standards are available:

- ISO 15415
- ISO 29158
- SEMI T10



Inputs

> To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array, other than the processing limitations of the sensor. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Parameter	Description
Use Region Region	Enable **Use Region** and set the values for the regions.
Instance Count	The maximum number of matrix codes to decode in a frame.
Grading Standard	The standard the tool will use to assess the quality of the matrix code. Grading measurements are added in the tool's **Output** section. One of the following: • ISO 15415 • ISO 29158 • SEMI T10
Compute Grading	The type of grading the tool returns in the grading measurements. One of the following: **Number** - Grade measurements return a numerical value between 0 and 4. **Letter** - Grade measurements return A, B, C, D, or F. **Letter and Number** - Both types of outputs are returned. Numerical measurements are distinguished from letter-based ones by "Number" in the name.
Read Mode	Whether the tool prioritizes speed or detection quality. **Speed** The tool stops searching for matrix codes as soon as one of the following happens: • It finds the number of codes selected in **Instance Count**. • It has finished processing the image. This mode results in shorter processing times. **Quality** The tool tries to improve its detection until it has finished processing the image. This mode results in better grading results.
Enable Datamatrix Rectangular Extension	Enables decoding of Datamatrix Rectangular Extension (DMRE).
Advanced	Expandable section containing advanced settings. **Decode Damaged Code** Whether to allow permissive decoding (decode codes containing partially incorrect

Parameter	Description
	information) and return unreliable codes (objects that are likely data matrix codes, but can't be decoded). **Set Code Dimensions** Sets the minimum and maximum number of pixels allowed for the sides of a data matrix code. Matrix codes with dimensions outside the range will not be detected. Set **Min Dimension** and **Max Dimension** to the required number of pixels. These dimensions override the dimensions resulting from using the optimization engine and reset the engine. **Optimization Engine** Place a representative matrix code under the camera, and click **Learn image** to determine the best options to use to read matrix codes. Click **Reset** to clear.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

Most tools provide measurements, geometric features, or data as outputs.

Outputs section with a measurement expanded to show user-configurable decision min/max fields and an external ID

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

Some measurements are only displayed when you select a grading standard in **Compute Grading** (see *Parameters* on the previous page). "Modules" refers to the cells or "dots" in the grid of the matrix code.

Measurements (all grading standards)

Measurement
Total Count The number of matrix codes decoded.
Decoded String The decoded string, packed as an array of float-point values.
Symbol Height **Symbol Width** The number of vertical and horizontal cells in the matrix code, respectively.
Errors Corrected The number of corrected modules.

Measurement
ECC Family The family the matrix belongs to (for example, 200 means the code belongs to the ECC200 family).
GS1 Encoded Returns 1 if the decoded string uses the GS1 standard.
Reliability A confidence score related to whether the code is actually a data matrix.

When **Compute Grading** is set to "Letter and Number", the tool outputs both "Number Grade" measurements containing a value between 4 and 0 and "Grade" outputs that contain A, B, C, D, or F. If the setting is either "Letter" or "Number", the tool only returns "Grade" outputs, which contain the selected type.

Measurements (ISO 15415)

Measurement
Decode Grade **Decoding Number Grade** Whether the code was decoded.
Axial Non Uniformity Grade **Axial Non Uniformity Number Grade** Grade for uneven scaling along the X and Y axis.
Grid Non Uniformity Grade **Grid Non Uniformity Number Grade** How much the measured position of the grid intersection point deviates from the expected position. Codes sheered along the X and/or Y axis can result in a low grade. The grade can be unstable and its usefulness is limited.
Symbol Contrast Grade **Symbol Contrast Number Grade** The difference between the dark and light areas of a code, including the quiet zones.
Modulation Grade **Modulation Number Grade** Uniformity of the reflectance of dark and light areas throughout the code. Irregular or mottled regions can result in a low grade.
Reflectance Margin Grade **Reflectance Margin Number Grade** How distinguishable the modules are relative to the global threshold (the mean reflectance of the brightest and the darkest module).
Fixed Pattern Damage Grade **Fixed Pattern Damage Number Grade**

Measurement
A measure of any damage (missing data or data where it is not expected) to the finder pattern (the L-shaped along the left and bottom of the code), the quiet zone (margins outside the matrix), or the clock track (the alternating "dots" along the top). The value returned is an average of the grades of these areas of concern. A higher value indicates the matrix code is more correctly located and identified.
Unused Error Correction Grade **Unused Error Correction Number Grade** Grade representing how little error correction to reconstruct data in damaged or poorly printed codes was used. A higher grade means less error correction was applied.
Horizontal Print Growth **Vertical Print Growth** **Horizontal Print Growth Number** **Vertical Print Growth Number** How much the matrix code may have grown or shrunk compared to the expected size horizontally and vertically, respecitively.

Measurements (ISO 29158)

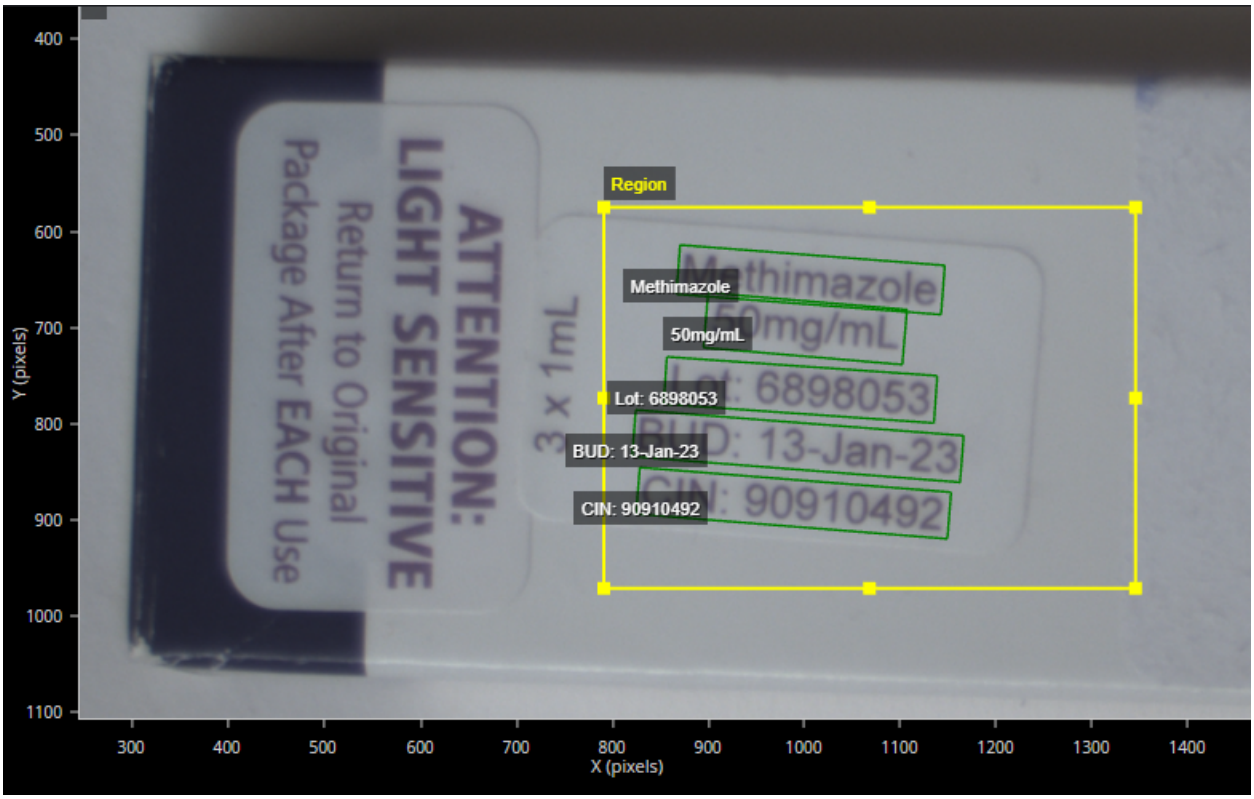
Measurement
Cell Contrast Grade **Cell Contrast Number Grade** Relative difference between the mean reflectance of the darkest and brightest cells in the code.
Cell Modulation Grade **Cell Modulation Number Grade** The variability and reliability of the black and white colors. A low grade means the tool will have difficulty distinguishing between black and white cells.
Minimum Reflectance Grade **Minimum Reflectance Number Grade** The ratio between the smallest and largest gray values in the barcode. For the "Number" grade, the value is 4 if the smallest gray value of the barcode is <= half of its highest value, and 0 otherwise.
Fixed Pattern Damage Grade **Fixed Pattern Damage Number Grade** How likely the symbol is correctly located and identified in the code.
Mean Light In Required Bounds **Mean Light In Required Number Bounds** Whether mean light is 70%~86% of the maximum gray level.
Mean Light **Mean Number Light** The mean value of the pixels in the center of the light cells.

Measurements (SEMI T10)

Measurement
Symbol Contrast SNR **Symbol Contrast SNR** The relative strength of the noise in the code. A higher grade means less noise is present.
Vertical Mark Growth **Vertical Mark Growth** **Horizontal Mark Growth** **Horizontal Mark Growth** The relative horizontal and vertical size of a mark (dark cell) compared to a space (white cell), respectively. The highest grade means cells are the same size.
Vertical Mark Misplacement **Vertical Mark Misplacement** **Horizontal Mark Misplacement** **Horizontal Mark Misplacement** The relative distance between the expected positions cell centers and their measured positions.
Data Matrix Cell Height **Data Matrix Cell Height** **Data Matrix Cell Width** **Data Matrix Cell Width** The height and width of the cells.
Finder Pattern Defects **Finder Pattern Defects** Similar to Cell Defects, but limited to L-shamped finder pattern. The tool may not recognize a code as valid if this is too high.
Cell Defects **Cell Defects** Ratio of pixels that have the wrong color to the total number of pixels in the code. Only returned if the code can be decoded.

Image OCR

The Image OCR tool lets you perform optical character recognition to extract strings of text from Image data. The tool lets you optionally set the number of expected lines, as well as the expected structure and composition of the text (number and types of characters, and so on) to improve recognition. The tool is font-independent and already trained. Text can have any orientation in the image.



For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Inputs

> To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

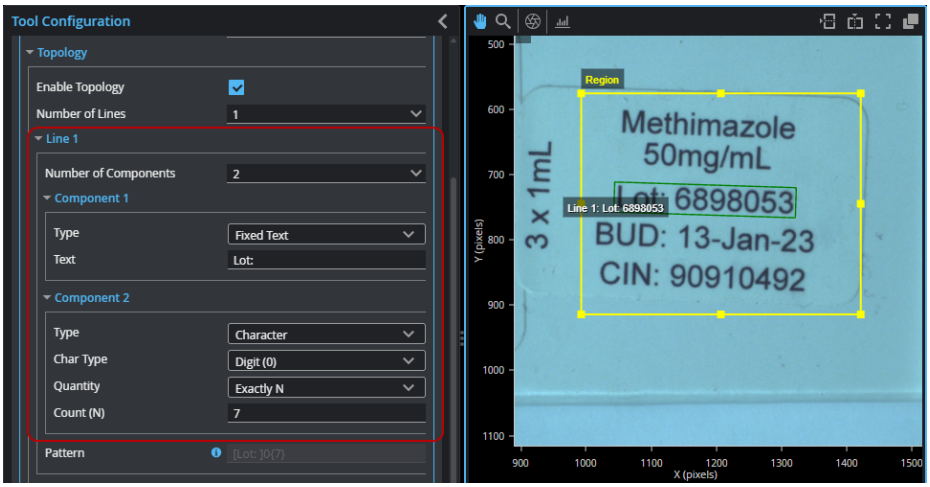
Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array, other than the processing limitations of the sensor. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Parameters

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Use Region Region	When **Use Region** is enabled, configure the region in the **Region** section.
Instance Count	The expected number of occurrences of recognized text to be returned by the tool. If multiple instances are found, these are returned as an array.
Topology Enable Topology	Expand **Topology** and check **Enable Topology** to configure a topology. A topology lets you specify the expected structure and composition of the text, that is, the number of lines, and for each line, the expected types and number of characters, as well as any fixed text the line will contain. Configuring a topology can improve the reliability of the tool's optical character recognition and output. **Number of Lines** The expected number of lines in the instance. For each expandable **Line {n}** section, you set the number of components (**Number of Components**), and then the composition of the components. **Type** One of the following: Fixed Text or Character. When set to Fixed Text, you provide some expected text. Text that is expected in each frame or instance (such as "Lot:" or "Exp. date:") can help improve character recognition. When you set **Type** to Character, you provide the character type (**Char Type**) and the number of characters (an exact number, or a range). For example, the following topology settings capture a line starting with the text "Lot: ", followed by exactly 7 digits.

Parameter	Description
	 The character types correspond to the following: • A: an uppercase ASCII letter • a: a lowercase ASCII letter • 0: a digit • P: an ASCII punctuation character • V: a character belonging to either A or 0 • X: a character belonging to either A, a, 0 or P • _: a space • S: a character belonging to either X or a space **Pattern** This represents the search string, as represented by the Euresys simplified format.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

Outputs section with a measurement expanded to show user-configurable decision min/max fields and an external ID

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

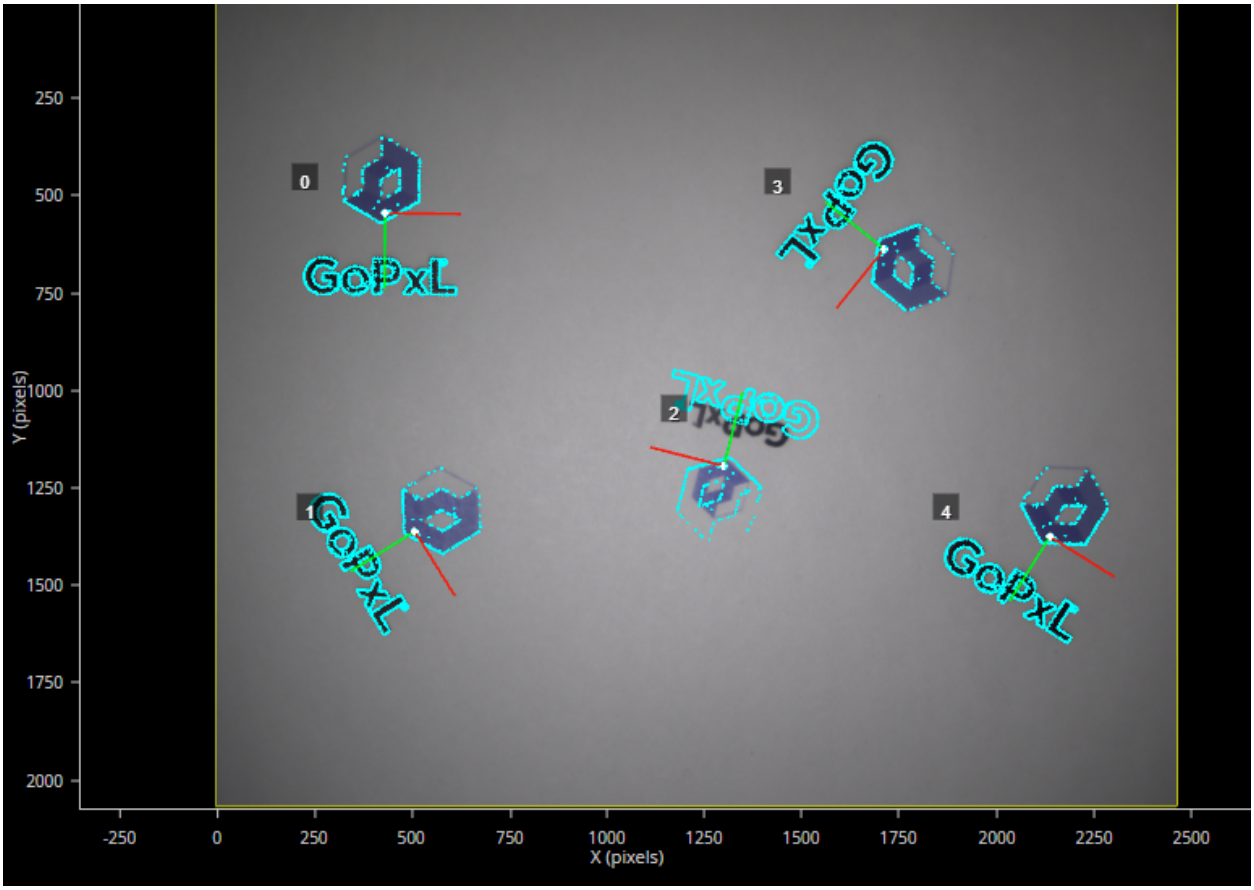
Measurements

Measurement
Text The recognized text.

Image Pattern

The Image Pattern tool locates one or more instances of a feature or part that matches a reference pattern you define. You can define multiple patterns, and you can edit and delete existing patterns. You can also download patterns to back them up or transfer them to another device, and upload them to a device.

In the following, the tool finds five instances of a pattern. Note that the pattern displayed in the data viewer is not scaled up or down to correspond to the size of an instance, such as the middle pattern below.



The tool provides two algorithms:

- "Locator": Ideal for complex patterns, as its detection is more reliable.
- "Find": Faster than the Locator algorithm, but less accurate.

Inputs

	To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

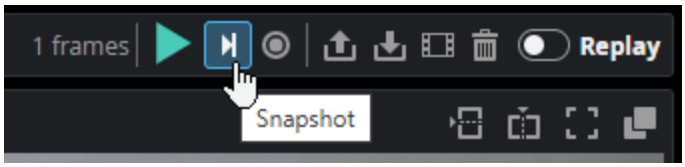
Name	Description
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.

Creating a Pattern

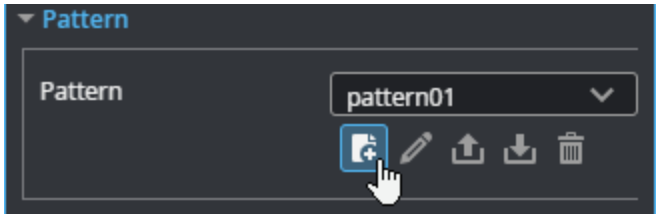
To create a pattern, do the following:

1. Make sure you have calibrated the camera and configured its acquisition settings on the **Acquire** page.

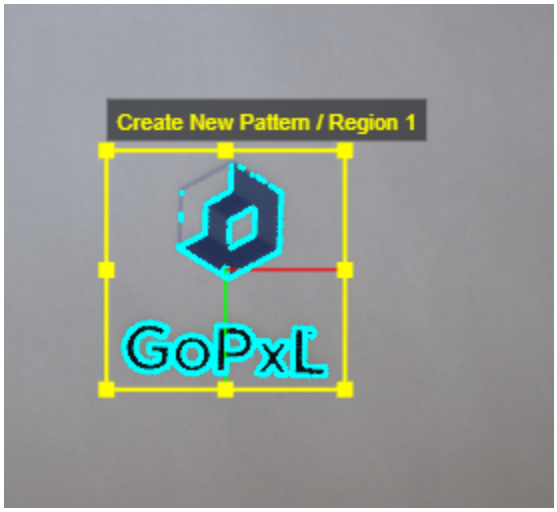
   For more information, see *Acquire - Calibration and Image Acquisition* on page 107.
2. Place a representative, "good " target under the camera.
3. Acquire an image by pressing the snapshot button.
4. Add an Image Pattern tool.

   The first time you add the tool in GoPxL, because there are no patterns on the device, the tool automatically displays the **Create New Pattern** settings. If this is the case, go to step 7.
5. If you have previously created a pattern, expand the **Pattern** section and click the Create button.

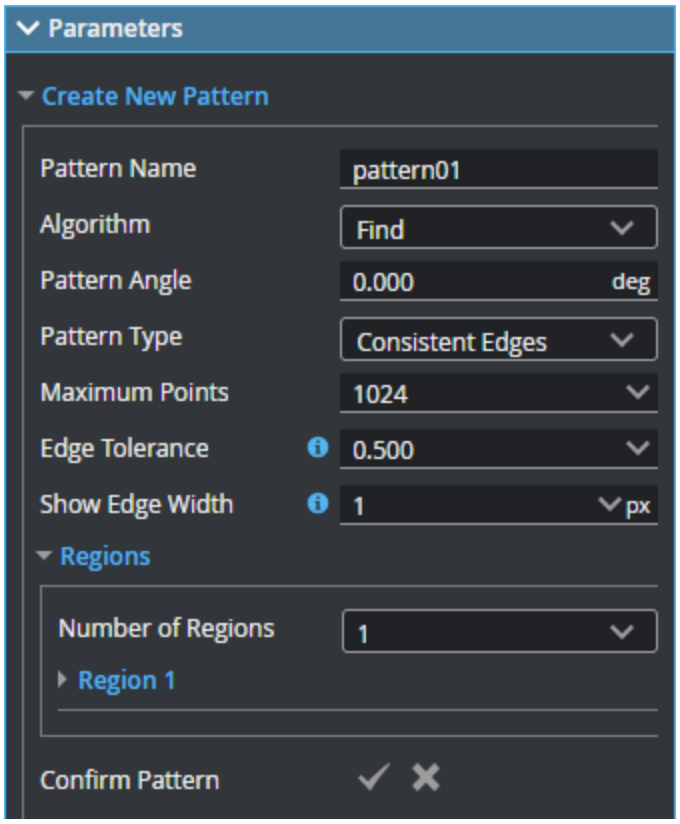
   

6. Move and resize the region of interest (ROI) so that it encompasses the feature or part on your target.

Limit the size of the ROI to improve matching with the pattern later on.

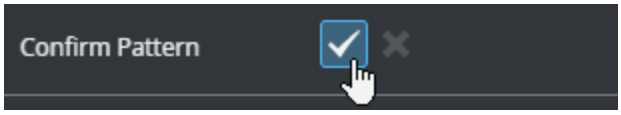
7. In the **Create New Pattern** section, provide a name in **Pattern Name**.



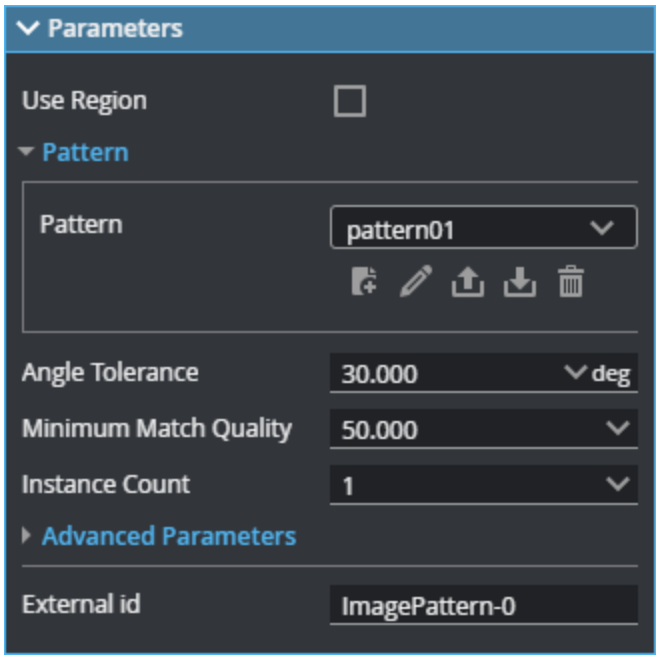
8. Modify the settings for the pattern and your application.

   For information on the settings, see *Pattern Parameters* on page 248.

9. Click the check button next to **Confirm Pattern** at the bottom of the **Create New Pattern** section.



   The pattern is added to the list of patterns available to Image Pattern tools. The tool hides the pattern parameters and displays the tool's matching parameters.

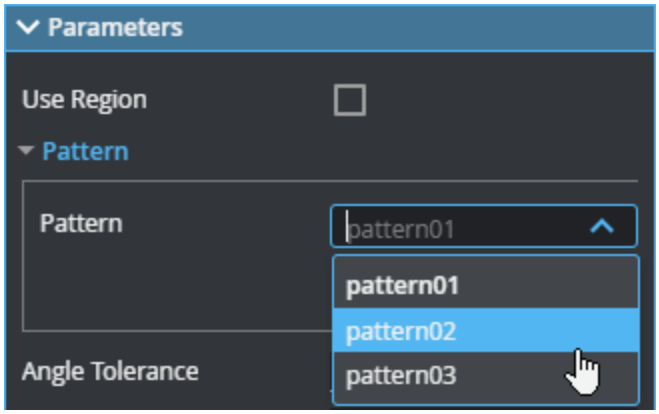


For information on the matching parameters, see *Tool (Matching) Parameters* on page 251.

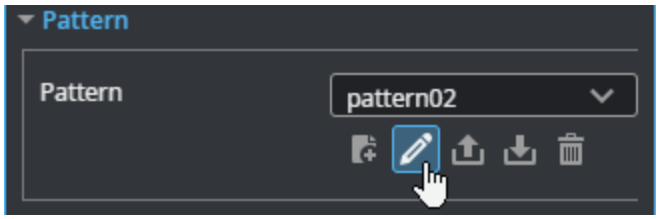
Editing a Pattern

To edit a pattern, do the following:

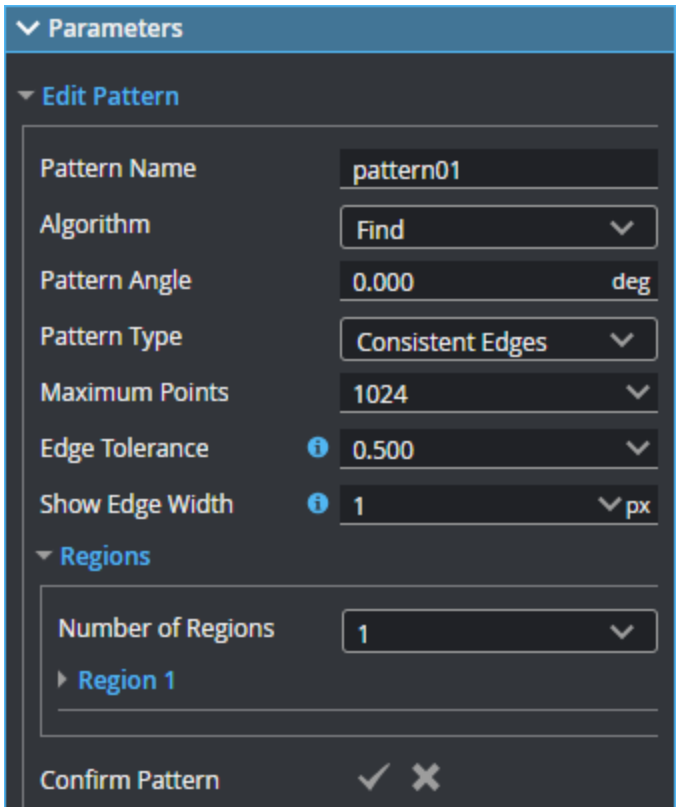
1. In the tool's **Parameters** section, expand the Pattern section, and choose a pattern.



2. Click the Edit button.



The tool displays the pattern parameters.



3. Modify the pattern parameters as needed.

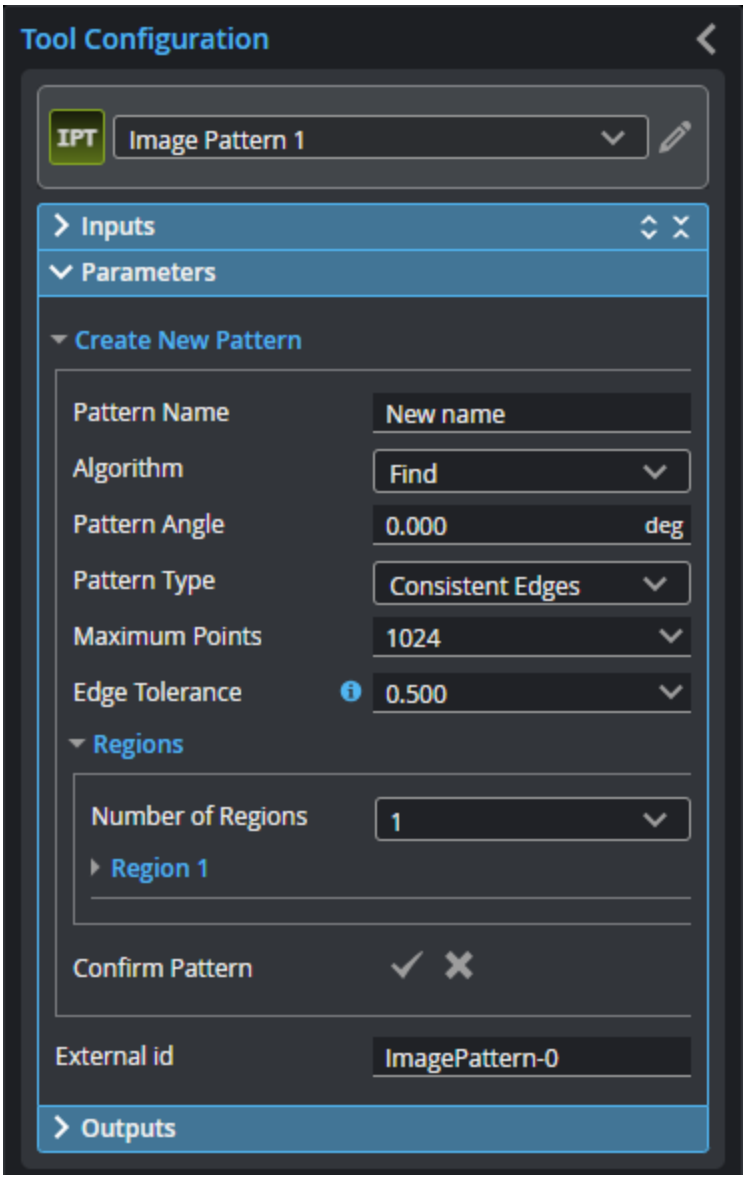
   For information on the pattern parameters, see *Pattern Parameters* below.

Parameters

You create, configure, and manage patterns in the expandable **Parameters** section. Once you have at least one pattern created, you load it to configure the tool's matching parameters.

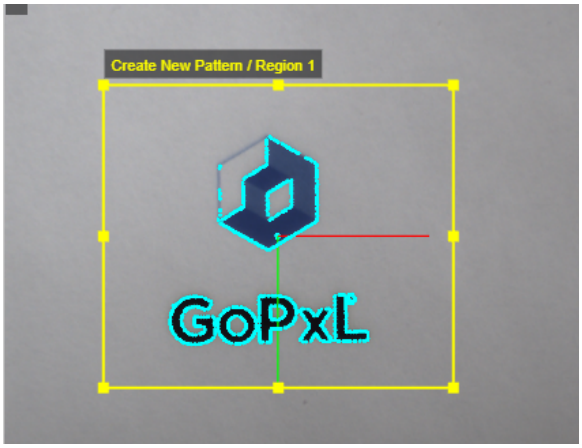
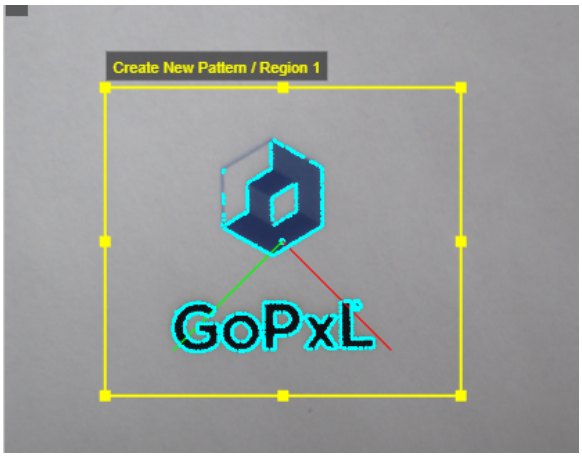
Pattern Parameters

You use the parameters described below to define a pattern.



Pattern Parameters

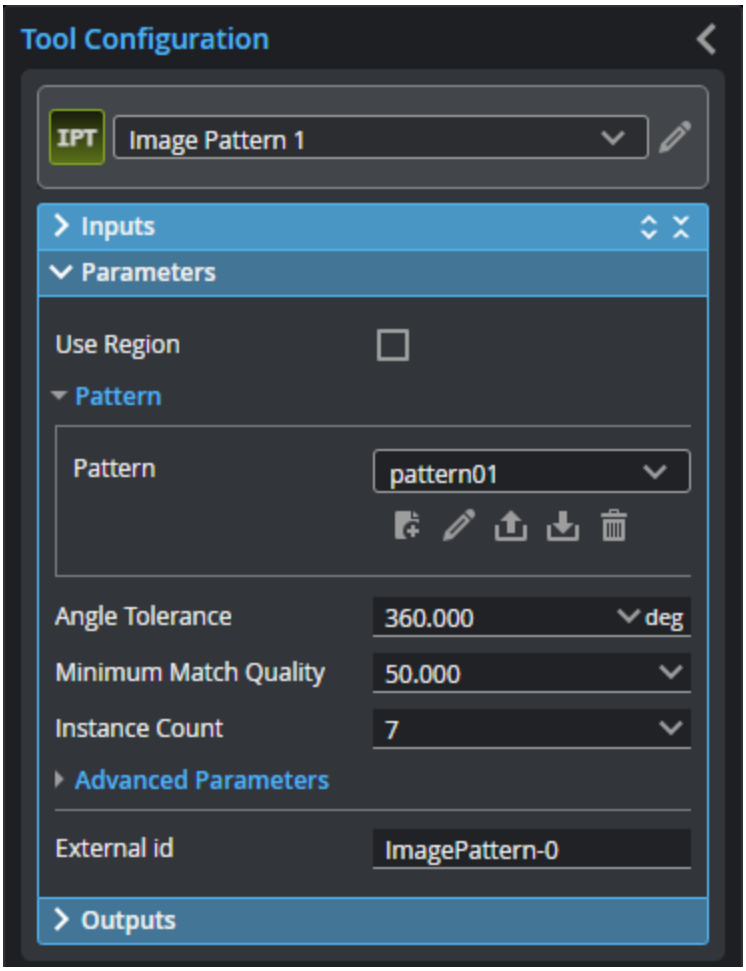
Parameter	Description
Pattern Name	The name of the pattern.
Algorithm	The algorithm the tool uses. One of the following: **Find** The algorithm is based on the Euresys library. **Locator** The algorithm is based on the HexSight library. This algorithm only uses the size and location of the region of interest.

Parameter	Description
Pattern Angle (Find algorithm)	The angle of the model. You typically leave this as 0.0 degrees. At zero degrees:  At 45 degrees: 
Pattern Type (Find algorithm)	The type of the pattern. One of the following: **Consistent Edges** Use this type when your pattern contrasts well with the neighboring pixels, with sharp edges, or when your pattern is very different from the rest of the image in the region or regions of interest. **Thin Structure** Use this type when your pattern is made up primarily of thin elements, when the edges of your pattern are consistent between thin elements and the neighboring pixels, or when the contrast is the same for each thin element. When this mode is

Parameter	Description
	selected, set **Thin Structure Mode** to one of the following: Auto: Automatically chooses the best contrast for the thin elements. Dark: Chooses thin elements that are darker than their neighborhood. Bright: Chooses thin elements that are brighter than their neighborhood.
Maximum Points (Find algorithm)	The maximum number of pixels the tool uses to create the pattern. Setting this to a lower value may reduce processing time when the tool tries to match features in an image to the pattern. Make sure that the tool matches the pattern to features as your application requires.
Edge Tolerance (Find algorithm)	The minimum gradient (steepness) for a pixel to be considered part of an edge. Higher tolerance will only pick the best edges.
Show Edge Width	Displays the detected direction at each edge point. The setting doesn't affect detection accuracy.
Region / Regions	Settings related to the region. The Find algorithm lets you define multiple, flexible regions. The Locator algorithm lets you define a single, rectangular region. For more information, see *Regions* on page 132.

Tool (Matching) Parameters

After you have created a pattern for your application, set the parameters described below for matching the patterns to your targets in production.

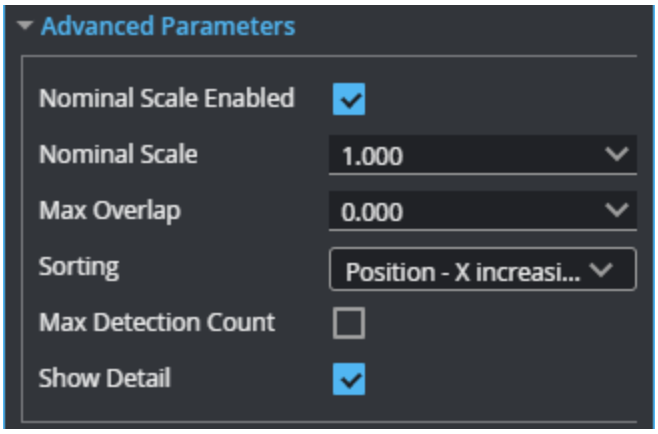


Matching Parameters

Parameter	Description
Use Region	Determines whether the tool limits searches for matching instances a user-defined region.
Region	Settings related to the region. For more information, see *Regions* on page 132.
Pattern	An expandable section that lets you choose a previously saved pattern to use for matching features in images. After you have chosen a pattern, you can set the tool's matching parameters (see below), or , edit, download, or delete a previously saved pattern, create a new pattern, edit
Angle Tolerance	The acceptable maximum angle (positive or negative rotation) of features in an image. Set this to 360 to allow any angle.
Minimum Match Quality	Minimum percentage of the pattern that must match in the image data for the tool to consider the instance as a valid match Instances of the pattern whose Match Quality measurement is less than this value will not be considered a match.

Parameter	Description
Instance Count	The maximum number of instances the tool will return.
Advanced Parameters	An expandable section containing advanced parameters. For more information, see *Advanced Matching Parameters* below.
Instance Count	The maximum number of instances the tool will locate.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

The matching parameters described below are available in the expandable **Advanced Parameters** section.



Advanced Matching Parameters

Parameter	Description
Nominal Scale Enabled	If **Nominal Scale Enabled** is checked, the tool displays a **Nominal Scale** setting. Otherwise, the tool displays **Minimum Scale** and **Maximum Scale** settings and uses the user-defined range.
Nominal Scale	The scale factor the tool requires to recognize an instance. Displayed when **Nominal Scale Enabled** is enabled.
Minimum Scale Maximum Scale	The minimum and maximum scale factors set the acceptable scaling range of an instance. Displayed when **Nominal Scale Enabled** is disabled.
Max Overlap	A value between 0 and 1 representing the maximum amount of overlap between instances, where 0.0 means all instances must be disconnected, and 1.0 means they can fully overlap.
Sorting	Orders measurements and features of the instances. One of the following: • Position - X increasing • Position - X decreasing • Position - Y increasing • Position - Y decreasing
Max Detection Count	Limits the number of initial match candidates to the value you provide. The

Parameter	Description
	number must be equal to or greater than the value of **Instance Count**. A large number can help the tool find difficult or partial matches, but increases processing time. A small number can reduce processing time, but reduces accuracy, especially with difficult or partial matches.
Show Detail	Toggles whether to overlay tool details. **Center only**: The center of the found pattern and its XYZ axes. **Center and All Edges**: The center of the found pattern and its XYZ axes, as well as the outline of the full pattern template. **Center and Matched Edges**: The center of the found pattern and its XYZ axes, as well as the matched edges only.
Instance Count	The maximum number of instances the tool will locate.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

Most tools provide measurements, geometric features, or data as outputs.

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

All outputs provide an external ID (available by expanding the output in the Outputs panel) for optional use in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Measurements

Measurement
Instance Count Returns the total number of instances (matches).
X **Y** The X and Y positions of the center of the instance.
Angle The angle of the instance in relation to the loaded pattern.
Scale The scale of the instance in relation to the loaded pattern.
Match Quality The match quality of the instance.

Features

Type	Description
Point	The point representing the center of an instance.
Line	The line representing the rotated X axis, with respect to the loaded pattern.

For more information on geometric features, see *Geometric Features* on page 137.

Image Pixel Count

The Image Pixel Count tool returns a count of the number of pixels in an image.

> When **Enable Array Output** is checked, the tool outputs the blobs in an array called Blob Surface. Be *sure* to enable Blob Surface in the **Outputs** section after enabling this parameter.

For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Inputs

> To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array, other than the processing limitations of the sensor. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Parameters

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

Most tools provide measurements, geometric features, or data as outputs.

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

All outputs provide an external ID (available by expanding the output in the Outputs panel) for optional use in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Measurements

Measurement
Count The number of pixels.

Data

Type	**Description**
Diagnostic Image	Image data you can use to evaluate the impact of the tool's parameters.

Image QR Code

The QR Code tool lets you decode QR code data encoded in Image data. The tool supports grayscale and color images.

The tool returns the decoded string, as well as other measures such as its dimensions and the ECC family to which it belongs. The tool also provides measurements that can help you assess the print quality of the code symbols. Indicators from the following standards are available:

- ISO 15415
- ISO 29158

Inputs

> To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

Inputs

Name	Description
Image Input	The data the tool applies measurements to or processes.
Anchor X	The X or Y measurement of another tool that this tool uses as a positional anchor.

Name	Description
Anchor Y	Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Parameter	Description
Use Region Region	Enable **Use Region** and set the values for the regions.
Instance Count	The maximum number of matrix codes to decode in a frame.
Grading Standard	The standard the tool will use to assess the quality of the matrix code. Grading measurements are added in the tool's **Outputs** section. One of the following: • ISO 15415 • ISO 29158
Compute Grading	The type of grading the tool returns in the grading measurements. One of the following: **Number** - Grade measurements return a numerical value between 0 and 4. **Letter** - Grade measurements return A, B, C, D, or F. **Letter and Number** - Both types of outputs are returned. Numeric measurements are distinguished from letter-based ones by "Number" in the name.
Read Mode	Whether the tool prioritizes speed or detection quality. **Speed** The tool stops searching for matrix codes as soon as one of the following happens: • It finds the number of codes selected in **Instance Count**. • It has finished processing the image. This mode results in shorter processing times. **Quality** The tool tries to improve its detection until it has finished processing the image. This mode results in better grading results.
Show Unreliable Codes	Shows unreliable codes (objects that are likely data matrix codes, but can't be decoded).

Parameter	Description
Advanced Parameters	Expandable section containing advanced settings. For more information, see *Parameters - Advanced* below.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Parameters - Advanced

Parameter	Description
Detection Mode	The detection method the tool uses to find QR codes. **Adaptive Threshold** - Detects finder patterns based on adaptive thresholding. **Gradient** - Detects finder patterns based on gradients.
Search Model Params	The QR code models the tool searches for. **QR Model 1** - Searches for QR Model 1 **QR Model 2** - Searches for QR Model 2 or 2005. **Micro QR** - Searches for Micro QR.
Search Parameters	**Min Version** / **Max Version** - The minimum and maximum of QR codes the tool searches for. **Acceptable Deformation** - A percentage representing the value of the short side of the code divided by its long side. **Min Score** Minimum pattern finder score needed to consider a finder pattern found. **Correction Aggressiveness** The minimum threshold for cell polarity confidence. When the confidence is below the threshold, the tool further processes the code to improve polarity detection. The cell polarity confidence is the confidence in the cell digitization result, on a scale from 0 to 1. Increasing the threshold can improve reading of overprinted or underprinted QR codes. Large values can affect the speed and increase false positive corrections. **Scan Precision** One of the following: • Automatic: The tool determines scan precision automatically. • Fine: The tool performs a fine scan to find the QR codes. Best for small QR codes.

Parameter	Description
	• Coarse: The tool performs a coarse scan to find the QR codes. Best for medium to large QR codes.
	Minimum Contrast Threshold The minimum difference in grayscale value between a pixel and its local background for the pixel to be part of the foreground.

Outputs

Most tools provide measurements, geometric features, or data as outputs.

Outputs section with a measurement expanded to show user-configurable decision min/max fields and an external ID

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

Some measurements are only displayed when you select a grading standard in **Compute Grading** (see *Parameters* on page 259). "Modules" refers to the cells or "dots" in the grid of the matrix code.

Measurements (all grading standards)

Measurement
Total Count The number of matrix codes decoded.
Decoded String The decoded string, packed as an array of float-point values.

When **Compute Grading** is set to "Letter and Number", the tool outputs both "Number Grade" measurements containing a value between 4 and 0 and "Grade" outputs that contain A, B, C, D, or F. If the setting is either "Letter" or "Number", the tool only returns "Grade" outputs, which contain the selected type.

Measurements (ISO 15415)

Measurement
Decode Grade **Decoding Number Grade** Whether the code was decoded.
Axial Non Uniformity Grade **Axial Non Uniformity Number Grade** Grade for uneven scaling along the X and Y axis.
Grid Non Uniformity Grade

Measurement
Grid Non Uniformity Number Grade How much the measured position of the grid intersection point deviates from the expected position. Codes sheered along the X and/or Y axis can result in a low grade. The grade can be unstable and its usefulness is limited.
Symbol Contrast Grade **Symbol Contrast Number Grade** The difference between the dark and light areas of a code, including the quiet zones.
Modulation Grade **Modulation Number Grade** Uniformity of the reflectance of dark and light areas throughout the code. Irregular or mottled regions can result in a low grade.
Reflectance Margin Grade **Reflectance Margin Number Grade** How distinguishable the modules are relative to the global threshold (the mean reflectance of the brightest and the darkest module).
Fixed Pattern Damage Grade **Fixed Pattern Damage Number Grade** A measure of any damage (missing data or data where it is not expected) to the finder pattern (the L-shaped along the left and bottom of the code), the quiet zone (margins outside the matrix), or the clock track (the alternating "dots" along the top). The value returned is an average of the grades of these areas of concern. A higher value indicates the matrix code is more correctly located and identified.
Unused Error Correction Grade **Unused Error Correction Number Grade** Grade representing how little error correction to reconstruct data in damaged or poorly printed codes was used. A higher grade means less error correction was applied.
Horizontal Print Growth **Vertical Print Growth** **Horizontal Print Growth Number** **Vertical Print Growth Number** How much the matrix code may have grown or shrunk compared to the expected size horizontally and vertically, respecitively.

Measurements (ISO 29158)

Measurement
Cell Contrast Grade **Cell Contrast Number Grade** Relative difference between the mean reflectance of the darkest and brightest cells in the code.
Cell Modulation Grade

Measurement
Cell Modulation Number Grade The variability and reliability of the black and white colors. A low grade means the tool will have difficulty distinguishing between black and white cells.
Minimum Reflectance Grade **Minimum Reflectance Number Grade** The ratio between the smallest and largest gray values in the barcode. For the "Number" grade, the value is 4 if the smallest gray value of the barcode is <= half of its highest value, and 0 otherwise.
Fixed Pattern Damage Grade **Fixed Pattern Damage Number Grade** How likely the symbol is correctly located and identified in the code.
Mean Light In Required Bounds **Mean Light In Required Number Bounds** Whether mean light is 70%~86% of the maximum gray level.
Mean Light **Mean Number Light** The mean value of the pixels in the center of the light cells.

Image Track

The Image Track tool lets you perform quality control and inspection of gaps, glue bead, pipes, and rectangular features in images. The tool supports multiple regions.

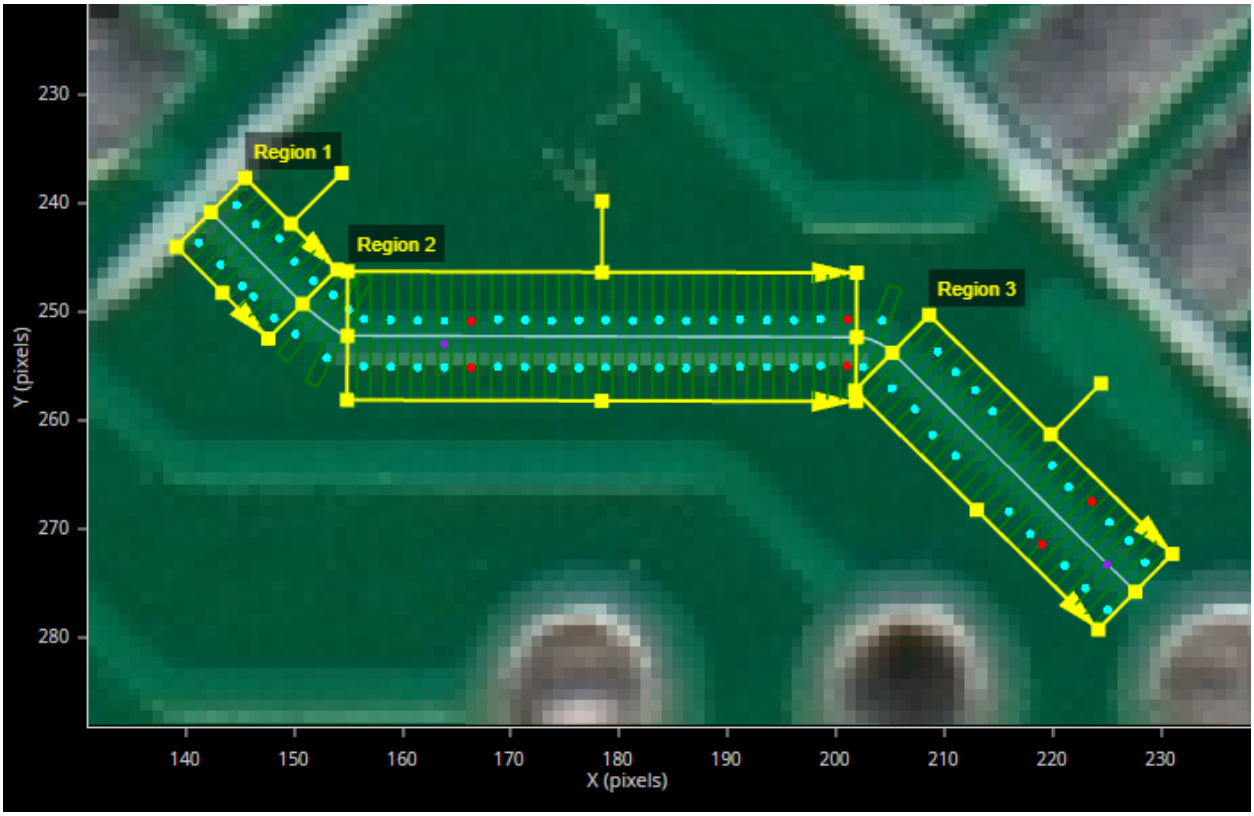
As you add regions, the tool displays a new region "connected" to the previous region. To define the expected track, you resize, position, and rotate the regions you add. The tool can "fill in" curves in spaces between regions, or ignore the spaces.

The tool tries to find track edges (rising or falling intensity transitions greater than the provided threshold) by searching in the defined sample areas (perpendicular to the center line) from the outer edge of the regions toward the center line.

The tool displays the following dots:

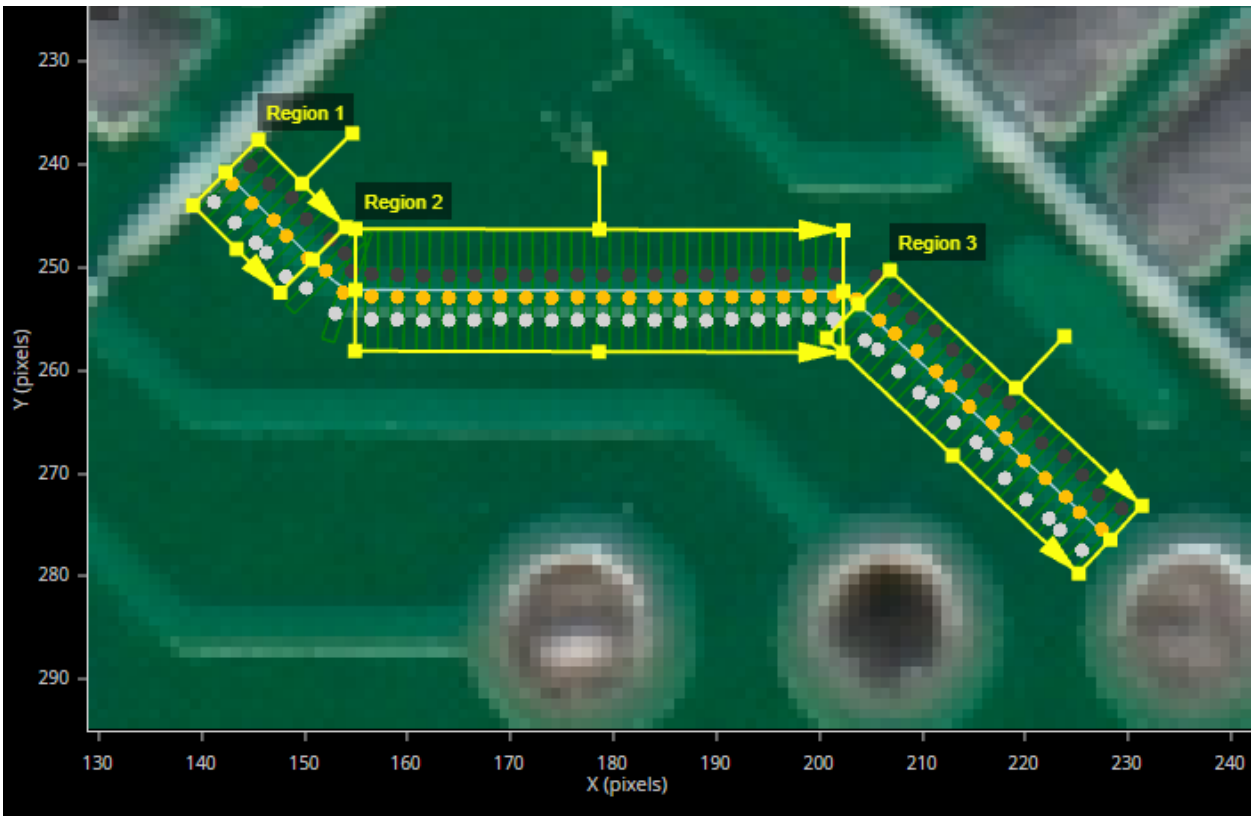
- Cyan dots represent valid edge points.
- Red dots represent invalid edge points (those excluded by the **Advanced Parameters** > **Distance Outlier Fraction** parameter).
- Violet dots represent either the Min Distance or Max Distance.

In the following, three connected regions follow a PCB trace. The arrows indicate the "direction" of the track and serve to show where new regions will be added.



The tool displays the Point geometric features related to the center points and the points at the edge of the track as larger dots. The colors are selected automatically by the tool, but you can change

these in the Displayed Outputs panel; for more information, see *Displaying and Pinning Outputs* on page 87. To display these points, enable them in the tool's **Outputs** section and select or pin the output.



Inputs

> To use a measurement as an anchor, it must be enabled and properly configured in the tool providing the anchor. For more information on anchoring, see *Measurement Anchoring* on page 138.

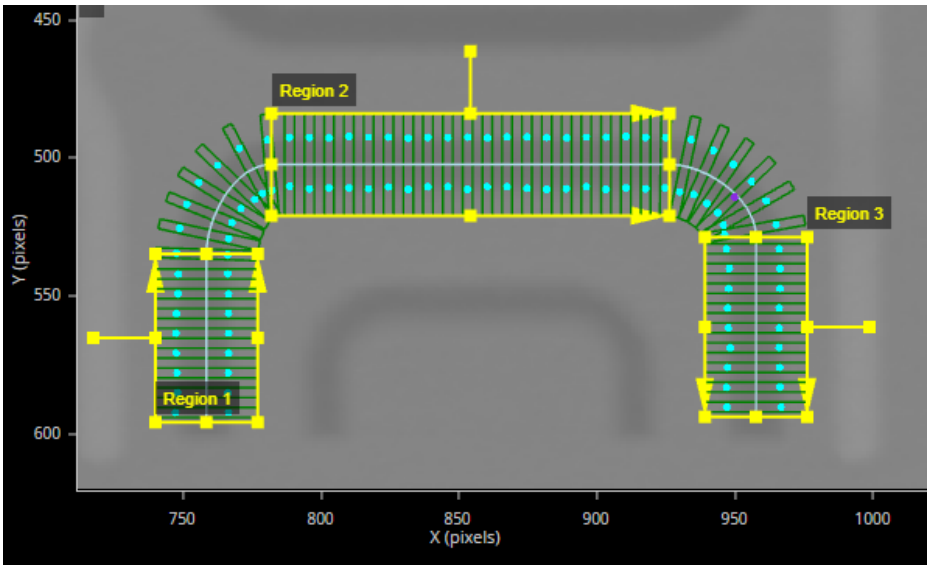
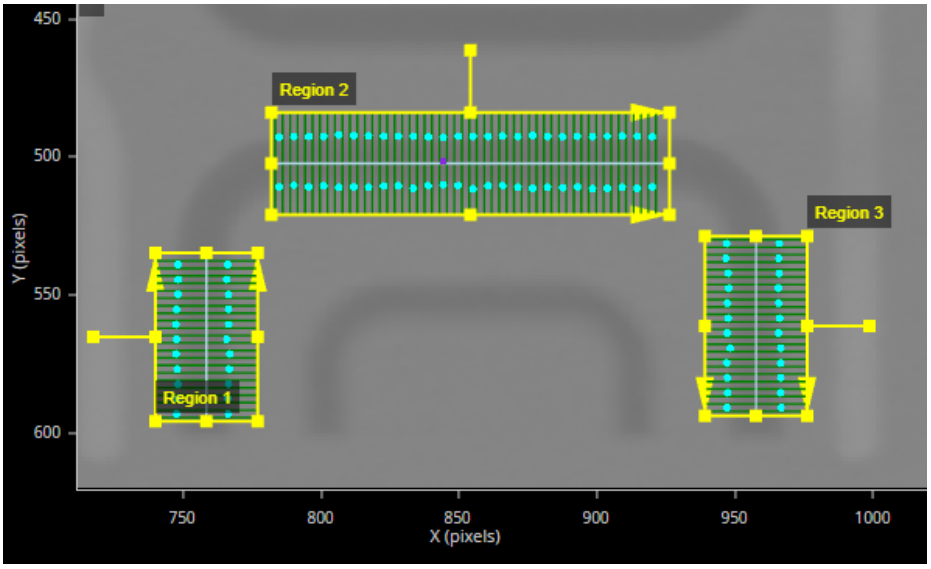
Inputs

Name	Description
Image Input	The data the tool applies measurements to or processes.
Anchor X Anchor Y	The X or Y measurement of another tool that this tool uses as a positional anchor. Positional anchors are optional.
Anchor Angle	The angle measurement of another tool to use as a rotational anchor for this tool. Rotational anchors are optional.

Parameters

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Region {n}	When expanded, displays the region-related settings. In addition to size, location, and rotation, you can define whether the tool adds sample areas between a region and the previous region by setting **Previous Path** to Spline. In the following, splines (with **Spline Roundness** set to 1) are enabled for Region 2 and Region 3.  When a spline is not enabled, sampling is limited to the regions. 
Add Region	Adds a region.
Threshold	The intensity threshold above or below which a pixel will be considered the edge of a track. On color devices, the RGB values are converted to an 8-bit monochrome

Parameter	Description
	image.
Step Direction	Whether the tool looks for the rising (increasing) or falling (decreasing) intensity values in the sample areas.
Selection Type	One of the following: Best or First. Determines whether the best transition (greatest intensity difference) or first transition exceeding the value of **Threshold** is used as an edge point.
Sample Count	The number of areas in which the tool searches for an edge. The areas are spaced equally across all regions, as well as areas between regions where splines are enabled.
Show Detail	Displays or hides area lines and center lines in the regions.
Advanced Parameters	When expanded, displays advanced parameters. **Distance Outlier Fraction** The percentage of outlier points to exclude. Setting this to a small value can help the tool fit the line better. **Step Smoothing** The window in which the tool performs smoothing. The size of the window along which the tool calculates an average for determining steps. The setting is useful for averaging out noise.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

All outputs provide an external ID (available by expanding the output in the Outputs panel) for optional use in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Measurements

Measurement
Min Distance **Max Distance** **Average Distance** The minimum, maximum, and average distances of edge points from the center line.
Distances An array of distances from the center line.

Data

Type	Description
Center Points **Left Points**	Arrays of the Point geometric features: center points, as well as points at the edge of the track. The Left and Right array names are arbitrary. In the following, the

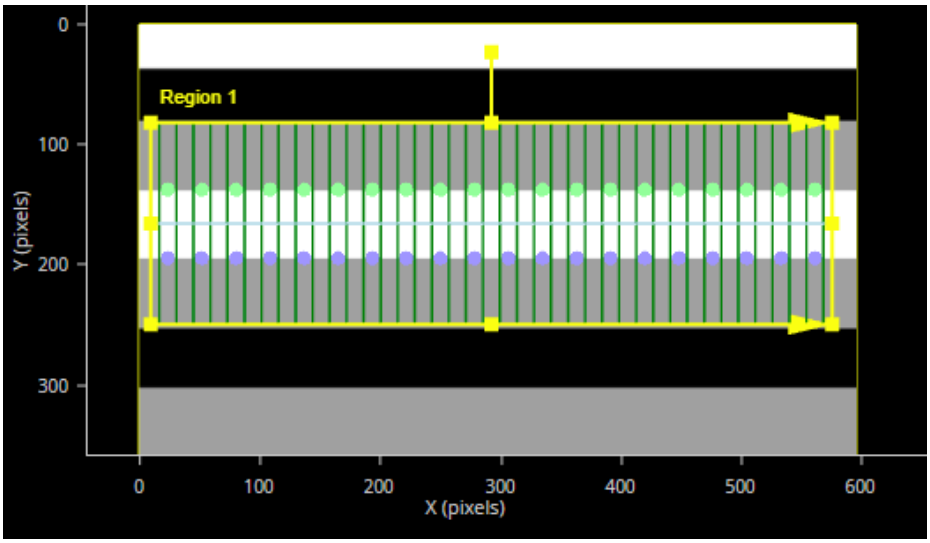
Type	Description
Right Points	green points at the top of the region are "Left" points. The violet ones are "Right" points. 

Image Transform

The Image Transform tool generates a new image based on the coordinate system of geometric features the tool uses as input. The tool can take a line or origin point to define this new coordinate system. You can then apply the built-in measurement tools or GDK tools to this new data. The tool can help improve repeatability of your measurements.

Inputs

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array, other than the processing limitations of the sensor. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Image Input	The data the tool applies measurements to or processes.
Line Point	The Line and Point geometric features the tool uses to perform transformations.

For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Parameters

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Region Type	The type of the region. One of the following: Full Image or Point as Reference. **Full Image** The entire image is transformed using the Line and Point geometric feature inputs, if they are set, and a fixed transformation if **Add Fixed Transform** is enabled and set. **Point as Reference** Crops the image using the size and position parameters under **Point as Reference Region**. If the Line and Point geometric feature inputs are set, the tool uses them to transform the image.
Region	The region to which the tool's measurements will apply. For more information, see *Regions* on page 132.
Add Fixed Transform	When enabled, displays positional offset and angle fields you can use to manually set transformations. If you also set geometric features as inputs, which are applied after any transformations supplied by the input geometric features. Setting a fixed transformation can be useful if the geometric features the tool uses results in data rotated to an unusual orientation; you could, for example, rotate the data 90 or 180 degrees so that it is in the "expected" orientation, or shift it so that it's easier to work with.

Outputs

Data

Type	Description
Transformed Image	The transformed image.

Measurement Formula Tool

The Measurement Formula tool lets you create a mathematical expression containing aliases to one or more measurements from other measurement tools, up to a maximum of 20. The result of the calculation is returned in the tool's Result measurement.

The tool's default timeout is 5 seconds: if the tool fails to complete the calculation after 5 seconds, the result is set to Invalid.

The tool uses the C++ Mathematical Expression Toolkit Library (ExprTk). For full details on what it supports, see the toolkit's page (http://www.partow.net/programming/exprtk/).

For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Inputs

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Input	Set this parameter to any available input. The selected input has no effect on the tool's calculations.
Variable {n}	A measurement from another tool that you use in the formula. The **Measurement Count** parameter controls the number of variable inputs.

Parameters

The following parameters are in the expandable **Parameters** section in the tool's configuration.

Parameters

Parameter	Description
Measurement Count	Sets the number of variable inputs in the **Inputs** section. For each, a **Variable {n}** input and a **Variable {n} Alias** parameter is added, and for use in the expression.
Formula	The mathematical expression. Use the variable aliases from the **Variable {n} Alias** parameters in the expression. For example: • M1*3+M2/2+4 • tan(M1/180*pi) • max(M1, M2, M3) The tool supports complex mathematical formulas. For example, the following:

Parameter	Description

	$\sqrt{\frac{1-a\times e^{2t}+w^{\pi}}{sin(\frac{2x}{\pi})+cos(\frac{\pi}{y})}}$
	can be produced using the following formula:
	`sqrt((1-a*exp(2t)+w^(pi))/(sin(2x/pi)+cos(pi/y)))`
	The following shows how to perform a *while* loop:
	`y:=0; while(x<10){y:=y+x;x+=1;};y`
	Note that ":=" is the assignment operator, and "=" is the equality operator.
	The following shows how to use an if-else structure:
	`if(x>2){x:=x+2;}else{x:=x-2;};x`

Variable {n} Alias	The alias of Variable {n} (that is, a measurement set in the **Inputs** section). Use these aliases in **Formula**.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

Outputs section with a measurement expanded to show user-configurable decision min/max fields and an external ID

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

Measurements

Measurement
Result The result of the calculation in **Formula**.

Script Tool

The Script tool executes a user-defined Python-based script.

For a list of the tool's supported classes and methods, see *Functions and Classes* on page 274.

For examples, see *Examples* on page 287.

The tool supports all GoPxL data types as input and output:

- Surface, Profile, Image, and Mesh data; intensity data is available with Surface and Profile data
- Measurements (values and decisions, and anchor sources for other tools)
- Strings
- Geometric features
- Arrays of the supported types

> The Script tool does *not* support writing files when it is running on a sensor or GoMax NX device, and will display a permissions error if you attempt this. Writing to files is only supported when the tool is running on a PC instance of GoPxL. When writing to files, you should explicitly set the path to a writable location.

The tool does not support using `print()`, `sys.stdout`, or Python's `logging` module functions to send messages to GoPxL's system log. Instead, use one of the "log_" functions. For more information, see *Logging* on page 280.

Script tools are added, configured, and removed like other measurement tools.

> The Script tool uses Python version 3.8.

For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Included Python Modules

In addition to the standard Python library modules, the tool also includes the NumPy 1.19 module. NumPy is useful for operating on arrays.

On Windows, you can install additional modules with `pip` by running the following command under the `bin\win64\python` folder in the main GoPxL folder:

```
python3.exe -m pip install <modulename>
```

Note that for some libraries you may need to specify a specific version to ensure compatibility with the version of Python included with GoPxL.

Inputs

You configure the tool's inputs in the expandable **Inputs** section.

Inputs

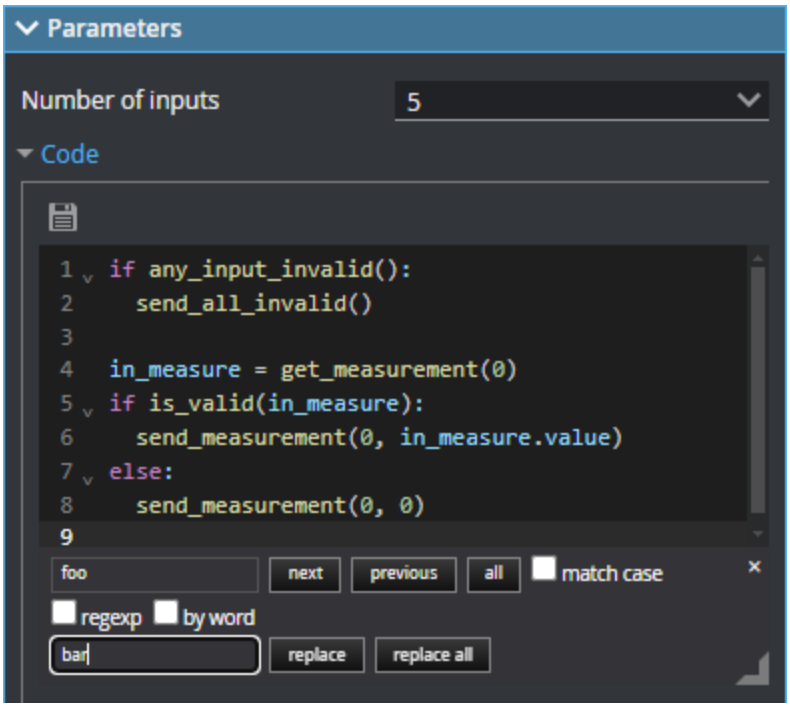
Name	Description
Input {n}	Selects the input for the script where {n} is the index of the input. All input types

Name	Description
	are allowed. Input cannot be 'None'. In the `get_` functions' `index` parameters, use this index. The **Number of inputs** parameter (see below) determines the number of inputs. This tool can optionally take an array as input. For more information, see *Arrays, Batching, and Aggregation* on page 126.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Name	Description
Number of Inputs	Specifies the number of inputs needed in the script.
Code	The Python code the tool runs. Supports expanding and collapsing of code blocks. You can resize the editor vertically with the grabber to the lower left of the editor. You can resize the width of the editor by resizing the **Tool Configuration** panel. The following keyboard shortcuts are available in the editor. **Ctrl-S**: Saves the script. The tool checks the syntax. Use Cmd-S on macOS. **Ctrl-F**: Opens a search and replace panel. The search and replace panel supports regular expressions. Use Cmd-F on macOS. To close the panel, click the "x" to the upper right.  **Ctrl-D**: Highlights all occurrences of the word under the cursor. Each time you press Ctrl-D after that, the cursor moved to the next occurrence. Use Cmd-D on macOS.

Name	Description
	Alt-ArrowLeft / Alt-ArrowRight: Move the cursor over the next element to the left or right, respectively. **Alt-ArrowUp / Alt-ArrowDown**: Move the selected lines up or down one line, respectively. Combining this with Shift copies lines instead. **Ctrl-/**: Toggles commenting. Use Cmd-/ on macOS. **Ctrl-[/ Ctrl-]**: Decreases or increases the indentation of the selected line. Use Cmd-[or Cmd-] on macOS.
Number of outputs	Specifies the number of outputs the script produces.
Output {n} type	Selects the output type for each output, where {n} is the index of the output.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

Most tools provide measurements, geometric features, or data as outputs.

Outputs section with a measurement expanded to show user-configurable decision min/max fields and an external ID

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

Measurements

Measurement
Output {n} Output from the Python script, where {n} is the index of the output. The number of outputs and the type of each output is determined by the **Number of outputs** and **Output type {n}** parameters, respectively.

Functions and Classes

Getting Input

The following Python functions are available for performing input operations and testing input validity.

The index of the input is given as an argument. If the input at the index is missing or it is the wrong type, None is returned. Functions will return an array if input is an array.

For information on the returned types, see *Data Structures* on page 280.

Input Functions

Function	Description
is_valid(obj)	Checks whether the object is valid or not.

Function	Description
	Parameters **obj**: Input message. *Returns* True if valid.
get_surface(index)	Gets a surface message or array of messages from the specified input. *Parameters* **index**: Input index. *Returns* SurfaceMsg, array of SurfaceMsg, or None.
get_profile(index)	Gets a profile message or array of messages from the specified input. *Parameters* **index**: Input index. *Returns* ProfileMsg, array of ProfileMsg, or None.
get_measurement(index)	Gets a measurement message or array of messages from the specified input. *Parameters* **index**: Input index. *Returns* MeasureMsg, array of MeasureMsg, or None.
get_point(index)	Gets a point message or array of messages from the specified input. *Parameters* **index**: Input index. *Returns* PointFeatureMsg, array of PointFeatureMsg, or None.
get_circle(index)	Gets a circle message or array of messages from the specified input. *Parameters* **index**: Input index. *Returns* CircleFeatureMsg, array of CircleFeatureMsg, or None.
get_plane(index)	Gets a plane message or array of messages from the specified input. *Parameters* **index**: Input index. *Returns* PlaneFeatureMsg, array of PlaneFeatureMsg, or None.
get_line(index)	Gets a line message or array of messages from the specified input. *Parameters* **index**: Input index. *Returns* LineFeatureMsg, array of LineFeatureMsg, or None.
get_image(index)	Gets an image message or array of messages from the specified input. *Parameters* **index**: Input index.

Function	Description
	Returns ImageMsg, array of ImageMsg, or None.
get_string(index)	Gets a string or array of strings from the specified input. *Parameters* **index**: Input index. *Returns* StrMsg, array of StrMsg, or None.
get_mesh(index)	Gets a mesh message or array of messages from the specified input. *Parameters* **index**: Input index. *Returns* MeshMsg, array of MeshMsgs or None.

Drawing Graphics

The Python functions listed below are available for drawing graphics in the data viewer. The default values used in draw() functions are as follows:

GRAPHIC_DEFAULT_SIZE = 5

GRAPHIC_DEFAULT_COLOR = Colors.CYAN

Draw Functions

Function	Description
draw_points(points, color = GRAPHIC_DEFAULT_COLOR, size = GRAPHIC_DEFAULT_SIZE, shape = MarkerShape.CIRCLE, output = None)	Draw points with customizable size, shape and color. *Parameters* **points**: A list of points. Each item can be a Point, Point3d, or a tuple containing 2 or 3 numeric values (e.g., (x, y) or (x, y, z)). **color**: RGB color to apply to the points. **size**: Size of the points to be drawn. **shape**: Shape of the points to be drawn. **output**: Output index where the graphics are associated. Use None for tool graphics.
draw_lines(points, color = GRAPHIC_DEFAULT_COLOR, width = GRAPHIC_DEFAULT_SIZE, hasStartPointArrow = False, hasEndPointArrow = False, output = None)	Draw graphic lines with customizable width and color. *Parameters* **points**: A list of points. Each item can be a Point, Point3d, or a tuple containing 2 or 3 numeric values (e.g., (x, y) or (x, y, z)). **color**: RGB color to apply to the points. **size**: Width of the line to be drawn. **hasStartPointArrow**: Whether the start point arrow should be drawn. **hasEndPointArrow**: Whether the end point arrow should be drawn. **output**: Output index where the graphics are associated. Use None for tool graphics.
draw_region(points, center, dimensions, angle,	Draw region with customizable dimensions. *Parameters*

Function	Description
output = None)	**center**: Region center. **dimensions**: Width, height and depth of the region. **angle**: Rotation angle (about Z Axis) of the region. **output**: Output index where the graphics are associated. Use None for tool graphics.
draw_label(text, position, output = None)	Draw label with customizable location. *Parameters* **text**: The label string. **position**: The position of the label. **output**: Output index where the graphics are associated. Use None for tool graphics.
draw_plane(normal, distance, output = None)	Draw plane with customizable normal and distance. *Parameters* **normal**: The normal of the plane. **distance**: The to the origin of the plane. **output**: Output index where the graphics are associated. Use None for tool graphics.
draw_ray(position, direction, width = GRAPHIC_DEFAULT_SIZE, color = GRAPHIC_ DEFAULT_COLOR,, output = None)	Draw points with customizable size, shape and color. *Parameters* **position**: The position of the ray. **direction**: The direction of the ray. **size**: Width of the line to be drawn. **color**: RGB color to apply to the points. **output**: Output index where the graphics are associated. Use None for tool graphics.

Sending Output

The following Python functions are available for performing output operations.

The index of the output is given as an argument.

For information on the output types, see *Data Structures* on page 280.

Output Functions

Function	Description
is_output_enabled(index)	Checks whether the output at the specified index is enabled. *Parameters* **index**: Output index. *Returns* True if the output is enabled.
send_surface(index, offset, scale, points, intensity=None, is_adjacent=None,	Sends a surface message or array of messages to the specified output. *Parameters* **index**: Output index. **offset**: Offset of the data or array of offsets.

Function	Description
header=None)	**scale**: Scale of the data or array of scales. **points**: Surface points data or array points data. Points for uniform data are in an array of size (height, width). Points for point cloud (non-uniform) data are in an array of size (height, width, 3). **intensity**: Surface intensity data of size (height, width), or array of intensity data. **is_adjacent**: Set to true if the surface data is adjacent. Can also be an array of adjacent info. **header**: Header part of the data message. Uses header from the first input if not set.
send_profile(index, offset, scale, points, intensity=None, slices=None, header=None)	Sends a profile message or array of messages to the specified output. *Parameters* **index**: Output index. **offset**: Offset of the data or array of offsets. **scale**: Scale of the data or array of scales. **points**: Profile point data or array of profile points. Data is in array of size (width) or (width,2). **intensity**: Profile intensity data of size (width), or array of profile intensities. **slices**: Not used. **header**: Header part of the data message. Uses header from the first input if not set.
send_measurement(index, value, label_ position=None, header=None)	Sends a measurement message or array of messages to the specified output. Note that the decision can't be set with this function because the decision of a measurement is only set after output using the Min and Max thresholds on the outputs. *Parameters* **index**: Output index. **value**: Value of the data message, or array of values. Numpy.nan if measurement is invalid. **label_position**: Position of point in measurement message, or array of positions. **header**: Header part of the data message. Uses header from the first input if not set.
send_point(index, position, header=None)	Sends a point message or array of messages to the specified output. *Parameters* **index**: Output index. **position**: Position of feature point in data message, or array of positions. **header**: Header part of the data message. Uses header from the first input if not set.
send_line(index, position, direction, header=None)	Sends a line message or array of messages to the specified output. *Parameters* **index**: Output index. **position**: Point vector in the line, or array of point vectors. **direction**: Direction vector of the line, or array of direction vectors.

Function	Description
	header: Header part of the data message. Uses header from the first input if not set.
send_circle(index, position, normal, radius, header=None)	Sends a circle message or array of messages to the specified output. *Parameters* **index**: Output index. **position**: Center position of the circle, or array of center positions. **normal**: Normal vector of the circular plane, or array of normal vectors. **radius**: Radius of the circle, or array of radii. **header**: Header part of the data message. Uses header from the first input if not set.
send_plane(index, normal, origin_distance, header=None)	Sends a plane message or array of messages to the specified output. *Parameters* **index**: Output index. **normal**: Normal vector of the plane, or array of normal vectors. **origin_distance**: Shortest distance from origin to plane, or array of distances. **header**: Header part of the data message. Uses header from the first input if not set.
send_image(index, pixels, pixel_format= PixelFormat.MONO_8, flipped_x=False, flipped_ y=False, transposed=False, header=None)	Sends an image message or array of messages to the specified output. *Parameters* **index**: Output index. **pixels**: Pixel data of image of size (height, width, pixel size), or array of image pixels. **pixel_format**: Pixel format of the image, or array of pixel formats. For more information, see *PixelFormat* on page 283. **flipped_x**: Set to `true` if the image data is flipped on the X axis, or array of flags. **flipped_y**: Set to `true` if the image data is flipped on the Y axis, or array of flags. **transposed**: Set to `true` if the image data is transposed, or array of flags. **header**: Header part of the data message. Uses header from the first input if not set.
send_string(index, string, position = None, decision = None, header=None)	Sends a string or array of strings to the specified output. *Parameters* **index**: Output index. **string**: The text to send. **position**: Position of point in measurement message. Type: Point3d(). **decision**: decision of measurement message. Type: MeasurementDecision() enum class. **header**: Header part of the data message. Uses header from the first input if not set.
send_mesh(index, vertex_ data, facet_data, vertex_	Sends a mesh message or array of messages to the specified output. *Parameters*

Function	Description
normal_data = None, facet_normal_data = None, vertex_texture_data = None, data_offset = None, data_range = None, header=None)	**index**: Output index. **vertex_data**: Vertex data points. Type: numpy array of floats (x,y,z). **facet_data**: Facet data points. Type: numpy array of uint32 triangle indexes (vertex 1, vertex 2,vertex 3). **vertex_normal_data**: Vertex normal data points. Type: numpy array of floats (x,y,z). **facet_normal_data**: Facet normal data points. Type: numpy array of floats (x,y,z). **vertex_texture_data**: Vertex texture data. Type: numpy array of uint8. **data_offset**: Offset of the data or array of offsets. Type: Type: (x,y,z) | Point3d() **data_range**: Range of the data or array of ranges. Type: (x,y,z) | Point3d() **header**: Header part of the data message. Uses header from the first input if not set.

Logging

Logging a message causes GoPxL to add a message to the log at the bottom of the user interface. There are three types of log messages.

Logging Functions

Function	Description
log_info(msg)	Logs a user informational message. *Parameters* **msg**: Message to be logged.
log_warning(msg)	Logs a user warning message. *Parameters* **msg**: Message to be logged.
log_error(msg)	Logs a user error message and displays a pop-up containing the message. *Parameters* **msg**: Message to be logged.

Memory

You can use a Script tool's `memory` dictionary for storing values during runtime. The dictionary is erased when scanning starts.

> Some classes are only available through the `tl` namespace (for example, `tl.Anchor3d`). These classes are indicated below.

Data Structures

Data Structures

Data Structure	Description
Region	2D Region box (center-based).

Data Structure	Description
	Constructor Region(x=None, z=None, width=None, height=None) *Parameters* **x**: X position of the center of the region. **z**: Z position of the center of the region. **width**: Width of the region. **height**: Height of the region.
Region3d	3D Region box (center-based). *Constructor* Region3d(x=None, y=None, z=None, width=None, length=None, height=None, z_angle=None) *Parameters* **x**: X position of the center of the region. **y**: Y position of the center of the region. **z**: Z position of the center of the region. **width**: Width of the region. **height**: Height of the region. **length**: Length of the region. **z_angle**: Z Angle of the region.
Rect3d	3D rectangular region box. *Constructor* Rect3d(x=None, y=None, z=None, width=None, height=None, depth=None) *Parameters* **x**: X position of the center of the region. **y**: Y position of the center of the region. **z**: Z position of the center of the region. **width**: Width of the region. **height**: Height of the region. **depth**: Depth of the region.
tl.Anchor3d	3D anchor (center-based). Use the `Anchor3d` class to receive anchor information as *input*. For anchor outputs, use Measurement outputs instead. This class is only accessible through the `tl` namespace (`tl.Anchor3d`). *Constructor* Anchor3d(x=None, y=None, z=None, z_angle=None) *Parameters* **x**: X position of the anchor. **y**: Y position of the anchor. **z**: Z position of the anchor. **z_angle**: Z Angle of the anchor.
Point	2D Point class. *Constructor* Point(x=None, y=None) *Parameters*

Data Structure	Description
	x: X position of the Point. **y**: Y position of the Point.
Point3d	3D Point class. *Constructor* Point3d(x=None, y=None, z=None) *Parameters* **x**: X position of the Point. **y**: Y position of the Point. **z**: Z position of the Point.
tl.Stamp	Represents acquisition metadata associated with a data message. Note that `width`, `height`, `x`, and `y` parameters refer to the dimensions of the camera image; for more information, see `points` in *tl.SurfaceMsg* on page 284. *Constructor* Stamp(frame=None, time=None, signature=None, status=None, state=None, tag=None, encoder=None, encoder_snapshot=None, exposure=None, delay=None, width=None, height=None, y=None, x=None, x_subsampling=None, y_subsampling=None, temperature=None, subframe=None, reserved=None) *Parameters* **signature**: Equal to `kSTAMP_SIGNATURE` for camera messages; otherwise unused. **status**: Digital inputs, gate state, and other information at the time of message capture. **state**: Camera state id associated with message. **tag**: Optional tag that can be used for debugging or other special purposes. **frame**: Frame index of message data (counts up from zero). **time**: Timestamp corresponding to message data (µs). **encoder**: Encoder value corresponding to message data (encoder ticks). **encoderSnapshot**: Encoder value at time of most recent encoder snapshot signal (encoder ticks). **exposure**: Source image exposure (ns). **delay**: Delay before exposure (ns). **width**: Source camera image width, before subsampling. **height**: Source camera image height, before subsampling. **y**: Source camera image Y position. **x**: Source camera image X position. **x_subsampling**: X subsampling amount, expressed as power of 2. **y_subsampling**: Y subsampling amount, expressed as power of 2. **temperature**: Temperature corresponding to message data (centidegrees Celsius). **subframe**: Subframe status information. **reserved**: Reserved for future use.
Transform3d	3D transform matrix. *Constructor* Transform3d(xx=None, xy=None, xz=None, xt=None, yx=None, yy=None,

Data Structure	Description
	yz=None, yt=None, zx=None, zy=None, zz=None, zt=None) *Parameters* **xx**, **xy**, **xz**, **xt**: X vector (first three) and origin. **yx**, **yy**, **yz**: Y vector and origin. **zx**, **zy**, **zz**: Z vector and origin.
tl.MsgHeader	Header class of data message. *Constructor* MsgHeader(stamp=None, transform=None, bounding_box=None) *Parameters* **stamp**: Data stamp. **transform**: Transform matrix. **bounding_box**: Bounding box for the data.
PixelFormat	Enum class of pixel formats for image data message. **Mono formats** MONO_8 = <PixelFormat.MONO_8: 0x0001> MONO_10 = <PixelFormat.MONO_10: 0x0003> MONO_12 = <PixelFormat.MONO_12: 0x0005> MONO_14 = <PixelFormat.MONO_14: 0x0025> MONO_16 = <PixelFormat.MONO_16: 0x0007> **RGB formats** RGB_8 = <PixelFormat.RGB_8: 0x0014> BGR_8 = <PixelFormat.BGR_8: 0x0015> RGB_10 = <PixelFormat.RGB_10: 0x0018> BGR_10 = <PixelFormat.BGR_10: 0x0019> RGB_12 = <PixelFormat.RGB_12: 0x001A> BGR_12 = <PixelFormat.BGR_12: 0x001B> RGB_14 = <PixelFormat.RGB_14: 0x005E> BGR_14 = <PixelFormat.BGR_14: 0x004A> RGB_16 = <PixelFormat.RGB_16: 0x0033> BGR_16 = <PixelFormat.BGR_16: 0x004B> **YUV/YCBR formats** Y_CB_CR_8_CB_Y_CR = <PixelFormat.Y_CB_CR_8_CB_Y_CR: 0x003A> Y_CB_CR_411_8_CB_YY_CR_YY = <PixelFormat.Y_CB_CR_411_8_CB_YY_CR_YY: 0x003C> Y_CB_CR_422_8_CB_Y_CR_Y = <PixelFormat.Y_CB_CR_422_8_CB_Y_CR_Y: 0x0043>

Data Structure	Description
	Bayer formats BAYER_GR8 = <PixelFormat.BAYER_GR8: 0x0008> BAYER_RG8 = <PixelFormat.BAYER_RG8: 0x0009> BAYER_GB8 = <PixelFormat.BAYER_GB8: 0x000A> BAYER_BG8 = <PixelFormat.BAYER_BG8: 0x000B> BAYER_GR10 = <PixelFormat.BAYER_GR10: 0x000C> BAYER_RG10 = <PixelFormat.BAYER_RG10: 0x000D> BAYER_GB10 = <PixelFormat.BAYER_GB10: 0x000E> BAYER_BG10 = <PixelFormat.BAYER_BG10: 0x000F>
tl.ImageMsg	ImageMsg class for image data message. *Constructor* ImageMsg(header=None, pixel_format=None, flipped_x=False, flipped_y=False, transposed=False, pixels=None, offset = None, scale = None, unit = None) *Parameters* **header**: Header part of the data message. **pixel_format**: Pixel format of the image. For more information, see *PixelFormat* on the previous page. **flipped_x**: Set to `true` if the image data is flipped on the X axis. **flipped_y**: Set to `true` if the image data is flipped on the Y axis. **transposed**: Set to `true` if the image data is transposed. **pixels**: Pixel data of image of size (height, width, pixel size). **offset**: Offset of the image, or array of images. Type Point(). **scale**: Scale of the image, or array of images. Type Point(). **unit**: Unit for spatial dimensions, or array of images. Type UnitType enum.
UnitType	Enum class of units for spatial dimensions. NONE = <UnitType.NONE: 0> MILLIMETER = <UnitType.MILLIMETER: 1> PIXEL = <UnitType.PIXEL: 2> *Properties* **label**: Return the string label of the unit. Useful for display or formatting.
tl.SurfaceMsg	SurfaceMsg class for surface data message. *Constructor* SurfaceMsg(header=None, offset=None, scale=None, points=None, intensity=None, is_adjacent=None) *Parameters* **header**: Header part of the data message. **offset**: Offset of the data. **scale**: Scale of the data.

Data Structure	Description
	points: Surface point data of size (height, width) or (height,width,3). **intensity**: Surface intensity data of size (height, width). **is_adjacent**: Set to `true` if the surface data is adjacent.
tl.ProfileMsg	ProfileMsg class for profile data message. *Constructor* ProfileMsg(header=None, offset=None, scale=None, points=None, intensity=None) *Parameters* **header**: Header part of the data message. **offset**: Offset of the data. **scale**: Scale of the data (with x and z). This is the spacing between points. **points**: Profile point data of size (width) or (width,2) . **intensity**: Profile intensity data of size (width).
tl.MeasureMsg	MeasureMsg Class for measurement data message. *Constructor* MeasureMsg(header=None, value=None, status=None, label_position=None) *Parameters* **header**: Header part of the data message. **value**: Value of the data message. **status**: Types from the MeasurementDecision enum class. For more information, see *MeasurementDecision* below. **label_position**: Position of point in measurement message.
MeasurementDecision	Enum class of decision type for measurement data message. FAIL= <MeasurementDecision.FAIL: 0> PASS= <MeasurementDecision.PASS: 1>
tl.PointFeatureMsg	PoinFeaturetMsg class for point feature message. *Constructor* PointFeatureMsg(header=None, position=None) *Parameters* **header**: Header part of the data message. **position**: Position of feature point in data message.
tl.LineFeatureMsg	LineFeatureMsg class for line feature message. *Constructor* LineFeatureMsg(header=None, position=None, direction=None) *Parameters* **header**: Header part of the data message. **position**: Point vector in the line. **direction**: Direction vector of the line.
tl.CircleFeatureMsg	CircleFeatureMsg class for circular feature message. *Constructor*

Data Structure	Description
	CircleFeatureMsg(header=None, position=None, normal=None, radius=None) *Parameters* **header**: Header part of the data message. **position**: Center position of the circle. **normal**: Normal vector of the circular plane. **radius**: Radius of the circle.
tl.PlaneFeatureMsg	PlaneFeatureMsg class for planar feature message. *Constructor* PlaneFeatureMsg(header=None, normal=None, origin_distance=None) *Parameters* **header**: Header part of the data message. **normal**: Normal vector of the plane. **origin_distance**: Shortest distance from origin to plane.
tl.StringMsg	StringMsg class. *Constructor* StringMsg(header=None, text=None, decision=None, label_position=None) *Parameters* **header**: Header part of the data message. **text**: The string. **decision**: Decision of measurement message. **label_position**: Position of point in measurement message.
tl.MeshMsg	Sends a mesh message or array of messages to the specified output. *Parameters* **header**: Header part of the data message. Type: MsgHeader(). **vertex_data**: Vertex data points. Type: numpy array of floats (x,y,z). **facet_data**: Facet data points. Type: numpy array of uint32 triangle indexes (vertex 1, vertex 2,vertex 3). **vertex_normal_data**: Vertex normal data points. Type: numpy array of floats (x,y,z). **facet_normal_data**: Facet normal data points. Type: numpy array of floats (x,y,z). **vertex_texture_data**: Vertex texture data. Type: numpy array of uint8. **data_offset**: Offset of the data or array of offsets. Type: Type: (x,y,z) | Point3d() **data_range**: Range of the data or array of ranges. Type: (x,y,z) | Point3d()
MarkerShape(Enum)	Marker shape types for graphics. CROSS = 0 PLUS = 1 CIRCLE = 2 SQUARE = 3 TRIANGLE = 4 DIAMOND = 5

Examples

Measurement input and output

The following example shows how to read a measurement value input and output a new measurement value that is equal to the input multiplied by a factor of two.

```
# Get the measurement object at input 0
measurement = get_measurement(0)

# Extract the value from the measurement
measurement_value = measurement.value

# Calculate a new measurement value
measurement_output = 2 * measurement_value

# Send the calculated value as a new measurement to output 0
send_measurement(0, measurement_output)
```

The following example shows how to read measurement value inputs and write them to a CSV file. (This example is only intended for use on a PC instance of GoPxL, as writing to files on-sensor or on a GoMax device is blocked.)

```
import csv

# Get measurement objects
measurement_1 = get_measurement(0)
measurement_2 = get_measurement(1)

# Extract values from measurement objects
measurement_1_value = measurement_1.value
measurement_2_value = measurement_2.value

# Path to the output CSV file
output_path = r'C:\GoTools\Script\measurement.csv'

# Write measurements to CSV file
with open(output_path, 'a') as f:
    writer = csv.writer(f)
    writer.writerow([measurement_1_value, measurement_2_value])
```

Math operations

You can access mathematical operations through the standard Python math module.

```
# Import the math module to access mathematical functions
import math

# Get the measurement object at input 0
measurement = get_measurement(0)
```

```
# Extract the value from the measurement
measurement_value = measurement.value

# Calculate the square root of the absolute value using the Math module
sqrt_value = math.sqrt(math.fabs(measurement_value))

# Send the square root as a new measurement to output 0
send_measurement(0, sqrt_value)
```

Accessing valid/invalid and decision pass/fail

You can use the `is_valid()` function of the `MeasureMsg` class to check whether the value is valid or invalid. An invalid measurement is represented by the value `numpy.nan`. Other input classes also have `is_valid()` functions, which return `true` if the object data is not null.

The pass/fail criteria for the input `MeasureMsg` can be checked using the `decision` member variable.

```
# Check if any inputs are invalid
if any_input_invalid():
    send_all_invalid()

# Get the input measurement
measurement0 = get_measurement(0)

# Access validity from a measurement
if is_valid(measurement0):
    log_info("The input measurement is Valid")
else:
    log_info("The input measurement is Invalid")

# Access decision from a measurement
if measurement0.decision == MeasurementDecision.PASS:
    log_info("The input measurement has a Pass decision")
else:
    log_info("The input measurement has a Failed decision")
```

Smoothing / moving average of measurement values

This script shows how to use the memory feature to implement a smoothing filter / moving average of measurement values. Values from each frame are stored in the memory and with each new frame the oldest item is removed from memory before calculating the average.

```
import numpy as np

measurement_1 = get_measurement(0)
new_value = measurement_1.value
#log_info(f'Measurement values: {new_value}')
```

```
class MovingAverage:
    def __init__(self, window_size=10):
        self.window_size = window_size

    def add_value(self, value):
        memory.setdefault("Values", []).append(value)
        if len(memory["Values"]) > self.window_size:
            memory["Values"].pop(0)
        return self.calculate_average()

    def calculate_average(self):
        if len(memory["Values"]) == 0:
            return np.nan  # if no value present
        return np.mean(memory["Values"])

moving_avg = MovingAverage(window_size=10)
avg = moving_avg.add_value(new_value)
# log_info(f'Avg: {avg}')

send_measurement(0, avg)
```

Stamp access

```
profile = get_profile(0)

stamp = profile.header.stamp

log_info('Stamp = {}'.format(vars(stamp)))

send_measurement(0, stamp.frame)
send_measurement(1, stamp.time)
send_measurement(2, stamp.encoder)
```

Arrays

The following code reads an array input and outputs an array.

```
# Get an array of measurement objects at input 0
measurements = get_measurement(0)

# Extract the values from each measurement object
measurement_values = []
for measurement in measurements:
    measurement_values.append(measurement.value)

# Calculate new values
measurements_output = [value * 2 for value in measurement_values]

# Send the calculated values as an array to output 0
send_measurement(0, measurements_output)
```

Inverting a Surface in Z with null handling

The following code inverts a Surface in Z with null handling. It also shifts the output Surface in X by the bounding box width in order to not overlap the original.

```
surf = get_surface(0)

flipped_points = surf.points.copy()
flipped_offset = surf.offset

flipped_points[flipped_points != -32768] *= -1
flipped_offset.x += surf.header.bounding_box.width

send_surface(0, flipped_offset, surf.scale, flipped_points, surf.intensity)
```

Combining two point cloud surfaces into a single message

```
s0 = get_surface(0)
s1 = get_surface(1)

if s0.scale.x != s1.scale.x or s0.scale.y != s1.scale.y or s0.scale.z != s1.scale.z or
s0.offset.x != s1.offset.x or s0.offset.y != s1.offset.y or s0.offset.z != s1.offset.z:
    log_error("The input surfaces are not using the same resolution and offset. Cannot
concatenate.")
    log_info(f"Scale0 = {vars(s0.scale)}")
    log_info(f"Scale1 = {vars(s1.scale)}")

    log_info(f"Offset0 = {vars(s0.offset)}")
    log_info(f"Offset1 ={vars(s1.offset)}")

    return

points_final = numpy.concatenate((s0.points, s1.points))
intensity_final = numpy.concatenate((s0.intensity, s1.intensity))

send_surface(0, s0.offset, s0.scale, points_final, intensity_final, is_adjacent = False)
# Set is_adjacent to True to visualize polygons and intensity
```

```
s0 = get_surface(0)
s1 = get_surface(1)

if s0.scale.x != s1.scale.x \
    or s0.scale.y != s1.scale.y \
    or s0.scale.z != s1.scale.z \
    or s0.offset.x != s1.offset.x \
    or s0.offset.y != s1.offset.y \
    or s0.offset.z != s1.offset.z:

    log_error("The input surfaces do not use the same resolution and offset. Cannot
concatenate.")
    log_info(f"Scale0 = {vars(s0.scale)}")
    log_info(f"Scale1 = {vars(s1.scale)}")
```

```
    log_info(f"Offset0 = {vars(s0.offset)}")
    log_info(f"Offset1 ={vars(s1.offset)}")

    return

points_final = numpy.concatenate((s0.points, s1.points))
intensity_final = numpy.concatenate((s0.intensity, s1.intensity))

send_surface(0, s0.offset, s0.scale, points_final, intensity_final, is_adjacent = False)
# Set is_adjacent to True to visualize polygons and intensity
```

Surface Transform based on Plane

```
surface_obj = get_surface(0)
plane_obj = get_plane(1)
int_surface_points = surface_obj.points
intensity_data = surface_obj.intensity

# Store original shape and reshape for processing
original_shape = int_surface_points.shape
num_points = original_shape[0] * original_shape[1]
points_reshaped = int_surface_points.reshape((num_points, 3))

# Convert 3DPoint objects to numpy arrays for calculations
scale = numpy.array([surface_obj.scale.x, surface_obj.scale.y, surface_obj.scale.z])
offset = numpy.array([surface_obj.offset.x, surface_obj.offset.y, surface_obj.offset.z])
normal_vec = numpy.array([plane_obj.normal.x, plane_obj.normal.y, plane_obj.normal.z])

# Convert integer surface to floating-point "world space"
float_surface_points = points_reshaped.astype(float) * scale + offset

# Normalize the normal vector
norm_of_normal = numpy.linalg.norm(normal_vec)
if norm_of_normal == 0:
    raise ValueError("The normal vector cannot be a zero vector.")
unit_normal = normal_vec / norm_of_normal

# Find the center of the object
center_point = float_surface_points.mean(axis=0)
# Find a point on the plane to establish a reference
point_on_plane = plane_obj.origin_distance * unit_normal

# Determine if the normal points towards or away from the center
vec_to_center = center_point - point_on_plane
if numpy.dot(vec_to_center, unit_normal) > 0:
    # The normal points towards the center of the object. We flip it to get the
    # "outward" normal, ensuring the plane is consistently treated as the "top".
    unit_normal = -unit_normal

# Define the target vector
down_vector = numpy.array([0.0, 0.0, -1.0])

# Calculate the rotation matrix to align the normal with the down_vector
rotation_axis = numpy.cross(unit_normal, down_vector)
```

```
cos_angle = numpy.dot(unit_normal, down_vector)

if numpy.allclose(cos_angle, 1.0):
    rotation_matrix = numpy.identity(3)
elif numpy.allclose(cos_angle, -1.0):
    perp_axis = numpy.array([1.0, 0.0, 0.0])
    if numpy.allclose(numpy.abs(numpy.dot(unit_normal, perp_axis)), 1.0):
        perp_axis = numpy.array([0.0, 1.0, 0.0])
    u = perp_axis / numpy.linalg.norm(perp_axis)
    rotation_matrix = 2 * numpy.outer(u, u) - numpy.identity(3)
else:
    # shortcut to the sin between the vectors (based on cross product definition)
    sin_angle = numpy.linalg.norm(rotation_axis)
    rotation_axis /= sin_angle
    K = numpy.array([[0, -rotation_axis[2], rotation_axis[1]],
                     [rotation_axis[2], 0, -rotation_axis[0]],
                     [-rotation_axis[1], rotation_axis[0], 0]])
    rotation_matrix = numpy.identity(3) + sin_angle * K + (1 - cos_angle) * (K @ K)

# Create the full 4x4 homogeneous transformation matrix for rotation only
translation_to_origin = numpy.identity(4)
translation_to_origin[:3, 3] = -center_point
rotation_4x4 = numpy.identity(4)
rotation_4x4[:3, :3] = rotation_matrix
translation_back = numpy.identity(4)
translation_back[:3, 3] = center_point
transformation_matrix = translation_back @ rotation_4x4 @ translation_to_origin

homogeneous_surface = numpy.hstack([float_surface_points, numpy.ones((num_points, 1))])
transformed_homogeneous = (transformation_matrix @ homogeneous_surface.T).T
transformed_float_surface = transformed_homogeneous[:, :3]

# Calculate a new offset to prevent overflow
min_coords = transformed_float_surface.min(axis=0)
```

Minimum and maximum Z value of a uniform Surface with null handling

```
surf = get_surface(0)

s = surf.points[surf.points != -32768]

max_z = numpy.max(s) * surf.scale.z + surf.offset.z
min_z = numpy.min(s) * surf.scale.z + surf.offset.z

send_measurement(0, min_z)
send_measurement(1, max_z)
```

Exporting a surface as a PCD file

```
import numpy as np
```

```
filename = "outsurface.pcd"
msg = get_surface(0)
height_map = msg.points
intensity_map = msg.intensity
scale = msg.scale


rows, cols = height_map.shape


# Prepare the header for the PCD file
header = (
    "# .PCD v0.7 - Point Cloud Data file format\n"
    "VERSION 0.7\n"
    "FIELDS x y z intensity\n"
    "SIZE 4 4 4 4\n"
    "TYPE F F F F\n"
    "COUNT 1 1 1 1\n"
    f"WIDTH {cols}\n"
    f"HEIGHT {rows}\n"
    "VIEWPOINT 0 0 0 1 0 0 0\n"
    f"POINTS {rows * cols}\n"
    "DATA ascii\n"
)


# Write the header and points to the PCD file
with open(filename, 'w') as file:
    file.write(header)
    for i in range(rows):
        for j in range(cols):
            x = i * scale.x
            y = j * scale.y
            z = height_map[i, j] * scale.z if height_map[i, j] != -32768 else 0
            intensity = intensity_map[i, j] if height_map[i, j] != -32768 else 0
            file.write(f"{x} {y} {z} {intensity}\n")


log_info(f"PCD file saved successfully as {filename}")
```

Grid of coordinates for anchoring

The following script shows how you can perform the same inspection using a single tool at many locations on a grid. It generates X and Y measurement arrays that you use as anchor inputs in another tool (the anchored tool).

In the anchored tool, you must check **Enable batching** so that the tool generates an array of outputs.

```
x = numpy.linspace(-15, 15, 10) # 10 points spanning 0..30 mm range
y = numpy.linspace(-15, 15, 10) # 10 points spanning 0..30 mm range
```

```
xv, yv = numpy.meshgrid(x, y)

send_measurement(0, xv.flatten())
send_measurement(1, yv.flatten())
```

Image Processing (Subtracting two images and normalizing the result)

```
im0 = get_image(0)
im1 = get_image(1)

delta = im0.pixels.astype(numpy.float32) - im1.pixels.astype(numpy.float32)
stdev = numpy.std(delta)

new_pixels = numpy.clip(delta / (2*stdev) * 128 + 128, 0, 255).astype(numpy.uint8)

send_image(0, new_pixels, PixelFormat.RGB_8)
send_measurement(1, stdev)
```

Image processing (forward to HTTP server)

```
# Script (using built in urlib package)
import urllib.request
import json
import numpy as np

# Assuming get_image(0).pixels returns a numpy array
array = get_image(0).pixels
array_bytes = array.tobytes()

# Encode shape information as JSON in the headers
headers = {'Content-Type': 'application/octet-stream', 'X-Array-Shape': json.dumps
(array.shape)}
url = 'http://127.0.0.1:5000/receive_array'

# Encode array bytes data and prepare request
req = urllib.request.Request(url, data=array_bytes, headers=headers)
urllib.request.urlopen(req)


# SERVER (server.py)

from flask import Flask, request, Response
import numpy as np
import cv2
import json

app = Flask(__name__)

@app.route('/receive_array', methods=['POST'])
def receive_array():
```

```
    # Get the content of the request
    data = request.data

    # Extract shape information from headers
    shape_info = json.loads(request.headers.get('X-Array-Shape'))
    height, width, channels = shape_info

    # Extract image bytes
    image_array = np.frombuffer(data, dtype=np.uint8)
    image_array = image_array.reshape((height, width, channels))

    # Display image
    #cv2.imshow('Received Image', image_array)
    #cv2.waitKey(0)
    #cv2.destroyAllWindows()

    # Print size of received image
    print("Received image size:", image_array.shape)

    return Response(status=200)

if __name__ == '__main__':
    app.run(debug=False, threaded=True, port=5000)
```

The following is a sample client to demonstrate the functionality of the HTTP server forwarding snippet above.

```
# SAMPLE CONSOLE CLIENT (client.py)
# A standalone demonstration of the functionality used by the image HTTP
# forwarding script.

import requests
import numpy as np
import json

# Assuming you have a function to generate numpy arrays as images
def generate_numpy_array():
    # Generating a sample numpy array
    array = np.random.rand(100, 100, 3) * 255  # Random 100x100x3 array of floats between
0 and 255
    return array.astype(np.uint8)

def send_array_to_server(array):
    url = 'http://127.0.0.1:5000/receive_array'
    array_bytes = array.tobytes()

    # Encode shape information as JSON in the headers
    headers = {'Content-Type': 'application/octet-stream', 'X-Array-Shape': json.dumps
(array.shape)}

    response = requests.post(url, data=array_bytes, headers=headers)

# Generate and send arrays to the server
while True:
    array = generate_numpy_array()
    send_array_to_server(array)
```

Image Processing - Warp an image based on four corners

```
import cv2
import numpy as np

def warp_image_to_rectangle(image: np.ndarray, src_corners: np.ndarray, output_size:
tuple) -> np.ndarray:
    """
    Resample an image to a rectangle based on four corner points.

    :param image: Input image as a NumPy array.
    :param src_corners: Source corner points as a (4, 2) NumPy array (float32).
    :param output_size: Desired output size (width, height).
    :return: Warped and resampled image.
    """
    if src_corners.shape != (4, 2):
        raise ValueError("src_corners must be a (4, 2) array of corner points.")

    width, height = output_size

    # Define destination rectangle corners
    dst_corners = np.array([
        [0, 0],
        [width - 1, 0],
        [width - 1, height - 1],
        [0, height - 1]
    ], dtype=np.float32)

    # Compute perspective transform matrix
    matrix = cv2.getPerspectiveTransform(src_corners.astype(np.float32), dst_corners)

    # Warp the image
    warped = cv2.warpPerspective(image, matrix, (width, height), flags=cv2.INTER_LINEAR)

    return warped

if any_input_invalid():
    send_all_invalid()
    return

image = get_image(0).pixels
c0 = get_point(1).position
c1 = get_point(2).position
c2 = get_point(3).position
c3 = get_point(4).position

src_corners = np.array([
    [c0.x, c0.y],
    [c1.x, c1.y],
    [c2.x, c2.y],
    [c3.x, c3.y]
], dtype=np.float32)

output_size = (800, 400)
result = warp_image_to_rectangle(image, src_corners, output_size)
result = result.reshape((output_size[1], output_size[0], 1))

send_image(0, result, PixelFormat.MONO_8)
```

Image Processing (Sending out an image with calibrated units)

```
im = get_image(0)
h = im.header
offset = Point(im.offset.x - 100 , im.offset.y - 100)
scale = Point(im.scale.x / 10, im.scale.y / 10)

send_image(0, im.pixels, pixel_format= PixelFormat.RGB_8,
           offset = offset, scale = scale, unit = UnitType.MILLIMETER)
```

Profile processing (using a profile from disk as a correction to new profiles from sensor)

```
# Save profile template to disk
import csv

profile = get_profile(0)
points = profile.points

template_path = r'C:\GoTools\Script\profile.csv'
numpy.savetxt(template_path, points, delimiter=',', fmt='%d')

send_measurement(0, 1)

# Load and correct
import csv

profile = get_profile(0)
points = profile.points

# load template
template_path = r'C:\GoTools\Script\profile_template.csv'

if 'template' not in memory:
    template_points = numpy.genfromtxt(template_path, delimiter=',', dtype=numpy.int16)
    memory['template'] = template_points
    log_info("Loaded template")
else:
    template_points = memory['template']

# subtract while handling null values
mask = (template_points != -32768) & (points != -32768)
result = numpy.full_like(points, -32768)
numpy.subtract(points, template_points, out=result, where=mask, dtype=numpy.int16)

send_profile(0, profile.offset, profile.scale, result)

# optionally send original profile and template as outputs
send_profile(1, profile.offset, profile.scale, points)
send_profile(2, profile.offset, profile.scale, template_points)

# optionally save all three into a new CSV file

output_path = r'C:\GoTools\Script\profile_log.csv'
```

```
combined_array = numpy.vstack((points, template_points, result)).T

with open(output_path, 'w', newline='') as csv_file:
    writer = csv.writer(csv_file)
    writer.writerow(['Profile', 'Template', 'Difference'])
    writer.writerows(combined_array)
```

Getting function signatures

Instead of looking up functions in the manual to see arguments to be passed in, you can use the `signature` function in the `inspect` module to get the signature of any function. For example, the code below displays an error message: (`index, value, label_position=None, header=None`).

```
from inspect import signature

log_error(str(signature(send_measurement)))
```

FTP - Writing surface data to disk, and sending it to an FTP server

This sample was tested with Rebex Tiny FTP server.

```
import os
import struct
import zlib
from datetime import datetime
from ftplib import FTP
import numpy as np
import time

# FTP Configuration
FTP_HOST = '192.168.1.200'       # CHANGE THIS TO THE FTP HOSTNAME OR IP ADDRESS
FTP_USER = 'tester'              # Default User for Rebex Tiny FTP
FTP_PASS = 'password'            # Default Password for Rebex Tiny FTP

try:
    # Get the surface data
    surface = get_surface(0)

    # Convert surface points to numpy array (height data)
    height_data = np.array(surface.points)

    # Intensity data (already in the 0-255 range, so no normalization needed)
    intensity_data = np.array(surface.intensity)

    # Normalize height data to 0-255, handle NaNs, and convert to uint8
    def normalize(data):
        return np.nan_to_num(((data - np.nanmin(data)) / (np.nanmax(data) - np.nanmin
(data) + 1e-8)) * 255).astype(np.uint8)

    # Correct orientation (flip vertically and transpose)
```

```
    def correct_orientation(arr):
        return np.flipud(arr.T)

    # Normalize and correct the orientation for height data
    height_img = correct_orientation(normalize(height_data))

    # Intensity data is already in the range 0-255, just ensure it is uint8
    intensity_img = correct_orientation(intensity_data.astype(np.uint8))

    # Write the PNG image to the specified path
    def write_png(path, image):
        try:
            if image.size == 0:
                log_info("Image is empty. Skipping write.")
                return

            # Helper function to create PNG chunks
            def chunk(typ, dat):
                return struct.pack('>I', len(dat)) + typ + dat + struct.pack('>I',
zlib.crc32(typ + dat) & 0xffffffff)

            # Open the file and write the PNG data
            with open(path, 'wb') as f:
                f.write(b'\x89PNG\r\n\x1a\n')
                f.write(chunk(b'IHDR', struct.pack(">IIBBBBB", image.shape[1], image.shape
[0], 8, 0, 0, 0, 0)))  # Write header chunk
                raw = b''.join([b'\x00' + image[i].tobytes() for i in range(image.shape
[0])])  # Write pixel data
                f.write(chunk(b'IDAT', zlib.compress(raw)))  # Write compressed image data
                f.write(chunk(b'IEND', b''))  # End of PNG file
                f.flush()  # Ensure data is written
                os.fsync(f.fileno())  # Sync the file to disk

        except Exception as e:
            log_info(f"Exception in write_png: {e}")

    # Define output directory and file paths for saving images
    output_dir = r'C:\Temp'  # Use a simpler directory for testing
    os.makedirs(output_dir, exist_ok=True)  # Create the output directory if it doesn't
exist
    timestamp = datetime.now().strftime('%Y%m%d_%H%M%S')
    path_height = os.path.join(output_dir, f"{timestamp}_height.png")
    path_intensity = os.path.join(output_dir, f"{timestamp}_intensity.png")

    # Write the height and intensity images to files
    try:
        write_png(path_height, height_img)
        log_info(f"Height image written: {path_height}")
    except Exception as e:
        log_info(f"Error writing height image: {e}")

    try:
        write_png(path_intensity, intensity_img)
        log_info(f"Intensity image written: {path_intensity}")
    except Exception as e:
        log_info(f"Error writing intensity image: {e}")

    # Function to wait for file size to stabilize before proceeding with upload
    def wait_for_file_stability(path, check_interval=0.1, max_wait_time=2):
```

```
        last_size = -1
        start_time = time.time()

        # Continuously check if the file size has stabilized within the given timeout
        while time.time() - start_time < max_wait_time:
            try:
                current_size = os.path.getsize(path)
                if current_size == last_size:
                    # If file size is stable, exit the loop
                    return True
                last_size = current_size
                time.sleep(check_interval)  # Sleep for a very short interval before
checking again
            except FileNotFoundError:
                pass  # Ignore if the file doesn't exist yet
        return False

    # FTP upload function to send files
    def upload(path):
        try:
            with FTP(FTP_HOST) as ftp:
                ftp.login(FTP_USER, FTP_PASS)
                with open(path, 'rb') as f:
                    ftp.storbinary(f'STOR {os.path.basename(path)}', f)  # Upload the file
            return 1
        except Exception as e:
            log_info(f"FTP error: {e}")
            return 0

    # Upload both height and intensity images via FTP
    send_measurement(0, upload(path_height))
    send_measurement(1, upload(path_intensity))

except Exception as e:
    log_info(f"Error in processing surface data: {e}")
    send_measurement(0, 0)  # Indicate failure if an exception occurs during processing
```

Moving Average Using Memory

```
import numpy as np


class MovingAverage:
    def __init__(self, window_size=10):
        self.window_size = window_size

    def add_value(self, value):

        # Add value if the 'Values' key is present else initialize with an empty list.
        memory.setdefault("Values", []).append(value)
        if len(memory["Values"]) > self.window_size:
            memory["Values"].pop(0)
        return self.calculate_average()

    def calculate_average(self):
```

```
        if not memory["Values"]:
            return np.nan
        return np.mean(memory["Values"])


# Get values
measurement = get_measurement(0)
measurement_value = measurement.value

# Process values
moving_avg = MovingAverage(window_size=10)
avg = moving_avg.add_value(measurement_value)

# Send values
send_measurement(0, measurement_value)
send_measurement(1, avg)
```

Communicate with Another Sensor via ASCII Using Socket Programming

```
import socket

HOST = "192.168.1.10"
PORT = 8190

DELIMITER = ','
TERMINATION = '\r\n'  # Matches %r%n set by default in GoPxL > Ethernet ASCII page

START_COMMAND = f'start{TERMINATION}'
STOP_COMMAND = f'stop{TERMINATION}'
TRIGGER_COMMAND = f'trigger{TERMINATION}'
SYSTEM_STATE_COMMAND = f"readprop{DELIMITER}/system#/runState{TERMINATION}"

try:
    s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    s.settimeout(15)
    s.connect((HOST, PORT))
    log_info(f"Connected to sensor at {HOST}:{PORT}")

    _ = s.send(SYSTEM_STATE_COMMAND.encode()) # Requires data to be sent in byte format
    response = s.recv(1024).decode().strip().split(',') # Receives up to 1024 bytes of
data from the socket connection
    log_info(f"Response received: {response}")

    if len(response) > 1 and response[0] == "OK":
        run_state = response[1]
        if run_state == "0": # If sensor is not running, first run then trigger
            _ = s.send(START_COMMAND.encode())
            _ = s.send(TRIGGER_COMMAND.encode())
        elif run_state == "1":
            _ = s.send(TRIGGER_COMMAND.encode())
        else:
            log_info(f"Unexpected system state: {run_state}")

except socket.timeout:
    log_error(f"Error: Operation timed out. ASCII is disabled or sensor is unreachable at
```

```
{HOST}:{PORT}.")
except Exception as e:
    log_error(f"An error occurred: {e}")
finally:
    s.close()
    log_info("Socket closed.")
```

Save screenshot with the displayed outputs in the data viewer

Before using this sample, you must install two python modules: selenium and webdriver_manager. Note that this script is intended for use on GoPxL for PC.

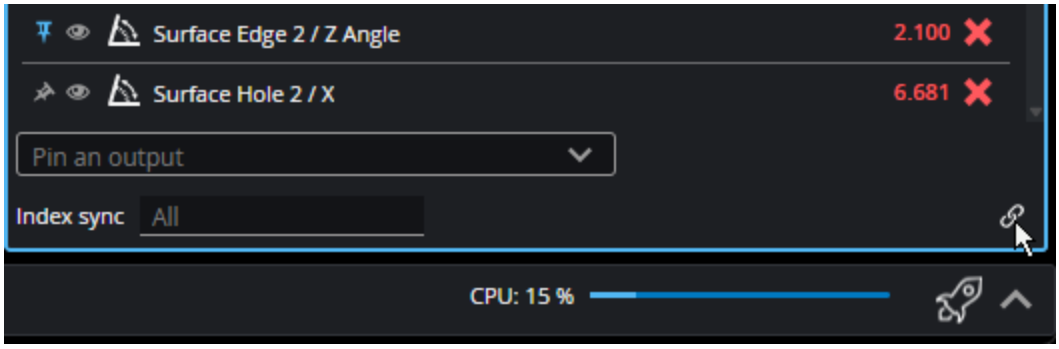
1. Using Command Prompt or PowerShell, go to the `GoPxl_UserTools\bin\win64\python` folder.
2. (Optional) To show the installed Python modules, `run python3.exe -m pip list`.
3. To install selenium, run `python3.exe -m pip install selenium`.
4. To install webdriver_manager, run `python3.exe -m pip install webdriver_manager`.

To save a screenshot in a script, after adding a Script tool and copying and pasting the sample code into the **Code** field, do the following:

1. If it's not already open, expand the **Displayed outputs** pane.
2. Configure which outputs you want displayed in the data viewer by pinning or hiding outputs.

   For more information on the Displayed outputs pane, see *Displaying and Pinning Outputs* on page 87.
3. Click the link icon at the lower right.

   This copies the current state of the data viewer, including the currently .



4. Paste the link into the code as the value of `url`.

The script saves a .png file to the location in the script.

```
from selenium import webdriver
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
from webdriver_manager.chrome import ChromeDriverManager
from datetime import datetime
import os
```

```
if not hasattr(self, 'long_mem'):
    self.long_mem = {}

if "service" not in self.long_mem:
    self.long_mem["service"] = Service(ChromeDriverManager().install())
service = self.long_mem["service"]

# Set the URL here
url = "www.google.com"
log_info(f"{type(url)}")

if "driver" not in self.long_mem:
    chrome_options = Options()
    chrome_options.add_argument("--headless=new")  # Use "--headless=new" for newer
versions of Chrome
    chrome_options.add_argument("--window-size=1920,1080")  # Important for full
screenshots
    #chrome_options.add_argument("--disable-gpu")  # Optional: helpful on Windows

    self.long_mem["driver"] = webdriver.Chrome(service=service, options=chrome_options)
    self.long_mem["last_url"] = url  # Initialize last_url to None

# Check if the URL has changed
if self.long_mem["last_url"] != url:
    log_info(f"changing url to {url}")
    self.long_mem["driver"].get(url)
    self.long_mem["last_url"] = url  # Update last_url to the current URL

driver = self.long_mem["driver"]

# Create a timestamped filename
timestamp = datetime.now().strftime('%Y-%m-%d_%H-%M-%S')
filename = f"screenshot_{timestamp}.png"

# Optional: specify a directory to save screenshots
output_dir = os.path.expanduser("~/Pictures/Screenshots")
os.makedirs(output_dir, exist_ok=True)

filepath = os.path.join(output_dir, filename)
driver.save_screenshot(filepath)

log_info(f"{filepath} saved")

send_measurement(0,0)
```

Buffering a sequence of surfaces and outputting them as an array

```
# specify the captured count when we send out the array
# it should be the same as the value of the parameter `Surface Count` in the Surface
Stitch tool
capturedCount = 3

surf = get_surface(0)
```

```
if 'captured' not in memory: # memory is type of dict
    memory['captured'] = [] # a list with fixed items

# if 'captured' not in memory or len(memory['captured']) != capturedCount: # memory is
type of dict
#     memory['captured'] = [None] * capturedCount # a list with fixed items

captured = memory['captured']

surf = get_surface(0)
captured.append(surf)

if len(captured) == capturedCount: # if the passed Captured value is equal to the item
count of the list
    offsets = []
    scales = []
    points = []
    intensities = []
    headers = []

    for s in captured:

        str = type(s)
        log_info(str)

        offsets.append(s.offset)
        scales.append(s.scale)
        points.append(s.points)
        intensities.append(s.intensity)

    send_surface(0, offsets, scales, points, intensity=intensities, header = surf.header)
    log_info('sending out a surface point cloud array')
    memory['captured'].clear()
```

Down-sampling Surface Point Cloud Array

```
import numpy as np

# Downsample factor
FACTOR = 4

# Get the list of surfaces
surfaces = get_surface(0)

if surfaces is not None:
    # Initialize lists to hold the output batch data
    out_points = []
    out_intensities = []
    out_scales = []
    out_offsets = []
    out_adjacents = []

    for surf in surfaces:
```

```
        # Downsample Points/Intensity
        # Use slicing [start::step] to keep every Nth pixel
        out_points.append(surf.points[::FACTOR, ::FACTOR])
        out_intensities.append(surf.intensity[::FACTOR, ::FACTOR])

        # Pass through scale, offset & is_adjacency
        out_scales.append(surf.scale)
        out_offsets.append(surf.offset)
        out_adjacents.append(surf.is_adjacent)

    # Send the surface point cloud array
    send_surface(
        0,
        offset=out_offsets,
        scale=out_scales,
        points=out_points,
        intensity=out_intensities,
        is_adjacent=out_adjacents
    )
```

Drawing Graphics

This script demonstrates the use of points, lines, regions, and labels.

```
# Calculates the average of two points and outputs coordinates
# as individual measurements and a combined Point feature.
# Inputs should be configured as:
#   Input 0 - Surface
#   Input 1 - Point 1 X
#   Input 2 - Point 1 Y
#   Input 3 - Point 2 X
#   Input 4 - Point 2 Y
# Outputs should be configured as:
#   Output 0 - Measurement (Midpoint X)
#   Output 1 - Measurement (Midpoint Y)
#   Output 2 - Point (Midpoint)

if any_input_invalid():
    send_all_invalid()
    return

# 1. Configuration: Visual Styles
POINT_SIZE   = 15
MID_SIZE     = 10

# 2. Data Extraction
x1, y1 = get_measurement(1).value, get_measurement(2).value
x2, y2 = get_measurement(3).value, get_measurement(4).value


# 3. Math: Calculate the Midpoint
mx = (x1 + x2) / 2.0
my = (y1 + y2) / 2.0
```

```
midpoint = (mx, my)

# 4. Visualization
# Draw a connection line between the two source points
draw_lines([(x1, y1), (x2, y2)], color=Colors.GRAY, width=1)

# Draw a region that perfectly bounds the two points
# Using abs() ensures valid positive dimensions for the region width and height
draw_region(midpoint, (abs(x2 - x1), abs(y2 - y1), 1), 0.0)

# Draw source points as white circles
# For the first point, make the color dependent on the decision Pass/Fail of the input
measurement
pointColor1 = Colors.RED
if get_measurement(1).decision == MeasurementDecision.PASS:
    pointColor1 = Colors.GREEN
draw_points((x1, y1), color=pointColor1, size=POINT_SIZE, shape=MarkerShape.CIRCLE)
draw_points((x2, y2), color=Colors.WHITE, size=POINT_SIZE, shape=MarkerShape.CIRCLE)

# Draw crosshair lines centered on the midpoint
# lineX spans the Y-distance; lineY spans the X-distance
# The lines are associated the X and Y tool outputs, so
# they are only drawn when that output is selected
lineX = [(mx, y1), (mx, y2)]
lineY = [(x1, my), (x2, my)]

draw_lines(lineX, color=Colors.CYAN, width=2, output=0)
draw_lines(lineY, color=Colors.CYAN, width=2, output=1)

# Label the midpoint with its coordinates
draw_label(f"Mid: {mx:.2f}, {my:.2f}", (mx, my))

# 5. Output Results
send_measurement(0, mx)
send_measurement(1, my)
send_point(2, midpoint)
```

Drawing green / red border around a Surface for Pass/Fail of a measurement

```
# --- GoPxL Surface Bounding Box Example ---
# Draws a red/green outline around the surface data extents
# based on the pass/fail decision of a specific measurement.
# Also outputs a string PASS/FAIL that can be pinned to keep
# the graphics visible.

if any_input_invalid():
    send_all_invalid()
    return

# 1. Configuration & Data Extraction
# For surface data:
surface = get_surface(0)
# For image data:
# image = get_image(0)
decision_meas = get_measurement(1)
```

```
# Offset and scale are also available on images
ox = surface.offset.x
oy = surface.offset.y

sx = surface.scale.x
sy = surface.scale.y

# Get dimensions from the points array shape (height, width)
height = surface.points.shape[0]
width = surface.points.shape[1]
# For images:
# image.pixels.shape[0]
# image.pixels.shape[1]

# 2. Math: Calculate Bounding Box Extents
min_x, min_y = ox, oy
max_x = ox + (width * sx)
max_y = oy + (height * sy)

# 3. Decision Logic
status_color = Colors.GREEN if decision_meas.decision == MeasurementDecision.PASS else
Colors.RED

# 4. Visualization
# Define the four corners of the surface area
outline_points = [
    (min_x, min_y), (max_x, min_y), # Top Edge
    (max_x, min_y), (max_x, max_y), # Right Edge
    (max_x, max_y), (min_x, max_y), # Bottom Edge
    (min_x, max_y), (min_x, min_y)  # Left Edge
]

# Draw the border and associate it with output 0 so it can be pinned
draw_lines(outline_points, color=status_color, width=5, output=0)

# 5. Output a string PASS/FAIL
send_string(0, "PASS" if  decision_meas.decision == MeasurementDecision.PASS else "FAIL")
```

User Data Input Tool

The User Data Input tool lets you send data to GoPxL from PLCs over EtherNet/IP, PROFINET, or via the SDK. After you configure the tool to indicate the data types and their offsets, the tool makes the data available in its outputs. A typical use case for this is to use the tool's string output as part of file names output with the data storage service, for example, part ID information. For information on the data storage service, see *Data Storage Service* on page 378.

For information on EtherNet/IP, see *EtherNet/IP Protocol* on page 324.

For information on PROFINET, see *PROFINET Protocol* on page 359.

For information on adding, managing, and removing tools, as well as detailed descriptions of settings common to most tools, see *Tool Configuration* on page 125.

Inputs

You configure the tool's inputs in the expandable **Inputs** section.

Inputs

Name	Description
Enable Batching	When **Enable Batching** is checked, the tool takes an array as input and processes each element in the array individually. There is no limit to the size of the array, other than the processing limitations of the sensor. When **Enable Batching** is unchecked, the tool takes either a single profile, or an array of profiles with a maximum of two elements. When **Enable Batching** is unchecked, the tool takes either a single surface, or an array of surfaces with a maximum of two elements. If the input is an array, the tool will aggregate the data in the array, and process that data as a whole. When **Enable Batching** is unchecked, the tool takes either a single profile, or an array of profiles with any number of elements. When **Enable Batching** is unchecked, the tool takes either a single surface, or an array of surfaces with any number of elements. If the input is an array, the tool will aggregate the data in the array, and process that data as a whole. For more information on arrays, batching, and aggregating, see *Arrays, Batching, and Aggregation* on page 126.
Trigger Input	The input the tool takes. This tool can optionally take an array as input. For more information, see Arrays, Batching, and Aggregation.

Parameters

You configure the tool's parameters in the expandable **Parameters** section.

Parameters

Name	Description
Byte Order	The "endianness" of the data being injected into the tool chain. For PLCs, consult

Name	Description
	the PLC manufacturer's documentation.
Output Collection	An expandable section containing a list of the data GoPxL is receiving. For each piece of data the tool is receiving, set the address and data type.
External ID	The external ID of the tool that appears in GoHMI Designer. For more information, see *GoHMI and GoHMI Designer* on page 425.

Outputs

This tool provides no outputs other than saving exported data in `C:\GoTools\DataExport`.

To set whether measurement values are considered a Pass or Fail, select a measurement in the outputs list, and then enable and configure its **Min** and **Max** parameters in the **Output Properties** section; you can enable one or the other, or both.

By default, **Min** and **Max** are both disabled, and a measurement is considered a Pass.

Data

Type	Description
Output {n}	The outputs available to other tools.

Communicate - PLC, Digital, GDP, HMI

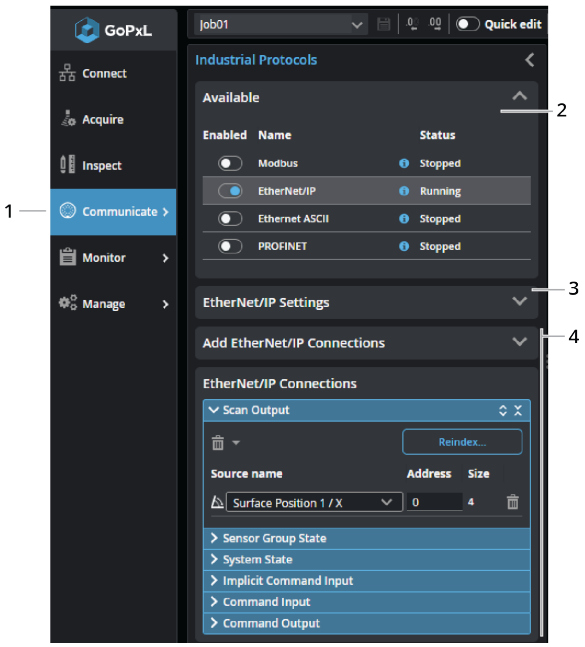
External devices such as PLCs (programmable logic controllers) can control Gocator sensors using various communication protocols and hardware interfaces. Sensors can also send scan data, measurement results, and measurement decisions to these devices. PLCs can in turn control other devices such as ejection chutes.

You can also use the GoPxL Data Protocol (GDP) with software written using the GoPxL SDK and REST API to control a sensor and receive data.

Lastly, on sensors and smart cameras, GoPxL can send acquired data to a remote FTP server. On GoMax Smart Vision Accelerators and GoPxL for PC, GoPxL can also save data to a local folder.

> If you are using a GoMax Smart Vision Accelerator, only connect Ethernet-based PLCs or PCs to the ETH1 port on the GoMax device (whether via a switch or not).

The following sections describe the **Communicate** category and the available protocols and interfaces.



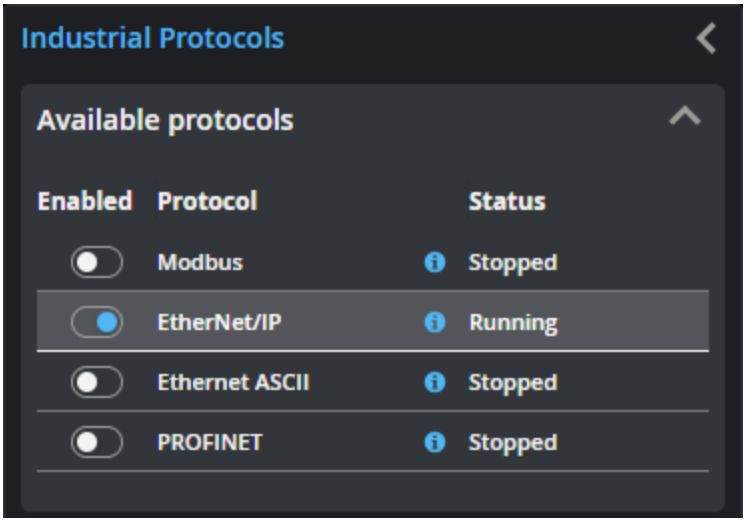
To select the communication type, expand **Communicate**, and choose the required type. The last selected type remains if you switch to a different category or page.

	Element	Description
1	Communicate category	Pages corresponding to the different control and data output methods available in GoPxL. For information on the available control methods, see *Communication interfaces* below.
2	Available Protocols	When you are on the Industrial page, this displays the available industrial protocols and whether a protocol is enabled. Use the toggles to start a protocol.
3	Settings	Lets you configure protocols.
4	Connections	Lets you add and remove the connections (such as stamps, measurements and decisions) to the connections map (industrial protocols) and outputs (Gocator protocol).

For more information on the available communication methods, see *Communication interfaces* below.

Communication interfaces

Interface	Description
GDP	The **Communicate** > **GDP** page lets you enable and configure the GoPxL Data Protocol, which sends to data and measurements to external applications that use the GoPxL SDK. For more information, see *GoPxL Data Protocol* on page 312. For information on configuring the GoPxL Data Protocol in GoPxL, see *GoPxL Data Protocol* on page 312.

Interface	Description
	For information on the SDK and REST API, see *GoPxL SDK and REST API* on page 449.
Industrial	The Industrial page lets you enable and configure the following protocols over an Ethernet connection: • Modbus • EtherNet/IP • Ethernet ASCII • PROFINET  **Modbus** The Modbus protocol lets you communicate with a sensor from a PLC that supports Modbus. For more information on configuring the Modbus protocol in GoPxL, see *Modbus TCP Protocol* on page 315. For the Modbus protocol specification, see *Modbus TCP Protocol Format - PLC Information* on page 319. **EtherNet/IP** The EtherNet/IP protocol lets you communicate with a sensor from a PLC that supports EtherNet/IP. For more information on configuring the EtherNet/IP protocol in the GoPxL UI, see *EtherNet/IP Protocol* on page 324. For the EtherNet/IP protocol specification, see *EtherNet/IP Protocol Format - PLC Information* on page 331. **Ethernet ASCII** The Ethernet ASCII protocol lets you communicate with a sensor using an

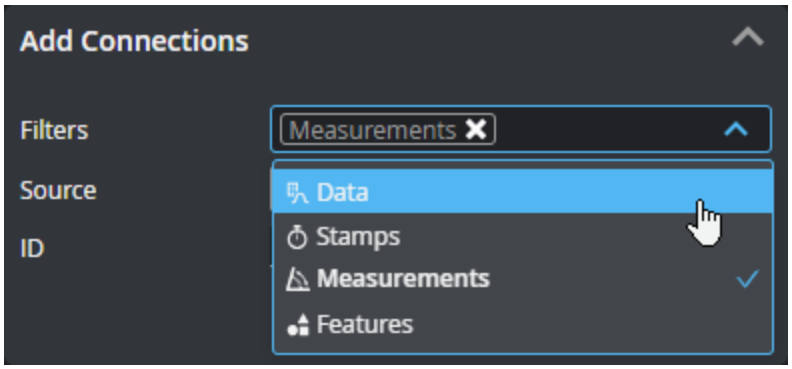
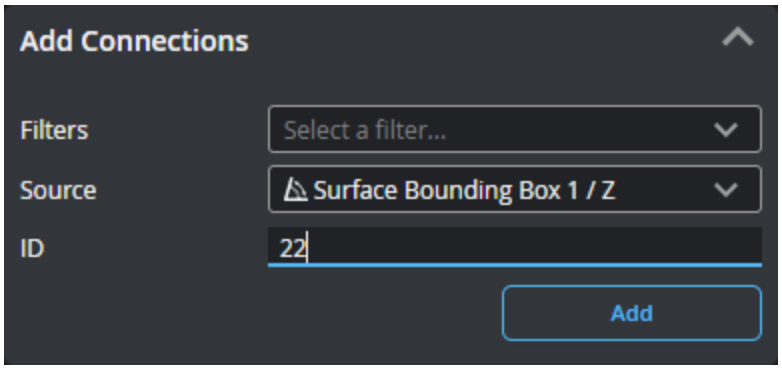
Interface	Description
	ASCII format. For more information on configuring the ASCII protocol over Ethernet in the GoPxL interface, see *Ethernet ASCII Protocol* on page 345. For a list of the commands, see *Ethernet ASCII Protocol Format - PLC Information* on page 350. **PROFINET** The PROFINET protocol lets you communicate with a sensor from a PLC that supports PROFINET. For more information on configuring the PROFINET protocol in the GoPxL UI, see *PROFINET Protocol* on page 359. For the PROFINET protocol specification, see *PROFINET Protocol Format - PLC Information* on page 364.
Digital	Lets you configure digital outputs. For more information, see *Digital Output* on page 369.
GoHMI	GoHMI and the GoHMI Designer let you create a custom user interface connected to Gocator outputs that users can access from a browser on a PC, a touchscreen, or a mobile device. For more information, see *GoHMI and GoHMI Designer* on page 425.
Data Storage	Lets you configure storing acquired data to a remote FTP server or to local storage. Note that sensors can only store data on a remote server. For more information, see .

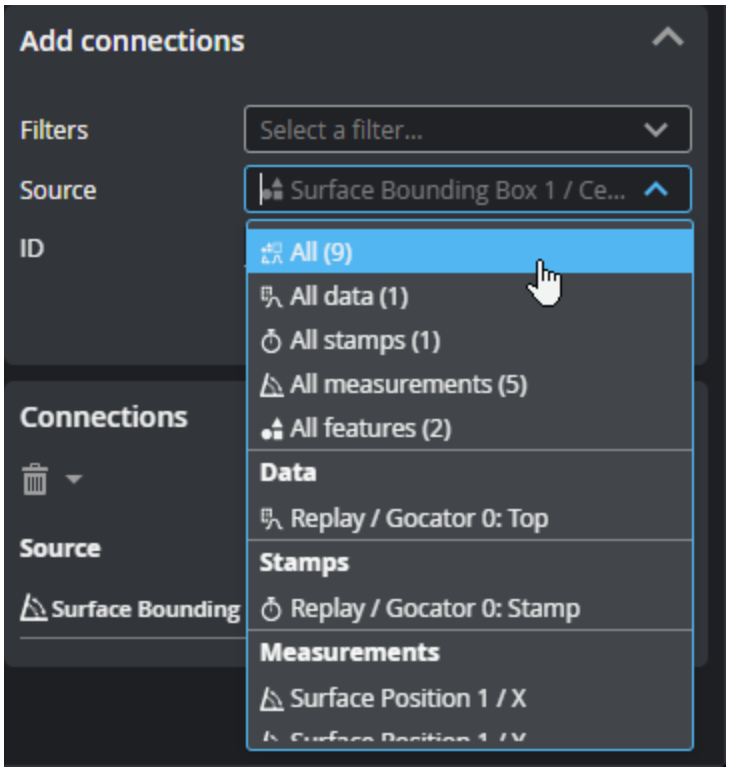
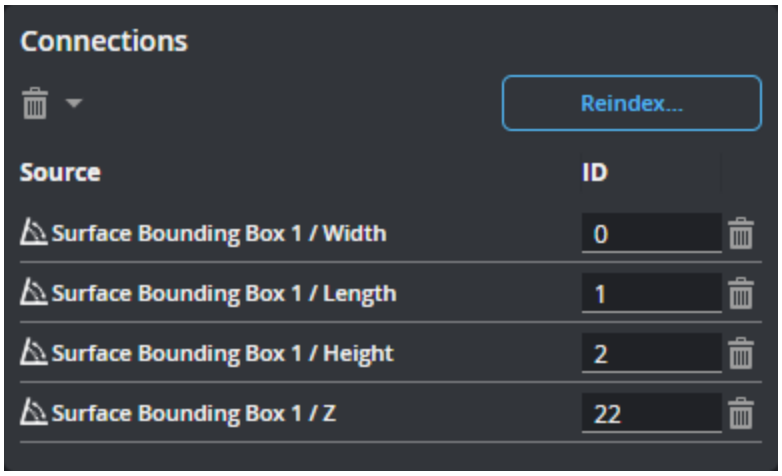
GoPxL Data Protocol

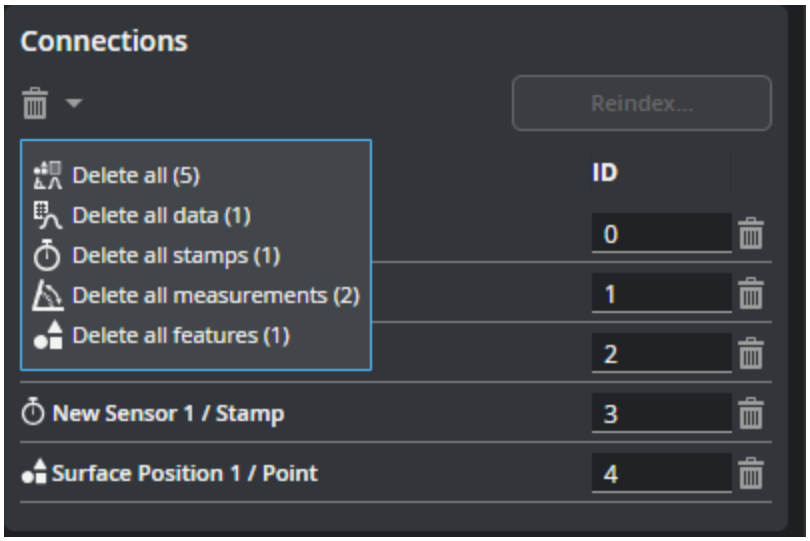
If you are using a GoMax Smart Vision Accelerator, only connect Ethernet-based PLCs or PCs to the ETH1 port on the GoMax device (whether via a switch or not).

The GoPxL Data Protocol sends TCP messages containing data and measurement results to client applications that use the GoPxL SDK. You can select which measurements and what type of scan data to send (range, profile, or surface depending on sensor model; image; or intensity). Note that GDP does not have to be enabled for the client application to read and write configurations, only to receive data and measurements.

	Element	Description
1	Protocol settings	Lets you disable or configure the protocol. For information on the settings available here, see *Configuration area parameters* on page 315.
2	Add Connections	Lets you add sources (data, stamp, measurement, or geometric feature) to send over the protocol. Added connections appear in the **Connections** section at the bottom of the panel If you have many sources, you can filter them by type to make it easier to find the source you want in the **Source** drop-down. You can enable

Element	Description
	more than one filter.  Choose a source in **Source**. If you leave ID as "Auto" GoPxL automatically increments the ID; otherwise, you can set it yourself. The field is hidden if more than one source is selected in **Source**.  You can also add all sources, or all sources of a given type. Adding multiple sources automatically increments the ID

	Element	Description
		 Click **Add** to add the source to the Connections table below.
3	Connections	The connections listed here will be sent by the GoPxL Data Protocol to an application developed using the GoPxL SDK. You can delete a connection using the trashcan icon in its row or change its ID. If you set an ID to the ID of another source, all other source automatically shift up to make room for it.  You can also use shortcut commands to delete all of the connections, or specific categories of connections.

Element	Description
	 If you set an ID to the index of another tool, all sources following the changed index are incremented. Click the **Reindex** button to reorder the connections and fill any gaps between index numbers.

Configuration area parameters

Parameter	Description
Enable	Toggles whether the Gocator communication protocol is enabled.
Automatic disconnect	When enabled, automatically disconnects from the client after the timeout value in **Timeout** if the sensor is unable to send data.
Timeout	Sets the timeout value when **Automatic disconnect** is enabled.

GoPxL Data Protocol

When the GoPxL Data Protocol is enabled in the GoPxL interface, you can use the GoPxL SDK and REST API to send commands from and receive data from SDK-based applications.

For information on configuring the Gocator protocol in GoPxL on the **Control** page, see *GoPxL Data Protocol* on page 312.

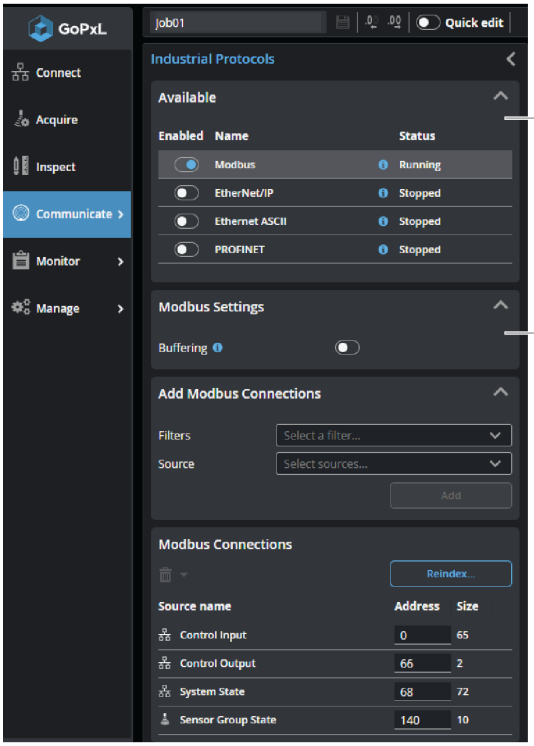
For information on the GoPxL SDK and the REST API, see *GoPxL SDK and REST API* on page 449.

Modbus TCP Protocol

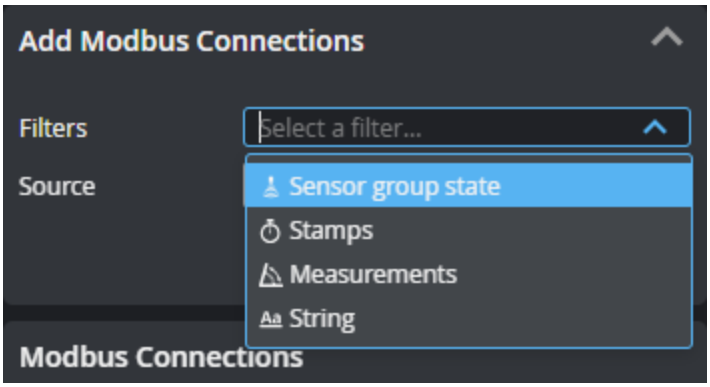
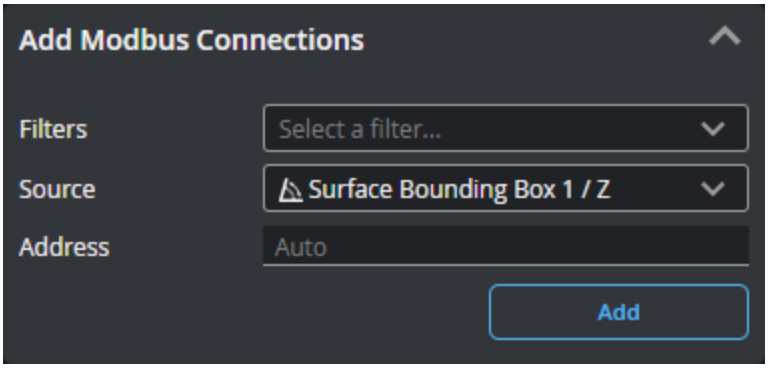
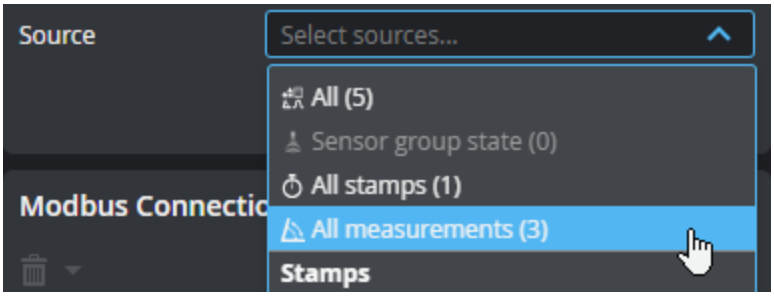
You can use a Modbus TCP programmable logic controller (PLC) over Ethernet to operate a sensor and receive values, such as measurements, stamps, or strings. Modbus only supports a subset of the tasks that can be accomplished in the web interface (for example, starting and stopping sensors, and switching jobs). Scan data can't be sent to the PLC.

If you are using a GoMax Smart Vision Accelerator, only connect Ethernet-based PLCs or PCs to the ETH1 port on the GoMax device (whether via a switch or not).

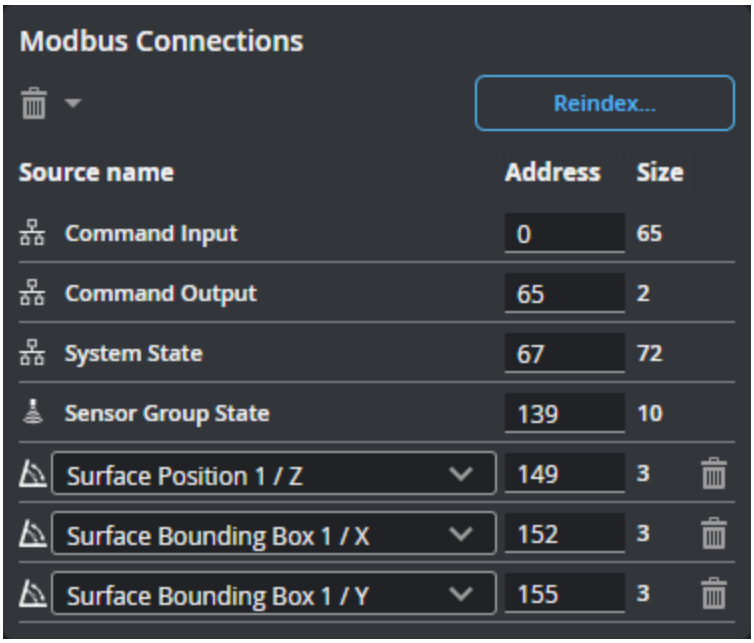
You configure the protocol on the **Communicate** > **Industrial** page. After configuring the Modbus protocol in GoPxL (see below), use the information in *Modbus TCP Protocol Format - PLC Information* on page 319 to configure your PLC.



	Element	Description
1	Available Protocols	Click a row in the table to choose which protocol to configure. Enable a protocol by toggling the slider to the right.
2	Modbus Settings	**Buffering** Toggle the slider to enable/disable buffering of scan outputs. If buffering is enabled, the PLC must send the 'Buffer Advance' command to advance the queue before reading the measurement results. For more information, see *Control Input* on page 321.
3	Add Modbus Connections	Lets you add sources (such as stamps, measurements, or strings) to send over the protocol. Added connections appear in the **Modbus Connections** section at the bottom of the panel. If you have many sources, you can filter them by type to make it easier to find the source you want in the **Source** drop-down . You can enable more than one filter.

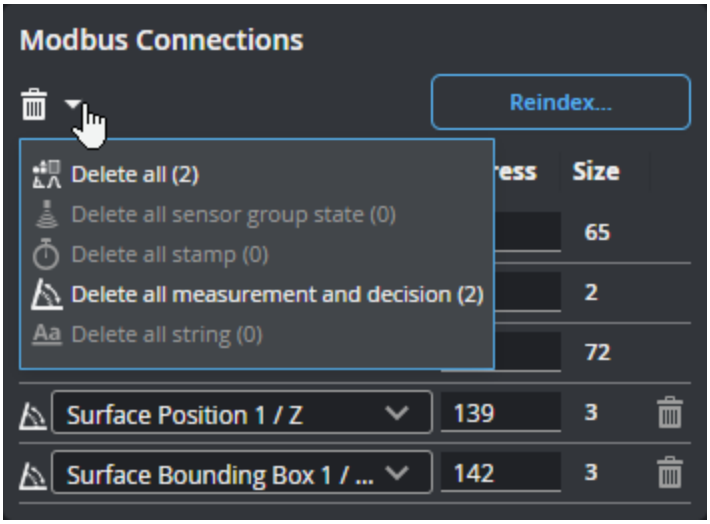
	Element	Description
		 Choose a source in the **Source** drop-down. If you leave **Address** as "Auto", GoPxL automatically increments the value; otherwise, you can set it yourself.  You can also add all sources, or all sources of a given type.  Click **Add** to add the source to the Modbus Connections table below.
4	Modbus Connections	The connections listed here will be sent by the protocol to a compatible PLC. You can delete a connection using the trashcan icon in its row, or change its ID. When changing an ID, make sure to use the size of the source to correctly calculate the minimum offset.

Element	Description



For string sources, you must specify a maximum expected length in the source's **Size** field.

You can also use shortcut commands to delete all of the connections, or specific categories of connections.



Use **Reindex...** to remove gaps of unused addresses between connections. You can also use **Reindex...** to set the value of the first index.

Modbus TCP Protocol Format - PLC Information

GoPxL supports the Modbus TCP communication to provide system, sensor group, and scan output data to a PLC, as well as accept incoming commands. After enabling Modbus in the GoPxL web interface (for more information, see *Modbus TCP Protocol* on page 315), the sensor starts listening on port 502 for one or more Modbus clients. GoPxL operates as a Modbus server (slave), and a PLC that connects to it acts as the Modbus client (master). All communication is initiated by the PLC, with GoPxL responding to the Modbus requests.

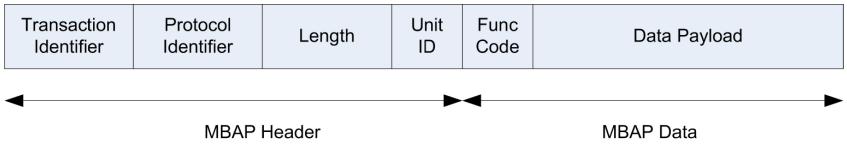
> The connection mappings for the industrial protocols (PROFINET, EtherNet/IP, and Modbus) have changed to correct a byte alignment issue. To allow communication between PLCs and GoPxL 1.2, you may need to update PLC logic created for GoPxL 1.1.
>
> For EtherNet/IP, the EDS file has been updated to reflect the changes.

While the Modbus protocol includes several types of requests, the two basic operations are reading and writing registers. Commands are invoked by the PLC sending write-register requests. Data is requested by the PLC by sending read-register requests.

There are 16-bits per register in the Modbus protocol. 32-bit and 64-bit values are stored in consecutive registers in Big-Endian format.

Messages

All Modbus TCP messages consist of an MBAP header (Modbus Application Protocol), a function code, and a data payload.



The MBAP header contains the following fields:

Modbus Application Protocol Header

Field	Length (Bytes)	Description
Transaction ID	2	Used for transaction pairing. The Modbus Client sets the value and the Server (the sensor) copies the value into its responses.
Protocol ID	2	Always set to 0.
Length	2	Byte count of the rest of the message, including the Unit identifier and data fields.
Unit ID	1	Used for intra-system routing purpose. The Modbus Client sets the value and the Server (the sensor) copies the value into its responses.

Modbus Application Protocol Specification describes the standard function codes in detail. Gocator supports the following function codes:

Modbus Function Code

Function Code	Name	Data Size (bits)	Description
3	Read Holding Registers	16	Read multiple data values from the sensor.
4	Read Input Registers	16	Read multiple data values from the sensor.
6	Write Single Register	16	Send a command or parameter to the sensor.
16	Write Multiple Registers	16	Send a command and parameters to the sensor.

The data payload contains the registers that can be accessed by Modbus TCP messages. If a message accesses registers that are invalid, a reply with an exception is returned. Modbus Application Protocol Specification defines the exceptions and describes the data payload format for each function code.

The sensor data includes 16-bit, 32-bit, and 64-bit data. All data are sent in big endian format, with the 32-bit and 64-bit data spread out into two and four consecutive registers.

32-bit Data Format

Register	Name	Bit Position
0	32-bit Word 1	31 .. 16
1	32-bit Word 0	15 .. 0

64-bit Data Format

Register	Name	Bit Position
0	64-bit Word 3	63 .. 48
1	64-bit Word 2	47 .. 32
2	64-bit Word 1	31 .. 16
3	64-bit Word 0	15 .. 0

Connections Map

The Modbus protocol supports various blocks types. Each block type can be either an input block or an output block. Input blocks contain write-only registers and are used by the Modbus client to send commands such as starting or stopping a scan. Output blocks contain read-only registers and are used to store various types information, such as control, system, and sensor group state data as well as scan output data.

The Modbus implementation uses the term "scanner" instead "sensor group." The two are interchangeable.

The starting address location of each block within the Modbus register map may be assigned using the GoPxL web interface.

The register address of a specific field within a block is equal to the block's address plus the field's address offset. For example, the command identifier in the Control Input block has an address offset

equal to 1. If the Control Input block is located at register address 100, the command identifier value is located at register address 101.

Supported Block Types

The block types supported by the Modbus protocol are listed in the table below. The size field indicates the number of registers that each block type requires.

Supported Block Types

Supported Block Type	Read or Write only?	Size (number of Modbus registers)	Description
"ControlInput"	WO	65	Command message set by the Modbus client.
"ControlOutput"	RO	2	Result data corresponding to a command that was written to the Control Input block.
"Measurement"	RO	3	Measurement data of a scan output measurement data message.
"Scanner"	RO	10	Information about a specific sensor group.
"Stamp"	RO	17	Stamp message information.
"String"	RO	4 (by default)	String data of a scan output string data message. Size can be configured through the GoPxL web interface.
"System"	RO	72	Information about the LMI device.

Control Input

The input of the Control Input block contains the fields described below:

Control Input Fields

Field	Type	Address offset	Description
Command Sequence Number	16u	0	The command sequence number is set by the external client to uniquely identify a command request. This number is sent back in the Control Output. This is an optional field but is recommended that Modbus clients use this to help the clients correlate the Command Status in the Control Output with a specific command request in Control Input.
Command Identifier	16u	1	Identifies the requested action from the external client. Supported actions: 0 = No Command 1 = Start Scanning 2 = Stop Scanning 3 = Align Sensor Group 4 = Clear Alignment 5 = Load Job File 6 = Software Trigger 7 = Buffer Advance

Field	Type	Address offset	Description
Command Arguments	16u	2	Arguments for commands that require arguments. For the Load Job File command, the command arguments will contain the *NULL terminated* job file name. The job file name must *not* include the job file extension. The job extension is automatically appended by the Modbus server. This field takes up a maximum of 63 registers.

Control Output

The output of the Control Output block contains the fields described below:

Control Output Fields

Field	Type	Address offset	Description
Command Sequence Number	16u	0	The command sequence number received from the client in the Control Input.
Command Status	16u	1	Status of the command, referenced by the Command Sequence Number. 0 = No Command Received 1 = In Progress 2 = Success 3 = Fail This field takes up 1 register.

System State

The output of the System State block contains the fields described below:

System State Fields

Field	Type	Address offset	Description
System Uptime	64u	0	Amount of time the application has been running, in seconds.
System State	16u	4	Whether system is running or not. 0 = System is stopped 1 = System is running
Current Job File Name Length	16u	5	Number of characters in the current job file name.
Current Job File Name	16u	6	Name of the job file currently loaded into the system, without the extension. Maximum length of job file name is 64 characters. Each register contains a single character. Job file name is *not* null terminated.
Buffer Count	16u	70	The current amount of buffered scan output data when buffering is enabled.

Field	Type	Address offset	Description
Buffer Overflow	16u	71	A value of 0 indicates that no overflow has occurred. A value of 1 indicates that overflow occurred and data is being lost. This field takes up 1 register.

Sensor Group State

The output of the Sensor Group State block contains the fields described below:

Sensor Group State Fields

Field	Type	Address offset	Description
Current Encoder Position	64s	0	The current encoder position.
Current Sync Time	64u	4	Current time of the clock used to stamp all scan output messages, in microseconds.
Alignment State	16u	8	Indicates whether all devices under the sensor group are aligned (that is, devices in the sensor group all have a non-identity transform matrix). 0 = Not Aligned 1 = Aligned 2 = Aligning
Laser Enabled	16u	9	Whether one or more devices under a scanner has its laser enabled to be able to turn on. 0 = All devices' lasers are disabled and cannot turn on 1 = One or more devices' lasers can be turned on This field takes up 1 register.

Stamp

The output of the Stamp block contains the fields described below:

Stamp Fields

Field	Type	Address offset	Description
Z Position	64s	0	Encoder position at time of last index pulse of last frame.
Encoder Position	64s	4	Encoder position of last frame when the image data was scanned/taken.
Time	64u	8	The timetick of the last frame. A timetick is a 64-bit positive integer that is guaranteed to increase monotonically starting from zero. It is guaranteed to be unique for every scan from a given sensor group. To convert this value to microseconds, use the following: `time in microseconds = timetick / 1.024`
Frame Index	64u	12	The frame number of the last frame.

Field	Type	Address offset	Description
			This field takes up 4 registers.
Digital Input States	16u	16	Digital input states of last frame. Bit 0: State of digital input 1 on master or sensor Bit 1: State of digital input 2 on master or sensor

Measurement

The output of the Measurement Block contains the fields described below:

Measurement Fields

Field	Type	Address offset	Description
Value	32s	0	Measurement value. This value is the actual measurement value multiplied by 1000. If the measurement value is invalid, the first register of the measurement value is set to 0x80000000. This means the first register of the measurement value is set to 0x8000 and the second register value is 0x0000.
Decision	16u	2	Measurement decision. Bit 0: 0 = Fail 1 = Pass Bit 1: 0 = Decision value is valid 1 = Decision value is invalid This field takes up 1 register.

String

The output of the String Block contains the fields described below:

Measurement Fields

Field	Type	Address offset	Description
String	string	0	String value. Unused space will be set to null (0x0). The string is otherwise not null terminated. If invalid, the block's buffer is entirely set to null, equivalent to an empty string. The string will be truncated if the block's size is smaller than the length of the string.

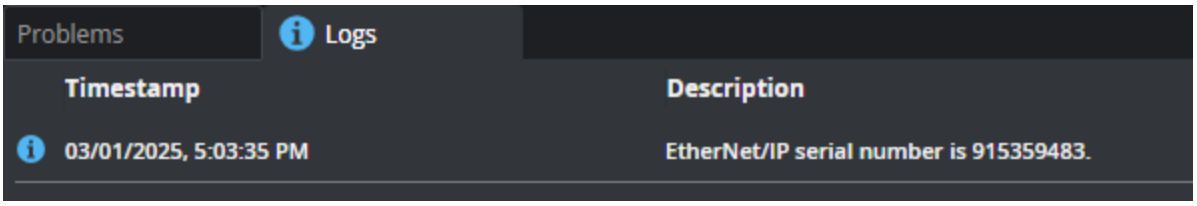
EtherNet/IP Protocol

You can use an EtherNet/IP programmable logic controller (PLC) over Ethernet to operate a sensor and receive values, such as measurements, stamps, or strings. EtherNet/IP only supports a subset of the tasks that can be accomplished in the web interface (for example, starting and stopping sensors, and switching jobs). Scan data can't be sent to the PLC.

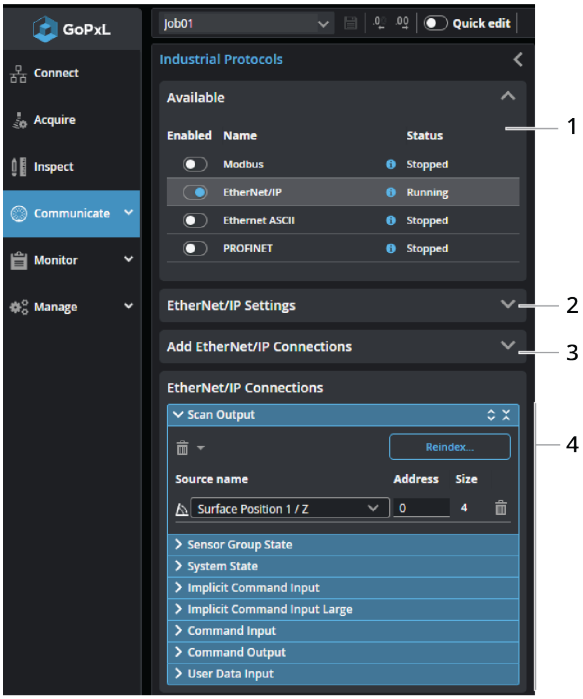
You can send data from a PLC to GoPxL after adding and configuring an instance of the User Data Input tool; for more information on the tool, see *User Data Input Tool* on page 308. To send data over EtherNet/IP to the tool, you must use a specific assembly for Explicit messaging (for information, see *Explicit Messaging User Data Input* on page 340) or a special command for Implicit messaging (command 9, see *Implicit Messaging Input* on page 341 or *Implicit Messaging Input Large* on page 342).

> If you are using a GoMax Smart Vision Accelerator, only connect Ethernet-based PLCs or PCs to the ETH1 port on the GoMax device (whether via a switch or not).

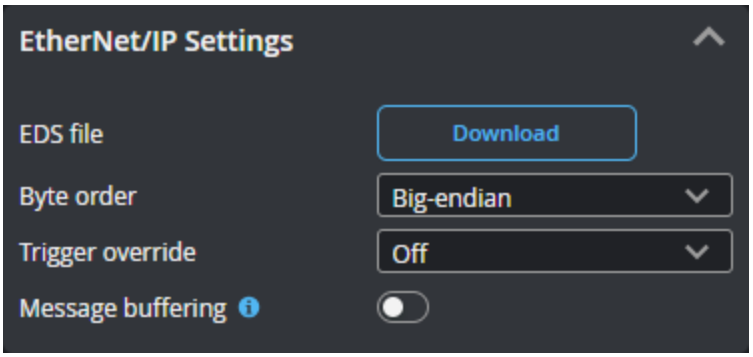
When you enable the protocol, a log entry displays the EtherNet/IP serial number you should use in your PLC.

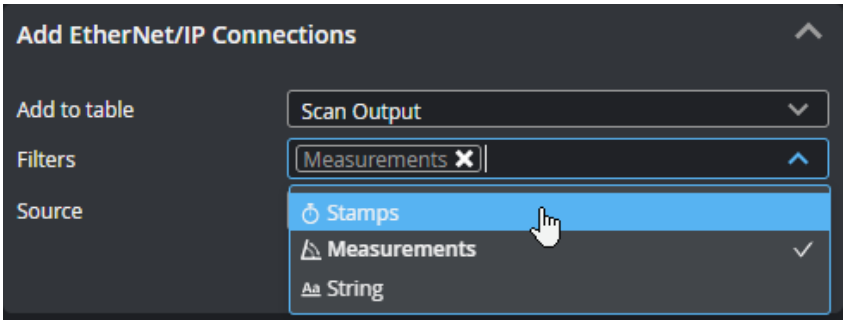


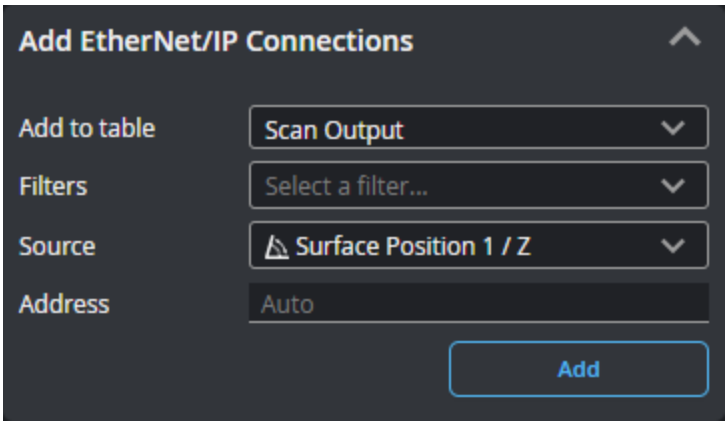
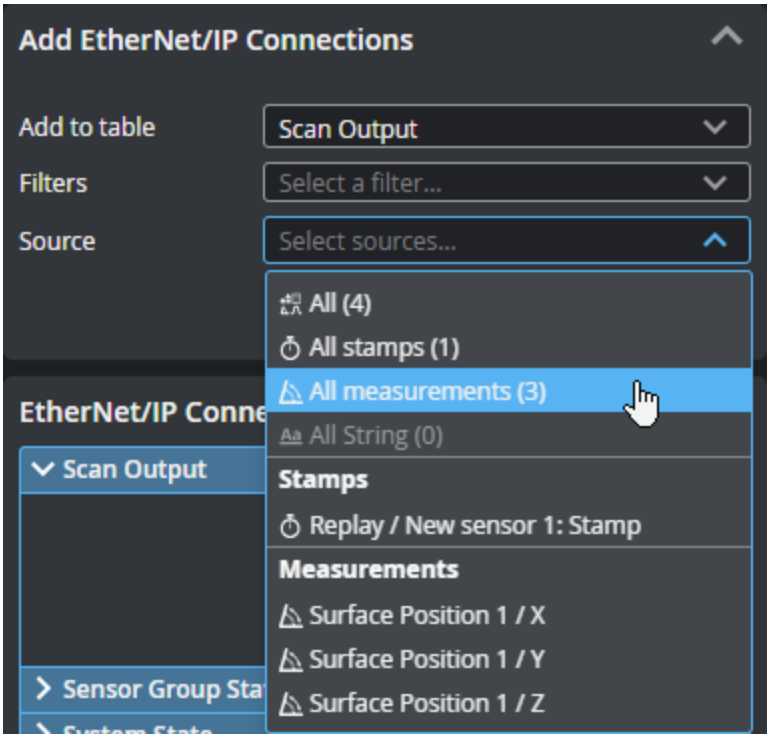
You configure the protocol on the **Communicate** > **Industrial** page. After configuring the EtherNet/IP protocol in GoPxL (see below), use the information in *EtherNet/IP Protocol Format - PLC Information* on page 331 to configure your PLC.



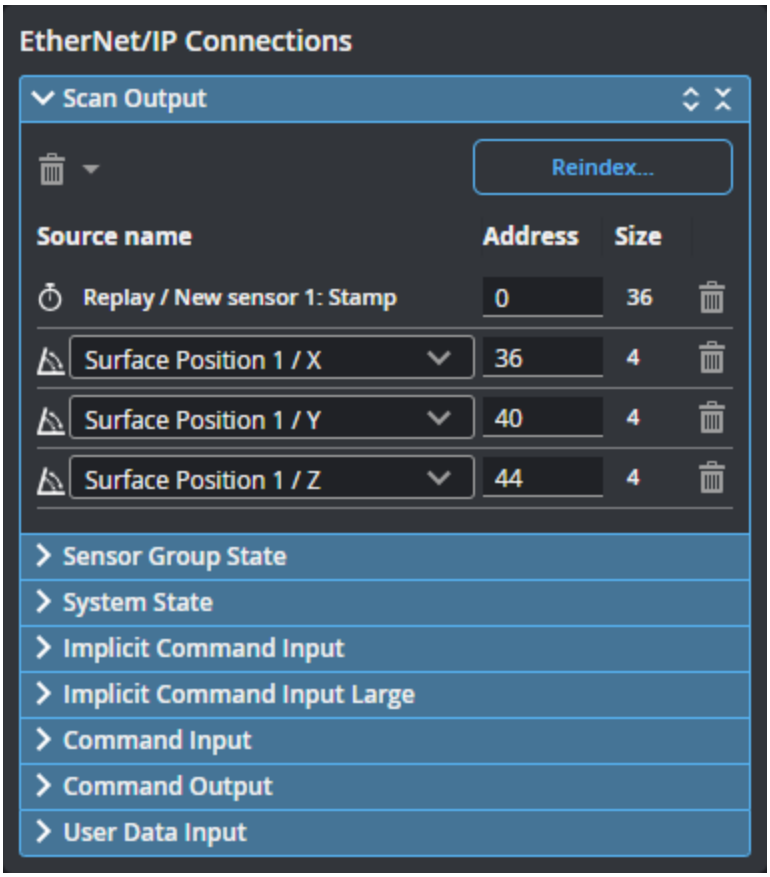
	Element	Description
1	Available Protocols	Click a row in the table to choose which protocol to configure. Enable a protocol by toggling the slider to the right.
2	EtherNet/IP Settings	You configure the protocol's settings in this section.

Element	Description
	 **EDS file** Click **Download** to save an EDS file to the client computer for later installation on the PLC. **Byte order** One of the following: **Big-endian**: The most significant byte (the "big end") of the data is placed at the byte with the lowest address. The rest of the data is placed in order of decreasing significance in the next three bytes of memory. **Little-endian** (default): The least significant byte (the "little end") of the data is placed at the byte with the lowest address. The rest of the data is placed in order of increasing significance in the next three bytes in memory. Most Allen-Bradley PLCs default to Little Endian addressing formats, but you should verify this when configuring the PLC. **Trigger override** One of the following: **Off**, **Cyclic**, and **Change of State**. When you set up the PLC to communicate with a Gocator using change of state implicit messaging, an event task must be created on the PLC to rapidly check whether the sensor is running; if the frame count increases, data is copied to an array. The event task period must allow the event task to be executed at a higher rate than Gocator frame rate. **Message buffering** Used to enable and disable buffering of scan outputs. You should enable buffering if you are using part detection and if multiple objects may be detected within a period shorter than the polling

	Element	Description
		rate of the PLC. If buffering is enabled with the EtherNet/IP protocol, the buffer is automatically advanced when the Sample State assembly object is read (see *EtherNet/IP Protocol Format - PLC Information* on page 331). Message buffering only applies to tool outputs sent over explicit messaging.
3	Add EtherNet/IP Connections	Lets you add sources (such as stamps, measurements, or strings) to send over the protocol. Added connections appear in the **EtherNet/IP Connections** section at the bottom of the panel. When GoPxL is running on a standalone sensor (one not controlling others), **Add to Table** only contains one value (Scan Output) that represents a register assembly. When GoPxL is running on a sensor controlling other sensors, or on a PC or GoMax NX, **Add to Table** contains an additional entry (Sensor Group State). Use this to choose which register a source is added to. If you have many sources, you can filter them by type to make it easier to find the source you want in the **Source** drop-down . You can enable more than one filter.  Choose the source in the **Source** drop-down. If you leave **Address** as "Auto", GoPxL automatically increments the value; otherwise, you can set it yourself. By default, connections are added directly after the last source in the connections map. You can optionally set the address at which a source is added using the **Address** parameter.

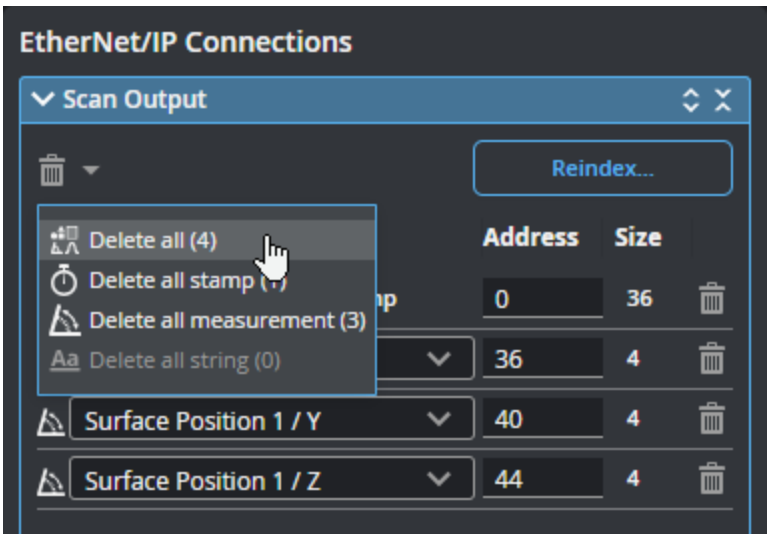
	Element	Description
		 You can also add all sources, or all sources of a given type.  Click **Add** to add the source to the EtherNet/IP Connections table below.
4	EtherNet/IP Connections	The connections listed here will be sent by the protocol to a compatible PLC. You can delete a connection using the trashcan icon in its row, or change its ID. When changing an ID, make sure to use the size of the source to correctly calculate the minimum offset.

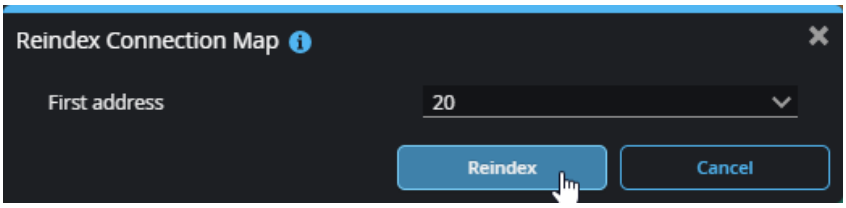
Element	Description



For string sources, you must specify a maximum expected length in the source's **Size** field.

You can also use shortcut commands to delete all of the connections, or specific categories of connections.



Element	Description
	Use **Reindex...** to remove gaps of unused addresses between connections. You can also set the value of the first index. 

EtherNet/IP Protocol Format - PLC Information

EtherNet/IP is an industrial protocol that allows bidirectional data transfer with PLCs that support the protocol. It encapsulates the object-oriented Common Industrial Protocol (CIP). EtherNet/IP communication enables the client to:

- Start and stop sensors
- Load job files
- Align sensors
- Receive sensor states, stamps, strings, and measurement results
- Retrieve data sent from a PLC to the User Data Input tool; for more information on this tool, see *User Data Input Tool* on page 308

This section describes the EtherNet/IP message and data formats. The commands described in the sections below are those specific to the Gocator protocol and not the complete EtherNet/IP reference command set.

Before configuring your PLC, make sure to enable and configure EtherNet/IP in GoPxL. For more information on this, see *EtherNet/IP Protocol* on page 324.

Implicit messaging is always enabled.

GoPxL supports EtherNet/IP communication to provide system state, sensor group state, and scan output data to a PLC, and also receive commands. After you enable EtherNet/IP in the GoPxL web interface, the sensor starts listening on the standard EtherNet/IP port 44818 for one or more EtherNet/IP clients. GoPxL operates as an EtherNet/IP server, and a PLC that connects to it acts as the EtherNet/IP client. All communication is initiated by the PLC, with GoPxL responding to the EtherNet/IP requests.

While the EtherNet/IP protocol includes several types of requests, the two core operations are reading and writing EtherNet/IP objects. Commands are invoked by the PLC sending *set* requests. Data is requested by the PLC by sending *get* requests for EtherNet/IP objects.

EtherNet/IP supports both explicit messaging and implicit messaging. Data can be read in Big-Endian or Little-Endian format based on the protocol's configuration selected in the web interface.

Supported EtherNet/IP Objects

All mandatory EtherNet/IP objects required for explicit messaging, including Assembly objects, are supported by the sensors. Except for mandatory objects (see below), the remaining objects are mapped to register assemblies.

Mandatory objects (*Identity Object (Class 0x01)* on page 334, *TCP/IP Object (Class 0xF5)* on page 334, and *Ethernet Link Object (Class 0xF6)* on page 335 objects) are always present in the EtherNet/IP service, regardless of the configured measurement object.

Implicit messaging supports only Assembly objects.

Object and Register Assembly Mapping

Object and Register Assembly Mapping

Object name	Register assembly	Supported Block Type Internal Identifier[1]	Description	Service	Messaging Type	Instance ID	Max Size (bytes)
Command input	Control Input	ControlInput	Receives commands from the PLC.	Explicit	SetAttributeSingle (0x10)	112 (0x70)	67
Command output	Control Output	ControlOutput	Sends command response or feedback to the PLC.	Explicit	GetAttributeSingle (0x0E)	113 (0x71)	3
Sensor Group State	Sensor Group State	Scanner	Provides status of the sensor group.	Explicit	GetAttributeSingle (0x0E)	114 (0x72)	18
System State	System State	System	Reports overall system health and connection info.	Explicit	GetAttributeSingle (0x0E)	115 (0x73)	76
Sample State	Sample State	MeasurementValue String Stamp	Returns the latest measurement or string results and stamp.	Explicit	GetAttributeSingle (0x0E)	116 (0x74)	375
User Data Input	User Data Input	UserDataInput	Allows PLC users to exchange a section of memory buffer with the Gocator device.	Explicit	SetAttributeSingle (0x10)	117 (0x75)	468
Implicit Messaging Input	Implicit Control Input	ControlInput	Receives control commands via cyclic connection.	Implicit	SetAttributeSingle (0x10)	100 (0x64)	67
Implicit Messaging Output	Sample State	MeasurementValue String Stamp	Sends measurement or string results and	Implicit	GetAttributeSingle (0x0E)	101 (0x65)	376

Object name	Register assembly	Supported Block Type Internal Identifier[1]	Description	Service	Messaging Type	Instance ID	Max Size (bytes)
			stamps to the PLC on cycle.				
Implicit Messaging Input Large	Implicit Control Input	ControlInputLarge	Receives control commands via cyclic connection. Provides additional space for use with the User Data Input tool.	Implicit	SetAttributeSingle (0x10)	102 (0x66)	496
Config Assembly	N/A	N/A	Configuration Assembly for Implicit I/O Connection Heartbeat	Implicit	N/A	151 (0x97)	0

[1] For more information on the supported block types, see *Supported Block Types* below.

Supported Block Types

The block types supported by the EtherNet/IP protocol are listed in the table below. The size field indicates the number of registers that each block type requires.

Supported Block Types

Block Type Internal Identifier	Read or Write only?	Size (number of EtherNet/IP registers)	Description
ControlInput	Write-Only	67	Accepts control commands from the EtherNet/IP client.
ControlOutput	Read-Only	3	The result of the last command written to the ControlInput block.
MeasurementValue	Read-Only	4	Measurement values.
String	Read-Only	8 (by default)	String values. Size can be configured through the GoPxL web interface.
Scanner	Read-Only	18	The current state and configuration of the sensor group.
Stamp	Read-Only	36	Stamp message information.
System	Read-Only	76	Overall device state and configuration of the GoPxL system.

Block Type Internal Identifier	Read or Write only?	Size (number of EtherNet/IP registers)	Description
ControlInputLarge	Write-Only	496	Accepts control commands and user data from the EtherNet/IP client. For information on the User Data Input tool, see *User Data Input Tool* on page 308.
UserDataInput	Write-Only	468	Accepts user data from the EtherNet/IP client. For information on the User Data Input tool, see *User Data Input Tool* on page 308.

Identity Object (Class 0x01)

The Identity Object (Class 0x01, Instance ID 0x01) provides read-only device identification attributes used by EtherNet/IP clients during explicit messaging.

Identity object (Class 0X01)

Attribute ID	Attribute Name	Data Type	Value	Description	Access
1	Vendor ID	UINT	1256	ODVA-assigned vendor ID for LMI	Get
2	Device Type	UINT	43	Device type	Get
3	Product Code	UINT	1	Product code	Get
4	Revision	Structure (USINT, USINT)	Major: 2, Minor: 1	Firmware revision	Get
6	Serial number	UDINT	[device-specific]	Sensor serial number	Get
7	Product Name	SHORT_STRING[32]	Gocator	Gocator product name	Get

TCP/IP Object (Class 0xF5)

The TCP/IP Object (Instance ID 0x01) provides read-only attributes related to the network configuration of the device. This object is used by EtherNet/IP clients to retrieve information such as the current IP address and interface status.

> GoPxL does not support writing or modifying TCP/IP settings via EtherNet/IP. Network parameters must be configured through the GoPxL web interface or external methods such as DHCP.

For complete structural definitions, see CIP Specification Volume 2, Chapter 5-3.

Attribute ID	Attibute Name	Data Type	Value	Description	Access
1	Status	UDINT	0	TCP interface status	Get
2	Configuration Capability	UINT	0	Indicates which configuration methods (e.g., BOOTP, DHCP) are supported.	Get
3	Configuration Control	UINT	0	Indicates which method is currently active.	Get

Attribute ID	Attibute Name	Data Type	Value	Description	Access
4	Physical Link Object	Structure	-	Contains: Path Size (UINT), Path (Padded EPATH) See §5.3.3.2.4 CIP V2.	Get
5	Interface Configuration	Structure	-	Contains: IP Address, Subnet Mask, Gateway, DNS, Domain See §5.3.3.2.5.	Get

Ethernet Link Object (Class 0xF6)

The Ethernet Link Object (Instance ID 0x01) exposes read-only attributes related to the physical Ethernet interface, such as link speed, status, and MAC address. These attributes provide diagnostic and identification information for EtherNet/IP clients.

For full attribute definitions and behavior, see CIP Specification Volume 2, Chapter 5-4.

Attribute ID	Attribute Name	Data Type	Value	Description	Access
1	Interface Speed	UDINT	1000	Data rate of the Ethernet interface, in Mbps.	Get
2	Interface Flags	UDINT	[bitfield]	Bit-level status flags: Bit 0 – Link Status (0 = Inactive, 1 = Active) Bit 1 – Duplex (0 = Half, 1 = Full)	Get
3	Physical Address	Array[6] OF USINT	e.g., 00 16 20 00 2E 42	Device MAC address.	Get

Assembly Object (Class 0x04)

All EtherNet/IP assembly objects in GoPxL are implemented using Class `0x04` and Attribute `0x03`, with each object distinguished by a unique Instance ID. Each instance represents a specific data exchange structure, either for input (write-only) or output (read-only).

A register assembly in GoPxL is a container that holds one or more register blocks. There is a strict one-to-one mapping between EtherNet/IP assembly objects and GoPxL register assemblies. This mapping allows PLCs to interact with specific control, system, or measurement data structures through EtherNet/IP.

Assembly objects are always exposed in the GoPxL EtherNet/IP interface, whether used in explicit or implicit messaging. However, data will only be populated if the corresponding register assembly is actively selected and configured in the GoPxL web interface. If the register assembly is not enabled, the object remains accessible but contains no meaningful data.

For Explicit Messaging, each assembly is read or written by the PLC using `GetAttributeSingle` or `SetAttributeSingle` services on demand.

For Implicit Messaging, each assembly is mapped to a cyclic connection. Once a connection is established by the PLC, data is exchanged automatically at the configured RPI (Requested Packet Interval).

Explicit Messaging Command Input

The Command Input block is used by an EtherNet/IP client to send control instructions to GoPxL. It maps to the `ControlInput` register assembly and uses Class `0x04`, Attribute `0x03`, Instance ID 112 (`0x70`).

This assembly supports the `SetAttributeSingle (0x10)` service and has a maximum size of 67 bytes.

Information	Value
Register Assembly	Control Input register assembly
Supported Block Type Identifier	ControlInput
Class	0x4
Instance	112 (0x70)
Attribute Number	3
Max Size	67 bytes

Input Structure

The input data block consists of the following fields:

Attribute	Name	Type	Value
Command Sequence Number	UINT16	0	Optional. A unique value set by the client to identify each command. Echoed back in the Control Output.
Command Identifier	UINT8	2	Identifies the action to perform: 0 = No Command 1 = Start Scanning 2 = Stop Scanning 3 = Align Sensor Group 4 = Clear Alignment 5 = Load Job File 6 = Software Trigger 9 = Set User Data Input
Unused	UINT8	3	Reserved unused field.
Command Arguments	ARRAY [UINT8] (max 63)	4	Optional. **Load Job File** The argument must be a null-terminated string containing the job filename. Do not include the file extension, as the control service server appends it automatically. Each register holds a single character, with a limit of 63 characters including the null terminator.

Attribute	Name	Type	Value
			Set User Data Input Use up to the full 63 registers for data input.

Explicit Messaging Command Output

The Command Output block provides the result of the last command sent to the device via the Control Input block. It maps to the `ControlOutput` register assembly and uses Class `0x04`, Attribute `0x03`, Instance ID 113 (`0x71`).

This assembly supports the `SetAttributeSingle (0x0E)` service and has a maximum size of 3 bytes.

Information	Value
Register Assembly	Control Output register assembly
Supported Block Type Identifier	ControlInput
Class	0x4
Instance	113 (0x71)
Attribute Number	3
Max Size of Register Assembly	3 bytes

Output Structure

The fields below are fixed in size and structure:

Attribute	Name	Type	Value
Command Sequence Number	UINT16	0	The sequence number echoed from the most recent Control Input.
Command Status	UINT8	2	Status of the referenced command: 0 = No Command Received 1 = In Progress 2 = Success 3 = Failure

Explicit Messaging Sensor Group State

The Sensor Group State block provides real-time status data about the active sensor group (scanner). It maps to the `Scanner` register assembly and uses Class `0x04`, Attribute `0x03`, Instance ID 114 (`0x72`).

This assembly supports the `GetAttributeSingle (0x0E)` service and has a maximum size of 18 bytes.

Information	Value
Register Assembly	Sensor Group State register assembly
Supported Block Type Identifier	Scanner

Information	Value
Class	0x4
Instance	114 (0x72)
Attribute Number	3
Max Size	18 bytes

Output Structure

The output of the Sensor Group Block contains the following fields:

Attribute	Name	Type	Value
Current Encoder Position	INT64	0	The current position from the encoder, used to synchronize with motion systems.
Current Sync Time	UINT64	8	Current system time used to timestamp scan outputs, in microseconds.
Alignment State	UINT8	16	Indicates sensor group alignment: 0 = Not Aligned 1 = Aligned 2 = Aligning
Laser Enabled	UINT8	17	Indicates laser readiness: 0 = All lasers disabled 1 = One or more lasers enabled

Explicit Messaging System State

The System State block reports the current runtime status of the GoPxL system. It maps to the `System State` register assembly and uses Class `0x04`, Attribute `0x03`, Instance ID 115 (`0x73`).

This assembly supports the `GetAttributeSingle (0x0E)` service and has a maximum size of 76 bytes.

Information	Value
Register Assembly	System State register assembly
Supported Block Type Identifier	System
Class	0x4
Instance	115 (0x73)
Attribute Number	3
Max Size	76 bytes
Supported Service	GetAttributeSingle (0x0E)

Output Structure

This block contains system-level runtime information and file state useful for diagnostics and monitoring:

Field	Type	Offset (bytes)	Description
System Uptime	UINT64	0	Number of seconds the GoPxL application has been running. On sensor or GoMax, this is the amount of time the GoPxL software on the device has been running. This will be close to the amount of time the sensor or GoMax has been powered on. On PC, this is the amount of time the GoPxLService instance has been running on the PC. This can be very different from the amount of time the PC has been powered on.
System State	UINT8	8	Indicates if the system is actively running: 0 = Stopped 1 = Running
Buffer Count	UINT8	9	Number of buffered scan output messages (when buffering is enabled).
Buffer Overflow	UINT8	10	0 = No overflow 1 = Overflow has occurred, indicating data is being lost.
Job File Name Length	UINT8	11	Number of characters in the loaded job file name.
Job File Name	ARRAY [UINT8] (max 64)	12	Name of the loaded job file, without the extension. Not null-terminated. Each register stores one character.

Explicit Messaging Sample State

The Sample State block reports metadata and measurement data related to the most recent scan frame. It maps to the `Stamp`, `MeasurementValue`, and `String` register assemblies and uses Class `0x04`, Attribute `0x03`, Instance ID 116 (`0x74`).

This assembly supports the `GetAttributeSingle (0x0E)` service and has a maximum size of 375 bytes.

Information	Value
Register Assembly	Sample State register assembly
Supported Block Type Identifier	Stamp, MeasurementValue, String
Class	0x4
Instance	116 (0x74)
Attribute Number	3
Max Size of Register Assembly	375 bytes
Supported Service	GetAttributeSingle (0x0E)

Stamp Block

The Stamp block provides frame-level metadata useful for alignment, timing, and synchronization across devices. Values reflect the most recent frame acquired by the system.

Field	Type	Offset (bytes)	Value
Z Position	INT64	0	Encoder position at time of last index pulse of the last frame.
Encoder Position	INT64	8	Encoder position at time of scan capture.
Time	UINT64	16	Unique, monotonically increasing timetick of the last frame. `Time (µs) = Timetick / 1.024`
Frame Index	UINT64	24	Frame number of the last scan.
Digital Input States	UINT8	32	Bitfield of digital inputs:Bit 0 = Digital Input 1Bit 1 = Digital Input 2
Buffer (reserved)	UINT8[3]	33-35	Reserved. Unused padding bytes.

Measurement Value Block

The MeasurementValue block returns the actual scan measurement value. GoPxL 1.2 does not support measurement + decision blocks—only measurement is reported.

Field	Type	Offset (bytes)	Value
[measurement value]	INT32	0	Measurement value ×1000. For example, 12.345 mm is sent as 12345. If the value is invalid, this field is set to 0x80000000.

String Block

The String block returns the string scan output.

Field	Type	Offset (bytes)	Value
[string value]	STRING	0	String value. Unused space will be set to null (0x0). The string is otherwise not null terminated. If invalid, the block's buffer is entirely set to null, equivalent to an empty string. The string will be truncated if the block's size is smaller than the length of the string. Block size is 8 bytes by default and can be configured in the GoPxL UI.

Explicit Messaging User Data Input

The User Data Input block lets a PLC exchange a section of memory buffer with the Gocator device. It uses Class `0x04`, Attribute `0x03`, and Instance ID 117 (`0x75`). This assembly supports the `GetAttributeSingle (0x0E)` service and has a maximum size of 468 bytes.

Only one instance of `UserDataInput` can be created and it is automatically created by the system.

Information	Value
Register Assembly	User Data Input register assembly

Information	Value
Supported Block Type Identifier	UserDataInput
Class	0x4
Instance	117 (0x75)
Attribute Number	3
Max Size of Register Assembly	468 bytes
Supported Service	SetAttributeSingle (0x10)

User Data Input Block

The `UserDataInput` block represents a data buffer of 468 bytes. It allows a PLC user to define and use any sections within this buffer and exchange that with GoPxL.

Field	Type	Offset (bytes)	Value
[Data]	Any	0	A memory region that lets the user define and use it.

Implicit Messaging Input

The Implicit Messaging Input object receives control commands from a PLC. It maps to the `ControlInput` register assembly and is assigned Instance ID 100 (`0x64`)

Information	Valu
Register Assembly	Implicit Control Input register assembly
Supported Block Type Identifier	ControlInput
Class	0x4
Instance	100 (0x64)
Attribute Number	3
Max Size of Register Assembly	67 bytes
Supported Service	SetAttributeSingle (0x10)

Input Structure

The input data block consists of the following fields.

Field	Type	Offset (bytes)	Value
Command Sequence Number	UINT16	0	Optional. A unique value set by the client to identify each command. Echoed back in the Control Output.
Command Identifier	UINT8	2	Identifies the action to perform: 0 = No Command 1 = Start Scanning 2 = Stop Scanning 3 = Align Sensor Group

Field	Type	Offset (bytes)	Value
			4 = Clear Alignment 5 = Load Job File 6 = Software Trigger 9 = Set User Data Input. For use with the User Data Input tool; for more information see *User Data Input Tool* on page 308.
Unused	UINT8	3	Reserved unused field.
Command Arguments	ARRAY [UINT8] (max 63)	4	Optional. **Load Job File** The argument must be a null-terminated string containing the job filename. Do not include the file extension, as the control service server appends it automatically. Each register holds a single character, with a limit of 63 characters including the null terminator. **Set User Data Input** Use up to the full 63 registers for data input.

Implicit Messaging Input Large

The Implicit Messaging Input Large object receives control commands from a PLC. It maps to the `ControlInputLarge` register assembly and is assigned Instance ID 102 (`0x66`)

The structures of the Implicit Messaging Input Large and Implicit Messaging Input block. The main differences are as follows:

The Implicit Messaging Input block allows up to 63 bytes to be used for a file name with the Load Job File command and user data for the Set User Input Data command.

The Implicit Messaging Input Large block allows up to 63 bytes for a file name with the Load Job File command, or up to 492 bytes for user data with the Set User Input Data command.

Information	Valu
Register Assembly	Implicit Control Input register assembly
Supported Block Type Identifier	ControlInput
Class	0x4
Instance	102 (0x66)
Attribute Number	3
Max Size of Register Assembly	496 bytes
Supported Service	SetAttributeSingle (0x10)

Input Structure

The input data block consists of the following fields.

Field	Type	Offset (bytes)	Value
Command Sequence Number	UINT16	0	Optional. A unique value set by the client to identify each command. Echoed back in the Control Output.
Command Identifier	UINT8	2	Identifies the action to perform: 0 = No Command 1 = Start Scanning 2 = Stop Scanning 3 = Align Sensor Group 4 = Clear Alignment 5 = Load Job File 6 = Software Trigger 9 = Set User Data Input. For use with the User Data Input tool; for more information see *User Data Input Tool* on page 308.
Unused	UINT8	3	Reserved unused field.
Command Arguments	ARRAY [UINT8] (max 492)	4	Optional. **Load Job File** The argument must be a null-terminated string containing the job filename. Do not include the file extension, as the control service server appends it automatically. Each register holds a single character, with a limit of 63 characters including the null terminator. **Set User Data Input** Use up to the full 492 registers for data input.

Implicit Messaging Output

The Implicit Messaging Output object streams live scan data from GoPxL to the PLC. It maps to the `Sample State` register assembly, which includes Stamp, MeasurementValue, and String blocks `0x04`, Attribute `0x03`, Instance ID 101 (`0x65`).

The structures of the Implicit Messaging Input Large and Implicit Messaging Input block. The main differences are as follows:

The Implicit Messaging Input block allows up to 63 bytes to be used for a file name with the Load Job File command and user data for the Set User Input Data command.

The Implicit Messaging Input Large block allows up to 63 bytes for a file name with the Load Job File command, or up to 492 bytes for user data with the Set User Input Data command.

Information	Valu
Register Assembly	Sample State register assembly
Supported Block Type Identifier	Stamp, MeasurementValue, String
Class	0x4
Instance	101 (0x65)
Attribute Number	3
Max Size of Register Assembly	376 bytes

Stamp Block

The Stamp block provides frame-level metadata useful for alignment, timing, and synchronization across devices. Values reflect the most recent frame acquired by the system

Field	Type	Offset (bytes)	Value
Z Position	INT64	0	Encoder position at time of last index pulse of the last frame.
Encoder Position	INT64	8	Encoder position at time of scan capture.
Time	UINT64	16	Unique, monotonically increasing timetick of the last frame. `Time (µs) = Timetick / 1.024`
Frame Index	UINT64	24	Frame number of the last scan.
Digital Input States	UINT8	32	Bitfield of digital inputs:Bit 0 = Digital Input 1Bit 1 = Digital Input 2
Buffer (reserved)	UINT8[3]	33-35	Reserved. Unused padding bytes.

Measurement Value Block

The MeasurementValue block returns the actual scan measurement value.

Field	Type	Offset (bytes)	Value
[measurement value]	INT32	0	Measurement value ×1000. For example, 12.345 mm is sent as 12345. If the value is invalid, this field is set to 0x80000000.

String Block

The Stringblock returns the string scan output.

Field	Type	Offset (bytes)	Value
[string value]	STRING	0	String value. Unused space will be set to null (0x0). The string is otherwise not null terminated. If invalid, the block's buffer is entirely set to null, equivalent to an empty string. The string will be truncated if the block's size is smaller than the length of the string. Block size is 8 bytes by default and can be configured in the GoPxL UI.

Ethernet ASCII Protocol

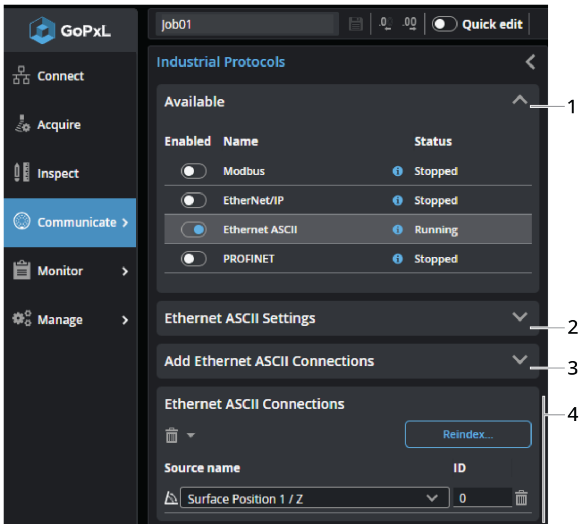
You can use a Programmable Logic Controller (PLC) that supports the Ethernet ASCII protocol to operate a sensor and receive values, such as measurements, stamps, or strings, and decisions (together called results).

Two modes are available: asynchronous and polling.

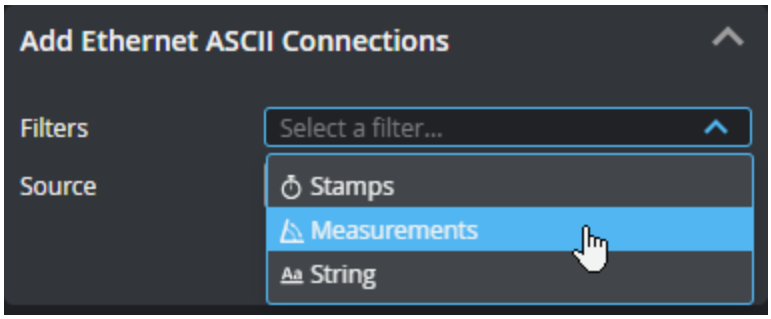
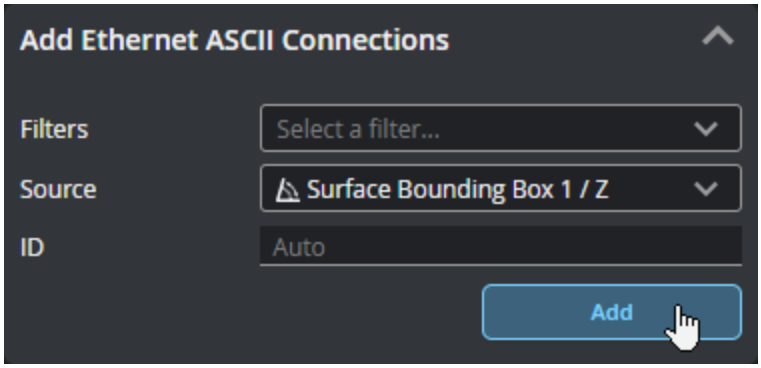
When the sensor is in asynchronous mode, results are sent automatically, as soon as they are available. Data is still available synchronously (that is polled) using the supported polling commands.

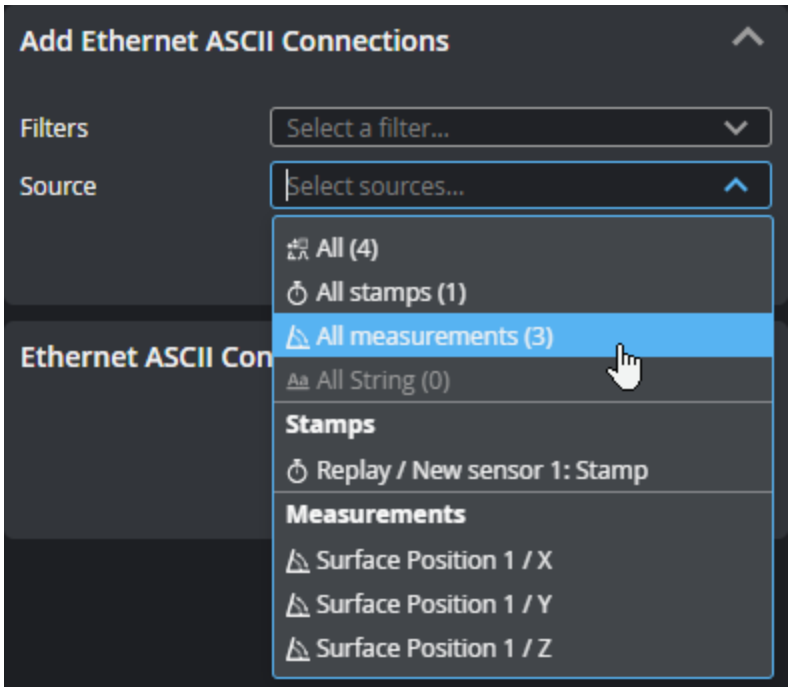
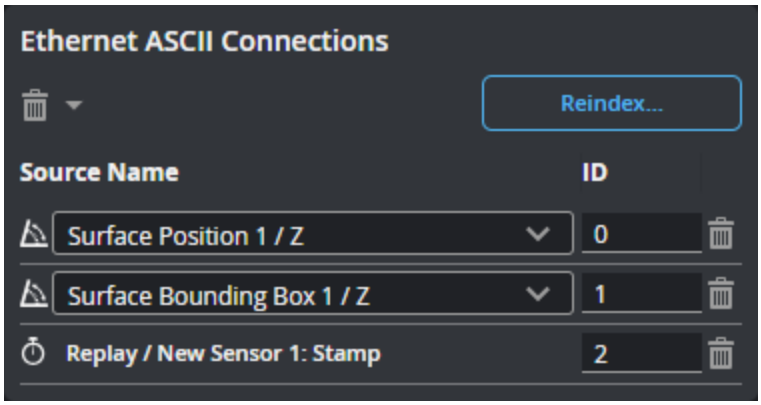
When the sensor is in polling mode (asynchronous mode is disabled), you must use commands on the PLC to operate the sensor and to retrieve results. Ethernet ASCII protocol in polling mode only supports a subset of the tasks that can be accomplished in the web interface, and only measurement results can be transmitted to the PLC (scan data is not sent).

You configure the protocol on the **Communicate** > **Industrial** page. After configuring the ASCII protocol parameters in GoPxL (see below), see *Ethernet ASCII Protocol Format - PLC Information* on page 350 for information on the available polling commands.

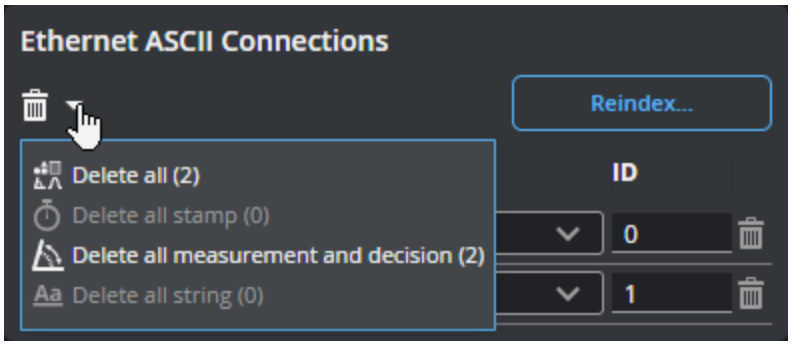


	Element	Description
1	Available protocols	Click a row in the table to choose which protocol to configure. Enable a protocol by toggling the slider to the right.
2	Ethernet ASCII Settings	Lets you configure the Ethernet ASCII protocol. For descriptions of the settings, see *Ethernet ASCII settings* on page 349.
3	Add Ethernet ASCII Connections	Lets you add sources (such as stamps or measurements) in the connection map. Added connections appear in the **Ethernet ASCII Connections** at the bottom of the panel. If you have many sources, you can filter them by type to make it easier to find the source you want in the **Source** drop-down. You can select more than one filter at once.

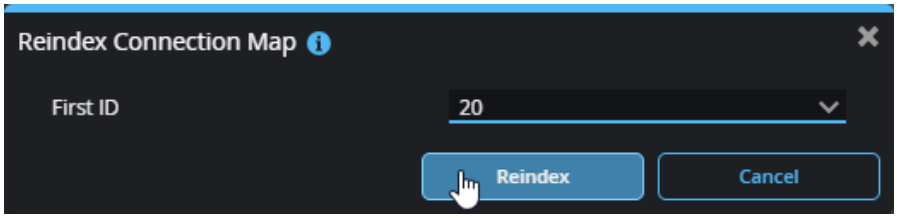
Element	Description
	 Choose the source in the **Source** drop-down. If you leave **Address** as "Auto", GoPxL automatically increments the value; otherwise, you can set it yourself. By default, connections are added directly after the last source in the connections map. You can optionally set the address at which a source is added using the **Address** parameter.  You can also add all sources, or all sources of a given type.

	Element	Description
		 Click **Add** to add the source to the EtherNet/IP Connections table below.
4	Ethernet ASCII Connections	The connections listed here will be sent by the protocol to a compatible PLC. You can delete a connection using the trashcan icon in its row, or change its ID.  You can also use shortcut commands to delete all of the connections, or specific categories of connections.

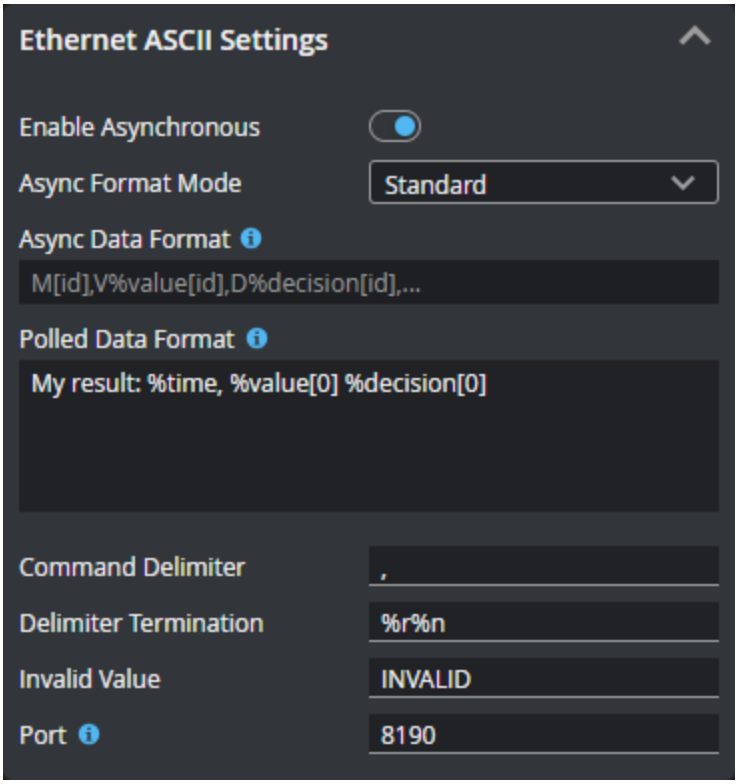
Element	Description



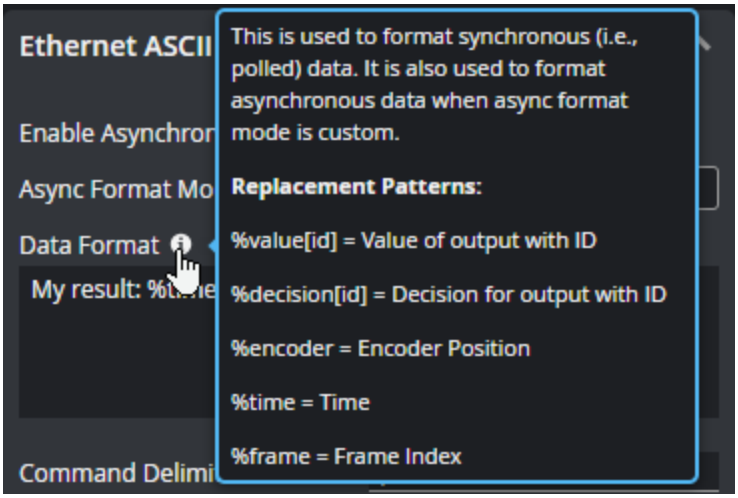
Use **Reindex...** to remove gaps of unused addresses between connections. You can also use **Reindex...** to set the value of the first index.

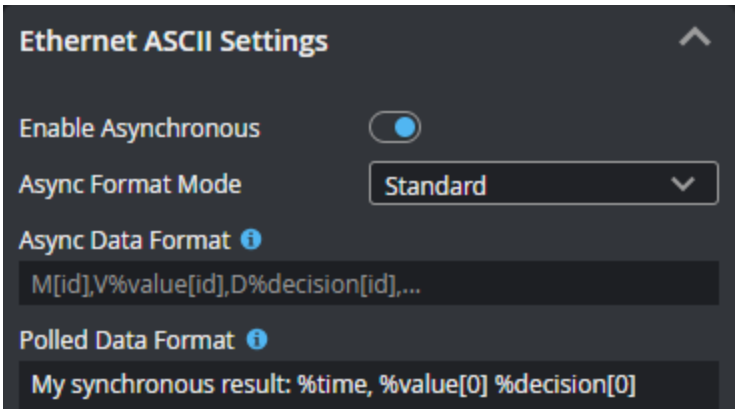


The Ethernet ASCII protocol provides several settings you must configure. The settings that are displayed change depending on the operation mode (polling vs. asynchronous) and the format mode.



Ethernet ASCII settings

Parameter	Description
Enable Asynchronous	Toggles asynchronous operation. In asynchronous mode, the data results are transmitted when they are available. Choose the **Async Format Mode** and, if you chose Custom in **Async Data Format**, set the format. (These parameters are only displayed when asynchronous mode is enabled.) Note that when asynchronous mode is enabled, data is still available using polling commands. When this option is disabled, the protocol only runs in polling (or synchronous) mode, and you must send commands on the data channel to request the latest result.
Async Format Mode	When you enable asynchronous operation, you must choose the format mode. The format mode is one of the following: Standard, Standard with Stamp, and Custom. For more information on the formats, see below.
Async Data Format and Polled Data Format Data Format	These fields contain the formats the protocol uses for the data it sends. GoPxL displays different "Format" fields depending on whether you enable asynchronous operation and what you set **Async Format Mode** to. For details on the supported replacement patterns you can use in the formats, click on the info icon next to the field's name:  When asynchronous operation is disabled, you configure the format in **Data Format**. When asynchronous operation is enabled and you choose either of the "Standard" modes, the format for asynchronous data is displayed in the read-only **Async Data Format** field. For polled data (which is always available), you can set a different format under **Polled Data Format**.

Parameter	Description
	 In asynchronous mode, when you set the mode to Custom, you set the format for both asynchronous and polling data in the **Data Format** field. C language *printf*-style formatting is also supported: for example, %sprintf[%09d, %value[0]]. This allows fixed length formatting for easier input parsing in PLC and robot controller logic.
Command Delimiter	Sets the command delimiter character.
Delimiter Termination	Sets the command termination character.
Invalid Value	Sets the invalid value characters.
Control Port	Sets the port. This is the port to which you PLC must connect.

Ethernet ASCII Protocol Format - PLC Information

You can use a programmable logic controller (PLC) that supports the Ethernet ASCII protocol to operate a sensor and receive values, such as measurements, stamps, or strings, and decisions. To use the ASCII protocol, it must be enabled and configured in the active job. By default, the control port used to connect to the sensor is 8190. For information on enabling Ethernet ASCII, as well as changing the control port and other protocol configurations in GoPxL, see *Ethernet ASCII Protocol* on page 345.

Two modes are available: asynchronous and polling.

When the sensor is in asynchronous mode, results are sent automatically, as soon as they are available. No commands are required, but the client can still send a command to the sensor to receive output data or to control the device.

When the sensor is in polling mode, you use commands to operate the sensor and to retrieve results. Ethernet ASCII protocol in polling mode only supports a subset of the tasks that can be accomplished in the web interface, and only measurement results can be transmitted to the PLC (scan data is not sent).

This section describes the following aspects of the Ethernet ASCII protocol:

- the polling operation commands you can use to control a sensor and retrieve results from it
- result formats

The protocol communicates using ASCII strings. The output result format from the sensor is user-configurable.

Commands and Formats

Commands are sent from the client to the sensor. The commands are not case sensitive. However, the `readprop` command's metrics path argument *is* case sensitive.

The command format is:

<COMMAND><DELIMITER><PARAMETER><TERMINATION>

If a command has more than one parameter, each parameter is separated by the delimiter.

Similarly, the reply has the following format:

<STATUS><DELIMITER><OPTIONAL RESULTS><DELIMITER>

The status can either be "OK" or "ERROR". The optional results can be relevant data for the command if successful, or a text based error message if the operation failed. If there is more than one data item, each item is separated by the delimiter.

You can change the delimiter and termination characters in the **Special Character** settings in the GoPxL interface. For more information, see *Ethernet ASCII Protocol* on page 345.

> Measurement IDs come from the **ID** field in the connection map. Be sure to add measurements to the connection map and make note of the IDs of the measurements you need to add to the data format.

Ethernet ASCII client commands

Ethernet ASCII client commands

Command	Arguments	Description
start	-	Starts the sensor.
stop	-	Stops the sensor.
trigger	-	Software Trigger.
loadjob	Job name	Load the specified job file, up to 63 character long, including the last null terminating character.
result	-	Return the measurement or string data specified in the custom format string. For a description of the measurement value (prefixed with the letter "V") and decision (prefixed with the letter "D") format, see *Measurement Output Format* on page 358 and *String Output Format* on page 359.
	One or more measurement or string IDs, separated by command delimiter.	Return the measurement or string data for each measurement or string ID in the following format string: M[ID],V[VALUE],D[DECISION],... Note that string data will not include a decision.

Command	Arguments	Description
		Example: result,0,1 Output: OK,M0,V-5000,D0,S1,VsampleString
measurement	One or more measurement IDs, separated by command delimiter.	Return the measurement data for each measurement ID in the following format string: M[ID],V[VALUE],D[DECISION],... For a description of the measurement value (prefixed with the letter "V") and decision (prefixed with the letter "D") format, see *Measurement Output Format* on page 358. Example: measurement,0,1 Output: OK,M0,V-5000,D0,M1,V5000,D1
string	One or more string IDs, separated by command delimiter.	Return the string data for each string id in the following format string: S[ID],V[VALUE],,... For a description of the string value (prefixed with the letter "V"), see *String Output Format* on page 359. Example: string,0,1 Output: OK,S0,VsampleString,S1,V2ndString
value	One or more measurement or string IDs, separated by command delimiter.	Return the measurement or string value for each measurement or string ID in the following format string: M/S[ID],V[VALUE],... For a description of the measurement/string value (prefixed with the letter "V"), see *Measurement Output Format* on page 358 and *String Output Format* on page 359. Example: value,0,1 Output: OK,M0,V-5000,S1,VsampleString
decision	One or more measurement IDs, separated by command delimiter.	Return the measurement decision for each measurement ID in the following format string: M[ID],D[DECISION],... For a description of the measurement decision (prefixed with the letter "D") format, see *Measurement Output Format* on page 358. Example: decision,0,1 Output: OK,M0,D0,M1,D1
stamp	-	Return the time, encoder and frame index stamp values for the first stamp ID found. Order of values returned is: Time, Encoder, Frame. For the "time" argument, this is the timetick of the last frame. A timetick is a 64-bit positive integer that is guaranteed to increase monotonically starting from zero. It is guaranteed to be unique for every scan from a given sensor group. To convert this value to microseconds, use the following: `time in microseconds = timetick / 1.024` Example: stamp

Command	Arguments	Description
		Output: OK,Time,381497381349,Encoder,0,Frame,18,
	One or more of these arguments in any order, separated by the command delimiter: "time" "encoder" "frame"	Return the specified stamp value(s) in the order of the provided arguments. Example: stamp,time,frame Output: OK,381497381349,18
	A stamp ID.	Return the time, encoder, and frame index stamp values for the passed stamp ID. Example: stamp,1
time	One or more stamp IDs, separated by command delimiter.	Return the stamp's time value for each stamp ID. The timetick of the last frame. A timetick is a 64-bit positive integer that is guaranteed to increase monotonically starting from zero. It is guaranteed to be unique for every scan from a given sensor group. To convert this value to microseconds, use the following: `time in microseconds = timetick / 1.024` Example: time,1 Output: OK,381497381349
encoder	One or more stamp IDs, separated by command delimiter.	Return the stamp's encoder value for each stamp ID. Example: encoder,1 Output: OK,0
frame	One or more stamp IDs, separated by command delimiter.	Return the stamp's frame index stamp value for each stamp ID. Example: frame,1 Output: OK,18
align	-	Performs the alignment operation.
clearalign	-	Clears sensor alignment.
readprop	One or more strings representing a resource whose property to return, separated by command delimiter.	Returns the resource JSON object specified by the resource path. It is possible to use JSON pointers to retrieve a specific value within the JSON document. Example #1: readprop,/system Output: OK,{"_links":{"self":{"href":"/system"}},"_schema":{"properties":{"autoLiveSimEnabled":{"sortOrder":200,"title":"autoLiveSimEnabled","type":"boolean"},"autoStart":{"sortOrder":100,"title":"autoStart","type":"boolean"},"runState":{"enum":[0,1,2],"enumText":

Command	Arguments	Description
		["Ready","Running","Conflict"],"readOnly":true,"sortOrder":0,"title":"runState","type":"integer"}},"sortOrder":0,"title":"","type":"object"},"autoLiveSimEnabled":true,"autoStart":true,"runState":1}, Example #2: readprop,/system#/runState,/system#/autostart Output: OK,1, true, Example #3: readprop,/jobs#/loadedJob Output: OK,"Job02", Example #4: readprop,/jobs#/loadedJobDirty Output: OK,false,

The following lists the possible errors for each command.

Ethernet ASCII error handling

Command	Arguments	Description
start	ERROR,Could not start the sensor	
stop	ERROR,Could not stop the sensor	
trigger	ERROR,Could not trigger	
loadjob	ERROR,Job name required.	Job name has to be provided as a command argument.
	ERROR,Failed to load job <job name>.gpjob	
result	ERROR,There is no measurement data to output. Please confirm that the sensor is running	Could not find any measurements. Make sure protocol's connection map is not empty and the sensor is running.
	ERROR,Invalid parameter. Please verify your input	Provided command argument was invalid. It must be a number.
	ERROR,Specified measurement ID not found. Please verify your input	Could not find a measurement data for specified ID.

Command	Arguments	Description
	ERROR,The custom format string is not valid. Please verify your input	Custom format string is invalid.
measurement	ERROR,One or more measurement ID must be provided	Measurement ID has to be provided as an argument.
	ERROR,There is no measurement data to output. Please confirm that the sensor is running	Could not find any measurements. Make sure protocol's connection map is not empty and the sensor is running.
	ERROR,Invalid parameter. Please verify your input	Provided command argument was invalid. It must be a number.
	ERROR,Specified measurement ID not found. Please verify your input	Could not find a measurement data for specified ID.
string	ERROR,One or more string ids must be provided.	String ID has to be provided as an argument.
	ERROR,There is no data to output. Please confirm that the sensor is running.	Could not find any data. Make sure protocol's connection map is not empty and the sensor is running.
	ERROR,Invalid parameter. Please verify your input.	Provided command argument was invalid. It has to be a number.
	ERROR,Specified id not found. Please verify your input.	Could not find an output for specified ID.
value	ERROR,One or more measurement/string ids must be provided.	Measurement/string ID has to be provided as an argument.
	ERROR,There is no data to output. Please confirm that the sensor is running.	Could not find any data. Make sure protocol's connection map is not empty and the sensor is running.
	ERROR,Invalid parameter. Please	Provided command argument was invalid. It has to be a number.

Command	Arguments	Description
	verify your input.	
	ERROR,Connection with id not found.	Could not find data for specified ID.
decision	ERROR,One or more measurement ids must be provided.	Measurement ID has to be provided as an argument.
	ERROR,There is no data to output. Please confirm that the sensor is running.	Could not find any data. Make sure protocol's connection map is not empty and the sensor is running.
	ERROR,Invalid parameter. Please verify your input.	Provided command argument was invalid. It has to be a number.
	ERROR,Connection with id not found.	Could not find data for specified ID.
	ERROR,String connection with id does not contain a decision.	A string id was included in the decision command.
	ERROR,Connection with id %u is not a measurement or string.	An id mapping to neither measurement nor string was included in the decision command.
stamp	ERROR,Invalid stamp command format.	Invalid command arguments.
	ERROR,One or more stamp ids must be provided.	Stamp ID has to be provided as an argument.
	ERROR,Invalid parameter. Please verify your input.	Provided stamp ID was invalid. It has to be a number.
	ERROR,Stamp with id not found.	Could not find stamp data for specified ID.
	ERROR,Connection id is not a stamp.	Specified id was not a stamp.
time	ERROR,One or more stamp ids must be provided.	Stamp ID has to be provided as an argument.
	ERROR,Invalid parameter. Please	Provided stamp ID was invalid. It has to be a number.

Command	Arguments	Description
	verify your input.	
	ERROR,Stamp with id not found.	Could not find stamp data for specified ID.
	ERROR,Connection id is not a stamp.	Specified id was not a stamp.
encoder	ERROR,One or more stamp ids must be provided.	Stamp ID has to be provided as an argument.
	ERROR,Invalid parameter. Please verify your input.	Provided stamp ID was invalid. It has to be a number.
	ERROR,Stamp with id not found.	Could not find stamp data for specified ID.
	ERROR,Connection id is not a stamp.	Specified id was not a stamp.
frame	ERROR,One or more stamp ids must be provided.	Stamp ID has to be provided as an argument.
	ERROR,Invalid parameter. Please verify your input.	Provided stamp ID was invalid. It has to be a number.
	ERROR,Stamp with id not found.	Could not find stamp data for specified ID.
	ERROR,Connection id is not a stamp.	Specified id was not a stamp.
align	ERROR,Could not align	
clearalign	ERROR,Could not clear alignment	
readprop	ERROR,String representing a resource must be provided	Resource path must be provided as an argument.
	ERROR,Could not read property	Resource may not exist.

Data Output Format Mode

Polling Mode

Polled data output is displayed using the custom format. You can modify this format in the GoPxL interface. For more information, see *Ethernet ASCII Protocol* on page 345. Because polling

mode is always available, even when asynchronous output is enabled, you can always modify the custom format.

Format Mode	Description
Custom	This format can be changed/edited by the user. Default formatting is: %time, %value[0] %decision[0] It is possible to format the specified measurement or string value according to the format string. This pattern does not apply to measurement decisions. Example: %sprintf[%9.3f, %value[0]] to format the measurement value of measurement id 0 as a floating-point number with 9 digits and 3 decimal places.

Asynchronous Mode

When asynchronous output mode is enabled, the data output can be in one of three formats:

Format Mode	Description
Standard	This is a pre-defined format that cannot be changed. The format is available as Read-Only information for reference. Formatting is: M[id],V%value[id],D%decision[id],... Note that string ids will not include decisions.
Standard with Stamp	The format is the Standard format pre-pended with the Time and Encoder information. This is a pre-defined format that cannot be changed. The format is available as Read-Only information for reference. Formatting is: T%time,E%encoder,M[id],V%value[id],D%decision[id],.. Note that string ids will not include decisions.
Custom	This format is the same as the format used by Poll mode and can be changed/edited by the user. Default formatting is: %time, %value[0] %decision[0] It is possible to format the specified measurement value according to the format string. This pattern does not apply to measurement decisions. Strings can also be formatted using sprintf. Example: %sprintf[%9.3f, %value[0]] to format the measurement value of measurement id 0 as a floating-point number with 9 digits and 3 decimal places.

Measurement Output Format

A measurement contains the fields described below.

Field	Type	Address offset	Description
Value	32s	0	Measurement value. This value is

Field	Type	Address offset	Description
			the actual measurement value multiplied by 1000. If the measurement value is invalid, the string "INVALID" is the output for the measurement value.
Decision	8u	4	Measurement decision. Bit 0: 0 = Fail 1 = Pass Bit 1: 0 = Decision value is valid 1 = Decision value is invalid

String Output Format

A string contains the fields described below.

Field	Type	Address offset	Description
Value	string	0	String value. If the string value is invalid, the invalid value parameter is output.

PROFINET Protocol

You can use a PROFINET Programmable Logic Controller (PLC) over Ethernet to operate a sensor and receive values, such as measurements, stamps, or strings. PROFINET only supports a subset of the tasks that can be accomplished in the web interface (for example, starting, stopping, and aligning sensors, and switching Jobs). Scan data can't be sent to the PLC.

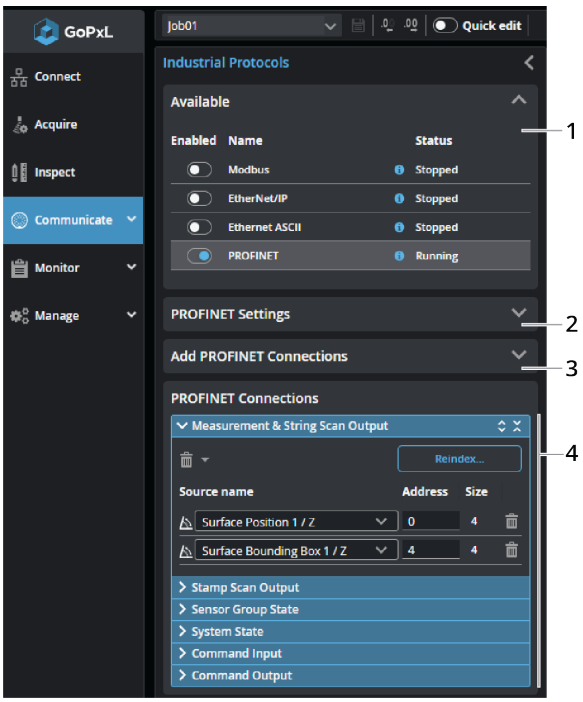
You can send data from a PLC to GoPxL after adding and configuring an instance of the User Data Input tool; for more information on the tool, see *User Data Input Tool* on page 308. To send data over PROFINET to the tool, you must use a special Implicit messaging command (command 9) on the PLC; for more information, see *PROFINET Protocol Format - PLC Information* on page 364.

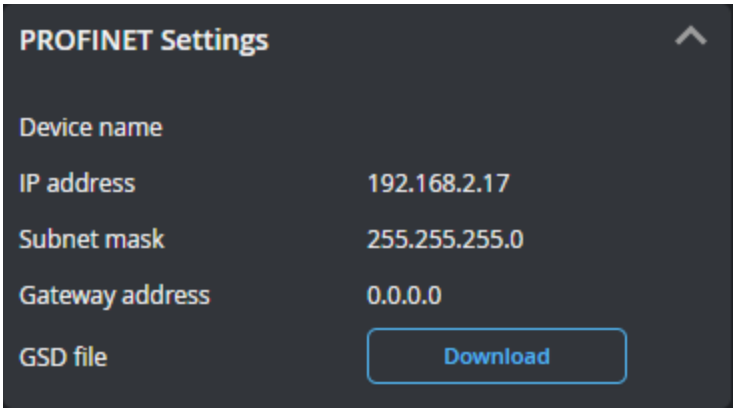
GoPxL supports PROFINET on sensors, on GoMax ORIN / ORIN+ / NX, and in PC instances of GoPxL.

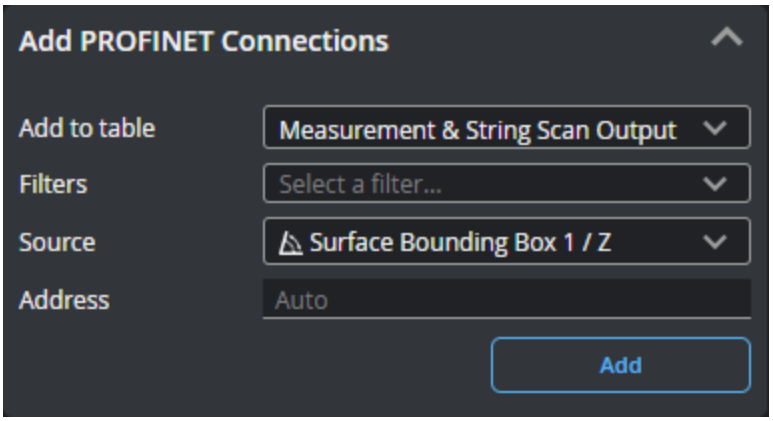
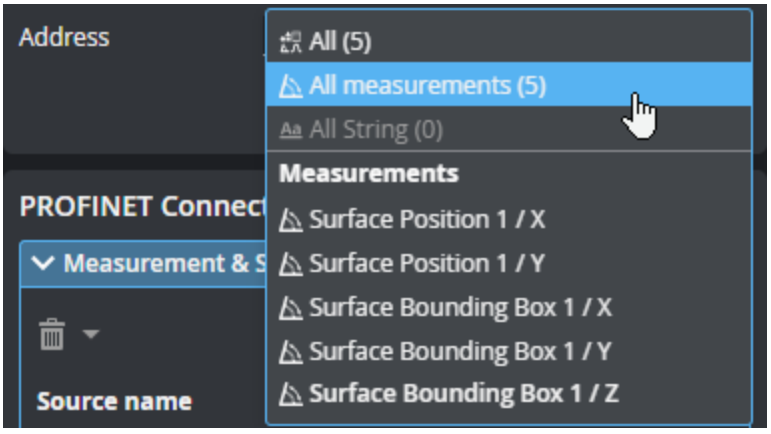
If you are using a GoMax Smart Vision Accelerator, only connect Ethernet-based PLCs or PCs to the ETH1 port on the GoMax device (whether via a switch or not).

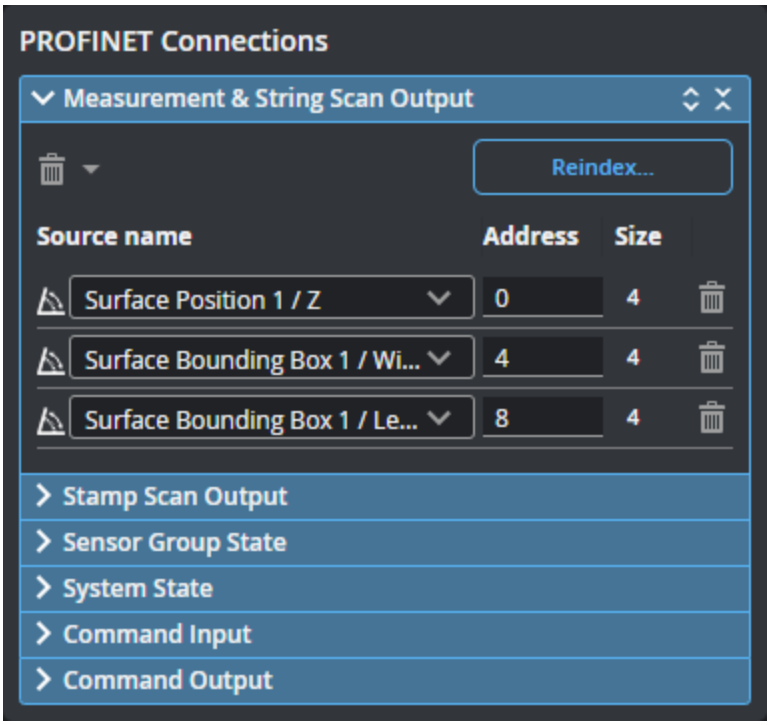
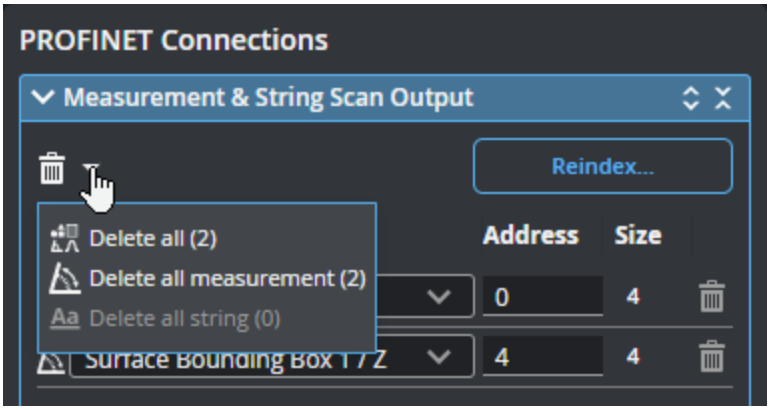
Due to library limitations, PROFINET cannot be stopped manually after it has started. To stop it, disable the service (an error will be displayed in the log), save your job file, and restart the sensor.

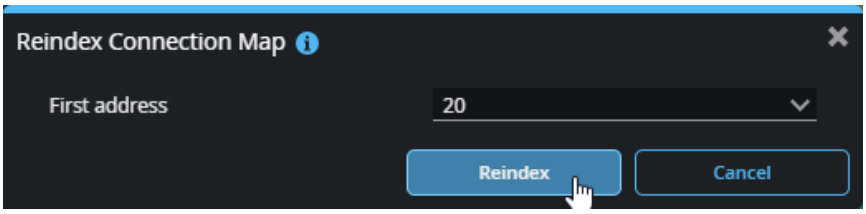
You configure the protocol on the **Communicate** > **Industrial** page. After configuring the PROFINET protocol in GoPxL (see below), use the information in *PROFINET Protocol Format - PLC Information* on page 364 to configure your PLC.



	Element	Description
1	Available protocols	Click a row in the table to choose which protocol to configure. Enable a protocol by toggling the slider to the right.
2	PROFINET Settings	 **GSD File** Click **Download** to save a GSD file to the client computer for later installation on the PLC. This section also provides network information.

	Element	Description
3	Add PROFINET Connections	Lets you add sources (such as stamps, measurements, or strings) to send over the protocol. Added connections appear in the **PROFINET Connections** section at the bottom of the panel. When GoPxL is running on a standalone sensor (one not controlling others), **Add to Table** only contains two options (Measurement Scan Output and Stamp Scan Output), which represent register assemblies. When GoPxL is running on a sensor controlling other sensors, or on a PC or GoMax NX, **Add to Table** contains an additional entry (Sensor Group State). Use this parameter to choose which register a source is added to. Choose the source in the **Source** drop-down. If you leave **Address** as "Auto", GoPxL automatically increments the value; otherwise, you can set it yourself. By default, connections are added directly after the last source in the connections map. You can optionally set the address at which a source is added using the **Address** parameter.  You can also add all sources, or all sources of a given type.  Click **Add** to add the source to the PROFINET Connections table below.

	Element	Description
4	PROFINET Connections	The connections listed here are the ones that will be sent by the protocol to a compatible PLC. You can delete a connection using the trashcan icon in its row, or change its ID. When changing an ID, make sure to use the size of the source to correctly calculate the minimum offset.  For string sources, you must specify a maximum expected length in the source's **Size** field. You can also use shortcut commands to delete all of the connections, or specific categories of connections.  Use **Reindex...** to remove gaps of unused addresses between connections. You can also set the value of the first index.

Element	Description
	

Using PROFINET on a PC

If you are using PROFINET on a PC (using the GoPxL Manager or software developed using the GoPxL SDK), be aware of the following:

- The PC instance must either run in Administrator or be installed as a service. Otherwise, the TIA portal will not be able to discover the instance.
- The IP address of the PC instance must be a specific hardware address (such as the PC's network adapter), and not the "Any IP (0.0.0.0)" or "Localhost (127.0.0.1)".
- You must install a pcap-API compatible driver such as as Npcap or WinPcap. This lets the PC capture and send the raw network traffic necessary for the PC to interact directly with the PROFINET protocol.
- Because Windows is not a realtime OS, you must fully test a PC-based sensor system and limit the use of unnecessary software. Otherwise, you might encounter timing issues, and PLC requirements may not be met or connections may get dropped. The PC should have a CPU able of handling occasional processing spikes from either Windows or other software.

For information on running GoPxL on a PC, see *Running GoPxL on a Windows PC* on page 403.

PROFINET Protocol Format - PLC Information

PROFINET is an Industrial Ethernet network protocol that allows controllers such as PLCs to communicate with sensors. Sensors are PROFINET IO devices with Conformance Class A.

> The connection mappings for the industrial protocols (PROFINET, EtherNet/IP, and Modbus) have changed to correct a byte alignment issue. To allow communication between PLCs and GoPxL 1.2, you may need to update PLC logic created for GoPxL 1.1.
>
> For EtherNet/IP, the EDS file has been updated to reflect the changes.

GoPxL supports PROFINET communication to provide system state, sensor group state, stamp, and measurement output data to a PLC, as well as accept incoming commands. PROFINET defines the entire data exchange between controllers (called "IO-Controllers") and the devices (called "IO-Devices"). GoPxL operates as a PROFINET IO-Device, and a PLC that connects to it acts as the PROFINET IO-Controller. All communication is initiated by the PLC, with GoPxL responding to the PROFINET requests.

> GoPxL supports PROFINET on sensors, on GoMax ORIN / ORIN+ / NX, and in PC instances of GoPxL.

If you are using PROFINET on a PC, see *Using PROFINET on a PC* on the previous page for important information.

While the PROFINET protocol includes several types of requests, the two basic operations are reading and writing PROFINET objects. Commands are invoked by the PLC sending *set* requests. Data is requested by the PLC by sending *get* requests for PROFINET objects.

Connections Map

The PROFINET protocol supports the following objects.

Assembly Objects

A register assembly is a container that contains register blocks. This is a one-to-one mapping, which means each PROFINET module will have a corresponding control framework register assembly. These modules are added to the GoPxL irrespective of the selection of register assembly, but the data is added to this PROFINET module based on the selection of register assembly.

Assembly Objects

PROFINET Module Name	Register assembly	Supported Register Block Types	Service	Instance ID	Slot Number	Max size of Register Assembly
Command input module	Control Input register assembly	"ControlInput"	Write	0x30	1	67
Command output module	Control Output register assembly	"ControlOutput"	Read	0x31	2	3
System State module	System State register assembly	"System"	Read	0x32	3	76
Stamp module	Stamp Scan	"Stamp"	Read	0x33	4	36

PROFINET Module Name	Register assembly	Supported Register Block Types	Service	Instance ID	Slot Number	Max size of Register Assembly
	Output register assembly					
Measurement and String module	Measurement & String Scan Output register assembly	"MeasurementValue" "String"	Read	0x34	5	800
Sensor Group State module	Sensor Group State register assembly	"Scanner"	Read	0x35	6	18

The PROFINET protocol supports various block types. Each block type can be either an input block or an output block. Input blocks contain write-only objects and are used by the PROFINET IO-Controller to send commands such as starting or stopping a scan. Output blocks contain read-only objects and are used to store various types information, such as control, system, and sensor group state data as well as scan output data (stamp and measurements).

Supported Block Types

The block types supported by the PROFINET protocol are listed in the table below. The size field indicates the number of registers that each block type requires.

Supported Block Types

Block Type	Read or Write only?	Size (number of PROFINET registers)	Description
"ControlInput"	WO	67	Command message set by the PROFINET IO-Controller.
"ControlOutput"	RO	3	Result data corresponding to a command that was written to the Control Input block.
"MeasurementValue"	RO	4	Measurement data of a scan output measurement data message.
"Scanner"	RO	18	Information about a specific sensor group.
"Stamp"	RO	36	Stamp message information.
"String"	RO	8 (by default)	String data of a scan output string data message. Size can be configured through the GoPxL web interface.
"System"	RO	76	Information about the LMI device.

Control Input

The input of the Control Input Block contains the fields described below:

Control Input Fields

Field	Type	Address offset	Description
Command Sequence	16u	0	The command sequence number is set by the external IO-

Field	Type	Address offset	Description
Number			Controller to uniquely identify a command request. This number is sent back in the Control Output. This is an optional field but is recommended that PROFINET IO-Controllers use this to help the IO-Controllers correlate the Command Status in the Control Output with a specific command request in Control Input.
Command Identifier	8u	2	Identifies the requested action from the external client. Supported actions: 0 = No Command 1 = Start Scanning 2 = Stop Scanning 3 = Align Sensor Group 4 = Clear Alignment 5 = Load Job File 6 = Software Trigger 9 = Set User Data Input. For use with the User Data Input tool; for more information see *User Data Input Tool* on page 308.
Unused	UINT8	3	Reserved.
Command Arguments	Array[8u]	4	Optional. **Load Job File** The argument must be a null-terminated string containing the job filename. Do not include the file extension, as the control service server appends it automatically. Each register holds a single character, with a limit of 63 characters including the null terminator. **Set User Data Input** Use up to the full 63 registers for data input.

Control Output

The output of the Control Output Block contains the fields described below:

Control Output Fields

Field	Type	Address offset	Description
Command Sequence Number	16u	0	The command sequence number received from the IO-Controller in the Control Input.
Command Status	8u	2	Status of the command, referenced by the Command Sequence Number. 0 = No Command Received

Field	Type	Address offset	Description
			1 = In Progress 2 = Success 3 = Fail

System State

The output of the System State Block contains the fields described below:

System State Fields

Field	Type	Address offset	Description
System Uptime	64u	0	Amount of time the application has been running, in seconds.
System State	8u	8	Whether system is running or not. 0 = System is stopped 1 = System is running
Current Job File Name Length	8u	9	Number of characters in the current job file name.
Current Job File Name	Array[8u]	10	Name of the job file currently loaded into the system, without the extension. Maximum length of job file name is 64 characters. Each register contains a single character. Job file name is NOT null terminated.
Buffer Count	8u	74	The current amount of buffered scan output data when buffering is enabled.
Buffer Overflow	8u	75	A value of 0 indicates that no overflow has occurred. A value of 1 indicates that overflow occurred and data is being lost.

Sensor Group State

The output of the Sensor Group Block contains the fields described below:

Sensor Group State Fields

Field	Type	Address offset	Description
Current Encoder Position	64s	0	The current encoder position.
Current Sync Time	64u	8	Current time of the clock used to stamp all scan output messages, in microseconds.
Alignment State	8u	16	Indicates whether all devices under the sensor group are aligned (that is, all devices in the sensor group have a non-identity transform matrix). 0 = Not Aligned 1 = Aligned

Field	Type	Address offset	Description
			2 = Aligning
Laser Enabled	8u	17	Whether one or more devices under a sensor group has its laser enabled to be able to turn on. 0 = All devices' lasers are disabled and cannot turn on 1 = One or more devices' lasers can be turned on

Stamp

The output of the Stamp Block contains the fields described below:

Stamp Fields

Field	Type	Address offset	Description
Z Position	64s	0	Encoder position at time of last index pulse of last frame.
Encoder Position	64s	8	Encoder position of last frame when the image data was scanned/taken.
Time	64u	16	The timetick of the last frame. A timetick is a 64-bit positive integer that is guaranteed to increase monotonically starting from zero. It is guaranteed to be unique for every scan from a given sensor group. To convert this value to microseconds, use the following: `time in microseconds = timetick / 1.024`
Frame Index	64u	24	The frame number of the last frame.
Digital Input States	8u	32	Digital input states of last frame. Bit 0: State of digital input 1 on master or sensor Bit 1: State of digital input 2 on master or sensor
Buffer	8u	33	Unused
Buffer	8u	34	Unused
Buffer	8u	35	Unused

Measurement

The output of the Measurement Block contains the fields described below:

Measurement Fields

Field	Type	Address offset	Description
Value	32s	0	Measurement value. This value is the actual measurement value multiplied by 1000. If the measurement value is invalid, the 32-bit value is set to 0x80000000.

String

The output of the String Block contains the fields described below:

Measurement Fields

Field	Type	Address offset	Description
String	string	0	String value. Unused space will be set to null (0x0). The string is otherwise not null terminated. If invalid, the block's buffer is entirely set to null, equivalent to an empty string. The string will be truncated if the block's size is smaller than the length of the string.

Digital Output

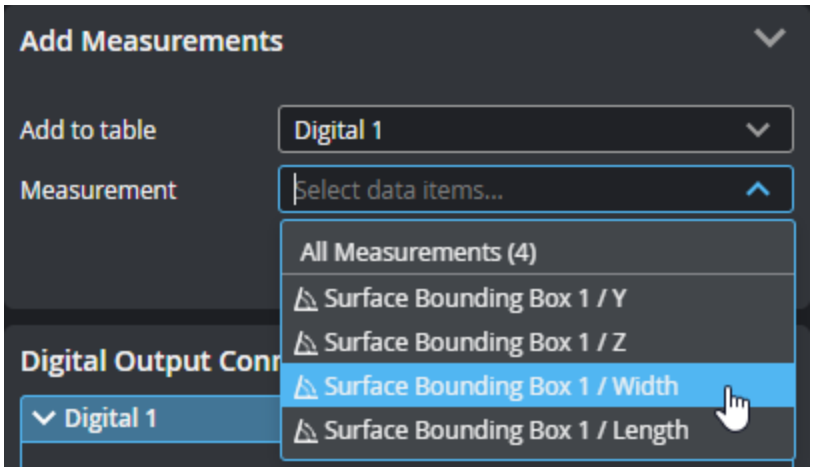
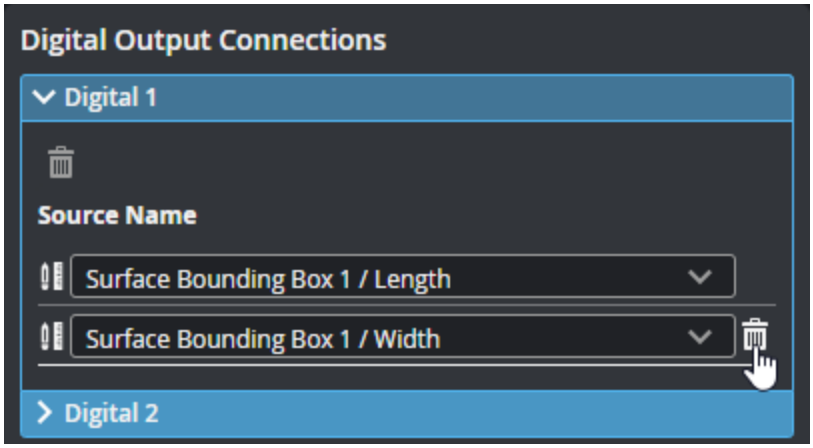
GoPxL can send digital signals over a sensor's digital outputs based on measurement decisions, software commands, and other events. These signals can then be used to control a PLC or other external device, such as indicator lights or air ejectors. The following trigger events are available:

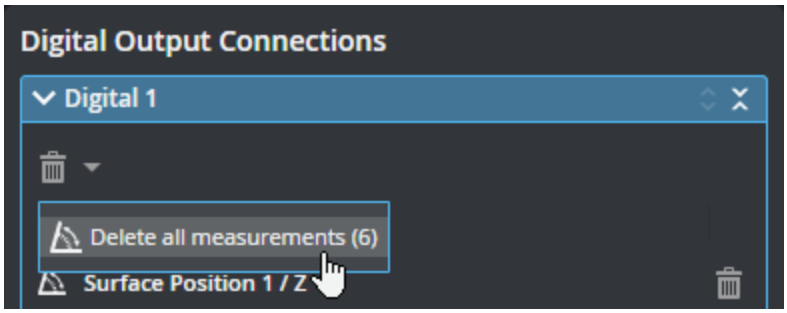
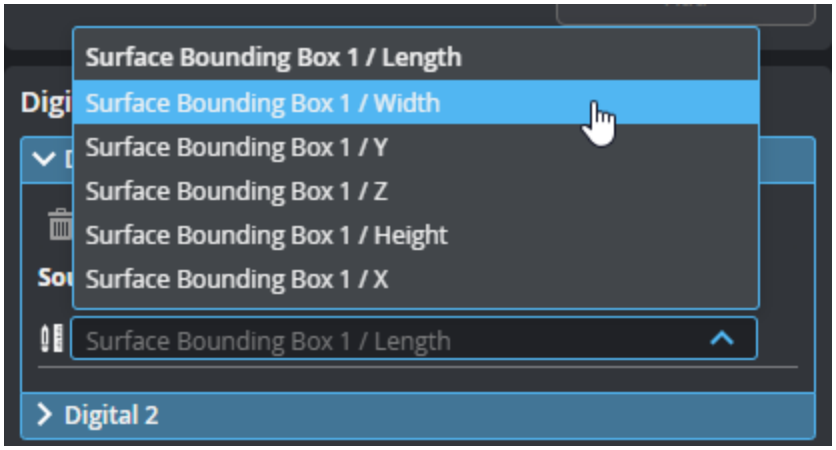
- Measurements
- Software (that is, a command sent from GoPxL SDK applications)
- Alignment state
- System state (that is, when the system is ready to accept digital input triggers)

GoPxL provides two digital output channels. Additionally, for Gocator 2D Smart Cameras, you can control the camera's signal lights. For information on wiring digital outputs to external devices, see the following: *Gocator Power / IO Connector* on page 65.

Counts of low and high digital signals, as well as digital output drops, are available on the **Health** page, in the expandable **Sensor {n}** sections. For more information, see *Health* on page 388.

	Element	Description
1	Settings	**Enable** lets you enable digital output. **Active Device** shows which sensor's digital output port GoPxL will send digital signals. Each expandable "Digital {n}" section lets you choose the type of trigger event and configure settings related to the signal. For Gocator 2D Smart Cameras, an additional Status light output is available. When you set **Trigger event** to Measurements, you use the **Add Measurements** and **Digital Output Connections** sections to add and manage the measurements that GoPxL sends as digital outputs. Otherwise, a digital signal is sent when the selected trigger event occurs. For information on the trigger settings, see *Digital Output Settings* on page 371.
2	Add Measurements	Lets you choose a measurement and which digital output the signal is sent over. This section is only displayed if at least one digital output's trigger is set to Measurements. To set a measurement for digital output, choose the output in **Add to**

	Element	Description
		table, choose the measurement in **Measurement**, and click **Add**. You can also add all measurements at once by selecting "All Measurements".  After adding a measurement, it appears in the **Digital Output Connections** section of the digital output you chose.
3	Digital Output Connections	Lets you view, delete, and change the measurements that trigger a digital output signal. This section is only displayed if at least one digital output's trigger is set to Measurements. To delete a measurement, click the trashcan icon next to it.  You can also use shortcut commands to delete all of the connections.

Element	Description
	 To change a measurement, click an output's drop-down and choose a different measurement. 

Digital Output Settings

Parameter	Description
Trigger event	GoPxL can send a digital output signal when one of several trigger events occurs. Depending on the event you choose, other parameters are displayed that you may need to configure. For more information, see the other parameter descriptions below. **Measurements** GoPxL sends a digital signal based on the decisions, for each frame, returned by the measurements added to the digital output. Set **Signal** to choose between a pulsed or continuous signal. Optionally, set a pulsed signal to be scheduled with **Scheduled**. You must set **Assert on** to configure what types of measurement decisions result in a digital signal. **Software** When GoPxL receives a software command from a GoPxL SDK application, it sends a digital signal over the configured digital output. Use **Signal** to choose between a pulsed or continuous signal. Optionally, set a pulsed signal to be scheduled with **Scheduled**. For pulsed or scheduled pulse, the sensor must be scanning before

Parameter	Description
	sending the command. For continuous signal, the sensor does not have to be scanning **Alignment** **System state** The System state output signal goes high after the sensor has started and is ready to receive digital input to start triggering for scanning, that is, "External Input" triggering is enabled in **Source** in the **Trigger** panel. For more information on triggers, see *Triggers* on page 110.
Invert output signal	Inverts the output signal. Normally, GoPxL will send a high digital signal when an event occurs. Toggling this setting causes GoPxL to send a low signal when the event occurs.
Assert on	When **Trigger event** is set to Measurement, use this setting to determine which type of decision in the measurements added to the output in **Digital output connections** cause GoPxL to send a digital signal. GoPxL checks decisions for each frame. One of the following: **Green for pass**: If all added measurements return a pass decision, GoPxL sends a digital signal. **Red for any fail or invalid**: If any added measurement returns a fail decision or Invalid, GoPxL sends a digital signal. **Always**: GoPxL sends a digital signal, for each frame, after the tools producing the selected decisions have finished processing, regardless of the result (pass, fail, or invalid). The output activates only once for each frame even if multiple decision sources are selected. LMI recommends using pulsed digital signals with this mode, as otherwise the signal will remain high indefinitely.
Signal	Measurement and Software trigger events let you choose the signal type (pulsed or continuous). **Pulsed** GoPxL sends a pulsed signal over the digital output, using the value you set in **Pulse width**. **Continuous** GoPxL sends a continuous signal over the digital output. When the trigger event is Measurement and **Assert on** is set to always, LMI recommends using pulsed digital signals instead of continuous.

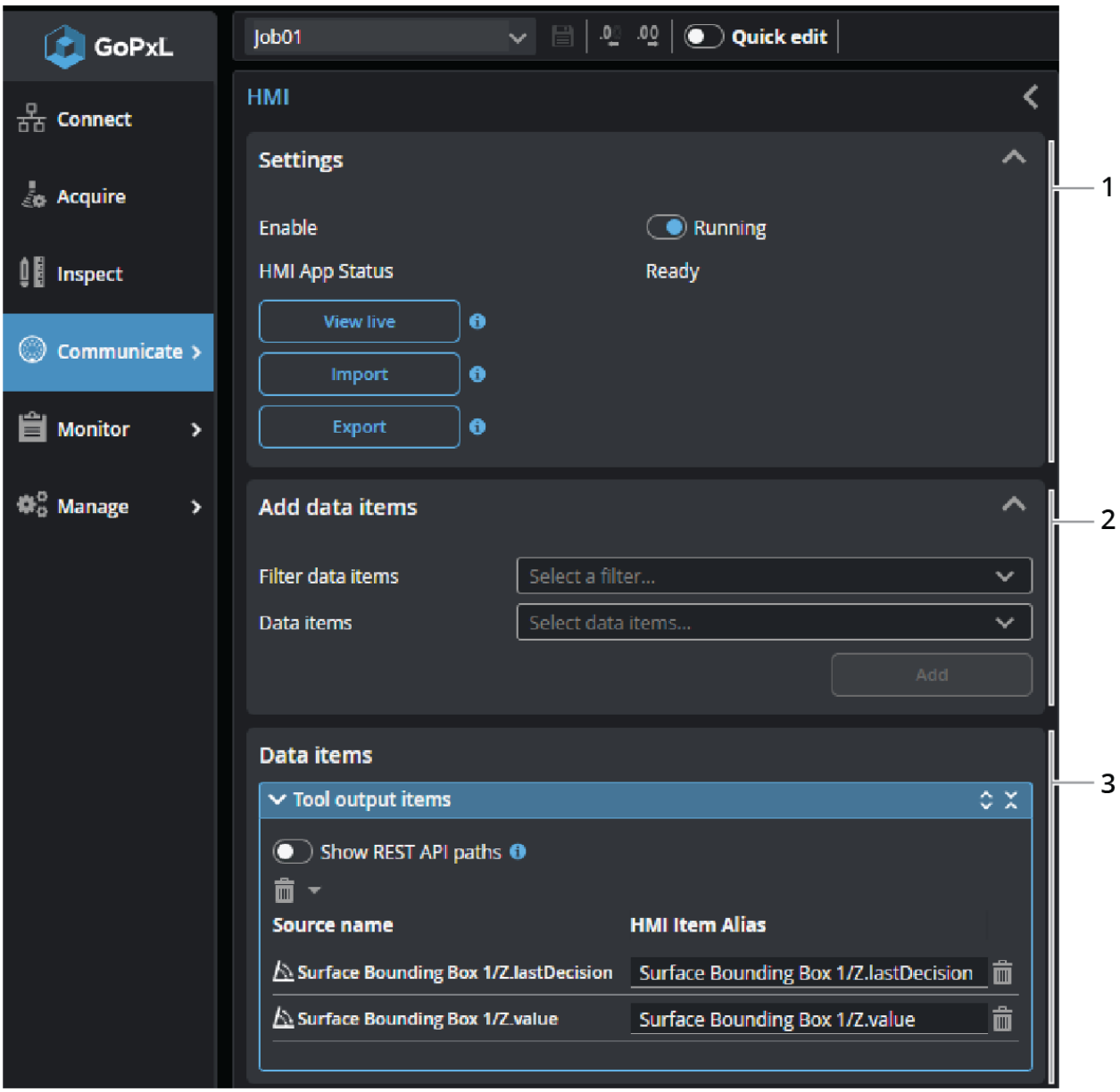
Parameter	Description
Pulse width	The pulse width used when the signal type is pulsed. Note that Measurement and Software trigger events let you choose between pulsed and continuous, whereas the exposure types are always pulsed.
Scheduled	When this setting is enabled, you can set a delay using **Delay** and **Delay Domain**. Only Measure and Software trigger events let you schedule output.
Delay Domain	The delay and domain GoPxL uses when **Scheduled** is enabled. The delay domain is either time-based or based on encoder distance.
Delay	The unit depends on whether the domain is time (microseconds) or encoder (millimeters).

HMI

The HMI panel lets you enable GoHMI, choose which outputs you want to send to an HMI application, and launch with the HMI. You can also import and export an HMI.

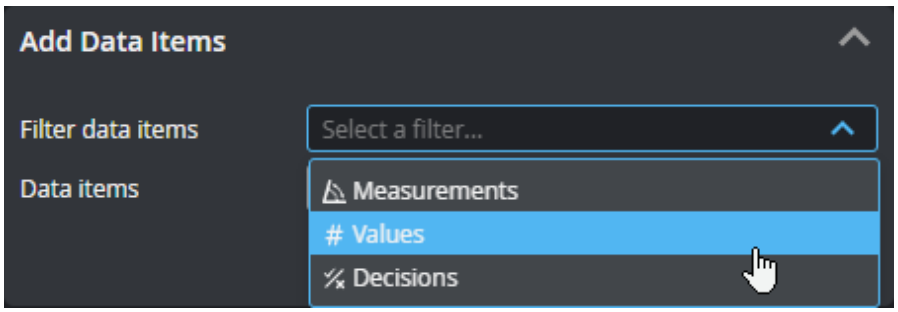
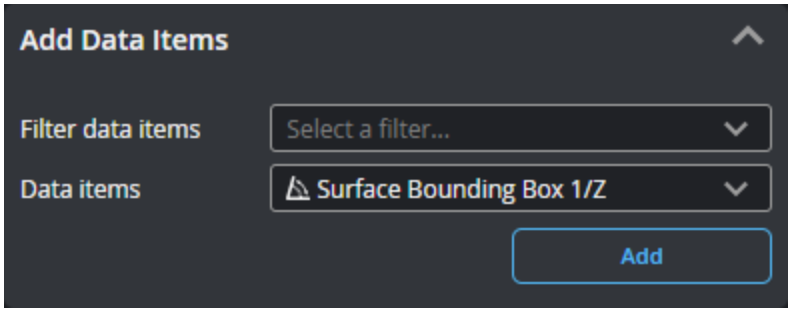
You can send measurement data items (which contain both values and decisions), or values or decisions individually.

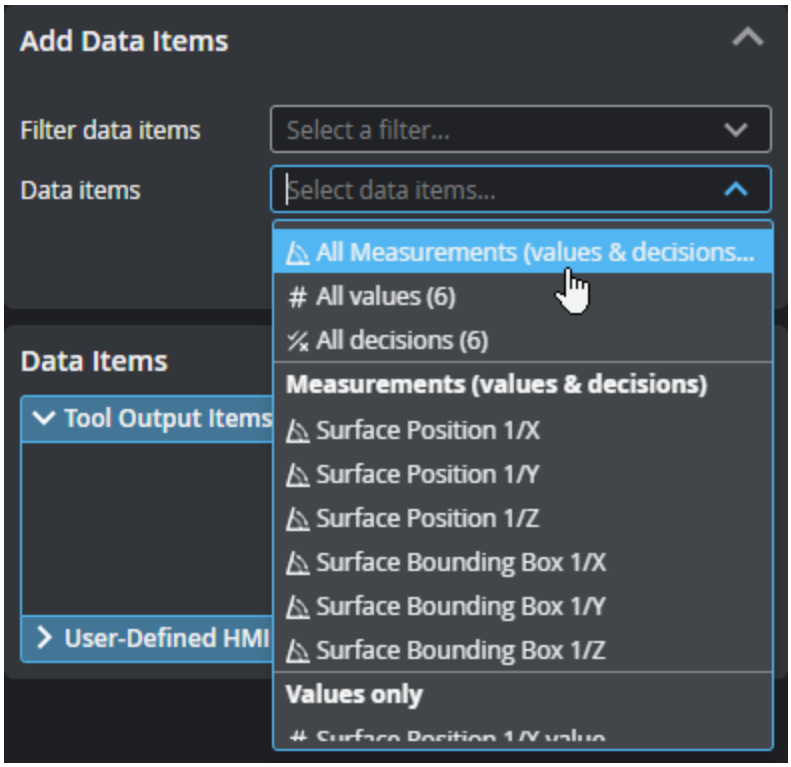
You configure the HMI on the **Communicate** > **HMI** page. For information on how to design an HMI app, see *GoHMI and GoHMI Designer* on page 425.



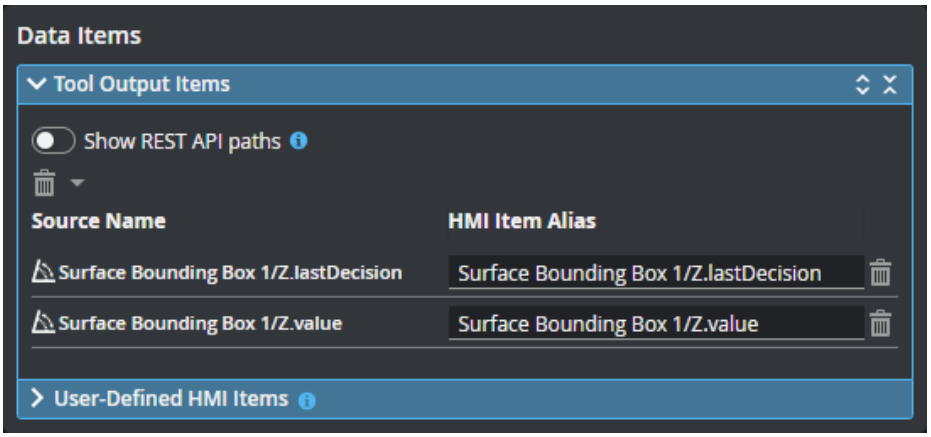
When you start HMI on a PC instance of GoPxL, a helper application (WebIQ Server) launches.

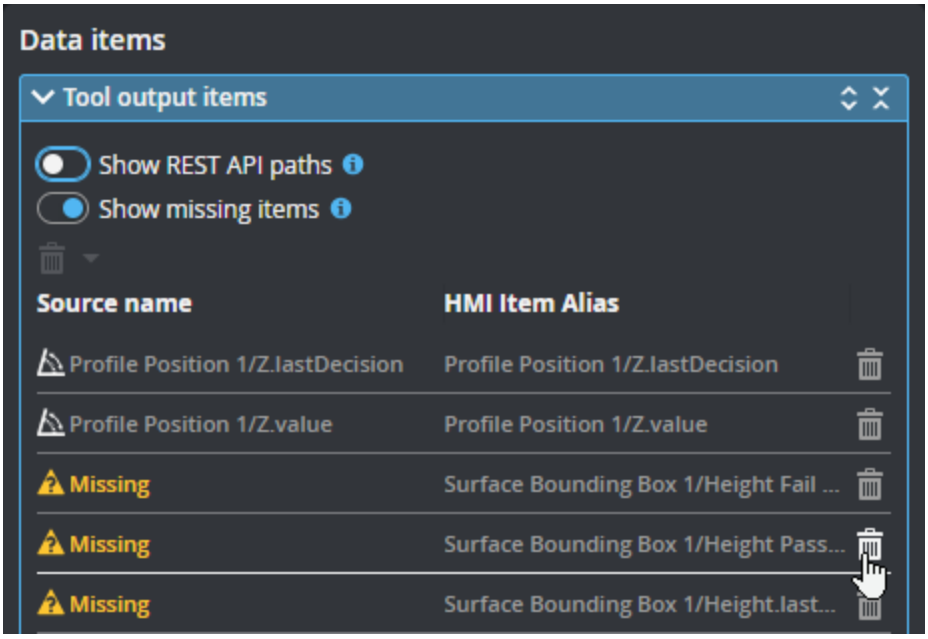
	Element	Description
1	Settings	After you have enabled the service, the following functions are available: **View live** Opens the HMI in a browser tab. **Import / Export**

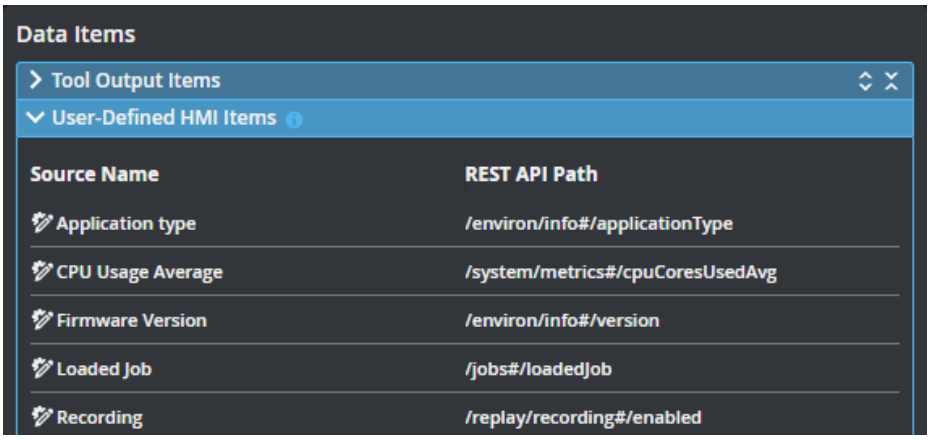
	Element	Description
		Lets you import and export HMIs so that you can transfer them to a different platform, for example, going from a PC to a GoMax device, or back up an HMI. **HMI App Status** Shows the status of the app.
2	Add Data Items	Lets you add sources (such as stamps or measurements) that GoPxL will send to the running HMI. Added connections appear in the **Data Items** section at the bottom of the panel. If you have many measurements, you can filter them by type to make it easier to find the item you want in the **Data Item** drop-down. You can enable more than one filter.  Choose an item in **Data items**. You can also  You can also add all items, or all items of a given type.

Element		Description
		
3	Data Items	The data listed here will be sent to the running HMI app. For more information, see *Data Items* below.

Data Items

Type	Description
Tool Output Items	This section contains the measurement decisions and values you added, which will be sent to the HMI.  You can delete an item using the trashcan icon in its row. You can also use shortcut commands to delete all of the items, or specific categories of items.

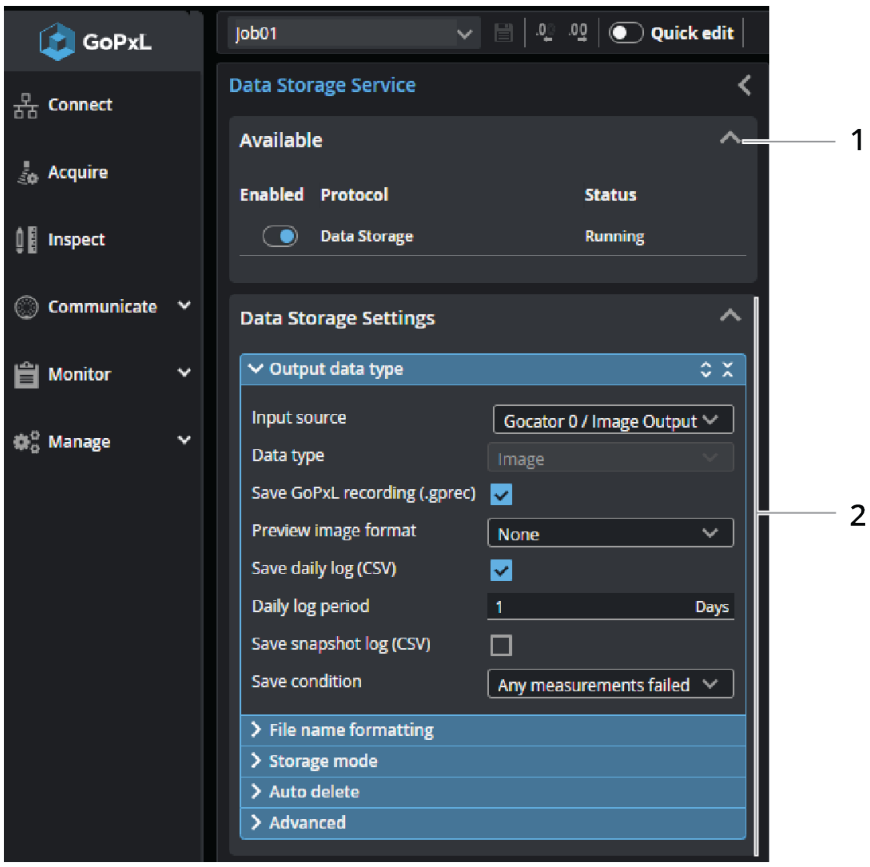
Type	Description
	Show REST API Paths Toggles between showing the full internal path to an item, and an alias, which is a simplified representation of the item. In GoHMI Designer, you will see the REST API path for the items you add. Enabling this setting in GoPxL can help make it easier to find which item is which when developing an HMI. **Show missing items** If the source for an item is missing (that is, the tool providing the source has been removed in the job, or an output has been disabled), this option is displayed. Enabling it lists connections whose source can't be found. You can delete the missing connection, enable the source measurement in the tool if it has just been disabled. If the tool itself is missing and you add it again, you may still need to delete missing items on the HMI panel, and add them back again, to make sure you add the correct ones. . 
User-Defined HMI Items	This section contains GoPxL REST API resources that are not tool outputs. They are defined and managed from within GoHMI Designer.

Type	Description
	

Data Storage Service

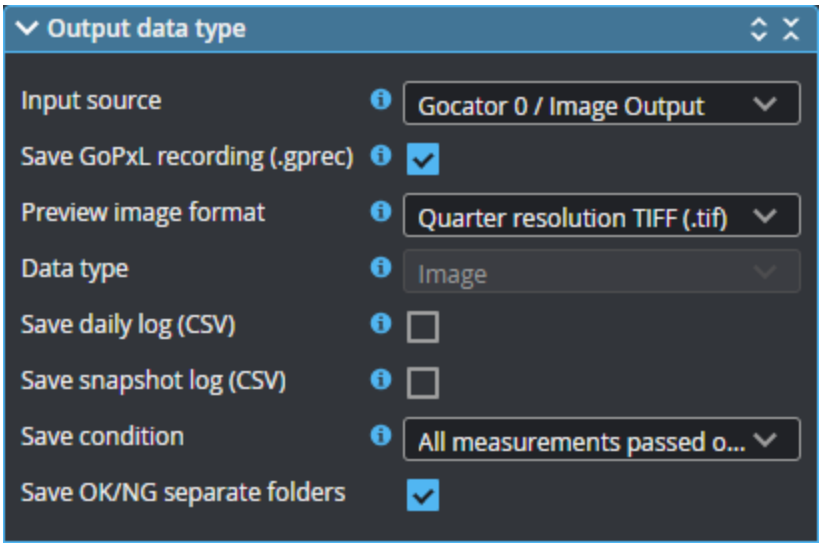
The data storage service lets you upload data acquired on sensors and smart cameras to a remote FTP server. On GoMax Smart Vision Accelerators, smart cameras, and GoPxL for PC, data can also be stored to a local folder. Auxiliary data, such as preview images and logs, can also be stored.

	On GoMax devices and 2D smart cameras, when the service is configured to save files on the device's internal FTP server (**Storage mode** > **Save location** is set to Local), the optional timestamp used in file names is based on the internal time of the device, and the time is reset when the device power cycles.



	Element	Description
1	Available	Disables the data storage service and displays its status.
2	Data Storage Settings	Lets you configure the data storage service. For more information, see the tables below.

You configure the source, save condition, and types of data the service saves to the FTP server in the expandable **Output data type** section.

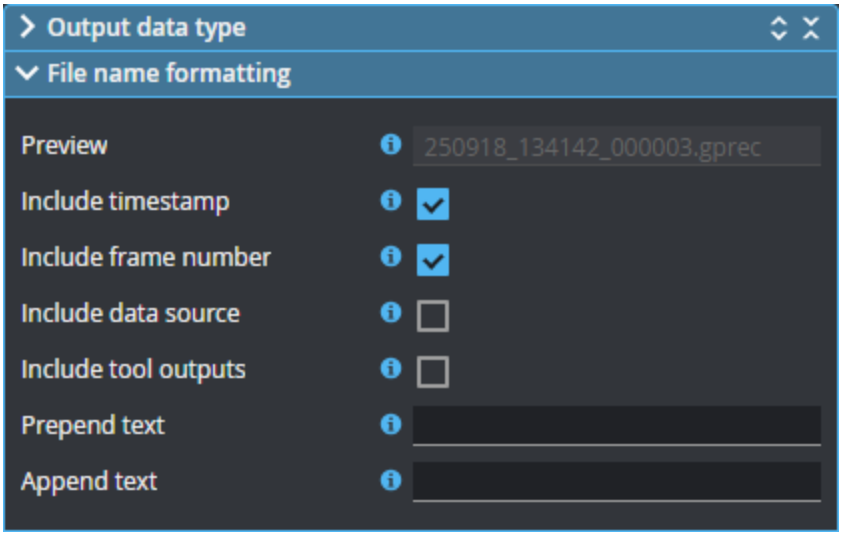


Output data type

Setting	Description
Input source	The source from which the tool takes data. The service does not support profile or measurement inputs.
Save GoPxL recording (.gprec)	Saves the current frame to a GoPxL recording file.
Preview image format	The format the tool uses when saving a preview image of the current frame. All formats are compressed; some are lossy, others lossless. One of the following: • None • JPG (lossy, 95% quality) • PNG • TIFF (16-bit lossless) • Quarter resolution TIFF (8-bit lossless)
Data type	The data types for which a preview image is stored.
Save daily log (CSV)	Saves a CVS log every 24 hours. Each time a frame is acquired, GoPxL adds information relative to that frame to this log.
Daily log period	The number of days the tool saves a log.
Save snapshot log (CSV)	Saves a CSV log for the current frame.
Save condition	The conditions that trigger saving data. For all conditions except "All measurements passed or failed" with **Save OK/NG separate folders enabled**, frames are saved in the root directory defined in service's **Storage mode** > **Path on FTP site** setting. One of the following: **All measurements passed or failed (everything)** Always saves a frame, whether measurements pass or fail, but also if you haven't

Setting	Description
	added any measurements. When this save condition is selected, you can choose whether to save frames in separate "OK" and "NG" folders by enabling **Save OK/NG separate folders**. When the option is disabled, all files are saved in the folder set in **Path on FTP** site, in the service's **Storage mode** section. **All measurements passed** **All measurements failed** Only saves a frame if all enabled measurements pass or fail, respectively. **Any measurements failed** Saves a frame if any of the enabled measures fail. **Specific measurement failed** Saves a frame when a selected measurement fails. Select the measurement in the **Measurement** drop-down. **Specific measurement range** Saves a frame only if the value of the measurement selected in **Measurement** is in the range defined by **Minimum** and **Maximum**; the range is inclusive.

You configure the format of the file names in the expandable **File name formatting** section.



File name formatting

Setting	Description
Preview	An example file name resulting from the configured formatting, such as the

Setting	Description
	following: `prependedText_250918_134142_000003_0.00_1.00_appendedText.gprec`
Include timestamp	Includes a timestamp of the following format: `ddmmyyy_hhmmss`
Include frame number	The six-digit number of the frame. This resets each time acquisition starts.

Include data source	The source selected in **Output data type** > **Input source**, using the following format:
	`{SOURCE_NAME} - {OUTPUT}`
	For example:
	`Gocator 0 - Image Output`

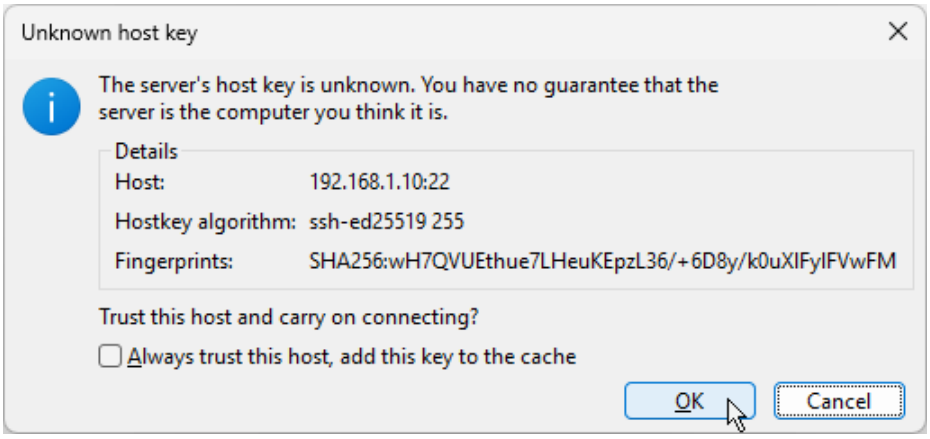
Include tool outputs	When this setting is enabled, the measurement value of the measurements you select in **Tool outputs selection** > **Measurement {n}** is added to the file names. Set **Tool outputs count** to include the measurement values of more than one measurement. Values are separated by an underscore character.
Prepend text Append text	Text added to the beginning of the file name, or at the end of the file name before the extension.

You choose the storage mode (a local SFTP server on the device or a remote FTP/SFTP server in the expandable **Storage mode** section.

Storage mode

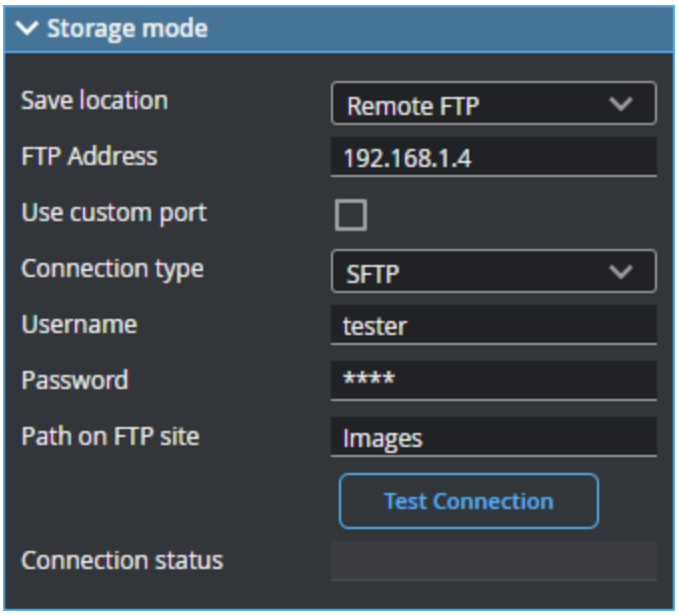
Setting	Description
Save location	Where the files are saved. The locations that are available depend on the device you are using or whether you are using GoPxL for PC. **Local** Only available on GoMax devices, GoPxL for PC, and Gocator 2D Smart Cameras. When you select Local, files are saved to an SFTP server running on the device (GoMax and Gocator 2D Smart Cameras) or the local file system (GoPxL on PC). Copy the SFTP server information (IP address of the device, port, and credentials) and connect to Gocator device's SFTP server using an FTP client application such as FileZilla or WinSCP. Note that the client application will probably display a message saying that the SFTP server's host key is unknown. Click OK to accept the connection. Typically you can also choose to always trust the host.

Setting	Description

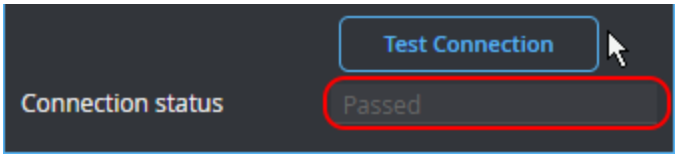


Remote FTP

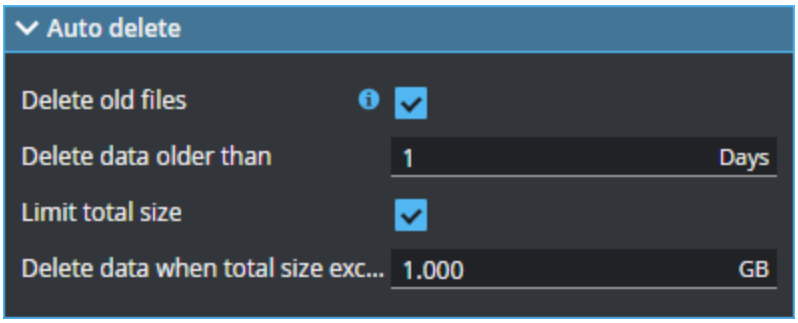
When you select Remote FTP, the data storage service saves frames to a remote FTP server. You must install and then configure the FTP server on the PC connecting to the device, and fill in the fields in the **Storage mode** section using the IP address and optional port of the server, the connection type, and the user name and password, so that GoPxL can connect to that FTP server.



You should test the credentials and connection information by clicking **Test Connection**. If the connection succeeds, GoPxL displays "Passed" in **Connection status**.



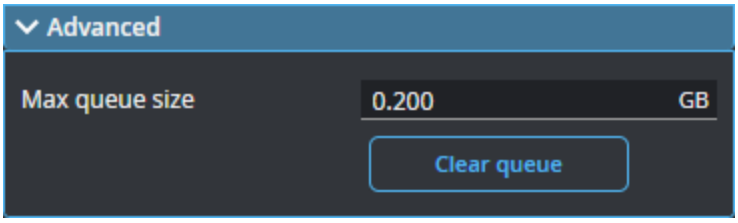
To ensure that the device receiving the frames doesn't run out of space, you can set limits in the expandable **Auto delete** section.



Storage mode

Setting	Description
Delete old files	When this setting is enabled, the service deletes files older than the number of days you specify in **Delete data older than**.
Limit total size	When this setting is enabled, the service starts deleting files to keep the amount of data stored below the value you set in **Delete data when total size exceeds**. Before sending data, the service compares the space used with this limit, and deletes older files until the amount of stored data is below the limit. For this reason, the amount of data stored may exceed the limit until the next frame save.

You set advanced settings in the expandable **Advanced** section.



Storage mode

Setting	Description
Max queue size	The maximum queue size for frames. Set this to a higher value if you need to run the device at a high frame rate to help ensure all frames are saved. Note however that the higher the value, the longer frames will continue to be saved after acquisition stops. Therefore, to avoid losing frames, you must keep the remote FTP server running, and the device saving frames powered up, until the frames in the queue have all been saved. You can clear the queue manually by clicking **Clear queue**.

Date and Time Service

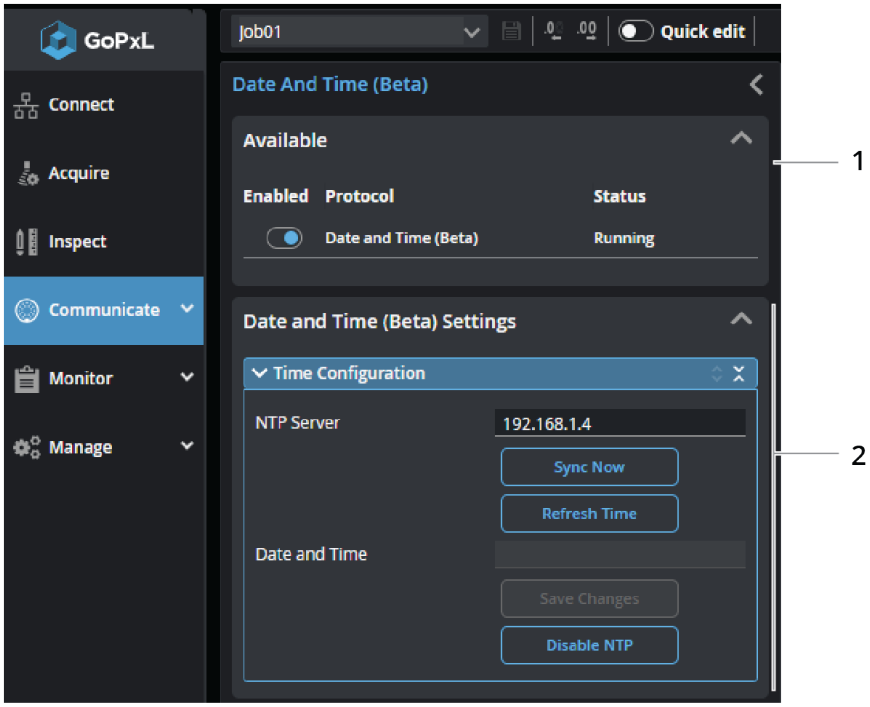
The date and time service is currently a beta feature.

When enabled, this service sets the device's date and time from an NTP server. This is useful when you use this service in conjunction with other GoPxL services or tools to include the correct date and time, for example:

- In file names of files saved using the Data Storage service, after enabling timestamp in that service; for more information, see *Data Storage Service* on page 378.
- In Script tools.

Note that typically you will set up an NTP server on the PC (the client) connecting to the device. For accurate date-time, the NTP server needs access to the internet. You may also need to configure the firewall of the client PC.

The service works on all Gocator devices and GoPxL for PC. Note that when you enable, disable, or make changes to the Date and Time service, you must restart the device (excluding GoPxL for PC).



	Element	Description
1	Available	Disables the data and time service and displays its status.
2	Date and Time Settings	Lets you configure the date and time service. For more information, see the table below.

You configure the date and time service in the expandable **Time Configuration** section.

Date and Time Settings

Setting	Description
NTP Server	The IP address of the network card of the NTP server.

Setting	Description
	Note that currently the service does not support DNS, so you must use a static IP address.
Sync Now	Connects to the NTP server and synchronizes the time.
Refresh Time	Queries the NTP server for date and time.
Date and Time	The current date and time on the device.
Save Changes	Saves changes relating to the server IP.
Disable NTP	Disables updates to the date and time.

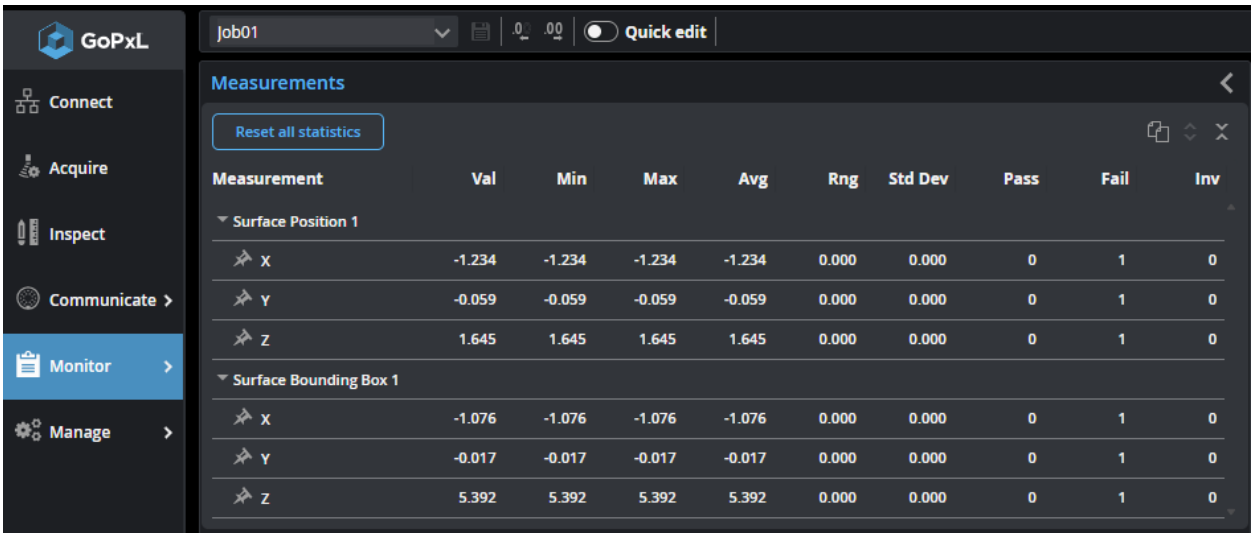
Monitor - Measurement Stats, Health & Performance

Available report pages

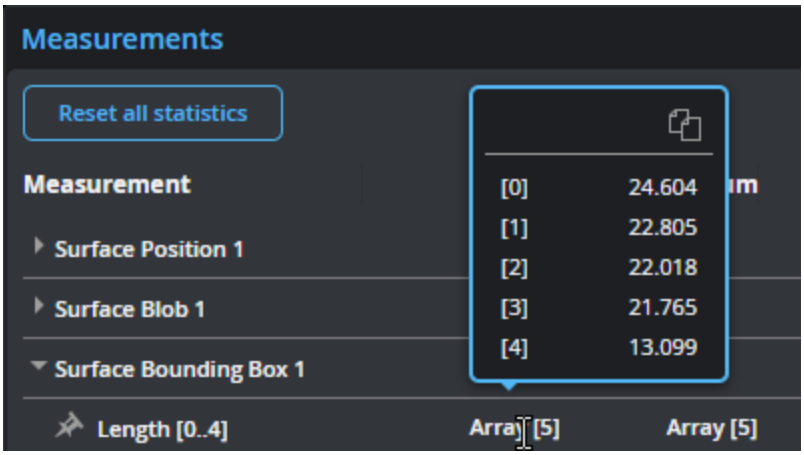
Report	Description
Measurements	The **Measurements** page provides statistics for each enabled measurement. For more information, see *Measurements* below.
Health	The **Health** page lets you monitor system and sensor health indicators, as well as various counters, such as processing drops. For more information, see *Health* on the next page.
Performance	The **Performance** page provides performance statistics. For more information, see *Performance* on page 391.

Measurements

The **Monitor** > **Measurements** page displays statistics for each measurement enabled in the **Inspect** page, grouped by the tool that contains the measurement. Values are reset when the sensor is started. To reset statistics manually, click the **Reset All Statistics** button.



For arrayed measurements, hover over "Array [n]" to see values in the array. Click the Copy icon to copy the values to the clipboard.



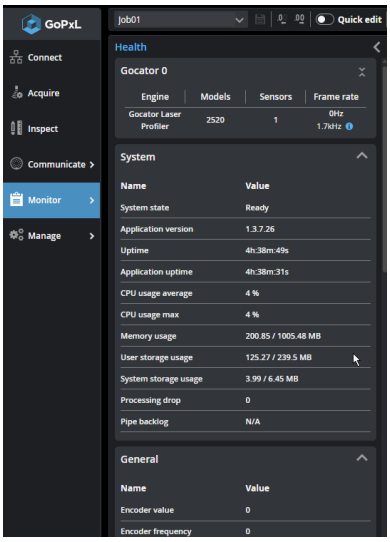
For each measurement, the following information is available:

Measurement statistics

Name	Description
Value	The most recent measurement value.
Minimum	The minimum measurement value that has been returned.
Maximum	The maximum measurement value that has been returned.
Average	The average of all measurement values collected since the sensor was started.
Range	The difference between Maximum and Minimum.
Std dev	The standard deviation of all measurement values collected since the sensor was started.
Pass count	The number of pass decisions the measurement has generated.
Fail count	The number of fail decisions the measurement has generated.
Invalid count	The number of frames that returned no valid measurement value.

Health

In the **Health** panel, you can monitor system, sensor group, and sensor health indicators, as well as various counters, such as processing drops.



The following tables list the information available in the Health panel.

> When a sensor is being accelerated (by a PC instance of GoPxL or by GoMax), the value reported in the System section comes from the accelerator. However, only the following metrics are appreciably different from the physical device when accelerated: Uptime, CPU Usage, Memory Usage, and System/User Storage Usage.

System

Name	Description
System state	Current system state indicating whether the device is scanning or not.
Application version	System software version.
Uptime	Time since the system was power-cycled or reset. (If the sensor is accelerated by a PC or a GoMax accelerator, this is the uptime of the PC or GoMax.)
Application uptime	Time since GoPxL started. (On a PC, indicates how long the PC has been running.)
CPU usage average	Average system CPU usage for the system.
CPU usage max	Maximum system CPU usage for the system.
Memory usage	Sensor memory (MB used / MB total available). When a sensor is accelerated, this displays N/A.
User storage usage	Sensor flash storage (MB used / MB total available). When a sensor is accelerated, this displays N/A.
System storage usage	Non-volatile storage available (MB used / MB total available).
Processing drop	The sum of various indicators related to processing drops including drops due to insufficient CPU and buffer overflows.
Pipe backlog	Number of bytes queued for processing.

General

Name	Description
Encoder value	Current encoder value (ticks).
Encoder frequency	Current encoder frequency (Hz).
Processing latency	Last delay from camera exposure start to when the results are ready for output.
Processing latency (Max)	Peak delay from camera exposure start to when the results are ready for output.
Scan count	Number of scans performed since system state last changed to Running.
Scanner drops	Number of sensor group (scanner) drops.
Current speed	Current scanning frame rate.
Trigger drops	Count of camera frames dropped due to excessive trigger speed.
CUDA in use	With G3 sensors accelerated by PC or GoMax, indicates whether GoPxL is using CUDA.

For each sensor in a system, GoPxL displays a table in the expandable **Sensors** section listing the health indicators for that sensor.

Sensor {n} health

Name	Description
Master connected	Whether the sensor is connected to a Master sensor networking hub.
Laser safety	Whether Laser Safety is enabled.
Sensor uptime	Time since the sensor was power-cycled or reset.
CPU usage average	Average system CPU usage for the sensor.
CPU usage max	Maximum system CPU usage for the sensor.
Ethernet receive rate	Rate at which data is being received by the sensor over Ethernet (Mbps).
Ethernet send rate	Rate at which data is being sent to the sensor over Ethernet (Mbps).
Ethernet link speed	Speed of the Ethernet link (Mbps).
Memory usage	Sensor memory usage (MB used / MB total available).
User storage usage	Sensor flash storage usage (MB used / MB total available).
System storage usage	Non-volatile storage available (MB used / MB total available).
Max spot count	Maximum available spots, across all cameras. (G2)
Total exposure time	Total amount of time the laser light (G2), projector (G3), or light source (G5) has been turned on to make scans.
Internal temperature	Internal sensor temperature (degrees Celsius).
Laser driver temperature	Laser driver control temperature (degrees Celsius). (G2)
Light driver temperature	Light driver control temperature (degrees Celsius). (G4 / G5)
LED PCB temperature	LED PCB temperature (degrees Celsius). (G4 / G5)
External thermistor temp.	External thermistor temperature (degrees Celsius). (G4 / G5)
Projector illumination	Projector temperature (degrees Celsius) (G3).

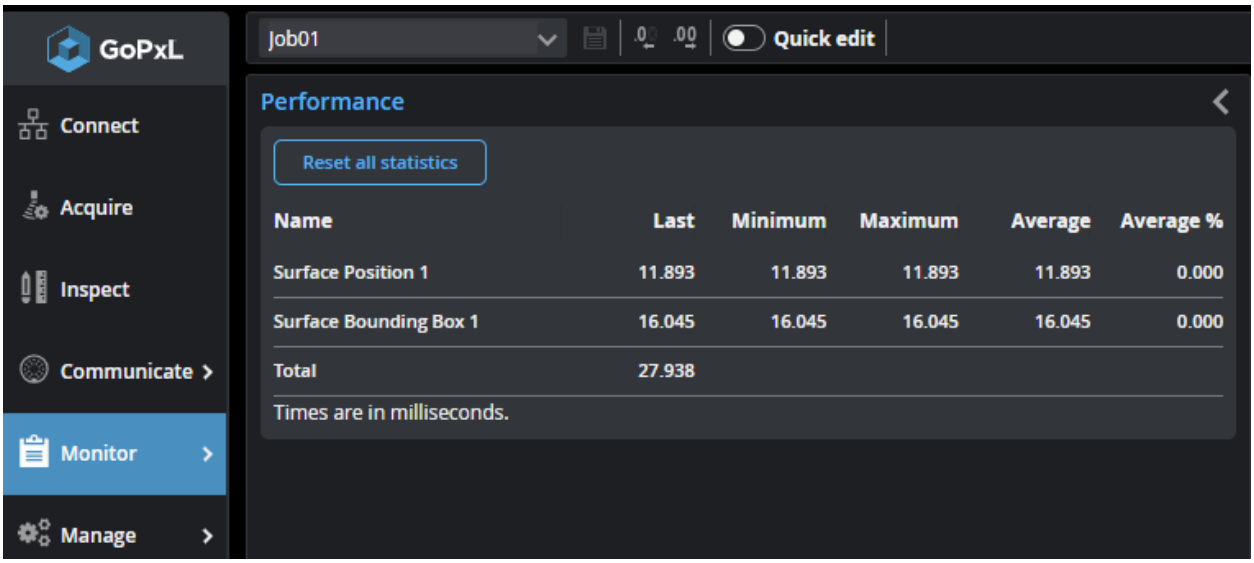
Name	Description
temp	
Trigger drops	Count of camera frames dropped due to excessive trigger speed.
Camera buffer drops	Count of camera buffer drops due to excessive trigger speed. For G3, the buffer drops are shown for each camera.
Digital Output {n} drops	Count of digital output drops because the output signal could not be generated.
Digital Output {n} high count	Number of times the digital output line was set to high.
Digital Output {n} low count	Number of times the digital output line was set to low.

Control

Name	Description
Gocator control drop	The number of Gocator protocol drops.

Performance

The **Monitor** > **Performance** page displays performance statistics (execution time in milliseconds) for each tool added in the **Inspect** page.



For each tool, the following information is available:

Performance Statistics

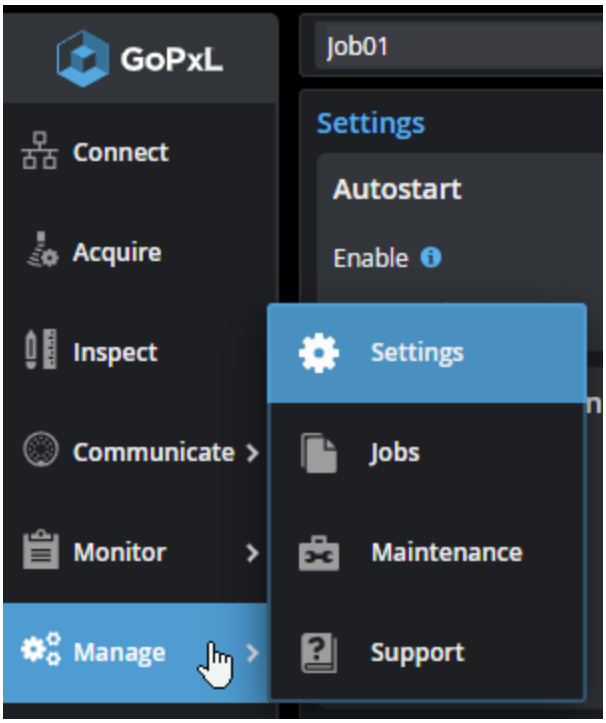
Name	Description
Last	The latest execution time of the tool.
Minimum	The minimum execution time of the tool.
Maximum	The maximum execution time of the tool.

Name	Description
Average	The average execution time of the tool.
Average %	The average percentage of the execution time of the tool.

Manage - Sensor Management and Maintenance

You use the pages in the **Manage** category to perform networking, management, and maintenance tasks (such as software updates and sensor factory restores).

To access a specific management page, hover the mouse pointer over **Manage** and choose the required page.



Manage Page Name	Description
Settings	The **Settings** page lets you configure the network settings of a sensor. For more information, see *Settings* below.
Jobs	The **Jobs** page lets you create, delete, and manage jobs. For more information, see *Jobs* on page 396.
Maintenance	The **Maintenance** page lets you upgrade a device's software, perform a backup or restore of a device, perform a factory restore of a device, or restart it. For more information, see *Maintenance* on page 399.
Support	The **Support** page provides software version information and lets you download a support file for troubleshooting; it also lets you upload a support file to GoPxL. You can also access and download the GoPxL documentation, download the SDK, and go to LMI's online support page. For more information, see *Support* on page 401.

Settings

The **Settings** page under **Manage** provides autostart and administrator login settings.

Autostart

If your application requires maintaining uninterrupted scanning, you can configure GoPxL to automatically restart sensors that are running after various events, by enabling **Autostart**. Unaccelerated single sensors and sensors grouped with a controlling sensor, as well as sensors accelerated by GoMax or a PC instance of GoPxL, will automatically start scanning in the situations listed below. The setting persists after restarts, and is a global setting (that is, not saved in a job). Performing a factory restore disables autostart.

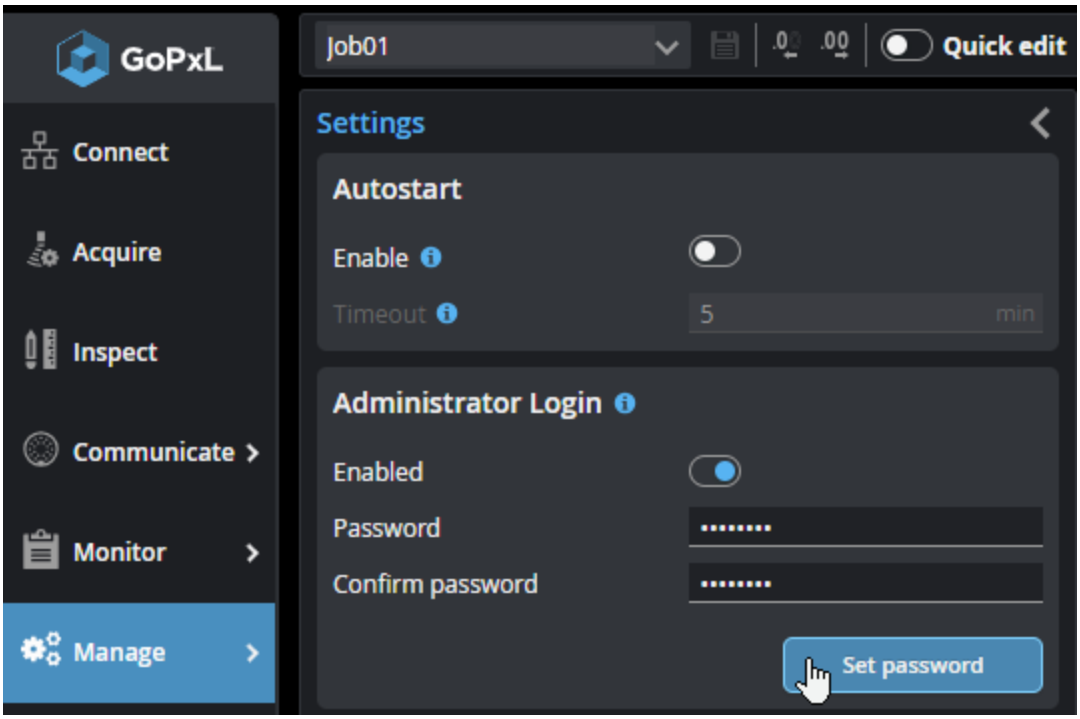
- "Launch" events:
  - Power cycle
  - Restarting a GoMax unit or relaunching a PC instance of GoPxL that is accelerating sensors
  - Restoring a running sensor with a backup in which Autostart is enabled
- Network disconnections with accelerated sensors (such as when a network cable is disconnected or a switch is power cycled)

> When a sensor is accelerated by GoMax or GoPxL on PC, only enable **Autostart** on the accelerating device. Similarly, if a sensor is grouped with other sensors, only enable **Autostart** on the host sensor (the sensor to which you connected other sensors in GoPxL). Otherwise, timing issues can result in.

Administrator Login

You can prevent unauthorized access to sensors by setting an administrator password. When a password is set, GoPxL displays a password field when the interface loads. To continue you must provide the password. In a password-protected system, there are no limitations in terms of functionality.

You set the administrator password on the **Settings** page:

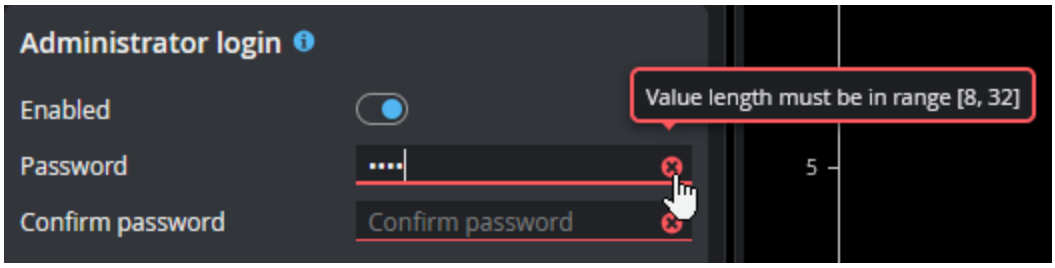


⚠	On a sensor, if you forget the password or want to completely clear a password, you must perform a factory restore. You can do this in GoPxL from **Manage** > **Maintenance** (see *Factory Restore* on page 401) or from the Discovery tool (see *GoPxL Discovery Tool* on page 473). When GoPxL is running on a PC, in both you must delete the password-protected instance and create a new instance.

Note the following:

- Passwords must be 8 to 32 characters long.
- All characters are accepted.
- The password can't be empty.
- There are otherwise no constraints or checks on passwords (that is, no minimum number of special characters, no combination of uppercase / lowercase, and so on).
- After setting a password, the password is stored even if administrator login is disabled. If you re-enable the password, the previously set password is active.
- You don't need to know the current password to change the password, so anyone with access to a logged-in sensor can change the password.

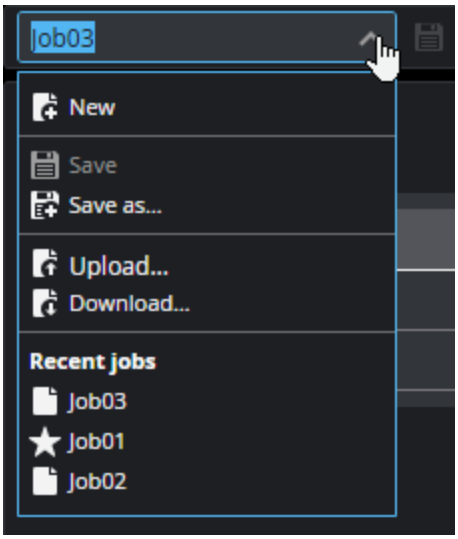
- If the password is not accepted by GoPxL, hover over the red "x" icon ( ) for more information.



Jobs

The **Jobs** page lets you manage jobs. The

	Element	Description
1	Job-related operations on global toolbar	Job menu and Save button. Use the job menu to create new jobs, save jobs, load recent jobs, and also upload or download jobs between the PC and the sensor. You can rename the currently loaded job from the job name field, simply by changing it and pressing enter or clicking outside the field.

	Element	Description
		 When a job has unsaved changes, it is displayed in italics with an asterisk. Click the Save button in the global toolbar to save it. A star icon indicates the job that has been set as the default.
2	Job-related operations	These controls let you load or reload jobs, create new jobs, and upload or download jobs between the PC and the sensor. For explanations of the job related operations, see *Job-related operations* below.
3	Job list	The list of available jobs. You can rename a job by double-clicking it in the list. When you move the mouse pointer over a job in the list, additional tools are displayed that let you rename the job, duplicate it, or delete it. When a job has unsaved changes, it is displayed in italics with an asterisk. Click the Save button in the global toolbar to save it.
4	Default job indicator	Clicking the default job indicator ( ) sets the job as the default, which will load when the sensor starts.
5	Loaded job indicator	The loaded job indicator ( ) shows which job is currently loaded.

The following table describes the job-related operations.

Job-related operations

Operation	Description
	Loads or reloads the job that is currently selected in the job list.
	Creates a new job called "untitled {n}" and loads it.
	Saves the current job. (Disabled if the job contains no changes.)
	Uploads or downloads a job between GoPxL and the client computer. Job files have a .gpjob extension.

Operation	Description
	Lets you edit the name of the job under the mouse pointer. You can also double-click on the job's name to stat editing it.
	Duplicates the job under the mouse pointer.
	Deletes the job under the mouse pointer.

Characters Allowed in Job Names

When setting a job's name, you can use alphanumeric characters (numbers and letters), including accented characters in languages other than English and characters in languages such as Chinese and Japanese. Some special characters are also allowed.

The following special characters are also supported:

Allowed Special Characters

space	& (ampersand)
- (hyphen)	( ) { } [ ] (parentheses, brackets, and braces)
_ (underscore)	+ (plus)
. (period)	, (comma)
' (single quote)	; (semi-colon)
@ (at)	= (equals)
# (pound)	

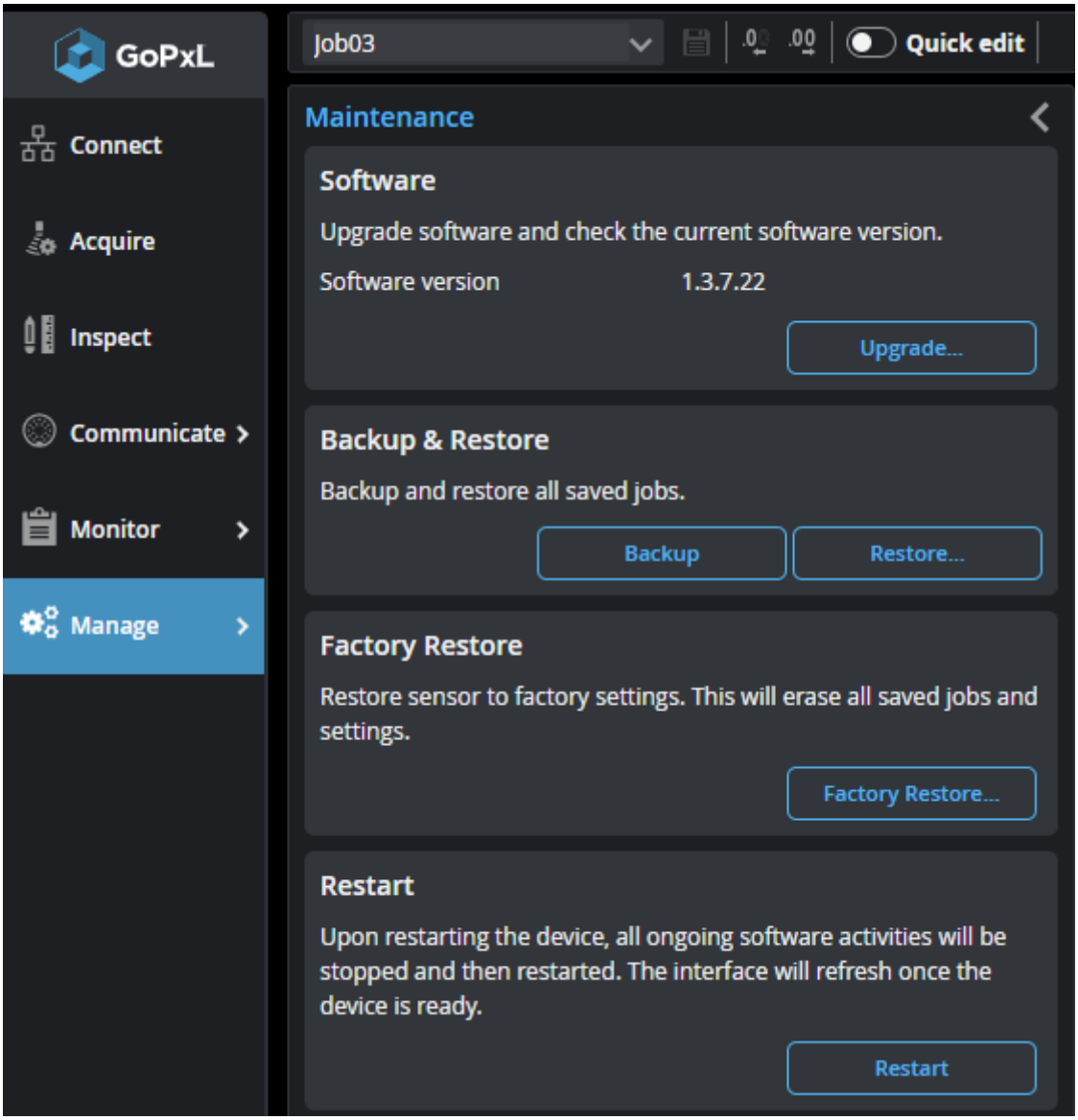
The following special characters are *not* supported:

Disallowed Special Characters

< (less than)	| (vertical bar or pipe)
> (greater than)	? (question mark)
: (colon)	* (asterisk)
" (double quote)	% (percent)
/ (forward slash)	~ (tilde)
\ (backslash)	$ (dollar)

Maintenance

You use the **Maintenance** page in the **Manage** category to perform basic maintenance.

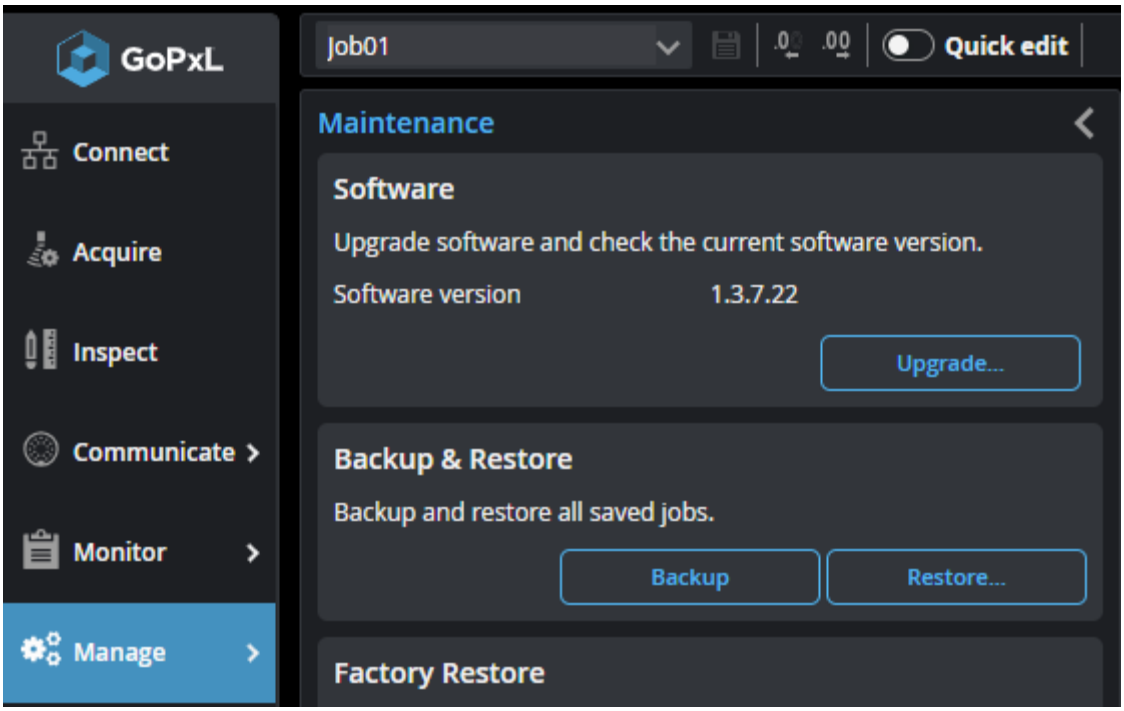


The **Maintenance** panel lets you do the following:

- **Upgrade...**: Upgrade the device's software. For more information on upgrading a sensor's software, see *Software Upgrade* on the next page.
- **Backup** / **Restore...**: Back up and restore all saved jobs and recorded data. For more information, see *Backup and Restore* on the next page.
- **Factory Restore...**: Restore GoPxL to factory defaults. This erases *all* saved jobs and settings, as well as files created by tools.
- **Restart**: Restart the sensor.

Software Upgrade

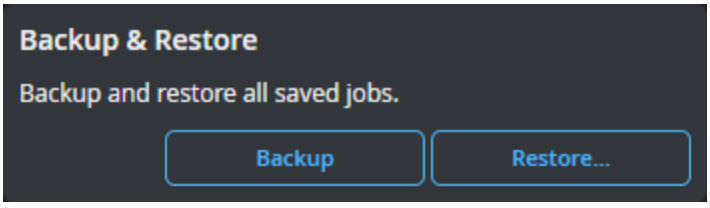
LMI recommends routinely updating software to ensure that sensors always have the latest features and fixes.



Backup and Restore

Sensor backups let you keep the state of a sensor or system in a safe location, and restore a sensor to a previous state in case you have made unwanted changes. You create backups in the **Maintenance** page. Backup files contain the following:

- All jobs
- Current recorded data
- Layout set up on the **Connect** page
- Alignment performed on the **Acquire** page
- Other configurations such as scan settings and control settings



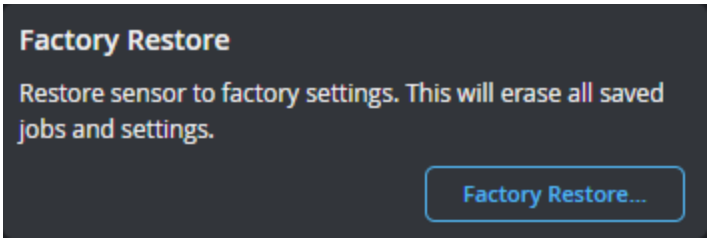
	You should always create a sensor backup file in the unlikely event that a sensor fails and a replacement sensor is needed. If this happens, the new sensor can be restored with the backup file.

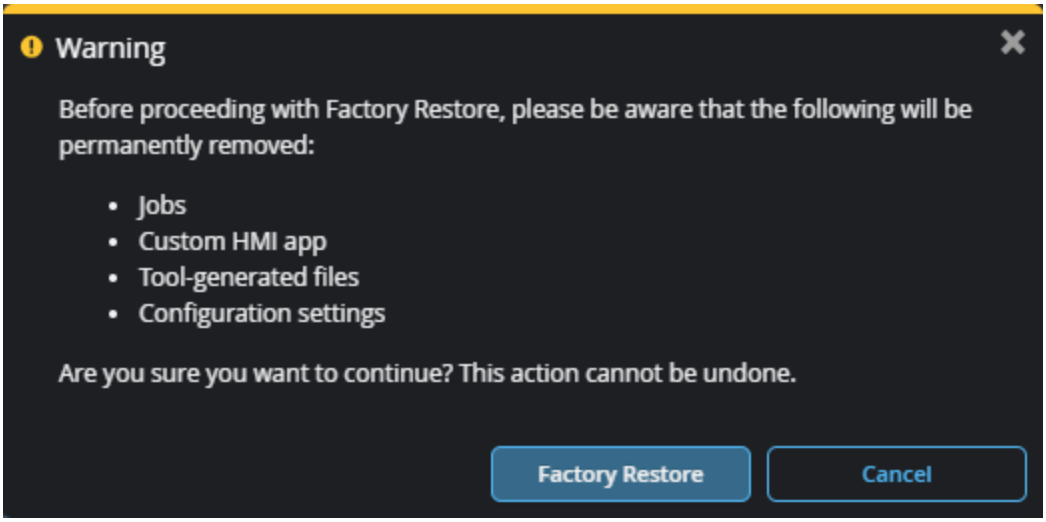
When you create a backup, your browser may ask if you want to allow downloading the file.

If you are running a sensor through a PC instance of GoPxL, performing backup, restore, and factory restore operations only affects the PC instance of GoPxL: jobs and recorded data saved on the sensor are not backed up and are not overwritten by a restore.

Factory Restore

You can perform a factory restore to quickly remove all jobs and configurations, as well as the files created by tools (such as Surface Pattern Matching, Surface Track, and so on). This also removes Custom HMI apps.



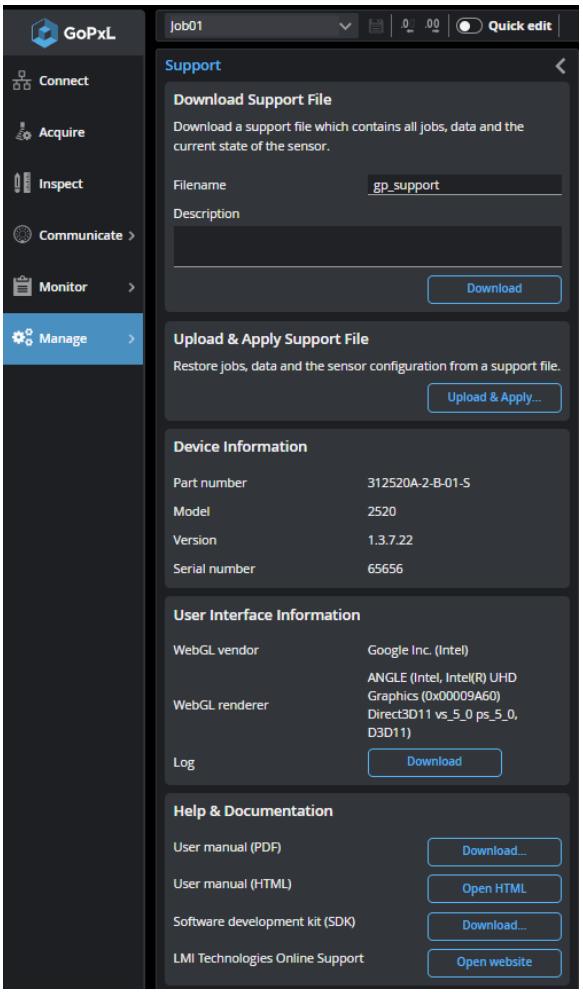


You should create a backup file of your sensor in case of unwanted changes and in the unlikely event that a sensor fails and a replacement sensor is needed. If this happens, the new sensor can be restored with the backup file.

Support

The **Support** page in the **Manage** category lets you download or upload a support file and get software version information. Support files are useful for supporting development and diagnosing issues.

You can also access the user manual, download the SDK, and go to online support from here. Additionally, you can download log; you would normally only need to download the log if told to do so by LMI's support team.



Support files contain everything in backup files, but also contain *additional diagnostic information*:

- All jobs
- Current recorded data
- Layout set up on the **Connect** page
- Alignment performed on the **Acquire** page
- Other configurations such as scan settings and control settings

Running GoPxL on a Windows PC

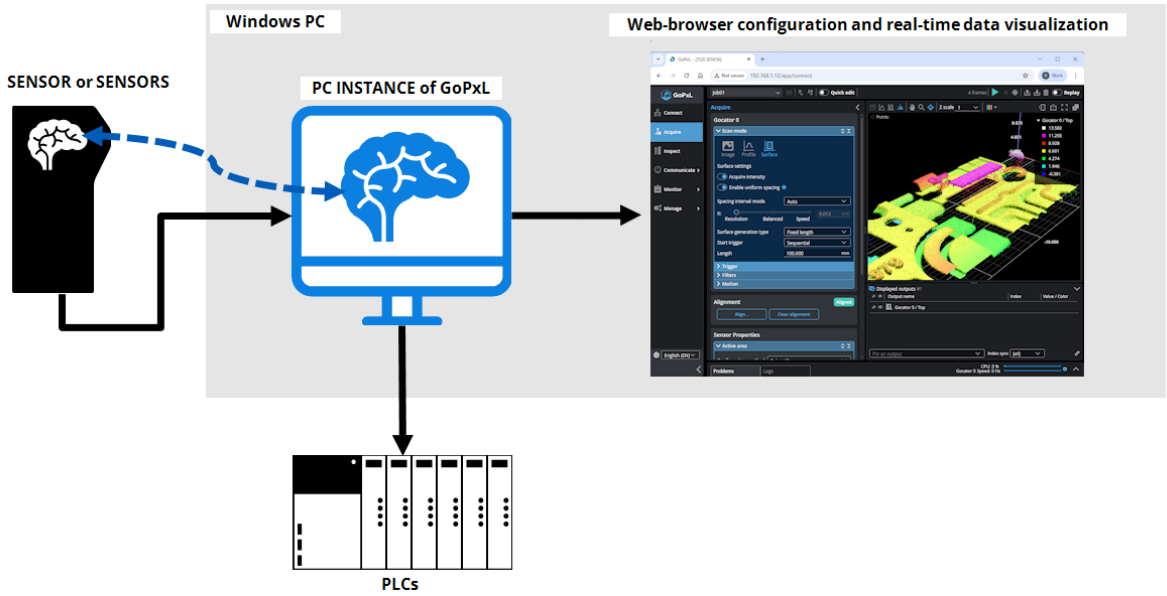
GoPxL is the software you use to configure Gocator sensors. Typically, you connect directly to a sensor's IP address via a web browser, and use the GoPxL software running on the sensor to configure and run the sensor. (By default, a sensor's IP address is 192.168.1.10. If you can't find a sensor, see *GoPxL Discovery Tool* on page 473.)

You can also run GoPxL on a Windows PC. We call GoPxL running on a PC a "PC (or local) instance of GoPxL."

If your Windows account has Administrator rights, you can install a PC instance of GoPxL as a Windows service. For more information, see *Running GoPxL Manager in Administrator Mode* on page 407.

A PC instance of GoPxL lets you do the following:

- Accelerate sensors: For applications where performance is critical, you should run a sensor through a PC instance of GoPxL. After you connect a sensor to GoPxL running on a PC, the instance automatically starts accelerating it, handling the processing of the scan data, rather than the sensor. This improves system performance, potentially letting you increase scan rates or add more tools. For more information, see *Accelerating a Sensor* on page 414.



- Load previously recorded scan data to examine it or to configure measurement tools offline, without a sensor, using that data. For more information, see *Loading Scan Data* on page 419.

For information on the ports a PC instance uses (for example, in order to ensure ports are not blocked over your network), see *Required Ports* on page 70.

If your Windows account has Administrator rights, you can install a PC instance of GoPxL as a Windows service. A PC instance installed (and running) as a service can run a sensor without the need for a user to be logged into Windows. For more information, see *Instance Settings* on page 411

The user interface of GoPxL running on a sensor and GoPxL running on a PC is almost identical. When running a sensor through a PC instance of GoPxL, you can't upgrade or restart the sensor. You must remove the sensor from the PC instance and then connect to the sensor directly to upgrade the sensor.

Job files you create in a PC instance of GoPxL are separate from any jobs that are on the sensor. Be sure to back up jobs in a PC instance if ever you need to run them directly on the sensor.

System Requirements

The following are the minimum system requirements for using GoPxL on a Windows PC:

- Processor: x86-64 (Core i3 or equivalent)
- RAM: 8 GB
- Hard drive: 64 GB
- Operating system: Windows 10 or 11 (64-bit only)
- Display: 1366 x 768

To accelerate more sensors or to run the system at higher speeds, use a computer with more system resources.

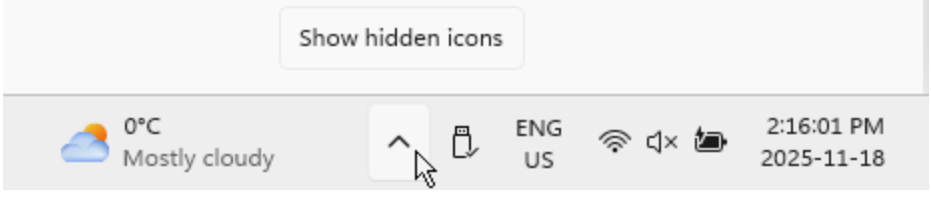
If you choose to run GoPxL Anomaly Detector on a PC instance of GoPxL, the PC must have a graphics card with a recent Nvidia GPU that supports Nvidia GeForce Game Ready Driver version 528.33 or higher, and a compute capability of 7.5 or higher. To verify the driver version of your video card, launch the NVIDIA Control Panel, and in the main panel of the application, make note of the driver version. To exceed the capabilities of GoMax ORIN in terms of training times and inference, an RTX 4070 or higher is recommended.

Launching GoPxL on a Windows PC

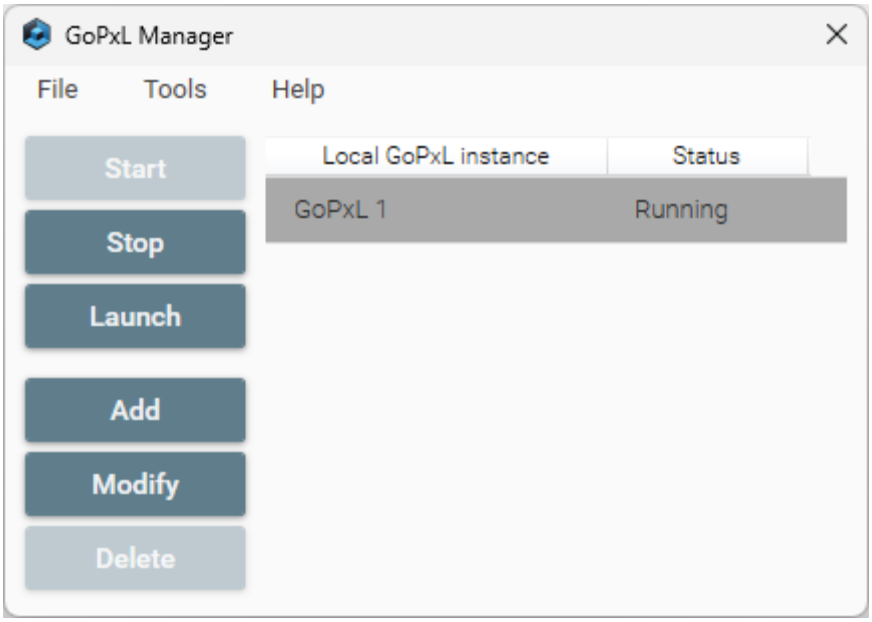
The first time you launch GoPxL for Windows, a GoPxL instance is automatically created and started, and the GoPxL interface opens in your default browser.

GoPxL Manager starts minimized in the Windows system tray.

Click the "Show hidden icons" button to find GoPxL Manager.



You use the GoPxL Manager application (which is started when you start GoPxL.exe) to add, remove, and configure instances. The following assumes you have connected a sensor to a network and that you can access its web interface.



The GoPxL Manager application. An instance is automatically added the first time you launch GoPxL.

To launch GoPxL on a PC

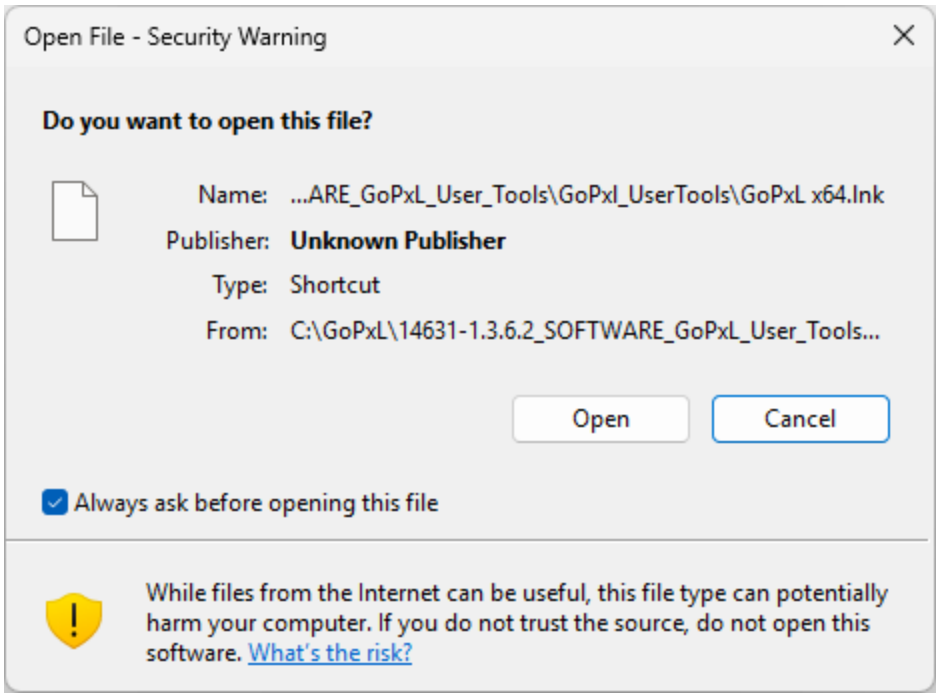
1. Download and unpack the GoPxL Utilities package (`14631-x.x.x.x_SOFTWARE_GoPxL_Utilities.exe`) by double-clicking on it.

   If you are using GoPxL on PC to accelerate a sensor, the version of the package must match the version of GoPxL on the sensor.

2. From the `GoPxL` folder, launch `GoPxL x64`.

   GoPxL Manager creates a local, PC-based instance of GoPxL and launches it in a new tab in your default browser, using IP address 127.0.0.1.

   Windows may display a security warning asking if you want to launch GoPxL. Click **Open** to continue.



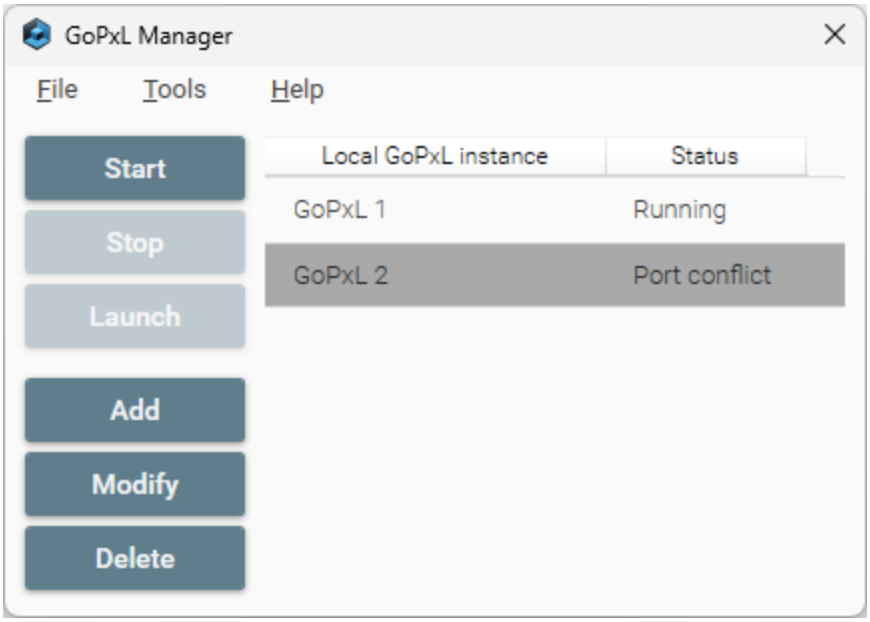
Additionally, your firewall may ask if you want to allow the GoPxL Service Application to communicate over your network. To avoid issues, LMI recommends allowing communication over both Public and Private networks.

Note that GoPxL Manager starts minimized in the tray area.

At this point, you can do one of the following:

- Add one or more sensors to the PC instance to accelerate the sensors. For information on adding a sensor to a sensor group, see *Accelerating a Sensor* on page 414.
- Open scan data someone has previously recorded (in a .gprec, .gpbak, or .gpsup file) and examine the data or use it to configure measurement tools without taking your sensor system offline. For more information, see *Loading Scan Data* on page 419. (For information on the file formats used with GoPxL, see *File Formats* on page 83.)

All ports used by PC instances must be unique. In some situations, it is possible to accidentally set the ports of two or more instances to the same port.



To correct this, select one of the instances, click **Modify**, and change the duplicated port to a different one. You can also delete any instances that use duplicate ports.

> You can select multiple instances in GoPxL Manager and perform most of the actions on the selected instances.
>
> Press Ctrl + Left click to toggle whether an additional instance is selected. Press Shift + Left click to select a range of instances (from the currently selected instance to the one you Shift + Left click).

Running GoPxL Manager in Administrator Mode

To perform certain operations, you need to start GoPxL Manager in Administrator mode.

- Change certain applications settings (for more information, see *Application Settings* on the next page).
- Install or uninstall a PC instance as a service, or delete it.
- Install licensing for tools like GoPxL Anomaly Detector.

> Before switching to Administrator mode, you should stop all running PC instances of GoPxL.

To run GoPxL Manager in Administrator mode, click **Run As Admin** either in the GoPxL Manager Settings dialog (see *Application Settings* on the next page), or in the Modify GoPxL Instance Configuration dialog (see *Instance Settings* on page 411).

> After you have finished making any of the changes listed above, you should fully exit GoPxL Manager to avoid permission issues with your instances. For more information, see *Closing GoPxL Manager and Stopping All PC Instances* on the next page.

Closing GoPxL Manager and Stopping All PC Instances

After you have finished making any of the changes listed in *Running GoPxL Manager in Administrator Mode* on the previous page, you should fully exit GoPxL Manager to leave Administrator mode to avoid permission issues with your instances as described below. You use the same steps if you just need to exit GoPxL Manager and stop all PC instances. Note that clicking the X button at the top right in GoPxL Manager only hides the utility's main window.

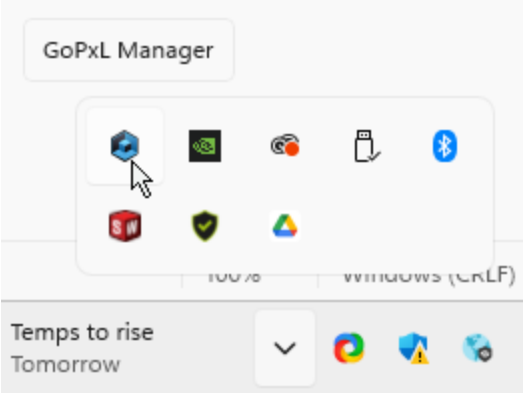
To close GoPxL Manager and all PC instances

1. Look for the small GoPxL Manager icon ( ) in the taskbar's notification area.

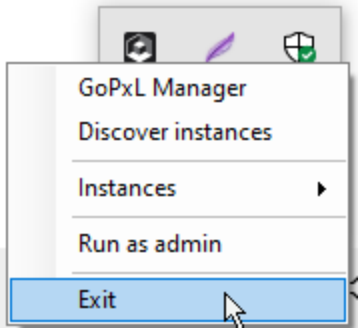
   This is normally at the bottom of the screen, to the right.

2. If you don't see the GoPxL Manager icon, click the "Show hidden icons" arrow.

   A menu containing the hidden icons appears.

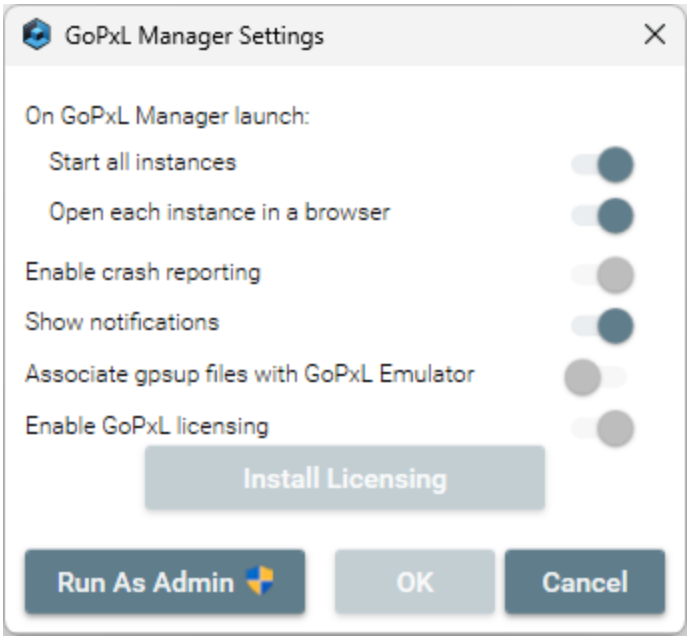


3. Right-click the GoPxL Manager icon.
4. In the menu that appears, click Exit.

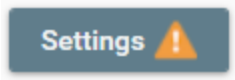


Application Settings

The following application settings are available. To open the settings, click **Tools** > **Settings** at the top of the application.



Parameters

Parameter	Description
On GoPxL Manager launch	**Start all instances** When enabled, all instances not installed as a service will automatically start when GoPxL Manager starts. **Open each instance in a browser** Each instance opens in the default browser. (Instances installed as a service will start when the computer is started.)
Enable crash reporting	If a PC instance of GoPxL crashes, a report will be generated. This can be useful if you need to troubleshoot problems with LMI support. LMI recommends leaving crash reporting enabled. If you choose to disable crash reporting, an icon on the **Settings** button will indicate that it is disabled.  To change this setting, you must be running GoPxL Manager as an administrator. For more information on using GoPxL Manager in Administrator mode, see *Running GoPxL Manager in Administrator Mode* on page 407.
Show notifications	GoPxL Manager will show notifications in the tray area, provided that notifications are enabled in Windows.
Associate gpsup files with GoPxL Emulator	Double-clicking a support file (.gpsup) launches the support file in the emulator. To enable this, you must first launch GoPxL Manager in admin mode by clicking **Run as Admin**.
Enable GoPxL licensing	In order to use tools such as GoPxL Anomaly Detector with live scan data, you

Parameter	Description
	must purchase a license and dongle. Then in GoPxL Manager, you must enable licensing and install the licensing software if necessary. For more information see *Installing Licensing Software* below. To change this setting, you must be running GoPxL Manager as an administrator. For more information on using GoPxL Manager in administrator mode, see *Running GoPxL Manager in Administrator Mode* on page 407.

Installing Licensing Software

If you have purchased a license and dongle for using tools such as GoPxL Anomaly Detector, you must install the licensing software, Wibu-Systems CodeMeter.

To install the licensing software

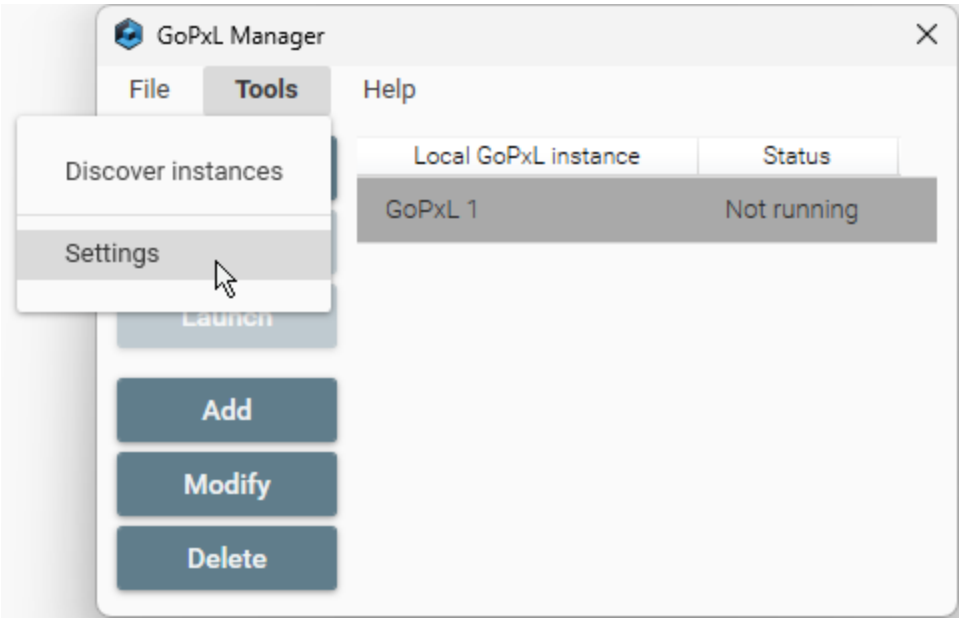
1. If it is not already running, start `GoPxL.exe` and open GoPxL Manager.

   For more information, see *Launching GoPxL on a Windows PC* on page 404.

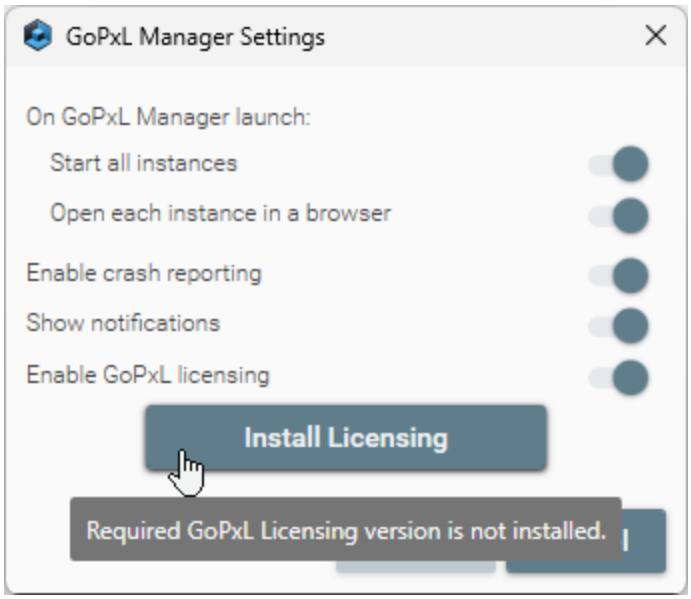
2. Restart GoPxL Manager in Administrator mode.

   For more information, see *Running GoPxL Manager in Administrator Mode* on page 407.

3. Click **Open** > **Settings**.



4. In GoPxL Manager Settings, if licensing isn't already enabled enable it.
5. Click **Install Licensing**.

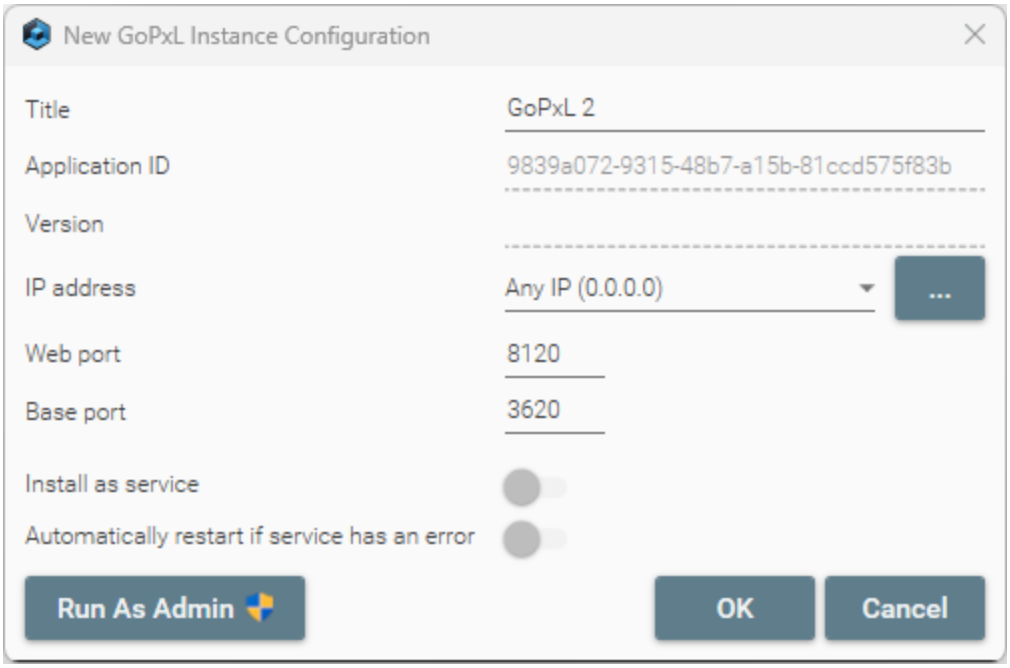


6. Follow the instructions in the CodeMeter installer.

   In the Custom Setup step, leave the selection of features as the default.

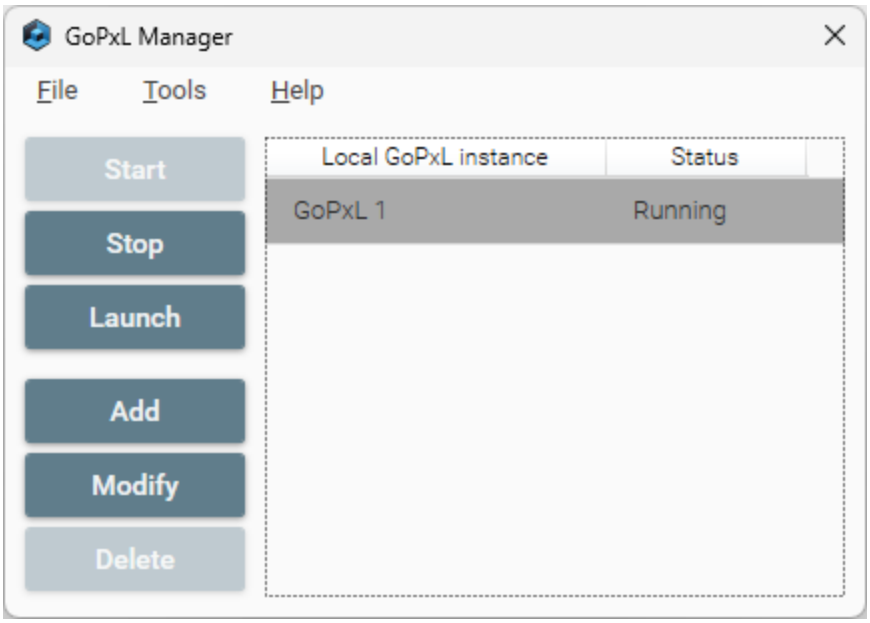
Instance Settings

When adding or modifying an instance, you can change various settings (see *Instance Configuration* on the next page).



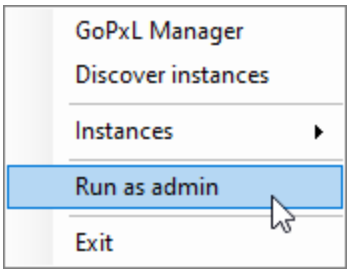
To add an instance, click **Add** and configure it if needed, and then click OK.

To modify an instance, select it in the list of instances and click **Modify**. To make changes to an instance, it must be stopped; select the instance and click **Stop** if needed.



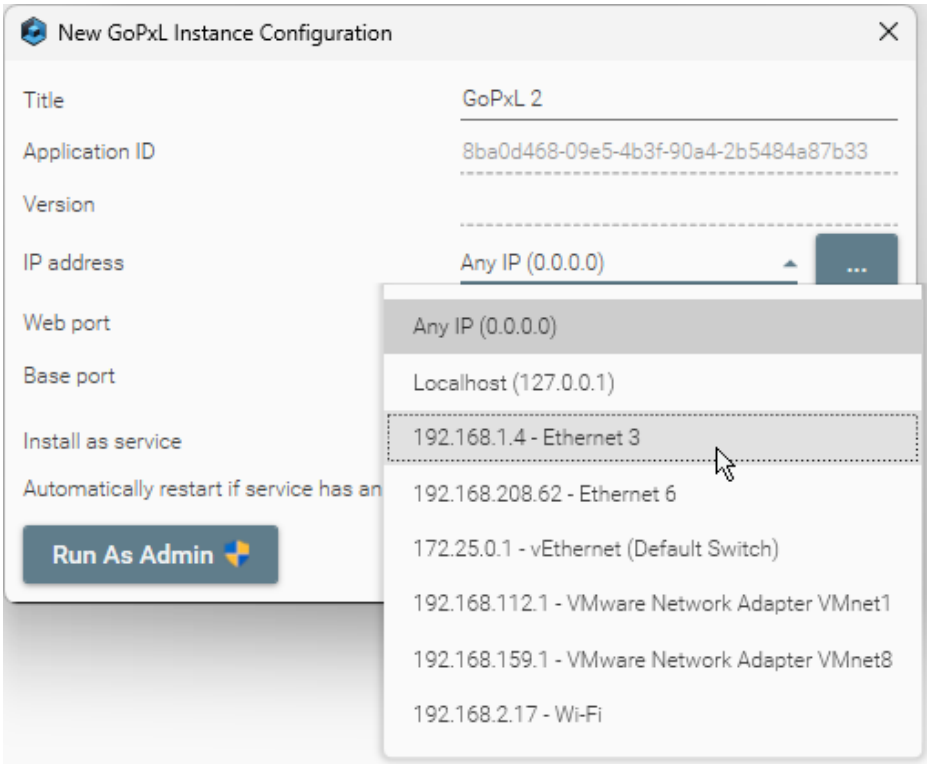
GoPxL Manager with one instance running.

To change the settings of a service instance or delete it, you must be running GoPxL Manager as an administrator. You can do this by clicking **Run As Admin** in the Instance Configuration dialog (see above), which restarts GoPxL Manager with administrator rights. You can also run right-click its icon in the Windows taskbar and click **Run as admin**.



Instance Configuration

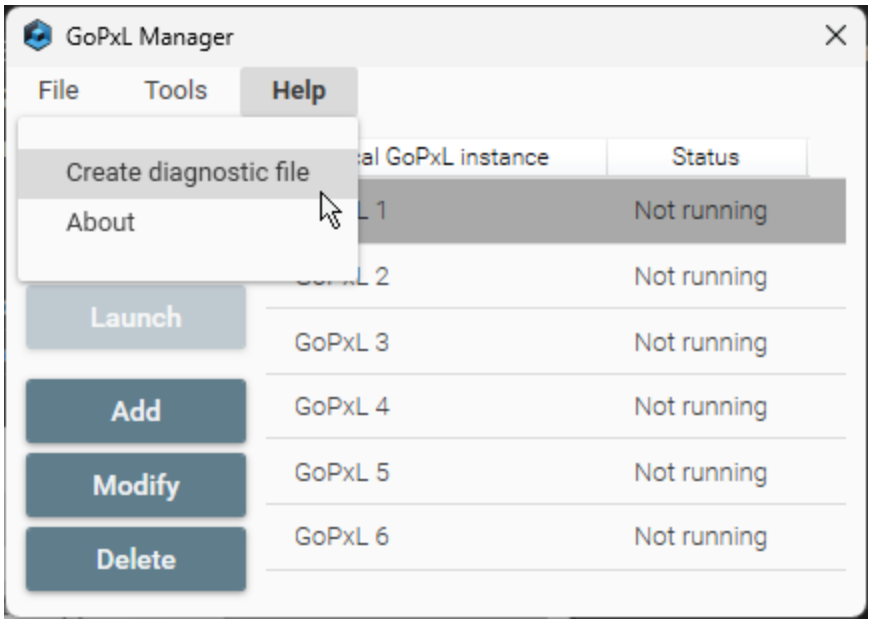
Parameter	Description
Title	The name of the instance in GoPxL Manager.
Application ID	The ID of the GoPxL instance. Not editable.
Version	The version of the GoPxL instance. Not editable.
IP address	The IP address of the instance (that is, the IP the instance is bound to). The drop-down lists the available interfaces on your PC.

Parameter	Description
	 Using the address you provide followed by the instance's Web port in a web browser connects to GoPxL. You can find the Web port of an instance in the Instance Configuration dialog or in the Discovery tool by clicking **Discover Instances**. Choose a specific Ethernet or Wi-Fi interface to bind the instance to that interface only. The instance will be visible to other PCs on your network at that IP address, provided that this is not blocked by a firewall. Choose "Any IP (0.0.0.0)" to bind the instance to *any* available interfaces. You can then connect to the sensor using any of the interfaces available on the PC (Ethernet, Wi-Fi, localhost, and so on). Except for the localhost address, the instance will be visible to other PCs on your network at those IP addresses, provided that this is not blocked by a firewall. Note that when you launch the GoPxL interface of an instance set to "Any IP" from GoPxL Manager, it opens at http://127.0.0.1/ followed by the Web port. Also, in the Discovery tool, the first IP found by the tool is listed in the table. If you are using PROFINET, you must set this to a specific interface: PROFINET does not support "Any IP" or Localhost. Choose Localhost to bind the instance to http://127.0.0.1/. The instance will only be visible to the PC it's running on.
Web port	The main ports the instance uses. These must be unique for each instance.

Parameter	Description
Base port	**Web port** The port you use when connecting to the instance, for example, to `http://127.0.0.1:8120/`. **Control port** The port you use with the GoPxL SDK and REST API (see *GoPxL SDK and REST API* on page 449).
Install as a service	A PC instance installed (and running) as a service can run a sensor without the need for a user to be logged into Windows.
Automatically restart if service has an error	If an instance of GoPxL is configured to be a service and it has an error, it will restart automatically.

Creating a Diagnostic File

To help troubleshoot issues related to PC instances of GoPxL, you can create a diagnostic file containing logs and instance files. You usually only need to do this when asked to do so by LMI support.



After choosing Create diagnostic file, Windows will prompt you for a location.

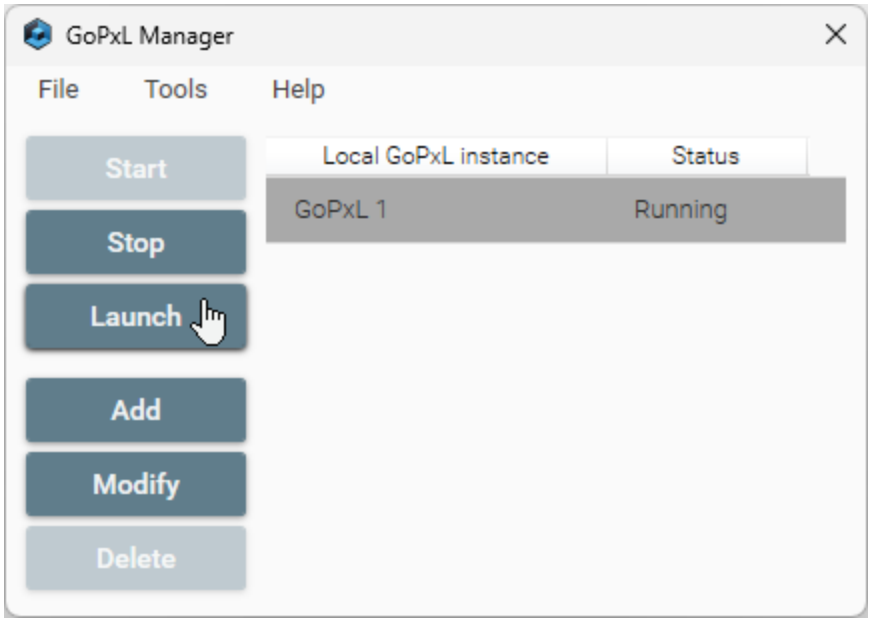
Accelerating a Sensor

You accelerate a sensor (or group of sensors) by connecting it to a PC instance of GoPxL.

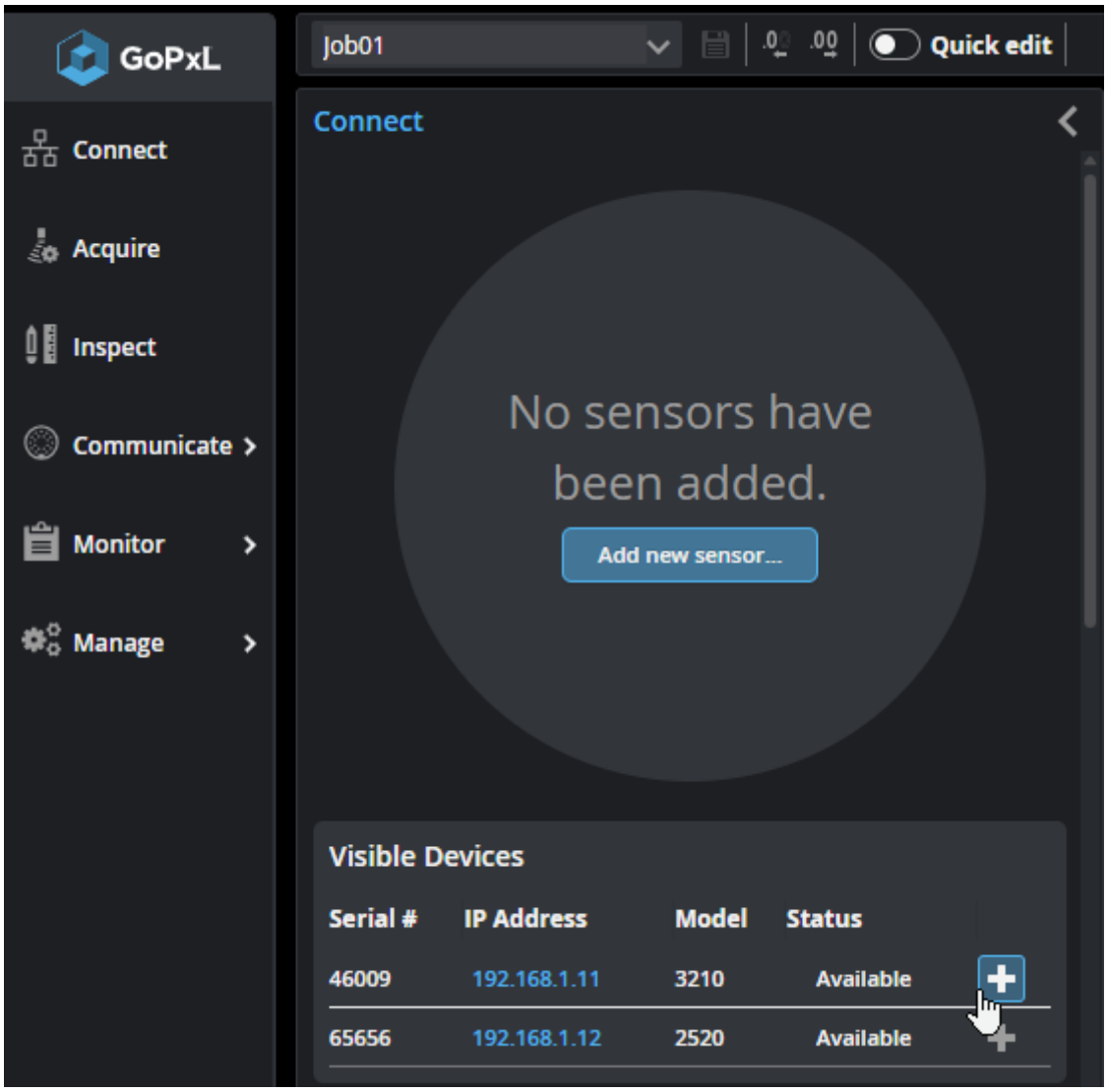
To accelerate a sensor

1. In a web browser, go to the IP address of the PC instance.

You can also launch the PC instance from GoPxL Manager.

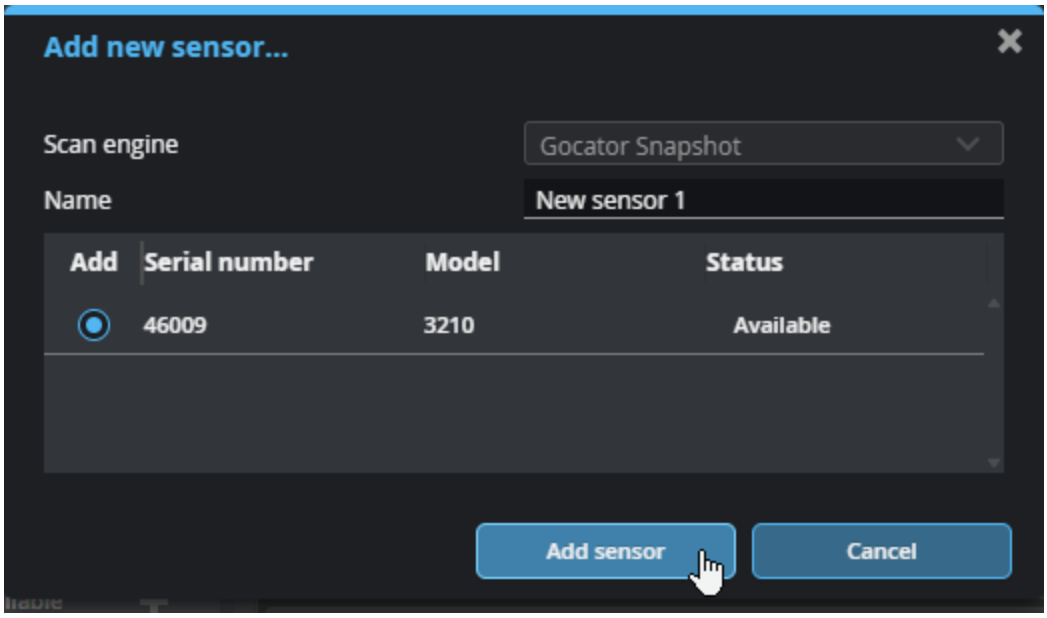


2. On the **Connect** page, in the **Visible Devices** area, click the plus symbol next to the sensor you want to add.

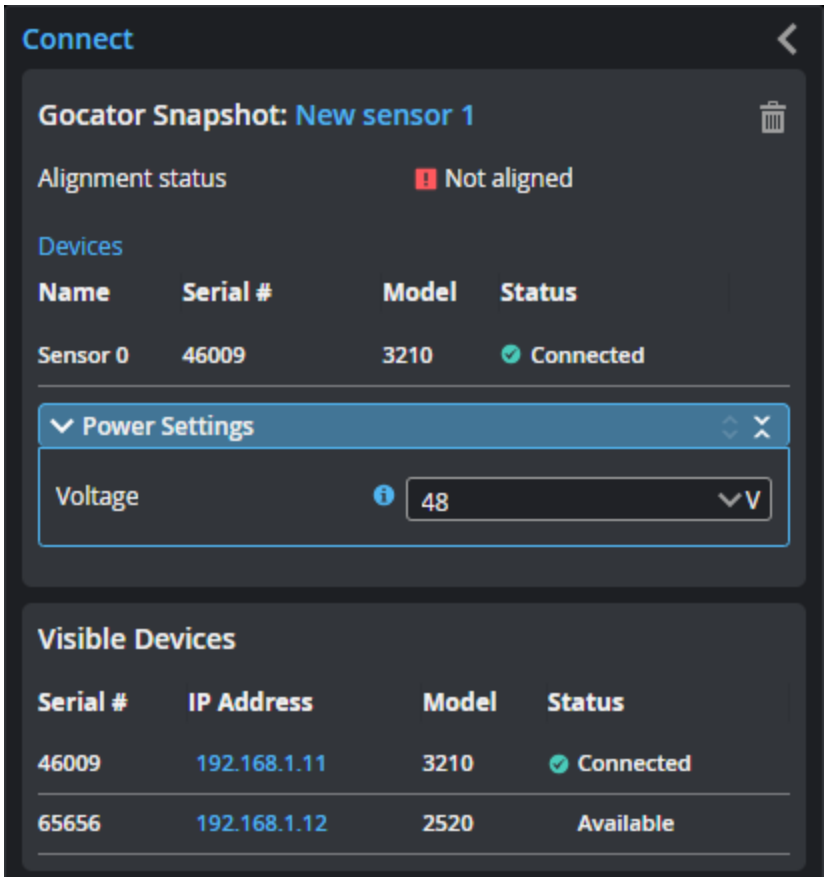


3. In **Add new sensor...**, if the sensor is not already selected, click next to it in the **Add** column, and click **Add Sensor**.

   You can optionally set the name of the sensor.



The physical sensor is now connected to the PC instance of GoPxL and is automatically accelerated.



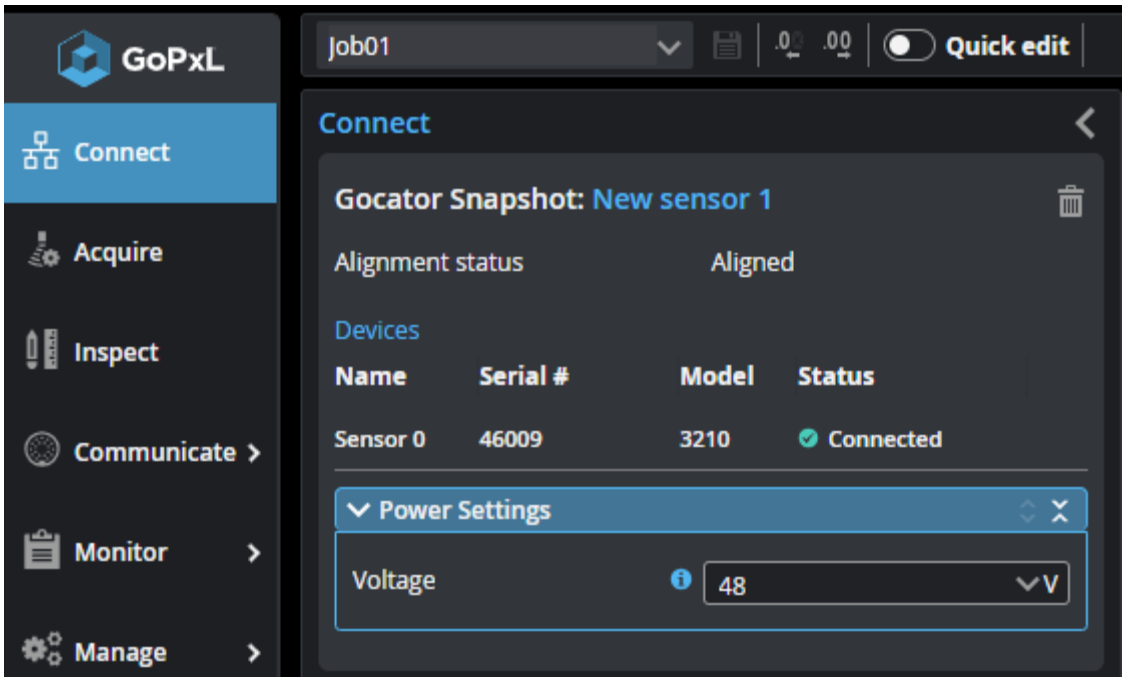
You access the sensor via the IP address of the PC instance, through which you control and configure it.

Stopping Acceleration

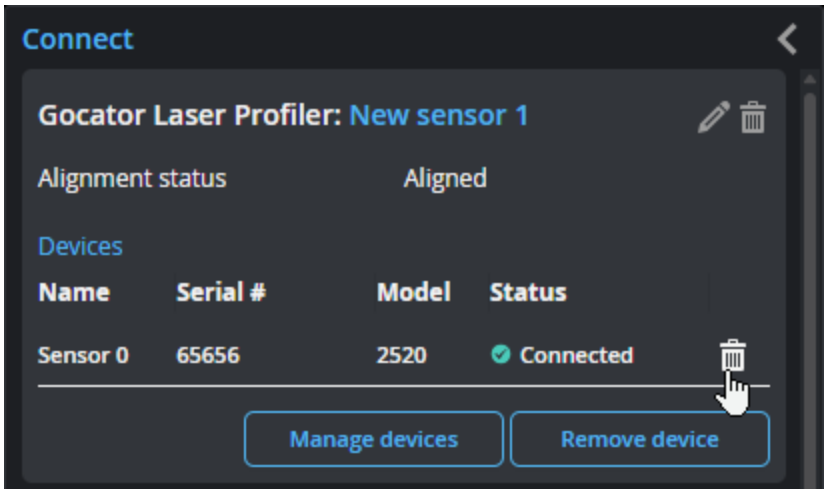
To stop accelerating a sensor, you remove it from the PC instance of GoPxL it's connected to. You also need to remove a sensor from a PC instance to upgrade the sensor's firmware.

To remove a sensor from a PC instance

1. In the GoPxL interface of the PC instance, go to the **Connect** page.

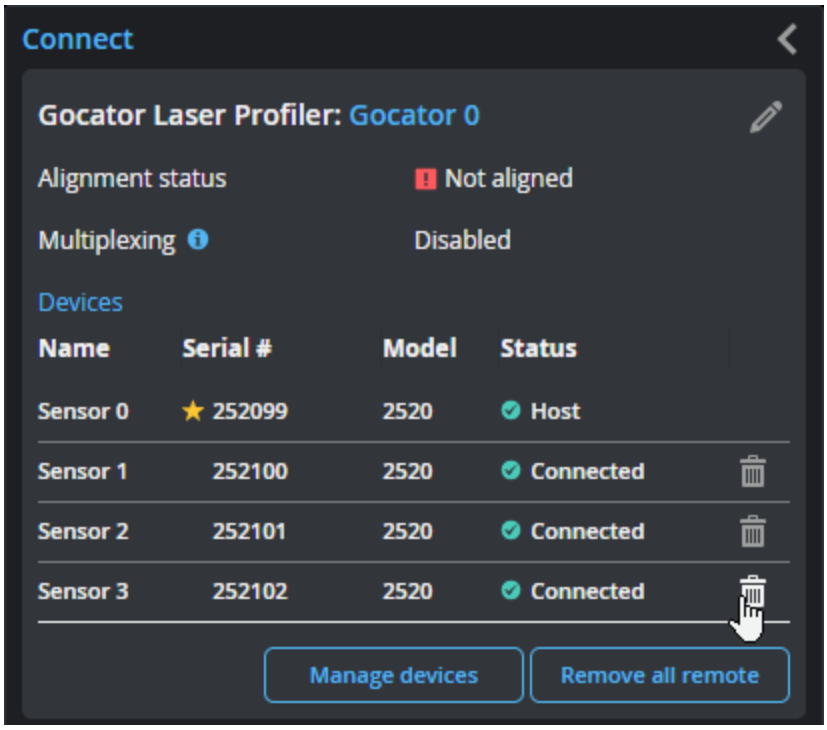


2. With a single accelerated line profile sensor, click the trashcan icon or **Remove device**.

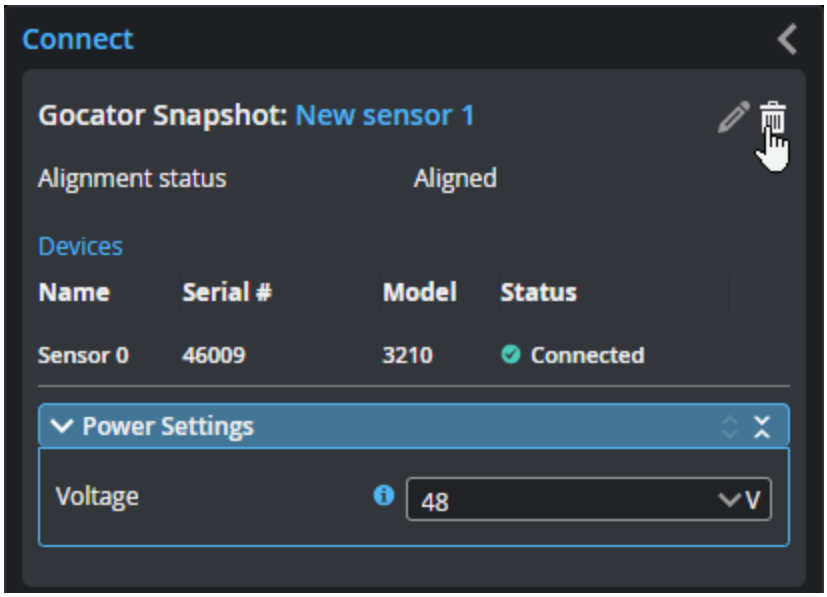


Remove sensor from the group: Choose this if you have started configuring settings on the **Acquire** page and just need to add a different sensor to the sensor group.

3. With a multi-sensor line profile system, click a trashcan icon to remove a single sensor, or **Remove all remote** to remove all sensors.



4. With G3 sensors, click the trashcan icon.



The sensor is removed from the PC instance.

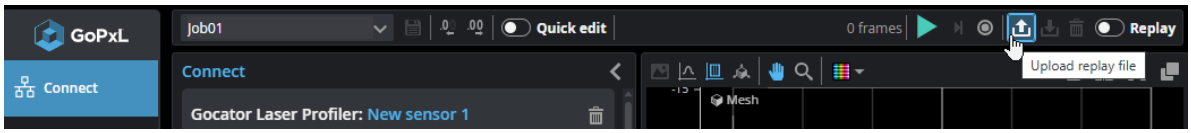
Loading Scan Data

To examine previously recorded scan data offline, you load a .gprec file into the PC instance of GoPxL.

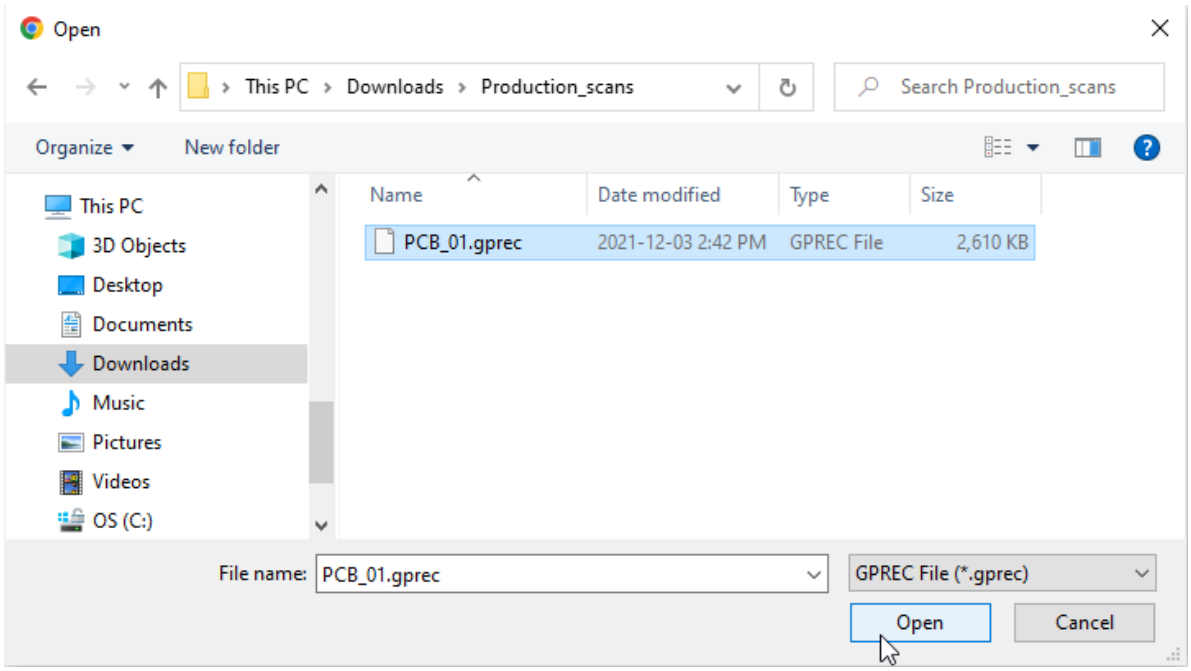
	You can also load a backup file or a support file to examine previously recorded scan data.

To load recorded data (.gprec)

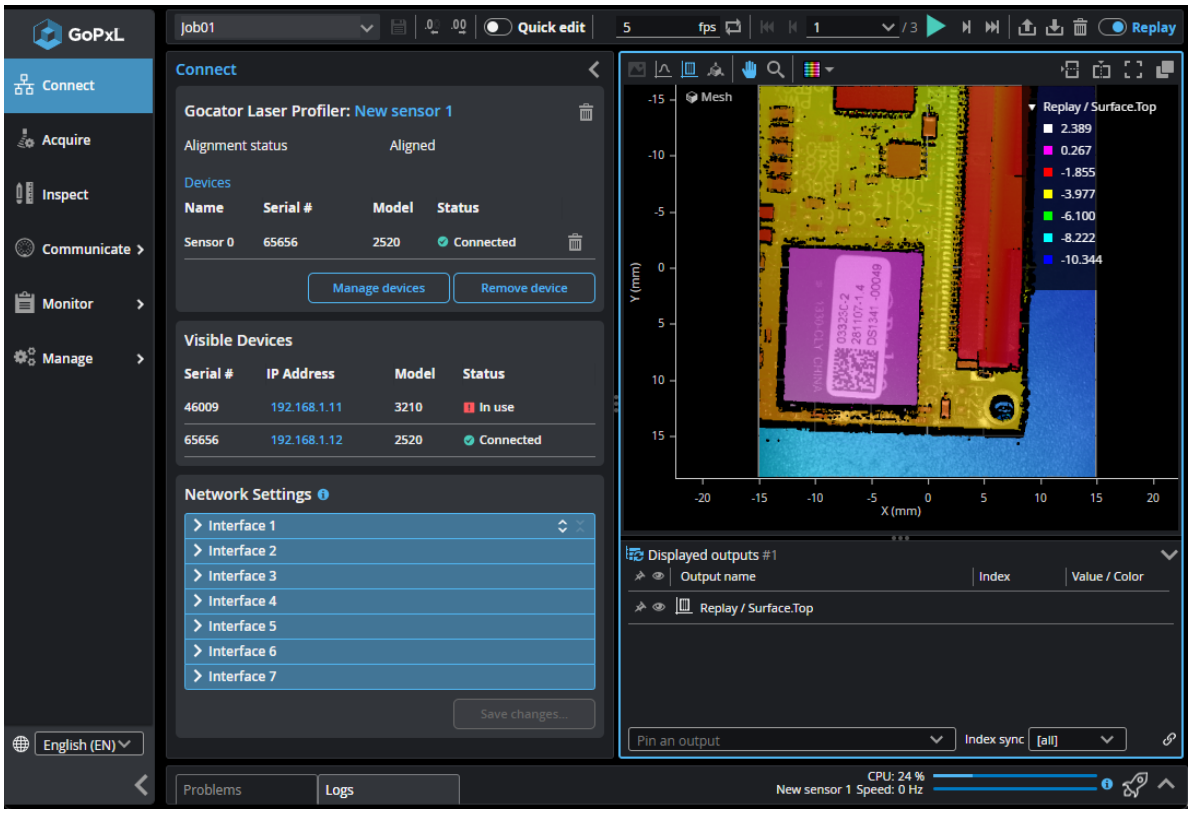
1. In the GoPxL interface, above the data viewer, click the Upload replay file button.



2. In the **Open** dialog, locate and select the recording (.gprec) you want to load, and click **Open**.



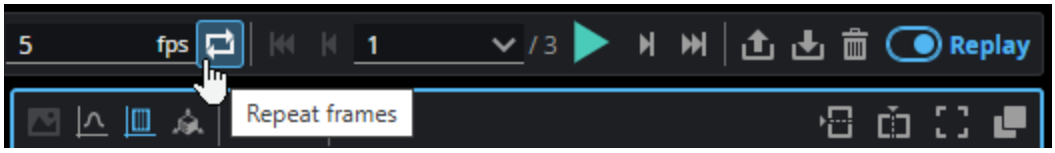
The recording loads in GoPxL, and GoPxL automatically switches to Replay mode.



You can now examine the recorded data and configure tools run on it.

3. Use the replay controls above the data viewer to move through the frames, go to a specific frame, toggle and configure the speed of repeat frames mode (continuous loop), and so on.

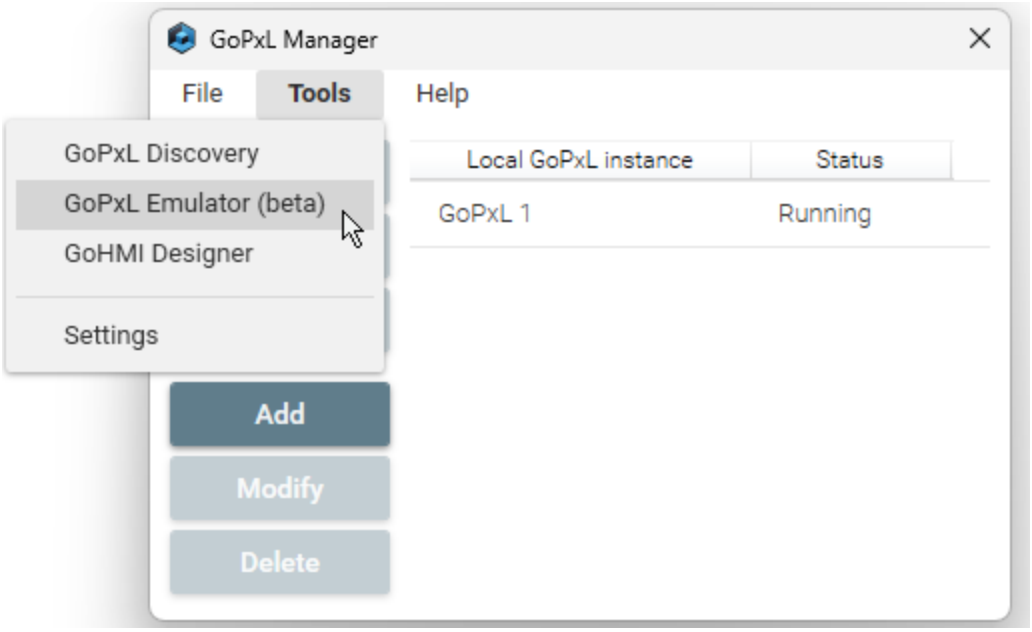
   Hover over the replay controls for tooltips describing their function.



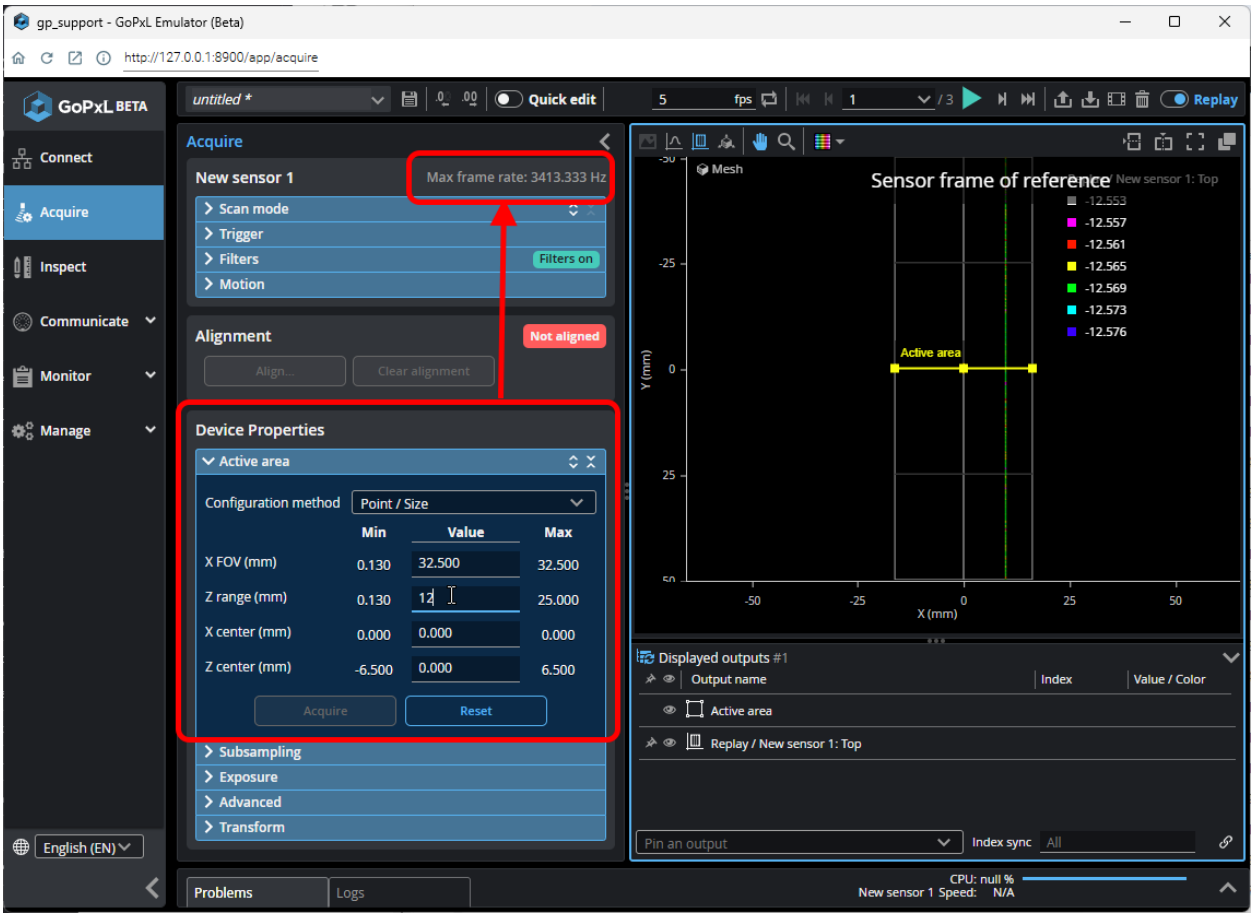
Using repeat frames mode with a low frame speed is useful for adjusting measurement tool settings and seeing the impact on scan data, without having to click Start or step through the frames of scan data manually. Repeat frames mode is also useful for demos.

GoPxL Emulator (Beta Feature)

You can access a GoPxL emulator in the GoPxL Manager, under the **Tools** menu.



You can open support files (.gpsup) for single 3D sensor models only. After you select a support file, the emulator starts. You can then evaluate the maximum frame rate for active area changes, on the **Acquire** page.

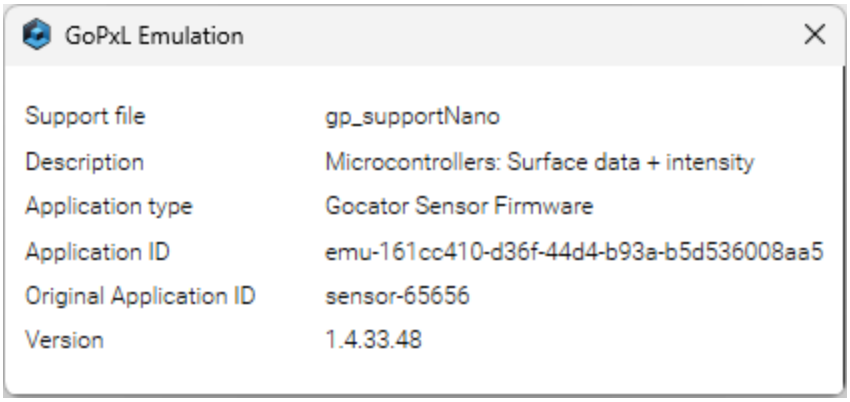


You can also associate .gsup files with the emulator by enabling **Associate gpsup files with GoPxL Emulator** in GoPxL Manager; for more information, see *Running GoPxL on a Windows PC* on page 403. Lastly, you can emulate support files saved from the emulator.

The emulator supports the following:

- Click the Home ( ) and Refresh ( ) buttons to go to the default page and refresh the current page, respectively.
- Click the Open in external browser button ( ) to open the emulated GoPxL instance in your default web browser.
- Click the Info button ( ) for information on the support file.

Note that **Version** refers to the version of GoPxL from which the support file was saved.



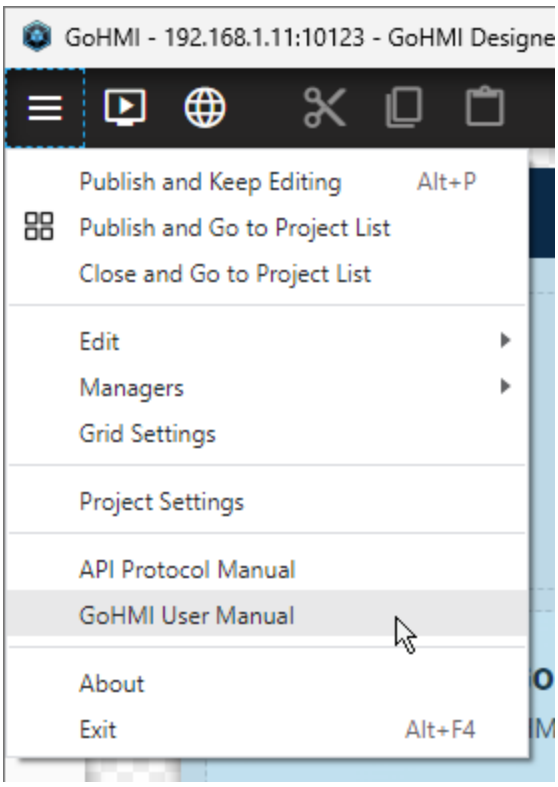
Note the following limitations:

- Support files from GoPxL 1.3 and earlier versions cannot be loaded.
- Support files from GoMax, Gocator 2D Smart Camera, and multi-sensor systems are not yet fully supported.

GoHMI and GoHMI Designer

GoHMI lets you create flexible human-machine interfaces (HMIs) and provide a custom GUI containing system and inspection metrics. Your HMI can also be touch-enabled. You create and edit HMIs in GoHMI Designer. You can design different layouts for different devices so that, for example, your HMI will adapt to the screen it is being displayed on (a PC monitor, a laptop, a tablet, or a phone), as well as its orientation on the screen.

For complete information on creating an HMI app, open the menu in the upper left of the designer, and choose one of the manuals.



> If you don't see changes you have made in the GoHMI app, or an HMI you've just uploaded, you may need to clear your browser's cache to make it available: the GoHMI is cached for 60 seconds.

Before using the designer, you must enable GoHMI in GoPxL; for more information, see *Enabling and Configuring GoHMI in GoPxL* on page 430.

> GoHMI and GoHMI Designer only support a single HMI app, called "GoHMI." Also, GoHMI can only be enabled on one PC instance at a time. If you try to start GoHMI another instance, GoPxL displays an error.

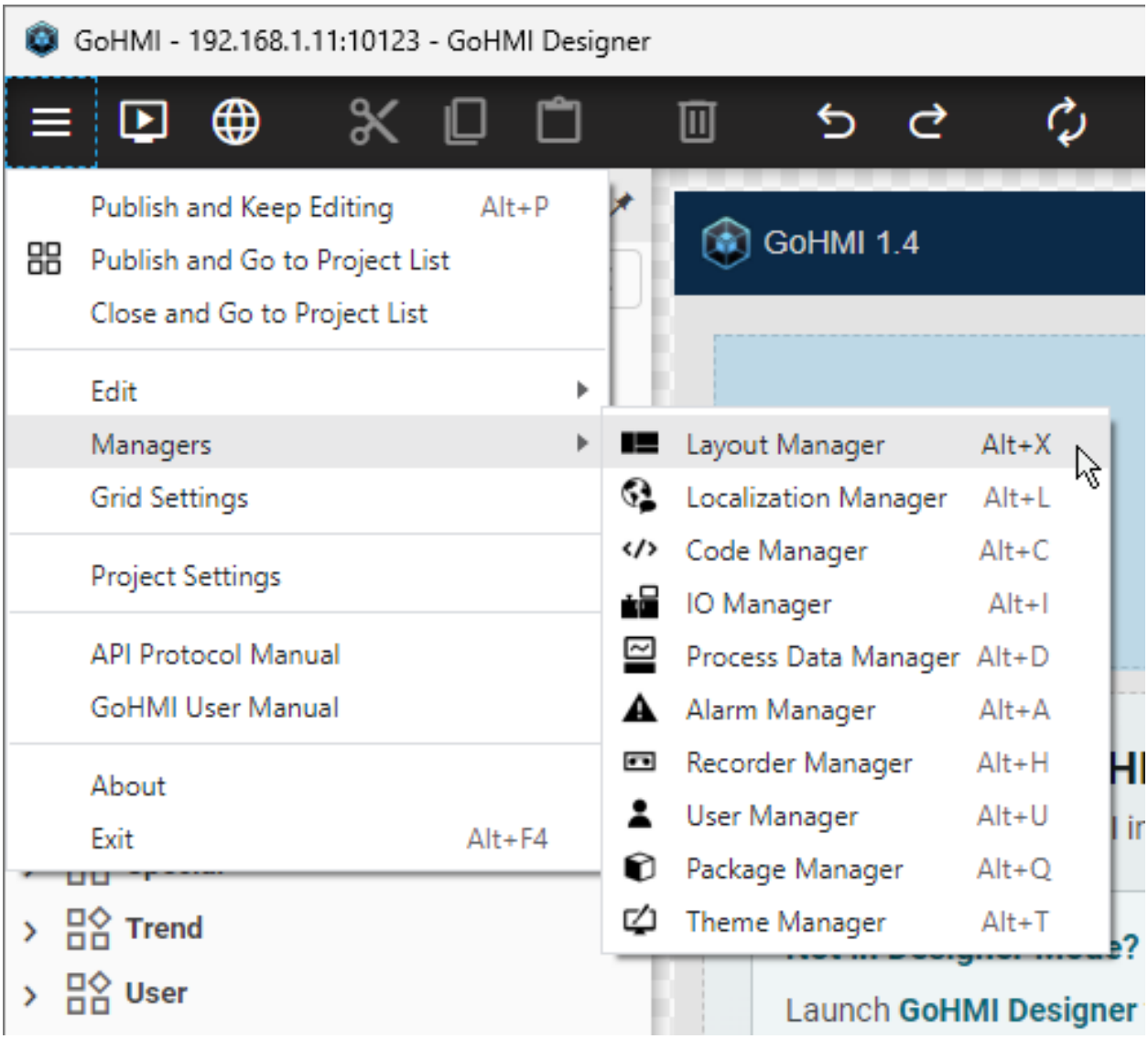
LMI provides a default HMI app with GoHMI that you use as a starting point or simply to explore the capabilities of GoHMI. When you launch GoHMI Designer, the default app loads in the designer.

GoHMI Designer is available in the GoPxL Utilities package (`14631-x.x.x.x_SOFTWARE_GoPxL_Utilities.zip`), available on LMI's Product Downloads page (https://lmi3d.com/product-downloads/). After you have downloaded the package, unzip it and make note of the folder's location; the shortcut to the designer is in the `GoPxL` folder.

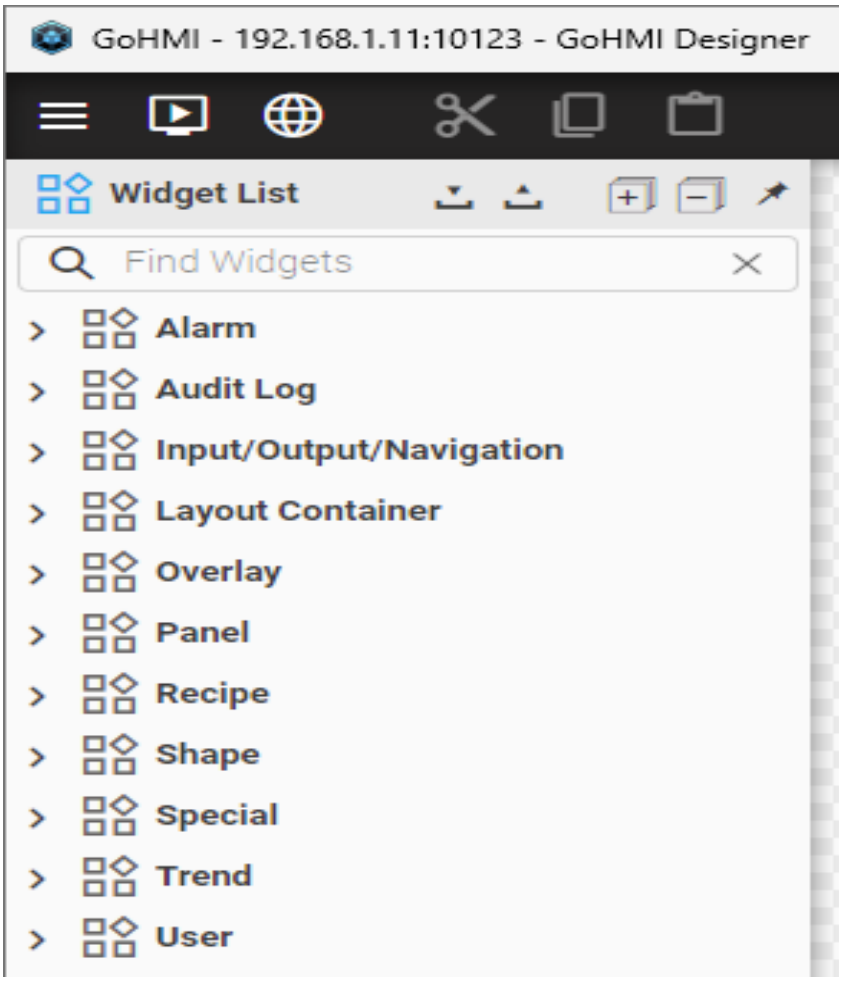
GoHMI Designer requires the Microsoft Visual C++ Redistributable package. This package is usually already installed on most PCs. If you are unable to launch HMI Designer, you may need to install this package, which can be downloaded at https://docs.microsoft.com/en-us/cpp/windows/latest-supported-vc-redist?view=msvc-170.

If you decide to run your HMI application on a sensor, instead of on a PC instance or a GoMax NX device, you *should not* use the HMI data trending functionality. Data trending writes data repeatedly to storage, which can shorten the lifespan of the device's flash memory. Also, if the sensor's storage becomes full, the device may not be able to start. If you need data trending, you should run the HMI on a PC or on GoMax.

GoHMI Designer organizes the available design functionalities and settings into "managers" and settings pages. After you have opened an HMI app in GoHMI Designer, you can access the managers from the upper left menu in **Managers**.



In the Layout Manager, the side panel lets you access the list of widgets (user interface controls such as buttons) and layout containers you use to create your HMI. Click on "Widget List" to open the list. The widgets are grouped into categories. Below, the "Widgets" category is open.



GoHMI application layout development involves the concept of containers. Containers contain widgets or other containers (that is, containers can be nested). Containers can display their contents horizontally (for example, a row of buttons and fields) or vertically (for example, a stack of buttons). By nesting containers and adjusting their alignment, you have complete control over your HMI app's layout and design.

The following provides a high level view of what you need to do to start creating an HMI. The first steps below refer to general acquisition and measurement tool setup, which is described elsewhere in the manual. The remaining steps are described in further details below.

1. Install a sensor or group of sensors.

   For more information, see the installation and networking topics under .

2. Configure the sensor acquisition settings.

   For information on configuring acquisition settings, see *Acquire - Calibration and Image Acquisition* on page 107.

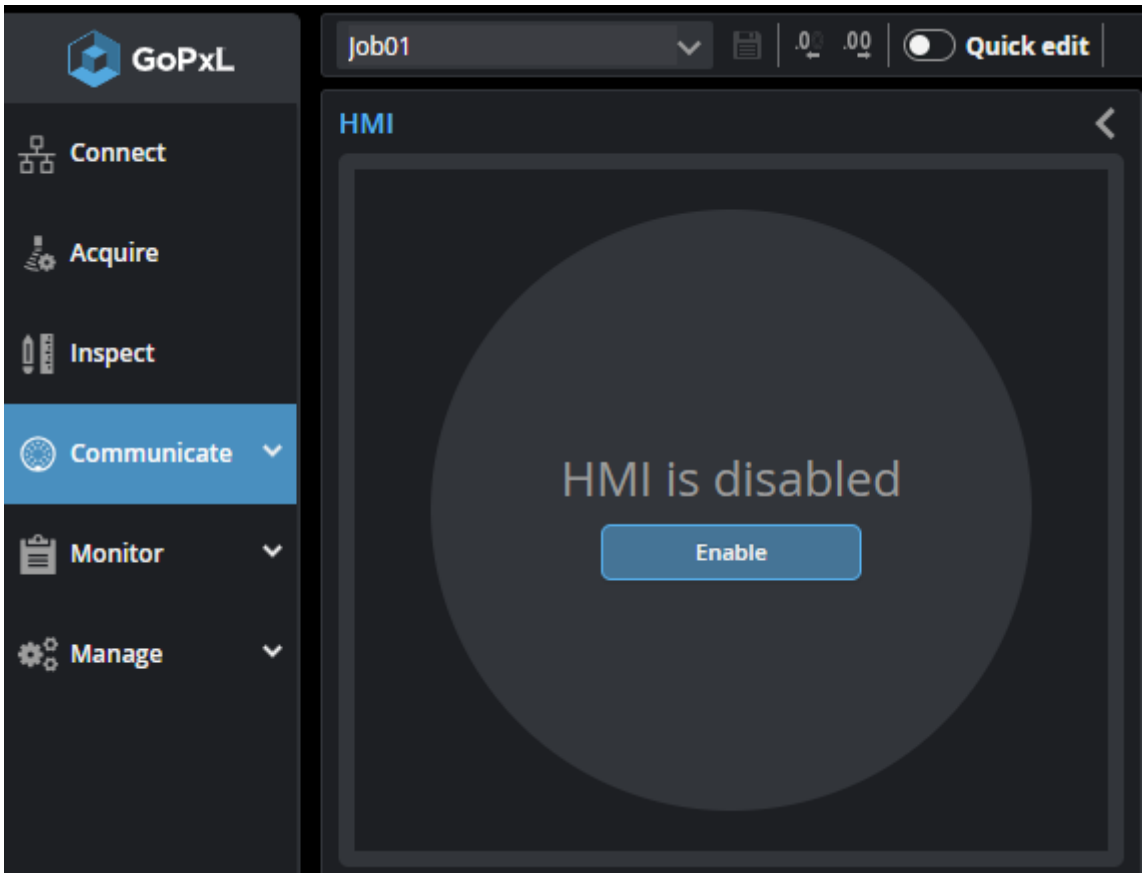
3. Add and configure measurement and processing tools.

   For more information, see *Inspect - Measurement and Processing* on page 117 or the topics for specific tools.

4. (Optional) If your application requires the Gocator communication protocol or any of the industrial protocols, enable and configure these.

   For more information on the protocols available in GoPxL, see *Protocols (PLCs and other hardware)* on page 472.

5. Enable the HMI control settings in the **Communicate** > **HMI** page using the toggle and configure GoHMI.



   For more information, see *Enabling and Configuring GoHMI in GoPxL* on the next page.

6. Launch GoHMI Designer.

   For more information, see *Launching GoHMI Designer* on page 435.

The following lists the browser requirements of GoHMI apps:

- Chrome 80 (released 04.02.2020)
- Firefox 74 (released 10.03.2020)
- Edge 80 (released 07.02.2020)
- Safari 14 (released 16.09.2019)
- Opera 67 (released 25.02.2020)

Overview of Managers and Settings Pages

For information on the editors (called "managers") you use in GoHMI Designer, see the following table.

Overview of managers and settings pages

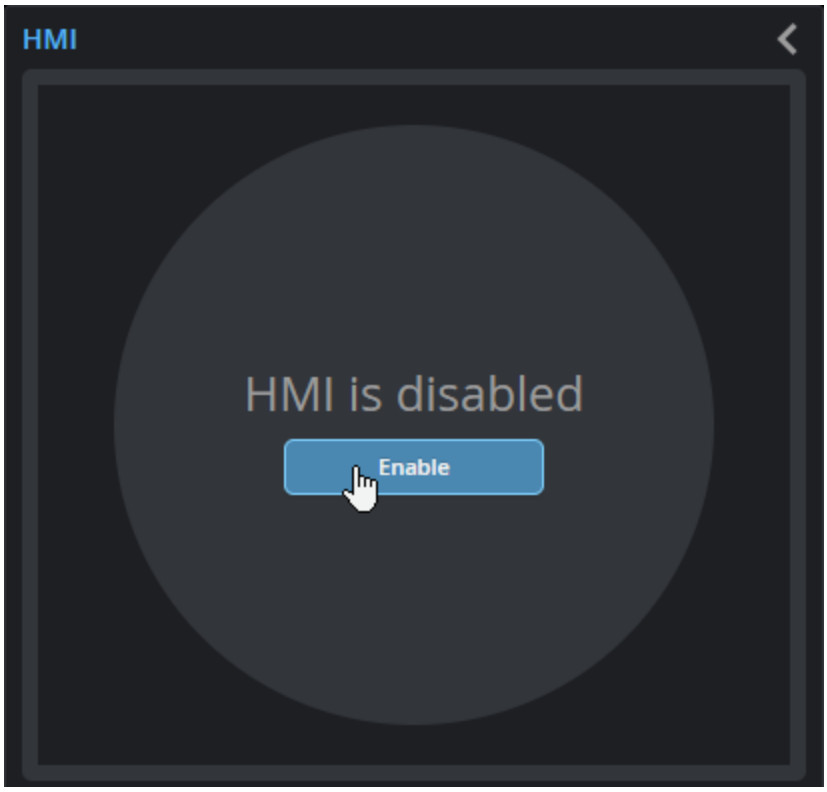
Task	Manager	Description
Creating HMIs with Multiple Views	• Layout Manager	Creating views in your HMI using panels in GoHMI Designer.
Laying Out Your HMI	• Layout Manager	Styling your HMI in GoHMI Designer.
Using Widgets	• Layout Manager	Using the widgets provided in GoHMI Designer.
Styling the HMI	• Layout Manager	Laying out views and widgets in GoHMI Designer.
Using UI Actions	• Layout Manager	Defining UI-actions (such as mouse clicks on buttons in the HMI) to implement simple behaviors in GoHMI Designer.
Localizing the HMI	• Localization Manager	Localizing your HMI in GoHMI Designer.
Using the Integrated Code Editor	• Code Manager	Using scripts to add capabilities and features to your web HMI or to modify the appearance of widgets.
Processing Data Interface and Data Items	• IO Manager • Process Data Manager	Configuring the data interface and connecting your HMI to a sensor system using GoHMI Designer.
Simulating Process Values	• Process Data Manager	Simulating process values in GoHMI Designer to evaluate or to demonstrate your web HMI without a connected sensor system.
Using Alarms and Alarm Lists	• Alarm Manager	Defining alarms and messages, store them as historical alarms, and display the corresponding messages in the HMI.
Using Recorder and Trend Display	• Recorder Manager	Recording process variables and displaying them in a trend diagram in the HMI. The integrated data recorder supports both live trends and trends from historical data.
Using Recipes and Recipe Widgets	• Recipe Template Manager	Generating recipe templates and useful metadata such as "article numbers," "customer description," and so on, which serve only for documentation. In applications, recipe process values can be applied or captured.
Managing Users and Access Groups	• User Manager	Creating users and user groups and how to assign access rights.
Managing Packages	• Package Manager	Extending GoHMI Designer with additional packages for widgets, themes, and starter apps.
Adjusting App Settings	• HMI Project Settings	Changing application settings such as project settings in GoHMI Designer.
Adjusting Grid Settings	• Grid Settings	Adjusting the snap-to grid size in the designer.

Enabling and Configuring GoHMI in GoPxL

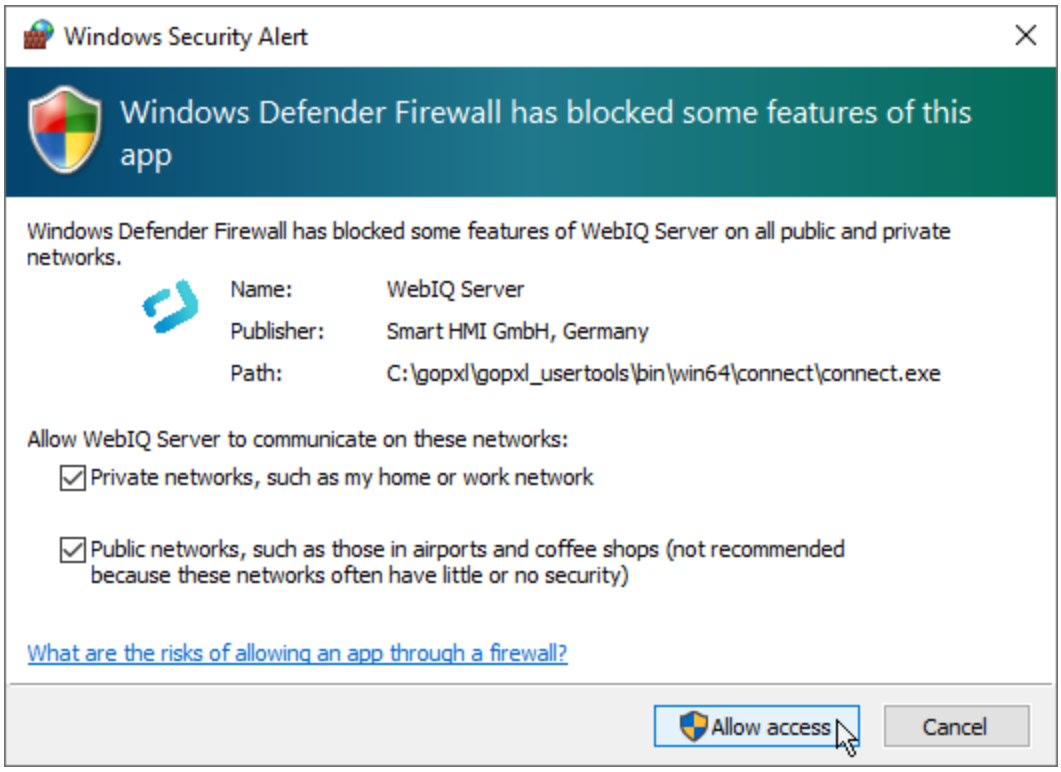
This section describes how to enable and configure GoHMI in the GoPxL interface. Before continuing, make sure you have created a sensor system, configured acquisition, and added and enabled the tools and measurements you want to display in the HMI.

To enable and configure GoHMI

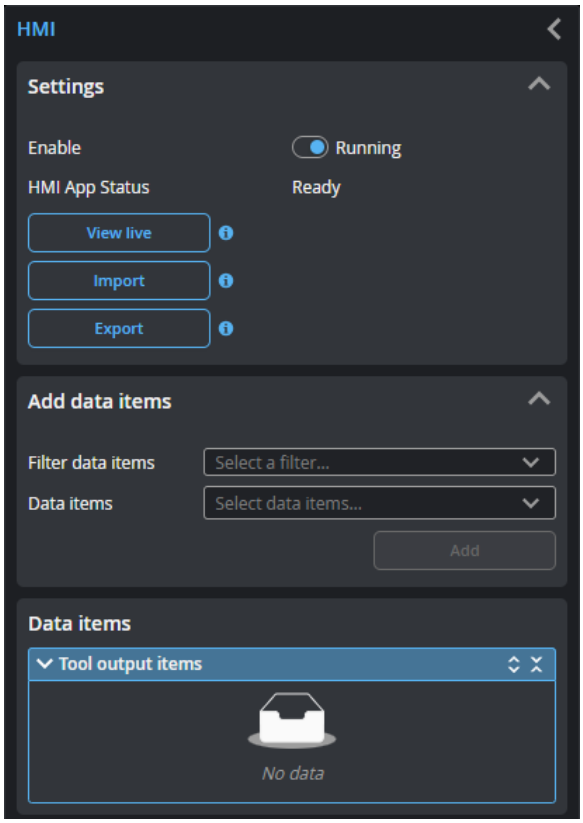
1. On the **Communicate** page, in the **HMI** panel, enable the HMI service.



If you are running GoPxL on a PC, a Windows Security Alert referring to WebIQ (the service on which GoHMI Designer runs) may appear. Enable both networks.



The HMI service launches, and the HMI panel displays the following:



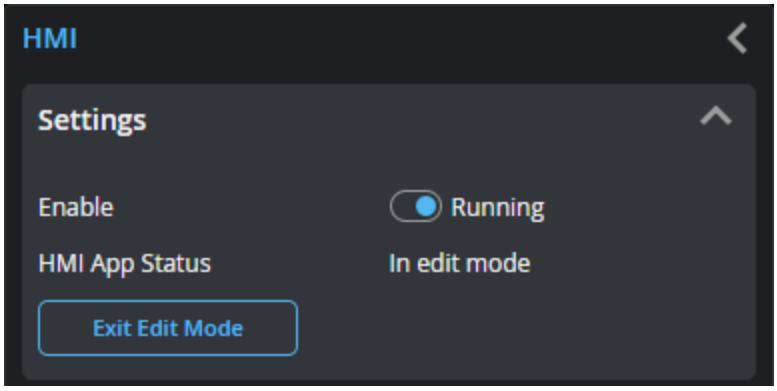
In addition to letting you disable the service and see the HMI app's status, the **Settings** panel lets you perform various operations.

When the application is running, you can open the HMI app in your browser by clicking the **View live** button. You can make note of the URL of the HMI in this tab for future use.

Import lets you import an HMI application.

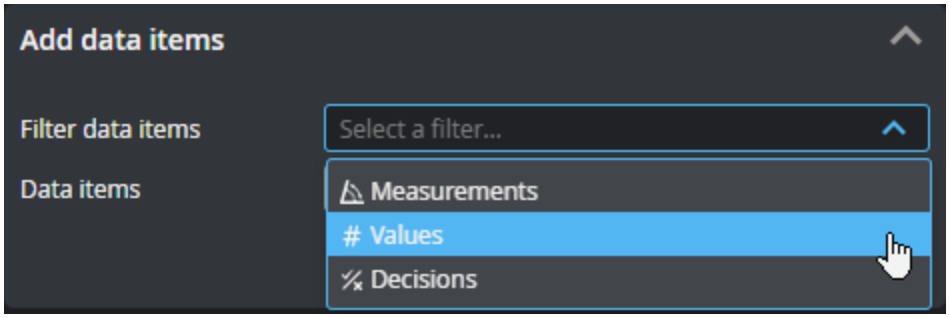
Export lets you export the HMI application currently available on the device (or in the PC instance), to back it up or transfer it to another sensor or PC.

2. If you see "In edit mode" as the HMI status instead of "Ready", you must remove it from the GoHMI Designer.

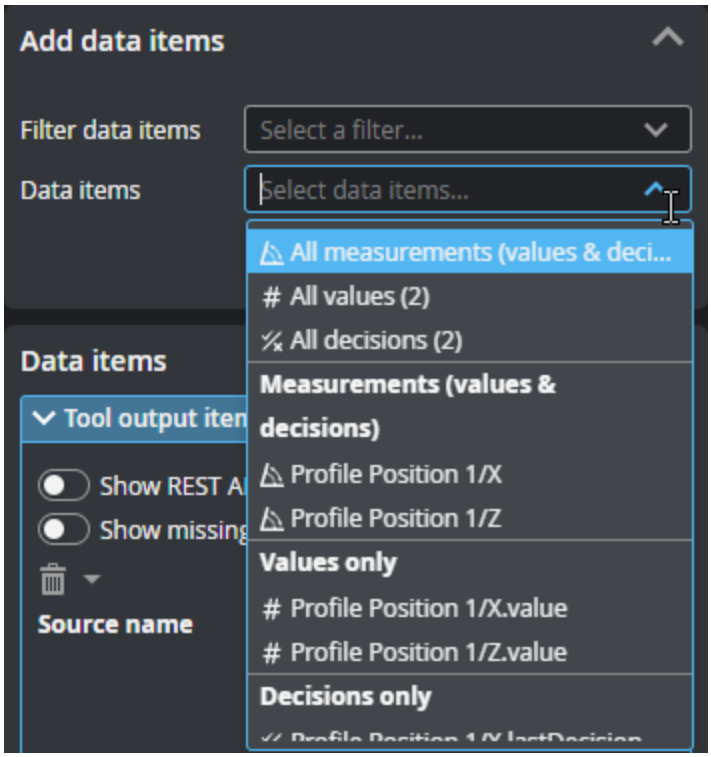


The status **In edit mode** means that the HMI app is currently loaded into the GoHMI Designer workspace. If you know you have finished making changes in GoHMI Designer, and have closed it without publishing it (to the sensor or PC instance of GoPxL), you can "force" exiting edit mode by clicking **Exit Edit Mode**. Note that doing this removes any unpublished changes in GoHMI Designer. For information on properly closing the designer and publishing changes, see *Publishing Changes or Exiting* on page 440.

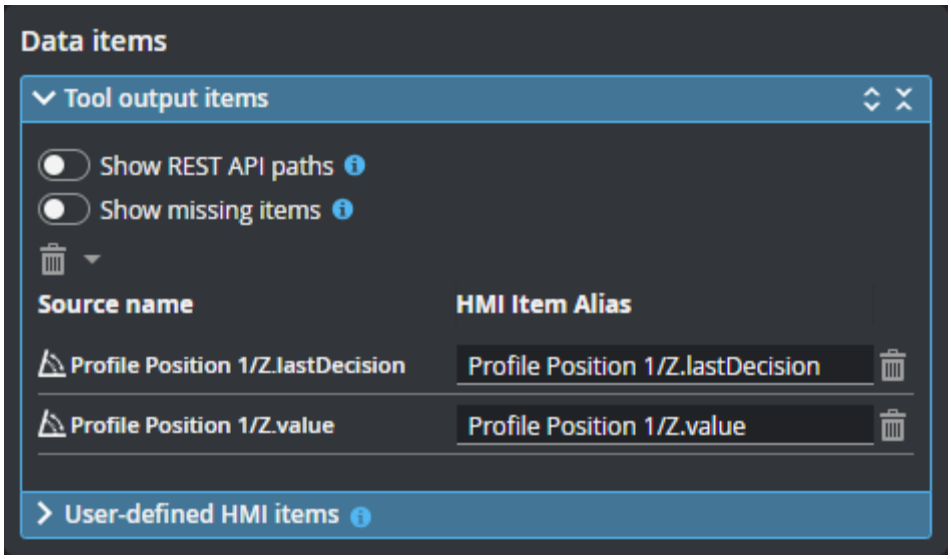
3. (Optional) You can filter the outputs to limit them to only values or decisions.



4. In the **Add Data Items** section, expand the **Data outputs** drop-down and click an output you want to send to the HMI.



After a few seconds, the output appears in the list of outputs that GoPxL will send to the HMI.



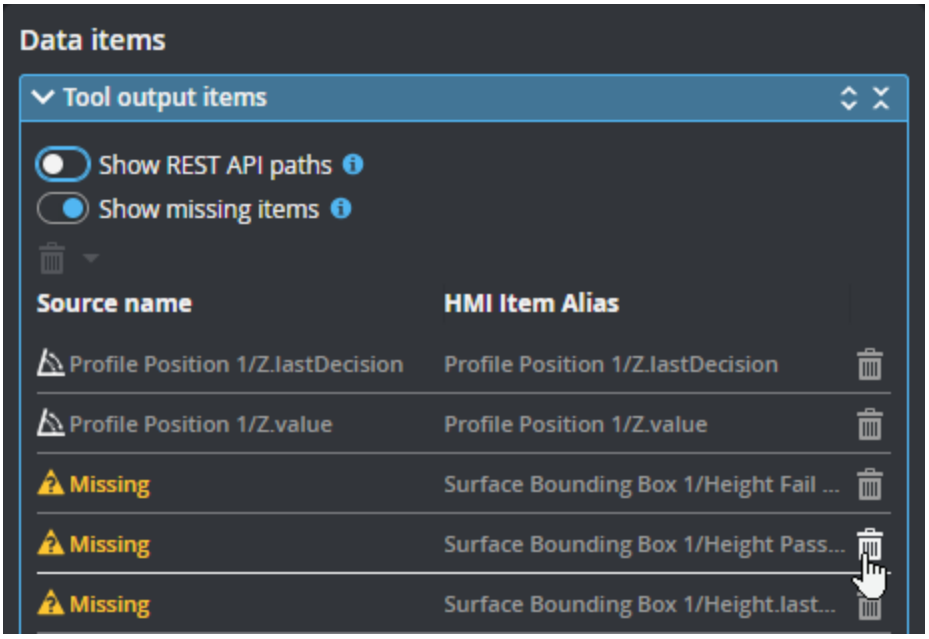
If you don't see the output you want to add in **Data items**, check that you have enabled it in the measurement tool, on the **Inspect** page.

5. (Optional) Toggle **Show REST API paths** to switch between showing the HMI item aliases and REST API of the outputs.

   HMI item aliases are how data outputs appear in GoHMI Designer.

   REST API paths are how the data output is represented in the REST API.

If GoPxL outputs referenced in an HMI are accidentally disabled or renamed in GoPxL, or the tool containing the outputs is removed, a **Show missing items** toggle will be displayed. Toggle this on to see which outputs are missing, and resolve the issue as required.

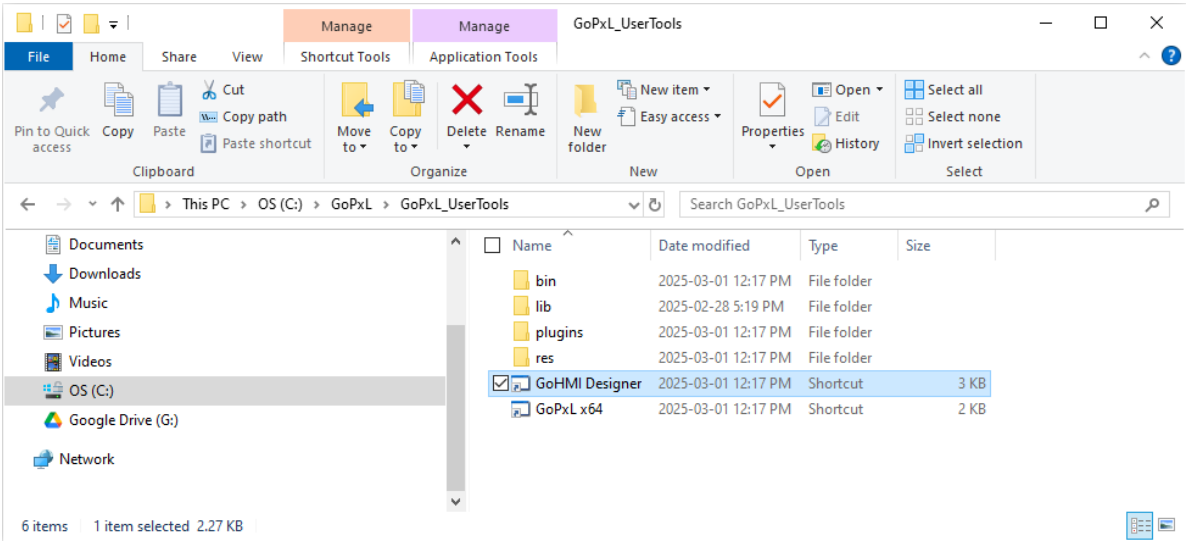


Launching GoHMI Designer

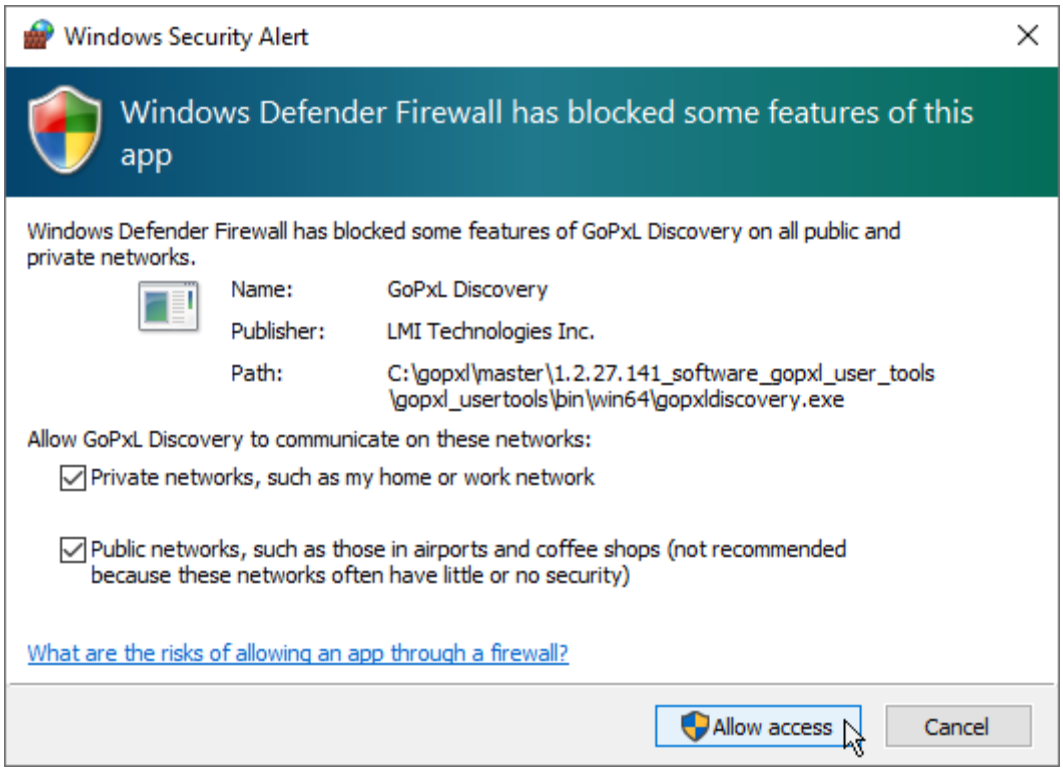
After you have enabled the HMI service in GoPxL and configured it by adding the outputs you want to send to the HMI, you can launch GoHMI Designer to create or edit an HMI.

To launch GoHMI Designer

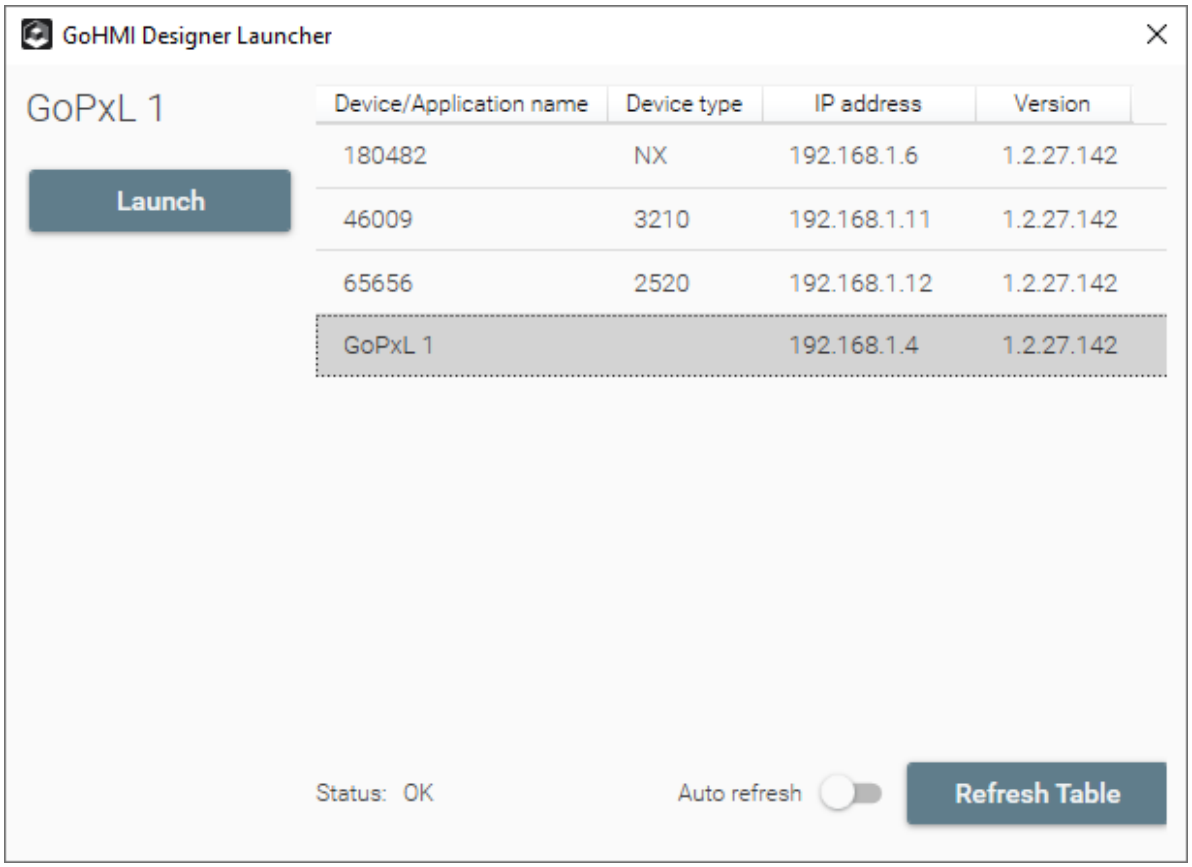
1. Go to where you extracted the GoPxL tools, and double-click the GoHMI Designer shortcut.



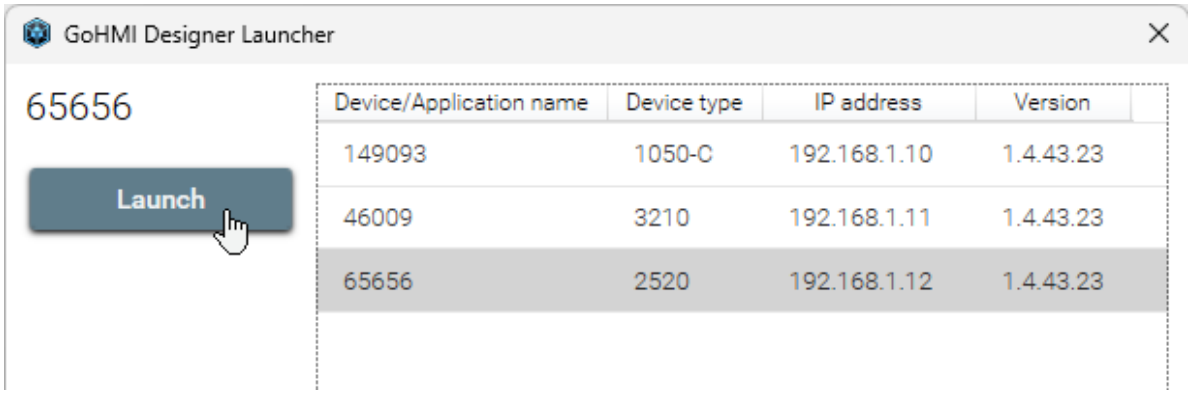
2. (Optional) If a Windows Security Alert opens, enable both networks and click **Allow access**.



GoHMI Designer Launcher opens and lists the available instances of GoPxL. In the following, we see two sensors, a PC instance of GoPxL called "GoPxL 1," and a GoMax NX device.



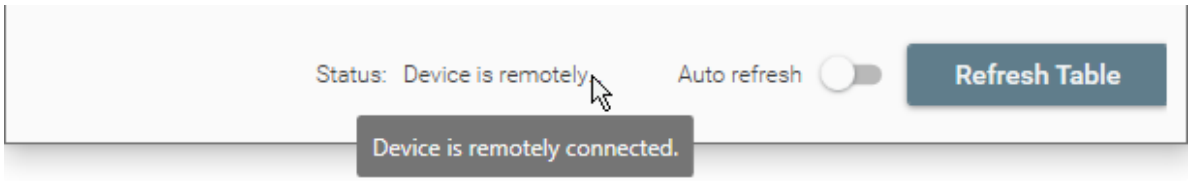
3. In the launcher, select the row of the GoPxL instance for which you want to launch GoHMI Designer and click **Launch**.



You may need to refresh the table to get the most up-to-date list by clicking **Refresh Table**. You can toggle **Auto Refresh** to have the table refresh automatically.

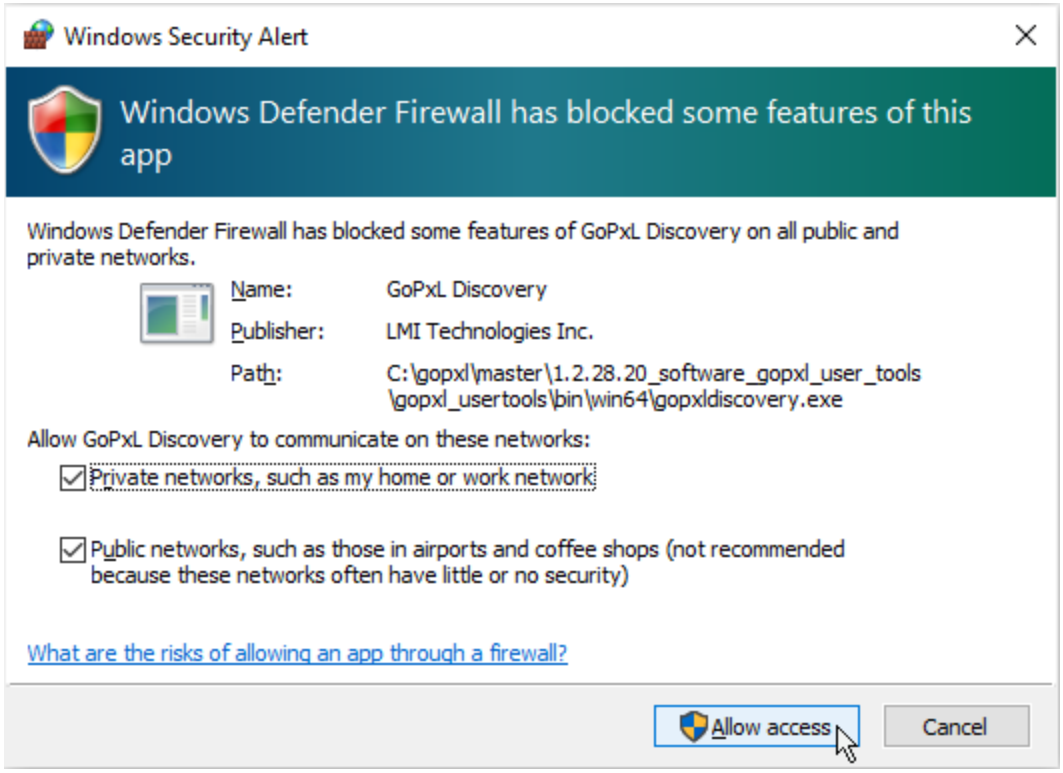
Note that you can only launch one instance of GoHMI Designer at a time.

If a sensor is currently running through a PC instance of GoPxL, its status is "Device is remotely connected." In this case, you can only launch GoHMI Designer from the PC instance through which the sensor is running.

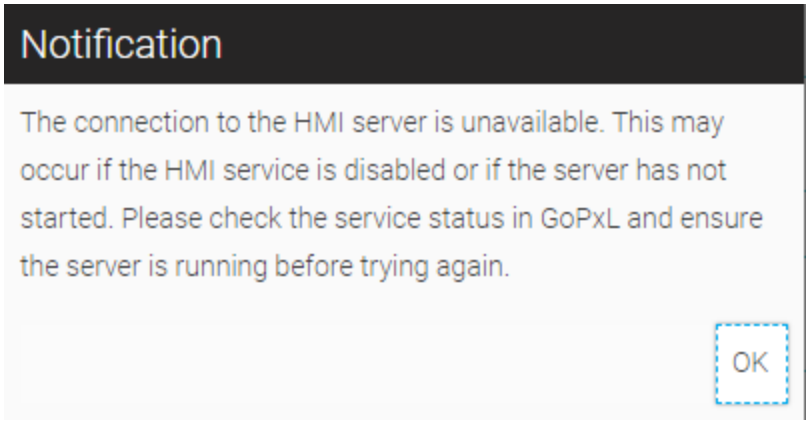


The launcher closes and GoHMI Designer starts.

A Windows Security Alert may appear. If so, allow the designer to communicate over both networks.



If you see a notification warning that the connection to the server has been interrupted, make sure you have enabled the HMI service in GoPxL. After enabling the service, click OK in the notification. (For more information, see *Enabling and Configuring GoHMI in GoPxL* on page 430.)



4. If necessary, log in to the designer.

   By default, use "gohmi" for the user name and "gohmi" or "gohmi123" for the password.

Opening, Editing, and Saving a Project

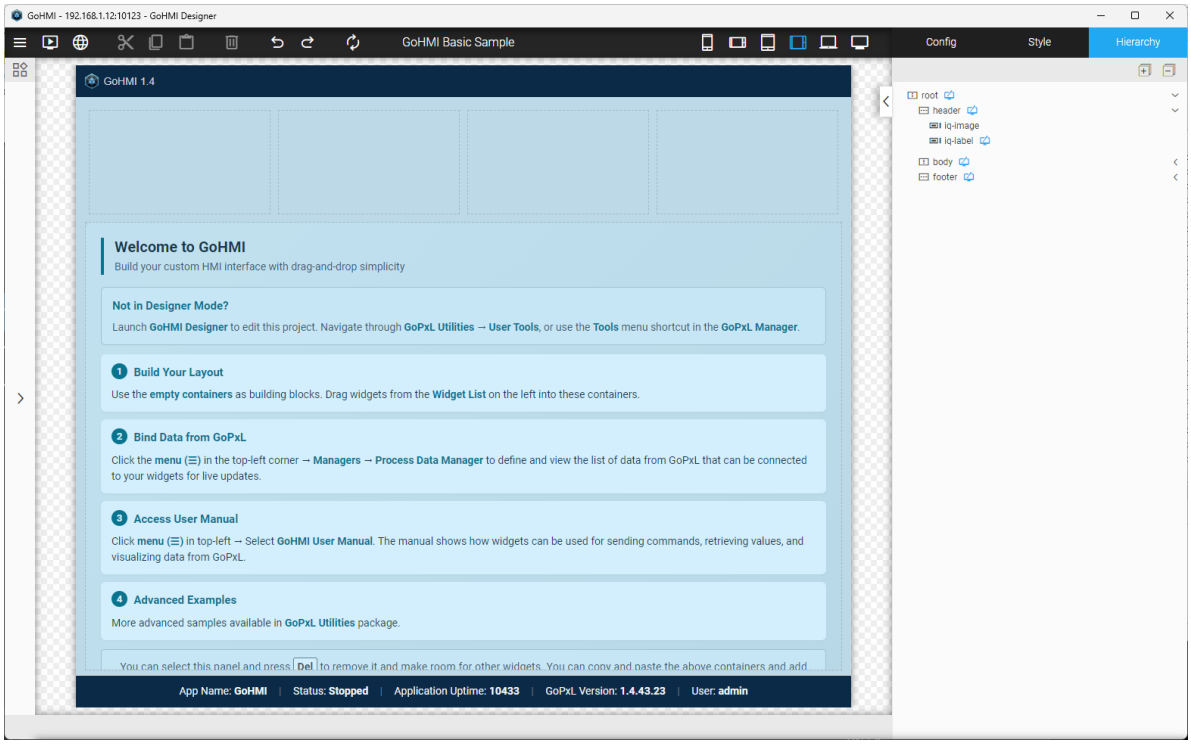
This section describes how to open the GoHMI project (called GoHMI), modify it, and save it.

> GoHMI Designer only supports a single HMI application at a time.

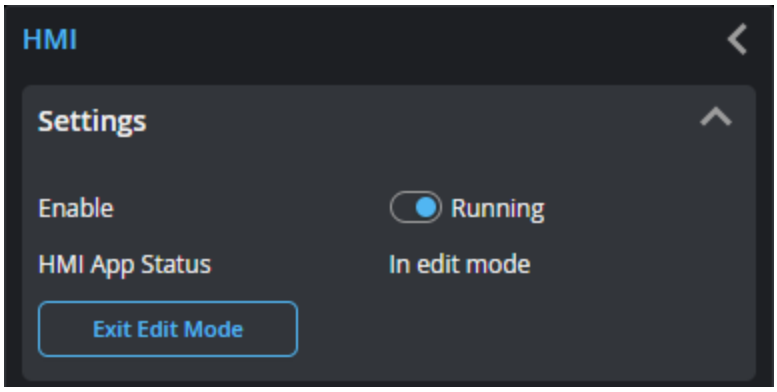
To open a project

1. In the GoHMI Designer main screen, hover the mouse over the GoHMI project and click it.

The project opens in the Layout Manager.

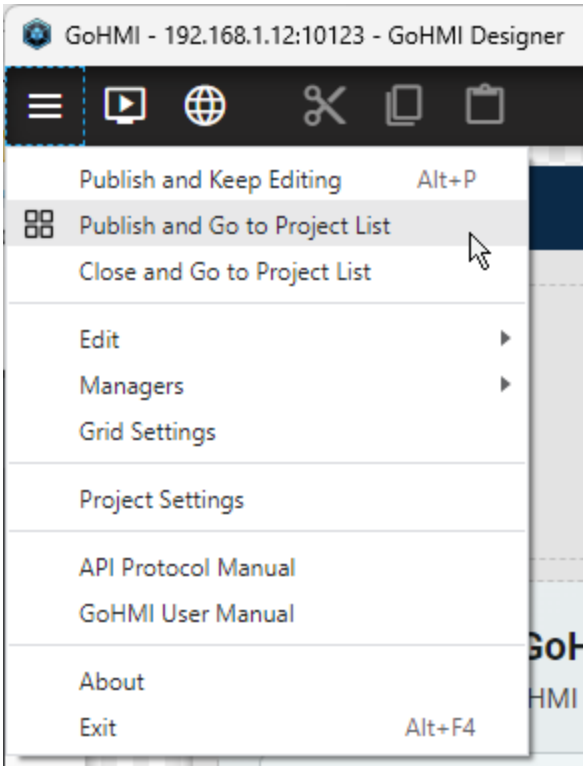


Note that after the HMI loads in GoHMI Designer, in the GoPxL interface the HMI app status becomes "In edit mode."



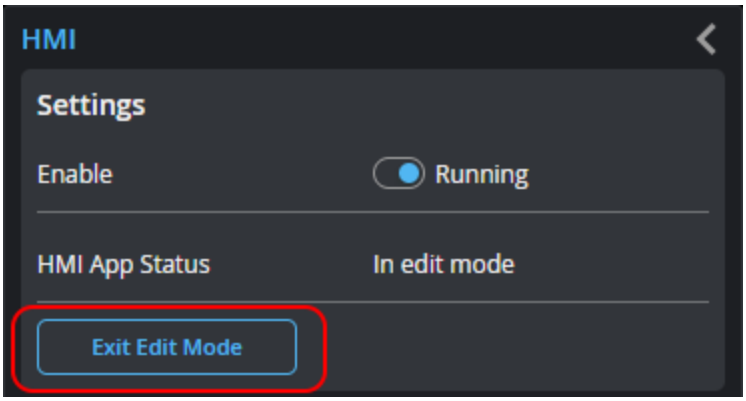
Publishing Changes or Exiting

As you work on an HMI, the designer automatically saves changes locally. To save changes onto a device (sensor or GoMax) or into a PC instance of GoPxL, you must "publish" the HMI app using one of the two options from the main menu at the upper left of the Designer.



When you have finished working on your HMI, in the menu, you should choose either **Publish and Go to Project List** or, if you have previously published all changes, **Close and Go to Project List**. Then, from the project list, close the designer by clicking the Close icon (✕) at the upper right.

In some situations, such as when you have closed the designer without publishing unneeded changes, you can "force" exiting edit mode by clicking **Exit Edit Mode** in GoPxL, in the **Communicate** > **HMI** page. Note that doing this removes any unpublished changes in GoHMI Designer, so make sure it is safe to do this.



Changing Mapped Outputs in GoPxL

Before trying to add or remove measurements at a later date to outputs sent to the HMI application, the **HMI App Status** in GoPxL must say "Ready". If **HMI App Status** shows "In edit mode" and if you have unpublished changes in the designer that you want to keep, follow the steps in *Launching GoHMI Designer* on page 435 and then, in GoHMI Designer, publish the changes. For more information on publishing changes, see *Publishing Changes or Exiting* on the previous page. Otherwise, you can force GoPxL to exit edit mode by clicking **Exit Edit Mode**, but you will lose any unpublished changes.

Importing and Exporting an HMI

You can import an HMI app from the GoPxL interface. After importing an HMI app, you can then edit it in GoHMI Designer. After you have imported an HMI you can also launch it in a browser from the GoPxL interface by clicking **Start** and then **View live**.

⚠	Importing an HMI app overwrites the HMI app currently in GoHMI Designer. Consider exporting the HMI app from the GoPxL interface on a regular basis.

To back up an HMI you have created or to provide it to someone else, you can export it.

🗋	Imported and exported HMI apps are zip files.

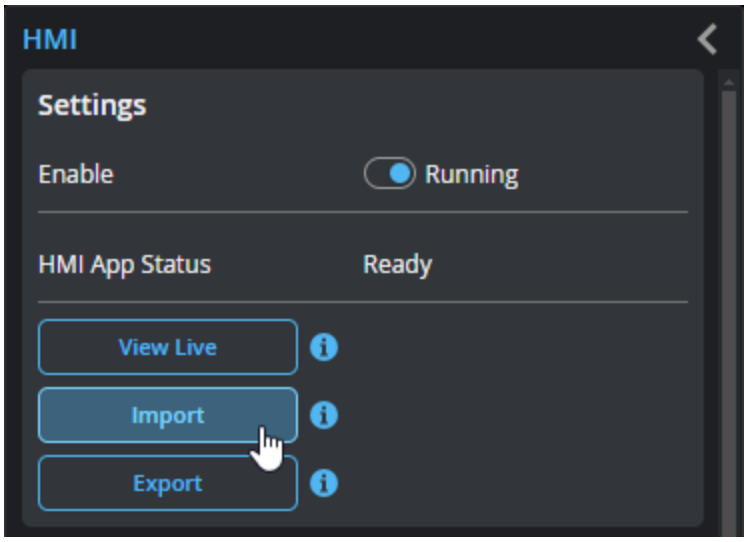
To import an HMI app

1. Make sure you have backed up the current HMI or that it is OK to overwrite it.

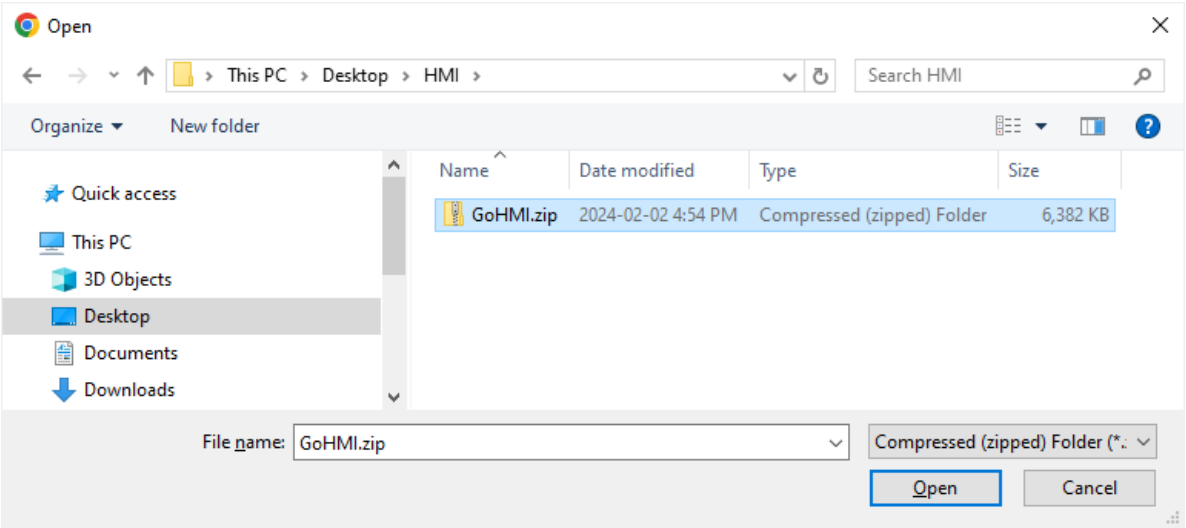
   Importing an HMI app overwrites the app in GoHMI Designer.

   For information on exporting an HMI, see *To export an HMI app* on the next page.

2. In the GoPxL interface, click **Import**.



3. In the Open dialog that displays, go to the location of the zipped HMI app project and click **Open**.

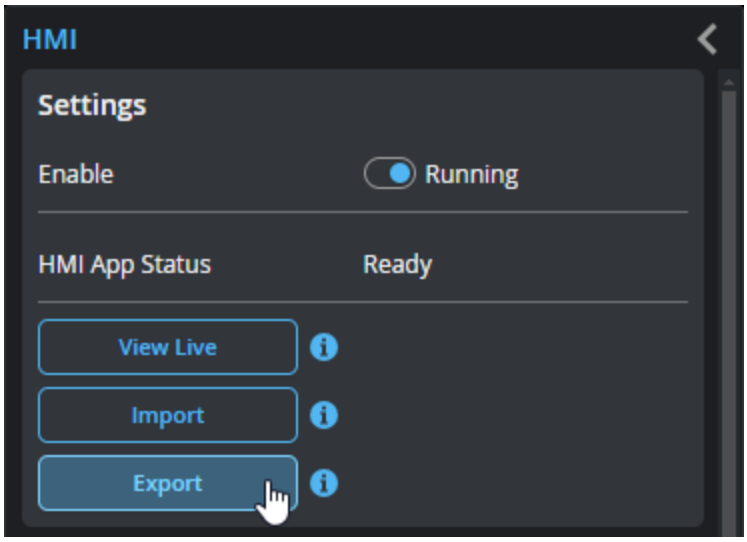


GoPxL imports the HMI app. The app is now available for editing in GoHMI Designer (see See *Opening, Editing, and Saving a Project* on page 439).

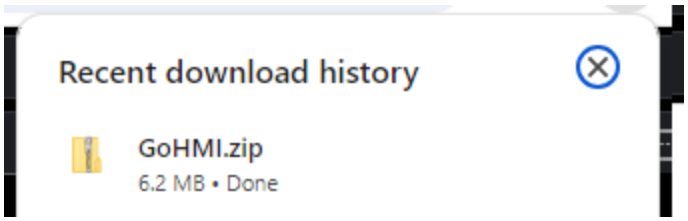
If you want to back up your HMI app or import it into another device or PC instance, you export it in GoPxL interface.

To export an HMI app

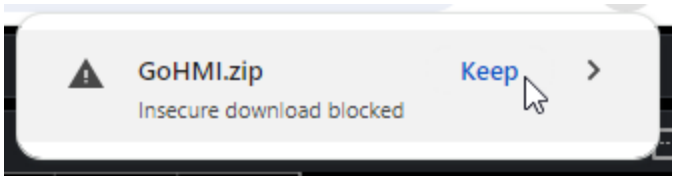
- In the GoPxL interface, click **Export**.



GoPxL downloads the HMI app and saves it as a zip file (GoHMI.zip).



You may need to tell your browser to accept the download (Chrome).



Using a Data Viewer in an HMI

You add a data viewer to an HMI by adding an IFrame widget in GoHMI Designer, and then copying a link from the GoPxL web interface (in the Displayed Outputs panel) and pasting it into the IFrame. The link tells the HMI the IP address of the sensor, as well as which sensor output and measurements it should display.

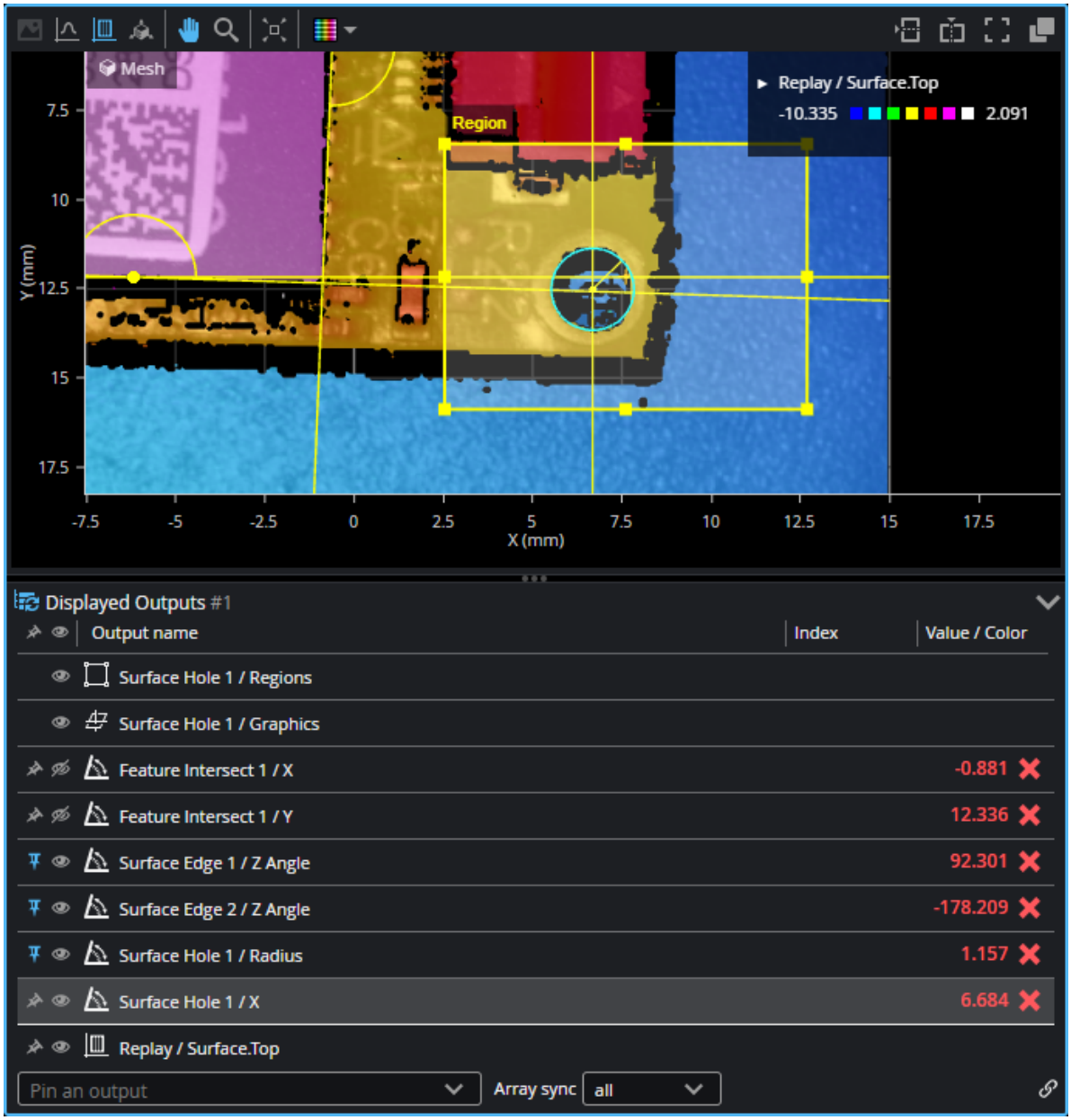
> The default HMI app includes an IFrame widget configured to display the data viewer. The IP address used in the link is 127.0.0.1:8100, which is valid for a PC instance of GoPxL (if one is running at that address and port).
>
> If you want to try out the default app with your sensor, follow the steps below to get the link from your sensor, and update the IFrame's **Source URL** field with that link (see steps 1 to 3, and step 8).

To add a data viewer to an HMI

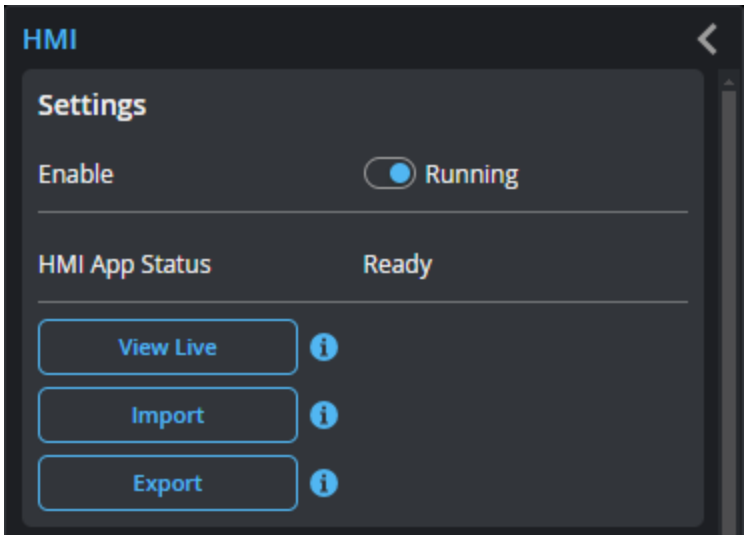
1. In GoPxL, set up a data viewer as required.

   For example, you can pin measurements so that they appear in the data viewer.

2. Under the data viewer window, make sure the Displayed Outputs pane is expanded.



3. Click the link icon ( ) in the lower right of the Displayed Outputs pane to copy the path for the data viewer.
4. Make sure GoHMI is enabled and configured in the **Communicate** > **HMI** page.

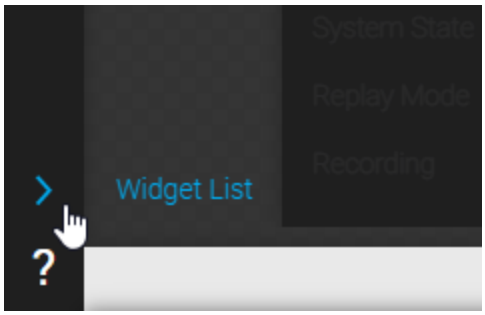


For more information, see *HMI* on page 373.

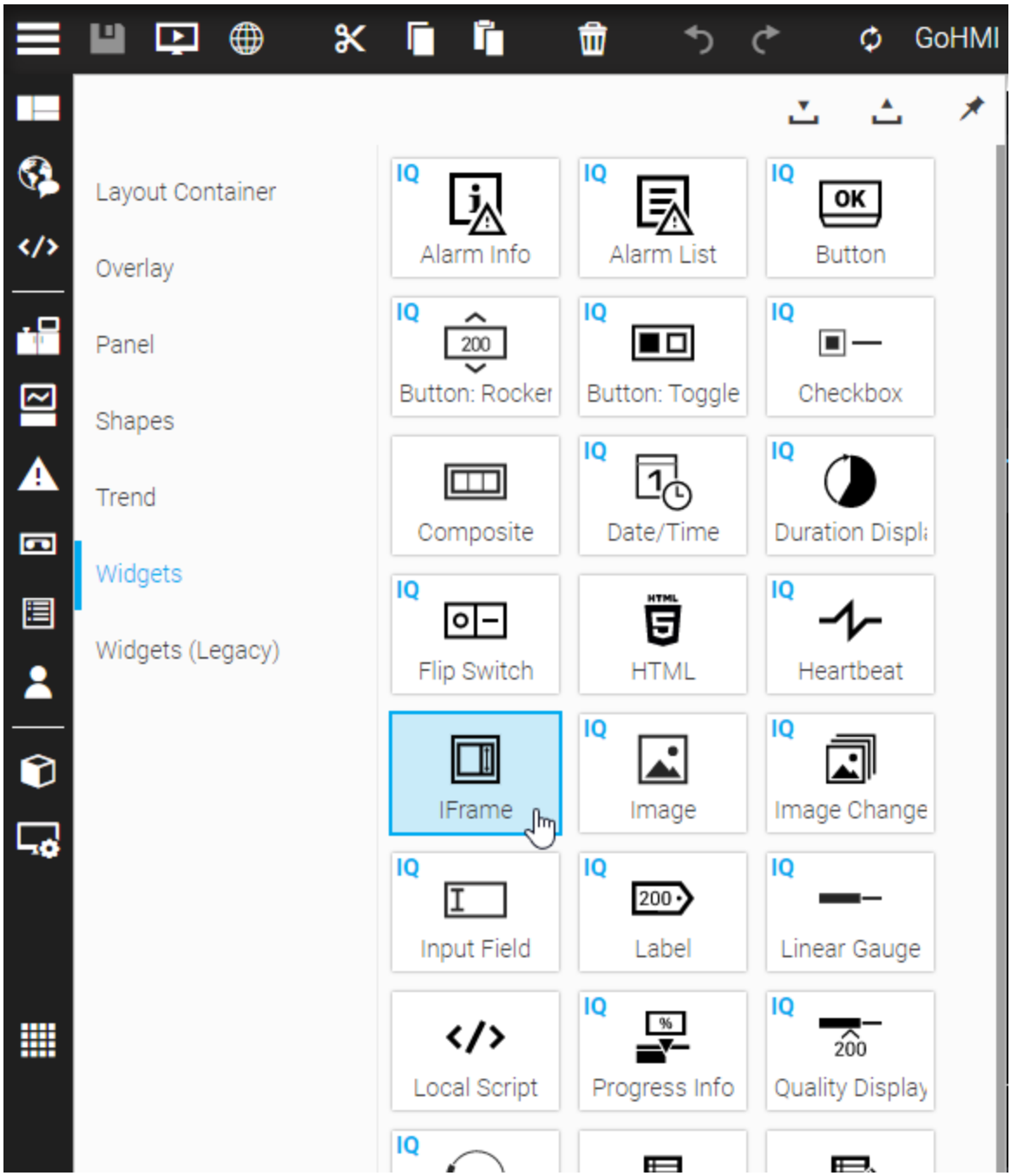
5. Launch GoHMI Designer.

   For information on launching GoHMI Designer, see *Launching GoHMI Designer* on page 435.

6. In GoHMI Designer, in the Layout Manager, expand the widget list.

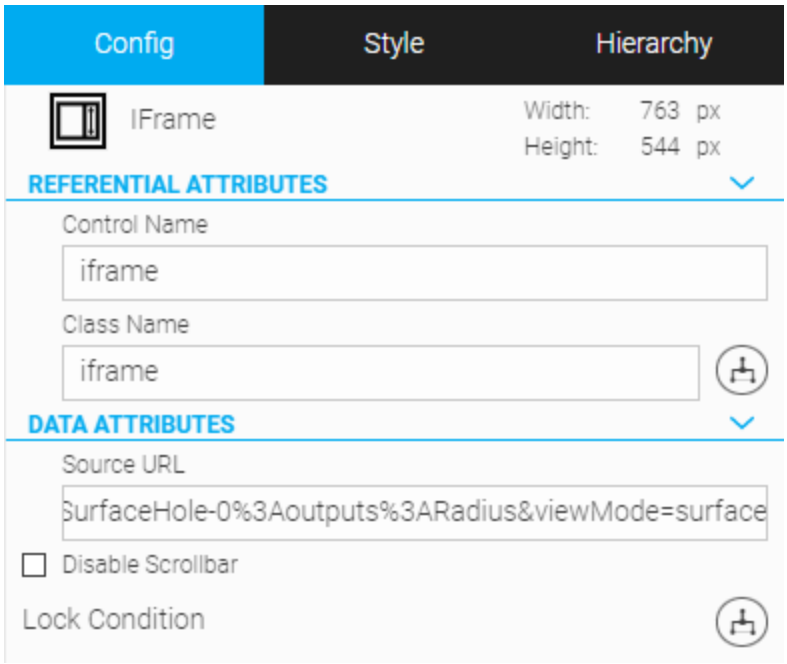


7. From the widget list, drag an IFrame widget into the HMI.



Resize the IFrame widget if necessary.

8. In the **Config** tab, paste the URL you copied in step 1 into **Data Attributes** > **Source URL**.

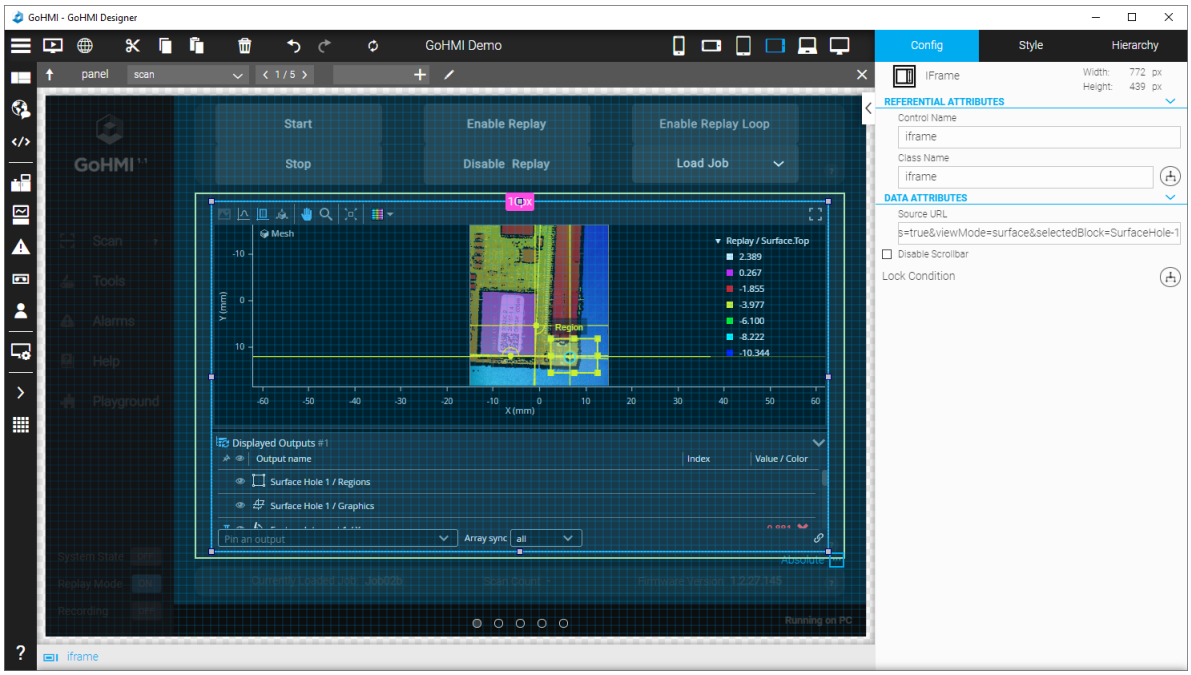


The URL will look something like this:

```
http://127.0.0.1:8100/?host=127.0.0.1&webPort=8100&visualizerOnly=true&displayedOutputs=Surface.Top,tools%3AFeatureIntersect-5%3Aoutputs%3AX,tools%3AFeatureIntersect-5%3Aoutputs%3AY,tools%3ASurfaceEdge-2%3Aoutputs%3AZAngle,tools%3ASurfaceHole-1%3Aoutputs%3AX&viewMode=surface
```

After a few seconds, the data viewer from the connected instance of GoPxL appears in the IFrame widget.

The following shows what the default HMI app looks like after updating the IFrame widget's source URL.



Limiting Flash Memory Write Operations

Several operations and REST API / SDK functions write to the sensor's flash memory. The lifetime of the flash memory is limited by the number of write cycles. Therefore it is important to avoid frequent write operation to the sensor's flash memory when you design your system with the SDK or GoHMI.

Power loss during flash memory write operation will also cause sensors to enter rescue mode.

GoPxL SDK and REST API

The GoPxL Software Development Kit (GoPxL SDK) includes open-source software libraries and documentation that you can use to programmatically access and control GoPxL devices.

> The GoPxL SDK and REST API use the term "scanner" instead of "sensor group." The two are interchangeable.

> The GoPxL SDK uses mm, mm^2, mm^3, hertz, microseconds, and degrees as standard units unless otherwise mentioned.

The following sections describe the Software Development Kit (GoPxL SDK).

Installation Guide

Location of GoPxL SDK and Documentation

The GoPxL SDK, and the SDK and REST API references are available in the `14630-x.x.x.x_SOFTWARE_GoPxL_SDK.zip` package.

The zip file contains both a C++ and a .NET SDK organized as follows:

- `GoPxL_SDK_Cpp\`
- `GoPxL_SDK_dotNet\`

Visual Studio Solution Files and Linux Makefile

For Windows, Visual Studio solution files are included for both the C++ and .NET SDKs.

- C++: `GoPxL_SDK_Cpp\Gocator`
- .NET: `GoPxL_SDK_dotNet\Gocator`

For Linux, first ensure that you have the g++ compiler package and then run the following command from the `GoPxL_SDK_Cpp\Gocator\GoPxLSdk` directory:

```
make -f GoPxLSdk-Linux_X64.mk
```

Note that only the C++ SDK is supported on Linux.

Building the SDK

To compile in Visual Studio, you may need to retarget the solution to the installed Windows SDK version. To retarget a solution, open the solution in Visual Studio, right-click the solution in the Solution Explorer, choose **Retarget solution** in the context menu. In the dialog box that appears, select the desired version from the drop-down list and click **OK**.

The compiled Linux output libraries can be found in the `GoPxL_SDK/lib/linux_x64` directory.

Running SDK Applications

When running an SDK application, be aware that performance may vary significantly between Debug and Release configurations. The Debug configuration disables optimizations and includes additional debugging information, which may result in slower execution times, specifically during asynchronous data retrieval.

System Requirements

The GoPxL SDK can be compiled using the following environment.

For Windows:

- Visual Studio 2017 or later
- Windows SDK (included with Visual Studio installation)

For Linux:

- GCC X64 compiler, version 7.5.0 (GccX64_7.5.0-p3)
- GNU Make

Accessing Documentation

The full SDK class references and REST API resources are available at the following locations:

GoPxL SDK Class Reference

C++: `GoPxL_SDK_Cpp\doc\class-reference.html`

.NET: `GoPxL_SDK_dotNet\doc\class-reference.html`

REST API Resources

C++: `GoPxL_SDK_Cpp\doc\api-protocol-manual.html`

.NET: `GoPxL_SDK_dotNet\doc\api-protocol-manual.html`

Sample Projects

Sample projects showing how to perform various operations are provided, each one demonstrating a specific area. Examples are located in the following folders:

- C++: `GoPxL_SDK_Cpp\Gocator\GoPxLSdk\samples`
- .NET: `GoPxL_SDK_dotNet\Gocator\GoPxLSdkNet\samples`

Header Files

When including header files from the GoPxL SDK, reference the GoPxLSDK directory as the source directory. For example:

```
#include <GoPxLSdk/GoRestClient.h>.
```

Additionally, note that the SDK header files reference files from the kApi directory. For example:

`#include <kApi/kApiDef.h>`.

Getting Started with the SDK

GoPxL offers a simple programming API interface for connecting, configuring, and acquiring data from sensors and the GoMax vision accelerator.

> Example code shown below is for C++.

All GoPxL SDK programs should have the sequence described below, at a minimum.

Connecting a Device

The following sequence assumes you know the IP address of the device. If you do not know the IP address, or if the address is dynamic, an application can discover the devices that are available on the network using the GoPxL SDK discovery capability. For more information, see *Discover Devices* on the next page.

1. Initialize the SDK library

   For C++ SDK programs only, construct the SDK library by calling `GoApiLib_Construct`. You must call `GoApiLib_Construct` before any API functions are used. (.NET SDK programs don't require this.)

```
// Constructs GoPxL API core framework.
GoApiLib_Construct(&goApiLib)
```

   After completing your operations with the SDK library, don't forget to release the resources using `kDestroyRef()`.

```
// Releases resources and resets the object handle to kNULL
kDestroyRef(&goApiLib)
```

2. Instantiate a `GoSystem` object and connect the device.

```
// Define device IP address and control port
constexpr const char* DEVICE_IP_ADDRESS = "192.168.1.10";
constexpr const uint16_t CONTROL_PORT = GO_PXL_SDK_DEFAULT_CONTROL_PORT; // Defaults is 3600


// Parse device IP address
kIpAddress deviceIpAddress;
kIpAddress_Parse(&deviceIpAddress, DEVICE_IP_ADDRESS);

// Create a GoSystem object
GoSystem system;

// Set up connection
system.SetAddress(deviceIpAddress);
system.SetControlPort(CONTROL_PORT);

// Connect to the device
system.Connect();
```

GoSystem Class Member Functions

The `GoSystem` class represents a single Gocator system. A Gocator system consists of one or more sensors/devices. Below are the member functions of the GoSystem class

```
// SetAddress: Sets the IP address used for connecting to the device.
void SetAddress(const kIpAddress& ipAddress);

// Address: Gets the IP address used for connecting to the device.
kIpAddress Address() const;

// SetControlPort: Sets the control port used for connecting to the device.
void SetControlPort(k16u port);

// ControlPort: Gets the control port used for connecting to the device.
k32u ControlPort() const;

// Connect: Connects to the device.
void Connect();

// Disconnect: Disconnects from the device.
void Disconnect();

// IsConnected: Checks if the device is connected.
bool IsConnected();

// Start: Starts the device.
void Start();

// Stop: Stops the device.
void Stop();

// RunningState: Gets the current state of the device.
State RunningState();

// Client: Returns a reference to the REST API client for performing operations.
GoRestClient& Client();

// SensorPath: Retrieves the REST API resource path of a sensor based on its serial number.
ResourcePath SensorPath(SerialNum serialNum);
```

Discover Devices

If a device IP address is not known (for example, because it is assigned through DHCP), you can discover the devices available on the network using the GoPxL SDK discovery function:

1. Include the `GoDiscoveryClient` header file.
2. Create an instance of `GoDiscoveryClient` and call the `BlockingDiscover` function with a specified timeout.
3. Access the list of discovered instances.

```
#include "GoDiscoveryClient.h"
auto discovery = std::make_unique<GoDiscoveryClient>();
```

```
discovery->BlockingDiscover(3000);
auto& instances = discovery->InstanceList();
```

4. After discovering the instances, you can loop through the instances to get IP addresses and other instance-related information.

```
// Print instance IP address and GDP port.
for (const auto& instance : instances) {
   std::cout << "IP Address: " << kIpAddress_AsString(instance.GetIpAddress()) << std::endl;
   std::cout << "GDP Port: " << instance.GetGdpPort() << std::endl;
}
```

For more information, see the *Discover* sample.

Configure Device Settings

You configure GoPxL device settings, which are represented as JSON resources, through JSON-based REST API commands. The `GoRestClient` class provides access to read and write settings. You can use methods like Call, Read, Update, Create, and Delete, among others, to interact with the device settings.

Start by instantiating a `GoSystem` object. Through this object, you can directly access the `GoRestClient` instance via `system.Client()`. Then, to perform `Read()` or other operations, simply continue chaining the methods, for example, `system.Client().Read()`.

These methods follow a similar format:

```
"command /path/to/callable/resource {'argument':'value'}"
```

For example the following command:

```
call /jobs/commands/rename { "sourceName" : "job1", "destName" : "renamedJob1" }
```

can be implemented as follows:

```
// Rename job.
auto payload = GoJson(R"({
    "sourceName" : "SDK-demo",
    "destName" : "SDK-demo-job"
})");

system.Client().Call("/jobs/commands/rename", payload);
```

For the full list of resources available, see the *GoPxL 1.1 - REST API Resources and User Guide*:

C++: `GoPxL_SDK_Cpp\doc\api-protocol-manual.html`

.NET: `GoPxL_SDK_dotNet\doc\api-protocol-manual.html`

It's important to understand that GoPxL device settings are divided into two general types:

- sensor-level: These settings only apply to individual sensors.
- scanner-level: These settings apply to all of the sensors in a group of sensors (a "scanner").

To help explain the functions, some sections below include definitions of the API resources.

Accessing Sensor Resources

The most common use case is modifying settings that apply to individual sensors, such as exposure or active area.

If you know a device's serial number, you can access the corresponding resource using the `SensorPath` function. You can then modify the settings under the sensor's resources.

```
// Update single exposure value.
auto sensorPath = system.SensorPath(sensorSerialNumber);
auto payload = GoJson(R"({
    "parameters" : {
        "exposureSettings" : {
            "exposureMode" : 0,
            "singleExposure" : 400
        }
    }
})");

system.Client().Update(sensorPath, payload);
```

Accessing Scanner Resources

You access system-level settings (those that apply to a group of sensors) through the `scanner` resources.

If you know the engine ID and scanner ID (available in the REST API reference documentation), you can use the following to access the sensor group and modify the trigger setting:

```
// Access and modify trigger settings for the scanner.
auto scannerPath = "/scan/engines/LMILaserLineProfiler/scanners/scanner-0";
auto payload = GoJson(R"({
    "parameters" : {
        "triggerSettings" : {
            "source" : 3
        }
    }
})");

system.Client().Update(scannerPath, payload);
```

Configuring scanner resources through the SDK is less common. These resources are related to physical setup. Normally, you configure these settings through the user interface during the initial system installation and setup, and the settings don't need to change for different production runs.

Accessing an Array Element

Some resources are returned as an array. One example is the `visibleSensors` resource, which can have multiple elements.

The following returns an array of sensors under the `/scan/visibleSensors` resource path.

```
sensors = system.Client().Read("/scan/visibleSensors/").GetResponse().Content().At
("sensors");

// Get the first element in the array
auto sensor = sensors.Begin();
auto sensorSerialNumber = sensor.Value().At("/serialNumber").Get<std::string>();

// Alternatively, iterate through each sensor in the array
for (auto sensor = sensors.Begin(); sensor != sensors.End(); ++sensor)
    auto sensorSerialNumber = sensor.At("/serialNumber").Get<std::string>();
}
```

Receive Device Data

GoPxL streams out acquired data (such as Profile and Surface data, and measurements) using the GoPxL Data Protocol (GDP). To receive the data, you use a `GoGdpClient` instance, which you first need to connect to the device. You can use `GoPxLSdk::MessageType` to identify the type of data received and the associated stamp information. Stamps contain encoder and trigger timing information, current time, and frame index, and are grouped into datasets with their corresponding data.

Data can be received on the same thread used for controlling the sensors (that is, synchronously) or on a different thread (that is, asynchronously). Note that data is received as the `GoDataSet` type. For more information on `GoDataSet`, see *GDP Data Types* on page 458.

For synchronous data retrieval, use the `ReceiveData()` method:

1. Instantiate a GoPxL Data Protocol client and connect to a device.

```
#include <GoPxLSdk/GoGdpClient.h>

constexpr k16u GDP_PORT = GO_PXL_SDK_DEFAULT_GDP_SERVER_PORT; // Default port is 3601

// Instantiate a GoGdpClient object
auto gdpClient = std::make_unique<GoGdpClient>();

// Connect the GoGdpClient to the device using sensor IP and GDP port
gdpClient->Connect(SENSOR_IP, GDP_PORT);
```

2. Receive data.

```
// Establish a synchronous data transfer with a 3-second timeout.
gdpClient->ReceiveDataSync(3000);

```

```
// Retrieve the received data set.
const GoDataSet& receivedDataSet = gdpClient->DataSet();
```

Then loop through each message (a unit of data received from the GoPxL sensor) to handle the data based on its type (Profile, Surface, Measurements, and so on) using the `MessageType` enum. For example:

```
for (size_t i = 0; i < receivedDataSet.Count(); ++i)
{
    switch (receivedDataSet.GdpMsgAt(i).Type())
    {
        case GoPxLSdk::MessageType::STAMP:
            // Handle stamp message
            ...

        case GoPxLSdk::MessageType::UNIFORM_SURFACE:
            // Handle uniform surface data
            ...
    }
}
```

For more information on synchronous data retrieval, see the `ReceiveProfile` and `ReceiveSurface` samples.

For asynchronous data retrieval, you use the following steps:

1. Instantiate a GoPxL Data Protocol client and connect to a device.
2. Start the data reception thread

   Call `GoGdpClient::ReceiveDataAsync(func)` to initiate data reception asynchronously. This starts a background thread (`receiveThread`) to receive and deserialize messages continuously.
3. Handle received data.

   Upon receiving individual data messages, they are reassembled into a `GoDataSet` object. The data reception continues until you call `GoGdpClient::Close()`.
4. Process the data.

   The received data sets are added to a message queue (`Go::MsgQueue<GoDataSet> dataQueue`). A separate thread (`dataThread`) removes the data sets from the queue for processing.
5. Access the data

   Define a callback function to handle the received data sets.

The following is an example of retrieving data asynchronously:

```
void onData(const GoDataSet& receivedDataSet)
{
    for (size_t i = 0; i < receivedDataSet.Count(); i++)
    {
        switch (receivedDataSet.GdpMsgAt(i).Type())
        {
            case GoPxLSdk::MessageType::STAMP:
            {
                const GoGdpStamp& stampMsg = (const
```

```
    GoGdpStamp&)receivedDataSet.GdpMsgAt(i);
                // Process stamp message
                ...
            }
            // Handle other message types as needed
        }
    }
}

// Connect to GDP and start receiving data asynchronously
gdpClient->Connect(SENSOR_IP, GDP_PORT);
gdpClient->ReceiveDataAsync(onData);
```

For details, see the `ReceiveAsync` sample.

The following lists the `GoGdpClient` class methods you can use to interact with the GoPxL Data Protocol (GDP) server.

```
// Constructs GoGdpClient.
GoGdpClient();

// Destructs GoGdpClient.
~GoGdpClient();

// Connects to the GdpServer.
void Connect(kIpAddress ipAddr, k16u port);

// Closes the TCP connection.
void Close();

// Receives data synchronously from the GDP server.
void ReceiveDataSync(k64u receiveTimeoutInMilliseconds);

// Receives data asynchronously from the GDP server.
void ReceiveDataAsync(std::function<void(const GoDataSet& receivedDataSet)>& func);

// Gets the pointer to the received data set over the Goctor Data Protocol.
const GoDataSet& DataSet() const;
```

Receive Health and Metrics Information

Health information, along with various metrics, is transmitted using the REST API. To handle received metrics data, set up a callback function as shown below:

```
// Callback function for handling received metrics data.
static void onMetrics(const GoStreamResponse&)
{
    // Extract payload & content from the received data
    auto content = notification.Payload();
  int appUptime = content.Get<int>("/appUpTime");
  ...
}

```

```
// Set up the callback function for receiving metrics data asynchronously.
system.Client().SetStreamHandler(onMetrics);

// Start the stream to receive metrics data from the specified path
system.Client().StartStream("/system/metrics");

// Stop the stream and disconnect from receiving metrics data, call StopStream
system.Client().StopStream("/system/metrics");
```

For more information, see the `ReceiveHealth` sample.

Accelerated Devices

Control of devices accelerated by GoMax NX or a PC instance of GoPxL uses the same function API. The only difference is that you must use the IP address of the GoMax NX device or the PC running GoPxL when connecting to the device (that is, in `system->Connect()`).

GDP Data Types

Value Types

The GoPxL SDK is built on a set of basic data structures, utilities, and functions, which are contained in the *kApi* library.

The following basic value types are used by the *kApi* library.

Value Data Types

Type	Description
k8u	8-bit unsigned integer
k16u	16-bit unsigned integer
k16s	16-bit signed integer
k32u	32-bit unsigned integer
k32s	32-bit signed integer
k64s	64-bit signed integer
k64u	64-bit unsigned integer
k64f	64-bit floating number
kBool	Boolean, value can be kTRUE or kFALSE
kStatus	For the definitions of this type, in the folder containing the GoPxL SDK, see Platform\kApi\kApi\kApiDef.h.
kIpAddress	IP address

Output Types

Data outputs are encapsulated as data messages in the GoPxL SDK. The following message types are available in the SDK.

Output Types

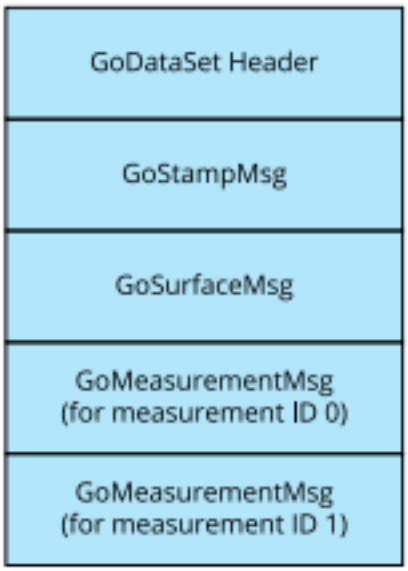
Type	Description
GoGdpBoundingBox	Bounding Box results output based on part matching results
GoGdpFeaturePoint	Feature point output
GoGdpFeaturePlane	Feature plane output
GoGdpFeatureLine	Feature line output
GoGdpFeatureCircle	Feature circle output
GoGdpImage	Video image
GoGdpMeasurement	Measurement output
GoGdpMesh	Mesh data output
GoGdpNull	Contains no valid data. Received when a measurement tool's data output doesn't generate data when triggered. This messages allow user to synchronize message from different outputs based on number of message received.
GoGdpProfilePointCloud	Profile point cloud data (non-uniform spacing)
GoGdpProfileUniform	Uniform spacing / resampled profile data
GoGdpRendering	Graphics objects associated with a tool outputs*
GoGdpSpots	Detected spots when overlay exposure is enabled*
GoGdpStamp	Acquisition stamp
GoGdpSurfacePointCloud	Surface point cloud data (non-uniform spacing)
GoGdpSurfaceUniform	Uniform spacing / resampled surface data

*Not commonly used in a production system.

All of the messages above are extended from `GoGdpMsg`. For more information, see *GoDataSet* below.

GoDataSet

Data is passed to the data handler in a `GoDataSet` object. The `GoDataSet` object is a container that can hold any type of data, including scan data (sections or surfaces), measurements, or results from various operations. Data inside the `GoDataSet` object is represented as messages. The following illustrates the content of a `GoDataSet` object of a Surface mode setup with two measurements.



The following lists the available GoDataSet types:

```
enum struct MessageType : k16u {
    SIGNAL                  = 1,    // Signals that data on a stream is invalidated.
    NULL_TYPE               = 10,   // Null data type that contains a status code.
    STAMP                   = 11,   // Stamp information.
    UNIFORM_PROFILE         = 12,   // Uniform profile data.
    PROFILE_POINT_CLOUD     = 13,   // Raw profile data.
    UNIFORM_SURFACE         = 14,   // Raw surface data.
    SURFACE_POINT_CLOUD     = 15,   // Raw surface data.
    IMAGE                   = 16,   // Image data.
    SPOTS                   = 17,   // Spot data associated with the image.
    MESH                    = 18,   // Mesh data.
    MEASUREMENT             = 19,   // Measurement data.
    RENDERING               = 70,   // Graphical data associated with an output.
    POINT_FEATURE           = 71,   // Point feature data.
    LINE_FEATURE            = 72,   // Line feature data.
    PLANE_FEATURE           = 73,   // Plane feature data.
    CIRCLE_FEATURE          = 74,   // Circle feature data.
    HEALTH                  = 100   // Health data.
};
```

Measurement Values and Decisions

There are two outputs for each measurement: a value and a decision. A measurement value is returned as a 64-bit number.

The measurement decision specifies the state of the measurement value as follows:

Measurement Decisions

Decision	Description
1	The measurement value is between the maximum and minimum decision values (inclusive). This is a pass decision.

Decision	Description
0	The measurement value is outside the maximum and minimum. This is a fail decision.

Limiting Flash Memory Write Operations

Several operations and REST API / SDK functions write to the sensor's flash memory. The lifetime of the flash memory is limited by the number of write cycles. Therefore it is important to avoid frequent write operation to the sensor's flash memory when you design your system with the SDK or GoHMI.

Power loss during flash memory write operation will also cause sensors to enter rescue mode.

Python GDK

The GoPxL Python GDK lets you develop custom measurement and processing tools that you can incorporate into your application. Once integrated into a GoPxL or GoMax firmware, the tools behave like the built-in ones.

As with the Script tool, you use Python to add functionality and custom algorithms to GoPxL. However, using the Python GDK to create tools has these advantages:

- You can expose parameters to control the algorithm.
- Your intellectual property is better protected, if you bundle the custom tool in a firmware.
- You can debug your Python code in an IDE that supports Python.

GoPxL includes a Python interpreter and tool server that provide support for Python-based tools and the Script tool. When you use GoPxL on PC, the Python engine is started by default as part of launching a GoPxL instance.

When you develop custom tools for GoPxL, you can use an external Python interpreter, instead of the included one, and optionally a virtual environment. Doing this is useful if you will develop tools using specific libraries or environments, for example, implementing an AI program with the GDK.

Requirements and Installation

If you are using the distribution of Python provided with the GDK, there is nothing to install. If you are using a Python distribution installed on your PC, LMI recommends version 3.8.10, as later versions have not been tested with the GDK.

You should also install the following packages using `pip install`.

- numpy
- opencv-python (optional, currently 4.5.4.60)

You can get the Python GDK by downloading and unzipping the GoPxL Utilities package (14631-1.2.x.x_SOFTWARE_GoPxL_Utilities.zip, available on LMI's Product Downloads page). The Python GDK is under the `GoPxL/plugins/GdkPy` folder.

The following lists the files you will typically need to use or modify in the `/plugins/GdkPy` folder:

- `configure_developer_start.bat`: Disables the built-in Python server in PC instances of GoPxL. When the built-in server is disabled, the `run_server.py` program (see below) provides Python GDK tools to any PC instance run from the parent folder. You *must* run this batch file before starting a PC instance to debug custom tools, or if you need to use your own distribution of Python to run the tools. .

- `configure_built_in_start.bat`: Enables the built-in Python server in PC instances of GoPxL. You *must* run this script to re-enable the internal server before starting an instance of GoPxL if you are not using your own distribution of Python.
- `run_developer_server.bat`: Starts the Python tool server program (run_server.py) from the GDK's Python distribution. You should use this to start the Python tool server, instead of `run_server.py`.
- `run_server.py`: The Python tool server program, which loads the tools in the `plugins/GdkPy/user_tools/` (and `plugins/GdkPy/samples/`) and sets up logging. The script runs in the Python environment that executes it. You typically run this program from within a debugger, but you can also run it from the command line to run tools that use your own Python distribution. For information on debugging, see *Debugging Custom Tools* on page 467. For information on deploying tools, see *Deploying Your Custom Tools* on page 469.
- `gdk` folder: This folder contains the `tool.py` and `tool_api_def.py` files. Together, these files provide the tool API and data types you use to create custom tools, respectively. You should familiarize yourself with the content of these files.
- `samples` folder: This folder contains several sample Python GDK tools. You should copy these into the files into the `user_tools` folder as a starting point for your own custom tools.
- `standard_tools` folder: This folder contains the Script tool and the library it uses. You do not need to modify these files.
- `user_tools` folder: You should put your custom Python GDK tools and any required libraries in this folder. When you run `run_server.py`, the tool server loads these tools.
- `utilities` folder: This folder contains utilities you use to add a custom tool to a GoPxL firmware or GoPxL for PC. For more information, see *Deploying Your Custom Tools* on page 469.

Developing Custom Tools

For full code completion such as IntelliSense and debugging support, we recommend using Visual Studio Code or an IDE that supports Python. In the following, we use Visual Studio Code.

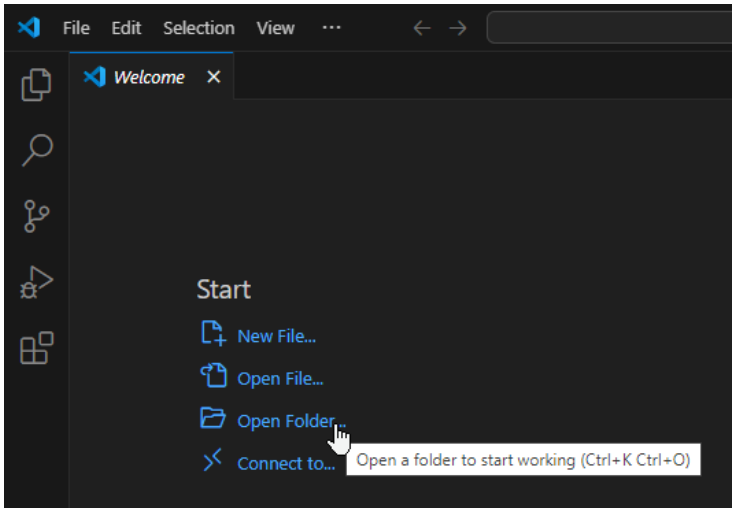
For information on setting up debugging, see *Python GDK* on the previous page.

> OpenCV is only available on GoPxL running on a PC and on GoMax. To use OpenCV in a tool, you must use `import cv2` in your tool's Python source code. Gocator sensors do not support the OpenCV module in Python tools.

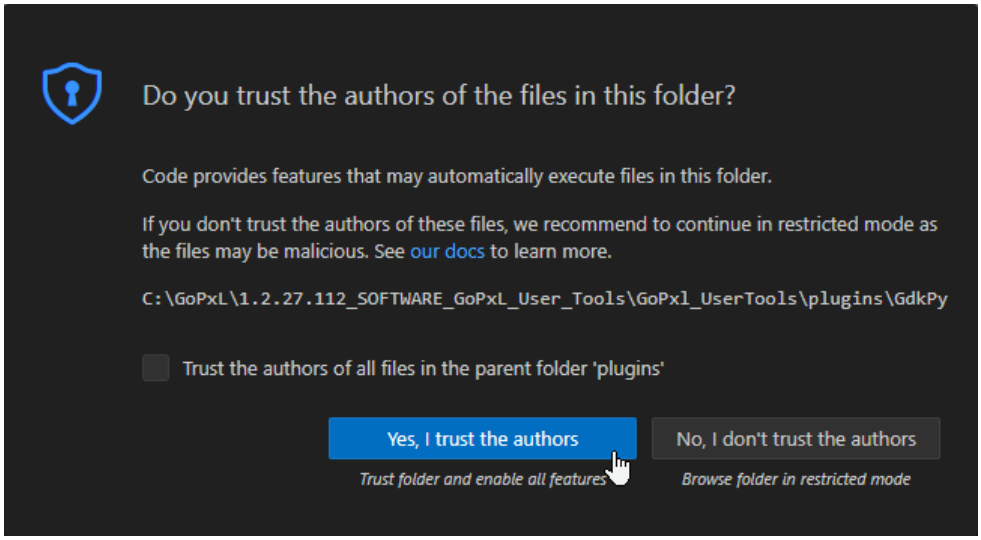
To create a Python GDK tool

1. If you have not already done so, install Visual Studio Code, start the program, and install the Python Extension.

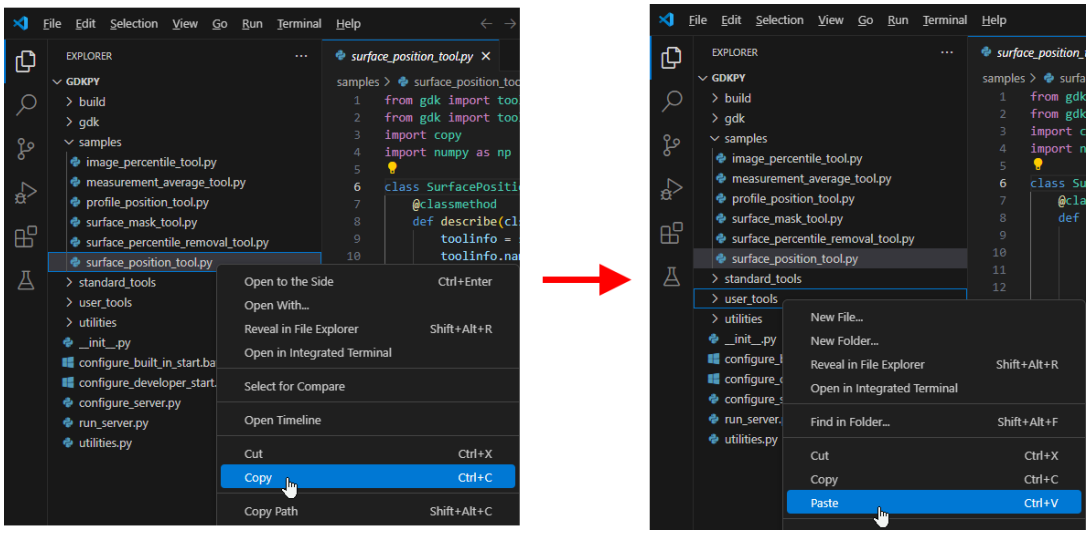
2. Open the `plugins\GdkPy` folder in the IDE.



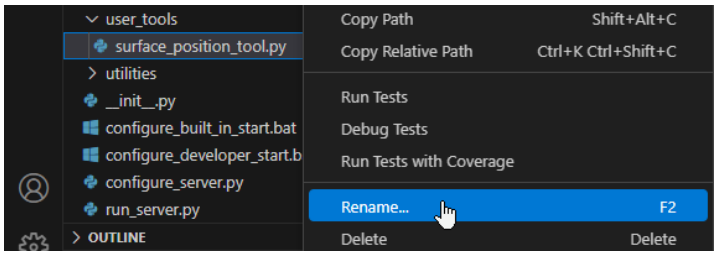
You may need to confirm that you trust the authors of the files in the folder.



3. Copy a file from the `samples` folder into the `user_tools` folder and rename it.

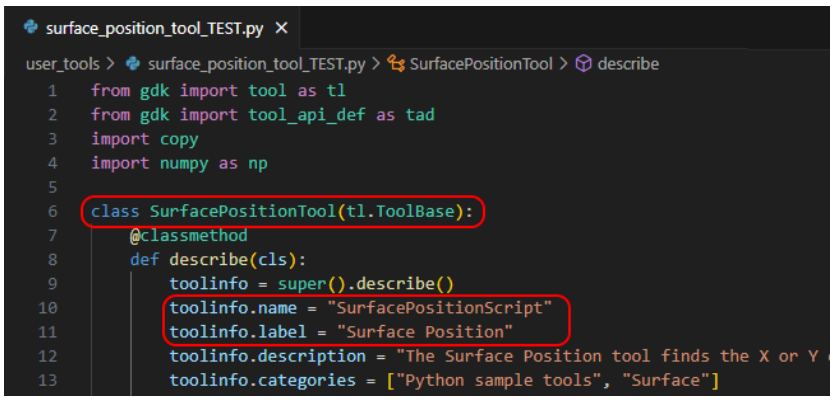


You can use any name for the tool's Python file, but using the main input type as a prefix is a useful convention, for example, "surface_" for Surface tools.



4. Modify the tool's code.

To start (and to avoid conflicts), you should change the name of the class everywhere in the code, as well as the strings assigned to `toolinfo.name` and `toolinfo.label` in the `describe` method.



You also define the tool's inputs, anchors, outputs, and parameters in the `describe` method. Note that the order of the items of a given category (that is, inputs, parameters, outputs, or anchors) in the code is the order in the tool.

```
class SurfacePositionTestTool(tl.ToolBase):
    @classmethod
    def describe(cls):
        toolinfo = super().describe()
        toolinfo.name = "SurfacePositionTestScript"
        toolinfo.label = "Surface Position Test"
        toolinfo.description = "Surface Position Test is for exploring the Python GDK API."
        toolinfo.categories = ["Python sample tools", "Surface"]

        # Inputs
        toolinfo.inputs["Input"] = tl.InputInfo("Surface", tl.DataType.UNIFORM_SURFACE)

        # Anchors
        toolinfo.anchor = tl.AnchorInfo(x=True, y=True, z=True, z_angle=False)

        # Outputs
        toolinfo.outputs["X"] = tl.OutputInfo("X", tl.DataType.MEASUREMENT_VALUE_X)

        toolinfo.outputs["Y"] = tl.OutputInfo("Y", tl.DataType.MEASUREMENT_VALUE_Y)

        toolinfo.outputs["Z"] = tl.OutputInfo("Z", tl.DataType.MEASUREMENT_VALUE_Z)

        toolinfo.outputs["Point"] = tl.OutputInfo("Position", tl.DataType.FEATURE_POINT)

        # Parameters
        toolinfo.parameters["UseRegion"] = tl.ParamInfo("Use Region", tl.ParamType.BOOL, True)
        toolinfo.parameters["Region"] = tl.ParamInfo(
            "Region",
            tl.ParamType.SURFACE_REGION,
            tl.Region3d(0, 0, 0, 5, 5, 5),
        )

        # feature position options
        cls.position_options = {0: "Average", 1: "Median", 2: "Centroid", 3: "Min X", 4: "Max X",
                                5: "Min Y", 6: "Max Y", 7: "Min Z", 8: "Max Z"}
        toolinfo.parameters["Features"] = tl.ParamInfo(
            "Features", tl.ParamType.INT, 0, options=cls.position_options
        )

        return toolinfo
```

You should closely examine the `tool.py` and the `tool_api_def.py` files in the `gdk` folder (which form the GDK's API), as well as the source code of the sample tools, to better understand how to develop your own tools. Observing how Python GDK tools work using a Python debugger can also help you develop your own tools.

5. (Optional) Place any additional modules your tool needs to import in the `user_tools` folder.

Enabling the Local Python Tool Server

Before you can start debugging a custom tool, or run a custom tool on PC using your own distribution of Python, you must run the `configure_developer_start.bat` batch file. This batch file disables the Python tool server built into GoPxL for PC.

After this, you must run the `run_developer_server.bat` script to start the Python tool server manually. If you want to debug a tool, run `run_server.py` instead.

For information on starting the tool server for debugging purposes, see *Debugging Custom Tools* on the next page.

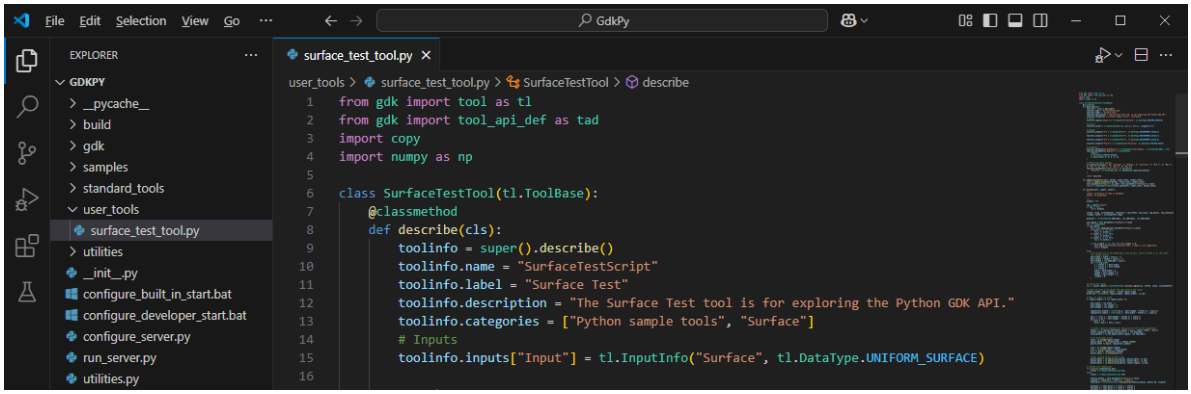
For information on starting the tool server with your own distribution of Python, see *Deploying Your Custom Tools* on page 469.

Debugging Custom Tools

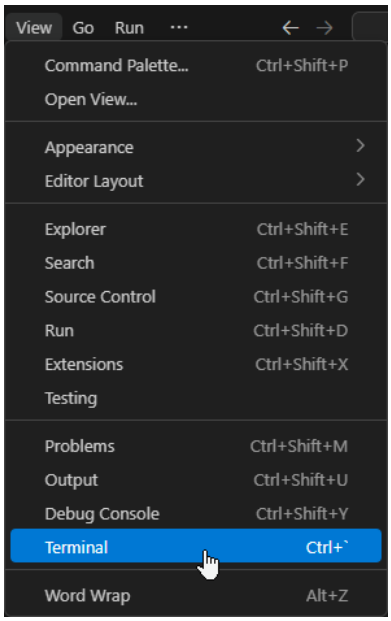
You can debug a Python GDK tool using an IDE that supports Python.

To debug your Python tool using Visual Studio Code

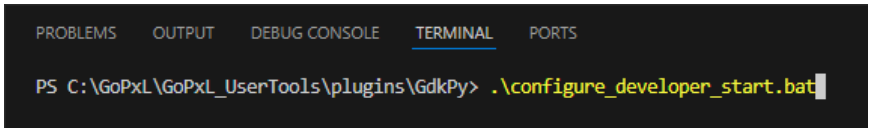
1. Open the `plugins\GdkPy` folder in the IDE.



2. Open the terminal from the **View** menu or by pressing Ctrl+` (Ctrl + back tick).

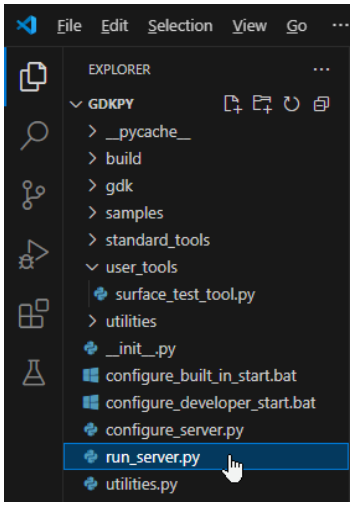


3. At the prompt, type `.\configure_developer_start.bat`.



This disables the Python tool server built into GoPxL for PC. If you do not run this script, you will not be able to debug your tool.

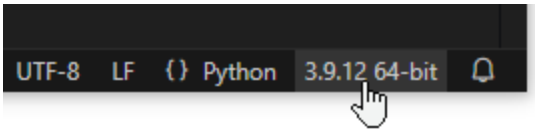
4. From the Explorer view, open the `run_server.py` file.



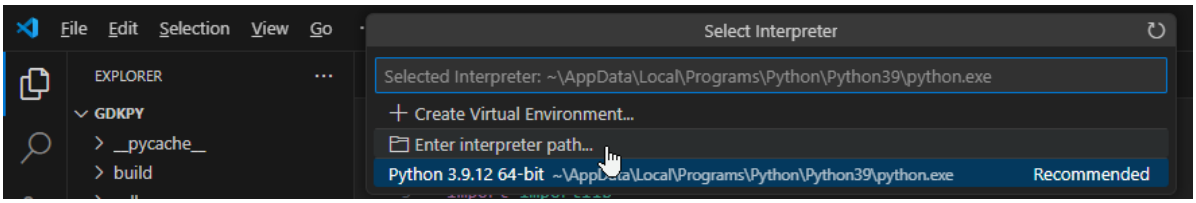
5. (Optional) If Python is not installed on your PC, you can point Visual Studio Code to the Python interpreter used by the internal tool server. To change the interpreter, do the following:

   a. At the lower right, click the Python version in the bottom right corner.

   If you have previously used Python in Visual Studio Code, you may see a version already selected.

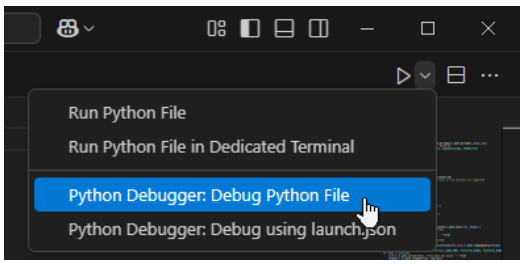


   b. At the top of the editor, click **Enter interpreter path**, and provide the path to the built-in interpreter.

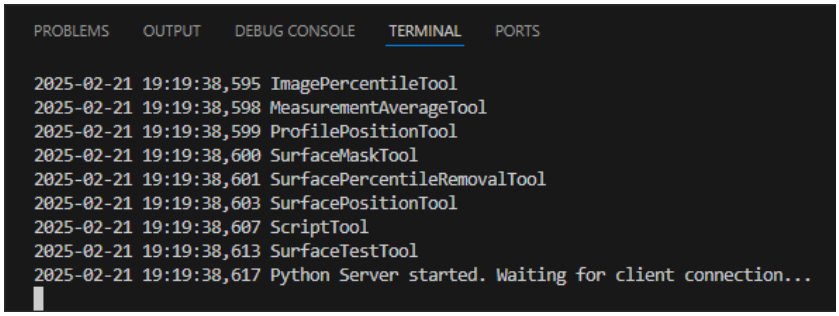


   The built-in Python interpreter is `bin\win64\python\python3.exe`.

6. Start debugging run_server.py.

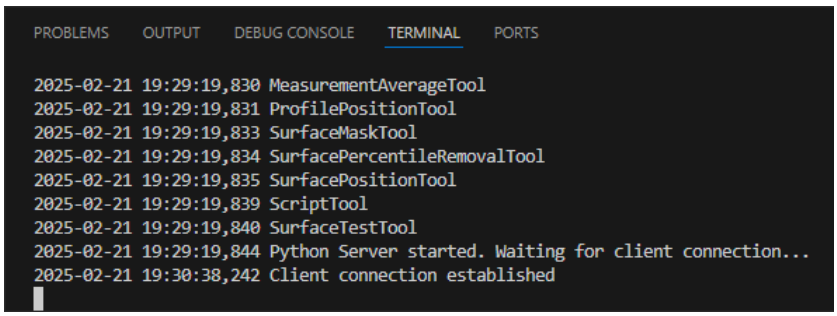


The tool server starts and waits for a connection from a PC instance of GoPxL.



7. Start an instance of GoPxL.

   The instance of GoPxL connects to the tool server.



8. Upload some previously scanned data or scan some new data and set the tool's input to this data.

   Your tool's input `must` be set in order to debug it.

9. Set a break point in your tool's Python code.

   The `process` method is good point to start: the tool's main processing occurs here.

Deploying Your Custom Tools

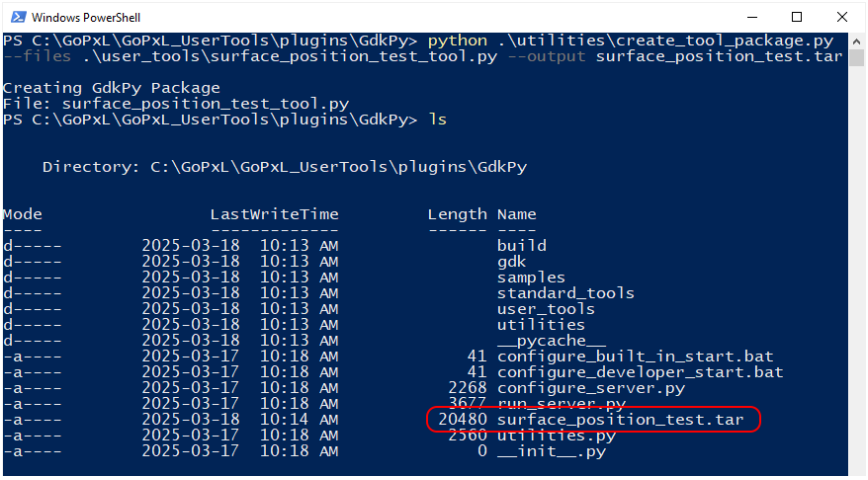
After you have finished developing your tool, you can deploy it as part of either a GoPxL firmware or GoPxL for PC. You use the `GdkPy\utilities\create_tool_package.py` script to create a .tar tool package, which you add to GoPxL using the GoPxL Add-on Tool Manager.

The `create_tool_package.py` script accepts the following arguments:

Arguments

Name	Description
-h, --help	Show the help.
--files	The files to include in the .tar tool package. You can also pass a folder containing multiple Python tools. Files are added recursively.
--output	The name of the tool package with a .tar extension. If you omit this argument, the tool is packaged as `my_tool.tar`.
--ignore	Files or wildcards to ignore. By default, *.pyc files are excluded.

1. Open a command prompt in the folder `plugins\GdkPy` and run `\utilities\create_tool_package.bat` (which calls `create_tool_package.py`), providing the path to your tool.



The .tar file is created in the folder from which you started the script.

2. Use the GoPxL Add-on Tool Manager utility to add the generated `.tar` tool package to a firmware (`.dat` file) or to GoPxL for PC (in the `GoPxL` folder, where the bin folder is).

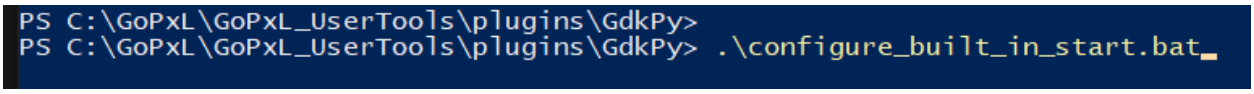
   For more information on using this utility, see *GoPxL Add-on Tool Manager* on page 482.

3. If you have added the custom tool to GoPxL for PC, open a command prompt and go to the `GdkPy` folder.

   a. Do one of the following:

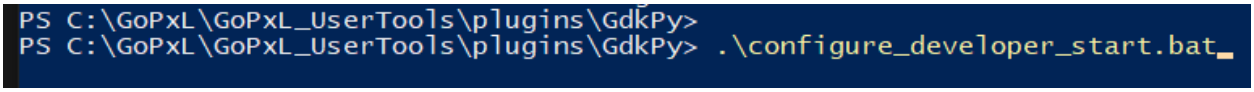
   Using the Python distribution included with GoPxL for PC:

   Run `.\configure_built_in_start.bat`.

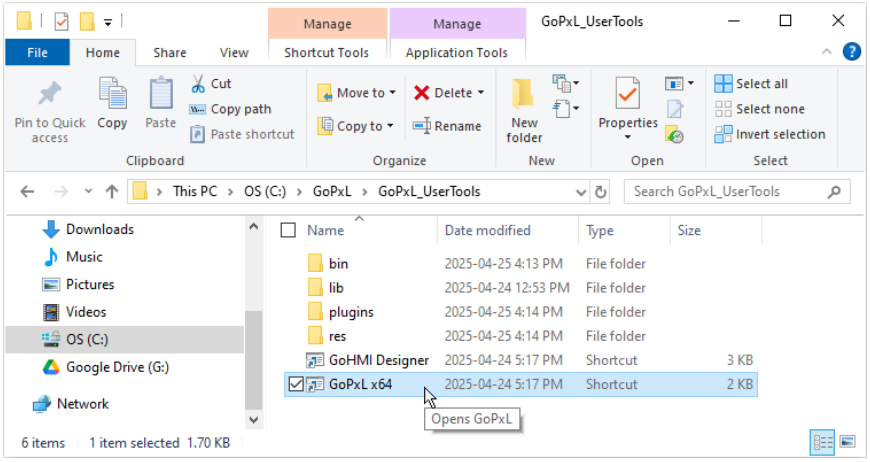


   Using your own Python distribution:

   Run `.\configure_developer_start.bat`.



   b. Go to the main GoPxL_UserTools folder, and start GoPxL.



4. If you have added the custom tool to a firmware, upload it to a device.

For more information updating a sensor's firmware, see *Software Upgrade* on page 400.

Integrations

GoPxL currently provides three types of integrations that you configure within GoPxL: GoPxL Data Protocol (for use with the GoPxL SDK and API), industrial protocols, and GoHMI.

For information on the Gocator protocol, see *GoPxL Data Protocol* on page 315.

For information on the protocols supported by GoPxL, see *Protocols (PLCs and other hardware)* below.

For information on GoHMI, see *GoHMI and GoHMI Designer* on page 425.

Protocols (PLCs and other hardware)

GoPxL supports various protocols for communicating with sensors over Ethernet (TCP/IP).

The following protocols are available:

- GoPxL Data Protocol
- Modbus
- EtherNet/IP
- PROFINET
- ASCII

> The connection mappings for the industrial protocols (PROFINET, EtherNet/IP, and Modbus) have changed to correct a byte alignment issue. To allow communication between PLCs and GoPxL 1.2, you may need to update PLC logic created for GoPxL 1.1.
>
> For EtherNet/IP, the EDS file has been updated to reflect the changes.

The connection mappings for the industrial protocols (PROFINET, EtherNet/IP, and Modbus) have changed to correct a byte alignment issue. To allow communication between PLCs and GoPxL 1.2, you may need to update PLC logic created for GoPxL 1.1.

For EtherNet/IP, the EDS file has been updated to reflect the changes.

Utilities

The following sections describe some of the utilities tools provided with a Gocator sensor, as well as the CSV format that a sensor can export. For information on the integrations available with a sensor, see *Integrations* on page 472.

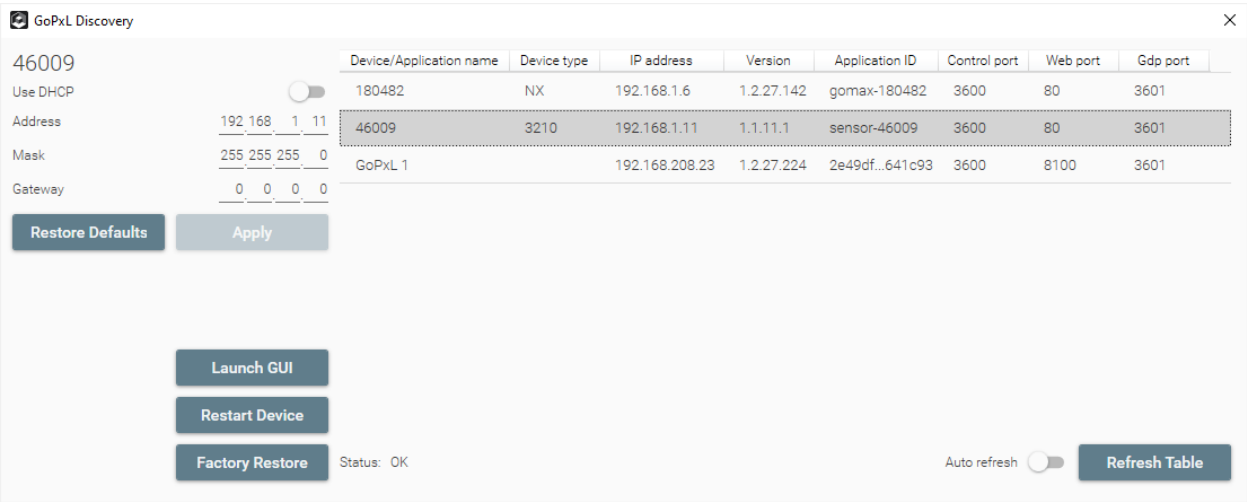
You can get the tools in the utilities package (`14631-1.2.30.5_SOFTWARE_GoPxL_Utilities.exe`) from the downloads area of the LMI Technologies website: https://lmi3d.com/product-downloads/. Run the installer, and provide a location where the content of the package will be unzipped.

The following tools are available:

- Discovery tool: Used to find devices on a network. See *GoPxL Discovery Tool* below.
- Replay Utils tool: Lets you convert between .gprec files and .srf/.sur/.pcd/.pro files. It also lets you convert .gprec files to CSV files. For more information, see *Replay Utility Tool* on page 476.
- Rescue tool: TBD.

GoPxL Discovery Tool

The GoPxL Discovery tool lets you perform various operations with sensors running GoPxL, PC instances of GoPxL, and GoMax NX units. The tool also lets you see sensors running Gocator Classic firmware, and launch their user interface in a browser, for example, to make it easier to upgrade a sensor from Classic to GoPxL.



You can do the following with a sensor running GoPxL:

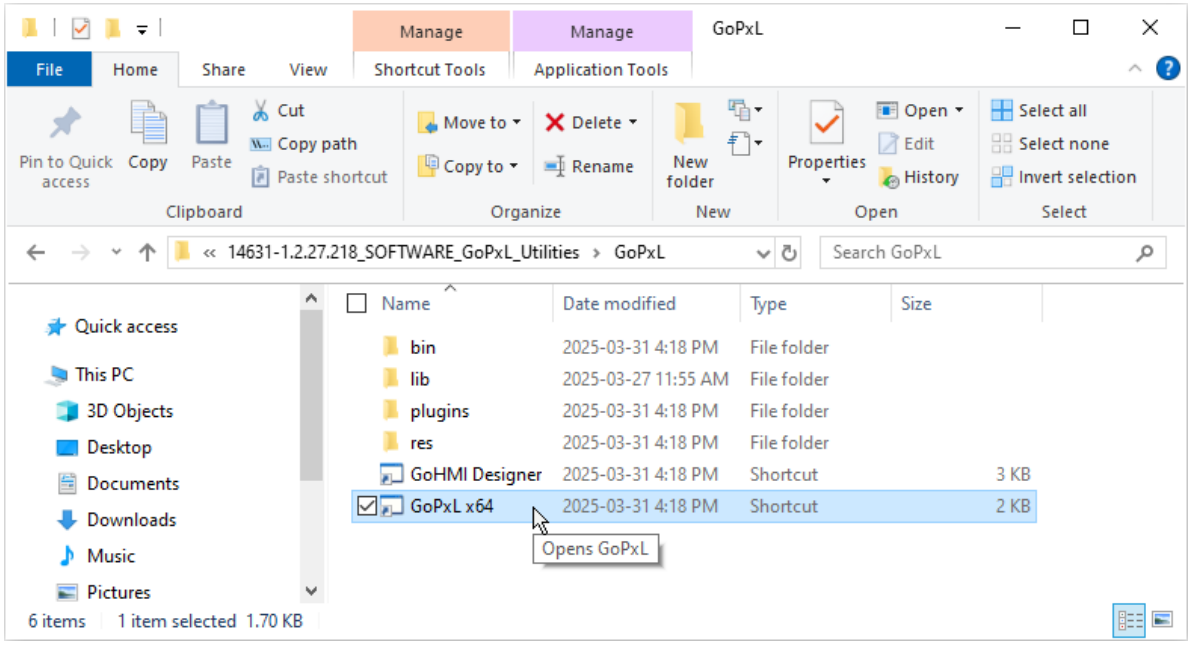
- Find a sensor if you have forgotten its IP address.
- View port information.
- Set the network configuration (IP address, mask, and so on) of a sensor, and also restore these values to the default.
- Launch the GUI of a sensor in your browser.
- Restart the sensor.
- Perform a factory restore on a sensor.

You can do the following with a PC instance of GoPxL or a GoMax NX:

- Find its IP address.
- View port information.
- Launch the GoPxL of the PC instance or GoMax NX in your browser.

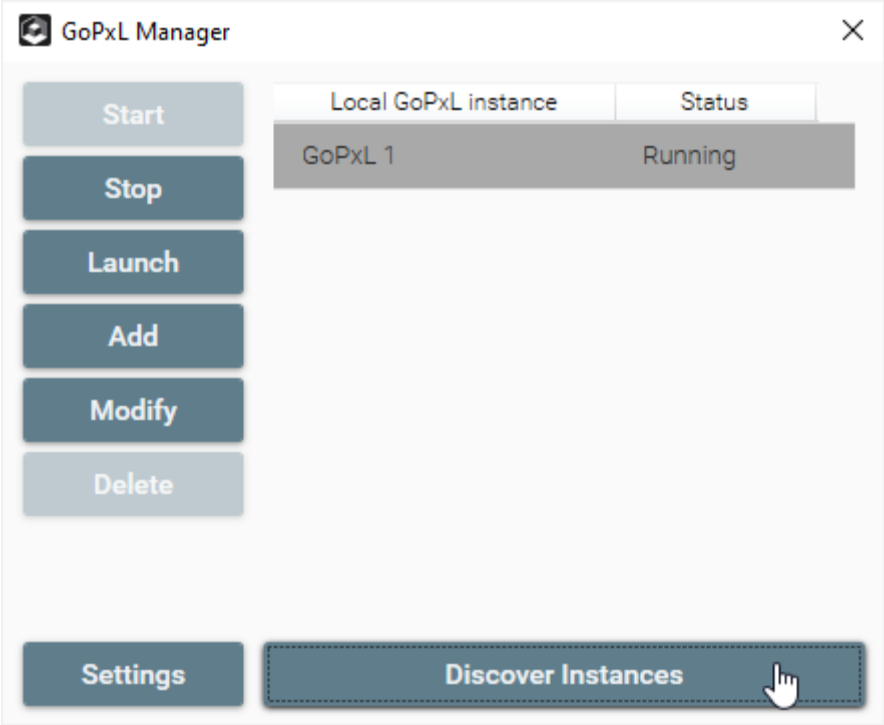
To run the GoPxL Discovery tool

1. If it isn't already running, start the Windows version of GoPxL.

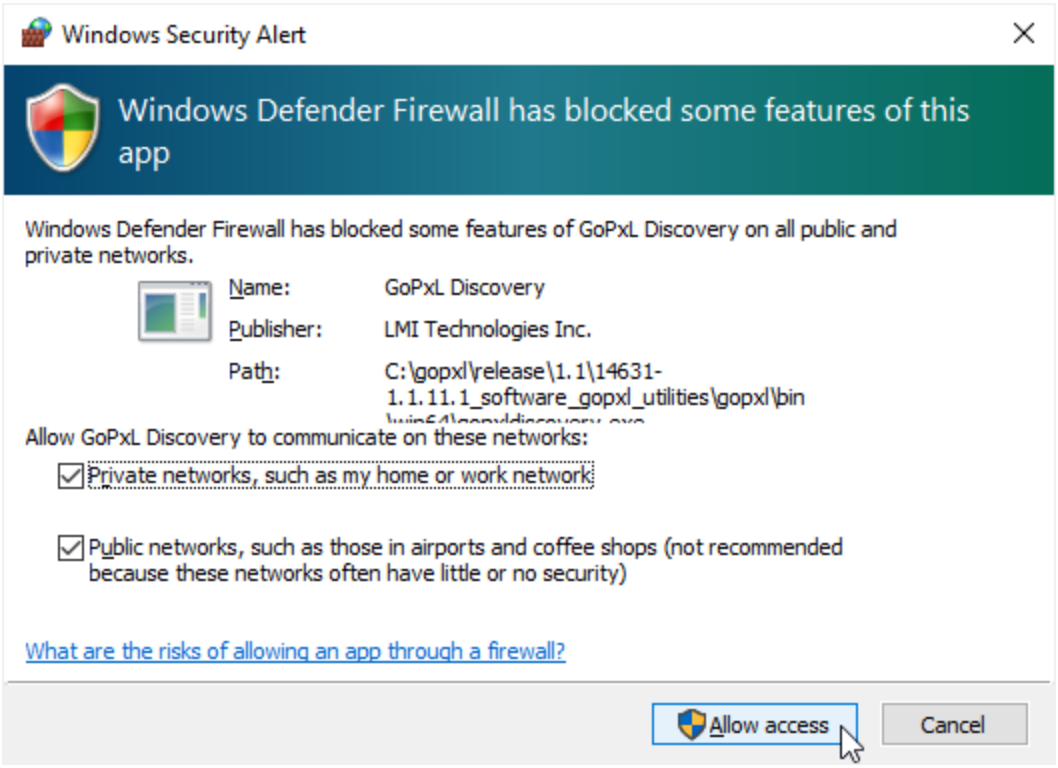


   GoPxL Manager opens.

2. In the manager tool, click **Discover Instances**.



Your firewall may ask if you want to allow the GoPxL Service Application and Python to communicate over your network. To avoid issues, LMI recommends allowing communication over both Public and Private networks. Because the notifications appear one on top of the other, make sure to check the appropriate boxes and carefully click **Allow access** for both alerts.



The GoPxL Discovery tool opens.

When the GoPxL Discovery tool opens, it lists the sensors running GoPxL and the PC instances of GoPxL that are currently visible on the network. Sensors are identified in the **Device/Application Name** column by their serial number, and in the **Device Type** column by their model. If a sensor is running through a PC instance of GoPxL, its status (at the bottom of the tool) is "Device is remotely connected", and you can't perform any operations on it from the tool. Otherwise, the sensor's status is "OK".

GoPxL instances are identified in the **Device/Application Name** column by their name.

The tool provides important network-related information, namely, IP addresses and Web, Control, and GDP ports. The Web port is the port used to connect to the GoPxL interface in a browser. For sensors, the default port is 80; you don't need to specify this in the URL when connecting directly to a sensor. The Control port and the GDP ports are used by the SDK and the REST API. (For an introduction to these, see *GoPxL SDK and REST API* on page 449.)

If a sensor or GoPxL instance is missing, click the **Refresh Table** button. Optionally, you can also enable the **Auto refresh** toggle; the table refreshes .

To perform operations on a sensor or a GoPxL instance, click its row in the tool and then perform the operation.

Replay Utility Tool

The Replay Utility tool (ReplayUtils.exe) is a console application that lets you do the following conversions with scan data. You can run it from a command line using Windows Command Prompt or Windows PowerShell.

- .gprec <=> .srf/.sur/.pcd (valid for surface recording data)
- .gprec <=> .pro (valid for profile recording data)
- .gprec <=> .csv (valid for profile/surface recording data)
- folder of .gprec files <=> .gprec file (valid for surface/profile/image/video data)

Intensity information is included when converting to and from .pcd and .srf files. Intensity is not supported when converting to and from the .sur format.

You can get the tool in the utilities package (`14631-x.x.x.x_SOFTWARE_GoPxL_Utilities.zip`) from the downloads area of the LMI Technologies website: https://lmi3d.com/product-downloads/.

After downloading the utilities package, unzip the file. Then, open a command prompt or PowerShell instance in the folder containing the utility. To open a command line quickly, in File Explorer go to the folder containing the tool, hold Shift, right-click in empty area, and choose "Open command window here" or "Open PowerShell window here."

Run ReplayUtils.exe (under Tools\ReplayUtils). You must either copy the .gprec file into the folder containing the utility, or include its entire path in the call to the utility, for example, `ReplayUtils.exe -i \path\to\file\myData.gprec -f 0 -o myOutput.sur`.

The tool provides the following parameters:

Parameters

Name	Description
--help	Show the help.
-i --input	Specify the file name of the input file or the input directory. Required.
-f --frame	Specify the *zero-based index* of the frame to be exported in the recording data. For example, to export the first frame in the scan data, use "-i 0"; for the second frame, use "-i 1". (Default is 0.)
-w --width	Specify the width of the exported surface. (Only valid for importing .pcd files.)
-h --height	Specify the height of the exported surface. (Only valid for importing .pcd files.)
-sxz --swapXZ	Whether to swap the X and Z coordinates. (Only valid for importing .pcd files.)
-syz --swapYZ	Whether to swap the Y and Z coordinates. (Only valid for importing .pcd files.)
-z --zoom	Specify the zooming factor of the exported surface. (Only valid for importing .pcd files.)
-r --remove	Remove the specific point. (Only valid for importing .pcd files.)
-a --all	Export all the frames if the import is recording data.
-o --output	Specify the file name of the output file. If the imported file isn't recording data and the exported file name isn't specified, the utility uses the imported file name as the exported file name.

CSV File Format

An exported CSV file contains a series of sections. Each section begins with a row containing the name of the section, and ends with a row containing the string "End". An empty line separates each section.

Each section usually contains one or more subsections. Each subsection has a header row containing a list of field names, followed by one or more rows of data. There is usually no empty line between the subsections.

Other structures within sections are possible.

The "Part" section is unrelated to part detection. It represents Surface data when **Enable uniform spacing** is checked.

Info

This section contains basic system information. It has one header row and one value row. The fields are described below:

Info Fields

Field	Description
CSV Version	Version of the CSV file format.
Sensor Count	Number of sensors in the system.
Trigger Mode	Trigger source: 0 – Time 1 – Encoder 2 – Digital input 3 – Software
Trigger Rate	Frame rate for time trigger (Hz).
Trigger Delay Domain	Output delay domain: 0 – Time (µs) 1 – Encoder (mm)
Trigger Delay	Output delay (µs or mm, depending on delay domain defined above).
Operation Mode	The scan mode.
XResolution	System X resolution (mm).
YResolution	System Y resolution (mm).
ZResolution	System Z resolution (mm).
YSpeed	Y Speed (mm/s).
Layout	Sensor orientation: 0 – Normal (single-sensor system) / Wide (dual-sensor system) 1 – Opposite 2 – Reverse 3 – Grid

DeviceInfo

This section contains information about each device in the system. There is one header row, and one value row per device.

Device and firmware information (the ID, Model, and Version fields) is not currently output.

DeviceInfo Fields

Field	Description
ID	Device serial number
Model	Device part number
Version	Firmware version

Field	Description
Exposure Mode	Exposure mode: 0 – Single exposure 1 – Multiple exposures 2 – Dynamic exposure
Exposure 0 through Exposure 4	Multiple exposures
Exposure Min	Dynamic exposure min
Exposure Max	Dynamic exposure max
FOV X	Active area X
FOV Y	Active area Y
FOV Z	Active area Z
FOV Width	Active area width
FOV Height	Active area length (Y). (Note difference in terminology.)
FOV Depth	Active area height (Z). (Note difference in terminology.)
Transform X	Transform X offset (mm)
Transform Y	Transform Y offset (mm)
Transform Z	Transform Z offset (mm)
Transform X Angle	Transform X Angle (degrees)
Transform Y Angle	Transform Y angle (degrees)
Transform Z Angle	Transform Z angle (degrees)

RecordingFilter

Recording filters are not currently supported.

Profile

This section describes uniform (or resampled) profile data, which is produced when the sensor is in Profile mode and uniform spacing is enabled. It has two sub-sections: attributes and data.

The attribute section has only one row of data.

Attribute Section Fields

Field	Description
Frame Count	Total number of frames
Column Count	Number of columns
X Offset	X offset (mm)
Y Offset	Y offset (mm)
Z Offset	Z offset (mm)

The data section has one or more rows of data per frame (for example, range and intensity).

Data Section Fields

Field	Description
Frame	Frame index
Source	Source (for example, 0 for Top)
Time	Stamp time
Encoder	Stamp encoder
Z Encoder	Stamp encoder Z
Inputs	Stamp inputs
Exposure	Stamp exposure (µs)
Y	Y value (mm)
Axis	Axis: Z (range) or I (Intensity)
(x values)	Each column in the sub-header is a resampled X position Each column in the data is the range (mm) or intensity (count)

RawProfile

This section describes point cloud profile data (or unresampled / raw data), which is produced when the sensor is in Profile mode and uniform spacing is disabled. It has two sub-sections: attributes and data.

The attribute section has only one row of data.

Attribute Section Fields

Field	Description
Frame Count	Total number of frames
Column Count	Number of columns
X Offset	X offset (mm)
Y Offset	Y offset (mm)
Z Offset	Z offset (mm)

The data section has one or more rows of data per frame (for example, range and intensity).

Data Section Fields

Field	Description
Frame	Frame index
Source	Source (for example, 0 for Top)
Time	Stamp time
Encoder	Stamp encoder
Z Encoder	Stamp encoder Z
Inputs	Stamp inputs
Exposure	Stamp exposure (µs)
Y	Y value (mm)

Field	Description
Axis	Axis: X, Z, or I (Intensity)
(x values)	Each column in header is an index. Each column in data is the X/Z value (mm) or intensity (count)

Part

This section describes uniform Surface data, which is produced when the sensor is in Surface mode and uniform spacing is enabled.

> Only the data for the frame currently selected in the UI is exported when you export part data to a CSV file.

The section has two sub-sections: attributes and data.

The attribute section has only one row of data.

Attribute Section Fields

Field	Description
Frame	Frame index
Source	Source (for example, 0 for Top)
Time	Stamp time
Encoder	Stamp encoder
Z Encoder	Stamp encoder Z
Inputs	Stamp inputs
Row Count	Number of rows
Column Count	Number of columns
X Offset	X offset (mm)
Y Offset	Y offset (mm)
Z Offset	Z offset (mm)

The data section contains the data of a single surface scan. Each data row corresponds to one Y position. The first row contains the X values, and the first column contains the Y values. The region inside contains the range values (mm) for the corresponding row and column.

SurfacePointCloud

This section describes point cloud data (unresampled surface data), which is produced when the sensor is in Surface mode and uniform spacing is disabled. It has two sub-sections: attributes and data.

The attribute section has only one row of data.

Attribute Section Fields

Field	Description
Frame	Total number of frames

Field	Description
Source	Source (for example, 0 for Top)
Time	Stamp time
Encoder	Stamp encoder
Z Encoder	Stamp encoder Z
Inputs	Stamp inputs
Row Count	Number of rows
Column Count	Number of columns
X Offset	X offset (mm)
Y Offset	Y offset (mm)
Z Offset	Z offset (mm)

The data section contains the data of a single surface scan. The first row (header) can be ignored.

The data are (x,y,z) tuples expanded into a flat list of values, for example:

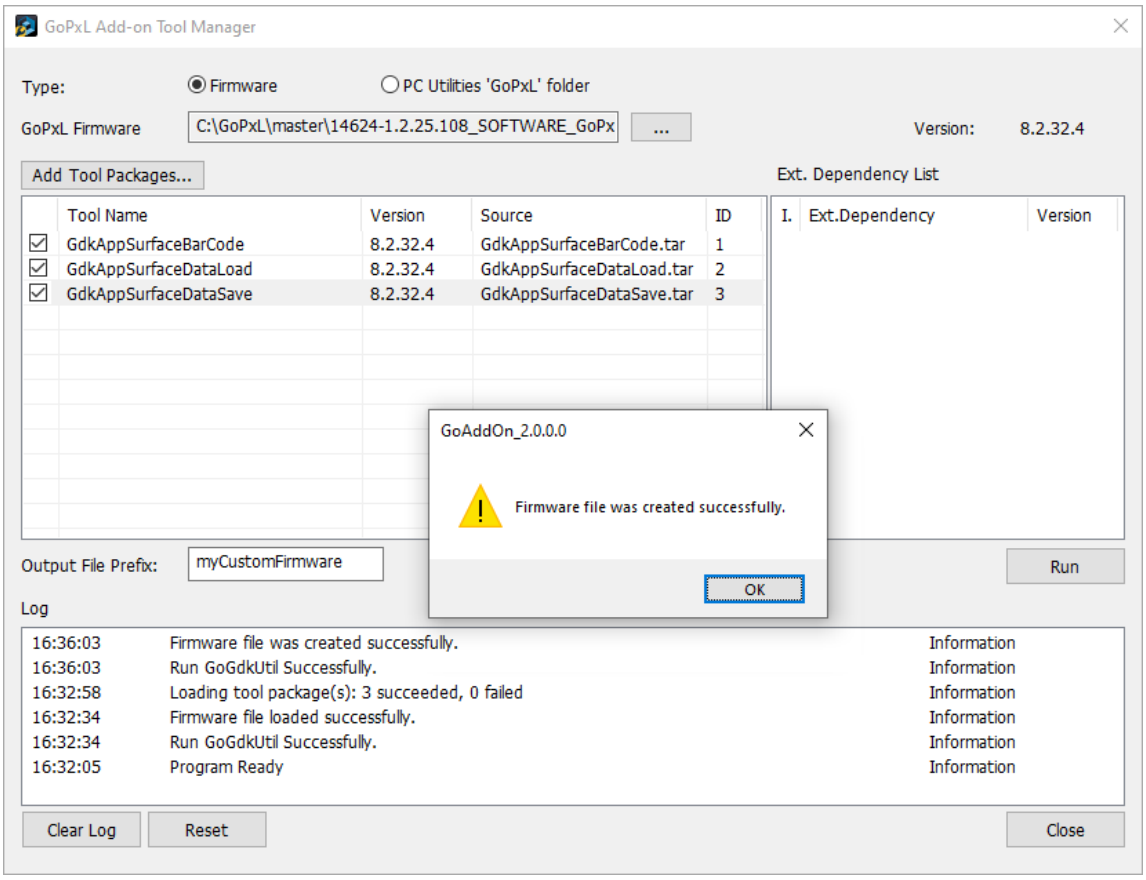
```
p0x, p0y, p0z, p1x, p1y, p1z, ..., pnx, pny, pnz
p(n+1)x, p(n+1)y, p(n+1)z, ...
...
```

Because the data has multiple rows and columns, it forms a rectangular grid of (x,y,z) tuples. The rows and columns do not correspond exactly to X and Y values, but suggest adjacency. i.e. positions with consecutive row indices or column indices are generally adjacent to each other in (x,y,z) coordinates.

The values are provided in mm, with the resolution and offset already calculated in the values.

GoPxL Add-on Tool Manager

The GoPxL Add-on Tool Manager lets you quickly and easily add LMI-provided beta tools to GoPxL. You can also use the Manager add Python GDK tools to a firmware. After you add tools to a firmware file, upload the firmware to a compatible sensor; for more information, see *Software Upgrade* on page 400.



The tool manager is available in the 14632-x.x.x.x_SOFTWARE_GoPxL_Tools_AddOn_Beta.zip package, available on the LMI Product Downloads page (https://lmi3d.com/product-downloads/), under the *Beta Software Releases* section.

To get the package:

1. Go to https://lmi3d.com/product-downloads/ and log in to your account.
2. Select Gocator in the brand drop-down.
3. Choose the package that corresponds to the firmware you wish to run the tool manager on and download it.

> Remember that the tools available in the package are *beta* tools. LMI does not recommend using them in production settings.

The general workflow with the GoPxL Add-on Tool Manager is as follows:

1. Load a .dat GoPxL firmware file (or point the tool manager to the "GoPxL" folder in an unzipped Utilities package).
2. Load one or more beta tool packages (in .tar archives).
3. Enable the tools you want to add to the loaded firmware (in the checkbox column).
4. Run the tool manager on the firmware and tool packages.

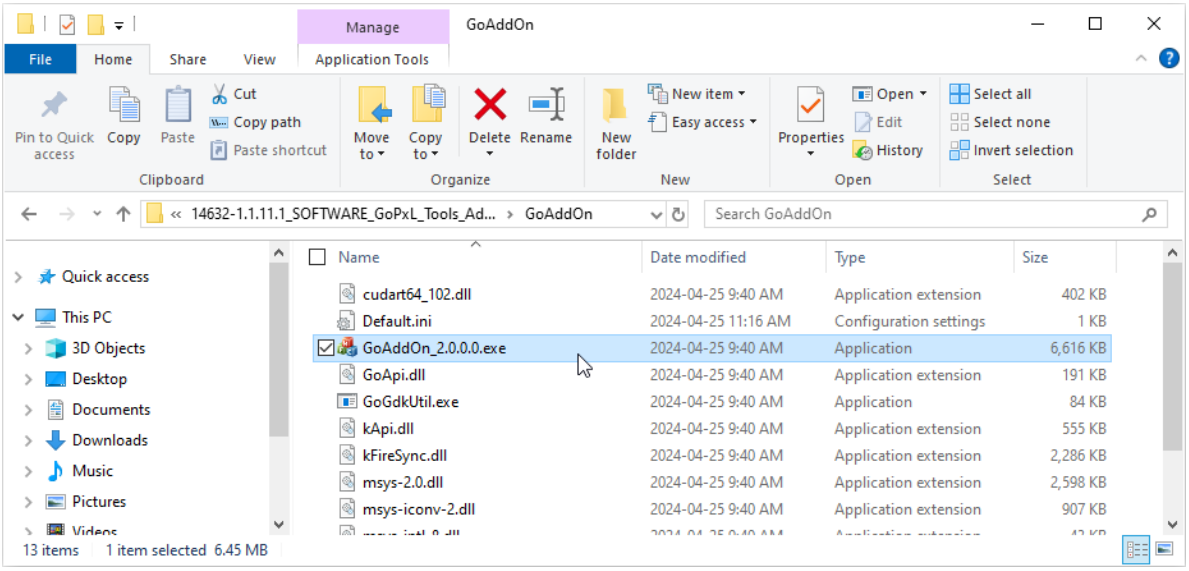
The result is a modified firmware file or GoPxL for PC that contains the tools you selected.

At any point before creating the new firmware with the tool manager, you can click **Reset** to remove the loaded firmware and tool packages, and start over.

Adding Beta Tools

To add beta tools to a firmware or to GoPxL for PC:

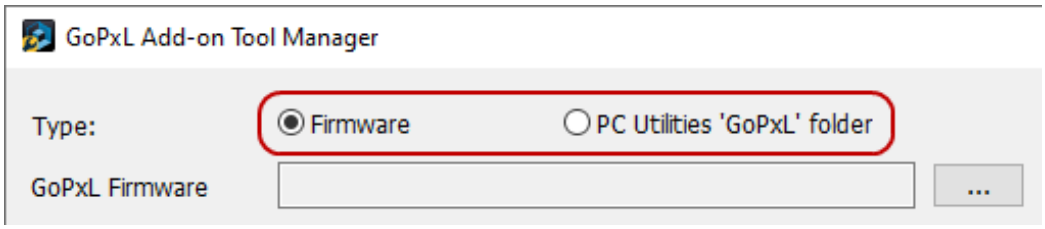
1. If you haven't already done so, download and unzip the package containing the GoPxL Add-on Tool Manager to a convenient location on your computer.
2. Launch the tool manager (GoAddOn_x.x.x.x.exe) from the \GoAddOn subfolder.



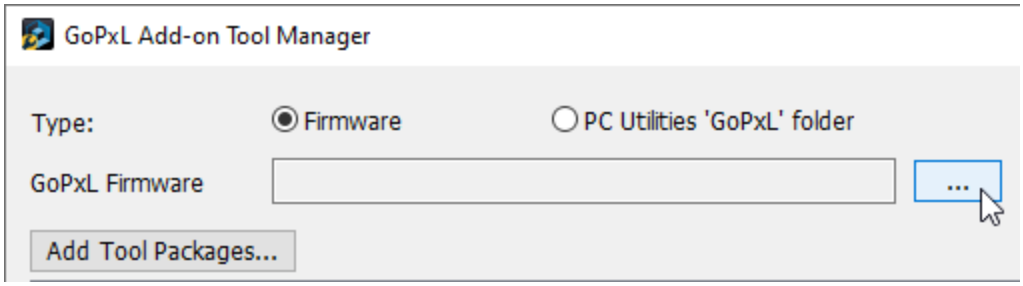
3. In the tool manager, set the type.

   To create a custom firmware you will upload to a sensor, set the type to **Firmware**.

   To create a custom Windows version of GoPxL, set the type to **Accelerator/Emu**.

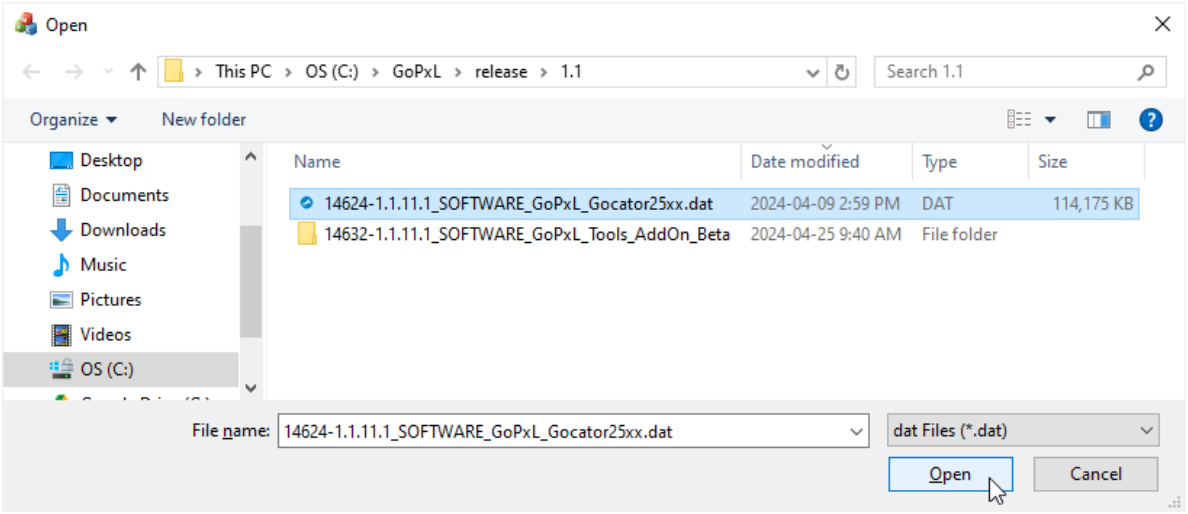


4. Below the type, click the ... button to choose the firmware or GoPxL folder.

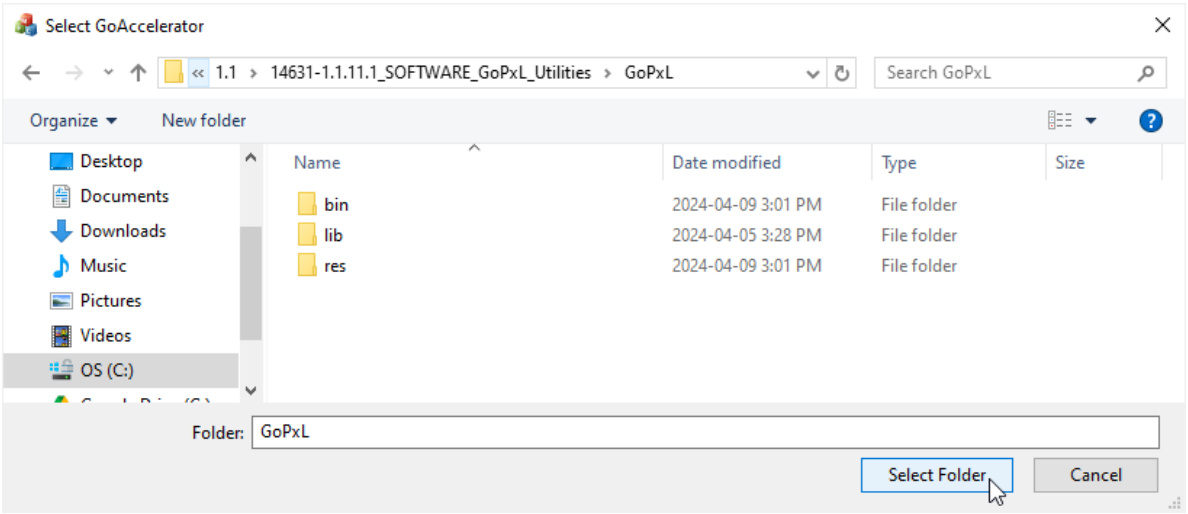


5. In the Open dialog that displays, navigate to the location of the .dat firmware file or the folder you want to use.

   Choosing a firmware:

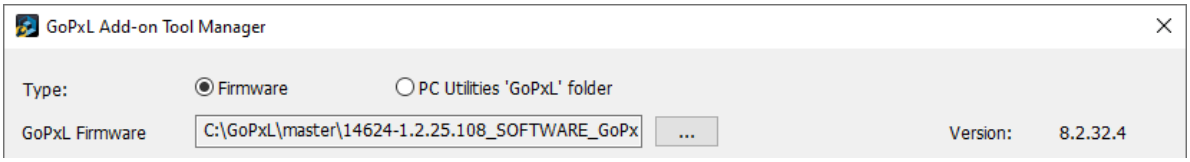


   Choosing a folder containing GoPxL for Windows:



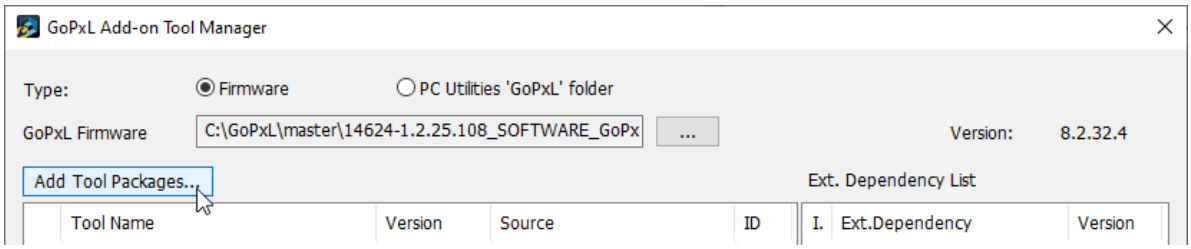
   Note that you choose the `GoPxL` folder.

   The tool manager loads GoPxL.

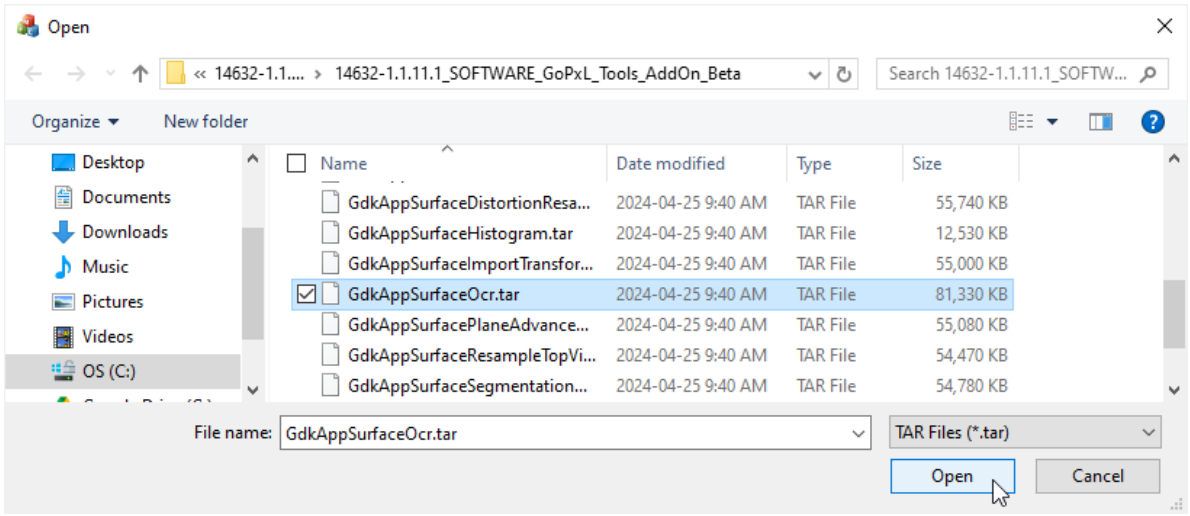


   If the version of GoPxL was previously modified using the tool manager, the tools will be listed in the tool manager. For information on removing tools, see *Removing Beta Tools* on page 487.

6. Click the **Add Tool Packages...** button above the tool list.



7. In the Open dialog, navigate to the folder containing the \GoAddOn folder, select a .tar tool package, and click **Open**.

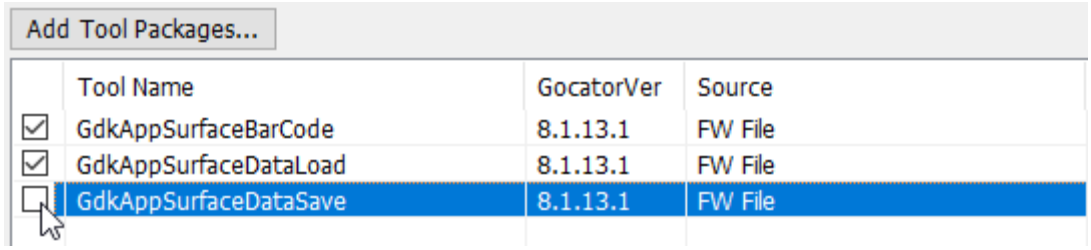


The tool manager adds the package to the tool list.

Note that you can add multiple packages at once from the Open dialog.

> The version of a tool package must match the version of the loaded firmware or folder. If a package's version does not match the firmware's, it is highlighted with grey in the tool list and can't be selected and included in the firmware. (The version of a tool package comes from the version of the GDK used to create it. It is not currently possible for users to create their own tool packages.).

8. When you have finished adding the packages, in the tool list, check the checkbox next the tools you want to add to the loaded firmware or folder.



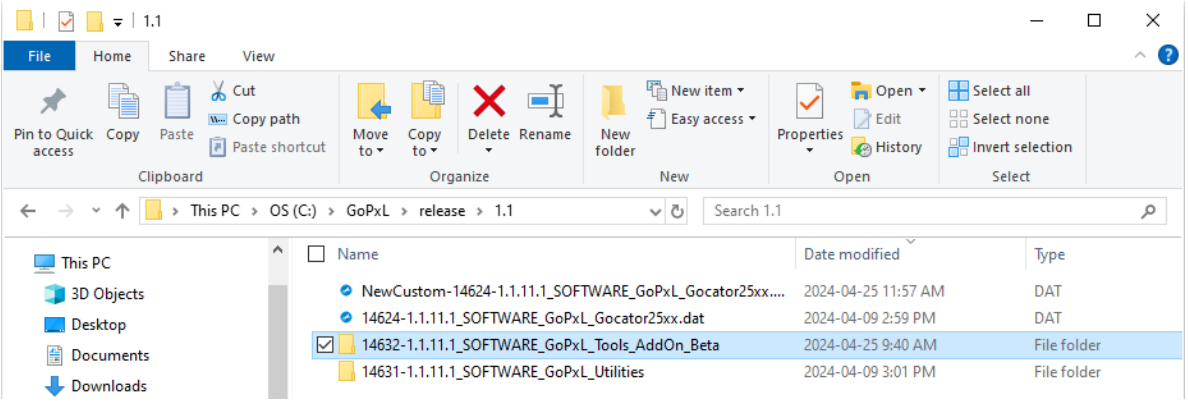
> Be sure to check at least one tool. If a tool is not checked, it will not be added to GoPxL.

9. (Optional) If you are adding tools to a firmware, in the **Output File Prefix** field, change the default "NewCustom" to something that will help you remember what the new firmware is for.

This field is hidden when you are adding tools to GoPxL for Windows folder.

10. Click **Run**.

    If you are adding tools to a firmware, the tool manager creates a new firmware file that contains the beta tools you selected, using the prefix you provided for the filename. The new firmware is created in the same location as the original firmware you loaded.



> The tool manager will overwrite an existing firmware without warning.

If you are adding tools to a GoPxL for PC folder, the tool manager updates it with the added tools.

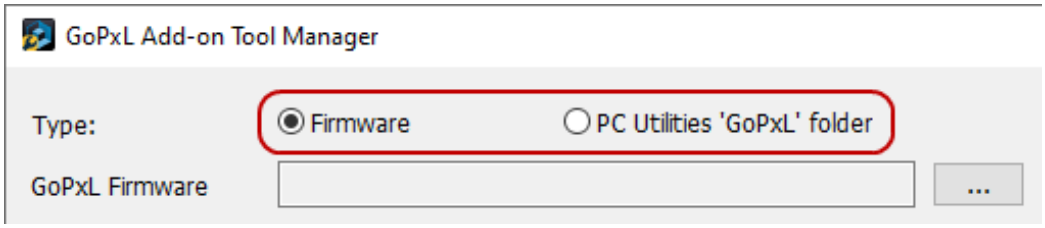
After you have successfully created the new firmware or PC version of GoPxL, you can upload it to any compatible sensor (for more information, see *Software Upgrade* on page 400) or run it (for more information, see *Running GoPxL on a Windows PC* on page 403).

Removing Beta Tools

To remove a tool you previously added to a firmware or from GoPxL for PC:

1. Follow steps 1 to 5 in *To add beta tools to a firmware or to GoPxL for PC:* on page 484

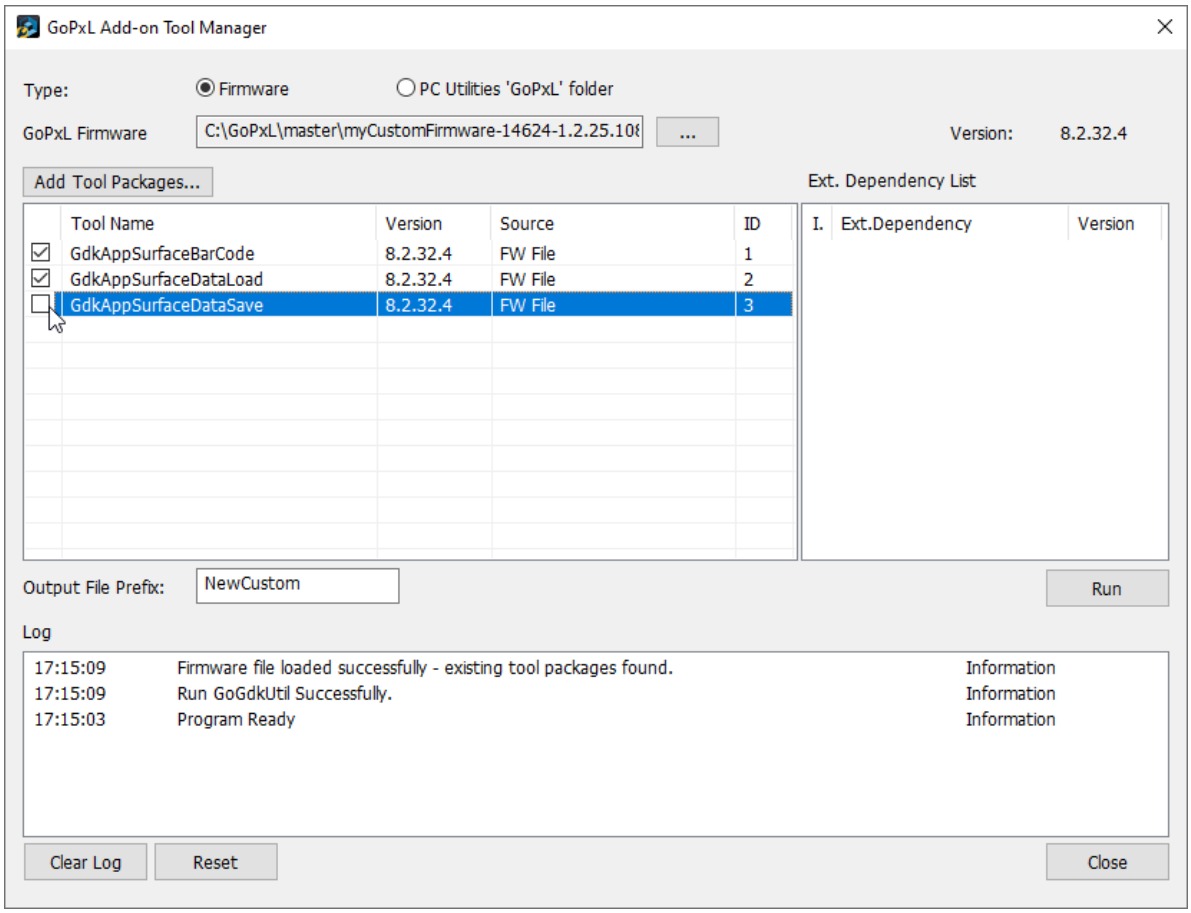
   The type should match the type used when the tool was added.



   You must navigate to the location of the previously modified GoPxL (firmware or PC version).

   After loading GoPxL, the tool list is automatically populated.

2. In the tool list, deselect the tools you want to remove.



3. Click **Run**.

   The GoPxL Add-on Tool Manager saves a new version of GoPxL with the deselected tools removed.

   The tool manager overwrites existing firmware files with no warning.

Pattern Editor

The pattern editor lets you modify patterns created in the Surface Pattern Matching tool . Although the patterns created in the Surface Pattern Matching tool will often result in good matches with your targets, you can use the pattern editor to improve the models, specifically by doing the following:

- Remove unwanted contours the Surface Pattern Matching tool has detected on edges in the scan data.
- Re-detect contours from the scan data using higher or lower levels of input image resolution (taken from the scan data) or contrast levels, compared to what the Surface Pattern Matching tool does internally.
- Identify certain contours as being required for a match to occur.
- Identify certain contours as being used to determine the position of a matched instance.

> The Pattern Editor contains a **Pattern** setting (Surface or Image). Leave this set to Surface. The Image setting is reserved for future use.

The editor uses two folders in `C:\GoTools\`, on the PC it is running on.

`GoPxLSurfacePatternMatchingSensor`

When you choose a sensor or PC instance of GoPxL in the **Source** drop-down and click **Connect**, the editor empties this folder and copies any model files on the source into the folder. For this reason, if you have previously worked on model from this folder, make sure you have uploaded any changed models to the sensor or PC instance. The models in this folder are listed in the Model Files until you switch **Source** to PC Work Directory, or empty it by clicking **Clear Sensor Work Directory**.

The folder is referred to as the "Sensor Work Directory" in the editor.

`GoPxLSurfacePatternMatching`

This is called the "PC Work Directory" in the editor. You typically use this folder to transfer model files between sensors or PC instances of GoPxL. Also, because it is only emptied if you manually do it by clicking **Clear PC Work Directory**, it may be better to work from this folder. To get models in this folder, you first connect to a source and then click **Copy all files to the PC work directory**.

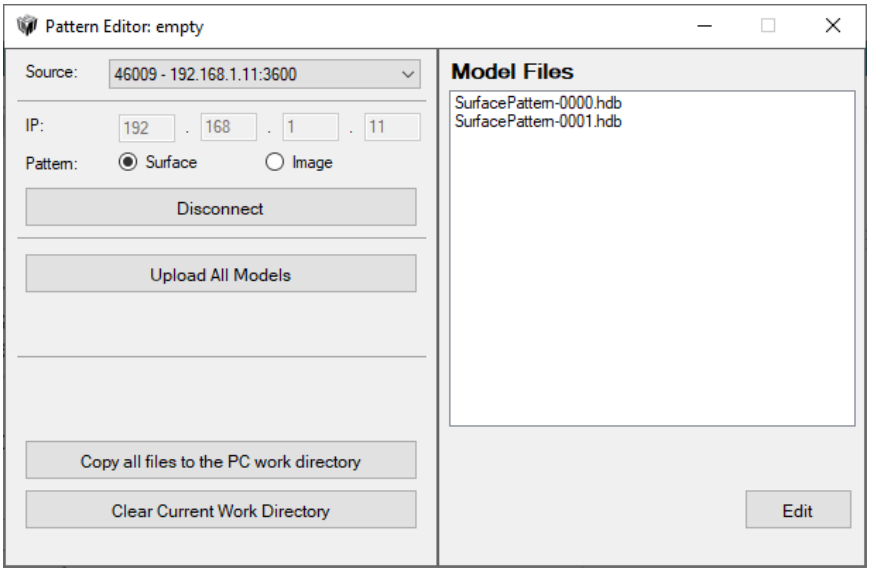
To see the files in this folder, switch **Source** to "PC Work Directory".

The pattern editor is available in the GoPxL Utilities package (`14631-x.x.x.x_SOFTWARE_GoPxL_Utilities.zip`, in the `Tools\Pattern Editor` folder. You can find the package on LMI's Product Downloads page (https://lmi3d.com/product-downloads/).

Launching the Pattern Editor

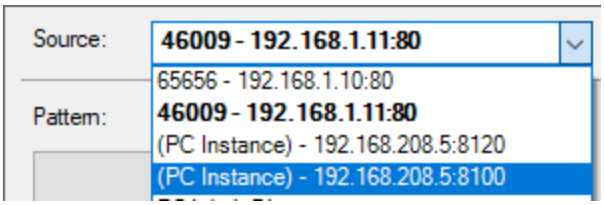
The first time you run the pattern editor, it may take longer to launch, as it will register certain DLLs required by the application.

The pattern editor can work with model files that come from an unaccelerated sensor or from a PC instance of GoPxL (which may or may not be accelerating a sensor). In all cases, the editor works with copies of the files in working folders in the local PC file system.



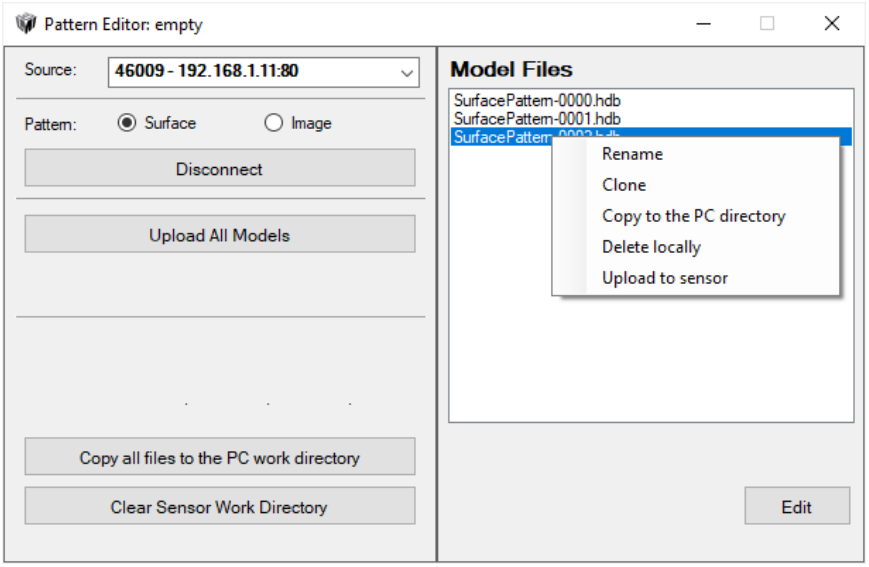
The Pattern Editor application

Parameters

Name	Description
Source	The source from which the editor fetches model files.  The currently connected sensor is displayed in bold in the **Source** list. The list contains the following types of entries: **Sensor** Sensors are listed with their serial number followed by their IP address. **PC Instances**

Name	Description
	Instances of GoPxL running on the PC are identified by "(PC Instance)", followed by their IP address. **PC Work Directory** This is a local directory (`C:\GoTools\GoPxLSurfacePatternMatching`) that you typically use to transfer model files between sensors and PC instances of GoPxL.
Pattern	Leave this set to Surface.
Connect / Disconnect	Connects to the source in **Source**. Note that connecting to a sensor deletes all models in the sensor working folder (`C:\GoTools\SurfacePatternMatchingSensor`) and then downloads the models currently on the sensor to that folder. For this reason, if you have modified models with the Pattern Editor, make sure to upload them before disconnecting and reconnecting. To fetch any models you have created on-sensor after connecting, disconnect using the **Disconnect** button, and then reconnect.
Upload All Models	Uploads the models from the `GoPxLSurfacePatternMatchingSensor` local folder to the connected source (to a sensor or a PC instance of GoPxL).
Edit	Opens the selected model in the model editor; you can also double-click a model in the list to edit it. For more information, see *Overview of the Editor* on the next page.
Copy all files to the PC work directory Copy all files to the sensor work directory	Copies files between the PC Work Directory and the Sensor Work Directory. The name of this button depends on whether **Source** is set to **PC Work Directory**, or to a sensor or PC instance of GoPxL.
Clear Sensor Work Directory Clear PC Work Directory	Removes all models from the local working folder. The name of this button depends on whether **Source** is set to **PC Work Directory**, or to a sensor or PC instance of GoPxL. When you clear the sensor work directory, you are only deleting the files locally, not on the sensor or PC instance.

By right-clicking on a model in the list of models, you can perform the operations listed below.



Rename: Renames the model.

Clone: Makes a copy of the selected model using the name you provide.

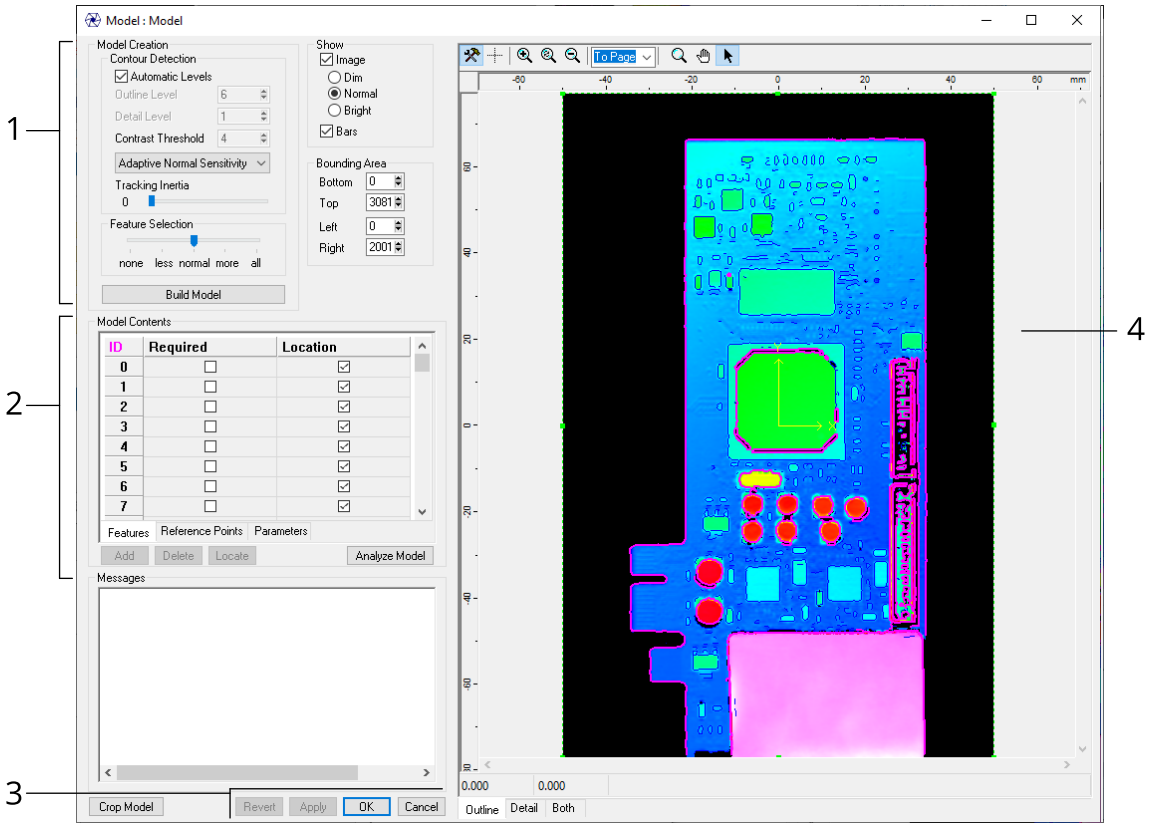
Copy to the PC directory / **Copy to the sensor directory**: Copies the model to a local directory.

Delete locally: Deletes the model from the `GoPxLSurfacePatternMatchingSensor` folder. Not displayed when **Source** is set to "PC Work Directory".

Upload to sensor: Uploads the model to the connected sensor.

Overview of the Editor

After clicking Edit in the pattern editor helper application, the selected model opens in the editor window.



	Element	Description
1	Model Creation pane	Settings related to contour detection and feature selection. After configuring these settings, or resizing the model's bounding box (green dotted line), you must rebuild the model using the **Build Model** button.
2	Model Contents	The list of the features in a model (contours used in recognition and location of an instance). Note that some model contents (reference points and some settings in the **Parameters** tab) are not currently supported by the Surface Pattern Matching tool.
3	Save and discard buttons	Used to apply changes to a model, revert to the model's original state when it was loaded, and so on.
4	Outline, Detail, and Both tabs	The editor tabs that show the Outline and Detail levels of the model. The **Both** tab shows both levels together, but you can't edit models on this tab.

Models

Models are made up of features selected from the source contours detected either by the Surface Pattern Matching tool or by the model editor itself (if you rebuild the model using the **Build Model** button). The features are used to identify and locate instances in the scan data.

Contours, and the features selected from the contours for use in recognizing and locating an instance, work on two “coarseness” levels: the Outline level and the Detail level.

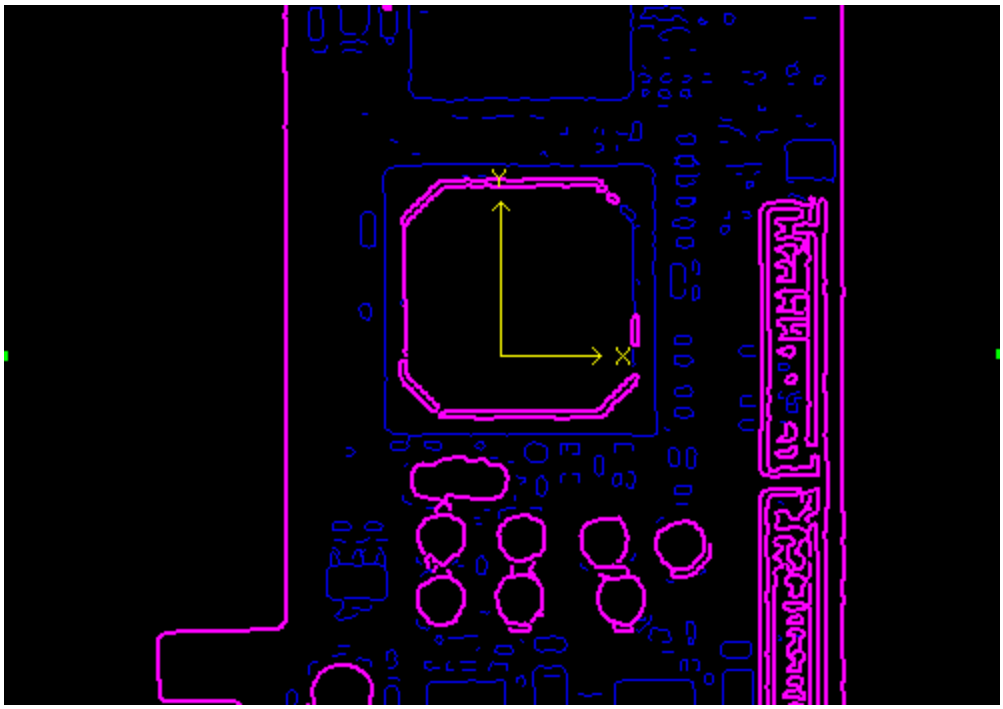
Outline: Used to quickly identify potential instances of a pattern in scan data. The Outline level is the "coarse" level of contours / features. The features at this level can be less stable, as they are not used to calculate the location of the instance. For example, a label whose position might change from frame to frame or a hole whose size might change from frame to frame could be kept at the Outline level.

Detail: Used to confirm whether an instance is in fact valid and to refine its location. The Detail level is the "fine" level of contours / features. The features at this level must be more stable and rigid with respect to one another. For this reason, given Surface scan data, include features that are all on the same plane to ensure that their positions will not be unstable due to parallax or other scanning issues. Furthermore, features on a part that might vary in size from frame to frame, or change position (such as a label), should be excluded.

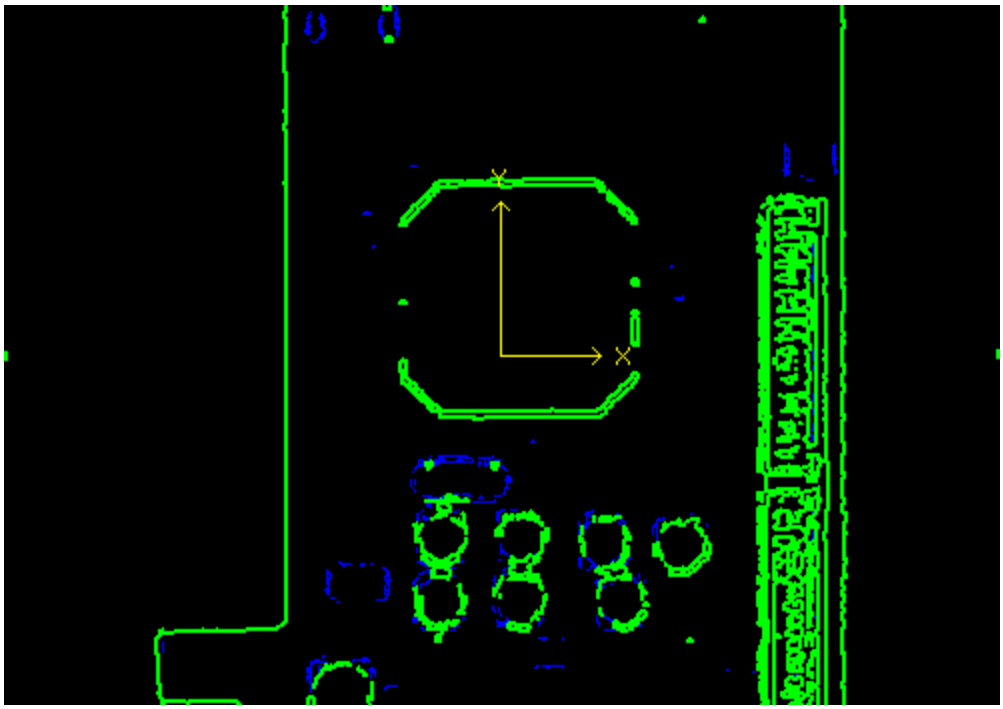
That said, the Outline and Detail levels will often be similar in terms of which features are included.

You can edit (add and remove) features at these levels separately, in the Outline and Detail panels in the main editor window. For more information on adding and removing features, see *Adding and Removing Features Manually* on the next page.

In the editor, “unused” contours (those not selected to take part in instance recognition or locating) are indicated with dark blue paths. Features (contours selected to take part in instance recognition or locating) are indicated with either magenta paths (at the Outline level) or with green paths (at the Detail level); the features in a model are listed in the **Model Contents** pane.



Dark blue unused contours and magenta features at the Outline level.



Dark blue unused contours and green features at the Detail level.

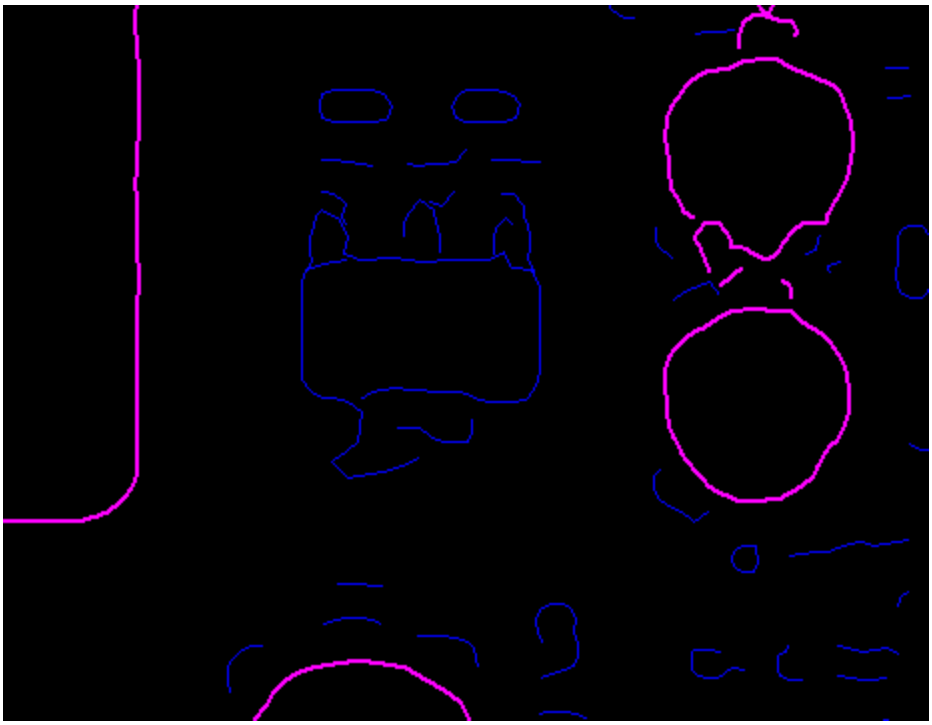
In the Surface Pattern Matching tool, only the Detail level of features is displayed.

Adding and Removing Features Manually

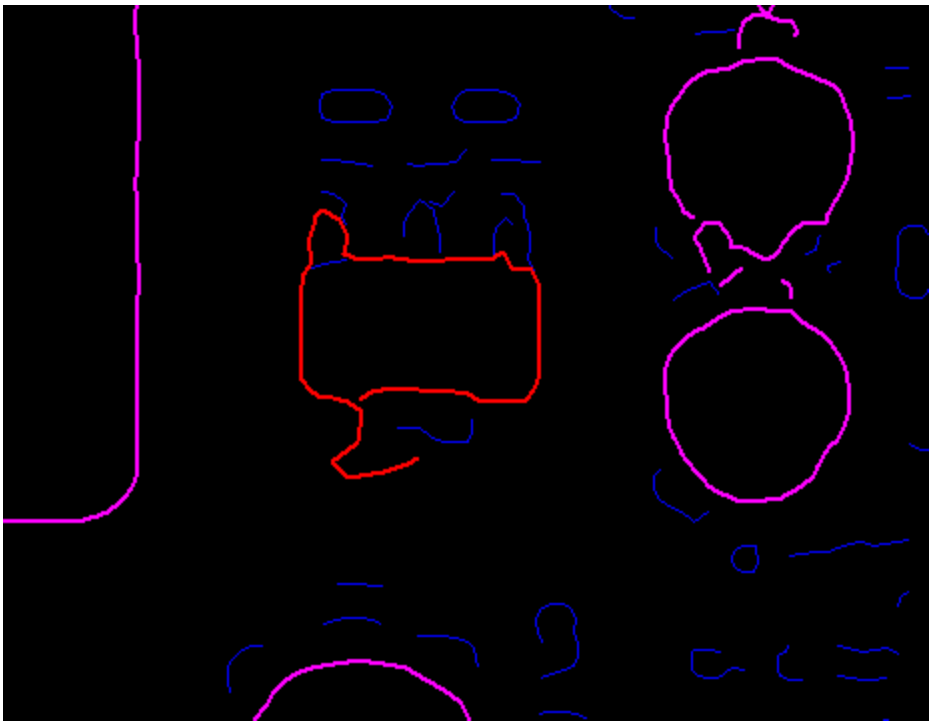
You can manually add features to a model from the source contours, or remove features currently in a model, at both the Outline and Detail levels. This can be useful if the model produced by the Surface Pattern Matching tool includes features related to parts of targets that could change or be present/absent from frame to frame. You should only include features that are constant from frame to frame.

Adding and removing features works in the same way in the Outline and Detail tabs.

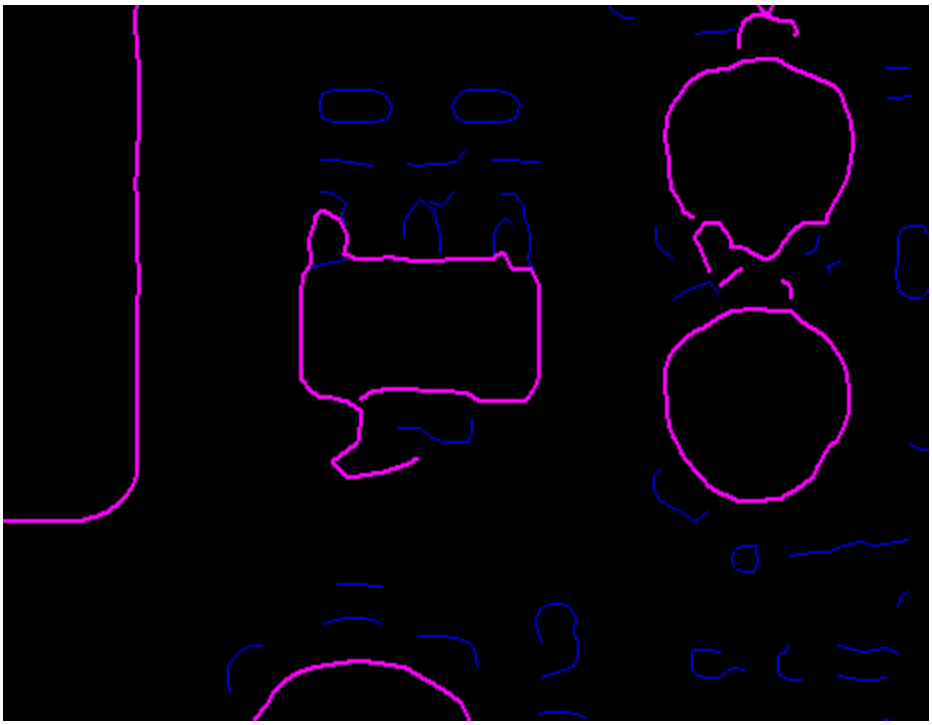
To add a feature from a source contour, double-click a dark blue contour in either the Outline or Detail tab and click Add or press the Insert key on your keyboard.



Dark blue unused contour (contours already added as features in the model are magenta).

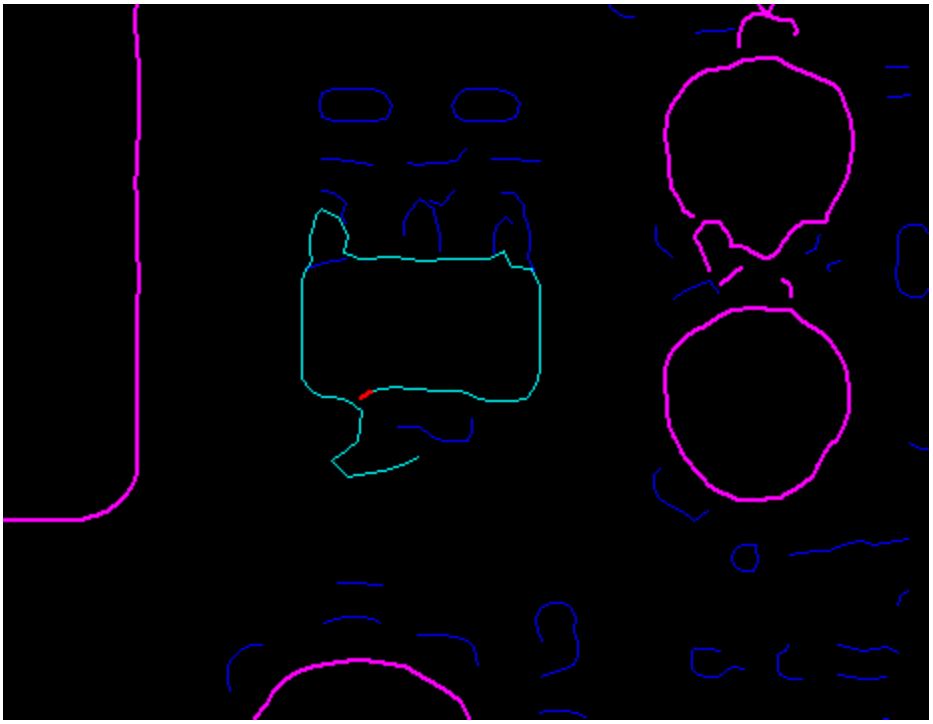


Contour selected by double-clicking it.



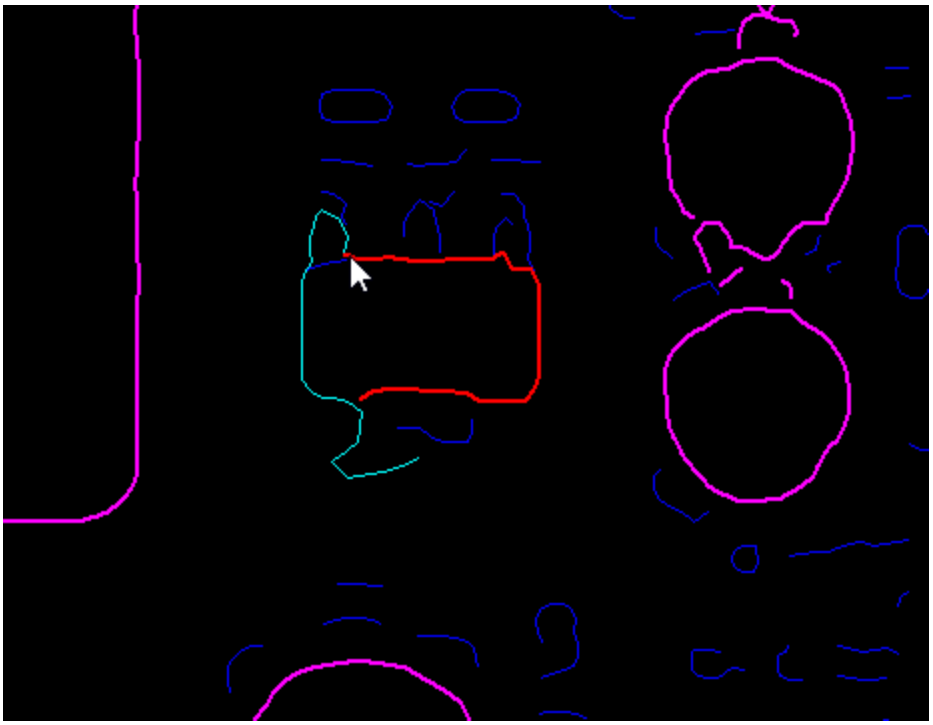
Contour added as a feature in the model (magenta).

If you single-click a dark blue unused contour, it turns cyan and lets you select segments of the contour.

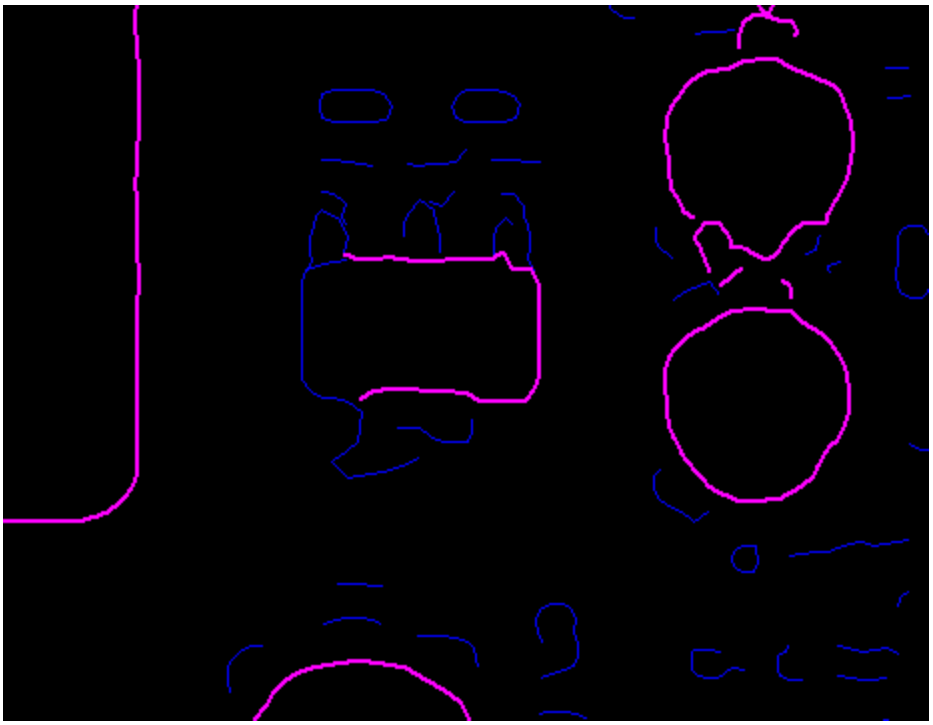


Contour highlighted in cyan with a selected segment (red).

Pressing the Ctrl key on your keyboard and clicking another segment selects a portion of the cyan path.



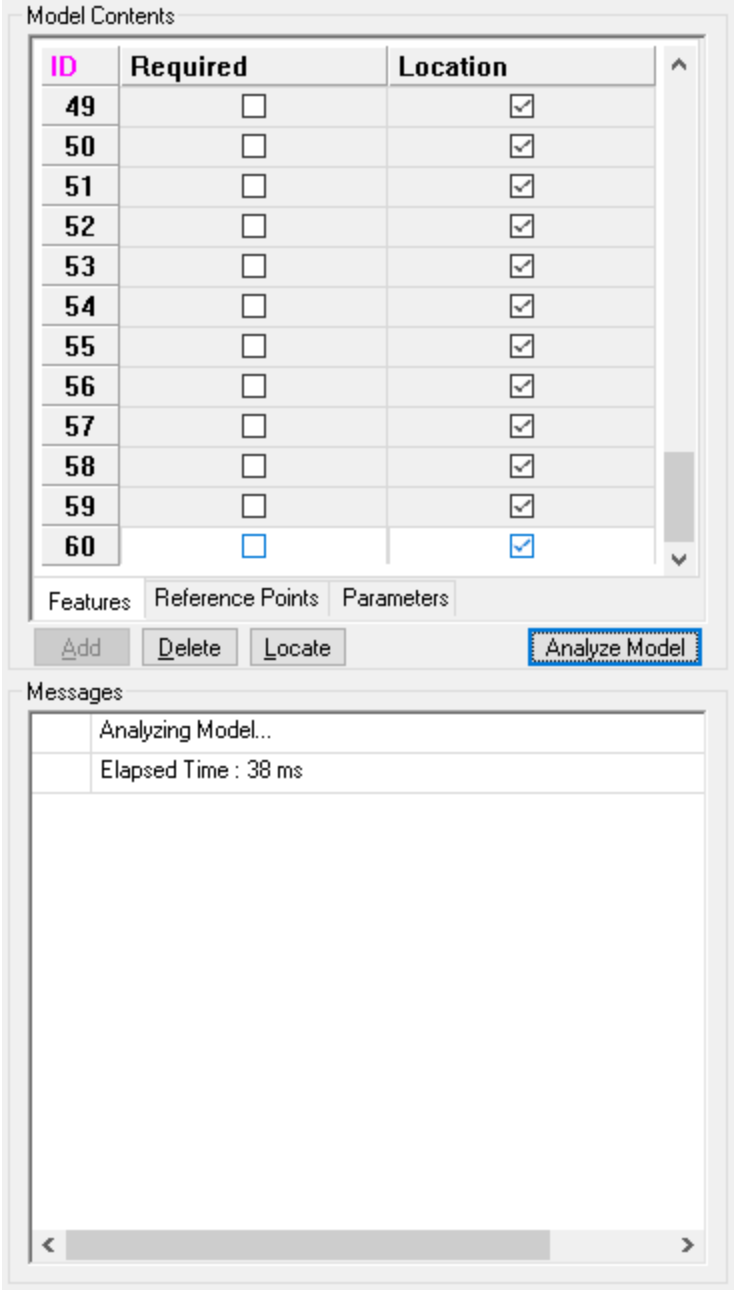
Clicking **Add** or pressing the Insert key on your keyboard adds the segment of the contour as a feature to the model.



After adding a feature, it is added to the list of features on the **Feature** tab in the **Model Contents** panel. Do not press the build button after adding or removing features, or you will lose the modifications that were just made. You must however save the changes; for more information, see *Saving and Discarding Changes* on page 505.

To remove a feature, click a magenta or green path in the editor to select it and click **Delete** or press the Delete key on your keyboard. After removing a feature, it is removed from the list of features on the Feature tab in the Model Contents panel. You do not need to build the model, but must save the changes; for more information, see *Saving and Discarding Changes* on page 505.

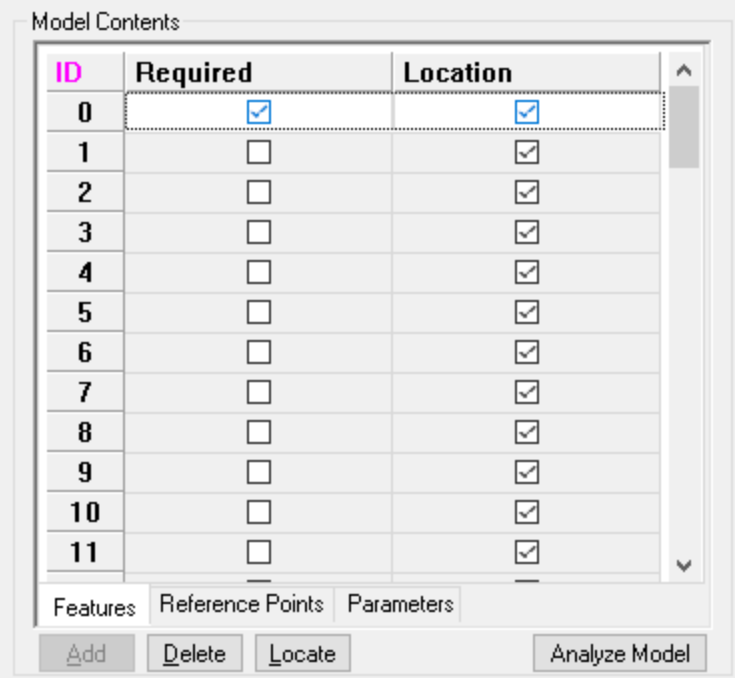
After adding a feature to a model or removing a feature from a model, you should analyze the model by clicking **Analyze Model**. Make note of any errors in the **Messages** panel.



No error messages after clicking Analyze Model.

Setting Required and Locating Features

In the list of features in the **Model Contents** pane, you can indicate that a feature is “required” or that it is used to calculate the location of an instance by checking the appropriate checkbox next to the feature.



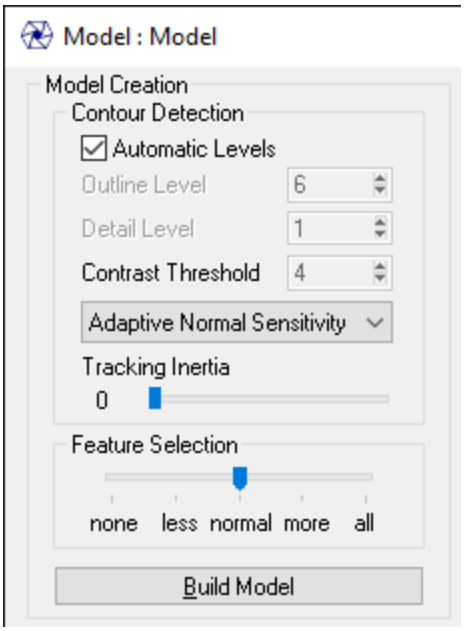
When Required is checked for a feature, it *must* be found by the Surface Pattern Matching tool in order for an instance to be identified.

When Location is checked for a feature, the Surface Pattern Matching tool uses the feature to calculate the location of instances. If a feature's location is not checked, it is only used for instance recognition. An example of the latter is a tag or label glued to an object. Although the label's contours (it's shape or what is written on it) might be unique enough to help recognize an instance, it's position *on* the object (that is, relative to the other features) might vary in its position from frame to frame. For this reason, it might be useful for instance recognition, but not for determining the location of the object.

Model Creation Settings and Rebuilding

The Surface Pattern Matching tool uses internally fixed settings to detect contours in the scan data and then select features from those source contours. In the pattern editor, you can increase or decrease the contour detection levels, change the contrast threshold, and so on, and then rebuild the model. This can be useful if the Surface Pattern Matching tool's internal settings do no produce the right amount of source contours and subsequently features.

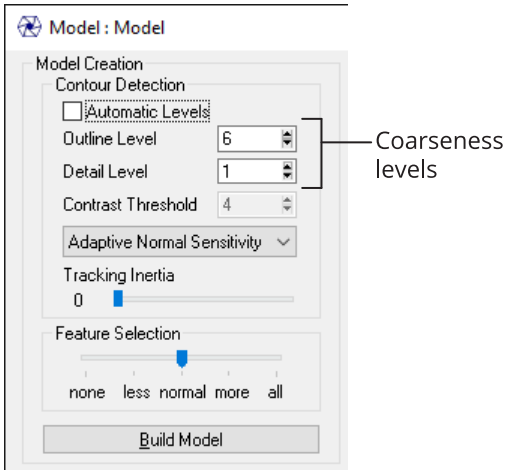
The settings described here are found in the **Model Creation** section of the model editor.



After making changes to any of these settings, you must rebuild the model by clicking **Build Model**, and then save the changes. You should also click **Analyze Model** after rebuilding a model. Pay special attention to messages in the **Messages** pane at the bottom of the editor to make sure there are no errors. For more information on saving changes, see *Saving and Discarding Changes* on page 505.

Coarseness Levels

By default, the pattern editor uses automatically determined contour coarseness values (at both the Outline and Detail levels) to detect contours in the scan data image. If you uncheck **Automatic Levels**, you can change the **Outline Level** and **Detail Level** values to generate more or fewer contours, from which you can then select features that more reliably represent your target.

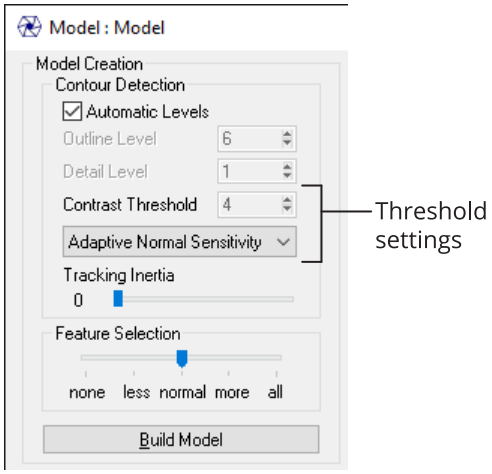


The **Outline Level** and **Detail Level** values range from 1 to 16. At the lowest value, contours are detected in a full-resolution version of the image based on the scan data, which results in more contours from which to choose features. At higher values, contours are detected in a reduced-

resolution version of the image based on the scan data: the resolution is reduced by the setting's value, which results in fewer contours being detected. Note that **Detail Level** must be less than or equal to **Outline Level**.

Thresholds

You can adjust the level of sensitivity the pattern editor uses to detect contours in the scan data image.



By default, the sensitivity is set to Adaptive Normal Sensitivity, but you can set it to one of the following adaptive sensitivity levels, or to a fixed threshold value (see below).

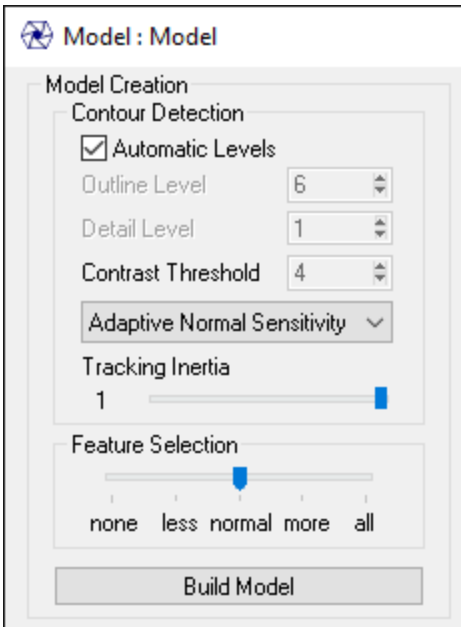
Adaptive High Sensitivity: Results in more low-contrast contours, but also noise.

Adaptive Low Sensitivity: Results in strongly defined contours and eliminates noise, but may miss important contour segments.

If you set the dropdown to Fixed Value, you can then set a fixed threshold in **Contrast Threshold**. The **Contrast Threshold** value corresponds to the minimum step required to detect corners. A lower value generates more contours when you rebuild the model, but may also result in noise.

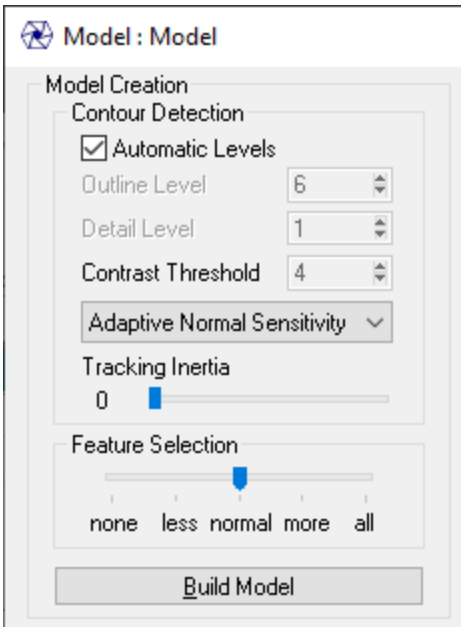
Tracking Inertia

Setting the Tracking Inertia slider to 1 closes small gaps in the source contours, connecting contours that might otherwise be broken into smaller sections.



Feature Selection

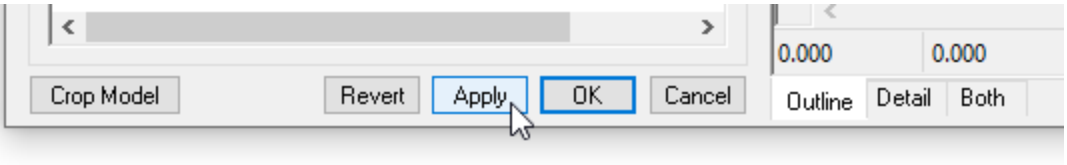
This setting ranges from **none** to **all**, which determines which features the pattern editor selects from the detected contours and adds to the model when you rebuild it. You should use **none** (which adds no features to the model) if you want to manually add features to the model from the detected contours. The **normal** setting tries to add the most appropriate features to the model; use this setting with simple to moderately complex parts. The **all** setting adds all detected contours as features to the model; only use this with very complex parts, such as electronic parts.



Saving and Discarding Changes

After making changes to a model (either adding or removing features, or re-detecting contours by clicking **Build Model**), you *must* do the following:

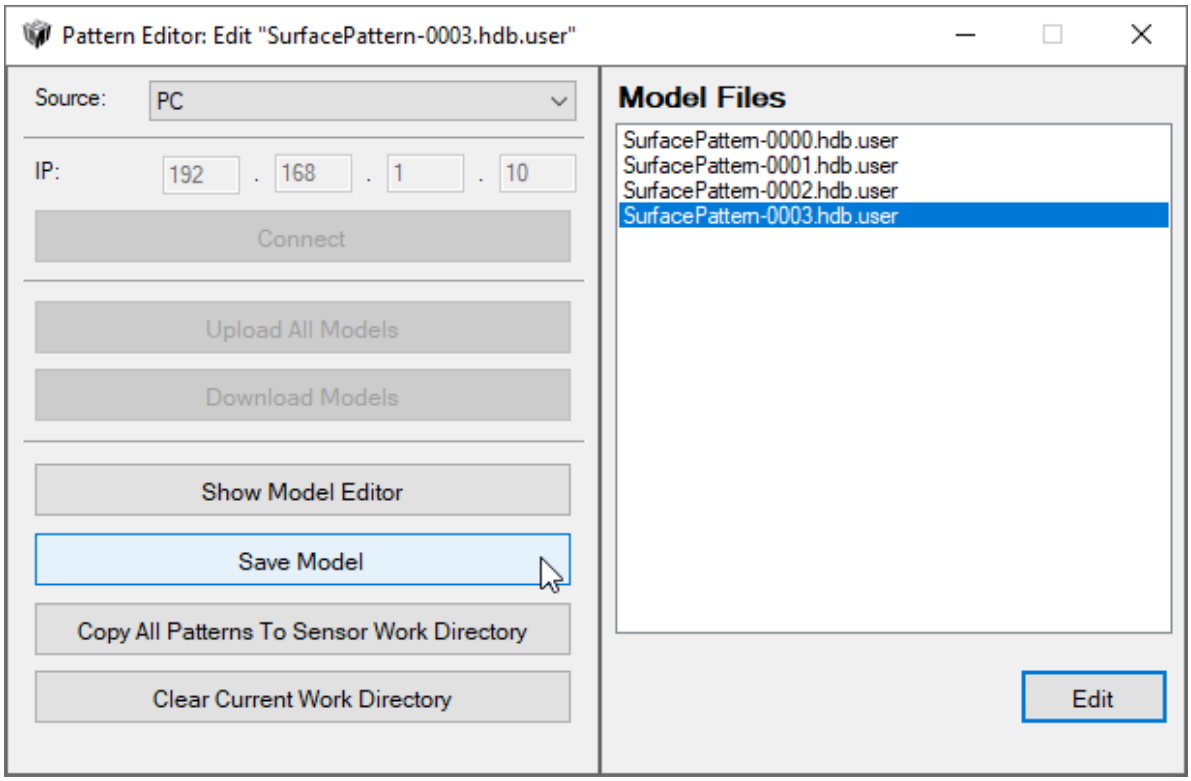
1. In the model editor, at the bottom of the window, click **Apply** or **OK**.



   Clicking **Apply** leaves the model editor open. Do this if you want to continue working on a model (for example, if you want to test the model in Gocator before closing the model).

   Clicking **OK** closes the model editor.

2. In the pattern editor helper application, click the model you were working on, and click **Save Model**.



   The model is saved to the working folder.

> After making changes to a model, you must re-load the pattern in the instance of Surface Pattern Matching to see the changes.

Before saving changes (either by clicking **Apply** or **OK**), you can revert the model to its initial state by clicking **Revert**.

Miscellaneous

Reference points, which you can create in the pattern editor on the **Reference Points** tab in the **Model Contents** panel, are not currently supported by the Surface Pattern Matching tool.

The Surface Pattern Matching tool does not currently support the custom shading area (on the **Parameters** tab in the **Model Contents** panel).

You can change the region of the model by resizing the green bounding box (or setting its dimensions in the **Bounding Area** section in the model editor) and rebuilding the model. After resizing and rebuilding the model to the desired region, you can use the **Crop Model** button to reduce the size of the final model surface, and reduce the pattern file size.

In the Show section in the model editor, you can hide the scan data to see only the contours and features by unchecking **Image**. Note that the Dim / Normal / Bright options below **Image** only apply to intensity data.

Accessories

2D Camera Accessories

Description	Part Number
Gocator 2D 1120-M, 12 MP mono, 1/1.8" sensor format	601050A-M-01
Mounting Plate, Gocator 2D base	301389
Mounting plate for RMX140 to Gocator 2D	301355
L bracket for RMX140	TBD
RMX140 ring light, 50 degree lens, white	301373
Diffuser for RMX140	301381
Polarizer for RMX140	301380
Lens, C-mount, 8.5 mm, up to 2/3" sensor format	301358
Lens, C-mount, 12 mm, up to 2/3" sensor format	301359
Lens, C-mount, 16 mm, up to 2/3" sensor format	301357
Ethernet, High flex, M12 to RJ45, 5 Meter	301351
Power/IO, High flex, M12 to open, 5 meter	301366
Power Supply, AC/DC DIN RAIL, 24V, 10A, 240W	301363
Power cord, Plug Type B (NA) and open wire end, 1.9 meter	301377

Troubleshooting

Review the guidance in this chapter if you are experiencing difficulty with a sensor system.

Mechanical/Environmental
The sensor is warm.

- It is normal for a sensor to be warm when powered on. A sensor is typically 15° C warmer than the ambient temperature.

Connection
When attempting to connect to the sensor with a web browser, the sensor is not found (page does not load).

- Verify that the sensor is powered on and connected to the client computer network. The Power Indicator LED should illuminate when the sensor is powered.
- Check that the client computer's network settings are properly configured.

When attempting to log in, the password is not accepted.

- Use the Sensor Recovery tool. See *GoPxL Discovery Tool* on page 473 for steps to reset the password.

Performance
The sensor CPU level is near 100%.

- Consider reducing the speed. If you are using a time or encoder trigger source, see *Triggers* on page 110 for information on reducing the speed. If you are using an external input or software trigger, consider reducing the rate at which you apply triggers.
- Review the measurements that you have programmed and eliminate any unnecessary measurements.

Return Policy

Return Policy
Before returning the product for repair (warranty or non-warranty) a Return Material Authorization (RMA) number must be obtained from LMI. Please call LMI to obtain this RMA number.

Carefully package the sensor in its original shipping materials (or equivalent) and ship the sensor prepaid to your designated LMI location. Please ensure that the RMA number is clearly written on the outside of the package. Inside the return shipment, include the address you wish the shipment returned to, the name, email and telephone number of a technical contact (should we need to discuss this repair), and details of the nature of the malfunction. For non-warranty repairs, a purchase order for the repair charges must accompany the returning sensor.

LMI Technologies Inc. is not responsible for damages to a sensor that are the result of improper packaging or damage during transit by the courier.

Software Licenses

YOLOv9

Website:

https://github.com/WongKinYiu/yolov9

License:

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Copyright (C) 2024 Chien-Yao Wang, I-Hau Yeh, Hong-Yuan Mark Liao

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see https://www.gnu.org/licenses/.

CLI11

Website:

https://github.com/CLIUtils/CLI11

License:

CLI11 1.8 Copyright (c) 2017-2019 University of Cincinnati, developed by Henry Schreiner under NSF AWARD 1414736.

All rights reserved. Redistribution and use in source and binary forms of CLI11, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

xxhash

Website:

https://github.com/Cyan4973/xxHash

License:

xxHash Library

Copyright (c) 2012-present, Yann Collet

All rights reserved.

BSD 2-Clause License (http://www.opensource.org/licenses/bsd-license.php)

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN

CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

---

xxhsum command line interface

Copyright (c) 2013-present, Yann Collet

All rights reserved.

GPL v2 License

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA.

JSON for C++

Website:

https://github.com/nlohmann/json

License:

MIT License

Copyright (c) 2013-2019 Niels Lohmann

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF

CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

OpENer

Website:

https://github.com/EIPStackGroup/OpENer

License:

SOFTWARE DISTRIBUTION LICENSE FOR THE

ETHERNET/IP(TM) COMMUNICATION STACK

(ADAPTED BSD STYLE LICENSE)

Copyright (c) 2009, Rockwell Automation, Inc. ALL RIGHTS RESERVED. EtherNet/IP is a trademark of ODVA, Inc.

Redistribution of the Communications Stack Software for EtherNet/IP and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright and trademark notices, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of Rockwell Automation, ODVA, nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission from the respective owners.

The Communications Stack Software for EtherNet/IP, or any portion thereof, with or without modifications, may be incorporated into products for sale. However, the software does not, by itself, convey any right to make, have made, use, import, offer to sell, sell, lease, market, or otherwise distribute or dispose of any products that implement this software, which products might be covered by valid patents or copyrights of ODVA, Inc., its members or other licensors nor does this software result in any license to use the EtherNet/IP mark owned by ODVA. To make, have made, use, import, offer to sell, sell, lease, market, or otherwise distribute or dispose of any products that implement this software, and to use the EtherNet/IP mark, one must obtain the necessary license from ODVA through its Terms of Usage Agreement for the EtherNet/IP technology, available through the ODVA web site at www.odva.org. This license requirement applies equally (a) to devices that completely implement ODVA's Final Specification for EtherNet/IP (“Network Devices”), (b) to components of such Network Devices to the extent they implement portions of the Final Specification for EtherNet/IP, and (c) to enabling technology products, such as any other EtherNet/IP or other network protocol stack designed for use in Network Devices to the extent they implement portions of the Final Specification for EtherNet/IP. Persons or entities who are not already licensed for the EtherNet/IP technology must contact ODVA for a Terms of Usage Agreement.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

picoc

Website:

https://github.com/jpoirier/picoc

License:

Copyright (c) 2009-2011, Zik Saleeba

Copyright (c) 2015, Joseph Poirier

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

* Neither the name of the Zik Saleeba nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

tar (binary only)

License:

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord

with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modifiedit, and giving a relevant date.

- b) The work must carry prominent notices stating that it isreleased under this License and any conditions added under section7. This requirement modifies the requirement in section 4 to “keep intact all notices”.

- c) You must license the entire work, as a whole, under thisLicense to anyone who comes into possession of a copy. ThisLicense will therefore apply, along with any applicable section 7additional terms, to the whole of the work, and all its parts,regardless of how they are packaged. This License gives nopermission to license the work in any other way, but it does notinvalidate such permission if you have separately received it.

- d) If the work has interactive user interfaces, each must displayAppropriate Legal Notices; however, if the Program has interactiveinterfaces that do not display Appropriate Legal Notices, yourwork need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product(including a physical distribution medium), accompanied by theCorresponding Source fixed on a durable physical mediumcustomarily used for software interchange.

- b) Convey the object code in, or embodied in, a physical product(including a physical distribution medium), accompanied by awritten offer, valid for at least three years and valid for aslong as you offer spare parts or customer support for that productmodel, to give anyone who possesses the object code either (1) acopy of the Corresponding Source for all the software in theproduct that is covered by this License, on a durable physicalmedium customarily used for software interchange, for

a price nomore than your reasonable cost of physically performing thisconveying of source, or (2) access to copy theCorresponding Source from a network server at no charge.

- c) Convey individual copies of the object code with a copy of thewritten offer to provide the Corresponding Source. Thisalternative is allowed only occasionally and noncommercially, andonly if you received the object code with such an offer, in accordwith subsection 6b.

- d) Convey the object code by offering access from a designatedplace (gratis or for a charge), and offer equivalent access to theCorresponding Source in the same way through the same place at nofurther charge. You need not require recipients to copy theCorresponding Source along with the object code. If the place tocopy the object code is a network server, the Corresponding Sourcemay be on a different server (operated by you or a third party)that supports equivalent copying facilities, provided you maintainclear directions next to the object code saying where to find theCorresponding Source. Regardless of what server hosts theCorresponding Source, you remain obligated to ensure that it isavailable for as long as needed to satisfy these requirements.

- e) Convey the object code using peer-to-peer transmission, providedyou inform other peers where the object code and CorrespondingSource of the work are being offered to the general public at nocharge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may

be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from theterms of sections 15 and 16 of this License; or

- b) Requiring preservation of specified reasonable legal notices orauthor attributions in that material or in the Appropriate LegalNotices displayed by works containing it; or

- c) Prohibiting misrepresentation of the origin of that material, orrequiring that modified versions of such material be marked inreasonable ways as different from the original version; or

- d) Limiting the use for publicity purposes of names of licensors orauthors of the material; or

- e) Declining to grant rights under trademark law for use of sometrade names, trademarks, or service marks; or

- f) Requiring indemnification of licensors and authors of thatmaterial by anyone who conveys the material (or modified versions ofit) with contractual assumptions of liability to the recipient, forany liability that these contractual assumptions directly impose onthose licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's “contributor version”.

A contributor's “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from

you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

ant-design

Website:

https://ant.design/

array-move

Website:

https://github.com/sindresorhus/array-move

License:

MIT License

Copyright (c) Sindre Sorhus <sindresorhus@gmail.com> (sindresorhus.com)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

lodash

Website:

https://lodash.com/

License:

Copyright OpenJS Foundation and other contributors <https://openjsf.org/>

Based on Underscore.js, copyright Jeremy Ashkenas, DocumentCloud and Investigative Reporters & Editors <http://underscorejs.org/>

This software consists of voluntary contributions made by many individuals. For exact contribution history, see the revision history available at https://github.com/lodash/lodash

The following license applies to all parts of this software except as documented below:

====

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

====

Copyright and related rights for sample code are waived via CC0. Sample code is defined as all source code displayed within the prose of the documentation.

CC0: http://creativecommons.org/publicdomain/zero/1.0/

====

Files located in the node_modules and vendor directories are externally maintained libraries used by this software which have their own licenses; we recommend you read them, as their terms may differ from the terms above.

mobx

Website:

https://mobx.js.org/README.html

ramda

Website:

https://ramdajs.com/

License:

The MIT License (MIT)

Copyright (c) 2013-2019 Scott Sauyet and Michael Hurley

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

rc-menu

Website:

https://github.com/react-component/menu

License:

The MIT License (MIT)

Copyright (c) 2014-present yiminghe

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

react

Website:

https://reactjs.org/

License:

MIT License

Copyright (c) Facebook, Inc. and its affiliates.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

react-dnd

Website:

https://github.com/react-dnd/react-dnd/

License:

The MIT License (MIT)

Copyright (c) 2015 Dan Abramov

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

react-router

Website:

https://github.com/ReactTraining/react-router

License:

MIT License

Copyright (c) React Training 2016-2018

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF

CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

rxjs

Website:

https://github.com/ReactiveX/rxjs

License:

Apache License

Version 2.0, January 2004

http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

"License" shall mean the terms and conditions for use, reproduction, and distribution as defined by Sections 1 through 9 of this document.

"Licensor" shall mean the copyright owner or entity authorized by the copyright owner that is granting the License.

"Legal Entity" shall mean the union of the acting entity and all other entities that control, are controlled by, or are under common control with that entity. For the purposes of this definition, "control" means (i) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (ii) ownership of fifty percent (50%) or more of the outstanding shares, or (iii) beneficial ownership of such entity.

"You" (or "Your") shall mean an individual or Legal Entity exercising permissions granted by this License.

"Source" form shall mean the preferred form for making modifications, including but not limited to software source code, documentation source, and configuration files.

"Object" form shall mean any form resulting from mechanical transformation or translation of a Source form, including but not limited to compiled object code, generated documentation, and conversions to other media types.

"Work" shall mean the work of authorship, whether in Source or Object form, made available under the License, as indicated by a copyright notice that is included in or attached to the work (an example is provided in the Appendix below).

"Derivative Works" shall mean any work, whether in Source or Object form, that is based on (or derived from) the Work and for which the editorial revisions, annotations, elaborations, or other modifications represent, as a whole, an original work of authorship. For the purposes of this License,

Derivative Works shall not include works that remain separable from, or merely link (or bind by name) to the interfaces of, the Work and Derivative Works thereof.

"Contribution" shall mean any work of authorship, including the original version of the Work and any modifications or additions to that Work or Derivative Works thereof, that is intentionally submitted to Licensor for inclusion in the Work by the copyright owner or by an individual or Legal Entity authorized to submit on behalf of the copyright owner. For the purposes of this definition, "submitted" means any form of electronic, verbal, or written communication sent to the Licensor or its representatives, including but not limited to communication on electronic mailing lists, source code control systems, and issue tracking systems that are managed by, or on behalf of, the Licensor for the purpose of discussing and improving the Work, but excluding communication that is conspicuously marked or otherwise designated in writing by the copyright owner as "Not a Contribution."

"Contributor" shall mean Licensor and any individual or Legal Entity on behalf of whom a Contribution has been received by Licensor and subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable copyright license to reproduce, prepare Derivative Works of, publicly display, publicly perform, sublicense, and distribute the Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of this License, each Contributor hereby grants to You a perpetual, worldwide, non-exclusive, no-charge, royalty-free, irrevocable (except as stated in this section) patent license to make, have made, use, offer to sell, sell, import, and otherwise transfer the Work, where such license applies only to those patent claims licensable by such Contributor that are necessarily infringed by their Contribution(s) alone or by combination of their Contribution(s) with the Work to which such Contribution(s) was submitted. If You institute patent litigation against any entity (including a cross-claim or counterclaim in a lawsuit) alleging that the Work or a Contribution incorporated within the Work constitutes direct or contributory patent infringement, then any patent licenses granted to You under this License for that Work shall terminate as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the Work or Derivative Works thereof in any medium, with or without modifications, and in Source or Object form, provided that You meet the following conditions:

(a) You must give any other recipients of the Work or Derivative Works a copy of this License; and

(b) You must cause any modified files to carry prominent notices stating that You changed the files; and

(c) You must retain, in the Source form of any Derivative Works that You distribute, all copyright, patent, trademark, and attribution notices from the Source form of the Work, excluding those notices that do not pertain to any part of the Derivative Works; and

(d) If the Work includes a "NOTICE" text file as part of its distribution, then any Derivative Works that You distribute must include a readable copy of the attribution notices contained within such NOTICE file, excluding those notices that do not pertain to any part of the Derivative Works, in at least one of the following places: within a NOTICE text file distributed as part of the Derivative Works; within the

Source form or documentation, if provided along with the Derivative Works; or, within a display generated by the Derivative Works, if and wherever such third-party notices normally appear. The contents of the NOTICE file are for informational purposes only and do not modify the License.

You may add Your own attribution notices within Derivative Works that You distribute, alongside or as an addendum to the NOTICE text from the Work, provided that such additional attribution notices cannot be construed as modifying the License. You may add Your own copyright statement to Your modifications and may provide additional or different license terms and conditions for use, reproduction, or distribution of Your modifications, or for any such Derivative Works as a whole, provided Your use, reproduction, and distribution of the Work otherwise complies with the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise, any Contribution intentionally submitted for inclusion in the Work by You to the Licensor shall be under the terms and conditions of this License, without any additional terms or conditions. Notwithstanding the above, nothing herein shall supersede or modify the terms of any separate license agreement you may have executed with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade names, trademarks, service marks, or product names of the Licensor, except as required for reasonable and customary use in describing the origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or agreed to in writing, Licensor provides the Work (and each Contributor provides its Contributions) on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE. You are solely responsible for determining the appropriateness of using or redistributing the Work and assume any risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall any Contributor be liable to You for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising as a result of this License or out of the use or inability to use the Work (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if such Contributor has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing the Work or Derivative Works thereof, You may choose to offer, and charge a fee for, acceptance of support, warranty, indemnity, or other liability obligations and/or rights consistent with this License. However, in accepting such obligations, You may act only on Your own behalf and on Your sole responsibility, not on behalf of any other Contributor, and only if You agree to indemnify, defend, and hold each Contributor harmless for any liability incurred by, or claims asserted against, such Contributor by reason of your accepting any such warranty or additional liability.

Support

For help with a component or product, please submit a technical support request at http://support.lmi3d.com/.

If you are unable to submit a support request or prefer to contact LMI by phone or email, use the contact information below.

> Response times for phone or email support requests are longer than requests submitted through LMI's support request system.

North America

Phone	Vancouver +1 833 Gocator (+1 833 462 2867) Detroit +1 248 298 2839
Fax	+1 604 516 8368
Email	support@lmi3d.com

Europe

Phone	+31 45 850 7000
Fax	+31 45 574 2500
Email	support@lmi3d.com

Contact

LMI Technologies has sales offices and distributors worldwide. All contact information is listed at lmi3D.com/contact.