USER MANUAL

Gocator Multi-Point Scanners

Gocator 205, 210, 230 & 250

Firmware version: 5.3.33.49

Document revision: A

Copyright

Proprietary

Trademarks and Restrictions

Information contained within this manual is subject to change.

This product is designated for use solely as a component and as such it does not comply with the standards relating to laser products specified in U.S. FDA CFR Title 21 Part 1040.

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

Introduction

This documentation describes how to connect, configure, and use a Gocator. It also contains reference information on the device's protocols and job files, as well as an overview of the development kits you can use with Gocator. Finally, the documentation describes the Gocator emulator.

The documentation applies to the following:

- Gocator 200 series

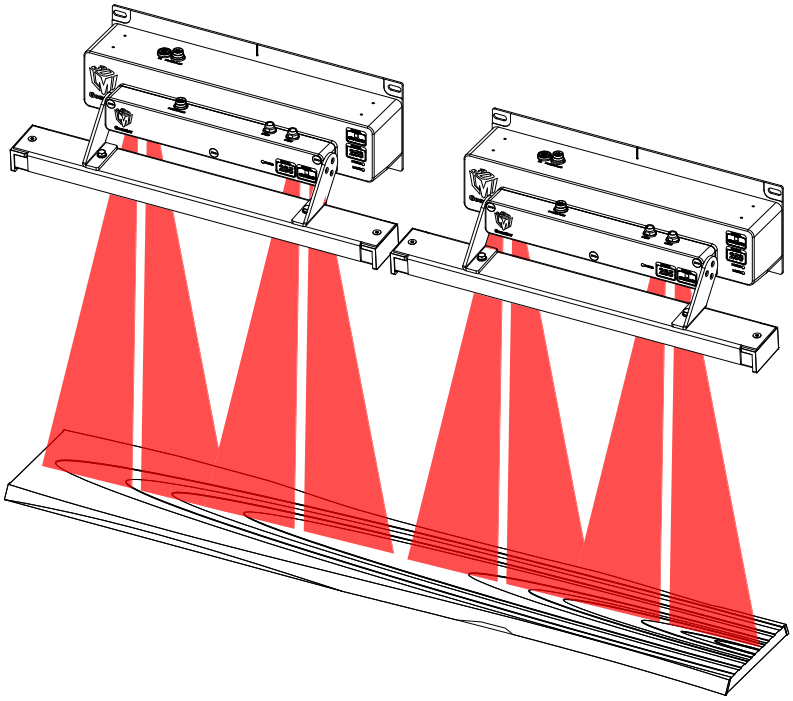
Notational Conventions
This documentation uses the following notational conventions:

⚠	Follow these safety guidelines to avoid potential injury or property damage.

🗋	Consider this information in order to make best use of the product.

Gocator 200 Series Overview

The Gocator 200 series of scanners lets you build scanning systems using a modular design that allows you to mix 3D profiles, tracheid detection, and color vision. Gocator 200 series systems can be used to perform automatic wood-grading in machine centers found in saw and planer mills. The Gocator 200 series is designed for transverse board scanning.



Gocator 200 series system scanning board

Typically, you will use the two provided software development kits to implement system configuration and data processing: GoSDK and GoWebScanSDK.

- GoSDK: Provides lower level device control and communication functions.
- GoWebScanSDK: Provides a library of common functions found in wood-scanning applications and returns complete board data.

Measurements performed on the returned data are user-implemented, which gives you maximum flexibility in system design. For more information on the development kits, see *GoSDK and GoWebScanSDK* on page 61.

Testing of settings on individual devices during initial setup, as well as troubleshooting and system monitoring, can be performed using the built-in web interface (for more information, see *Gocator Web Interface* on page 102.)

The measurement tools built into the Gocator web interface are not intended for wood applications.

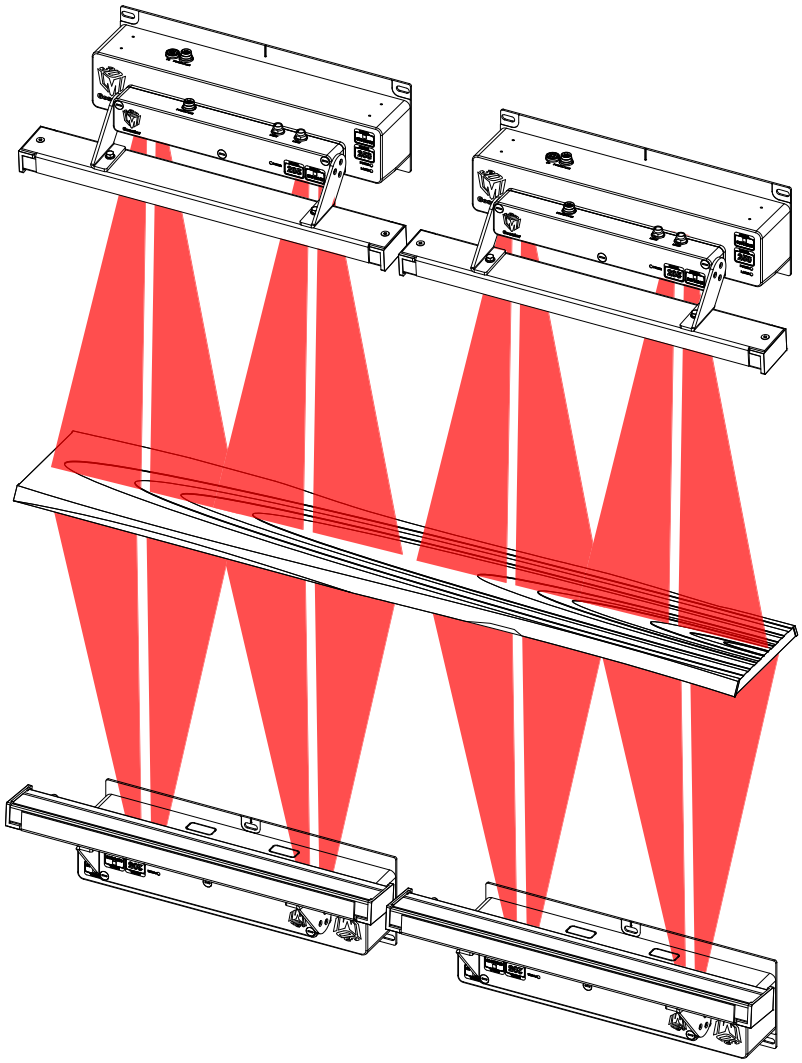
Gocator 210, 230, 240, and 250
Gocator 210, 230, 240, and 250 scanners are multi-point laser devices that return profiles of material passing beneath them. Gocator 230/240/250 scanners can run at 4 kHz with an 8" measurement range; otherwise, with a 10" measurement range, the scan rate is limited to 3 kHz

Gocator 250 scanners can also be used to measure the tracheid effect in boards. For more information, see *Tracheids* on page 13.

Gocator 205
Gocator 205 is a bolt-on color vision module that supports the detection and measurement of surface defects such as knots, splits, and rot. Measurement of their size and location is key to grade-based recovery optimization. It is used in conjunction with one or two strobed white light LED light bars that provide efficient and effective illumination. This provides a simple and reliable method of capturing features on material surfaces along the length of the system.

True Differential Measurement

Gocator 200 series scanners can be aligned co-planar top-to-bottom and side-to-side down the length of the system to provide large-scale differential measurement. This prevents profile measurement errors that could be introduced by hard-to-control mechanical issues such as chain vibration and material bouncing as it is transported through the scanner system.



Gocator 200 series system scanning board - Top and bottom system

High Scan Rates

Gocator 200 series multi-point scanners provide high profile scan rates yet at the same time maintain excellent dark wood performance (equivalent to level 19 on the Kodak gray scale chart), insensitivity to laser saturation, and immunity to ambient light.

For more information at model-dependent speeds and resolutions (for more information, see *Gocator 200 Series* on page 342).

Temperature-Compensated Ranges

Gocator 200 series scanners are calibrated at several temperature points in their operational range to ensure reliable and accurate range measurement throughout changes in ambient temperature.

Tracheids

When a laser spot is projected onto healthy tracheid wood cells, laser light is scattered into the cells in the direction of cell growth. If the wood fiber is dead (as in a knot), the laser light does not scatter. This effect is measured via GoSDK by fitting an ellipse to the laser spot, which is nominally circular in the absence of any tracheid effect. The following information based on the ellipse fit is available:

- Angle of the major axis of the ellipse
- Area of the ellipse
- Length of the major axis of the ellipse
- Length of the minor axis of the ellipse
- Scatter, defined as the ratio of the major and minor axis

Migrating from chroma+scan

Gocator 200 series multi-point sensors can be used to implement the same kinds of transverse wood-processing systems possible with chroma+scan scanners. In general, specifications for the physical setup (for example, mounting and frame design) of systems based on Gocator 200 series sensors are the same. Furthermore, clearance distances and measurement ranges are equivalent to chroma+scan scanners. Resolution and speed are the same if not better than the corresponding chroma+scan scanners.

However, the hardware and software architecture has changed to give system developers full, low-level control of system functionality. The following table gives a brief overview of the differences between chroma+scan systems and Gocator 200 series systems:

	chroma+scan	Gocator 200 series
Client PC with...		
	Client application written using Zen API and FireSync Host Protocol (OR provided kClient application)	User-written application developed using GoSDK and GoWebScanSDK
Station PC with...		
	Proprietary, closed source Station services; OS-dependent Ethernet drivers	Station PC not required: Station services provided by GoWebScanSDK

The main difference is that systems developed using Gocator 200 series sensors no longer require Station PCs running specialize software to provide Station services. The functionality provided by the closed source Station services is now accessed in GoWebScanSDK, which, along with the base GoSDK, is used to write a client application to control and run a system based on Gocator 200 series scanners, using an instance of GoWebScan. Note that elsewhere in this documentation and in the source code, the term "station" is used loosely to refer to the control of a group of sensors. Source code is available for both SDKs.

For initial system setup, you can use the Gocator web interface for troubleshooting and to quickly view data from individual scanners. For more information on this interface, see *Gocator Web Interface* on page 102.

The following table lists the main tasks and processes implemented using GoWebScanSDK or GoSDK. Note that most tasks and process listed under the GoWebScanSDK column can be implemented using the lower level GoSDK. GoWebScanSDK simply provides classes useful for typical wood scanning applications, which greatly reduces development time.

Task		GoSDK	GoWebScanSDK	Gocator Web Interface
Misc				
	Initial test setup of individual sensors			✓
	Troubleshooting and monitoring of live laser profile data from a single sensor			✓
chroma+scan client				
	Sensor enumeration and assignment	✓		
	Modes of operation		✓	
	Sensor upgrade logic	✓		
	Sensor and system health	✓		
	System, group, sensor setup	✓		
	Start/stop	✓		
Station services				
	System architecture mapping	✓		
	Sensor synchronization (time or encoder mode)	✓		
	Top/bottom staggering	✓		
	Data storage structures	✓		
	System alignment (for information on alignment target specifications, see *Preparing the Alignment Bar* on page 42) Multi-station alignment		✓	
	Data acquisition and transfer to data processor (GoWebScan)	✓		

Task		GoSDK	GoWebScanSDK	Gocator Web Interface
	Profile Merging (between sensors)		✓	
	Tracheid processing and merging (tracheid map)		✓	
	Profile processing		✓	
	Vision processing		✓	
	Tracheid processing		✓	
	Vision & Profile merging (between Gocator 205 and Gocator 210/230/240/250)		✓	
	Board state machine		✓	
	Board / target detection logic		✓	
	Web mode		✓	
	Edge filtering		✓	
	White balance		✓	
	Bayer decoding		✓	
	X and Y resampling		✓	
	Resampling color pixel to match height of board		✓	
	Event channel	✓		
	Board post-processing (spike filtering, gap filling and blending)		✓	
	Sensor multiplexing	✓		
	Retrieve digital model of scanned object from GoWebScanSDK.	✓		
	Use replay data (previously recorded scan data)		✓	
	Outputting calibration bar image after calibration		✓	
	Using 10-inch measurement range	✓		

For more information on GoWebScanSDK, see *GoWebScanSDK* on page 70.

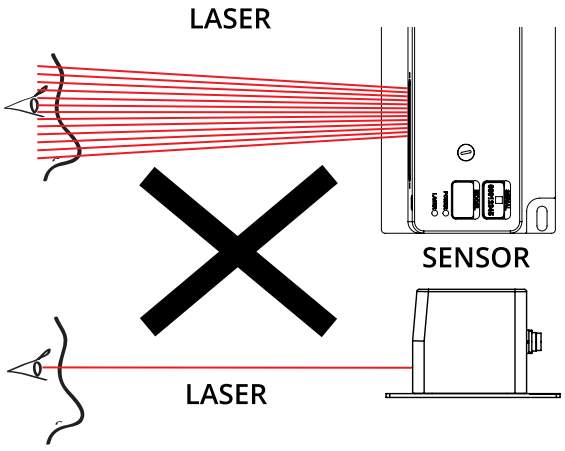
Safety and Maintenance

The following sections describe the safe use and maintenance of Gocator sensors.

Laser Safety

Gocator sensors contain semiconductor lasers that emit visible or invisible light and are designated as Class 2, 2M, Class 3R, or Class 3B, depending on the laser option. For more information on the laser classes used in these sensors, *Laser Classes* on the next page.

Gocator sensors are referred to as *components*, indicating that they are sold only to qualified customers for incorporation into their own equipment. These sensors do not incorporate safety items that the customer may be required to provide in their own equipment (e.g., remote interlocks, key control; refer to the references below for detailed information). As such, these sensors do not fully comply with the standards relating to laser products specified in IEC 60825-1:2014 and FDA CFR Title 21 Part 1040.



WARNING: DO NOT LOOK DIRECTLY INTO THE LASER BEAM

⚠ Use of controls or adjustments or performance of procedures other than those specified herein may result in hazardous radiation exposure.

References

1. *International standard IEC 60825-1:2014 (2001-08) consolidated edition*, Safety of laser products – Part 1: Equipment classification, requirements and user's guide.

2. *Technical report 60825-10*, Safety of laser products – Part 10. Application guidelines and explanatory notes to IEC 60825-1:2014.
3. *Laser Notice No. 50*, FDA and CDRH (https://www.fda.gov/Radiation-EmittingProducts/ElectronicProductRadiationControlProgram/default.htm)

Laser Classes

Class 3B laser components

Class 3B components are unsafe for eye exposure.

- Usually only eye protection is required. Protective gloves may also be used.
- Diffuse reflections are safe if viewed for less than 10 seconds at a minimum distance of 13 cm.
- There is a risk of fire if the beam encounters flammable materials.
- The laser area must be clearly identified.
- Use a key switch or other mechanism to prevent unauthorized use.
- Use a clearly visible indicator to show that a laser is in use, such as “Laser in operation.”
- Restrict the laser beam to the working area.
- Ensure that there are no reflective surfaces in the working area.

For more information, see *Precautions and Responsibilities* below.

Precautions and Responsibilities

Precautions specified in IEC 60825-1:2014 and FDA CFR Title 21 Part 1040 are as follows:

Requirement	Class 3B
Remote interlock	Required*
Key control	Required – cannot remove key when in use*
Power-on delays	Required*
Beam attenuator	Required*
Emission indicator	Required*
Warning signs	Required*
Beam path	Terminate beam at useful length
Specular reflection	Prevent unintentional reflections
Eye protection	Required under special conditions

Requirement	Class 3B
Laser safety officer	Required
Training	Required for operator and maintenance personnel

**LMI Class 3B laser components do not incorporate these laser safety items. These items must be added and completed by customers in their system design. For more information, see Class 3B Responsibilities below.*

Class 3B Responsibilities

LMI Technologies has filed reports with the FDA to assist customers in achieving certification of laser products. These reports can be referenced by an accession number, provided upon request. Detailed descriptions of the safety items that must be added to the system design are listed below.

Remote Interlock

A remote interlock connection must be present in Class 3B laser systems. This permits remote switches to be attached in serial with the keylock switch on the controls. The deactivation of any remote switches must prevent power from being supplied to any lasers.

Key Control

A key operated master control to the lasers is required that prevents any power from being supplied to the lasers while in the OFF position. The key can be removed in the OFF position but the switch must not allow the key to be removed from the lock while in the ON position.

Power-On Delays

A delay circuit is required that illuminates warning indicators for a short period of time before supplying power to the lasers.

Beam Attenuators

A permanently attached method of preventing human access to laser radiation other than switches, power connectors or key control must be employed.

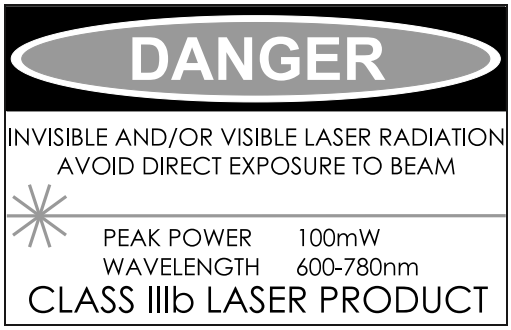
Emission Indicator

It is required that the controls that operate the sensors incorporate a visible or audible indicator when power is applied and the lasers are operating. If the distance between the sensor and controls is more than 2 meters, or mounting of sensors intervenes with observation of these indicators, then a second power-on indicator should be mounted at some readily-observable position. When mounting the warning indicators, it is important not to mount them in a location that would require human exposure to the laser emissions. User must ensure that the emission indicator, if supplied by OEM, is visible when viewed through protective eyewear.

Warning Signs

Laser warning signs must be located in the vicinity of the sensor such that they will be readily observed.

Examples of laser warning signs are as follows:



FDA warning sign example

IEC warning sign example

Systems Sold or Used in the USA

Systems that incorporate laser components or laser products manufactured by LMI Technologies require certification by the FDA.

Customers are responsible for achieving and maintaining this certification.

Customers are advised to obtain the information booklet *Regulations for the Administration and Enforcement of the Radiation Control for Health and Safety Act of 1968: HHS Publication FDA 88-8035*.

This publication, containing the full details of laser safety requirements, can be obtained directly from the FDA, or downloaded from their web site at https://www.fda.gov/Radiation-EmittingProducts/ElectronicProductRadiationControlProgram/default.htm.

Electrical Safety

⚠	Failure to follow the guidelines described in this section may result in electrical shock or equipment damage.

Sensors should be connected to earth ground

All sensors should be connected to earth ground through their housing. All sensors should be mounted on an earth grounded frame using electrically conductive hardware to ensure the housing of the sensor is connected to earth ground. Use a multi-meter to check the continuity between the sensor connector and earth ground to ensure a proper connection.

Use a suitable power supply

The power supply used with sensors should be an isolated supply with inrush current protection or be able to handle a high capacitive load. Verify the voltage input requirements for your sensor in the sensor's specifications; for specifications, see *Gocator 200 Series* on page 342.

Use care when handling powered devices

Wires connecting to the sensor should not be handled while the sensor is powered. Doing so may cause electrical shock to the user or damage to the equipment.

Handling, Cleaning, and Maintenance

🗋	Dirty or damaged sensor windows (emitter or camera) can affect accuracy. Use caution when handling the sensor or cleaning the sensor's windows.

Keep sensor windows clean
Use dry, clean air to remove dust or other dirt particles. If dirt remains, clean the windows carefully with a soft, lint-free cloth and non-streaking glass cleaner or isopropyl alcohol. Ensure that no residue is left on the windows after cleaning.

Turn off lasers when not in use
LMI Technologies uses semiconductor lasers in Gocator sensors. To maximize the lifespan of the sensor, turn off the laser when not in use.

Avoid excessive modifications to files stored on the sensor
Sensor settings are stored in flash memory inside the sensor. Flash memory has an expected lifetime of 100,000 writes. To maximize lifetime, avoid frequent or unnecessary file save operations.

Environment and Lighting

Avoid strong ambient light sources
The imager used in this product is highly sensitive to ambient light. Do not operate this device near windows or lighting fixtures that could influence measurement or data acquisition. If the unit must be installed in an environment with high ambient light levels, a lighting shield or similar device may need to be installed to prevent light from affecting measurement.

Avoid installing sensors in hazardous environments
To ensure reliable operation and to prevent damage to sensors, avoid installing the sensor in locations

- that are humid, dusty, or poorly ventilated;
- with a high temperature, such as places exposed to direct sunlight;
- where there are flammable or corrosive gases;
- where the unit may be directly subjected to harsh vibration or impact;
- where water, oil, or chemicals may splash onto the unit;
- where static electricity is easily generated.

Ensure that ambient conditions are within specifications
Sensors are suitable for operation between 0–50° C and 25–85% relative humidity (non-condensing). Measurement error due to temperature is limited to 0.015% of full scale per degree C. The storage temperature is -30–70° C.

The Master network controllers are similarly rated for operation between 0–50° C.

⚠	The sensor must be heat-sunk through the frame it is mounted to. When a sensor is properly heat sunk, the difference between ambient temperature and the temperature reported in the sensor's health channel is less than 15° C.

⚠	Sensors are high-accuracy devices, and the temperature of all of its components must therefore be in equilibrium. When the sensor is powered up, a warm-up time of at least one hour is required to reach a consistent spread of temperature in the sensor.

Analog Support

Most G2 and all G3 and G200 sensors shipped on or after July 18, 2022, no longer support analog output. When a non-analog sensor is used with an existing firmware, analog configuration will still be displayed in the Output page in the Gocator web interface, but will be non-functional. G1 sensors and vertical model sensors will continue to support analog output. For more information, see the notification on LMI's website (https://lmi3d.com/resource/gocator-analog-output-removal/).

Getting Started

The following sections provide system and hardware overviews, in addition to installation and setup procedures.

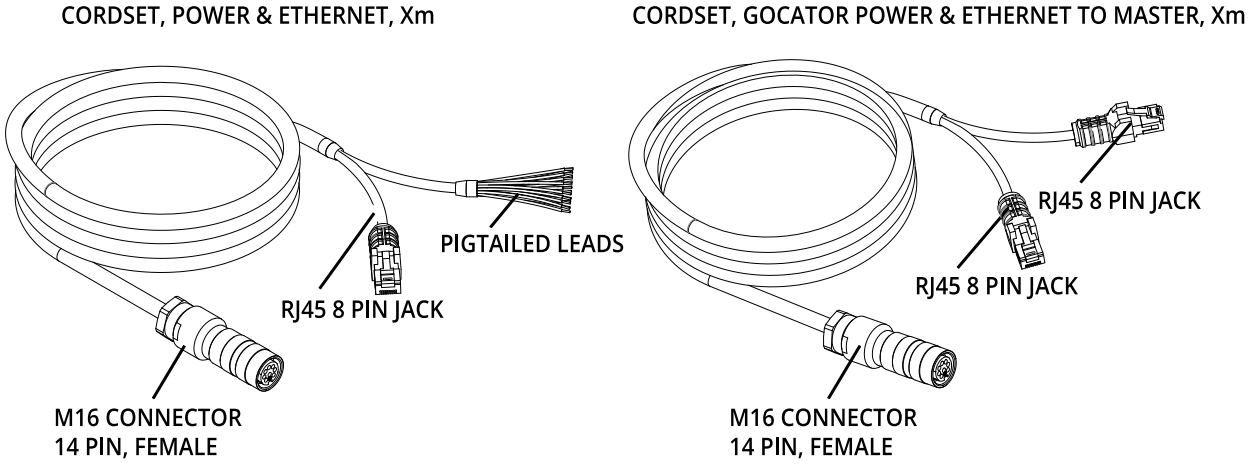
Hardware Overview

The following sections describe Gocator and its associated hardware.

Gocator Cordsets

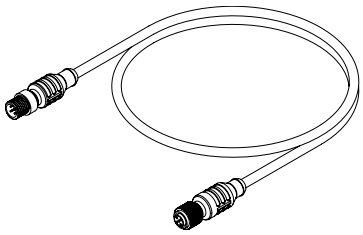
Gocator multi-point scanners use the Power & Ethernet cordset. The color vision module uses two types of cordsets: the Power & Ethernet cordset and up to two light bar cordsets.

The Power & Ethernet cordset provides power and laser safety to the sensor. It is also used for sensor communication via 1000 Mbit/s Ethernet with a standard RJ45 connector. The Master version of the Power & Ethernet cordset provides direct connection between the sensor and a Master network controller, excluding Master 100 (for more information, see *Master Network Controllers* on page 357).



The maximum cordset length is 60 m.

The Gocator 205 vision module uses a light bar cordset to provide power to a light bar; up to two light bars can be connected to the module.



LMI provides a 0.5 meter cordset for use when the light bar is mounted directly to the Gocator 205 scanner using the optional mounting hardware kit. When the light bar is mounted in another location, you may create your own cordsets based on the following specifications:

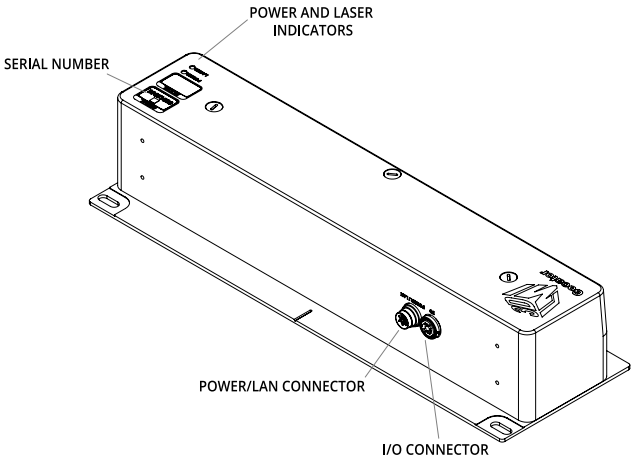
- 5 Pin M12 male to female cable
- 250 V / 4 amp rating
- 60 meter maximum

For pinout details, see *Gocator Power/LAN Connector* on page 353.

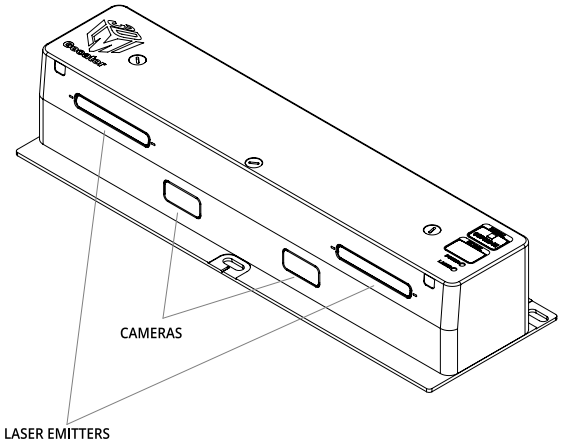
See *Accessories* on page 369 for cordset lengths and part numbers. Contact LMI for information on creating cordsets with customized lengths and connector orientations.

Gocator Sensor

Gocator 200 series sensors are transverse scanning devices.



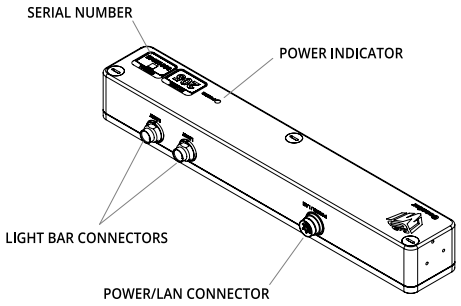
Gocator 210/230/250 (top)



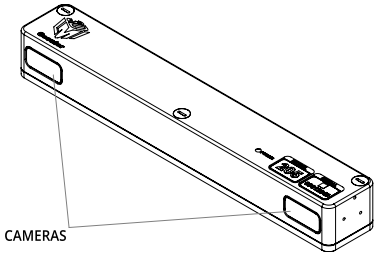
Gocator 210/230/250 (bottom)

Item	Description
Cameras	Observe laser light reflected from target surfaces.
Laser Emitters	Emit structured light for laser profiling.
I/O Connector	Not used.
Power / LAN Connector	Accepts power and laser safety signals and connects to 1000 Mbit/s Ethernet network. Connect a Power/LAN cordset to this connector.
Power Indicator	Illuminates when power is applied (blue).
Laser Indicator	Illuminates when laser safety input is active (amber).
Serial Number	Unique serial number.

Gocator Color Vision Module



Gocator 205 (top)



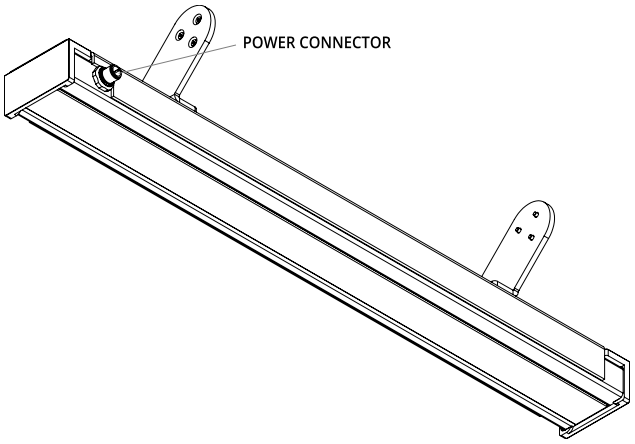
Gocator 205 (bottom)

Item	Description
Cameras	The cameras observe LED light reflected from target surfaces.
Power / LAN Connector	Accepts power and connects to 1000 Mbit/s Ethernet network. Connect a Power/LAN cordset to this connector.
Light Bar Connectors	Provide power for up to two light bars. Connect a light bar to Gocator 205 cordset to this connector.
Power Indicator	Illuminates when power is applied (blue).
Serial Number	Unique serial number.

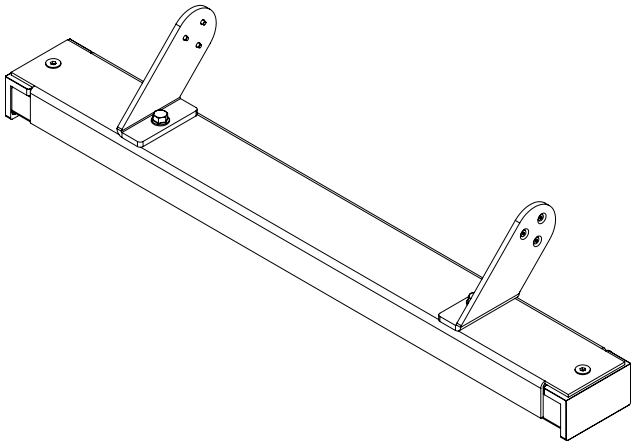
Gocator Light Bar

One or two light bars are used in a Gocator system to provide light for a Gocator 205 color module. The Gocator 205 color module can provide power to both light bars. Light bars are connected using a light bar to Gocator 205 cordset.

One light bar can be mounted on the Gocator 205 color module using optional mounting hardware.



Gocator 200 series light bar (rear)



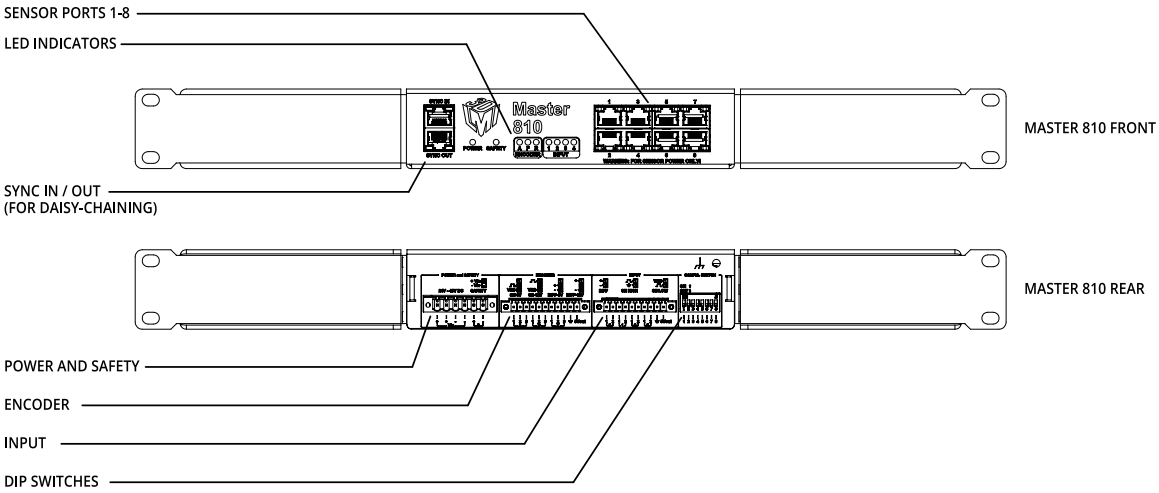
Gocator 200 series light bar (front)

Master 810 / 2410

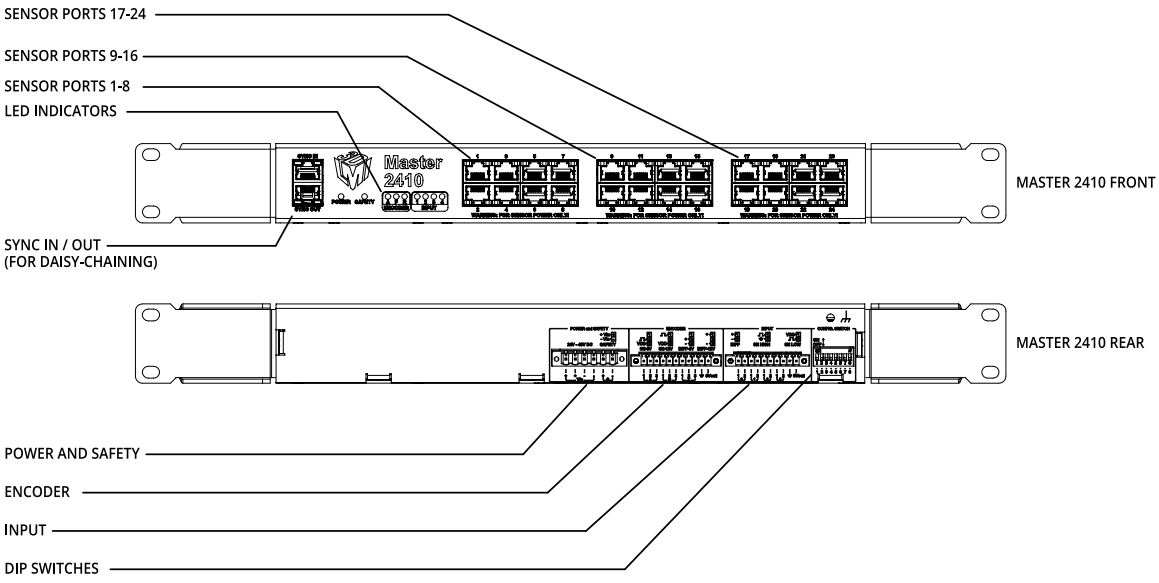
The Master 810 and 2410 network controllers let you connect multiple sensors to create a multi-sensor system:

- Master 810 accepts up to eight sensors
- Master 2410 accepts up to twenty-four sensors

Both models let you divide the quadrature frequency of a connected encoder to make the frequency compatible with the Master, and also set the debounce period to accommodate faster encoders. For more information, see *Configuring Master 810* on page 39. (Earlier revisions of these models lack the DIP switches.)



Master 810



Master 2410

Item	Description
Sensor Ports	Master connection for sensors (no specific order required).
Power and Safety	Power and safety connections. Safety input must be high in order to scan with laser-based sensors.
Encoder	Accepts encoder signal.
Input	Accepts digital input.
DIP Switches	Configures the Master (for example, allowing the device to work with faster encoders). For information on configuring Master 810 and 2410 using the DIP switches, see *Configuring Master 810* on page 39.

For pinout details, see *Master 810/2410* on page 358.

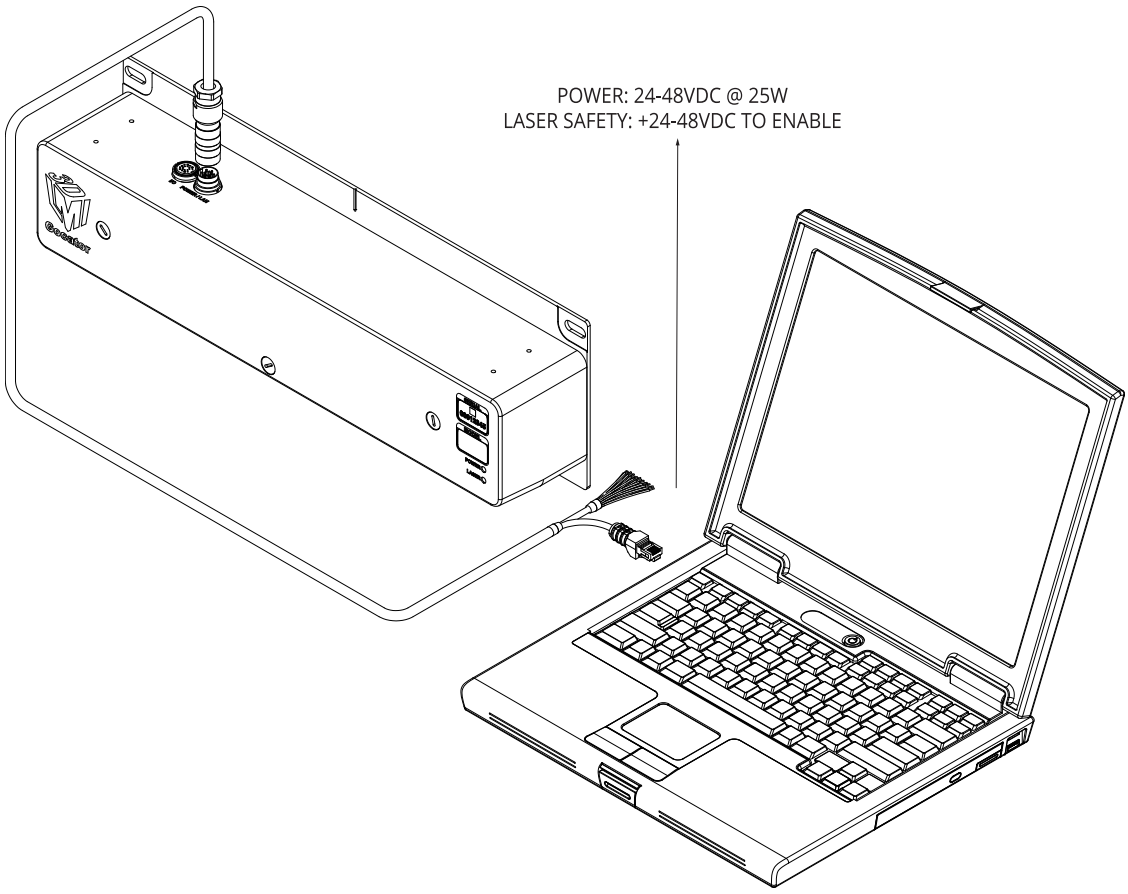
System Overview

Gocator sensors can be installed and used in a variety of scenarios. Sensors can be connected as standalone devices, dual-sensor systems, or multi-sensor systems.

Standalone System

Standalone systems are typically used when only a single sensor is required. The device can be connected to a computer's Ethernet port for setup and can also be connected to devices such as encoders, photocells, or PLCs.

Standalone systems are not typically used in wood applications.

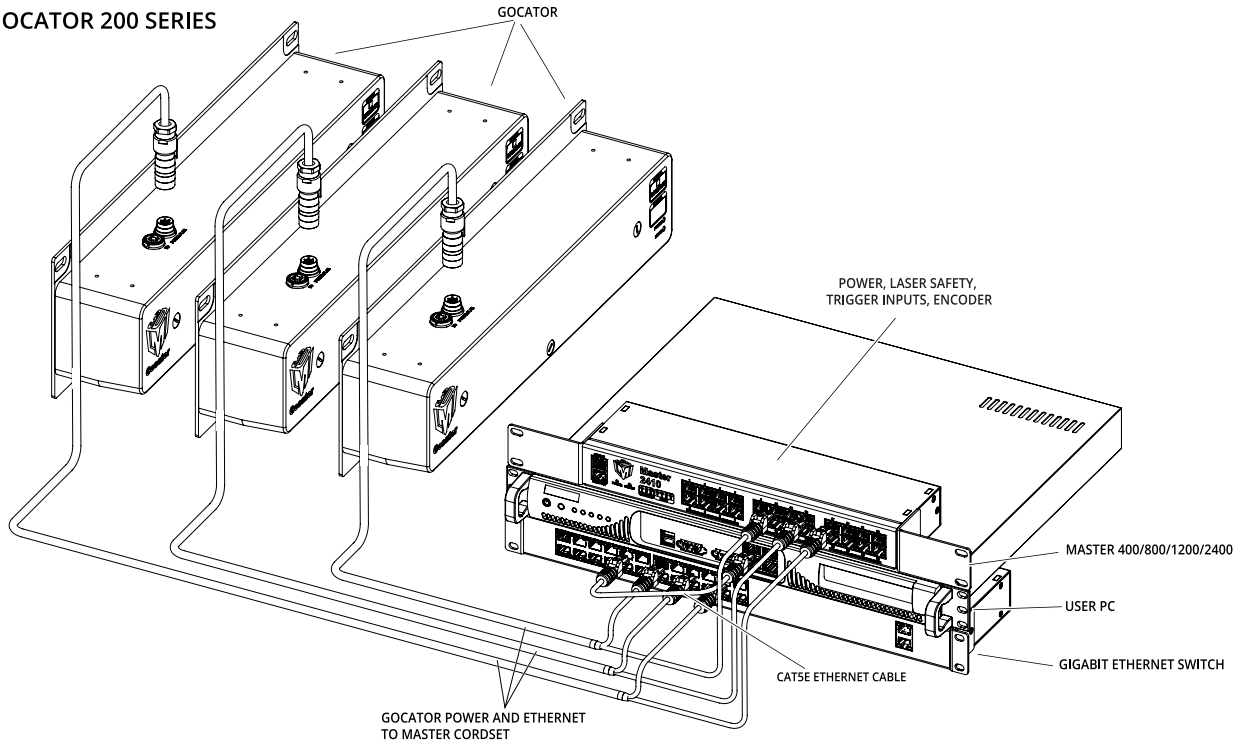


Multi-Sensor System

A Master network controller (excluding Master 100) can be used to connect two or more sensors into a multi-sensor system. Master cordsets are used to connect the sensors to a Master. The Master provides a single point of connection for power, safety, encoder, and digital inputs. A Master 400/800/810/1200/2400/2410 can be used to ensure that the scan timing is precisely synchronized across sensors. Sensors and client computers communicate via an Ethernet switch (1 Gigabit/s recommended).

Multi-sensor Gocator 200 systems can only be configured using GoSDK. For more information, see *GoSDK and GoWebScanSDK* on page 61.

GOCATOR 200 SERIES



Installation

The following sections provide grounding, mounting, and orientation information.

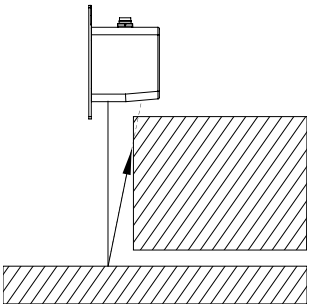
Mounting

Scanners

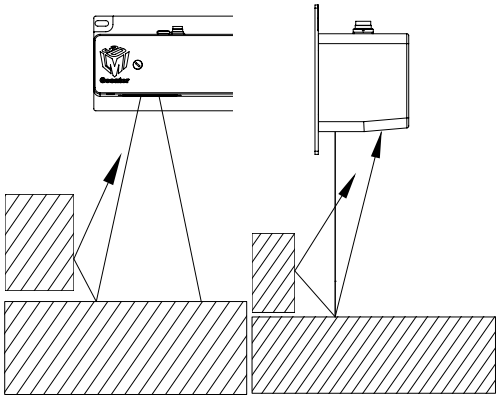
A Gocator multi-point scanner is mounted through three holes on the device's back plate. Refer to the dimension drawings of the sensors in *Specifications* on page 342 for the appropriate screw diameter, pitch, and length, and bolt hole diameter.

Scanners can be mounted with either M8 or 5/16" hardware. Provision to adjust the position and orientation of the sensor to align its laser plane with the laser planes of other sensors, above and beside, is highly recommended. This alignment is critical to prevent sensor crosstalk and ensure true differential measurements; aligned laser planes also provide a better appearance to the end user of the system.

Scanners should not be installed near objects that might occlude a camera's view of the projected light.



Scanners should not be installed near surfaces that might create unanticipated light reflections.



⚠	The sensor must be heat sunk through the frame it is mounted to. When a sensor is properly heat sunk, the difference between ambient temperature and the temperature reported in the sensor's health channel is less than 15° C.

⚠	Gocator sensors are high-accuracy devices. The temperature of all of its components must be in equilibrium. When the sensor is powered up, a warm-up time of at least one hour is required to reach a consistent spread of temperature within the sensor.

⚠	To prevent accidental laser light exposure, in multi-sensor systems, attach light shields between scanners. For hole positions, see the scanner specification drawings. (LMI does not supply light shields.)

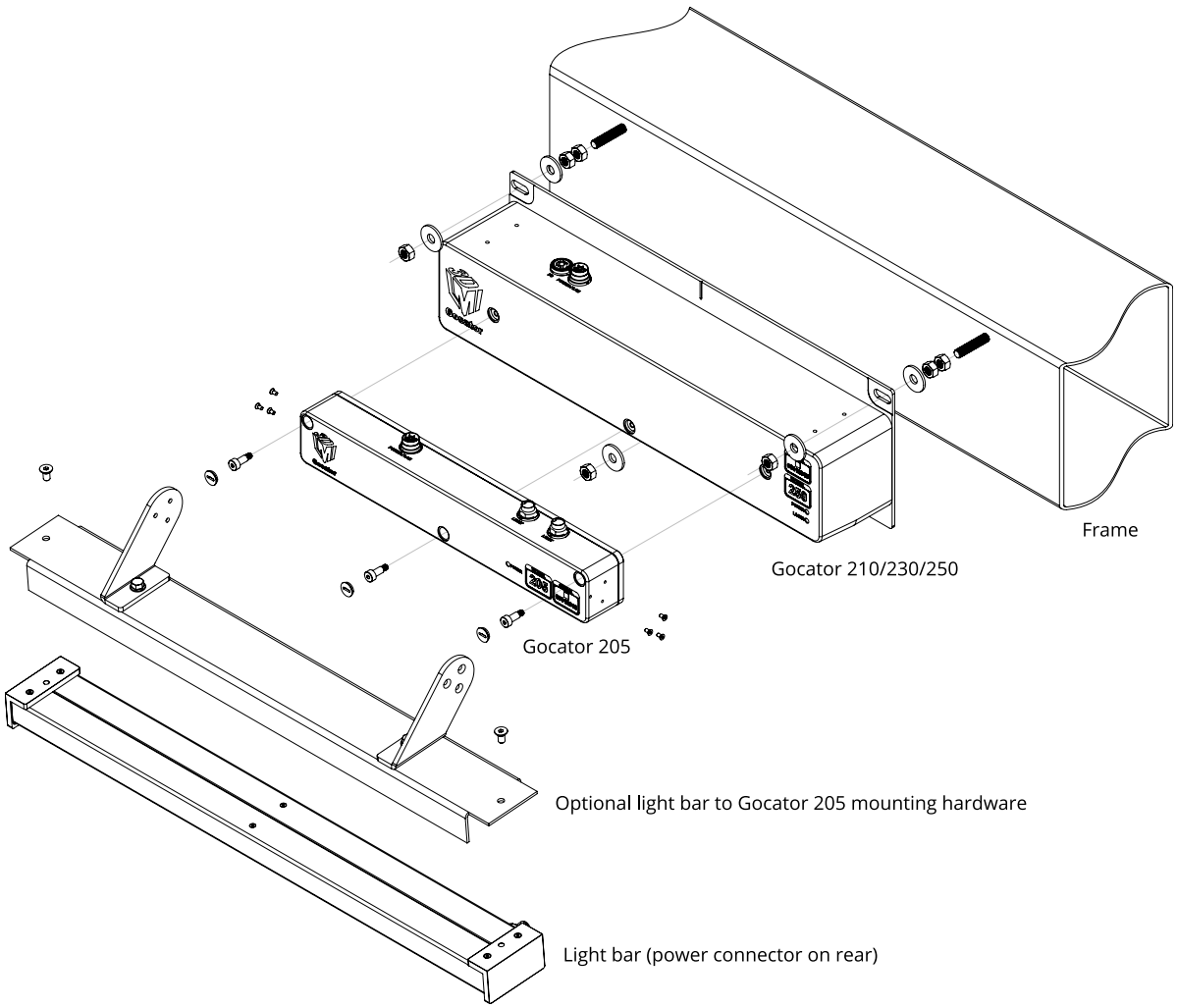
Camera Module and Light Bars

A Gocator multi-point scanner can optionally be used in conjunction with a camera module and one or two light bars. The camera module is mounted to three holes on the front of the scanner. Light bars in turn are mounted to the camera module using optional mounting hardware.

Assembling a Sensor + Camera + Light Bar System

The following shows how a Gocator 205 and a light bar mount to a Gocator 200 series sensor.

🗋	Light bar to Gocator 205 mounting hardware is optional.



As with standalone sensors, systems should not be installed near surfaces that might occlude a camera's view of the projected light (laser or white light) or surfaces that might create unanticipated light reflections. For full details, see the envelope drawing for your sensor in *Gocator 200 Series* on page 342.

⚠	To prevent accidental laser light exposure, in multi-sensor systems, attach light shields between scanners. For light shield hole positions and specifications, see the scanner specification drawings. (LMI does not supply light shields.)

🗋	Light bar to Gocator 205 mounting hardware is optional.

Frame Design

The scan frame supports the scanners above and below the transport mechanism to the maximum size of the targets to be scanned. Typically, for wood applications, 10 to 12 scanners are mounted above the transport mechanism and another 10 to 12 sensors are mounted below. This provides complete coverage of the top and bottom surfaces of the boards. In some wood applications, scanners are only mounted above the targets.

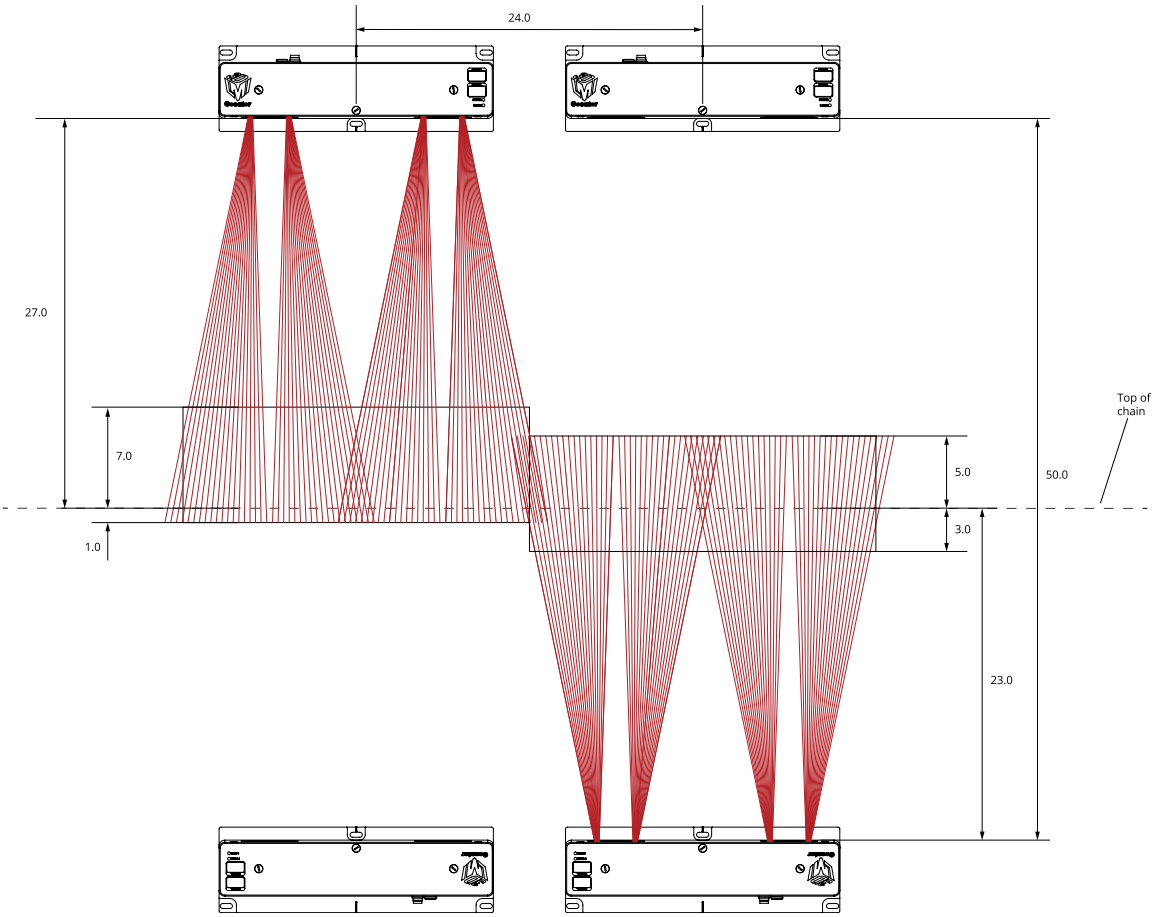
In web applications, fewer scanners are often used.

A means of adjusting the scanners' location and orientation is required to ensure that the scanners do not experience crosstalk and also to ensure that the target thickness is measured differentially.

The frame must also be connected to earth ground to provide a ground path for the sensors.

Gocator 230/240/250

The following suggested frame design applies to Gocator 230/240/250, with or without the optional camera module (Gocator 205). The camera module is not shown in these illustrations. Note that this design is appropriate for systems configured to use either an 8" or 10" measurement region.

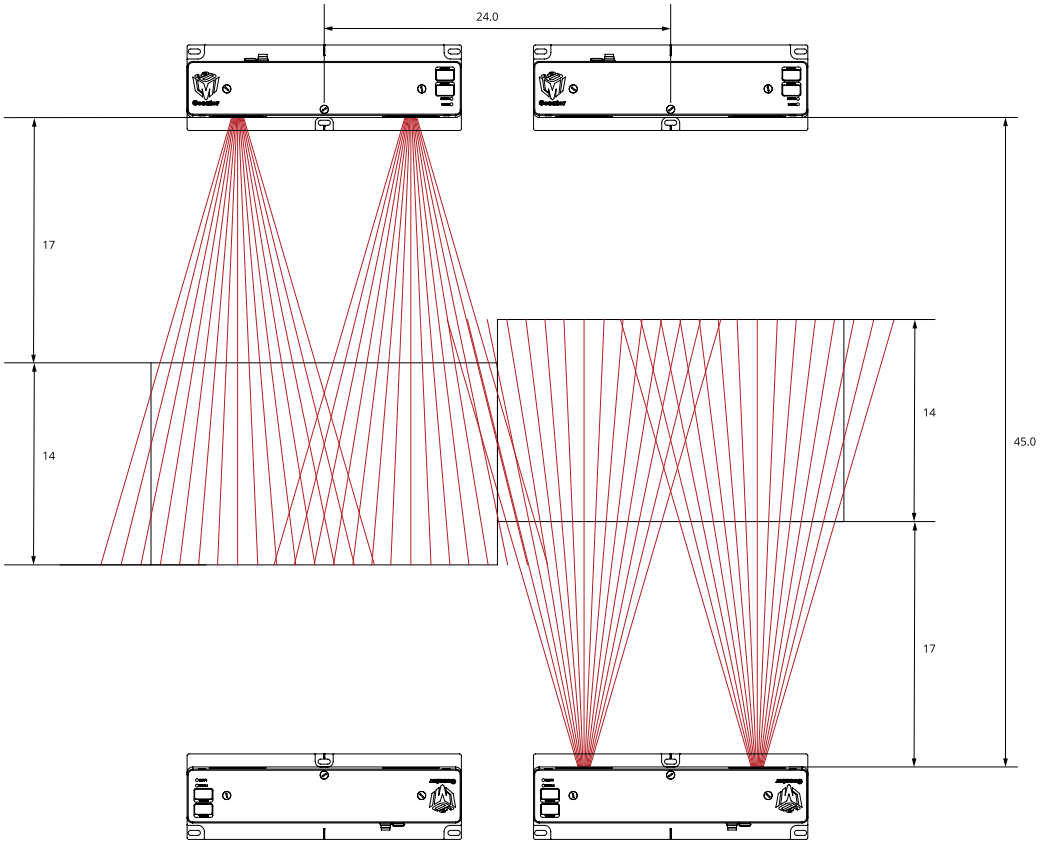


The configuration above provides a full 6-inch overlapped scan zone starting from 1 inch below the chainways and extends to 5 inches above the chainways. The extended range of these scanners provides an additional 2 inches of coverage from the top sensors above this overlapped zone, as well as an additional 2 inches below the overlapped zone from the bottom sensors.

An alternative system configuration for Gocator 230/240/250 is to overlap the sensors for the full 8 inches of the sensors' range by moving the bottom sensors 2 inches closer to the chainways. This would reduce the sensor separation to 48 inches and provide 7 inches of scan zone above the chain and 1 inch below.

Gocator 210

The following suggested frame design applies to Gocator 210.



Grounding

Components of a sensor system should be properly grounded.

Gocator

Gocator sensors should be grounded to the earth/chassis through their housings and through the grounding shield of the Power and Ethernet cordset. Sensors have been designed to provide adequate grounding through their mounting screws. Always check grounding with a multi-meter to ensure electrical continuity between the mounting frame and the sensor's connectors.

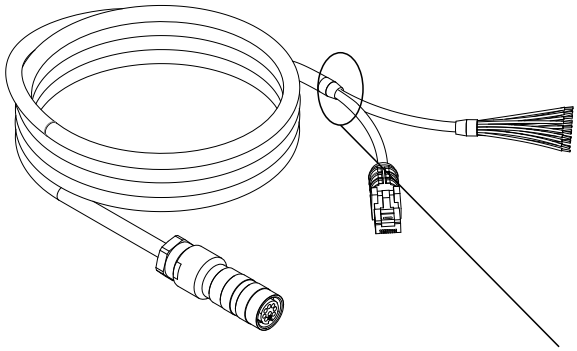
	The frame or electrical cabinet that the sensor is mounted to must be connected to earth ground.

Recommended Practices for Cordsets

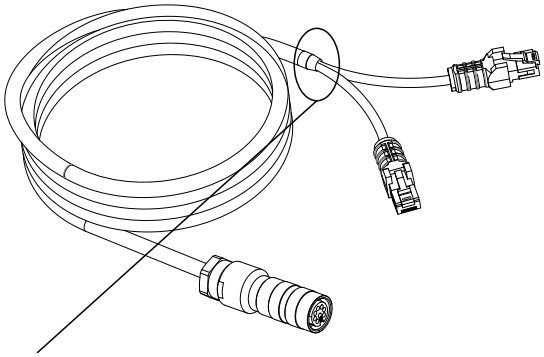
If you need to minimize interference with other equipment, you can ground the Power & Ethernet or the Power & Ethernet to Master cordset (depending on which cordset you are using) by terminating

the shield of the cordset before the split. The most effective grounding method is to use a 360-degree clamp.

CORDSET, POWER & ETHERNET, Xm



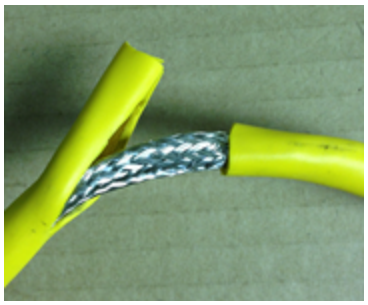
CORDSET, GOCATOR POWER & ETHERNET TO MASTER, Xm



Attach the 360-degree clamp before the split

To terminate the cordset's shield:

1. Expose the cordset's braided shield by cutting the plastic jacket before the point where the cordset splits.



2. Install a 360-degree ground clamp.

Master Network Controllers

The rack mount brackets provided with all Masters are designed to provide adequate grounding through the use of star washers. Always check grounding with a multi-meter by ensuring electrical continuity between the mounting frame and RJ45 connectors on the front.

	When using the rack mount brackets, you *must* connect the frame or electrical cabinet to which the Master is mounted to earth ground.

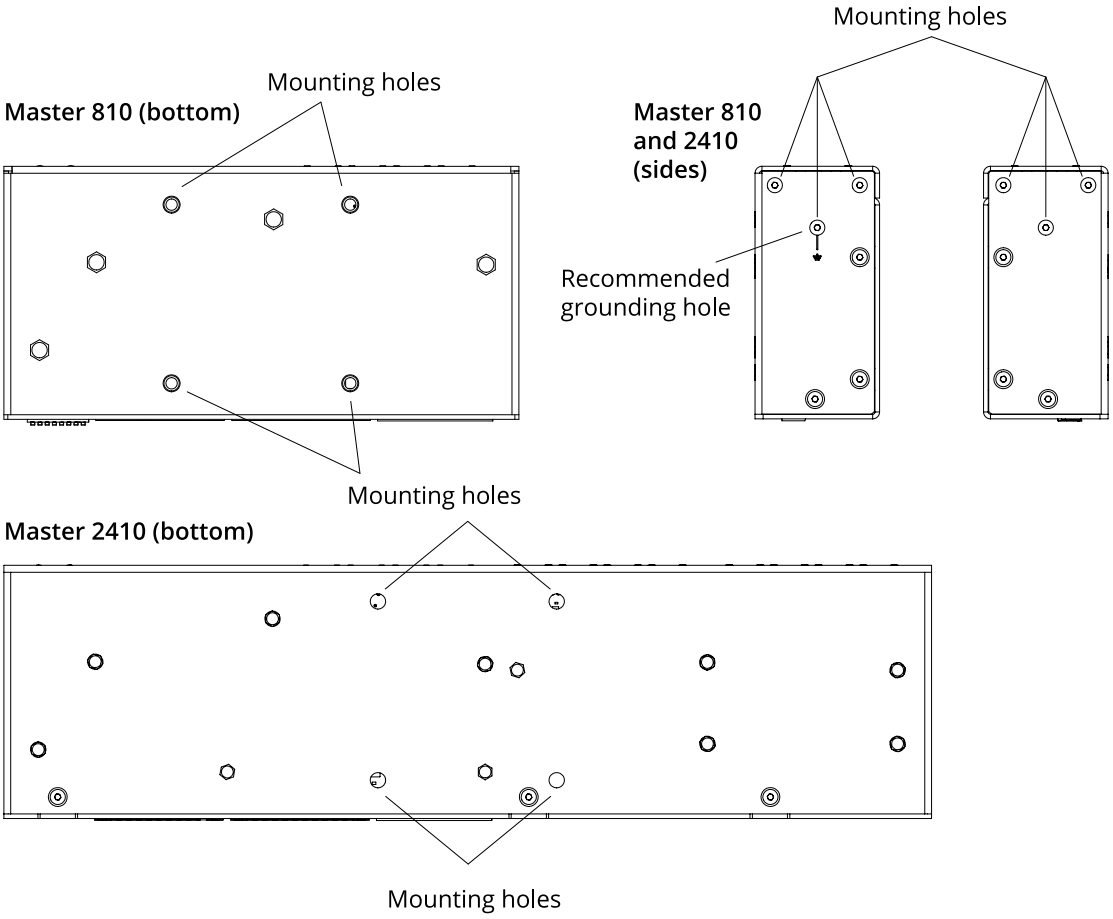
	You *must* check electrical continuity between the mounting frame and RJ45 connectors on the front using a multi-meter.

If you are mounting Master 810 or 2410 using the provided DIN rail mount adapters, you must ground the Master directly; for more information, see *Grounding When Using a DIN Rail (Master 810/2410)* below.

Grounding When Using a DIN Rail (Master 810/2410)

If you are using DIN rail adapters instead of the rack mount brackets, you must ensure that the Master is properly grounded by connecting a ground cable to one of the holes indicated below. The holes on the bottom of the unit accept M4 screws. The holes on the sides of the unit accept M3 screws.

> You can use any of the holes shown below. However, LMI recommends using the holes indicated on the housing by a ground symbol.



An additional ground hole is provided on the rear of Master 810 and 2410 network controllers, indicated by a ground symbol.

Additional Grounding Schemes

Potential differences and noise in a system caused by grounding issues can sometimes cause sensors to reset or otherwise behave erratically. If you experience such issues, see the *Gocator*

Grounding Guide (https://downloads.lmi3d.com/gocator-grounding-guide) in the Download center for additional grounding schemes.

Installing DIN Rail Clips: Master 810 or 2410

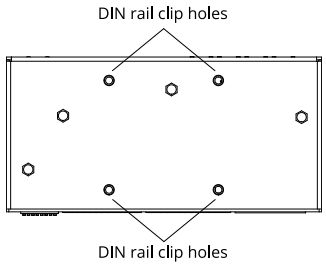
You can mount the Master 810 and 2410 using the included DIN rail mounting clips with M4x8 flat socket cap screws. The following DIN rail clips (DINM12-RC) are included:

Older revisions of Master 810 and 2410 network controllers use a different configuration for the DIN rail clip holes.

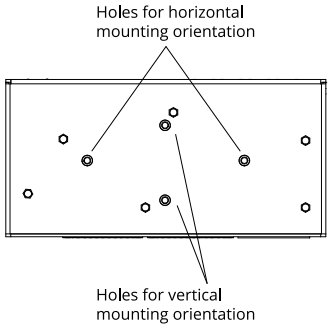
To install the DIN rail clips:

1. Remove the 1U rack mount brackets.
2. Locate the DIN rail mounting holes on the back of the Master (see below).

   Master 810:

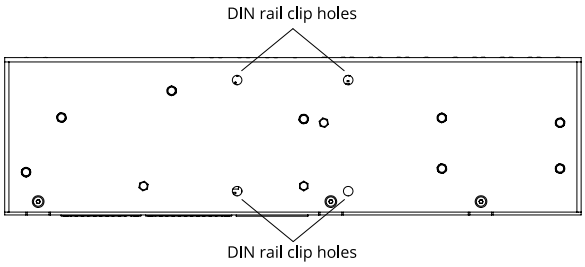


Current revision

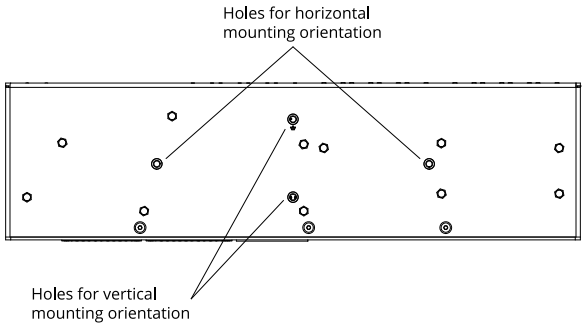


Older revision

   Master 2410:



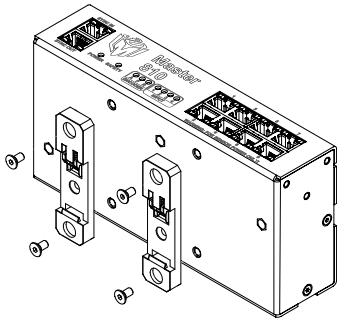
Current revision



Older revision

3. Attach the two DIN rail mount clips to the back of the Master using two M4x8 flat socket cap screws for each one.

   The following illustration shows the installation of clips on a Master 810 (current revision) for horizontal mounting:



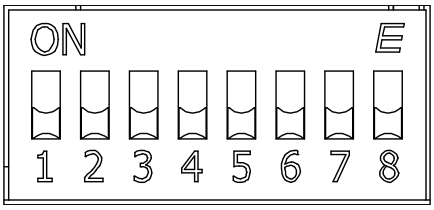
> Ensure that there is enough clearance around the Master for cabling.

Configuring Master 810

If you are using Master 810 with an encoder that runs at a quadrature frequency higher than 300 kHz, you must use the device's divider DIP switches to limit the incoming frequency to 300 kHz.

> Master 810 supports up to a maximum incoming encoder quadrature frequency of 6.5 MHz.

The DIP switches are located on the rear of the device.



Switches 5 to 8 are reserved for future use.

This section describes how to set the DIP switches on Master 810 to do the following:

- Set the divider so that the quadrature frequency of the connected encoder is compatible with the Master.
- Set the debounce period to accommodate faster encoders.

Setting the Divider

To set the divider, you use switches 1 to 3. To determine which divider to use, use the following formula:

Output Quadrature Frequency = Input Quadrature Frequency / Divider

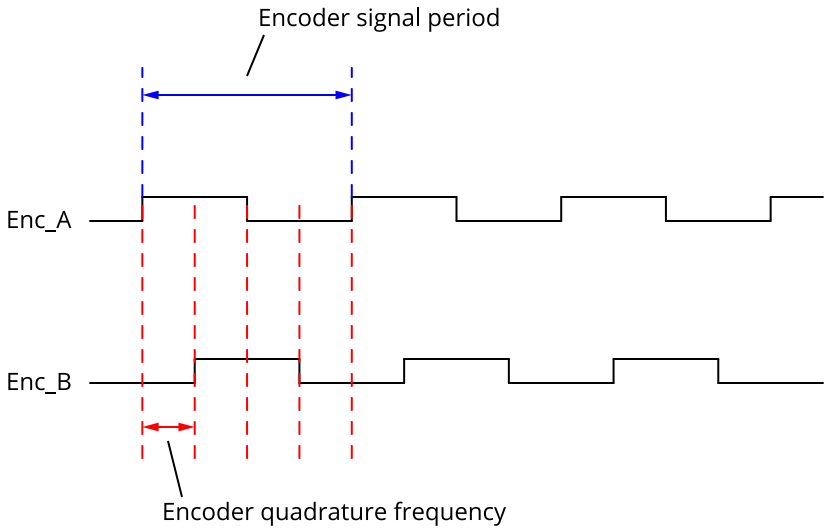
In the formula, use the *quadrature frequency* of the encoder (for more information, see *Encoder Quadrature Frequency* below) and a divider from the following table so that the Output Quadrature Frequency is no more than 300 kHz.

Divider	Switch 1	Switch 2	Switch 3
1	OFF	OFF	OFF
2	ON	OFF	OFF
4	OFF	ON	OFF
8	ON	ON	OFF
16	OFF	OFF	ON
32	ON	OFF	ON
64	OFF	ON	ON
128	ON	ON	ON

The divider works on debounced encoder signals. For more information, see *Setting the Debounce Period* on the next page.

Encoder Quadrature Frequency

Encoder quadrature frequency is defined as illustrated in the following diagram. It is the frequency of encoder ticks. This may also be referred as the native encoder rate.



You must use a quadrature frequency when determining which divider to use (see *Setting the Divider* on the previous page). Consult the datasheet of the encoder you are using to determine its quadrature frequency.

> Some encoders may be specified in terms of encoder signal frequency (or period). In this case, convert the signal frequency to quadrature frequency by multiplying the signal frequency by 4.

Setting the Debounce Period

If the quadrature frequency of the encoder you are using is greater than 3 MHz, you must set the debounce period to “short.” Otherwise, set the debounce period to “long.”

You use switch 4 to set the debounce period.

Debounce period	Switch 4
short debounce	ON
long debounce	OFF

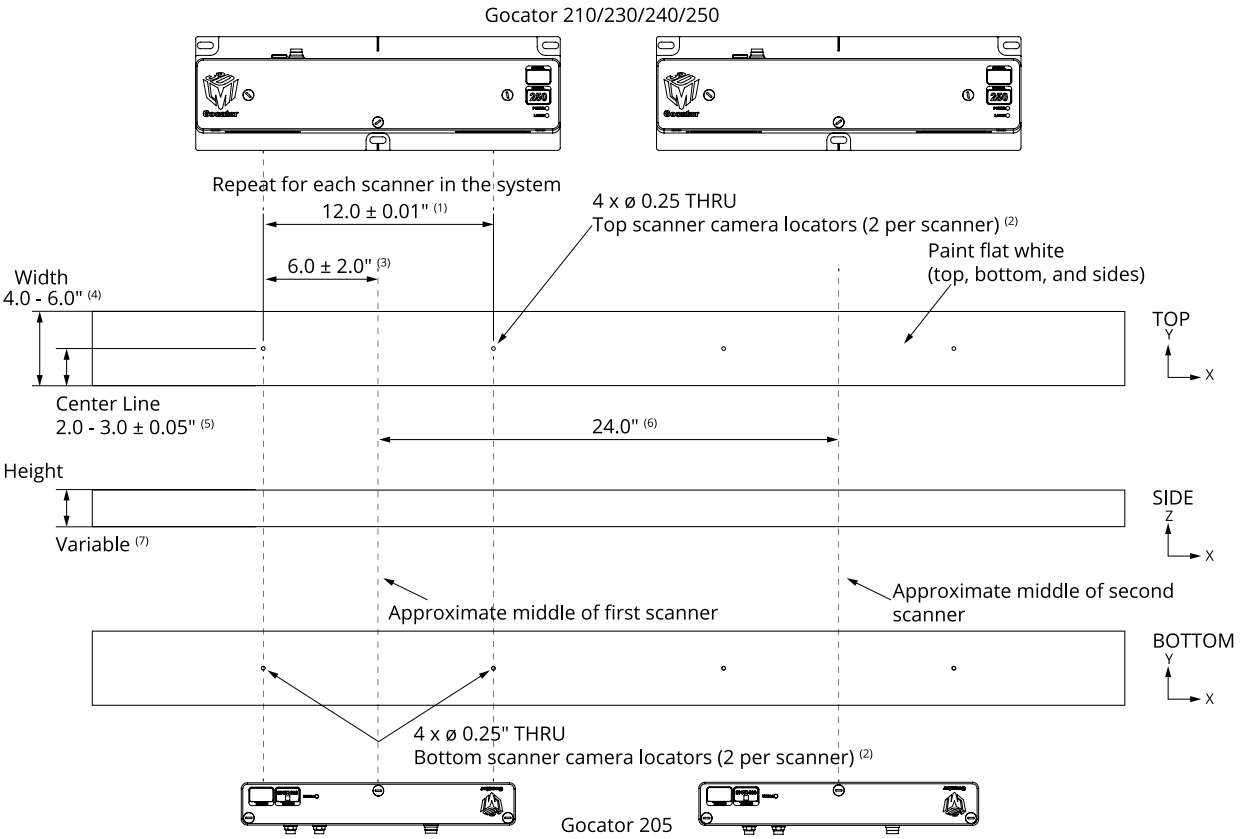
Preparing the Alignment Bar

A special alignment target is required to align a Gocator 200 series system. Alignment locates each sensor with respect to a global coordinate system defined relative to the target. The specifications of the target are provided below.

For systems incorporating the color vision module (Gocator 205), reference holes (referred to as locators) can be used to align the position of each color camera along the length of the system. The target illustrated below is designed for a Gocator 230, 240, or 250 system (with color vision modules), with sensors spaced on 24-inch centers. If your system does not incorporate Gocator 205, you may omit the locators.

The critical requirements when creating the alignment bar are as follows:

- Sensors must capture as little data from the inside of a hole as possible. Either countersink holes from the opposite side of the bar (if no sensors are positioned on the opposite side of the hole in a "Bottom" position), or paint the insides of the holes with a flat black paint. Otherwise, although the alignment should succeed, it will not be as accurate: it may result in unwanted offsets or angles in the transformations
- The alignment bar must be long enough to cover the entire field of view of the system along the X axis. For example, for a two-sensor system, it should be at least 48″ long.
- The alignment bar should be between 4″ - 6″ wide in the direction of travel.
- The suggested thickness of the alignment bar is 2″, but can be any value provided it lies within the sensor FOV.
- The alignment bar should be painted flat white on the top, bottom, and sides.
- Two locators should be aligned with the vision cameras of each Gocator 205, placed 12″ apart.



[1] The vision camera spacing is 12 ± 0.01″.

[2] Locators on the top and bottom are required for every vision camera in the system (two per Gocator 205 sensor). Alignment in the Y position between the top and bottom pairs is important, but they do not need to be at the same X position.

[3] The locators are ideally centered with the vision cameras in the system, which are ± 6″ from the sensor center. However, the locators can be positioned within 2″ of this ideal, as long as they are positioned 12 ± 0.01″ from each other.

[4] The width of the calibration bar can be between 4″ and 6″. The edges of the bar are used by the system to align all profile points and vision data on the Y axis. The width result is dependent on the encoder resolution settings in the system.

[5] The locators must be placed within ± 0.05″ of the center of the alignment bar along the Y axis.

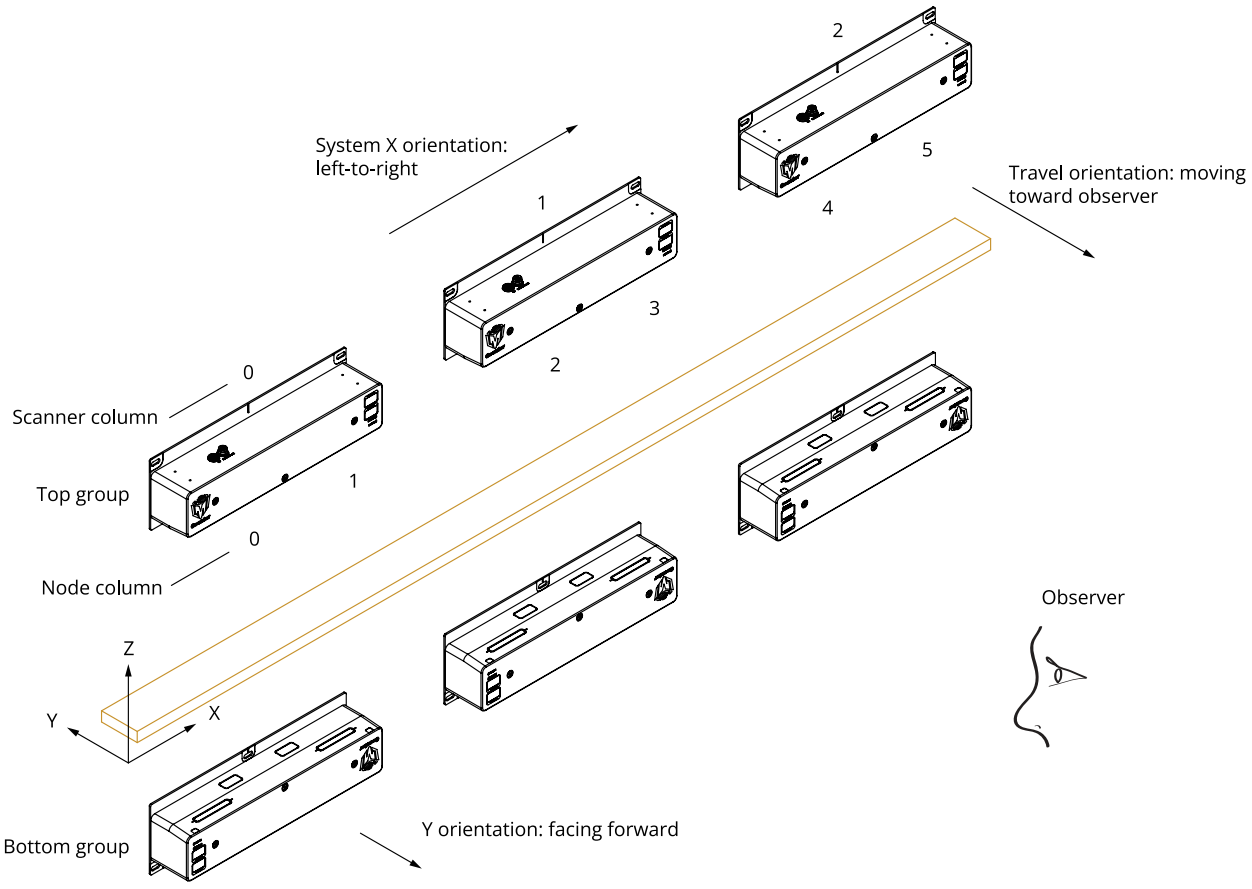
[6] Sensor spacing has been designed at 24″ for continuous vision coverage along the full system. However, sensors can be spaced further apart to allow for chainways and other obstructions, if desired. This can be configured in the system settings in the Settings.xml file.

[7] The thickness of the alignment bar is not critical, provided it is within the sensor's FOV: LMI suggest a value of 2.0″. This thickness must be accurately measured (for example, with CMM or calipers), and should be provided as an input to GoWebScan in the CalibrationTarget.xml file.

Layout and Orientation

System Layout

System layout refers to the orientation of the sensors and the conveyor. The following is a typical layout:



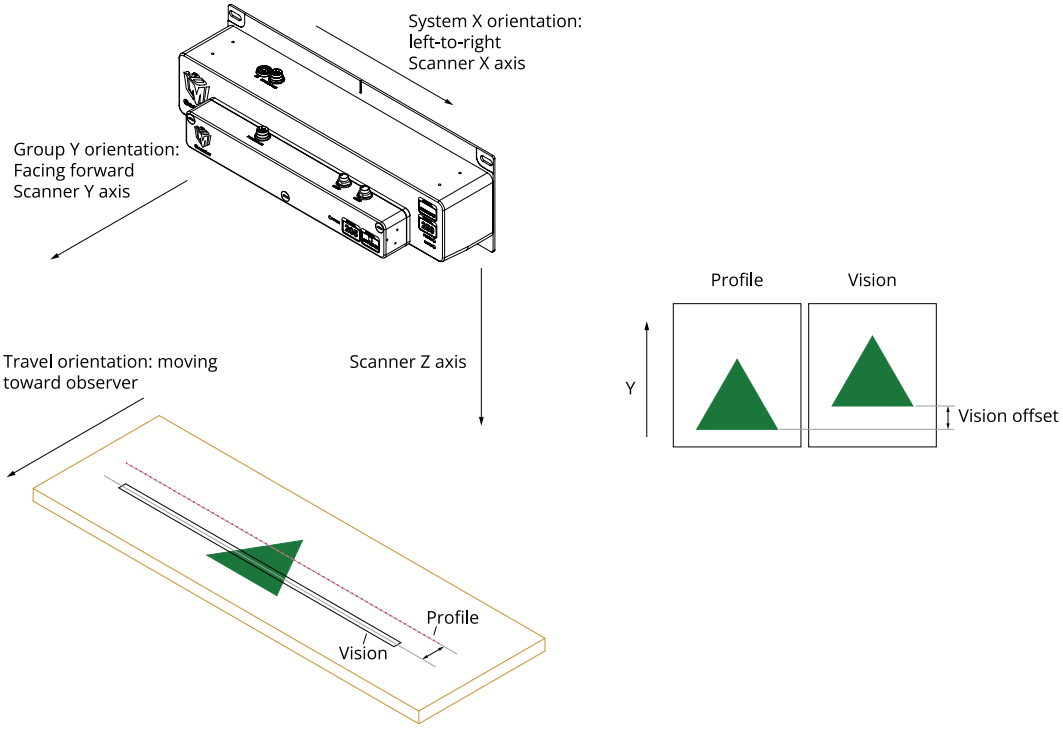
Note the following:

- Orientations are always defined in terms of an observer's perspective. The observer could be situated on either side of the system and define the system from either of those perspectives.
- Each sensor resides in a particular sensor column within its group. In a left-to-right system, the column (and the X axis) increases from the left. In a right-to-left system, the column (and the X axis) increases from the right. The column is much the same as index, but there can be empty columns in the layout (that is, a missing sensor).

- The Y axis increases in the opposite direction of the conveyor's forward motion. If the travel orientation changes, the direction of the system Y axis also changes.

Sensor Orientation

The Gocator G200 sensor shown below is in standard sensor orientation. Standard orientation refers to the relationship of between this sensor's coordinate system and various aspects of system orientation. It includes the optional G205 bolt-on vision module.



In standard orientation:

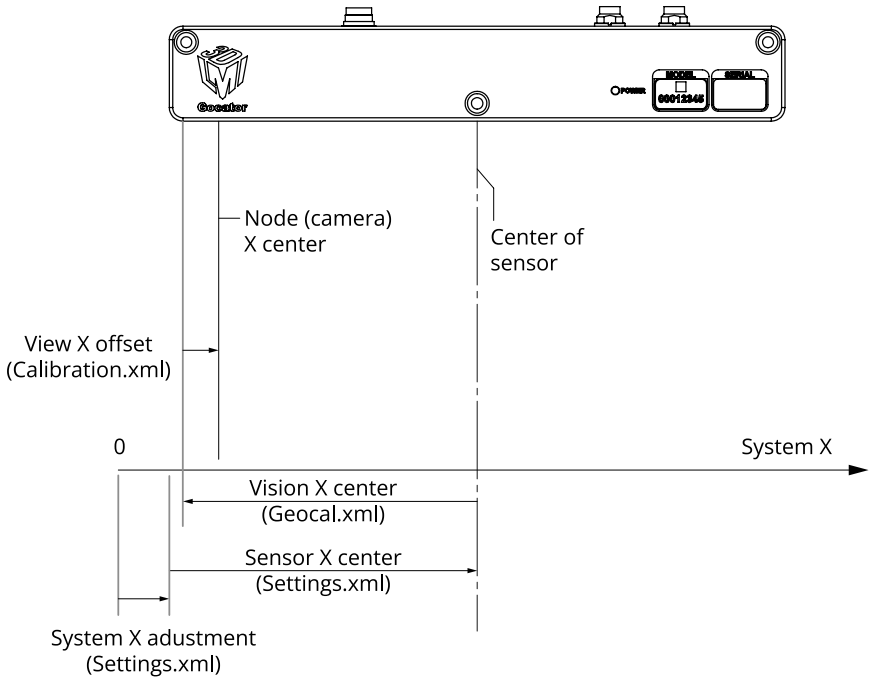
- The sensor X axis is aligned to the system X axis.
- The sensor Y axis is opposite to the system Y axis.
- The sensor Z axis is opposite to the system Z axis.
- Vision lags profile. Therefore vision Y values are offset by a vision offset for coarse alignment (derived from `GoGeoCal.xml`).

Note the following:

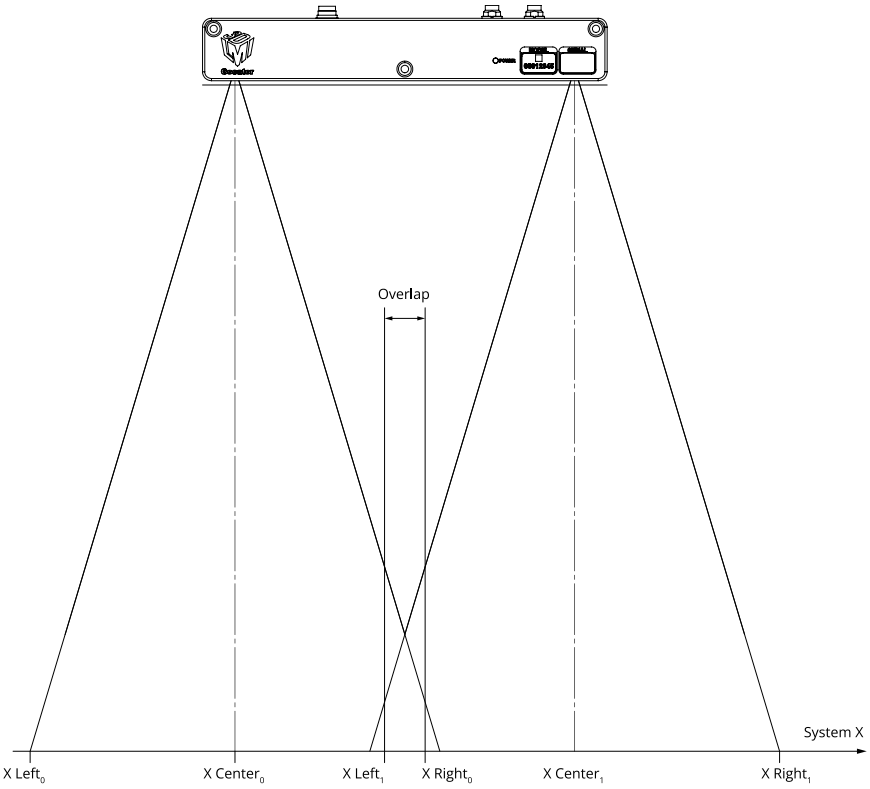
- The `ConfigSensor` class in GoWebScanSDK relates the orientation of a specific sensor to standard orientation using a set of Boolean orientation values. For example, if the `XOrientation` value is 1, the sensor is consistent with standard orientation, with respect to the X axis.
- Vision, profile, and tracheid inputs are reoriented early in GoWebScan processing to align with system coordinates.

X Axis Layout

One of the main functions of the `Config` class in GoWebScanSDK is to perform X axis layout. GoWebScan uses the values in the `Settings.xml`, `Calibration.xml`, and sensor `Geocal.xml` files to map the profile, vision, and tracheid data from each node (camera) into system X coordinates. The following drawing illustrates the center of a vision camera in system X coordinates:



After the X center of each node is determined, the start and end boundaries of each node are determined. These boundaries define the region in system space in which the node can produce data for the final result. If you have set a non-zero overlap amount, the fields of view of adjacent nodes can overlap by this amount. The overlapping regions are later blended together.


Overlap
System X
X Left0
X Center0
X Left1
X Right0
X Center1
X Right1

Modes

GoWebScan supports three system modes: Calibration, Detection, and Web. The example program demonstrates the use of these modes. GoWebScanSDK (via the `GoWebScanConfig` class) lets you switch the system mode without needing to restart GoWebScan. For more information, see *GoWebScanSDK* on page 70.

> The term "station" is used here and elsewhere to loosely refer to the software controlling a group of sensors, via an instance of `GoWebScan`.

Calibration Mode

In Calibration mode, the system assumes that the first scanned object is the system alignment bar. All objects after the first scanned object are ignored. Calibration algorithms are applied to the data and a series of messages are sent to the client.

The example lets you perform a calibration on a group of sensors (represented by an instance of the `GoWebScan` class, or a "station" in chroma+scan terminology). Calibrating a group of sensors sets their coordinate systems to a common X and Y reference.

The example also shows how to align calibrations from *multiple instances* of `GoWebScan` to a common X and Y reference. In a setup using multiple stations, for each station you must first run calibration mode, scanning the calibration bar each time, and then align the resulting calibrations.

Alignment determines the average system reference of all calibrations (in `Calibration.xml > XReference and YReference`), and then sets the adjustment (in `Settings.xml > Calibration > XAdjustment` and `YAdjustment`) to the average system reference minus that station's system reference. This can be done with the `GoWebScanUtils_AlignCalibrations` function.

Note that the example program requires access to the calibration and settings files of the other stations to perform alignment. Once the alignment is complete, the X and Y origins of the board messages from each station will be with respect to a common X and Y reference position and can be used to place the data into a common buffer. In the example program, once calibration has completed, the program switches to detection mode and produces a board image of the calibration bar (with the same data used for calibration).

Detection Mode

In Detection mode, the GoWebScan system monitors incoming sensor data to detect the presence of an object. When an object is detected, the system accumulates data. After a complete object has been captured, data is transmitted to the client.

Web Mode

In Web mode, the system applies calibration transformations and resamples data to the requested output resolutions, but does not attempt to detect discrete objects, producing several smaller arrays containing continuous tiles. This leaves the board detection logic up to the user. Web mode data is still affected by calibration and the selected resolutions.

Network Setup

The following sections provide procedures for client PC and sensor network setup.

> DHCP is not recommended for sensors. If you choose to use DHCP, the DHCP server should try to preserve IP addresses. Ideally, you should use static IP address assignment (by MAC address) to do this.

Client Setup

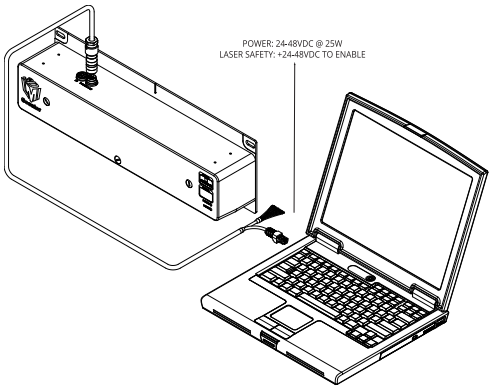
To connect to a sensor from a client PC, you must ensure the client's network card is properly configured.

Sensors are shipped with the following default network configuration:

Setting	Default
DHCP	Disabled
IP Address	192.168.1.10
Subnet Mask	255.255.255.0
Gateway	0.0.0.0

To connect to a sensor for the first time:

1. Connect cables and apply power.
   Sensor cabling is illustrated in *System Overview* on page 29.



2. Change the client PC's network settings.

Windows 7

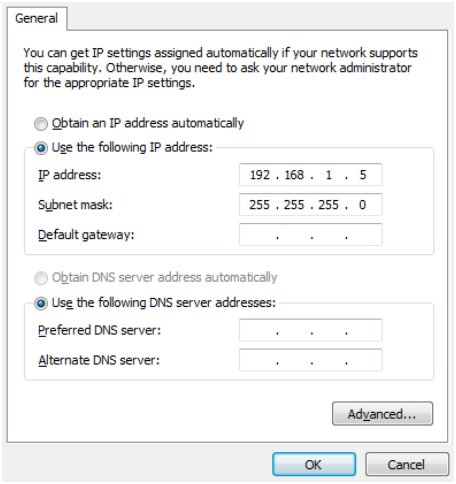
a. Open the Control Panel, select **Network and Sharing Center**, and then click **Change Adapter Settings**.

b. Right-click the network connection you want to modify, and then click **Properties**.

c. On the **Networking** tab, click **Internet Protocol Version 4 (TCP/IPv4)**, and then click **Properties**.

d. Select the **Use the following IP address** option.

e. Enter IP Address "192.168.1.5" and Subnet Mask "255.255.255.0", then click **OK**.

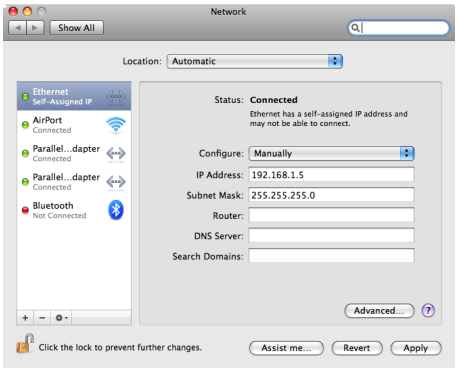


Mac OS X v10.6

a. Open the Network pane in **System Preferences** and select **Ethernet**.

b. Set **Configure** to **Manually**.

c. Enter IP Address "192.168.1.5" and Subnet Mask "255.255.255.0", then click **Apply**.



⚠ See *Troubleshooting* on page 341 if you experience any problems while attempting to establish a connection to the sensor.

Gocator Setup

The Gocator is shipped with a default configuration that will produce 3D data for most targets.

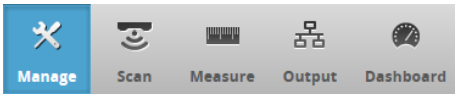
The following describes how to set up a sensor system for operations. After you have completed the setup, you can perform a scan to verify basic sensor operation.

Running a Standalone Sensor System

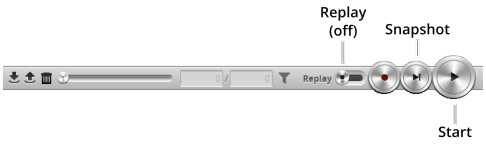
To configure a standalone sensor system:

1. Power up the sensor.
   The power indicator (blue) should turn on immediately.
2. Enter the sensor's IP address (192.168.1.10) in a web browser.
   The sensor interface loads.
   If a password has been set, you will be prompted to provide it and then log in.

3. Go to the **Manage** page.



4. Ensure that Replay mode is off (the slider is set to the left).



> Replay mode disables measurements.

5. Ensure that the Laser Safety Switch is enabled or the Laser Safety input is high.
6. Go to the **Scan** page.
7. Observe the profile in the data viewer
8. Press the **Start** button or the **Snapshot** on the **Toolbar** to start the sensor.
   The **Start** button is used to run sensors continuously.
   The **Snapshot** button is used to trigger the capture of a single frame.
9. Move a target into the sensor's projected light.
   If a target object is within the sensor's measurement range, the data viewer will display scan data, and the sensor's range indicator will illuminate.
   If no scan data is displayed in the data viewer, see *Troubleshooting* on page 341.
10. Press the **Stop** button.
    The projected light should turn off.

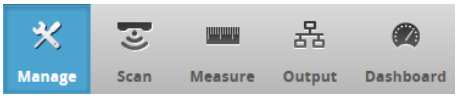
Running a Multi-Sensor System

All sensors are shipped with a default IP address of 192.168.1.10. Ethernet networks require a unique IP address for each device, so you must set up a unique address for each sensor.

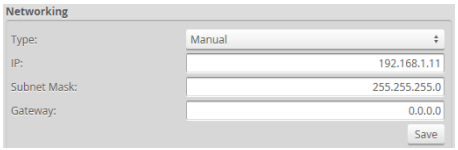
To configure a multi-sensor system:

1. Configure the first sensor in the system as described in *Running a Standalone Sensor System* on the previous page.
2. Unplug the Power/LAN connection of the configured sensor to power it down.
3. Connect the Power/LAN cordset to the new sensor.
   The power LED (blue) of the new sensor should turn on immediately.
4. Enter the new sensor's default IP address (192.168.1.10) in a web browser.
   The web interface loads.

5. Go to the **Manage** page.



6. Modify the IP address in the **Networking** category and click the **Save** button.
   You should increment the last octet of the IP address for each additional sensor you need to use. For example, if the IP address of the first sensor you configured is 192.168.1.10, use 192.168.1.11 for the second sensor; use 192.168.1.12 for the third; etc.
   When you click the **Save** button, you will be prompted to confirm your selection.



7. Power-cycle or reset the sensor.
   After changing a sensor's network configuration, the sensor must be reset or power-cycled before the change will take effect.
8. Repeat steps 3 to 7 for each additional sensor.

Bandwidth Requirements

The Gocator 200 series offers various data outputs and data rates depending on the scanners used and their configuration. You must consider the amount of data and the Ethernet and processing requirements when designing a Gocator 200 multi-scanner system.

Data Rates

The main limitation you must consider is the capacity of the Gigabit Ethernet (1 GigE) link between the scanner and the PC. The recommendations provided here assume that the Ethernet network used is dedicated to the G200 scanners (no other devices are consuming bandwidth).

The following table provides the maximum data rates for the various G200 models:

Scanner	Configuration	Data Rate
Gocator 210	2 kHz profile, 30 points	2 Mbit/s
Gocator 230 / 240[1]	3 kHz profile, 76 points	7.5 Mbit/s
	4 kHz profile, 76 points	10 Mbit/s
Gocator 240[2] / 250	3 kHz profile, 76 points Tracheid data at 1.5 kHz	29. 5 Mbit/s
	4 kHz profile, 76 points Tracheid data at 1.5 kHz	39 Mbit/s
Gocator 205 Full resolution (0.01" x 0.01")	3 kHz	178 Mbit/s
	4 kHz	237 Mbit/s
Gocator 205 Subsampled resolution (0.02" x 0.01")	3 kHz	89 Mbit/s
	4 kHz	119 Mbit/s

[1] Gocator 240 without tracheid upgrade.

[2] Gocator 240 with tracheid upgrade.

[3] Subsampled along the length of the board only.

Example System Testing

The typical maximum scanner capacity on a single 1 GigE link tested by LMI is six Gocator 250 scanners at full resolution with tracheid output, paired with six Gocator 205 vision cameras operating with the resolution subsampled along the length of the board (0.02" x 0.01").

Referring to the table above, this results in a total required bandwidth of roughly 900 Mbit/s, or 90% capacity of a 1 GigE network:

Scanner	Bandwidth	Number of devices	Sub-total
Gocator 250 3 kHz profile 76 points Tracheid data at 1.5 kHz	2 kHz profile, 30 points	6	177 Mbit/s

Scanner	Bandwidth	Number of devices	Sub-total
Gocator 205 Subsampled Resolution (0.02” x 0.01”)	3 kHz profile, 76 points	6	534 Mbit/s
		TOTAL	709 Mbit/s

Processing Considerations

In addition to the Ethernet bandwidth requirements described above, the data must also be processed by the PC receiving the data. For reference, the sample application provided with GoWebScanSDK to process the data and generate sample boards was characterized with the above system. This sample application includes:

- Demosaicing of the Gocator 205 vision data
- Alignment of the data
- Resampling and gap filling
- Board detection

This configuration uses about 90% of the Ethernet capacity and about 60% of an example PC’s processing power. Keep this in mind if you plan to design an application that will perform additional tasks.

Computer Specifications	Intel i5-7500T CPU @ 2.70 GHz 8 GB RAM Windows 10, 64-bit 256GB SSD
CPU usage	58 %
Memory usage	~240 MB

Required Ports

The following table lists the ports used by sensors, the Ethernet-based protocols, the SDK, and the PC-based accelerator. Use this information to determine whether you need to open ports on your network and to understand the traffic that a sensor system will produce over a network.

Ports used

Port	Data Packet Protocol	Description
80	TCP	Server for sensor web interface
502	TCP	Modbus protocol communication
2016	UDP	Internal (protocol-independent)
2017	TCP	Internal (protocol-independent)
2018	TCP	Internal (protocol-independent)

Port	Data Packet Protocol	Description
2019	TCP	Internal (protocol-independent)
2020	UDP	Gocator protocol discovery; SDK; accelerator
3189	TCP	Flash security policy server (only in Gocator 4.7 and earlier releases)
3190	TCP	Gocator protocol control channel; SDK; accelerator
3191	TCP	Emulator web port
3192	TCP	Gocator protocol upgrade channel; SDK; accelerator
3194	TCP	Gocator protocol health channel; SDK; accelerator
3195	TCP	Gocator protocol private data
3196	TCP	Gocator protocol discovery; SDK; accelerator
3197	UDP	Emulator scenario management (RPC)
3220	UDP	Gocator protocol discovery; SDK; accelerator
8190	TCP	ASCII protocol
44818	TCP	EtherNet/IP protocol (standard port)
44818	UDP	EtherNet/IP protocol (standard port)

For more information on how the different protocols use these ports, see the appropriate section in *Protocols* on page 256.

How Gocator Works

The following sections provide an overview of how Gocator acquires and produces data, detects and measures parts, and controls devices such as PLCs. Some of these concepts are important for understanding how you should mount sensors and configure settings such as active area.

3D Acquisition

After a sensor system has been set up and is running, it is ready to start capturing 3D data.

> Gocator sensors are always pre-calibrated to deliver 3D data in engineering units throughout their measurement range.

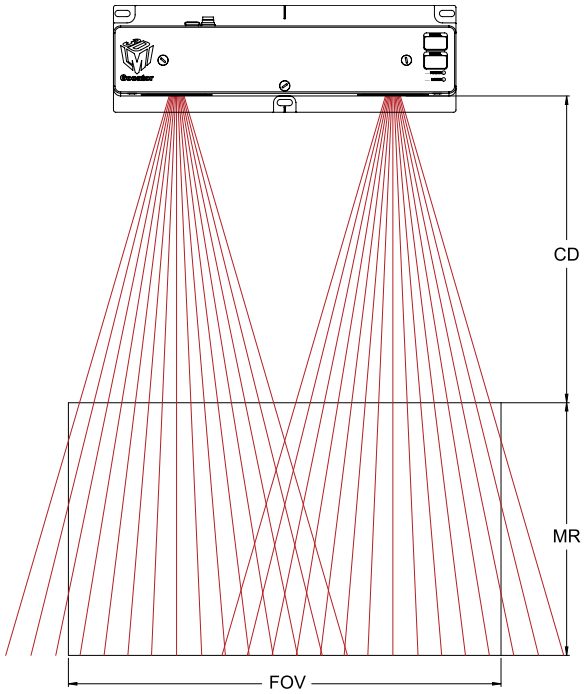
Clearance Distance, Field of View and Measurement Range

Clearance distance (CD), field of view (FOV), and measurement range (MR) are important concepts for understanding the setup of a sensor and for understanding results.

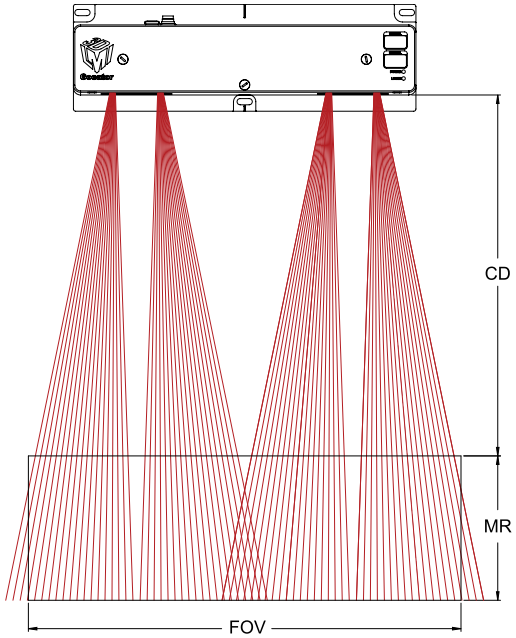
Clearance distance – The minimum distance from the sensor that a target can be scanned and measured. A target closer than this distance will result in invalid data.

Measurement range – The vertical distance, starting at the end of the clearance distance, in which targets can be scanned and measured. Targets beyond the measurement range will result in invalid data.

Field of view – The width on the X axis along the measurement range. At the far end of the measurement range, the field of view is wider, but the X resolution and Z resolution are lower. At the near end, the field of view is narrower, but the X resolution is higher. When resolution is critical, if possible, place the target closer to the near end.



Gocator 210: clearance distance, measurement range, and field of view



Gocator 230/240/250: clearance distance, measurement range, and field of view (with a 10" measurement range)

Resolution and Accuracy

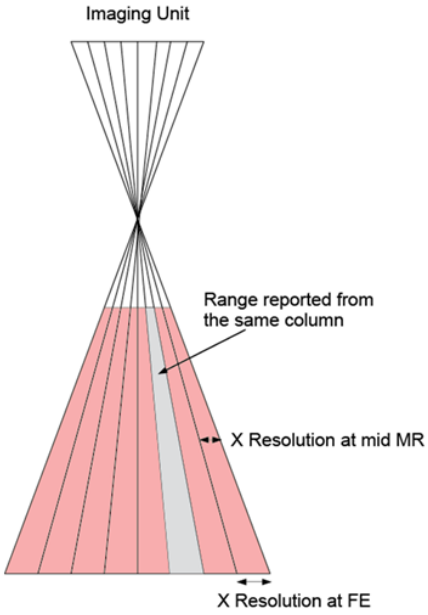
These terms are used in the Gocator datasheets to describe the measurement capabilities of the sensors.

X Resolution

X resolution is the horizontal distance between each measurement point along the laser line. This specification is based on the number of camera columns used to cover the field of view (FOV) at a particular measurement range.

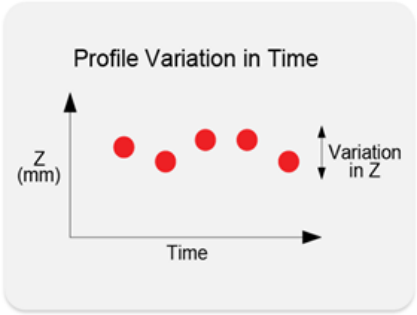
Because the FOV is trapezoidal (shown in red, below), the distance between points is closer at the near range than at the far range. This is reflected in the Gocator data sheet as the two numbers quoted for X resolution.

X Resolution is important for understanding how accurately width on a target can be measured.



Z Resolution

Z Resolution gives an indication of the smallest detectable height difference at each point, or how accurately height on a target can be measured. Variability of height measurements at any given moment, in each individual 3D point, with the target at a fixed position, limits Z resolution. This variability is caused by camera and sensor electronics.



This is reflected in the Gocator datasheets as the two numbers quoted for Z resolution.

Profile Output

Gocator represents a profile as a series of ranges, with each range representing the distance from the origin. Each range contains a height (on the Z axis) and a position (on the X axis) in the sensor's field of view.

Coordinate Systems

Sensor Coordinates

The measurement range (MR) is along the Z axis. The sensor’s field of view (FOV) is along the X axis.

GoSDK and GoWebScanSDK

Gocator 200 series sensors are configured using LMI's Software Development Kits. The following sections provide an overview of these SDKs

Setup and Locations

Class Reference

The full GoSDK class reference is found by accessing the following file:

14400-x.x.x.xx_SOFTWARE_GO_SDK\GO_SDK\doc\GoSdk\Gocator_2x0\GoSdk.html

Examples

Examples showing how to perform various operations are provided, each one targeting a specific area. For Visual Studio, the examples can be found in solution files specific to different versions of Visual Studio. For example, *GoSdk-2017.sln* is for use with Visual Studio 2017. A make file for Linux systems is also provided.

> To compile the examples in Visual Studio, you may be need to retarget the solution to the installed Windows SDK version. You can do this through the **Retarget solution** option in the solution context menu.

To run the GoSDK examples, make sure the required DLLs are copied beside the executable. In most cases only *GoSDK.dll* and *kApi.dll* are required, but with .NET and the accelerator additional DLLs are needed. Please refer to the SDK samples to see which DLLs are required. For GoWebScanSDK samples, also include *GoWebScanSdk.dll* (or *GoWebScanSdkd.dll* in debug configuration). For more information on

Example Project Environment Variable

All GoSDK example projects use the environment variable *GO_SDK_4*. The environment variable should point to the *GO_SDK* directory, for example, *C:\14400-4.0.9.156_SOFTWARE_GO_SDK\GO_SDK*.

Header Files

Header files are referenced with GoSdk as the source directory, for example: #include <GoSdk/GoSdk.h>. The SDK header files also reference files from the *kApi* directory.

Data Types

The following sections describe the types used by the SDK and the kApi library.

Value Types

GoSDK 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	Status, value can be kOK or kERROR
kIpAddress	IP address

Output Types

The following output types are available in the SDK.

Output Data Types

Data Type	Description
GoAlignMsg	Represents a message containing an alignment result.
GoBoundingBoxMatchMsg	Represents a message containing bounding box based part matching results.
GoDataMsg	Represents a base message sourced from the data channel. See *GoDataSet Type* on the next page for more information.
GoEdgeMatchMsg	Represents a message containing edge based part matching results.
GoEllipseMatchMsg	Represents a message containing ellipse based part matching results.
GoExposureCalMsg	Represents a message containing exposure calibration results.
GoMeasurementMsg	Represents a message containing a set of GoMeasurementData objects.
GoProfileIntensityMsg	Represents a data message containing a set of profile intensity arrays.
GoProfileMsg	Represents a data message containing a set of profile arrays.
GoRangeIntensityMsg	Represents a data message containing a set of range intensity data.
GoRangeMsg	Represents a data message containing a set of range data.
GoResampledProfileMsg	Represents a data message containing a set of resampled profile arrays.
GoSectionMsg	Represents a data message containing a set of section arrays.

Data Type	Description
GoSectionIntensityMsg	Represents a data message containing a set of profile intensity arrays.
GoStampMsg	Represents a message containing a set of acquisition stamps.
GoSurfaceIntensityMsg	Represents a data message containing a surface intensity array.
GoSurfaceMsg	Represents a data message containing a surface array.
GoVideoMsg	Represents a data message containing a video image.

Refer to the *GoSdkSamples* sample code for examples of acquiring data using these data types.

See *Setup and Locations* on page 61 for more information on the code samples.

GoDataSet Type

Data are passed to the data handler in a *GoDataSet* object. The *GoDataSet* object is a container that can contain any type of data, including scan data (), measurements, and results from various operations. Data inside the *GoDataSet* object are represented as messages.

The following illustrates the content of a *GoDataSet* object of a mode setup with two measurements.

After receiving the *GoDataSet* object, you should call *GoDestroy* to dispose the *GoDataSet* object. You do not need to dispose objects within the *GoDataSet* object individually.

All objects that are explicitly created by the user or passed via callbacks should be destroyed by using the *GoDestroy* function.

Measurement Values and Decisions

Measurement values and decisions are 32-bit signed values (k32s). See *Value Types* on the previous page for more information on value types.

The following table lists the decisions that can be returned.

Measurement Decisions

Decision	Description
1	The measurement value is between the maximum and minimum decision values. This is a pass decision.
0	The measurement value is outside the maximum and minimum. This is a fail decision.
-1	The measurement is invalid (for example, the target is not within range). Provides the reason for the failure.
-2	The tool containing the measurement is anchored and has received invalid measurement data from one of its anchors. Provides the reason for the failure.

Refer to the *SetupMeasurement* example for details on how to add and configure tools and measurements. Refer to the *ReceiveMeasurement* example for details on how to receive measurement decisions and values.

You should check a decision against <=0 for failure or invalid measurement.

Limiting Flash Memory Write Operations

Several operations and Gocator 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.

Power loss during flash memory write operation will also cause sensors to enter rescue mode.

This topic applies to all Gocator sensors.

SDK Write-Operation Functions

Name	Description
GoSensor_Restore	Restores a backup of sensor files.
GoSensor_RestoreDefaults	Restores factory default settings.
GoSensor_CopyFile	Copies a file within the connected sensor. The flash write operation does not occur if GoSensor_CopyFile function is used to load an existing job file. This is accomplished by specifying "_live" as the destination file name.
GoSensor_DeleteFile	Deletes a file in the connected sensor.
GoSensor_SetDefaultJob	Sets a default job file to be loaded on boot.
GoSensor_UploadFile	Uploads a file to the connected sensor.
GoSensor_Upgrade	Upgrades sensor firmware.
GoSystem_StartAlignment	When alignment is performed with alignment reference set to fixed, flash memory is written immediately after alignment. GoSensor_SetAlignmentReference() is used to configure alignment reference.
GoSensor_SetAddress	Configures a sensor's network address settings.
GoSensor_ChangePassword	Changes the password associated with the specified user account.

System created using the SDK should be designed in a way that parameters are set up to be appropriate for various application scenarios. Parameter changes not listed above will not invoke flash memory write operations when the changes are not saved to a file using the GoSensor_CopyFile function. Fixed alignment should be used as a means to attach previously conducted alignment results to a job file, eliminating the need to perform a new alignment.

GoSDK

The Gocator Software Development Kit (GoSDK) includes open-source software libraries and documentation that can be used to programmatically access and control Gocator sensors. To get the latest version of the *Gocator SDK* package, go to http://lmi3d.com/support, choose your product from the Product Downloads section, and download it from the Download Center.

For information on the ports the SDK uses (for example, in order to ensure ports are not blocked over your network), see *Required Ports* on page 55.

If you switch jobs or make changes to a job using the SDK or a protocol (from a PLC), the switch or changes are not automatically displayed in the web interface: you must refresh the browser to see these.

You can download the Gocator SDK from within the Web interface.

To download the SDK:

1. Go to the **Manage** page and click on the **Support** category
2. Next to **Software Development Kit (SDK)**, click **Download**
3. Choose the location for the SDK on the client computer.

Applications compiled with previous versions of the SDK are compatible with sensor firmware if the major version numbers of the protocols match. For example, an application compiled with version 5.0 of the SDK (which uses protocol version 5.0) will be compatible with a sensor running firmware version 5.1 (which uses protocol version 5.1). However, any new features in firmware version 5.1 would not be available.

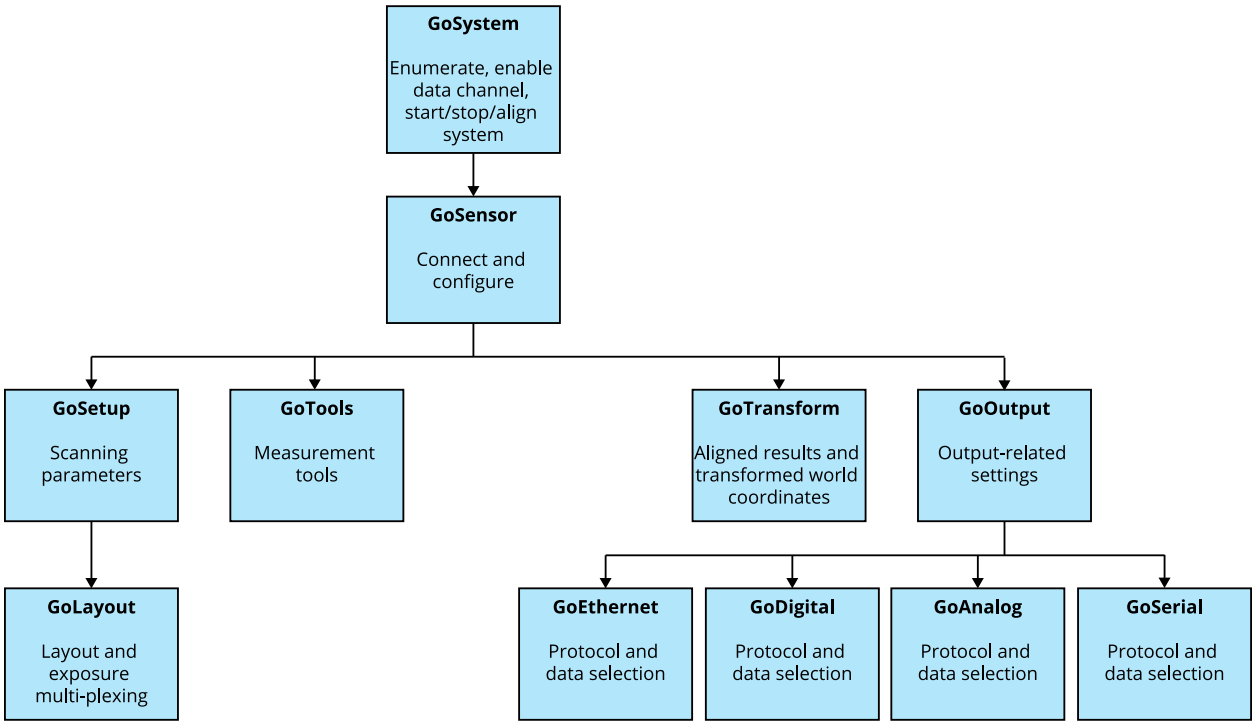
Applications compiled using SDK version 4.x are compatible with sensors running firware 5.x.

Applications compiled using SDK version 3.x are not compatible with sensors running firmware 4.x. In this case, you must rewrite the application with the SDK version corresponding to the sensor firmware in use.

For more information about programming with the SDK, refer to the class reference and sample programs included in the SDK.

Functional Hierarchy of Classes

This section describes the functional hierarchy of the classes in the Gocator SDK ("GoSDK"). In the following diagram, classes higher in the hierarchy often provide resources for classes lower in the hierarchy, and for this reason should be instantiated earlier in a client application.



GoSystem

The *GoSystem* class is the top-level class in the SDK. Multiple sensors can be enabled and connected in one *GoSystem*. Only one *GoSystem* object is required for multi-sensor control.

Refer to the *How To Use The Open Source SDK To Fully Control A Gocator Multi-sensor System* how-to guide in http://lmi3d.com/sites/default/files/APPNOTE_Gocator_4.x_Multi_Sensor_Guide.zip for details on how to control and operate a multi-sensor system using the SDK.

> All objects that are explicitly created by the user or passed via callbacks should be destroyed by using the *GoDestroy* function.

GoSensor

GoSensor represents a physical sensor. If the physical sensor is the Main sensor in a dual-sensor setup, it can be used to configure settings that are common to both sensors.

GoSetup

The *GoSetup* class represents a device's configuration. The class provides functions to get or set all of the settings available in the web interface.

GoSetup is included inside *GoSensor*. It encapsulates scanning parameters, such as exposure, resolution, spacing interval, etc. For parameters that are independently controlled for Main and Buddy sensors, functions accept a role parameter.

GoLayout

The *GoLayout* class represents layout-related sensor configuration.

GoTools

The *GoTools* class is the base class of the measurement tools. The class provides functions for getting and setting names, retrieving measurement counts, etc.

GoTransform

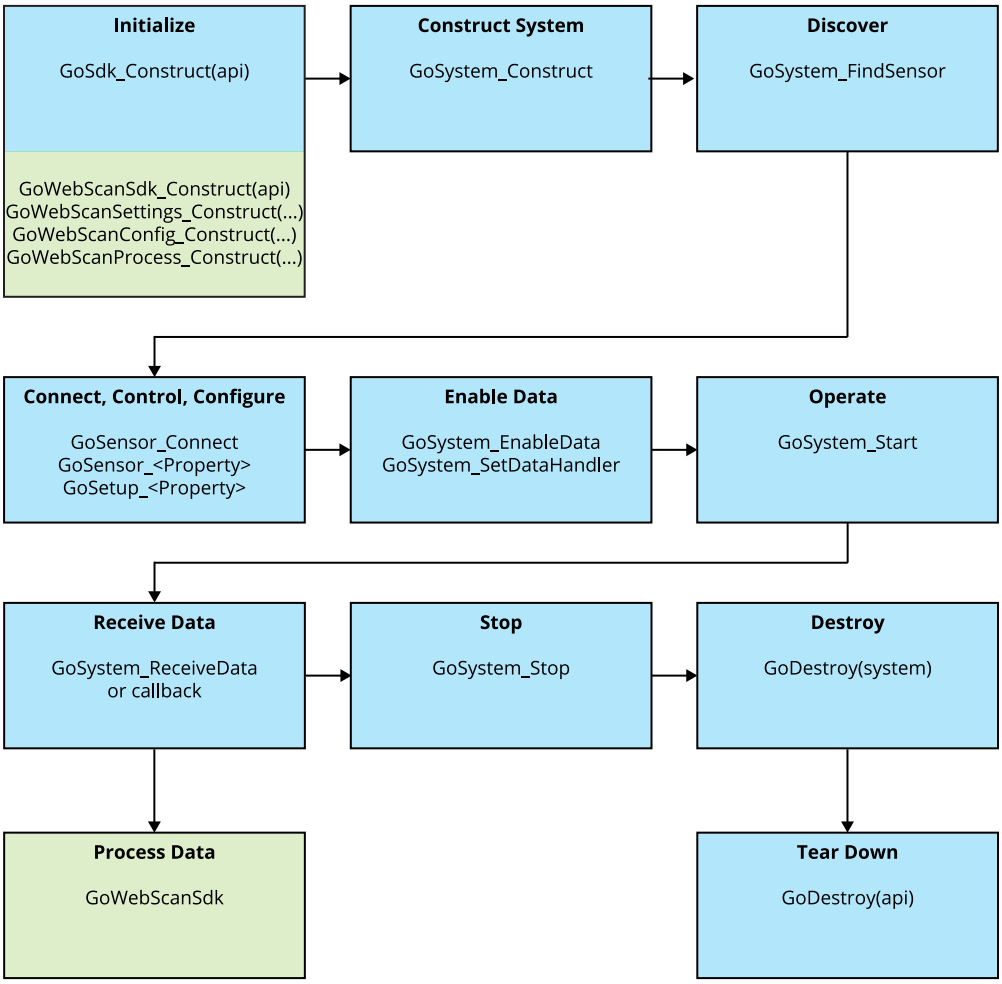
The *GoTransform* class represents a sensor transformation and provides functions to get and set transformation information, as well as encoder-related information.

GoOutput

The *GoOutput* class represents output configuration and provides functions to get the specific types of output. The *GoEthernet* class corresponds to the Ethernet output.

Operation Workflow

Applications created using the SDK typically use the following programming sequence. For more information on the *GoWebScanSdk* functions in the diagram below, see *GoWebScanSDK* on page 70.



See *Setup and Locations* on page 61 for more information on the code samples referenced below.

Sensors must be connected before the system can enable the data channel.

All GoSDK data functions are named *Go<Object>_<Function>*, for example, *GoSensor_Connect*. For property access functions, the convention is *Go<Object>_<Property Name>* for reading the property and *Go<Object>_Set<Property Name>* for writing it, for example, *GoMeasurement_DecisionMax* and *GoMeasurement_SetDecisionMax*, respectively.

Initialize GoSdk API Object

Before the SDK can be used, the *GoSdk* API object must be initialized by calling *GoSdk_Construct(api)*:

```
kAssembly api = kNULL;
if ((status = GoSdk_Construct(&api)) != kOK)
{
    printf("Error: GoSdk_Construct:%d\n", status);
    return;
}
```

When the program finishes, call *GoDestroy(api)* to destroy the API object.

Discover Sensors

Sensors are discovered when *GoSystem* is created, using *GoSystem_Construct*. You can use *GoSystem_SensorCount* and *GoSystem_SensorAt* to iterate all the sensors that are on the network.

GoSystem_SensorCount returns the number of sensors physically in the network.

Alternatively, use *GoSystem_FindSensorById* or *GoSystem_FindSensorByIpAddress* to get the sensor by ID or by IP address.

Refer to the *Discover* example for details on iterating through all sensors. Refer to other examples for details on how to get a sensor handle directly from IP address.

Connect Sensors

Sensors are connected by calling *GoSensor_Connect*. You must first get the sensor object by using *GoSystem_SensorAt*, *GoSystem_FindSensorById*, or *GoSystem_FindSensorByIpAddress*.

Configure Sensors

Some configuration is performed using the *GoSensor* object, such as managing jobs, uploading and downloading files, scheduling outputs, setting alignment reference, etc. Most configuration is however performed through the *GoSetup* object, for example, setting scan mode, exposure, exposure mode, active area, speed, alignment, filtering, subsampling, etc. Surface generation is configured through the *GoSurfaceGeneration* object and part detection settings are configured through the *GoPartDetection* object.

See *Functional Hierarchy of Classes* on page 65 for information on the different objects used for configuring a sensor. Sensors must be connected before they can be configured.

Refer to the *Configure* example for details on how to change settings and to switch, save, or load jobs. Refer to the *BackupRestore* example for details on how to back up and restore settings.

Enable Data Channels

Use *GoSystem_EnableData* to enable the data channels of all connected sensors.

Perform Operations

Operations are started by calling *GoSystem_Start*, *GoSystem_StartAlignment*, and *GoSystem_StartExposureAutoSet*.

Refer to the *StationaryAlignment* and *MovingAlignment* examples for details on how to perform alignment operations. Refer to the *ReceiveRange*, *ReceiveProfile*, and *ReceiveWholePart* examples for details on how to acquire data.

Example: Configuring and starting a sensor with the API

```
#include <GoSdk/GoSdk.h>

void main()
{
    kIpAddress ipAddress;
    GoSystem system = kNULL;
    GoSensor sensor = kNULL;
    GoSetup setup = kNULL;

    //Construct the GoSdk library.
    GoSdk_Construct(&api);

    //Construct a sensor system object.
    GoSystem_Construct(&system, kNULL);

    //Parse IP address into address data structure
    kIpAddress_Parse(&ipAddress, SENSOR_IP);

    //Obtain GoSensor object by sensor IP address
    GoSystem_FindSensorByIpAddress(system, &ipAddress, &sensor)

    //Connect sensor object and enable control channel
    GoSensor_Connect(sensor);

    //Enable data channel
    GoSensor_EnableData(system, kTRUE)

    //[Optional] Setup callback function to receive data asynchronously
    //GoSystem_SetDataHandler(system, onData, &contextPointer)
    //Retrieve setup handle
```

```
    setup = GoSensor_Setup(sensor);

    //Reconfigure system to use time-based triggering.
    GoSetup_SetTriggerSource(setup, GO_TRIGGER_TIME);

    //Send the system a "Start" command.
    GoSystem_Start(system);

    //Data will now be streaming into the application
    //Data can be received and processed asynchronously if a callback function has
been
    //set (recommended)
    //Data can also be received and processed synchronously with the blocking call
    //GoSystem_ReceiveData(system, &dataset, RECEIVE_TIMEOUT)
    //Send the system a "Stop" command.
    GoSystem_Stop(system);

    //Free the system object.
    GoDestroy(system);

    //Free the GoSdk library
    GoDestroy(api);
}
```

GoWebScanSDK

GoWebScanSDK is a library built on GoSDK that lets developers speed up the development and integration of wood applications. The library's source code is provided for users to modify and incorporate into their programs as per their licensing agreement with LMI Technologies.

The SDK provides optimized performance to support full 22-sensor profile and tracheid systems and 10-sensor vision-capable systems (lower due to the higher bandwidth requirements). It also provides coherency protection based on time and distance.

GoWebScanSDK supports all features currently supported by the chroma+scan 3 station services. This includes the following:

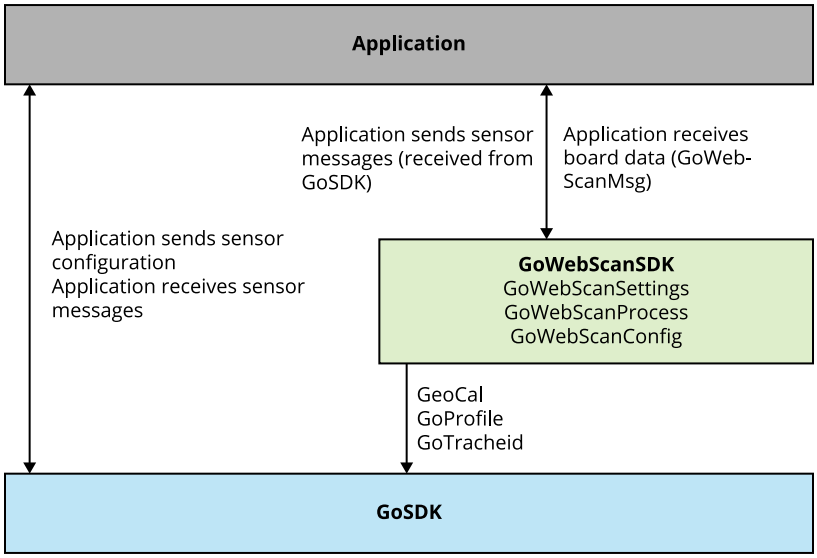
- Resampling of data in X and Y and merging of data across cameras
- Optimized performance to support a full 22 sensor system (for profile + tracheid) and 10 sensor system (for vision—lower due to higher bandwidth)
- Coherency protection based on time and distance
- Gap filling and blending
- Edge spike filtering
- Simple board state machine
- System calibration
- Profile: Z and Y offsets per spot

- Vision: Per-pixel gains, and x-center correction based on detected holes
- Time and encoder mode supported.

To prevent the input data thread from being blocked during the processing and output of board data, GoWebScan lets your client process output data asynchronously, via the `GoWebScanProcess_SetDataHandler` function and a passed callback function.

For full details on GoWebScanSDK, see the API documentation included with the package and the example source code provided with the SDK.

The following diagram shows how a typical application interacts with GoSDK and GoWebScanSDK.



As the diagram shows, in order to develop an application using GoWebScanSDK, you must also be familiar with GoSDK. For more information on GoSDK, including information on how you must incorporate GoWebScanSDK into the operation workflow, see *GoSDK* on page 64.

Main Classes

The following describes the main classes you must instantiate and work with in an application. If you need to customize GoWebScanSDK, refer to the GoWebScanSDK class reference and the GoWebScanSDK software design document.

GoWebScanSettings

Settings class for reading/writing system information to the `Settings.xml` file. Settings include the following, among others:

- Sensor orientation
- Chain orientation
- Sensor node IDs and groups
- Encoder resolution
- Desired profile, vision and tracheid sampling resolution
- Various post-processing algorithm parameters

A template Settings.xml file, populated with values, is provided with the GoSDK package.

For information on Settings.xml, see *Settings.xml File* below

GoWebScanConfig

GoWebScanConfig represents a global configuration class for translating user settings into system parameters and storing these. There are additional classes for lower levels of configuration, but this is the main configuration class you need to work with.

One of the primary functions of the class is to perform X axis layout. This process consists of using the values in Settings.xml, Calibration.xml, and sensor Geocal.xml files to map the profile, vision, and tracheid data from each node into system X coordinates.

After the X center of each node is determined, the start and end boundaries of each node are determined. These boundaries define the region in system space in which the node can make a contribution to the final result. The overlapping regions are later blended together.

GoWebScanProcess

GoWebScanProcess is the main class of GoWebScanSDK. Following initialization, the application passes inbound sensor messages to this class. *GoWebScanProcess* accepts a configuration from the configuration classes and delegates processing responsibilities to various processing task classes.

GoWebScanProcess also handles the initial dispatch to various data sampling tasks. From there, messages flow from task to task, eventually resulting in outbound messages sent back to *GoWebScanProcess* that represent a detected board. Outbound messages are queued by the *GoWebScanProcess* object and can be accepted by the application at any time.

GoWebScanSystemMsg and the Message Classes It Contains

The *GoWebScanSystemMsg* class represents messages returned by *GoWebScanProcess*. It contains a 2D array that contains the following message subtypes:

- *GoWebScanProfileTileMsg*
- *GoWebScanTracheidTileMsg*
- *GoWebScanVisionTileMsg*

Settings.xml File

The Settings.xml file contains the system information as listed in *GoWebScanSettings* on the previous page. If you wish to modify the settings file directly, use the following information. Settings are contained in a GoWebScanSettings element.

GoWebScanSettings Elements

Element	Type	Description
@schemaVersion	32u	SensorGroup settings version (2).
Setup	Section	Contains system-level settings. For a description of the Setup elements, see *Setup Elements* on the next page.
Sampling	Section	Contains sampling settings. For a description of the Sampling elements, see *Sampling Elements* on page 74.
Detection	Section	Contains board detection settings. For a description of the Detection

Element	Type	Description
		elements, see *Detection Elements* on page 76.
Calibration	Section	Contains system calibration settings. For a description of the Calibration elements, see *Calibration Elements* on page 77.
Members	Section	Lists the groups that are defined for this system. For a description of the Members elements, see *SensorGroup Elements* on page 77.

Setup Elements

Element	Type	Description
@schemaVersion	32u	SensorGroup settings version (1).
EncoderResolution	64f	Distance (mils) per encoder pulse. This value will be positive if the encoder count increases when the board travels in the forward direction.
SimulatedSpeed	64f	Optional field that can be used to simulate forward motion (mils/second). When this property is present in the settings file, the program will override the encoder values in data messages with simulated values that are based on time. Note that this setting affects messages only after they are passed to GoWebScanProcess.
TravelOrientation	32s	Specifies the direction of conveyor movement: 0 – Moving Away 1 – Moving Toward. Implies that objects on the conveyor are moving toward the observer when the system is viewed from the front.
TopYOrientation	32s	Specifies the Y orientation of the top-mounted sensors. (The sensor logo and labels are visible when sensors are facing toward.) 0 – Facing Away 1 – Facing Toward
BottomYOrientation	32s	Specifies the Y orientation of the top-mounted sensors. (The sensor logo and labels are visible when sensors are facing toward.) 0 – Facing Away 1 – Facing Toward
XOrientation	32s	Specifies the direction in which the X axis (length) increases: 0 – Right-to-left 1 – Left-to-right. In a left-to-right system (as viewed from the front), the zero reference is on the left and sensor indices (defined by the sensor "Name" property) increase left-to-right. In a right-to-left system, the zero reference is on the right and sensor indices increase right-to-left.
UseXCentres	Bool	Specifies whether the client should use the per-sensor X center values provided in the settings file, or calculate them from sensor indices (for example, 0:12000, 1:36000, 2:60000, and so on). In the latter case, it is

Element	Type	Description
		assumed that each sensor is mounted at a nominal 24” spacing. If this setting is enabled, you must set a value in the XCentre setting of each sensor.
TracheidEnabled	Bool	Specifies whether tracheid detection is enabled in the system.
VisionEnabled	Bool	Specifies whether vision is enabled in the system.
ProfileFrameRate	32u	Represents the frame rate in Hz that will be applied to all sensors in the system for profile data. Gocator 230, 240, and 250 sensors have a maximum frame rate of 4 kHz with an 8" measurement range; with a 10" measurement range, the maximum is 3 kHz. Gocator 210 sensors have a maximum frame rate of 2 kHz. Frame rates for Gocator 205 are calculated internally. This setting overrides the frame rate set on individual sensors via the web interface, GoSDK, or the Write File Gocator protocol command.
ProfileExposure	32u	Specifies the length of exposure, in milliseconds, to use for each profile. This setting overrides the exposure set on individual sensors via the web interface, GoSDK, or the Write File Gocator protocol command.
VisionExposureTop VisionExposureBottom	32u	Specifies the length of exposure, in milliseconds, to use for each vision capture (top and bottom, respectively). This setting overrides the exposure set on individual sensors via the web interface, GoSDK, or the Write File Gocator protocol command.
TracheidExposure	32u	Specifies the length of exposure, in milliseconds, to use for each vision capture. This setting overrides the exposure set on individual sensors via the web interface, GoSDK, or the Write File Gocator protocol command.
SystemAngle	64f	If you are using an angled layout to perform off-axis measurement, use this to set the mounting angle in degrees. For more information, see *Off-Axis Measurement* on page 80.
ActiveAreaZ	64f	The active area Z lower boundary, in mils.
ActiveAreaHeight	64f	The active area Z boundary height, in mils.

Currently, tracheid thresholds must be set on individual sensors via the web interface, GoSDK, or the Write File Gocator protocol command.

Currently, settings related to laser sleep and wake must be set on individual sensors via the web interface, GoSDK, or the Write File Gocator protocol command.

Sampling Elements

Element	Type	Description
ProfileXResolution	32s	The X resolution (mils per profile value, along board length) of the

Element	Type	Description
		profile data that is sent. The following values are supported: 200, 250, 500, and 1000.
ProfileYResolution	32s	The Y resolution (mils per profile value, across board width) of the profile data that is sent. The following values are supported: 5, 10, 20, 25, 40, 50, 100, and 200.
VisionXResolution	32s	The X resolution (mils per pixel, along board length) of the image data that is sent to the client. The following values are supported: 10, 20, 40, and 100.
VisionYResolution	32s	The Y resolution (mils per pixel, across board width) of the image data that is sent to the client. The following values are supported: 10, 20, 40, and 100.
VisionXSubsampling	32u	The subsampling applied to the image data sent to the client.
TracheidXResolution	32s	The X resolution (mils per tracheid value, along board length) of the tracheid data that is sent to the client. The following values are supported: 200, 250, 500, and 1000.
TracheidYResolution	32s	The Y resolution (mils per tracheid value, across board width) of the tracheid data that is sent to the client. The following values are supported: 20, 25, 40, 50, 100, and 200.
TileWidth	32s	The width of the data (mils) included in each profile, vision, or tracheid output tile. The following values are supported: 200, 400, 1000, and 2000.
ProfileInterpolation	32s	Determines the method used to resample profile points along the X axis: 0 – Nearest neighbor 1 – Linear interpolation
GapFill	32s	Maximum distance to fill gaps in data along the X axis (mils). For all data types (profile, tracheid, vision), this value determines the maximum gap between cameras that will be filled by connecting the data from adjacent cameras. Gaps can occur when target objects are close to the sensors, or when sensors are mounted with space between each device. For profile and tracheid cameras, this value also determines the maximum X distance for interpolation between two valid laser spots when resampling data within a single camera.
OverlapBlend	32s	Maximum amount of redundant camera information that can be blended together at the X boundary between two cameras (mils). Blending occurs only when the fields of view between adjacent cameras have non-zero overlap.
ObstructionFilter	Bool	Specifies whether to remove profile points contained within obstruction regions. 0 – Include all profile points in output data

Element	Type	Description
		1 – Remove profile points contained within obstruction regions from output data

Detection Elements

Element	Type	Description
TriggerStyle	32s	Determines the type of object detection trigger. 0 – Global 1 – Local An object is detected when the amount of visible target material (measured along the x-axis) exceeds the TriggerLength threshold. With a local trigger, the visible target material must be connected (contiguous). With a global trigger, the total amount of visible target material in the system field of view is considered.
TriggerLength	32s	The length (mils) of material used to trigger object on/off events, not including material inside obstruction regions.
EdgeMargin	32s	The size of the margins (mils) that will be added to the leading and trailing edge of each detected object.
MaximumWidth	32s	The maximum width of a detection output (mils), including edge margins. When objects exceed this width, they will be segmented into multiple output messages.
EdgeFilter	Bool	Specifies how edge measurement anomalies should be handled. 0 – Leave them unaltered 1 – Correct edge measurement anomalies
BackgroundFilter	Bool	Specifies what happens to when there is no corresponding profile data to tracheid and vision data. 0 – Include all data in detection output 1 – Remove tracheid and vision data when there is no corresponding profile data
SideExtension	Bool	When a laser spot falls on the side edge of a board, the sensor will report a range for the spot that is different from the range of the adjacent spot. Data interpolation between the two spots will be distorted, causing the board side edge to have a slope instead of a straight edge. If SideExtension is enabled, the side edge spots will report the same range as the adjacent spot, which will produce a straight edge. 0 – Disabled 1 – Enabled

Calibration does not work if "Simulated Speed" is used, unlike other Gocator families. The console example will not notify you if this option is used.

Calibration Elements

Element	Type	Description
DetectLocators	Bool	Should the server attempt to detect calibration reference holes (1), or assume that sensors are mounted at their nominal locations along the X axis (0)? This setting applies to vision-enabled systems only (systems with Gocator 205 sensors).
UseVisionIntensity	Bool	Specifies whether the VisionIntensity settings should be used (defined below). 0 – Disabled 1 – Enabled
VisionIntensityR	8u	Specifies the desired red intensity level for the calibration target (1-255).
VisionIntensityG	8u	Specifies the desired green intensity level for the calibration target (1-255).
VisionIntensityB	8u	Specifies the desired blue intensity level for the calibration target (1-255).
XAdjustment	32s	Offset applied to the X coordinates of all data (profile, vision, and tracheid) in Web or Detection mode. This setting is used for X alignment between multiple GoWebScan processes, and can be written by the user application at the end of calibration.
YAdjustment	32s	Offset applied to the Y coordinates of all data (profile, vision, tracheid) in Web or Detection mode. This setting is used for X alignment between multiple GoWebScan processes, and can be written by the user application at the end of calibration.
LeadYMargin TrailYMargin	32s	The margin (in mils) collected in the Y direction during calibration before and after the calibration bar, respectively. The recommended value is 4000.
UseObstructions	Bool	Enable this field to manually remove regions from the calibration input data. These regions are specified via sensor obstructions and typically represent conveyor hardware regions. When this field is disabled, the calibration will automatically detect conveyor hardware regions to remove from the input data.
DataCollectionDensityLevel	64f	Controls the Y axis density of data collected during system calibration. Typically a level of 8 is sufficient for most systems. On older single-core PCs, the density should be lowered to a level of 4.

SensorGroup Elements

Element	Type	Description
Name	Char array	Indicates whether the group of sensors is on the top or bottom. Value must be either "Top" or "Bottom".
Sections	Section	Contains a list of Section elements (described below). Sections are used to define Detection mode outputs that can be routed to separate destinations.

Element	Type	Description
Members	Sensor	Lists the individual sensors that are contained in this group (see below).

Section Elements

Element	Type	Description
Id	32s	A user-defined identifier for this section.
Type	32s	The type of the section. 0 – Profile 1 – Vision 2 – Tracheid
X0	32s	The start of the section along the X axis (length), in system coordinates (mils).
X1	32s	The end of the section along the x-axis (length), in system coordinates (mils). Should be greater than X0.

Sensor Elements

Element	Type	Description
Index	32s	The index of the sensor along the X axis.
ProfileSerialNumber	32s	The manufacturing serial number of the sensor, which can be seen on the sensor housing.
VisionSerialNumber	32s	The manufacturing serial number of the sensor, which can be seen on the sensor housing.
Enabled	Bool	Enables or disables the sensor.
XCentre	32s	Defines the mounting location of the sensor, in system coordinates (mils). This setting has no effect unless the Setup/UseXCentres setting is also enabled.
Obstructions	Obstruction	A list of obstructed zones that will be ignored by the system. Obstructions must be defined for any objects that regularly appear within the field of view of the sensors, such as chain runners. If these zones are not defined, the system will not operate correctly.

Obstruction Elements

Element	Type	Description
X0	32s	The start of the obstruction zone along the X axis (length), in sensor coordinates (mils).
X1	32s	The end of the obstruction zone along the X axis (length), in sensor coordinates (mils).
Z0	32s	The start of the obstruction zone along the Z axis (height), in sensor coordinates (mils).
Z1	32s	The end of the obstruction zone along the Z axis (height), in sensor coordinates (mils).

CalibrationTarget.xml File

Describes the calibration target. If this file is not present, the SDK assumes a default width of 4000 mils and a default height of 2000 mils. For more information on the calibration target, see *Preparing the Alignment Bar* on page 42.

CalibrationTarget Elements

Element	Type	Description
Width		The width of the calibration target, in mils.
Height		The height of the calibration target, in mils.

Calibration.xml File

System calibration output is stored in the Calibration.xml file. The system calibration file may not exist if system calibration has not been successfully completed. Deleting this file clears the current system calibration.

GoWebScanCalibration Elements

Element	Type	Description
YReference	64s	Calibration Y reference (mils).
XReference	64s	Calibration X reference (mils).
Views	Section	A view in the calibration file refers to a profile/vision/tracheid camera set. For a description of this node's elements, see .

View Elements

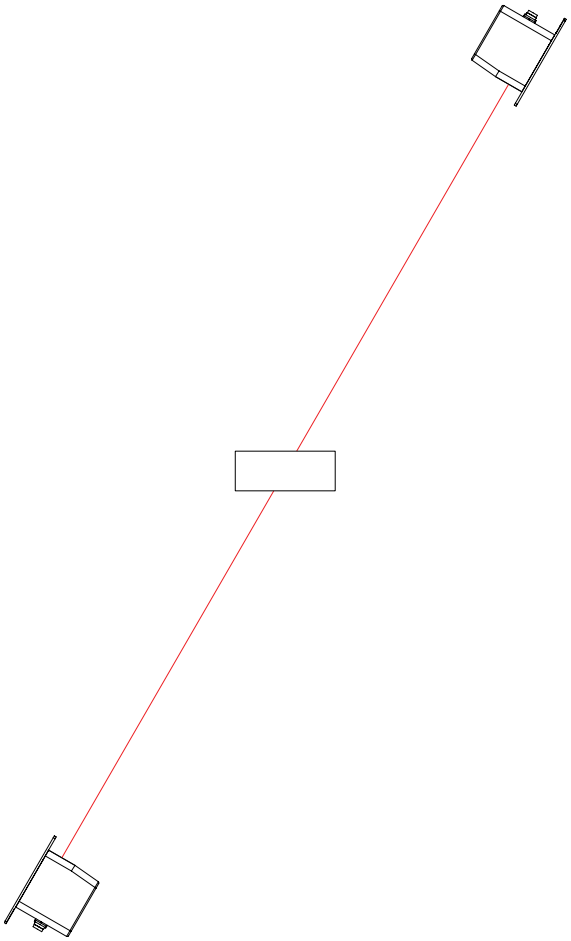
Element	Type	Description
SensorId	16s	Serial number of the sensor that contains this view.
BankId	32s	Index of view within sensor's reference frame (0 or 1).
Plane		Group identifier: Top (0) or Bottom (1).
Column	32s	View location within plane/group.
XOffset	64f	X location correction for this view, in mils.
Cfa	32s	The color filter array type. One of the following: 0: None 1: Bayer BGGR 2: Bayer GBRG 3: Bayer RGGB 4: Bayer GRBG
ProfileYOffsets	32s	An array of profile Y offsets, respectively, as signed 16-bit hex values. Each value is expressed with 4 characters (for example, "FFAA"), and represents alignment offsets in Y per spot in mils. There is one array element per laser dot.
ProfileZOffsets	16s	An array of profile Z offsets, respectively, as signed 16-bit hex values. Each value is expressed with 4 characters (for example, "FFAA"), and represents alignment offsets in Z per spot in mils.

Element	Type	Description
		There is one array element per laser dot.
VisionYOffsets	16s	Array of vision Y offsets as signed 16-bit hex values. Each value is expressed with 4 characters (for example, "FFAA"), and represents a Y correction in mils. The number of array elements is variable (minimum 2 elements). The actual Y offsets will be determined by resampling these Y offsets at the required resolution.
VisionRedGains VisionGreenGains VisionBlueGains	64s	A list of Row elements, where each row is an array of vision red, green, or blue gains as unsigned 8-bit hex values. Each value is expressed with 2 characters (for example, "FA"), and represents a log10 gain value (0x00=0.1, 0xFF=10.00). The number of row elements and the width of each row are variable (minimum 2 rows, 2 elements per row). Actual gains will be determined by resampling these gains at the required camera resolution. This setting applies to vision-enabled systems only.

Off-Axis Measurement

Setup

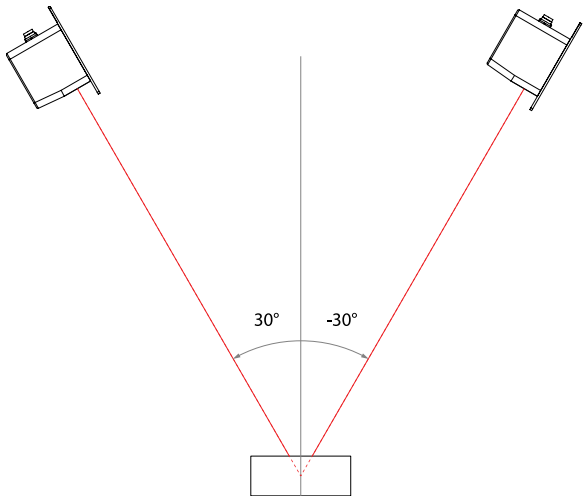
GoWebScan 5.3.32.4 and later supports *off-axis measurement*, that is mounting sensors at an angle to observe both the face and the edge of targets. Depending on your application, the ideal mounting of the sensors could be different from these drawings.



The mounting angle is expressed in the `SystemAngle` tag (in degrees) in the `Settings.xml` file:

```
<?xml version="1.0" encoding="UTF-8"?>
<GoWebScanSettings schemaVersion="2">
<Setup>
....
<SystemAngle>-30</SystemAngle>
</Setup>
...
```

When viewing the system from the front, if the top sensors are rotated to a position that is closer to the viewer, then the sign of the angle is positive. In the illustration below, the observer is assumed to be viewing the system from the left-hand side.



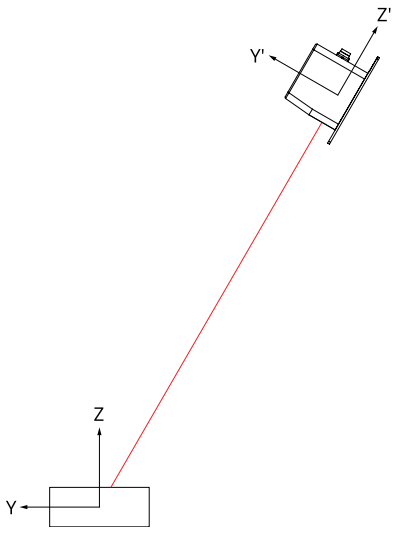
Use a value of zero degrees to specify an *on-axis measurement* system.

Measurement

Gocator supports off-axis measurement in Calibration and Detection modes.

The rules and behaviour of Calibration Mode are the same as for on-axis measurement. Internally, GoWebScan will use the SystemAngle parameter to account for geometric changes related to off-axis setup. The requirements for the calibration bar and the calibration procedure are the same.

The behaviour of off-axis systems in Detection Mode is similar to the behaviour of on-axis systems. However, the interpretation of the transmitted data changes slightly to account for the off-axis geometry.



As with on-axis systems, the Z axis represents height above the object, and the Y axis represents the dimension of motion.

The Z′ axis is the dimension of profile measurement, determined by the angle of the laser plane. The Y′ axis is orthogonal to the Z′ axis.

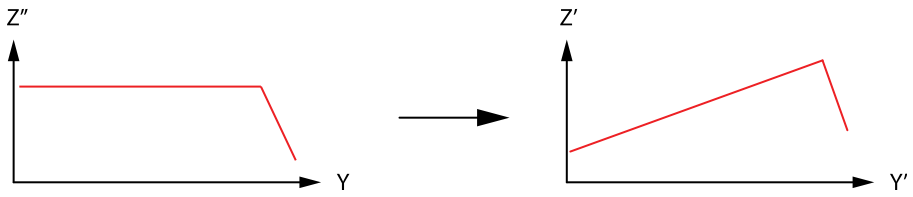
Profile measurements in Detection mode are transmitted in Y-Z″ coordinates. The Y axis is the dimension of motion (as above), whereas the Z″ axis represents instantaneous profile measurements, that is, measurements along the Z′ axis that are not adjusted for forward motion as the target travels.

The following transformations can be used to map Y-Z″ values to Y′-Z′ values:

```
Z' = Z" + Y * sin(θ)

Y' = Y * cos(θ)
```

In practical terms, this means adding a ramp function (Y * sin(θ)) to received Z″ values and reinterpreting the Y resolution of the resulting array (Y resolution becomes Y′ resolution).



Increasing Frame Rate and Transmit Limit

C revision Gocator 230, 240, and 250 sensors support increasing the frame rate to 4 kHz. The increased frame rate results in an increased data output rate, which means you must also increase the sensor transmit limits. Note that if X subsampling is disabled at 3 kHz, you will also have to increase the transmit limit.

> C revision Gocator 230, 240, and 250 sensors support an increased frame rate of 4 kHz *OR* an increased measurement range of 10" (instead of 8"). These sensors only support one or the other.

Each sensor has its own transmit limit, a parameter which caps the sensor's output data rate in order to prevent burst transmissions. Transmit limit is expressed as a percentage of the individual sensor's link capacity, which is 1 Gigabit per second.

Use the `GoWebScanUtils_CalculateProfileTransmitLimit` and `GoWebScanUtils_CalculateVisionTransmitLimit` functions available in the GoWebScan SDK to calculate the desired transmit limit of profile and vision sensors. You must call these functions only after other settings have been applied to the sensor, because the calculation depends on the setup of the sensor.

After you have calculated the transmit limits use the `GoSetup_SetTransmitLimit` function available in GoSDK to set the transmit limit. You can query the current transmit limit with `GoSetup_TransmitLimit` functions available in GoSDK set and get the transmit limit, respectively.

The following code shows how to do this with a profile sensor. The example program includes this and additionally shows how to set the transmit limit of a vision sensor.

```
if (!kSuccess(GoSystem_FindSensorById(context->system, profileId, &profileSensor)))
{
    printf("Error: SN %d not found\n", profileId);
    return kERROR;
}

setup = GoSensor_Setup(profileSensor);

// Apply any other sensor settings here.

kCheck(GoWebScanUtils_CalculateProfileTransmitLimit(context->settings,
profileSensor, &item.profileTransmitLimit));
kCheck(GoSetup_SetTransmitLimit(setup, item.profileTransmitLimit));
```

Increasing Measurement Range

C revision Gocator 230, 240, and 250 sensors support an increased measurement range of 10" compared to the default 8".

> C revision Gocator 230, 240, and 250 sensors support an increased frame rate of 4 kHz *OR* an increased measurement range of 10" (instead of 8"). These sensors only support one or the other.

By default, this range is 8 inches. To increase the measurement range to the full 10 inches, use the active area functions `GoSetup_SetActiveAreaZ` and `GoSetup_SetActiveAreaHeight` available in GoSDK. Note that the values are in millimeters.

```
if (!kSuccess(GoSystem_FindSensorById(context->system, profileId, &profileSensor)))
{
    printf("Error: SN %d not found\n", profileId);
    return kERROR;
}

setup = GoSensor_Setup(profileSensor);

if (!kSuccess(GoSetup_SetActiveAreaZ(setup, GO_ROLE_MAIN, MILS_TO_MM(GoWebScanSettings_
ActiveAreaZ(context->settings)))))
{
    printf("Error: Setting SN %d active area Z failed\n", profileId);
    return kERROR;
}
if (!kSuccess(GoSetup_SetActiveAreaHeight(setup, GO_ROLE_MAIN, MILS_TO_MM
(GoWebScanSettings_ActiveAreaHeight(context->settings)))))
{
    printf("Error: Setting SN %d active area height failed\n", profileId);
    return kERROR;
}
```

Recording and Playing Back Scan Data

GoWebScan can record scan data in Detection mode or Web mode (the resulting recording is the same regardless of the mode). You can then load the scan data for playback in your client application using the `GoWebScanRecording` class. Note that in order to replay scan data when the sensors are not connected, you must make sure that your client application, while recording, saves a `GeoCal.xml` file for each profile and vision sensor, and a `GoWebScanConfigSensorInfo.xml` for each profile sensor, in addition to the recording files themselves. The example application demonstrates how to use the `GoWebScanRecording` class and replay functionality, as well as an example of how to repeatedly reprocess a recording file. Recordings from previous versions of the SDK may still be played back, but only if these files exist for each of the sensors contained in the recording.

Ideally, recording files (*.kdat6) should be kept with the sensor XML files and the Settings.xml file to ensure they correspond to the same setup.

The size of the recording data in memory may be approximated based on the sensor count, frame rate, elapsed time, and size of frames. For example, a system with 22 Gocator 250 / Gocator 205 pairs running at 1500 Hz for 10 seconds may require 1,623,655,000 bytes, or 1.6 GB. The default maximum recording size is 2,147,483,648 bytes (2 GB). If you expect that the recording will be larger, then this value should be increased via the `GoWebScanProcess_MaxRecordSize` function. The example program includes use of this function.

Similarly, the default board count is 2. If more than 2 boards is required, you should increase this value with the GoWebScanProcess_SetMaxRecordBoardCount function.

Although the system should generally be run with all other memory-intensive programs closed, this is especially important while recording data.

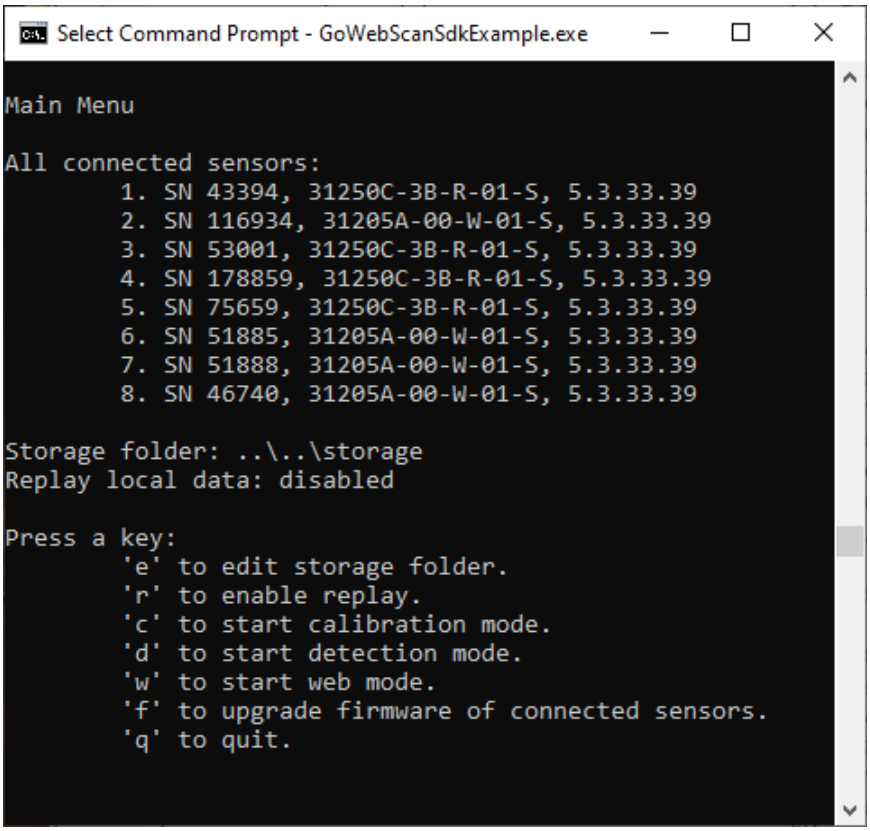
To record multiple boards, it may be more efficient to produce a separate recording file per board, rather than having a single large recording file. This would require stopping the conveyor belt and pausing the system, allowing it to save the recording to disk, then resuming.

If a recording is too big for a given setup, the size can be reduced by lowering the profile frame rate, or recording for a shorter period of time.

Using the Example Application

The example application provides a starting point for developing a calibration and board processing client, and shows how you can use the GoWebScan SDK. In addition to trying the application out, you should examine the source code in the `GoWebScanSdkExample` project to help you understand how the SDK works. The example is available as a project in the GoWebScan solution, in the SDK package.

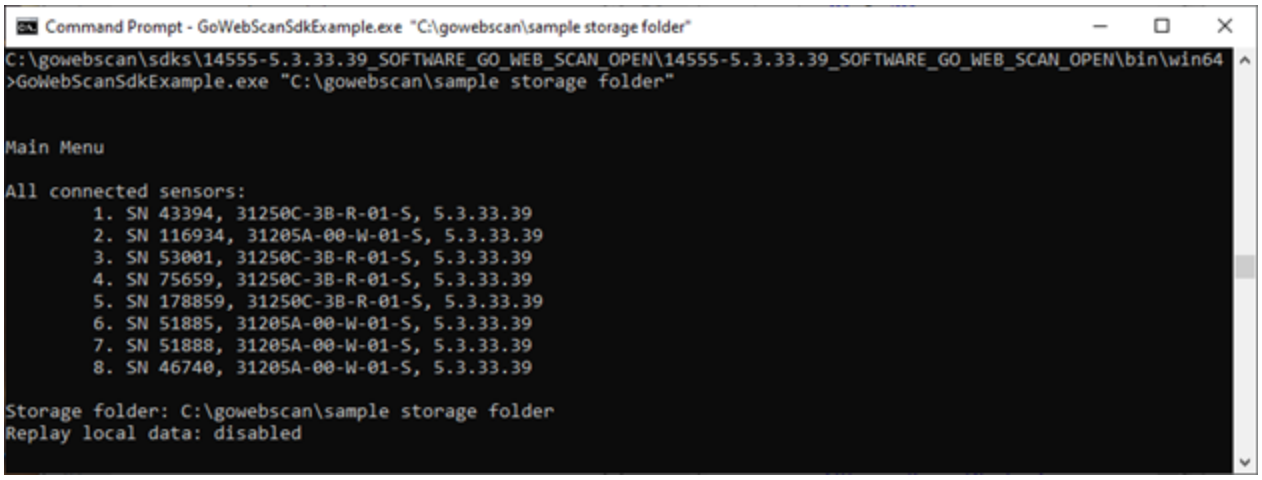
The example is a console application:



The application lists the connected sensors and lets you do the following:

- Choose between running in Calibration, Detection, or Web mode.
- Enable playing previously recorded data.
- Change the storage folder location.

  Note that you can also specify the storage folder location as an argument on the command line.



- Upgrade the firmware of the connected sensors.
- Record data for later playback. This data is also called "replay data."
- Align the calibrations of multiple systems.

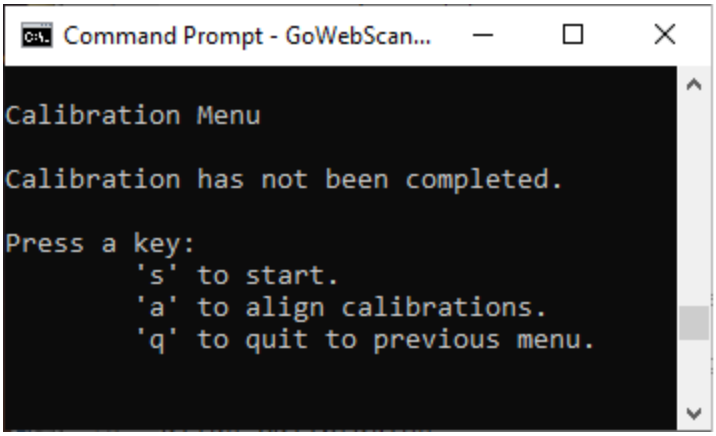
In the example program, after calibration, the mode switches to Detection mode and produces a board image of the calibration bar (with the same data used for calibration).

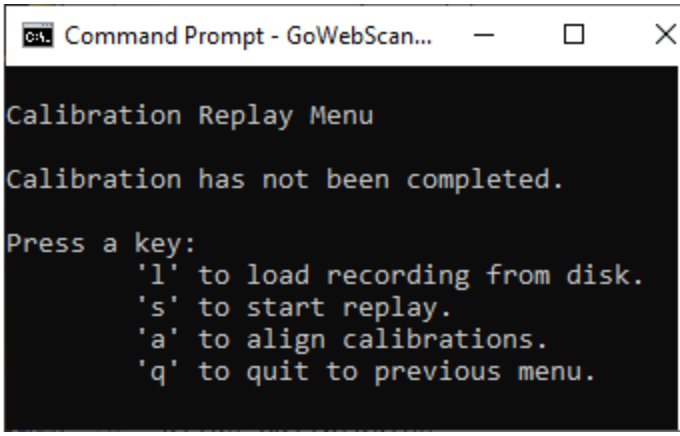
> For most interactions with the example program, type the letter or number of the desired operation and press Enter. After the application finishes processing data, type any alphanumeric key and press Enter.

Main Functions

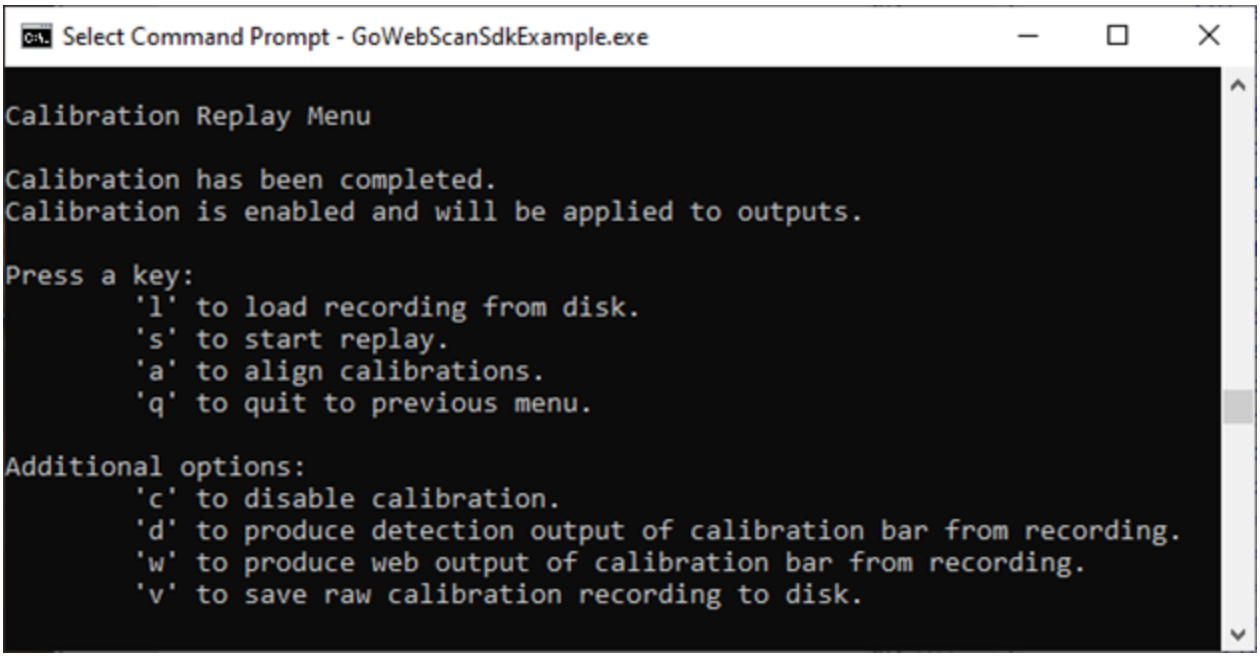
Calibration

The example program lets you start calibration, align calibrations of multiple systems, and (if replay data is enabled), load a recording for use with the calibration.

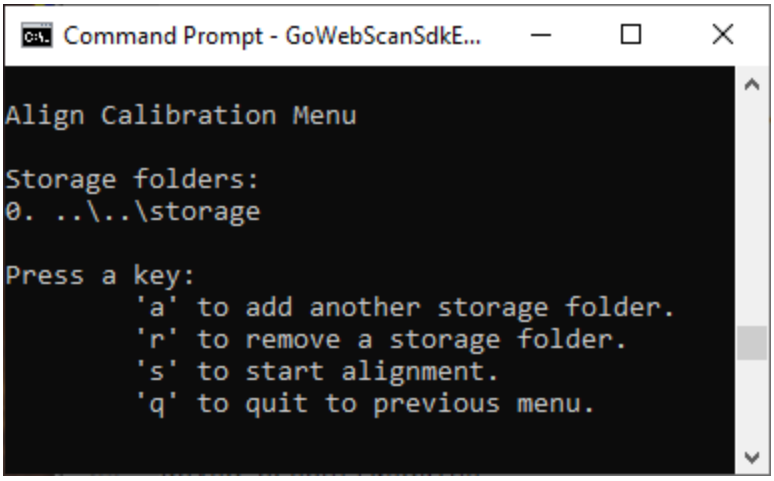




After calibration, using either live data or replay data, the sample lists additional options.

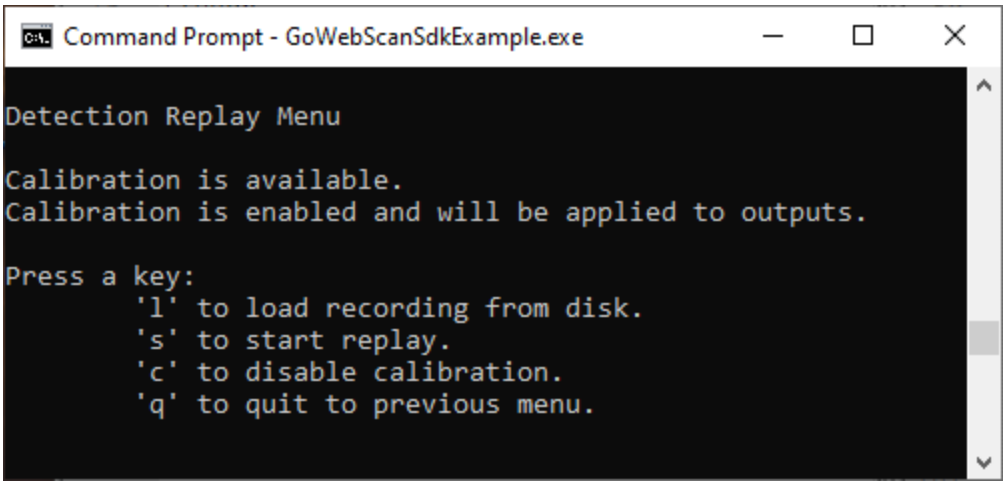


Entering the 'a' option to align calibrations results in a list of storage folders representing the different systems to be aligned, and options to add to this list, remove from this list, or start the alignment.



Detection and Web Replay Menu

Selecting detection or web mode from the main menu with replay enabled results in options load a recording, start replaying it, toggle whether or not calibration is used in the replay,

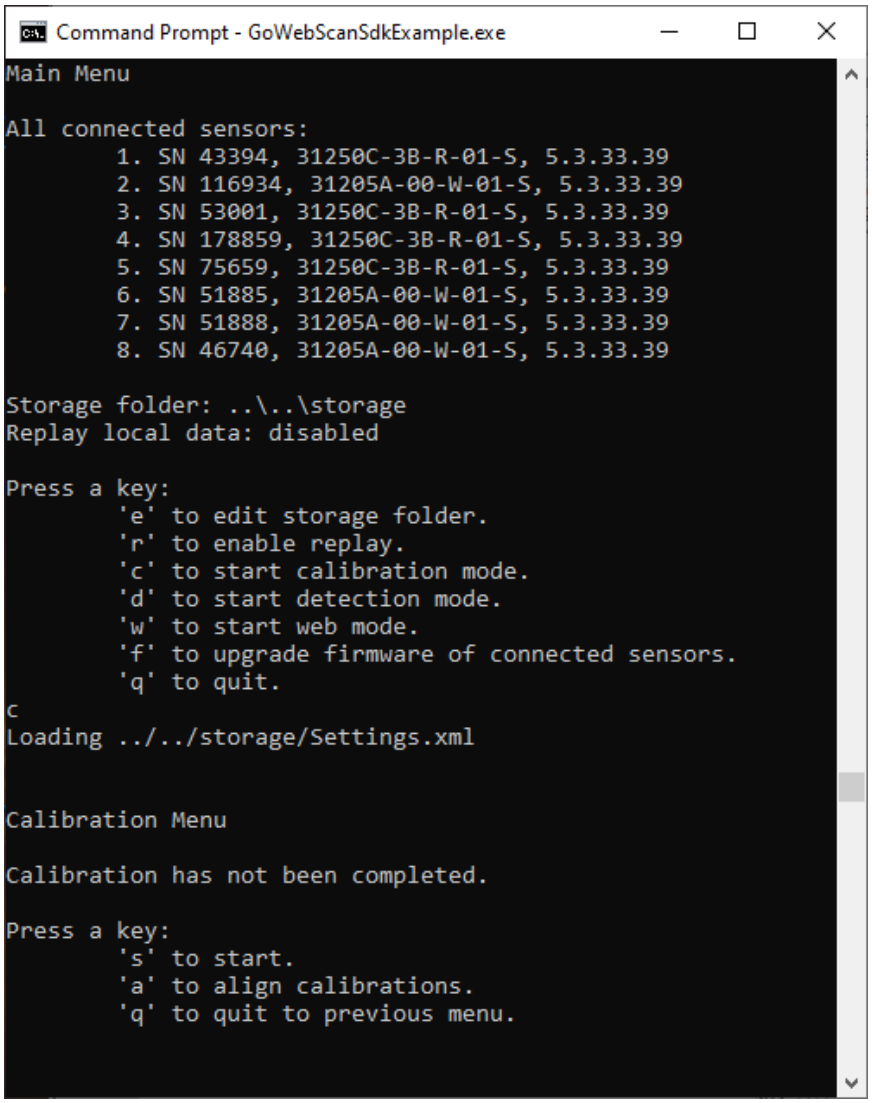


Recording Data

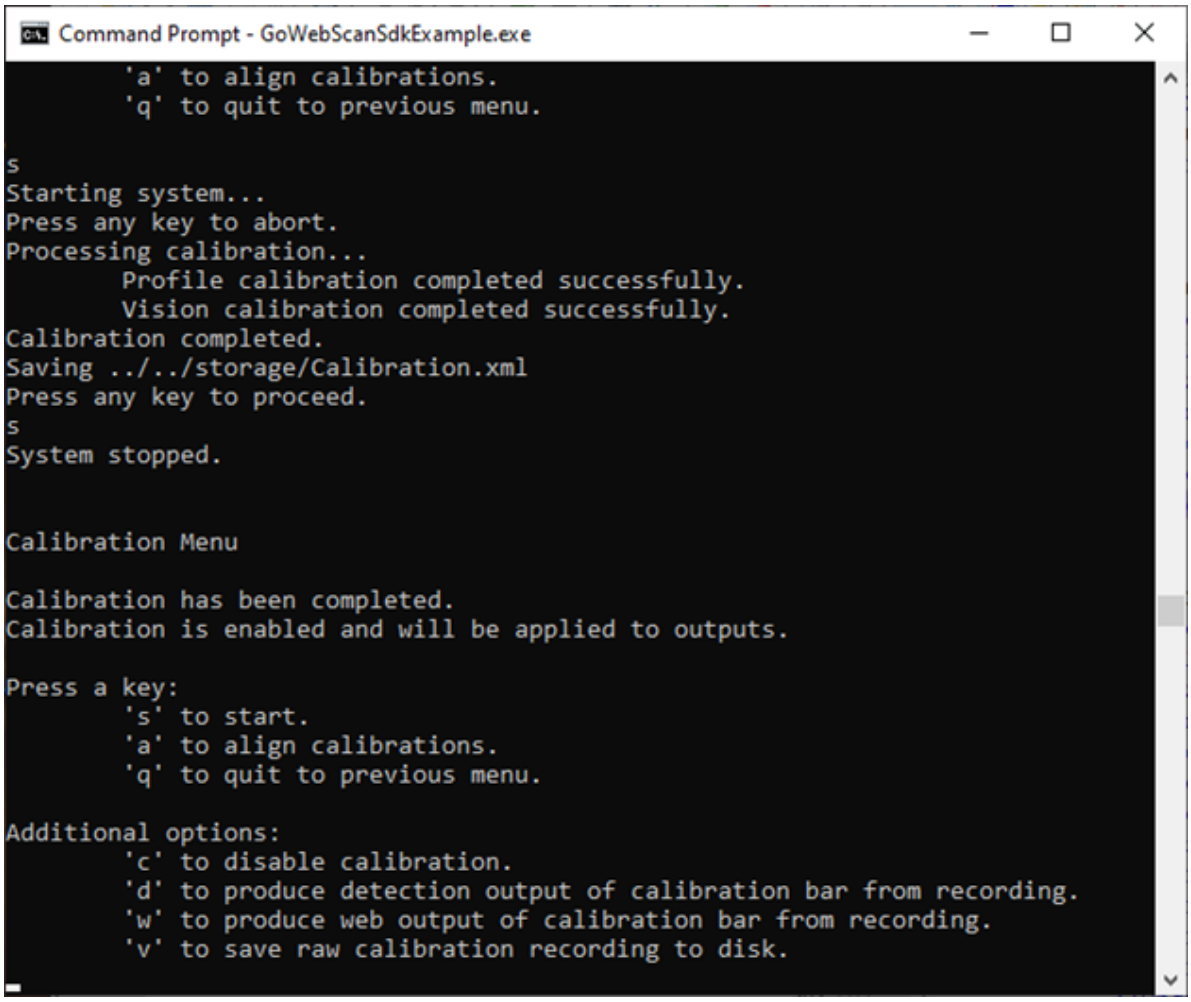
To record scan data and save the sensor configuration files (`GeoCal.xml` and `GoWebScanConfigSensorInfo.xml`), edit the `Settings.xml` file as desired, run the GoWebScanSdkExample.exe application and perform one of the following sets of steps. (For information on these settings, see *Settings.xml File* on page 72.)

To record data in Calibration mode:

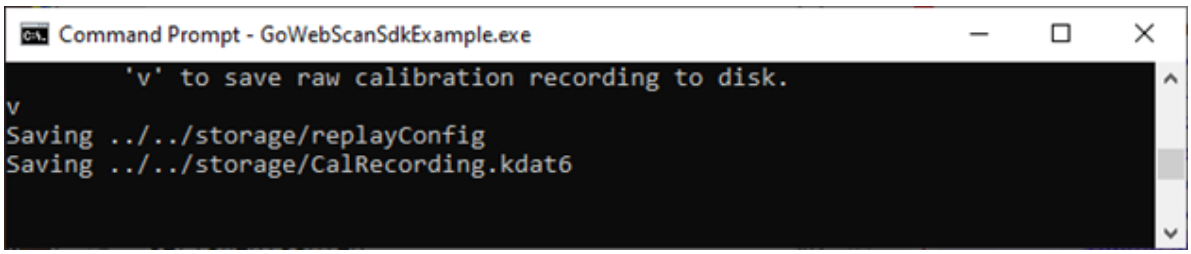
1. Make sure that replay mode is disabled.
2. Press 'c' for Calibration mode and then press ENTER.



3. Press 's' and then press ENTER to start, and perform the calibration.



4. After the example application has finished recording the scan data, press any key to stop.
5. Press 'v' and then ENTER to save the calibration bar data recording.

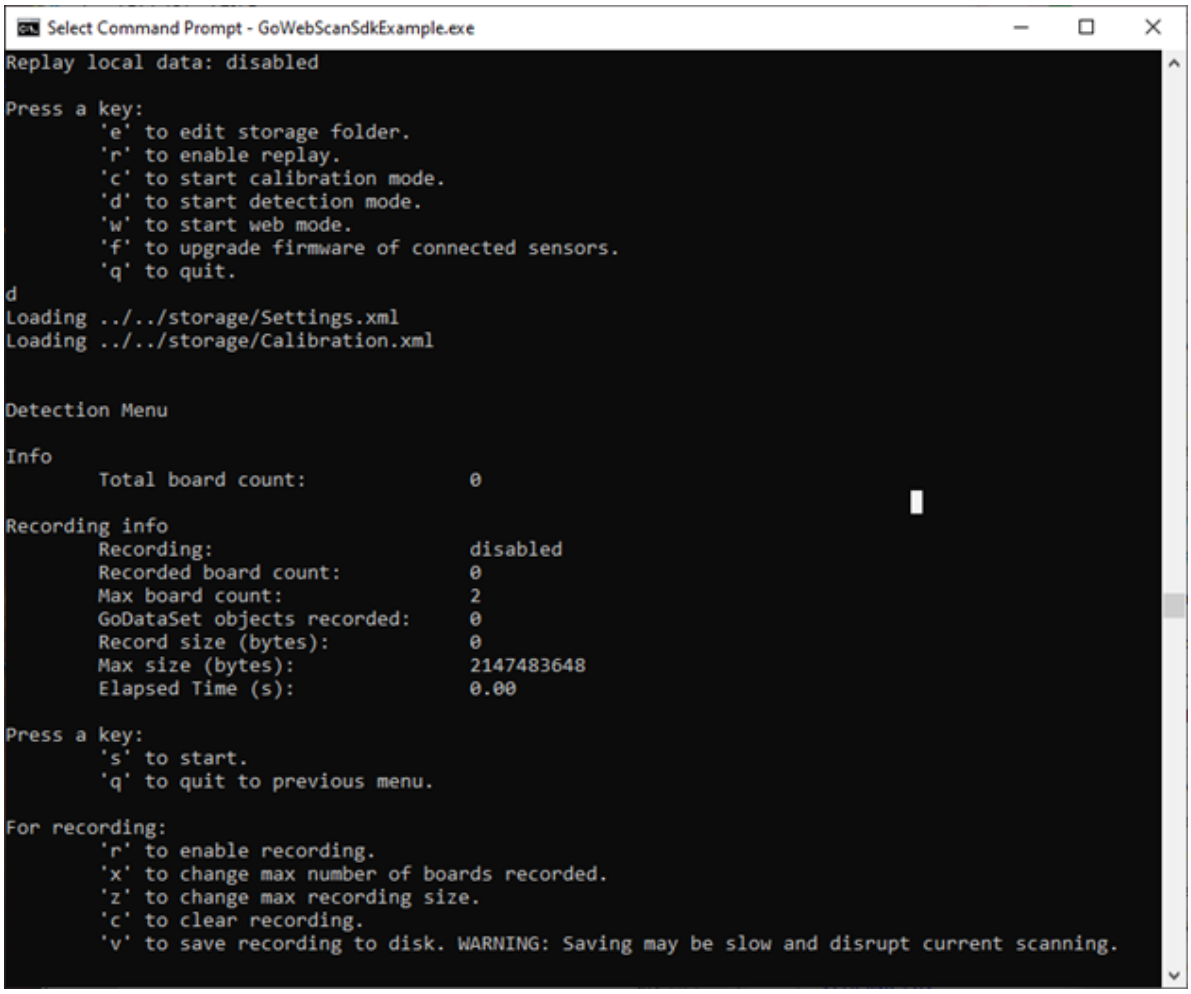


The storage folder should now contain the recording file (for example, "CalRecording.kdat6"), as well as a folder called `replayConfig` containing the `GeoCal.xml` and `GoWebScanConfigSensorInfo.xml` files.

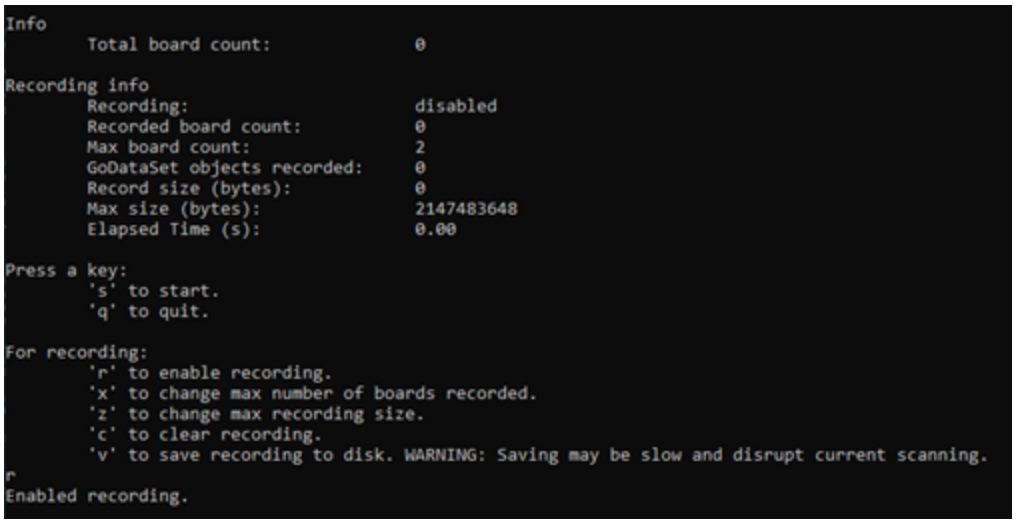
To record data in Detection or Web mode:

1. Make sure that replay mode is disabled.
2. Press 'd' for Detection mode or 'w' for Web mode and then press ENTER.

   You can use either of these modes for recording. The resulting recorded data is the same.



3. Press 'r' and then press ENTER to enable recording.



   For large recordings, you may need to increase the recording size (option 'z'). If several boards are going to be recorded, you will need to increase maximum board count (option 'x').

4. After choosing the recording settings, press 's' and then ENTER, and scan the board.

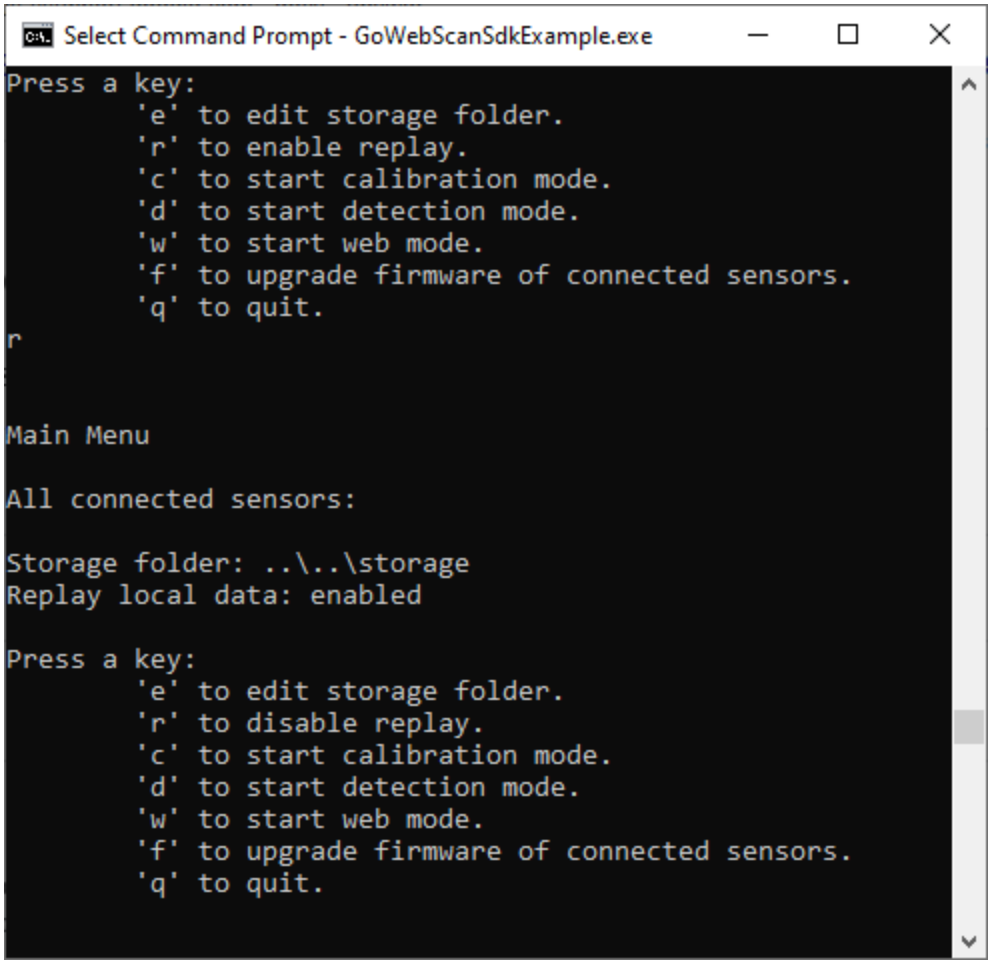
5. After the example application has finished recording the scan data, press any key to stop.
6. Press 'v' and then ENTER to save the recording and the sensor info files to disk.

   The storage folder should now contain the recording file (for example, "goDataSetRecording_00.kdat6"), as well as a folder called `replayConfig` containing subfolders representing each sensor. Each subfolder contains a `GeoCal.xml`, and, if the device is a profile sensor, a `GoWebScanConfigSensorInfo.xml` file.

Replaying Data

To replay recorded data, the Settings.xml file, the .kdat6 recording file, and the replayConfig folder and its contents must be in the storage folder. Disconnect the sensors from the computer, run the GoWebScanSdk example application, and perform the following steps:

1. Press 'r' and then press ENTER to switch to replay mode.

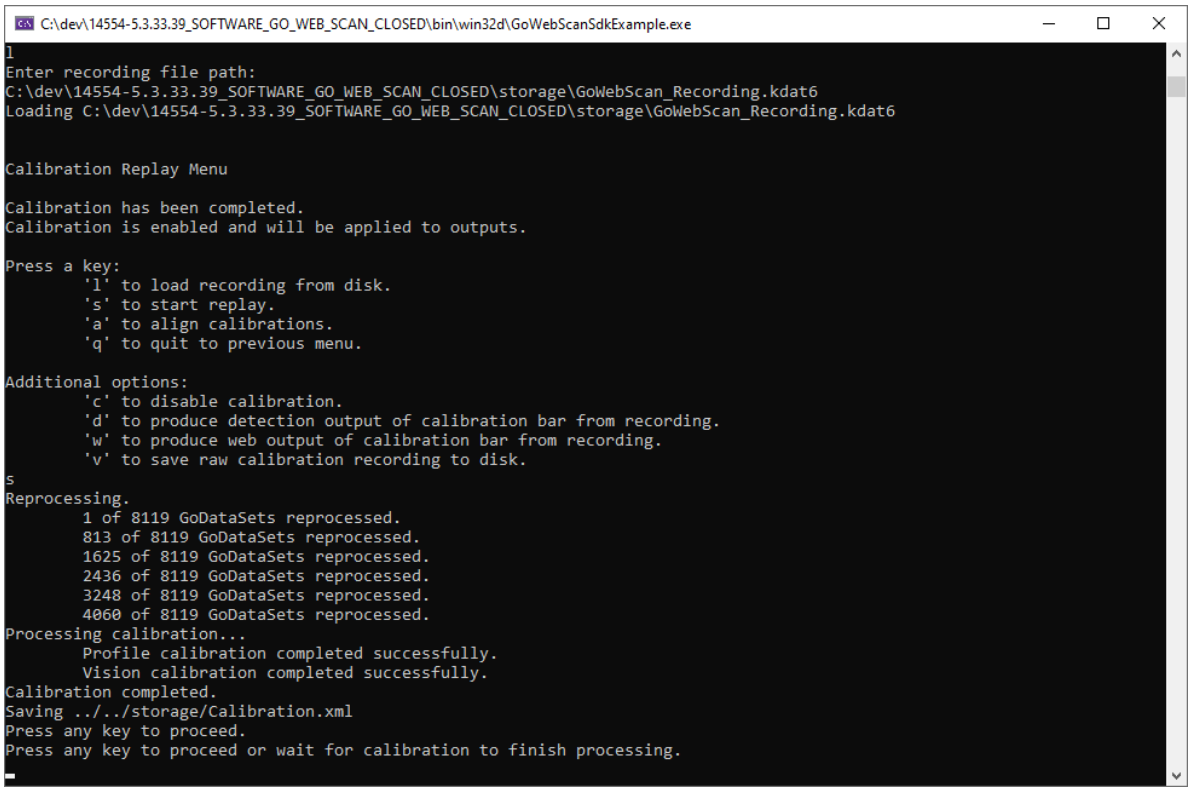


2. Type the option for the desired mode ('c' for calibration, 'd' for detection, or 'w' for web).

Depending on the mode you choose, do one of the following:

To replay data in Calibration mode

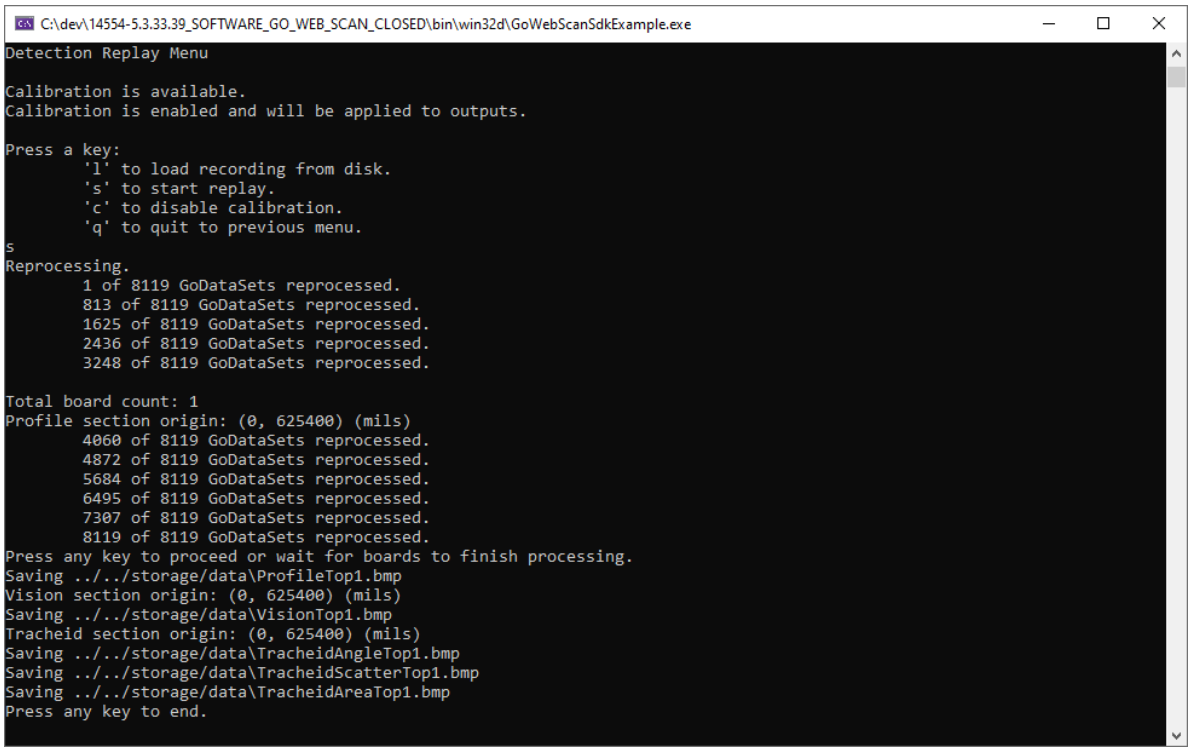
1. Press 'l' to load a recording, providing the recording file path.
2. Press 's' and then ENTER to start.



The example application reprocesses the recorded scan data and produces the same output as when the recording was originally made.

To replay data in Detection or Web mode

1. Press 'l' to load a recording, providing the recording file path.
2. Press 's' and then ENTER to start.



The example application reprocesses the recorded scan data and produces the same output as when the recording was originally made.

Note that the function GoWebScanProcess_Clear is called before each attempted replay in the code. It is necessary to call this to clear the system in the event that a recording to be processed contains earlier encoder value data than what was processed more recently.

Applying Gamma Corrections

To apply gamma corrections to vision board images, set the `visionGammaEnabled` flag to `kTRUE` in the example program code.

```
// Initialize the context struct
void ContextPointer_Init(DataContext* context)
{
...
    context->replayFile = kNULL;
    context->visionGammaEnabled = kTRUE;
    context->gammaLookup = kNULL;
}
```

Enabling this flag creates an 8-bit gamma correction lookup array:

```
kStatus ExampleMain(const kChar* defaultStorageFolderName)
{
...
        // Populate gamma lookup array
```

```
        if (contextPointer.visionGammaEnabled)
        {
            kTest(GoWebScanUtils_PopulateGammaLookup(&contextPointer.gammaLookup, GO_WEB_
SCAN_UTILS_LINEAR_TO_GAMMA_EXPONENT_ESTIMATE, kNULL));
        }
...
}
```

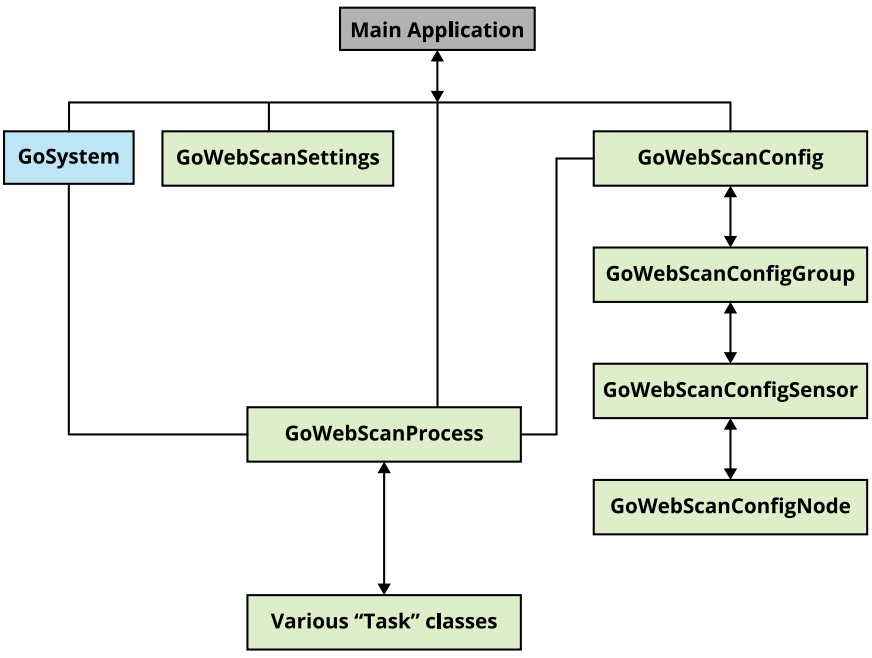
The gamma correction is then applied to the output vision data in `processVisionSectionDetection` and `processVisionSectionWeb`:

```
...
    if (context->visionGammaEnabled)
    {
        kCheck(kArray2_Construct(&gammaData, kTypeOf(kRgb), kArray2_Length
(visionData, 0), kArray2_Length(visionData, 1), kNULL));
        kCheck(GoWebScanUtils_ApplyGammaLookup((kRgb*)kArray2_Data(visionData),
(kRgb*)kArray2_Data(gammaData), kArray2_Count(visionData), context->gammaLookup));

        finalData = gammaData;
    }
    else
    {
        finalData = visionData;
    }
...
```

Required Operations in a Client Application

The following diagram shows the classes that a client application directly interacts with.



- Reads the system settings from the `GoWebScanSettings` object and translates these to system parameters in the `GoWebScanConfig` object.
- Uses the GoSDK to configure the sensors, retrieve files from storage, start the sensors, and receive messages.
- Passes received sensor messages to the `GoWebScanProcess` object, which is responsible for routing a sequence of tasks for that message (depending on its type) and dispatching it to the starting task. These tasks handle all the processing such as resampling, merging, and board detection.
- If any completed board message are available, queries the `GoWebScanProcess` object.
- Saves completed board messages to disk.

The following describes the main steps necessary in a client application:

1. Construct a `GoSdk` library and a `GoWebScanSdk` library.

```
// Construct Gocator API Library
if ((status = GoSdk_Construct(&goApi)) != kOK)
{
    printf("Error: GoSdk_Construct: %d\n", status);
    kTest(status);
}

// Construct GoWebScan API Library
if ((status = GoWebScanSdk_Construct(&goWebScanApi)) != kOK)
{
    printf("Error: GoWebScanSdk_Construct: %d\n", status);
    kTest(status);
}
```

2. Construct a `GoSystem` object and sets a data handler for received sensor messages. Using a data handler lets your application process output data in a separate thread, preventing any time-consuming processing from blocking the input data thread.

```
// Construct system
kTest(GoSystem_Construct(&contextPointer.system, kNULL));

// Set data handler to receive data asynchronously
if ((status = GoSystem_SetDataHandler(contextPointer.system, onData, &contextPointer)) !=
kOK)
{
    printf("Error: GoSystem_SetDataHandler: %d\n", status);
    kTest(status);
}
```

3. Load the settings from `Settings.xml` into a `GoWebScanSettings` object using `GoWebScanSettings_Load()`.

```
// Read system settings
kTest(kPath_Combine(context->storageFolderName, "Settings.xml", pathName, kCountOf
(pathName)));
kTest(GoDestroy(context->settings));
context->settings = kNULL;
kTest(loadSettings(&context->settings, pathName));

// Verify the system settings
if (!kSuccess(status = GoWebScanUtils_VerifySystemTiming(context->settings,
errorText, kCountOf(errorText))))
{
    printf("Error (code=%d): %s\n", status, errorText);
    kTest(status);
}
```

4. If in Calibration mode, load the `GoWebScanCalTarget` file. Otherwise, if in Web or Detection mode, load the `GoWebScanCal` file (Calibration.xml) previously created from a calibration.

```
// Read calibration target info
kTest(kPath_Combine(context->storageFolderName, "CalibrationTarget.xml",
pathName, kCountOf(pathName)));
if (!kSuccess(status = GoWebScanCalTarget_Load(&context->calTarget, pathName, kNULL)))
{
    printf("Error (code=%d): Unable to load calibration target dimensions. Assuming
defaults.\n", status);
    kTest(GoWebScanCalTarget_Construct(&context->calTarget, kNULL));
}


...

// Read calibration
kTest(kPath_Combine(context->storageFolderName, "Calibration.xml", pathName, kCountOf
(pathName)));
if (!kSuccess(loadCalibration(&context->calibration, pathName)))
{
    printf("Running uncalibrated system.\n");
}
```

5. Configure the sensors.

   With live sensors, connect to each enabled sensor, enables data, configures the timing, sets the active area, and sets the transmit limit, and so on. For systems with vision sensors, timing should be configured as in the example application to avoid interference.

   When in replay mode, gets sensor information from the previously saved `GoWebScanConfigSensorInfo.xml` and `GeoCal.xml` files.

```
// Set up sensor lists
kTest(kArrayList_Construct(&contextPointer.profileSensors, kTypeOf(GoSensor), 0, kNULL));
kTest(kArrayList_Construct(&contextPointer.visionSensors, kTypeOf(GoSensor), 0, kNULL));
kTest(kArrayList_Construct(&contextPointer.sensorInfo, kTypeOf
(GoWebScanConfigSensorInfo), 0, kNULL));
```

```
...

if (context->liveSensors)
{
    // Configure sensors
    kTest(configureSensors(context));
}
else
{
    // Load sensor info for replay
    kTest(loadSensorInfoForReplay(context));
}
```

6. Construct the `GoWebScanConfig` object.

   If in Detection mode, the application constructs the `GoWebScanProcess` object.

   If in Calibration mode, the application constructs the `GoWebScanCalCollector` and `GoWebScanCalProcessor` objects.

```
// Construct GoWebScan config class
kTest(GoWebScanConfig_Construct(&context->config, context->settings, context->calTarget,
kNULL, context->sensorInfo, context->mode, kNULL));

// Construct GoWebScan processor class for potential detection or web mode reprocessing
kTest(GoWebScanProcess_Construct(&context->process, context->config, kNULL));
```

7. With live sensors, the application starts the system using a scheduled start.

   If the system includes a vision sensor, the sensor start time calculates a delay for the scheduled start of the vision sensors.

```
for (i = 0; i < kArrayList_Count(context->profileSensors); i++)
{
    GoSensor profileSensor = *(GoSensor*)kArrayList_At(context->profileSensors, i);
    GoWebScanSettingsSensor sensorSettings = GoWebScanSettings_SensorAt(context-
>settings, i);

    if (GoWebScanSettingsSensor_Enabled(sensorSettings))
    {
        // Set up scheduled start of G250
        if (!kSuccess(GoSensor_ScheduledStart(profileSensor, (k64s)(currentTime +
systemStartupTime))))
        {
            printf("Error: Unable to start SN %d\n", GoWebScanSettingsSensor_
ProfileSerialNumber(sensorSettings));
            return kERROR;
        }

        if (GoWebScanSettings_VisionEnabled(context->settings))
        {
            GoSensor visionSensor = *(GoSensor*)kArrayList_At(context->visionSensors, i);

            delay = calculateVisionStartDelay(context, sensorSettings);
```

```
            // Set up scheduled start of G205
            if (!kSuccess(GoSensor_ScheduledStart(visionSensor, (k64s)(currentTime +
systemStartupTime + delay))))
            {
                printf("Error: Unable to start SN %d\n", GoWebScanSettingsSensor_
VisionSerialNumber(sensorSettings));
                return kERROR;
            }
        }
    }
}
```

8. If in replay mode, load the recording using `GoWebScanRecording_Load()` and process the recording data with the detection algorithm or the calibration algorithm, based on the user's selection.

```
printf("Enter recording file path: \n");
kTestArgs(scanf("%s", pathName) == 1);
kTest(loadRecording(&recordingObj, pathName));

*recording = recordingObj;
```

9. In the data handler for sensor messages from live sensors, the application performs one of the following:

   If in Detection or Web mode, the application passes the messages to `GoWebScanProcess_Process()` and queries `GoWebScanProcess_Receive()` if any completed boards are available.

   If in Calibration mode, the application passes the messages to `GoWebScanCalCollector_AddMsg()` until the collector is completed.

```
kStatus kCall onData(void* ctx, GoSensor system, GoDataSet dataSet)
{
    DataContext* context = (DataContext*)ctx;

    if (context->systemRunning)
    {
        if (context->mode == GO_WEB_SCAN_MODE_DETECTION || context->mode == GO_WEB_SCAN_
MODE_WEB)
        {
            // Send sensor data to GoWebScan processor
            kCheck(GoWebScanProcess_Process(context->process, dataSet));
        }
        else if (context->mode == GO_WEB_SCAN_MODE_CALIBRATION)
        {
            // Check if cal collector has completed
            if (!GoWebScanCalCollector_IsCompleted(context->calCollector))
            {
                // Add a message to the cal collector
                kCheck(GoWebScanCalCollector_AddMsg(context->calCollector, dataSet));
            }
        }
    }

    kCheck(GoDestroy(dataSet));

    return kOK;
```

```
}
```

10. If boards are available in Detection mode, in the `onBoard()` data handler the application extracts the profile, vision, or tracheid data, and maps the data to one of the user-specified sections.

    At this point the data output is complete.

11. In Calibration mode, the application produces a calibration dataset using `GoWebScanCalCollector_GetProcessorData()` and passes the dataset to `GoWebScanCalProcessor_Execute()`, which will return the completed `GoWebScanCal` object if no errors occurred and saves the calibration to `Calibration.xml`.

12. If recording is enabled (available in Detection and Web modes with live sensors), `GoWebScanProcess` will record copies of all incoming `GoDataSet` messages that are processed through `GoWebScanProcess_Process()`.

    Recording is enabled using `GoWebScanProcess_EnableRecording`,

    The recording can be saved by calling `GoWebScanProcess_SaveRecording()`. Saving recording data is useful for offline testing and debugging purposes.

Gocator Web Interface

Gocator 200 series systems are typically configured using GoSDK and GoWebScanSDK. You can however connect to an individual sensor and use the built-in web interface for initial setup to determine optimum settings such as exposure, for troubleshooting, and for monitoring purposes. To do so, you connect to the IP address of a sensor with a web browser; for more information on connecting to a sensor, see *Gocator Setup* on page 52.

> With the exception of tracheid threshold settings, any changes you make through the Gocator interface or through GoSDK are overridden or ignored by GoWebScanSDK. You *must* set tracheid threshold settings on each sensor, ideally using GoSDK.

The following sections describe the Gocator web interface.

Browser Compatibility

LMI recommends Chrome, Firefox, or Edge for use with the Gocator web interface.

Internet Explorer 11 is supported with limitations; for more information, see below.

Internet Explorer 11 Issues

If you use sensors with large datasets on Internet Explorer 11, you may encounter the following issues.

Internet Explorer Switches to Software Rendering

If the PC connected to a sensor is busy, Internet Explorer may switch to software rendering after a specific amount of time. If this occurs, data is not displayed in the data viewer, and the only reliable way to recover from the situation is to restart the browser.

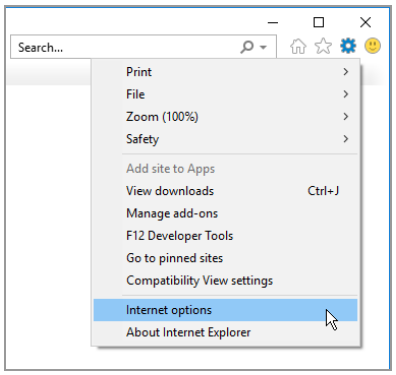
It is possible to remove the time limit that causes this issue, but you must modify the computer's registry. To do so, follow Microsoft's instructions at https://support.microsoft.com/en-us/help/3099259/update-to-add-a-setting-to-disable-500-msec-time-limit-for-webgl-frame.

Internet Explorer Displays "Out of Memory"

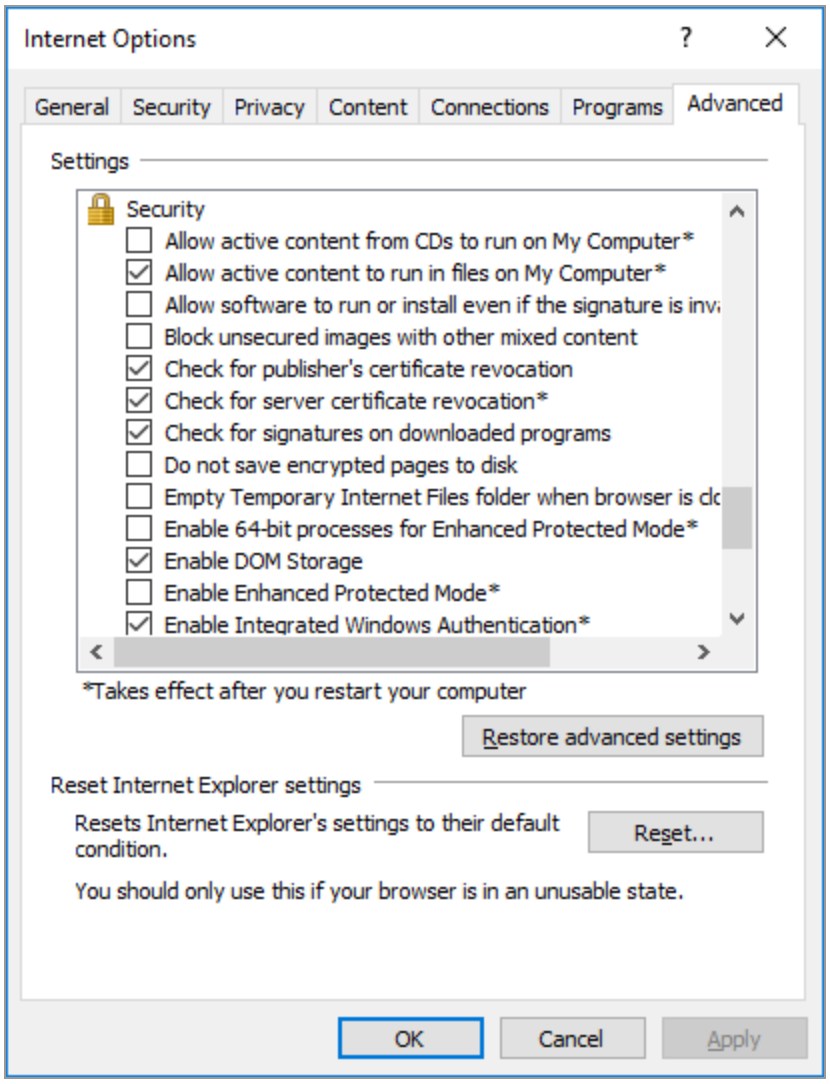
In some situations, you may encounter "Out of Memory" errors in the sensor's web interface. This issue can be resolved by checking two options in Internet Explorer.

To correct out of memory issues in Internet Explorer 11:

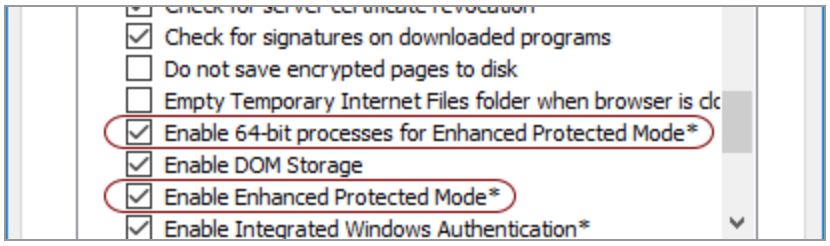
1. In upper right corner, click the settings icon (⚙), and choose **Internet options**.



2. In the dialog that opens, click the **Advanced** tab, and scroll down to the **Security** section.



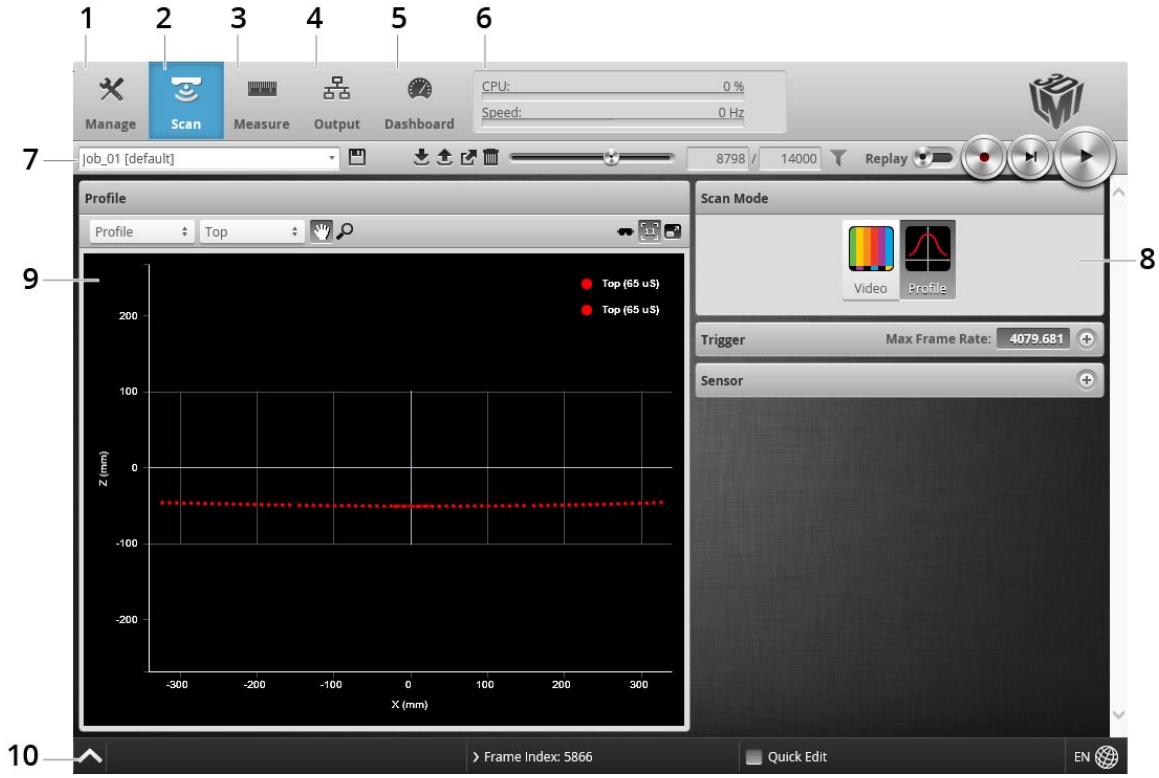
3. In the dialog, check *both* "Enable 64-bit processes for Enhanced Protected Mode" *and* "Enable Enhanced Protected Mode".



4. Click **OK** and then restart your computer for the changes to take effect.

User Interface Overview

The web interface is shown below.

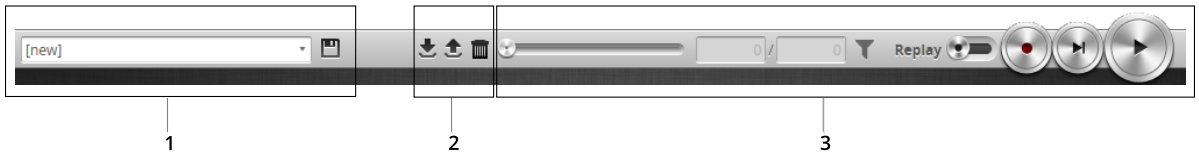


	Element	Description
1	Manage page	Contains settings for sensor system layout, network, motion and alignment, handling jobs, and sensor maintenance. See *Management and Maintenance* on page 116.
2	Scan page	Contains settings for scan mode, trigger source, detailed sensor configuration, and performing alignment. See *Scan Setup and Alignment* on page 129.
3	Measure page	Currently, Gocator 200 series sensors do not provide built-in measurement tools.

	Element	Description
4	Output page	Contains settings for configuring output protocols used to communicate measurements to external devices. See *Output* on page 170.
5	Dashboard page	Provides monitoring of measurement statistics and sensor health. See *Dashboard* on page 175.
6	CPU Load and Speed	Provides important sensor performance metrics. See *Metrics Area* on page 112.
7	Toolbar	Controls sensor operation, manages jobs, and filters and replays recorded data. See *Toolbar* below.
8	Configuration area	Provides controls to configure scan and measurement tool settings.
9	Data viewer	Displays sensor data.
10	Status bar	Displays log messages from the sensor (errors, warnings, and other information) and frame information, and lets you switch the interface language. For more information, see *Status Bar* on page 113.

Toolbar

The toolbar is used for performing operations such as managing jobs, working with replay data, and starting and stopping the sensor.



	Element	Description
1	Job controls	For saving and loading jobs.
2	Replay data controls	For downloading, uploading, and exporting recorded data.
3	Sensor operation / replay control	Use the sensor operation controls to start sensors, enable and filter recording, and control recorded data.

Creating, Saving and Loading Jobs (Settings)

A sensor can store several hundred jobs. 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 can be changed in the sensor's web interface, such as the ones in the **Manage**, **Measure**, and **Output** pages, are temporary until saved in a job file. Each sensor can have multiple job files. If there is a job file that is designated as the default, it will be loaded automatically when the sensor is reset.

When you change sensor settings using the sensor web interface in the emulator, some changes are saved automatically, while other changes are temporary until you save them manually. The following table lists the types of information that can be saved in a sensor.

Setting Type	Behavior
Job	Most of the settings that can be changed in the sensor's web interface, such as the ones in the **Manage**, **Measure**, and **Output** pages, are temporary until saved in a job file. Each sensor can have multiple job files. If there is a job file that is designated as the default, it will be loaded automatically when the sensor is reset.
Alignment	Alignment can either be fixed or dynamic, as controlled by the **Alignment Reference** setting in **Motion and Alignment** in the **Manage** page. Alignment is saved automatically at the end of the alignment procedure when **Alignment Reference** is set to **Fixed**. When **Alignment Reference** is set to **Dynamic**, however, you must manually save the job to save alignment.
Network Address	Network address changes are saved when you click the **Save** button in **Networking** on the **Manage** page. The sensor must be reset before changes take effect.

The job drop-down list in the toolbar shows the jobs stored in the sensor. The job that is currently active is listed at the top. The job name will be marked with "[unsaved]" to indicate any unsaved changes.

To create a job:

1. Choose **[New]** in the job drop-down list and type a name for the job.
2. Click the **Save** button or press **Enter** to save the job.
   The job is saved to sensor storage using the name you provided. Saving a job automatically sets it as the default, that is, the job loaded when then sensor is restarted.

To save a job:

- Click the **Save** button.

  The job is saved to sensor storage. Saving a job automatically sets it as the default, that is, the job loaded when then sensor is restarted.

To load (switch) jobs:

- Select an existing file name in the job drop-down list.

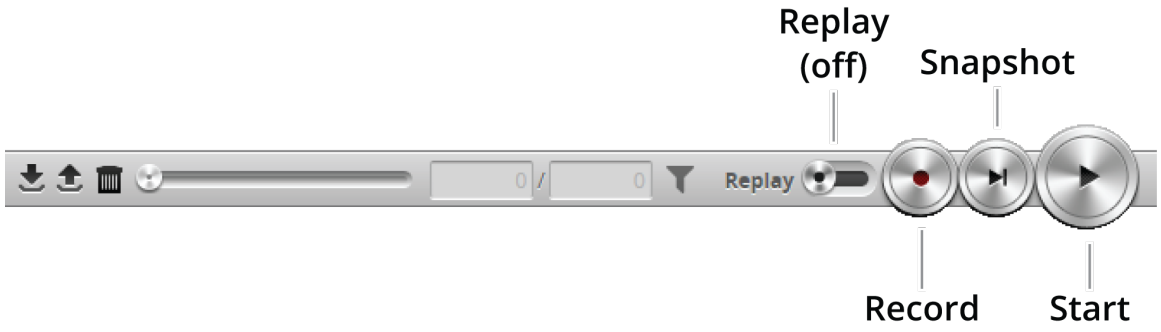
  The job is activated. If there are any unsaved changes in the current job, you will be asked whether you want to discard those changes.

You can perform other job management tasks—such as downloading job files from a sensor to a computer, uploading job files to a sensor from a computer, and so on—in the **Jobs** panel in the **Manage** page. See *Jobs* on page 120 for more information.

Recording, Playback, and Measurement Simulation

Sensors can record and replay recorded scan data, and also simulate measurement tools on recorded data. This feature is most often used for troubleshooting and fine-tuning measurements, but can also be helpful during setup.

Recording and playback are controlled using the toolbar controls.



Recording and playback controls when replay is off

To record live data:

1. Toggle **Replay** mode off by setting the slider to the left in the **Toolbar**.

   Replay mode disables measurements.

2. (Optional) Configure recording filtering.
   For more information on recording filtering, see *Recording Filtering* on page 109.

3. Click the **Record** button to enable recording.

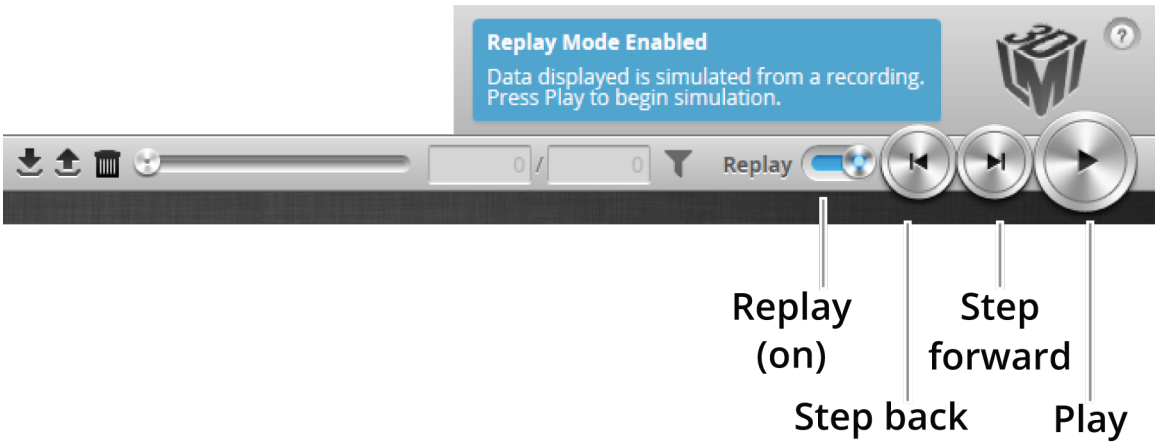
   

   The center of the Record button turns red.

   When recording is enabled (and replay is off), the sensor will store the most recent data as it runs. Remember to disable recording if you no longer want to record live data. (Press the **Record** button again to disable recording).

4. Press the **Snapshot** button or **Start** button.
   The **Snapshot** button records a single frame. The **Start** button will run the sensor continuously and all frames will be recorded, up to available memory. When the memory limit is reached, the oldest data will be discarded.

Newly recorded data is appended to existing replay data unless the sensor job has been modified.



Playback controls when replay is on

To replay data:

1. Toggle **Replay** mode on by setting the slider to the right in the **Toolbar**.
   The slider's background turns blue and a Replay Mode Enabled message is displayed.

2. Use the **Replay** slider or the **Step Forward**, **Step Back**, or **Play** buttons to review data.
   The **Step Forward** and **Step Back** buttons move the current replay location forward and backward by a single frame, respectively.
   The **Play** button advances the replay location continuously, animating the playback until the end of the replay data.
   The **Stop** button (replaces the **Play** button while playing) can be used to pause the replay at a particular location.
   The **Replay** slider (or **Replay Position** box) can be used to go to a specific replay frame.

To simulate measurements on replay data:

1. Toggle **Replay** mode on by setting the slider to the right in the **Toolbar**.
   The slider's background turns blue and a Replay Mode Enabled message is displayed.
   To change the mode, **Replay Protection** must be unchecked.

2. Go to the **Measure** page.
   Modify settings for existing measurements, add new measurement tools, or delete measurement tools as desired. For information on adding and configuring measurements, see *Measurement and Processing* on page 146.

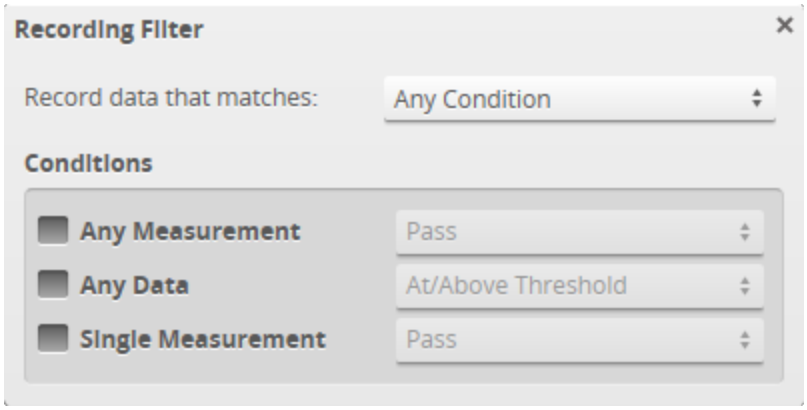
3. Use the **Replay Slider**, **Step Forward**, **Step Back**, or **Play** button to simulate measurements.
   Step or play through recorded data to execute the measurement tools on the recording.
   Individual measurement values can be viewed directly in the data viewer. Statistics on the measurements that have been simulated can be viewed in the **Dashboard** page; for more information on the dashboard, see *Dashboard* on page 175.

To clear replay data:

1. Stop the sensor if it is running by clicking the **Stop** button.
2. Click the **Clear Replay Data** button 🗑.

Recording Filtering

Replay data is often used for troubleshooting. But replay data can contain thousands of frames, which makes finding a specific frame to troubleshoot difficult. Recording filtering lets you choose which frames the sensor records, based on one or more conditions, which makes it easier to find problems.



How a sensor treats conditions

Setting	Description
Any Condition	The sensor records a frame when any condition is true.
All Conditions	The sensor only records a frame if all conditions are true.

Conditions

Setting	Description
Any Measurement	The sensor records a frame when *any* measurement is in the state you select. The following states are supported: • pass • fail or invalid • fail and valid • valid • invalid
Single Measurement	The sensor records a frame if the measurement with the ID you specify in **ID** is in the state you select. This setting supports the same states as the **Any Measurement** setting (see above).
Any Data	**At/Above Threshold**: The sensor records a frame if the number of valid points in the frame is above the value you specify in **Range Count Threshold**. **Below Threshold**: The sensor records a frame if the number of valid points is below the threshold you specify.

To set recording filtering:

1. Make sure recording is enabled by clicking the Record button.

2. Click the Recording Filtering button .
3. In the Recording Filtering dialog, choose how the sensor treats conditions:

   For information on the available settings, see *How a sensor treats conditions* on the previous page.
4. Configure the conditions that will cause the sensor to record a frame:

   For information on the available settings, see *Conditions* on the previous page.
5. Click the "x" button or outside of the Recording Filtering dialog to close the dialog.

   The recording filter icon turns green to show that recording filters have been set.

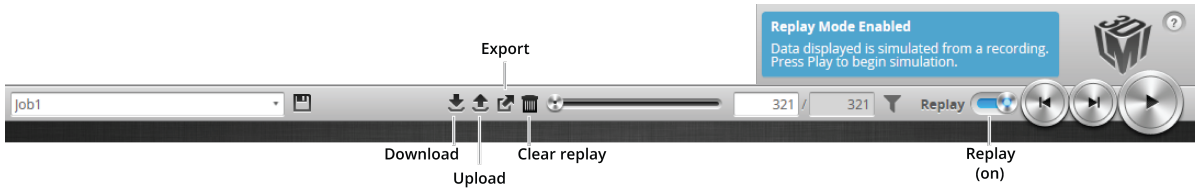
   When you run the sensor, it only records the frames that satisfy the conditions you have set.

Downloading, Uploading, and Exporting Replay Data

Replay data (recorded scan data) can be downloaded from a sensor to a client computer, or uploaded from a client computer to a sensor.

Data can also be exported from a sensor to a client computer in order to process the data using third-party tools.

> You can only upload replay data to the same sensor model that was used to create the data.



> Replay data is not loaded or saved when you load or save jobs.

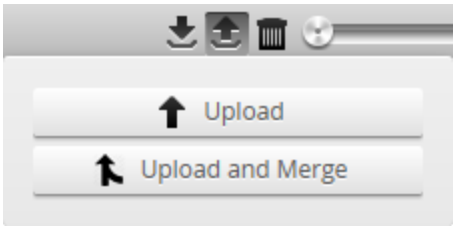
To download replay data:

1. Click the Download button .
2. In the **File Download** dialog, click **Save**.
3. In the **Save As...** dialog, choose a location, optionally change the name, and click **Save**.

To upload replay data:

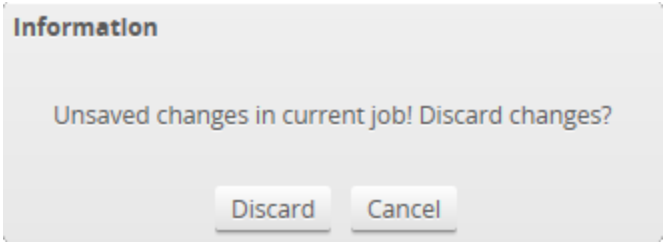
1. Click the Upload button .

   The Upload menu appears.



2. In the Upload menu, choose one of the following:
   - **Upload**: Unloads the current job and creates a new unsaved and untitled job from the content of the replay data file.
   - **Upload and merge**: Uploads the replay data and merges the data's associated job with the current job. Specifically, the settings on the **Scan** page are overwritten, but all other settings of the current job are preserved, including any measurements.

   If you have unsaved changes in the current job, the firmware asks whether you want to discard the changes.



3. Do one of the following:
   - Click **Discard** to discard any unsaved changes.
   - Click **Cancel** to return to the main window to save your changes.
4. If you clicked **Discard**, navigate to the replay data to upload from the client computer and click **OK**. The replay data is loaded, and a new unsaved, untitled job is created.

Replay data can be exported using the CSV format.

To export replay data in the CSV format:

1. In the **Scan Mode** panel, switch to .
2. Switch to Replay mode.

3. Click the Export button and select **All Data as CSV**.

   data at the current replay location is exported.
   Use the playback control buttons to move to a different replay location; for information on playback, see *To replay data* in *Recording, Playback, and Measurement Simulation* on page 107.

4. (Optional) Convert exported data to another format using the CSV Converter Tool. For information on this tool, see *CSV Converter Tool* on page 338.

> The decision values in the exported data depend on the *current* state of the job, not the state during recording. For example, if you record data when a measurement returns a *pass* decision, change the measurement's settings so that a *fail* decision is returned, and then export to CSV, you will see a *fail* decision in the exported data.

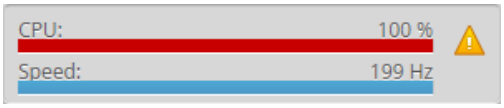
To export video data to a BMP file:

1. In the **Scan Mode** panel, switch to Video mode.
   Use the playback control buttons to move to a different replay location; for information on playback, see *To replay data* in *Recording, Playback, and Measurement Simulation* on page 107.
2. Switch to Replay mode.
3. Click the Export button and select **Video data as BMP**.

Metrics Area

The **Metrics** area displays two important sensor performance metrics: CPU load and speed (current frame rate).

The **CPU** bar in the **Metrics** panel (at the top of the interface) displays how much of the CPU is being utilized.



CPU at 100%

The **Speed** bar displays the frame rate of the sensor. A warning symbol (⚠) will appear next to it if triggers (external input or encoder) are dropped because the external rate exceeds the maximum frame rate.

Open the log for details on the warning. For more information on logs, see *Log* on the next page.

Data Viewer

The data viewer is displayed in both the **Scan** and the **Measure** pages, but displays different information depending on which page is active.

When the **Scan** page is active, the data viewer displays sensor data and can be used to adjust the active area and other settings. Depending on the selected operation mode (page 130), the data viewer can display video images. For details, see *Data Viewer* on page 140.

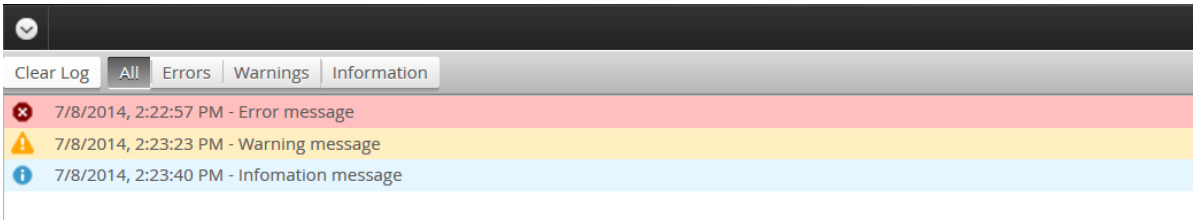
Status Bar

The status bar lets you do the following:

- See sensor messages in the log.
- See frame information.
- Change the interface language.
- Switch to Quick Edit mode.

Log

The log, located at the bottom of the web interface, is a centralized location for all messages that the sensor displays, including warnings and errors.



A number indicates the number of unread messages:

To use the log:

1. Click on the Log open button at the bottom of the web interface.
2. Click on the appropriate tab for the information you need.

Frame Information

The area to the right of the status bar displays useful frame information, both when the sensor is running and when viewing recorded data.

This information is especially useful when you have enabled recording filtering. If you look at a recording playback, when you have enabled recording filtering, some frames can be excluded, resulting in variable "gaps" in the data.

The following information is available:

Frame Index: Displays the index in the data buffer of the current frame. The value resets to 0 when the sensor is restarted or when recording is enabled.

Master Time: Displays the recording time of the current frame, with respect to when the sensor was started.

Encoder Index: Displays the encoder value at the time of the last encoder Z index pulse. Note this is not the same as the encoder value at the time the frame was captured.

Timestamp: Displays the timestamp the current frame, in microseconds from when the sensor was started.

To switch between types of frame information:

- Click the frame information area to switch to the next available type of information.

Quick Edit Mode

When this mode is enabled, the data viewer is not refreshed after each setting change. Also, when Quick Edit is enabled, in Replay mode, stepping through frames or playing back scan data does not change the displayed frame.

> Quick Edit mode is mostly useful on sensors that support measurement tools and GDK tools. Currently, Gocator 200 series sensors do not support these features.

> When a sensor is running, Quick Edit mode is ignored: all changes to settings are reflected immediately in the data viewer.

Interface Language

The language button on the right side of the status bar at the bottom of the interface lets you change the language of the interface.

To change the language:

1. Click the language button at the bottom of the web interface.
   

2. Choose a language from the list.

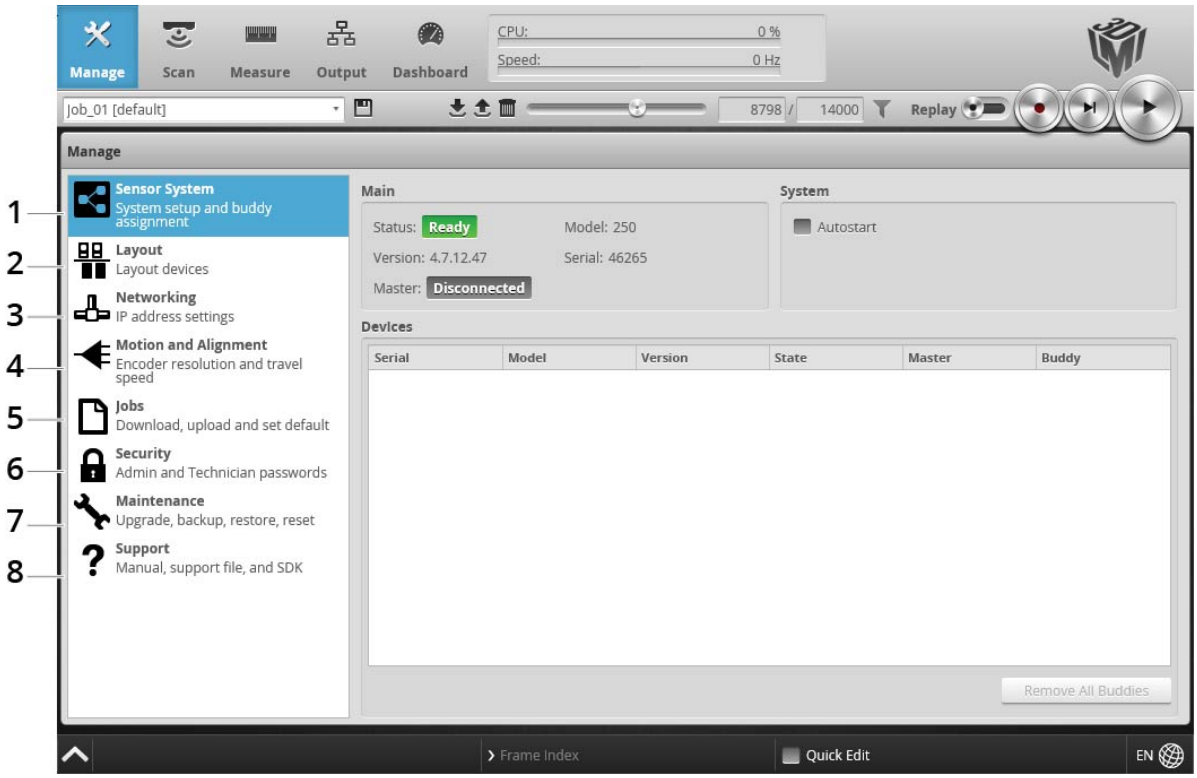
The interface reloads on the page you were working in, displaying the page using the language you chose. The sensor state is preserved.

Management and Maintenance

The following sections describe how to set up the sensor connections and networking, how to calibrate encoders and choose the alignment reference, and how to perform maintenance tasks.

Manage Page Overview

The sensor's system and maintenance tasks are performed on the **Manage** page.



	Element	Description
1	Sensor System	Contains sensor information and the autostart setting. See *Sensor System* on the next page.
2	Layout	Contains settings for configuring dual-sensor system layouts.
3	Networking	Contains settings for configuring the network. See *Networking* on the next page.
4	Motion and Alignment	Contains settings to configure the encoder. See *Motion and Alignment* on page 118.
5	Jobs	Lets you manage jobs stored on the sensor. See *Jobs* on page 120.
6	Security	Lets you change passwords. See *Security* on page 122.
7	Maintenance	Lets you upgrade firmware, create/restore backups, and reset sensors. See *Maintenance* on page 123.

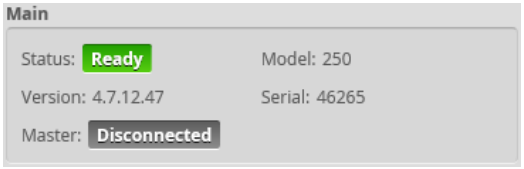
	Element	Description
8	Support	Lets you open an HTML version or download a PDF version of the manual, download the SDK, or save a support file. Also provides device information. See *Support* on page 126

Sensor System

The following sections describe the **Sensor System** category on the **Manage** page.

Sensor Autostart

With the **Autostart** setting enabled, scanning and measurements begin automatically when the sensor is powered on. Autostart must be enabled if the sensor will be used without being connected to a computer.



To enable/disable Autostart:

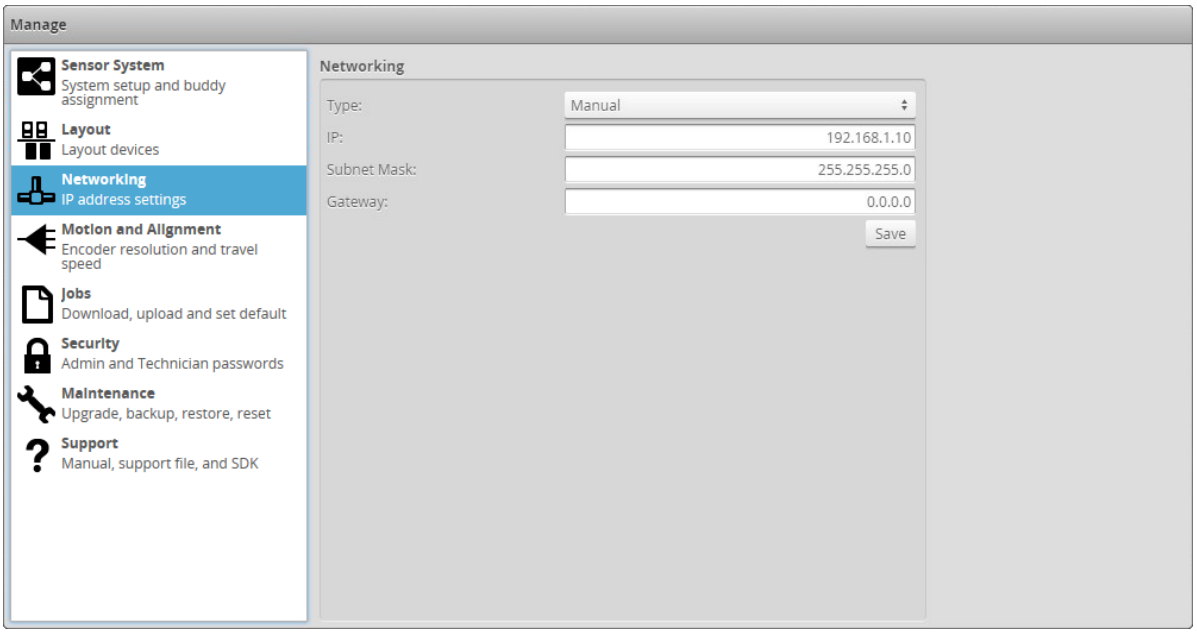
1. Go to the **Manage** page and click on the **Sensor System** category.
2. Check/uncheck the **Autostart** option in the **Main** section.

Layout

Gocator 200 sensors do not support configuring multi-sensor layouts from the Gocator web interface. Use GoWebScanSDK instead. For more information, see *GoSDK and GoWebScanSDK* on page 61.

Networking

The **Networking** category on the **Manage** page provides network settings. Settings must be configured to match the network to which the sensors are connected.

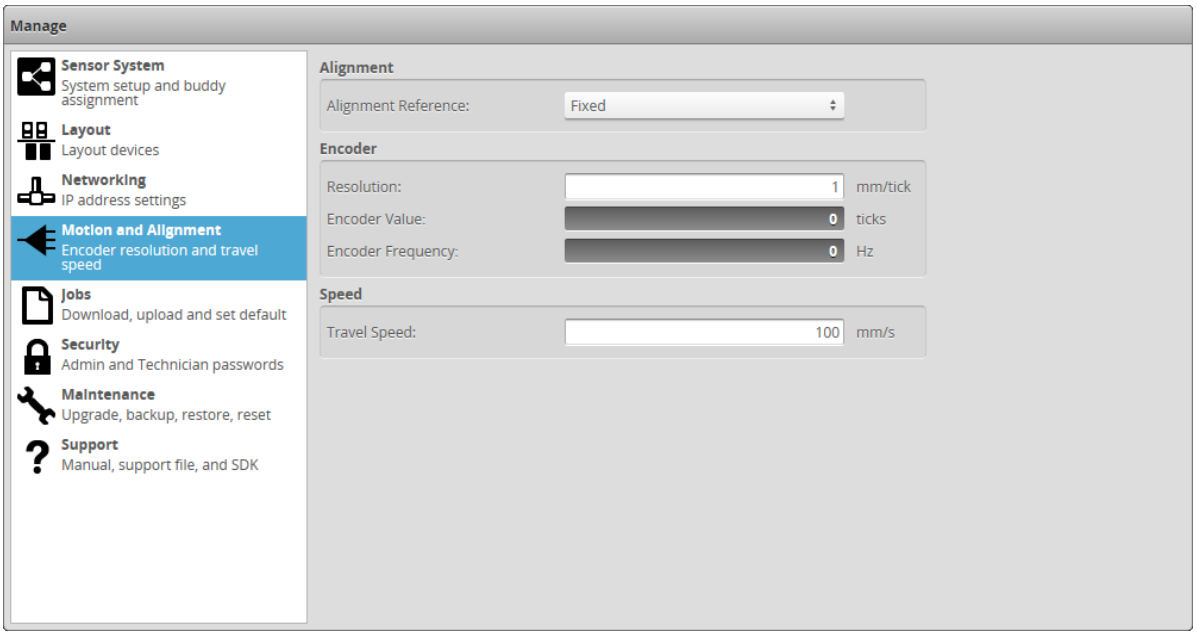


To configure the network settings:

1. Go to the **Manage** page.
2. In the **Networking** category, specify the Type, 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 on the **Save** button.
   You will be prompted to confirm your selection.

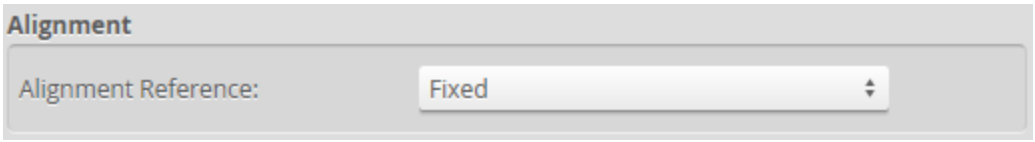
Motion and Alignment

The **Motion and Alignment** category on the **Manage** page lets you configure alignment reference, encoder resolution, and travel speed, and confirm that encoder signals are being received by the sensor.



Alignment Reference

The **Alignment Reference** setting can have one of two values: **Fixed** or **Dynamic**.



Setting	Description
Fixed	A single, global alignment is used for all jobs. This is typically used when the sensor mounting is constant over time and between scans, for example, when the sensor is mounted in a permanent position over a conveyor belt.
Dynamic	A separate alignment is used for each job. This is typically used when the sensor's position relative to the object scanned is always changing, for example, when the sensor is mounted on a robot arm moving to different scanning locations.

To configure alignment reference:

1. Go to the **Manage** page and click on the **Motion and Alignment** category.
2. In the Alignment section, choose **Fixed** or **Dynamic** in the **Alignment Reference** drop-down.

Encoder Resolution

You can manually enter the encoder resolution in the **Resolution** setting , or it can be automatically set by performing an alignment with **Type** set to **Moving**. Establishing the correct encoder resolution is required for correct scaling of the scan of the target object in the direction of travel.

Encoder resolution is expressed in millimeters per tick, where one tick corresponds to *one* of the four encoder quadrature signals (A+ / A- / B+ / B-).

> Encoders are normally specified in *pulses* per revolution, where each pulse is made up of the four quadrature *signals* (A+ / A- / B+ / B-). Because the sensor reads each of the four quadrature signals, you should choose an encoder accordingly, given the resolution required for your application.

To configure encoder resolution:

1. Go to the **Manage** page and click on the **Motion and Alignment** category.
2. In the **Encoder** section, enter a value in the **Resolution** field.

Encoder Value and Frequency

The encoder value and frequency are used to confirm the encoder is correctly wired to the sensor and to manually calibrate encoder resolution (that is, by moving the conveyor system a known distance and making a note of the encoder value at the start and end of movement).

Travel Speed

The **Travel Speed** setting is used to correctly scale scans in the direction of travel in systems that lack an encoder but have a conveyor system that is controlled to move at constant speed. Establishing the correct travel speed is required for correct scaling of the scan in the direction of travel.

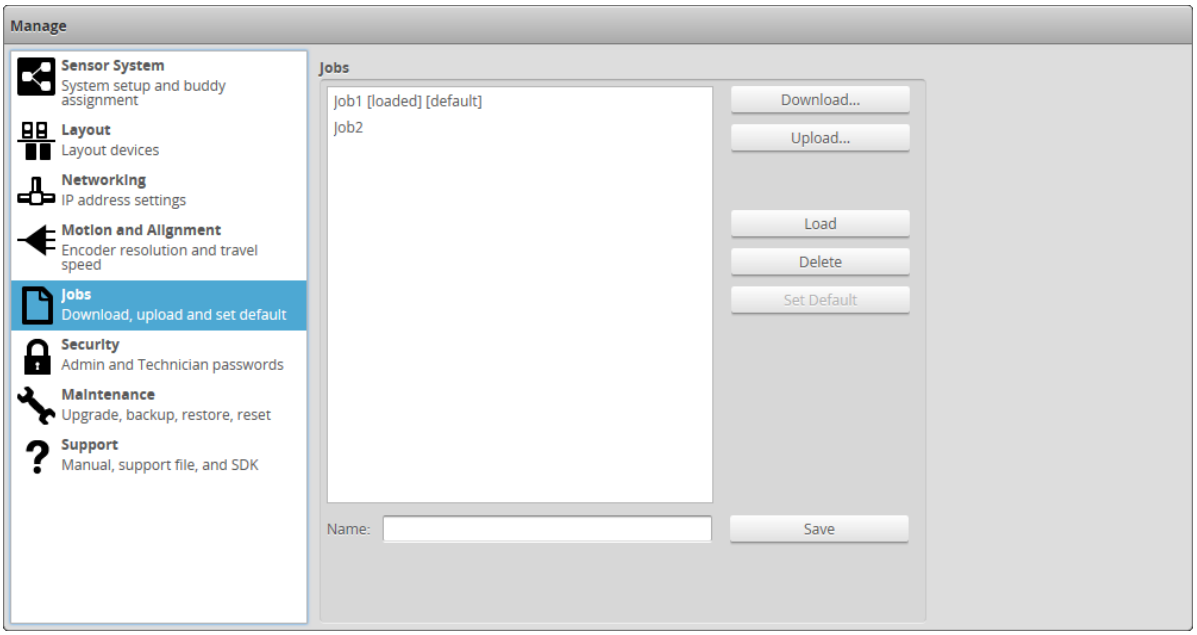
Travel speed is expressed in millimeters per second.

To manually configure travel speed:

1. Go to the **Manage** page and click on the **Motion and Alignment** category.
2. In the **Speed** section, enter a value in the **Travel Speed** field.

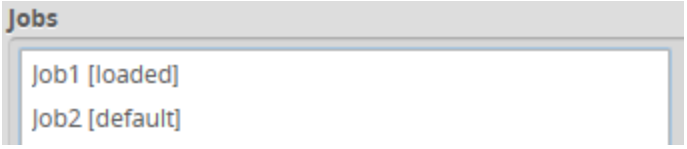
Jobs

The **Jobs** category on the **Manage** page lets you manage the jobs stored on a sensor.

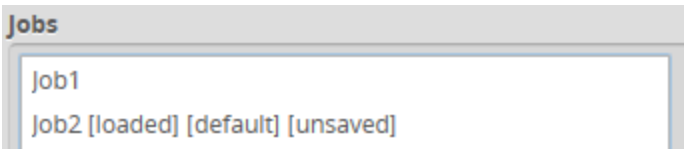


Element	Description
Name field	Used to provide a job name when saving files.
Jobs list	Displays the jobs that are currently saved in the sensor's flash storage.
Save button	Saves current settings to the job using the name in the **Name** field.
Load button	Loads the job that is selected in the job list. Reloading the current job discards any unsaved changes.
Delete button	Deletes the job that is selected in the job list.
Set as Default button	Sets the selected job as the default to be loaded when the sensor starts. When the default job is selected, this button is used to clear the default.
Download... button	Downloads the selected job to the client computer.
Upload... button	Uploads a job from the client computer.

Jobs can be loaded (currently activated in sensor memory) and set as default independently. For example, Job1 could be loaded, while Job2 is set as the default. Default jobs load automatically when a sensor is power cycled or reset.



Unsaved jobs are indicated by "[unsaved]".



To save a job:

1. Go to the **Manage** page and click on the **Jobs** category.
2. Provide a name in the **Name** field.
   To save an existing job under a different name, click on it in the **Jobs** list and then modify it in the **Name** field.
3. Click on the **Save** button or press **Enter**.
   Saving a job automatically sets it as the default, that is, the job loaded when then sensor is restarted.

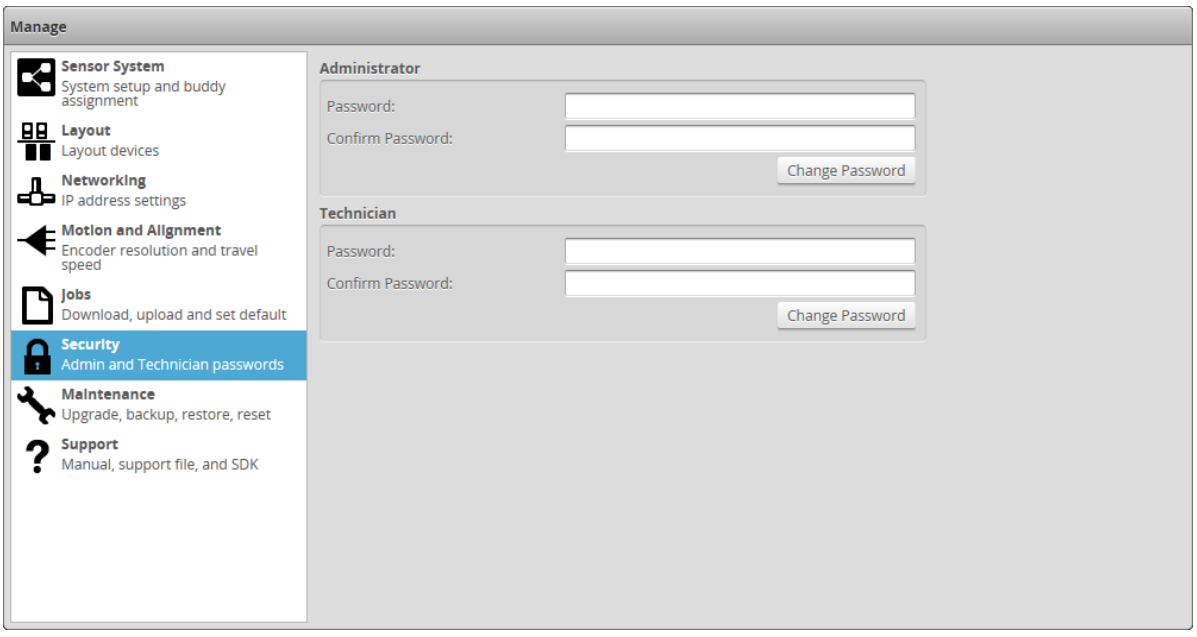
To download, load, or delete a job, or to set one as a default, or clear a default:

1. Go to the **Manage** page and click on the **Jobs** category.
2. Select a job in the **Jobs** list.
3. Click on the appropriate button for the operation.

Security

You can prevent unauthorized access to a sensor by setting passwords. Each sensor has two accounts: Administrator and Technician.

By default, no passwords are set. When you start a sensor, you are prompted for a password only if a password has been set.



Account Types

Account	Description
Administrator	The Administrator account has privileges to use the toolbar (loading and saving jobs, recording and viewing replay data), to view all pages and edit all settings, and to perform setup procedures such as sensor alignment.
Technician	The Technician account has privileges to use the toolbar (loading and saving jobs, recording and viewing replay data), to view the **Dashboard** page, and to start or stop the sensor.

The Administrator and Technician accounts can be assigned unique passwords.

To set or change the password for the Administrator account:

1. Go to the **Manage** page and click on the **Security** category.
2. In the **Administrator** section, enter the Administrator account password and password confirmation.
3. Click **Change Password**.
   The new password will be required the next time that an administrator logs in to the sensor.

To set or change the password for the Technician account:

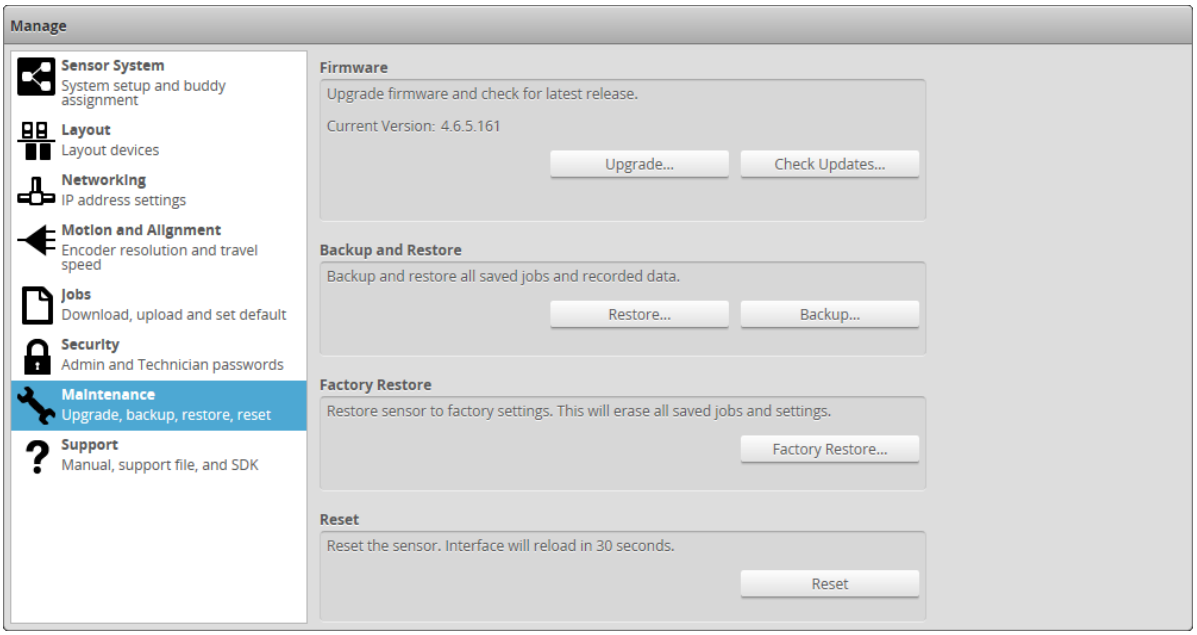
1. Go to the **Manage** page and click on the **Security** category.
2. In the **Technician** section, enter the Technician account password and password confirmation.
3. Click **Change Password**.
   The new password will be required the next time that a technician logs in to the sensor.

If the administrator or technician password is lost, the sensor can be recovered using a special software tool. See *Sensor Discovery Tool* on page 337 for more information.

Maintenance

The **Maintenance** category in the **Manage** page is used to do the following:

- upgrade the firmware and check for firmware updates;
- back up and restore all saved jobs and recorded data;
- restore the sensor to factory defaults;
- reset the sensor.

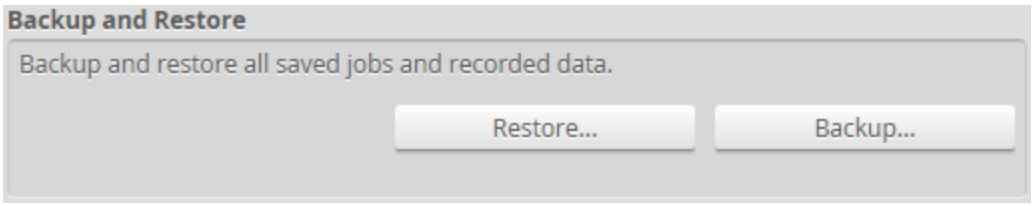


Sensor Backups and Factory Reset

You can create sensor backups, restore from a backup, and restore to factory defaults in the **Maintenance** category.

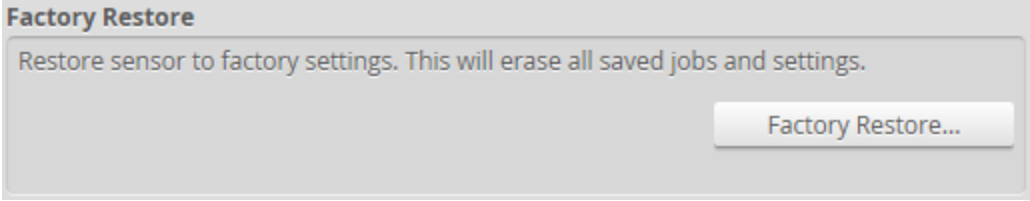
Backup files contain all of the information stored on a sensor, including jobs and alignment.

> An Administrator should create a 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.



To create a backup:

1. Go to the **Manage** page and click on the **Maintenance** category.
2. Click the **Backup...** button under **Backup and Restore**.
3. When you are prompted, save the backup.
   Backups are saved as a single archive that contains all of the files from the sensor.



To restore from a backup:

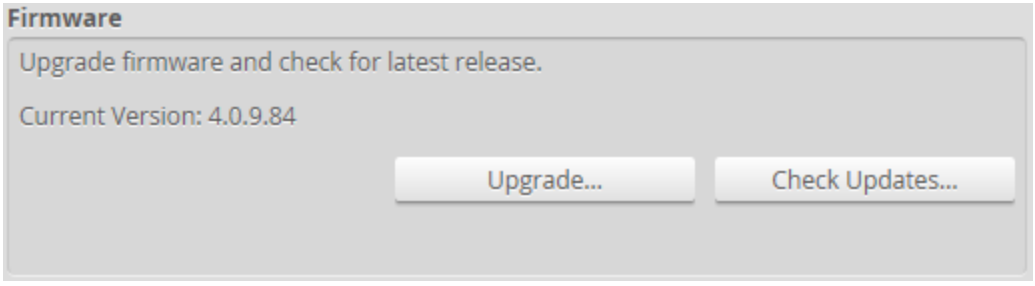
1. Go to the **Manage** page and click on the **Maintenance** category.
2. Click the **Restore...** button under **Backup and Restore**.
3. When you are prompted, select a backup file to restore.
   The backup file is uploaded and then used to restore the sensor. Any files that were on the sensor before the restore operation will be lost.

To restore a sensor to its factory default settings:

1. Go to the **Manage** page and click on **Maintenance**.
2. Consider making a backup.
   Before proceeding, you should perform a backup. Restoring to factory defaults cannot be undone.
3. Click the **Factory Restore...** button under **Factory Restore**.
   You will be prompted whether you want to proceed.

Firmware Upgrade

LMI recommends routinely updating firmware to ensure that sensors always have the latest features and fixes.



To download the latest firmware:

1. Go to the **Manage** page and click on the **Maintenance** category.
2. Click the **Check Updates...** button in the **Firmware** section.
3. Download the latest firmware.
   If a new version of the firmware is available, follow the instructions to download it to the client computer.

If the client computer is not connected to the Internet, firmware can be downloaded and transferred to the client computer by using another computer to download the firmware from LMI's website: *http://www.lmi3D.com/support/downloads*.

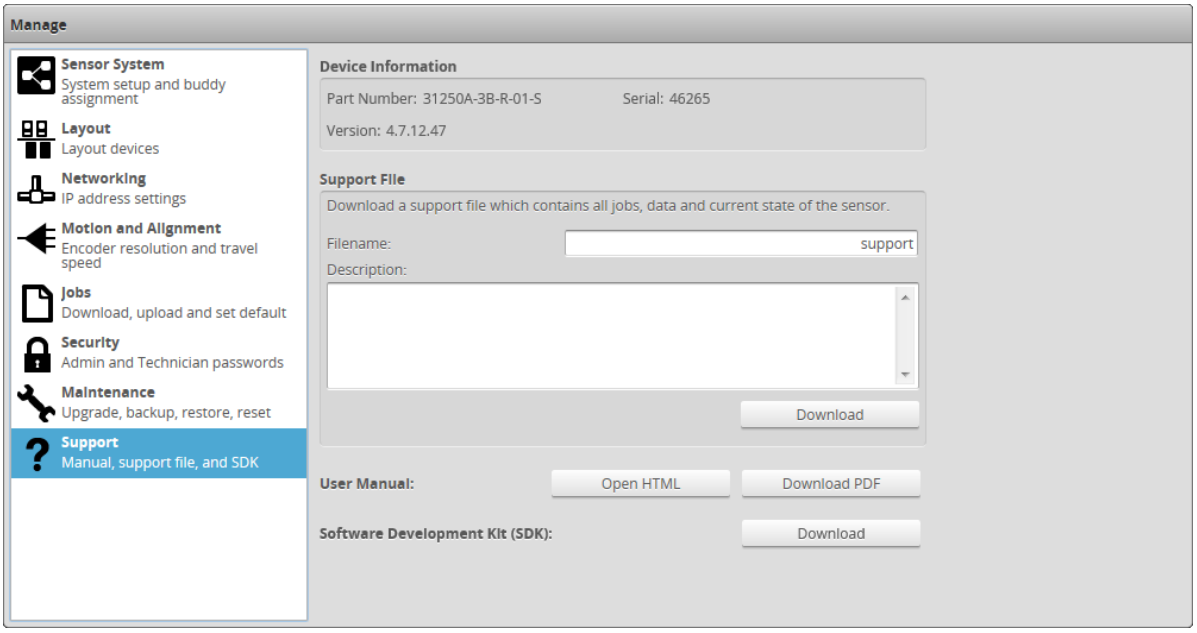
To upgrade the firmware:

1. Go to the **Manage** page and click on the **Maintenance** category.
2. Click the **Upgrade...** button in the **Firmware** section.
3. Locate the firmware file in the **File** dialog and then click open.
4. Wait for the upgrade to complete.
   After the firmware upgrade is complete, the sensor will self-reset. If a buddy has been assigned, it will be upgraded and reset automatically.

Support

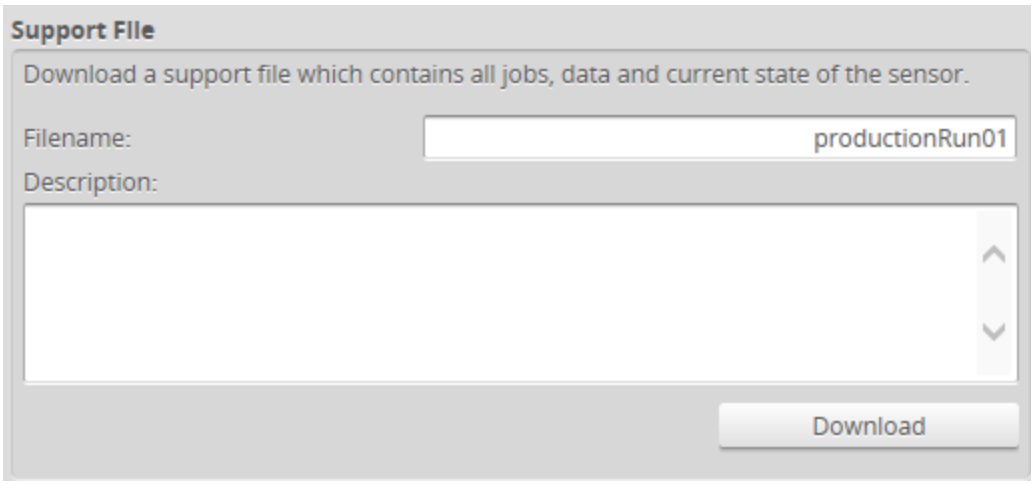
The **Support** category in the **Manage** page is used to do the following:

- Open an HTML version or download a PDF version of the manual.
- Download the SDK.
- Save a support file.
- Get device information.



Support Files

You can download a support file from a sensor and save it on your computer. You can then use the support file to create a scenario in the emulator (for more information on the emulator, see *Gocator Emulator* on page 177). LMI's support staff may also request a support file to help in troubleshooting.



To download a support file:

1. Go to the **Manage** page and click on the **Support** category.
2. In **Filename**, type the name you want to use for the support file.

   When you create a scenario from a support file in the emulator, the filename you provide here is displayed in the emulator's scenario list.

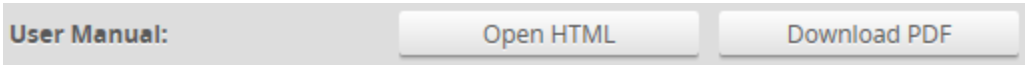
   Support files end with the .gs extension, but you do not need to type the extension in **Filename**.
3. (Optional) In **Description**, type a description of the support file.

   When you create a scenario from a support file in the emulator, the description is displayed below the emulator's scenario list.
4. Click **Download**, and then when prompted, click **Save**.

⚠	Downloading a support file stops the sensor.

Manual Access

You can access the Gocator manuals from within the Web interface.



🗋	You may need to configure your browser to allow pop-ups to open or download the manual.

To access the manuals:

1. Go to the **Manage** page and click on the **Support** category
2. Next to **User Manual**, click one of the following:
   - **Open HTML**: Opens the HTML version of the manual in your default browser.
   - **Download PDF**: Downloads the PDF version of the manual to the client computer.

Software Development Kit

You can download the Gocator SDK from within the Web interface.

To download the SDK:

1. Go to the **Manage** page and click on the **Support** category
2. Next to **Software Development Kit (SDK)**, click **Download**
3. Choose the location for the SDK on the client computer.

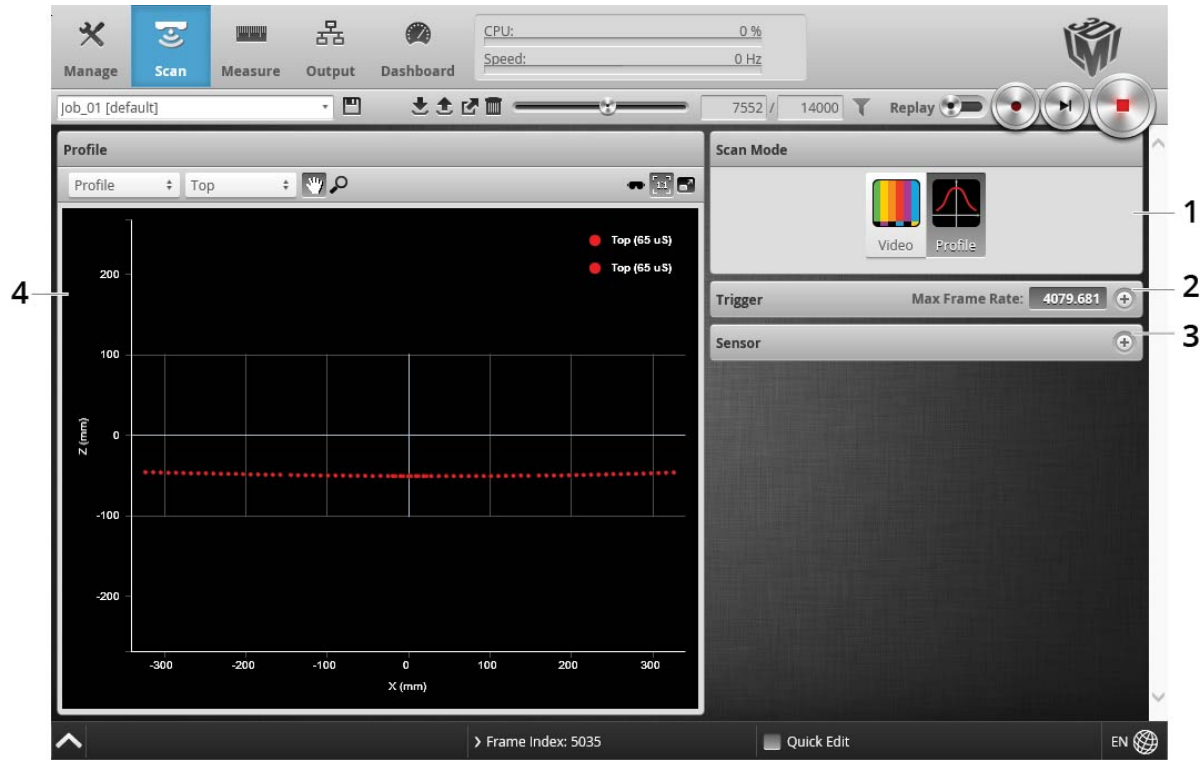
For more information on the SDK, see *GoSDK* on page 64.

Scan Setup and Alignment

The following sections describe the steps to configure sensors for data acquisition using the **Scan** page. Setup and alignment should be performed before adding and configuring measurements or outputs.

Scan Page Overview

The **Scan** page lets you configure sensors.



	Element	Description
1	Scan Mode panel	Contains settings for the current scan mode and other options. See *Scan Modes* on the next page.
2	Trigger panel	Contains trigger source and trigger-related settings. See *Triggers* on the next page.
3	Sensor panel	Contains settings for an individual sensor, such as active area or exposure. See *Sensor* on page 133.
4	Data Viewer	Displays sensor data and adjusts regions of interest. Depending on the current operation mode, the data viewer can display video images or scan data. See *Data Viewer* on page 140.

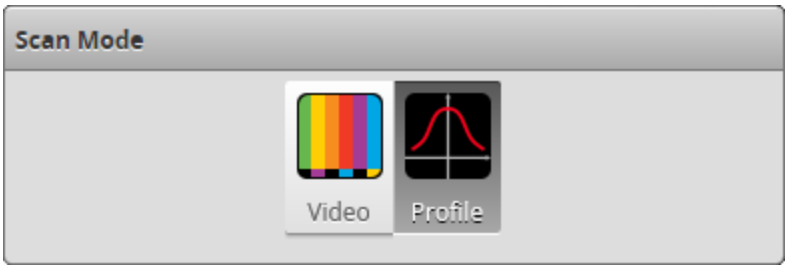
The following table provides quick references for specific goals that you can achieve from the panels in the **Scan** page.

Goal	Reference
Select a trigger source that is appropriate for the application.	Triggers (page 130)
Ensure that camera exposure is appropriate for scan data acquisition.	Exposure (page 136)
Find the right balance between data quality, speed, and CPU utilization.	Active Area (page 133) Exposure (page 136) Job File Structure (page 192)
Specify mounting orientations.	Layout (page 117)

Scan Modes

The sensor web interface supports a video mode and one or more data acquisition modes. The scan mode can be selected in the **Scan Mode** panel.

Gocator 205 only supports Video mode.



Mode and Option	Description
Video	Outputs video images from the sensor. This mode is useful for configuring exposure time and troubleshooting stray light or ambient light problems.
Profile	Outputs profiles and performs profile measurements. Video images are processed internally to produce laser profiles and cross-sectional measurements. This mode is not available on Gocator 205.

Triggers

Gocator 200 sensors do not support direct encoder input.

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

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.

Trigger Source	Description
External Input	A digital input can provide triggers in response to external events (e.g., 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. The **Trigger Drops Indicator** in the **Dashboard** page can be used to check for this condition.
Software	A network command can be used to send a software trigger. See *Protocols* on page 256 for more information.

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

For examples of typical real-world scenarios, see *Trigger Examples* below. For information on the settings used with each trigger source, see *Trigger Settings* on the next page.

Trigger Examples

Example: Encoder + Conveyor
Encoder triggering is used to perform measurements at a uniform spacing.

The speed of the conveyor can vary while the object is being measured; an encoder ensures that the trigger spacing is consistent, independent of conveyor speed.

Example: Time + Conveyor
Time triggering can be used instead of encoder triggering to perform measurements at a fixed frequency.

Spacing will be non-uniform if the speed of the conveyor varies while the object is being measured.

It is strongly recommended to use an encoder with transport-based systems due to the difficulty in maintaining constant transport velocity.

Example: External Input + Conveyor
External input triggering can be used to produce a snapshot for measurement.

For example, a photocell can be connected as an external input to generate a trigger pulse when a target object has moved into position.

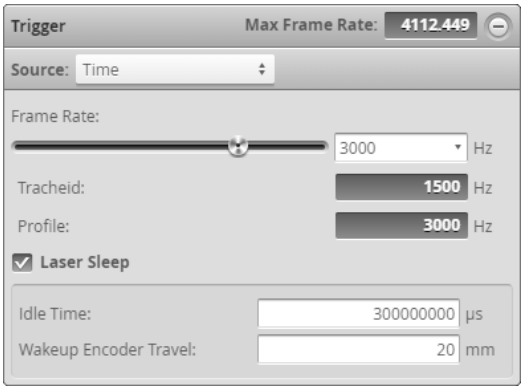
An external input can also be used to gate the trigger signals when time or encoder triggering is used. For example, a photocell could generate a series of trigger pulses as long as there is a target in position.

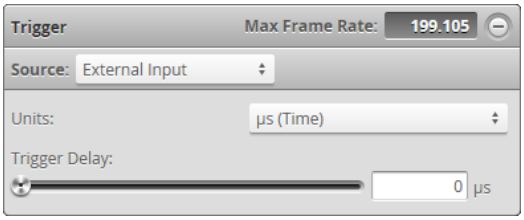
Example: Software Trigger + Robot Arm
Software triggering can be used to produce a snapshot for measurement.

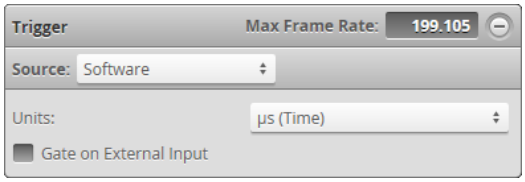
A software trigger can be used in systems that use external software to control the activities of system components.

Trigger Settings

The trigger source is selected using the **Trigger** panel in the **Scan** page.







After specifying a trigger source, the **Trigger** panel shows the parameters that can be configured.

Parameter	Trigger Source	Description
Source	All	Selects the trigger source (**Time**, **Encoder**, **External Input**, or **Software**).
Frame Rate	Time	Controls the frame rate. Select **Max Speed** from the drop-down to lock to the maximum frame rate. Fractional values are supported. For example, 0.1 can be entered to run at 1 frame every 10 seconds.
Gate on External Input	Time, Encoder	External input can be used to enable or disable data acquisition in a sensor. When this option is enabled, the sensor will respond to time or encoder triggers only when the external input is asserted.
Units	External Input, Software	Specifies whether the trigger delay, output delay, and output scheduled command operate in the time or the encoder domain.

Parameter	Trigger Source	Description
		The unit is implicitly set to microseconds with Time trigger source. The unit is implicitly set to millimeters with Encoder trigger source.
Trigger Delay	External Input	Controls the amount of time or the distance the sensor waits before producing a frame after the external input is activated. This is used to compensate for the positional difference between the source of the external input trigger (e.g., photocells) and the sensor. Trigger delay is only supported in single exposure mode; for details, see *Exposure* on page 136.
Laser Sleep	Time	Determines whether the laser automatically shuts off after a specified amount of time (Idle Time).
Idle Time	Time	The amount of idle time before the laser shuts off automatically.
Wakeup Encoder Travel	Time	The amount of encoder travel required to wake up the laser after it has shut off by Laser Sleep.

To configure the trigger source:

1. Go to the **Scan** page.
2. Expand the **Trigger** panel by clicking on the panel header.
3. Select the trigger source from the drop-down.
4. Configure the settings.
   See the trigger parameters above for more information.
5. Save the job in the **Toolbar** by clicking the **Save** button.

Maximum Encoder Rate

For sensors connected through a Master 400 or higher, with the encoder signal supplied to the Master, the maximum rate is about 300 kHz.

Sensor

The following sections describe the settings that are configured in the **Sensor** panel on the **Scan** page.

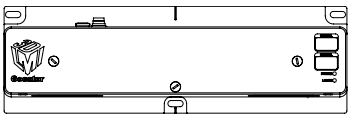
Active Area

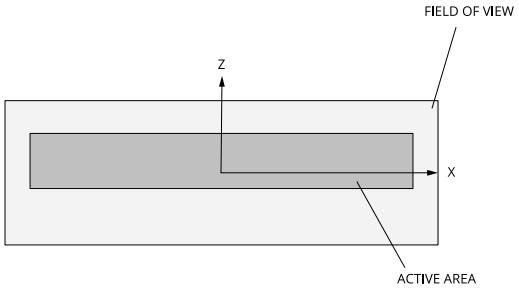
Active area refers to the region within the sensor's maximum field of view that is used for data acquisition.

> Active area cannot be configured on Gocator 205.

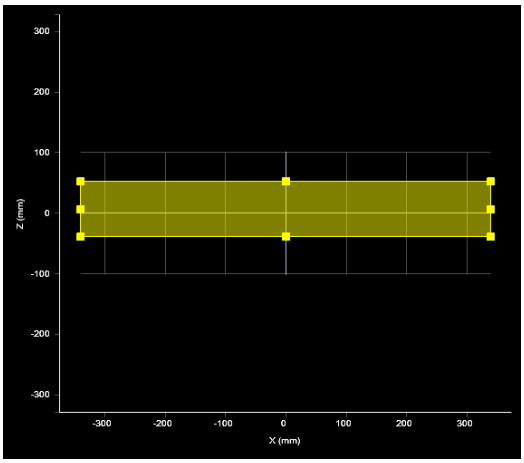
> Currently, applications created with GoWebScanSDK, ignore active area settings on individual sensors.

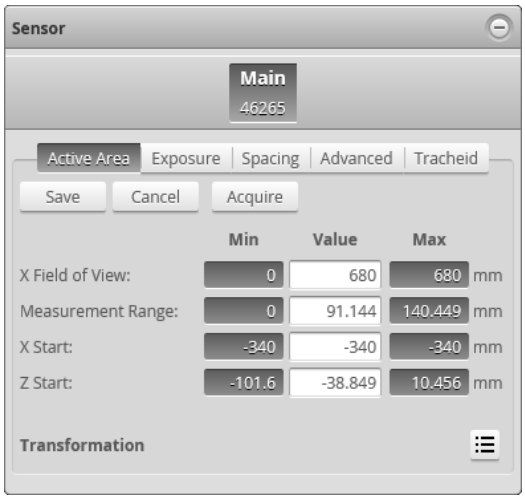
By default, the active area covers the sensor's entire field of view. By reducing the active area, the sensor can operate at higher speeds. You can also reduce the active area to exclude areas that are affected by ambient light.





Active area is set in the **Active Area** tab on the **Sensor** panel.





To set the active area:

1. Go to the **Scan** page.
2. Choose a mode other than Video mode.
3. Expand the **Sensor** panel by clicking on the panel header or the ⊕ button.
4. Click on the **Active Area** tab.

5. Click **Select**.
6. Click **Acquire** to see a scan while setting the active area.
   Acquiring a scan while setting the active area can help you determine where to size and place the active area.
   Gocator 2x0 multi-point scanners do not support this function.
7. Set the active area.
   Adjust the active area graphically in the data viewer or enter the values manually in the fields.
8. Click the **Save** button in the **Sensor** panel.
   Click the **Cancel** button to cancel setting the active area.
9. Save the job in the **Toolbar** by clicking the **Save** button.

Transformations

The transformation settings determine how data is converted from sensor coordinates to system coordinates (for an overview on coordinate systems, see *Coordinate Systems* on page 60).

Gocator 200 series multi-point sensors only support X Offset, Z Offset, and Angle Y transformations. Setting the other transformations has no effect.

Gocator 205 does not support transformations.

Parameter	Description
X Offset	Specifies the shift along the X axis.
Z Offset	Specifies the shift along the Z axis. A positive value shifts the data toward the sensor.
Angle Y	Specifies the tilt around the Y axis.

When applying the transformations, the object is first rotated around the Y axis before the X and Z offsets.

To configure transformation settings:

1. Go to the **Scan** page.
2. Choose a mode other than Video mode in the **Scan Mode** panel.

If Video mode is selected, you will not be able to change the settings.

3. Expand the **Sensor** panel by clicking on the panel header.
4. Expand the Transformations area by clicking on the expand button ☰.
   See the table above for more information.
5. Set the parameter values.
   See the table above for more information.
6. Save the job in the **Toolbar** by clicking the **Save** button 💾.
7. Check that the transformation settings are applied correctly after the sensor is restarted.

Exposure

Exposure determines the duration of camera and light-source on-time. Longer exposures can be helpful to detect light on dark or distant surfaces, but increasing exposure time decreases the maximum speed.
Gocator 200 series sensors only support single exposure mode.

> GoWebScanSDK overrides this setting.

Exposure Mode	Description
Single	Uses a single exposure for all objects. Used when the surface is uniform and is the same for all targets.

For more information on the different types of exposure options, see the sections below.

Video mode lets you see how the light appears on the camera and identify any stray light or ambient light problems. When exposure is tuned correctly, the projected light should be clearly visible along the entire length of the viewer. If it is too dim, increase the exposure value; if it is too bright decrease exposure value.

Single Exposure

The sensor uses a fixed exposure in every scan. Single exposure is used when the target surface is uniform and is the same for all targets.

To enable single exposure:

1. Place a representative target in view of the sensor.
   The target surface should be similar to the material that will normally be measured.
2. Go to the **Scan** page.
3. Expand the **Sensor** panel by clicking on the panel header or the ⊕ button.
4. Click the **Exposure** tab.
5. Select **Single** from the **Exposure Mode** drop-down.
6. Edit the exposure setting by using the slider or by manually entering a value.
   You can automatically tune the exposure by pressing the **Auto Set** button, which causes the sensor to turn on and tune the exposure time.

7. Run the sensor and check that laser profiling is satisfactory.

   If is not satisfactory, adjust the exposure values manually. Switch to **Video** mode to use video to help tune the exposure; see *Exposure* on the previous page for details.

Spacing

The **Spacing** tab lets you configure settings related to spacing (sub-sampling).

> Currently, applications created with GoWebScanSDK, ignore sub-sampling settings set on individual sensors. GoWebScanSDK-based applications use full spacing (the "1" option in the interface) for all data.

Sub-Sampling

Sub-sampling reduces the number of camera columns or rows that are used for laser profiling, reducing the resolution. Reducing the resolution can increase speed or reduce CPU usage while maintaining the sensor's field of view. Sub-sampling can be set independently for the X axis and Z axis.

> For typical wood applications, leave the values at the default of 1.

The **X** sub-sampling setting is used to decrease the profile's X resolution to decrease sensor CPU usage. The **X** setting works by reducing the number of image columns used for laser profiling.

The **Z** sub-sampling setting is used to decrease the profile's Z resolution to increase speed. The **Z** setting works by reducing the number of image rows used for laser profiling.

Sub-sampling values are expressed as fractions in the Web interface. For example, an X sub-sampling value of 1/2 indicates that every second camera column will be used for laser profiling.

> The **CPU Load** bar at the top of the interface displays how much the CPU is being used.

> Both the X and the Z sub-sampling settings must be decreased to increase speed.

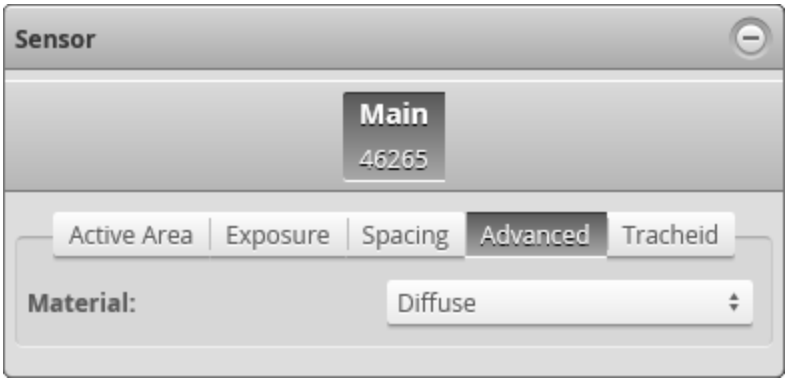
To configure X or Z sub-sampling:

1. Go to the **Scan** page.
2. Expand the **Sensor** panel by clicking on the panel header or the ⊕ button.
3. Click the button corresponding to the sensor you want to configure.
   The button is labeled **Top**, **Bottom**, **Top-Left**, or **Top-Right**, depending on the system.
   X and Z sub-sampling is configured separately for each sensor.
4. Click the **Spacing** tab.
5. Select an X or Z sub-sampling value.
6. Save the job in the **Toolbar** by clicking the **Save** button 💾.
7. Check that laser profiling is satisfactory.

Advanced

The **Advanced** tab contains settings to configure material characteristics.

> Currently, applications created with GoWebScanSDK, ignore material settings set on individual sensors.



To configure advanced settings:

1. Go to the **Scan** page.
2. Switch to Video mode.
   Using Video mode while configuring the settings lets you evaluate their impact.
3. Expand the **Sensor** panel by clicking on the panel header or the ⊕ button.
4. Click on the **Advanced** tab.
5. Configure material characteristics and camera gain.
   For more information, see *Material* on the next page and *Material Settings* on the next page.
6. Save the job in the **Toolbar** by clicking the **Save** button 💾.
7. Check that scan data is satisfactory.

Material

Data acquisition can be configured to suit different types of target materials. For many targets, changing the setting is not necessary, but it can make a great difference with others.

You can select preset material types in the **Materials** setting under the **Advanced** tab. The **Diffuse** material option is suitable for most materials.

When **Materials** is set to **Custom**, you can set camera gain. For more information, see *Material Settings* below.

Material Settings

You can set camera gain to improve data acquisition.

Setting	Description
Camera Gain	
	Digital camera gain can be used when the application is severely exposure limited, yet dynamic range is not a critical factor.

Tracheid

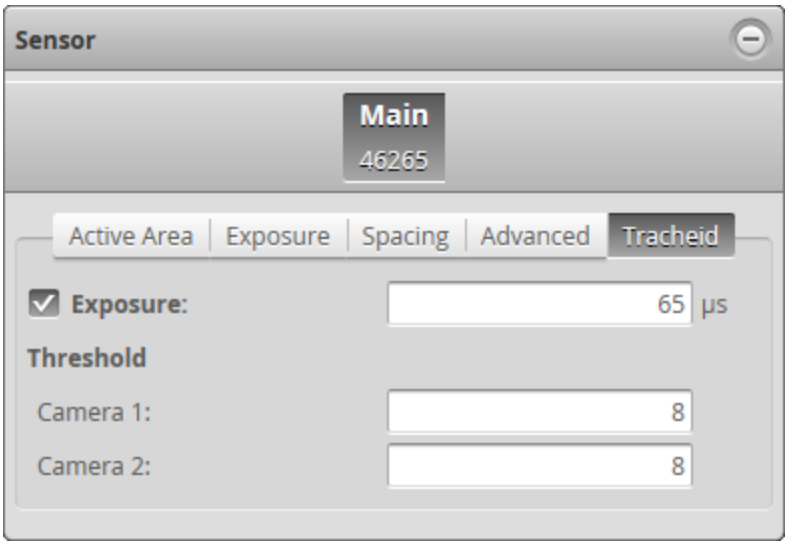
The tracheid settings are used when measuring tracheid cells. Tracheid is only available on Gocator 250 sensors.

> If you wish to configure tracheid threshold settings, you *must* set them on each sensor, ideally using GoSDK.

By default—when no separate tracheid exposure is set—tracheid data is captured at the same time as the profile data. Due to processing constraints, the sensor reports tracheid data at half the rate of profile data. For example, if the sensor reports a frame rate of 4 kHz, it captures profile data at 4 kHz and every other profile is also used to calculate tracheid data (2 kHz).

When you enable a separate tracheid exposure, tracheid data is captured during a separate exposure. In order to maintain a profile speed of 3 kHz in this mode, the sensor has a maximum frame rate of 4 kHz with 3 frames dedicated to profiles and 1 frame dedicated to tracheid, resulting in 3 kHz profile frame rate and 1 kHz tracheid frame rate. This 3:1 ratio is maintained if a lower frame rate is selected.

> The sensor always outputs tracheid data, even when a separate exposure is not set.



Setting	Description
Exposure	When this option is enabled, sets a separate exposure level to use for detecting tracheids. If this option is not enabled, the exposure level for profile data is used. For more information, see *Exposure* on page 136.
Threshold	Specifies a global modifier for the sensor's tracheid thresholds applied at the raw image leve. Increasing this value raises the threshold and tends to increase the dot independence. This helps increase the accuracy of near-horizontal angle measurements. Increasing this value can cause dots to appear rounder in the aspect ratio output. Decreasing this value lowers the threshold and tends to increase the scatter signal processed, resulting in a stronger aspect ratio. Lowering this value too much begins including more noise in the dot data. Recommended values: -20 to +20 (default is 0). Currently, tracheid thresholds must be set on individual sensors via the web interface, GoSDK, or the Write File Gocator protocol command, rather than via GoWebScanSDK or its Settings.xml file.

To configure tracheid settings:

1. Go to the **Scan** page.
2. Expand the **Sensor** panel by clicking on the panel header or the ⊕ button.
3. Click on the **Tracheid** tab.
4. Save the job in the **Toolbar** by clicking the **Save** button.
5. Check that scan data is satisfactory.

Data Viewer

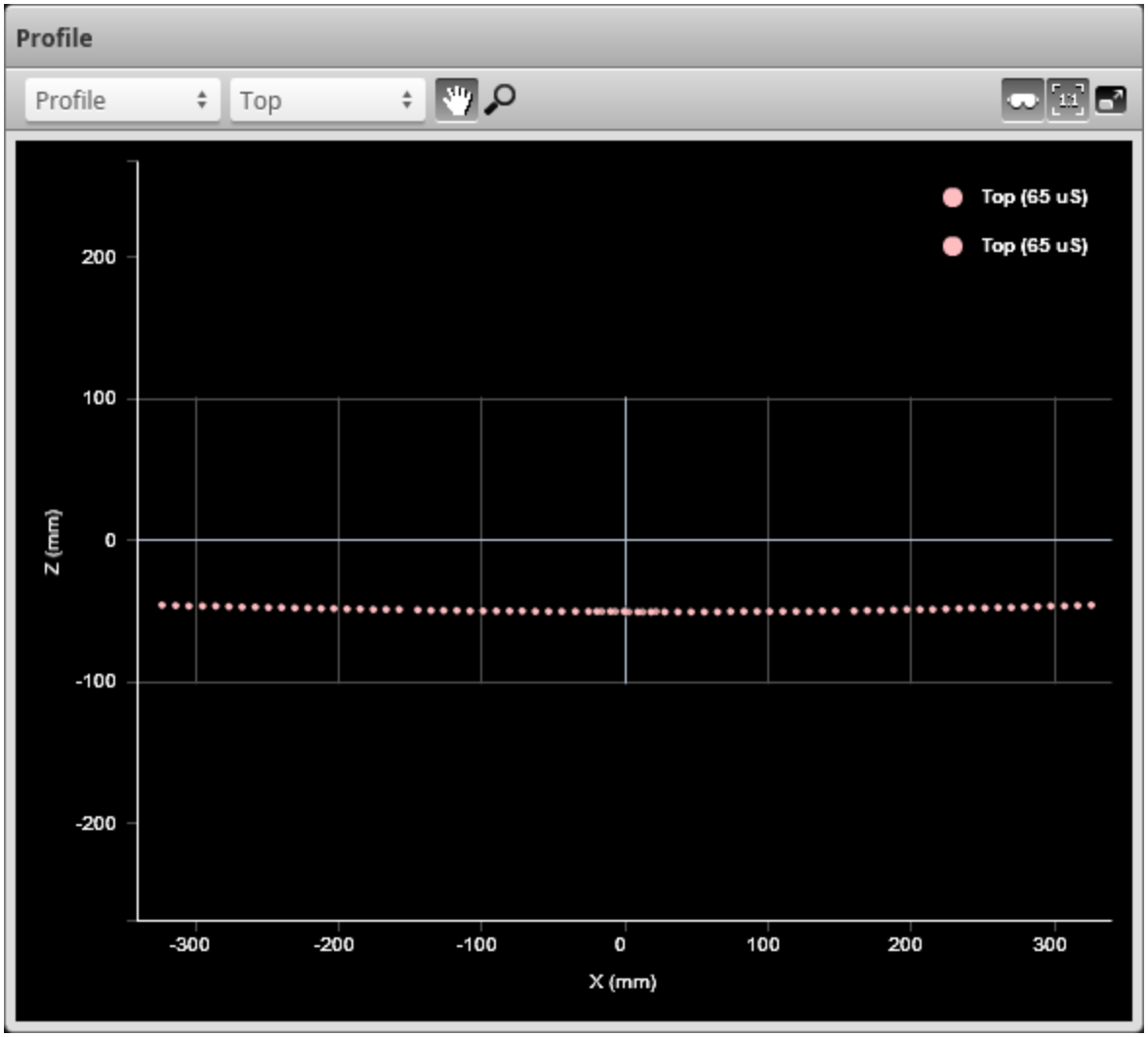
The data viewer can display video images and profiles. It is also used to configure the active area (*Active Area* on page 133) and measurement tools (see *Measurement and Processing* on page 146). The

data viewer changes depending on the current operation mode and the panel that has been selected.

Data Viewer Controls

The data viewer is controlled by mouse clicks and by the buttons on the display toolbar. The mouse wheel can also be used for zooming in and out.

When the sensor displays profiles, a safety goggle mode button ( ) is available above the data viewer. Enabling this mode changes some colors to ensure that profiles are visible in the data viewer when wearing laser safety goggles.



Video Mode

In Video mode, the data viewer displays images directly from the sensor's camera or cameras.

In this mode, you can configure the data viewer to display exposure information that can be useful in properly setting up the system for scanning.

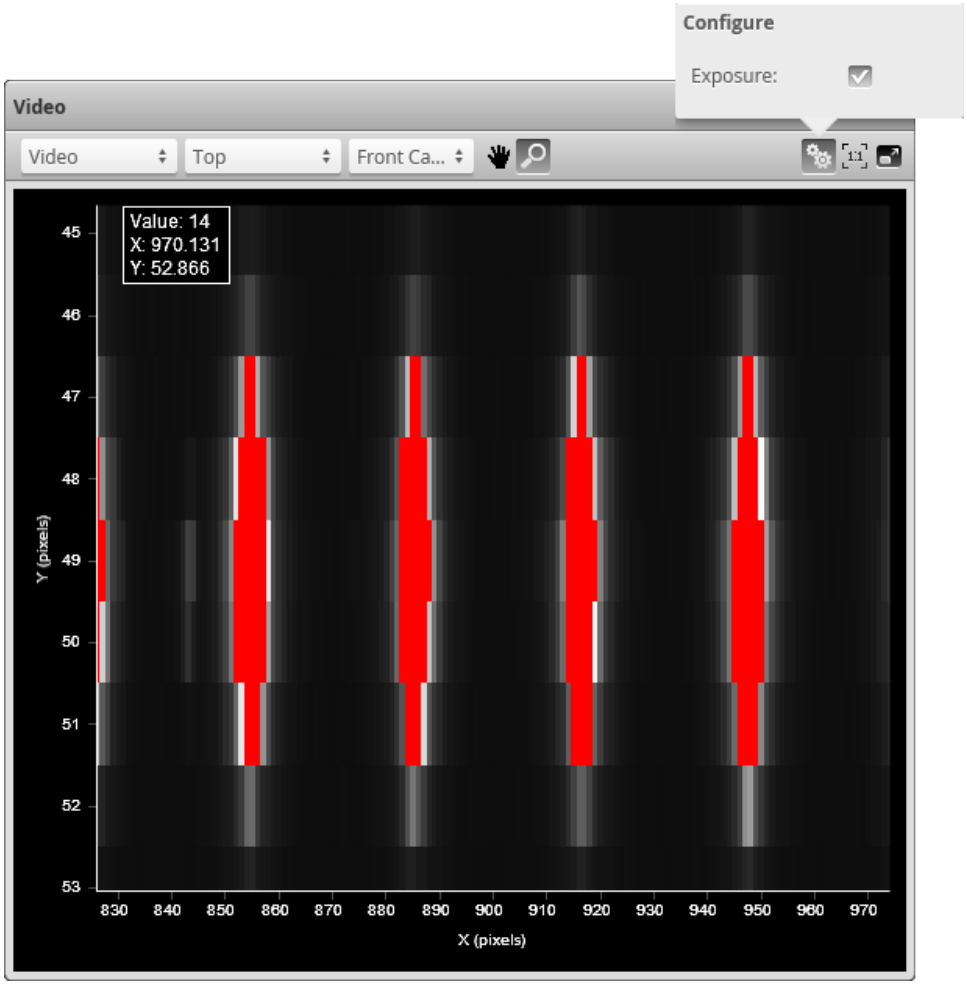
On Gocator 205, you can also toggle between displaying color or black and white images.

Exposure Information

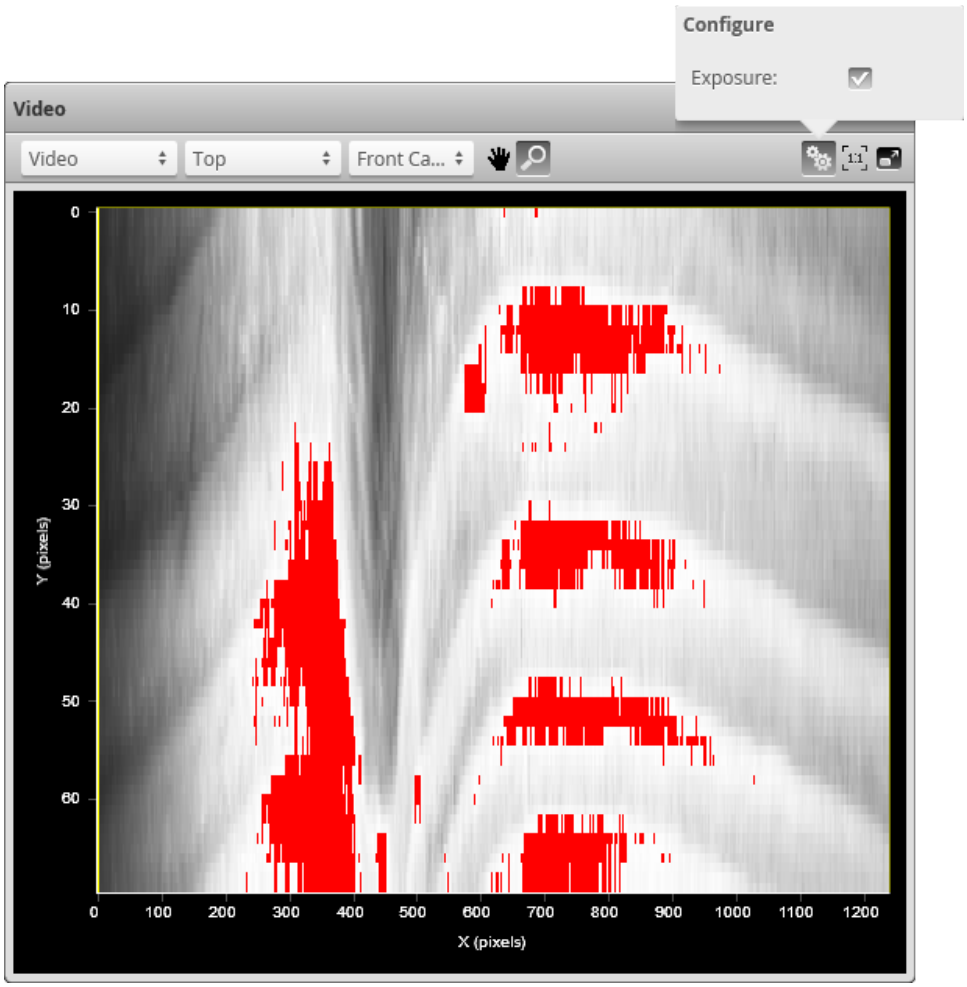
In Video mode, you can display exposure-related information. This information can help you correctly adjust the exposure settings.

Overexposure and Underexposure

You can display a color exposure overlay on the video image to help set the correct exposure.



Exposure information on a Gocator multi-point sensor



Exposure information on a Gocator multi-point sensor

The **Exposure** setting uses the following colors:

- Blue: Indicates background pixels ignored by the sensor.
- Red: Indicates saturated pixels.

Correct tuning of exposure depends on the reflective properties of the target material and on the requirements of the application. Settings should be carefully evaluated for each application.

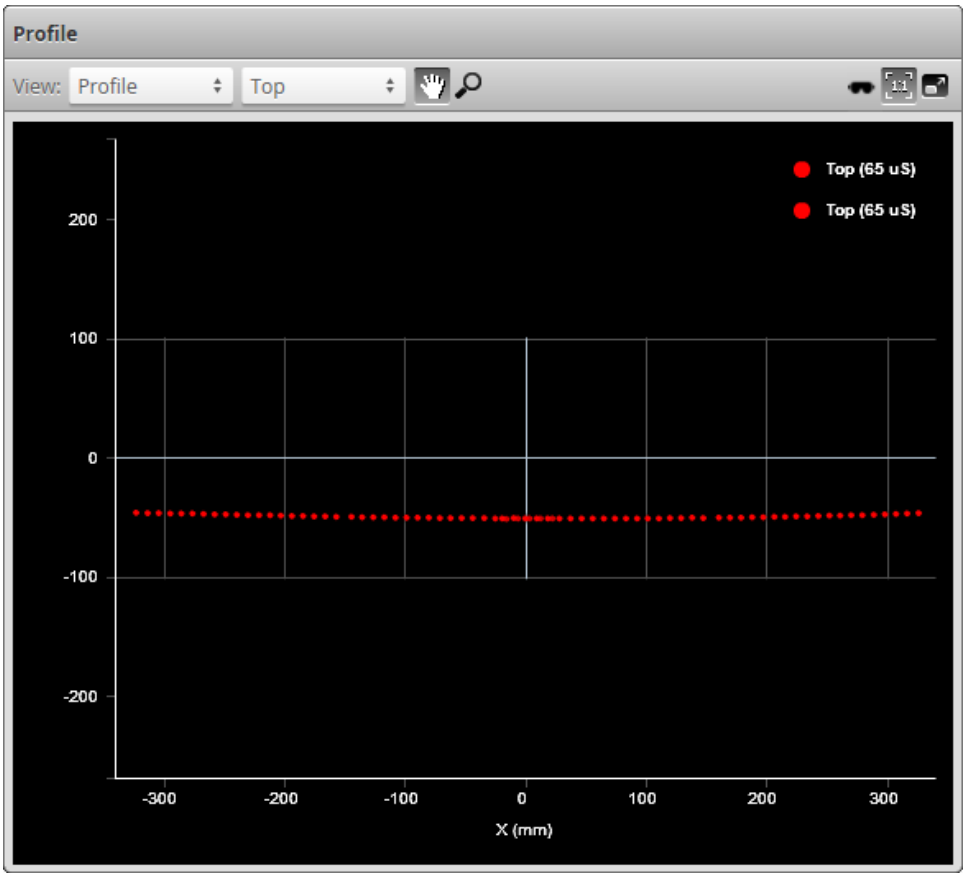
To display an overlay:

1. Go to the **Scan** page and choose **Video** mode in the **Scan Mode** panel.
2. Check **Exposure** at the top of the data viewer.

Profile Mode

When the sensor is in Profile scan mode, the data viewer displays profile plots.

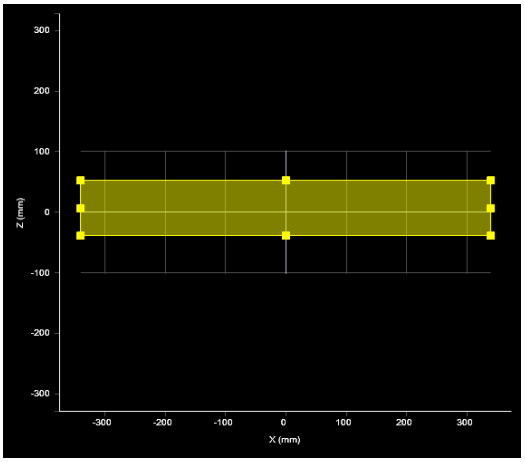
	Profile mode is not available on Gocator 205.



Region Definition

Regions, such as an active area or a measurement region, can be graphically set up using the data viewer.

When the **Scan** page is active, the data viewer can be used to graphically configure the active area. The **Active Area** setting can also be configured manually by entering values into its fields and is found in the **Sensor** panel (see *Sensor* on page 133).



To set up a region of interest:

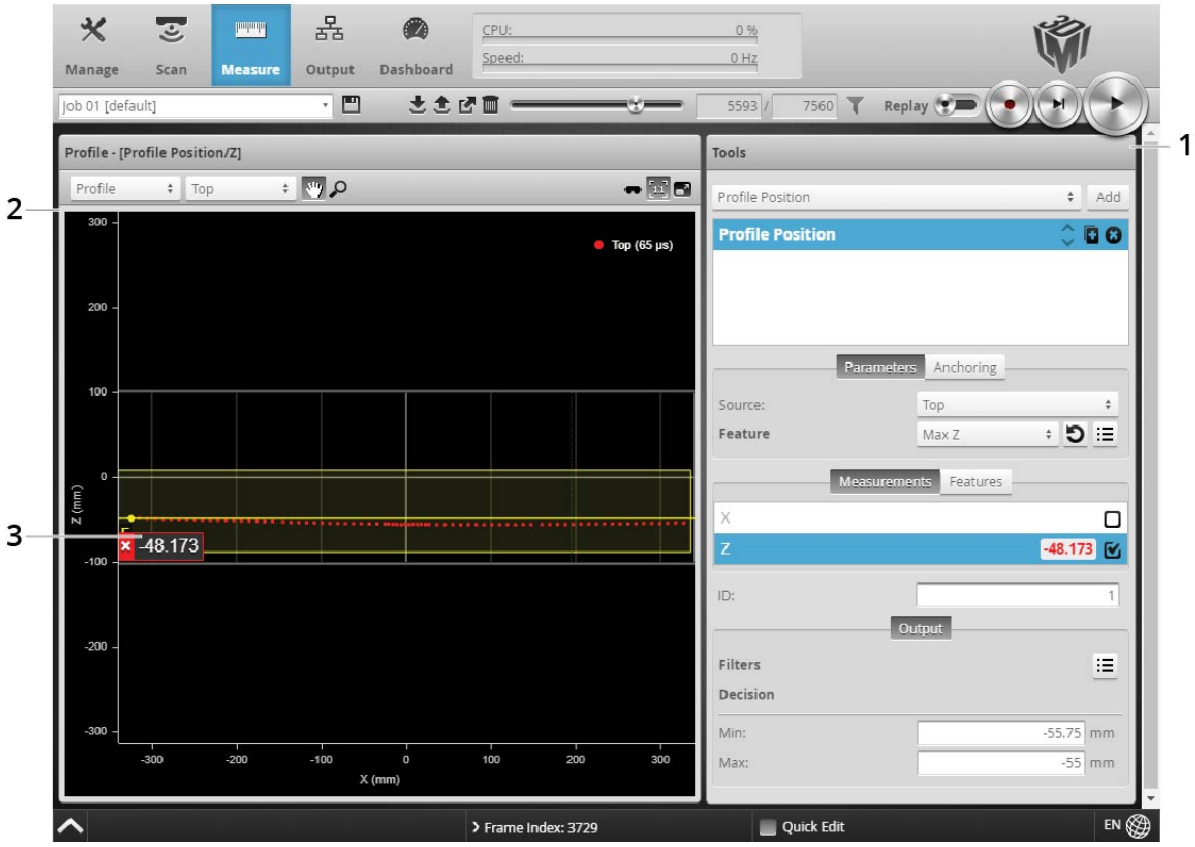
1. Move the mouse cursor to the rectangle.
   The rectangle is automatically displayed when a setup or measurement requires an area to be specified.
2. Drag the rectangle to move it, and use the handles on the rectangle's border to resize it.

Measurement and Processing

Measure Page Overview

Measurement tools are added and configured in the **Measure** page.

The content of the **Tools** panel in the **Measure** page depends on the current scan mode. In Video mode, tools are not available.



	Element	Description
1	Tools panel	Used to add, manage, and configure tools and measurements (see *Tools Panel* on page 148).
2	Data Viewer	Displays video and scan data, sets up tools, and displays result calipers related to the selected measurement. See *Data Viewer* on the next page.
3	Feature Area	Configurable region of interest from which feature points are detected. These feature points are used to calculate the measurements. The number of feature areas displayed depends on which measurement tool is currently selected.

Data Viewer

When the **Measure** page is active, the data viewer can be used to graphically configure measurement regions. Measurement regions can also be configured manually in measurements by entering values into the provided fields (see *Regions* on page 149).

For information on controls in the data viewer, see *Data Viewer Controls* on page 141.

For instructions on how to set up measurement regions graphically, see *Region Definition* on page 144.

Tools Panel

The **Tools** panel lets you add, configure, and manage measurement tools. Tools contain related measurements.

Some settings apply to tools, and therefore to all measurements; these settings are found in the **Parameters** tab below the list of tools. Other settings apply to specific measurements, and are found in a **Parameters** tab below the list of measurements; not all measurements have parameters.

for information on the measurement tools and their settings.

> Tool names in the user interface include the scan mode, but not in the manual. So for example, you will see in the user interface, but simply in the manual.

Adding and Configuring a Measurement Tool

Adding a tool adds all of the tool's measurements to the **Tools** panel. You can then enable and configure the measurements selectively.

To add and configure a tool:

1. Go to the **Scan** page by clicking on the **Scan** icon.
2. Choose mode in the **Scan Mode** panel.
   If one of these modes is not selected, tools will not be available in the **Measure** panel.
3. Go to the **Measure** page by clicking on the **Measure** icon.
4. In the Tools panel, select the tool you want to add from the drop-down list of tools.
5. Click on the **Add** button in the Tools panel.
   The tool and its available measurements are added to the tool list. The tool parameters are listed in the area below the tool list.
6. Select a measurement at the bottom of the tool panel.
7. Set any tool- or measurement-specific settings.
   For tool- and measurement-specific settings, see the topics for the individual tools.
8. Set the **Min** and **Max** decision values.
   For more information on decisions, see *Decisions* on page 152.
9. (Optional) Set one or more filters.
   For more information on filters, see *Filters* on page 153.
10. (Optional) Set up anchoring.
    For more information on anchoring, see *Measurement Anchoring* on page 154.

Source

This setting is always **Top** with G200 scanners.

Regions

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

> You can disable regions entirely and cause the measurement tool uses the entire active area by unchecking the checkbox next to the **Regions** setting.

All tools provide region settings under the upper **Parameters** tab. This region applies to all of a tool's measurements.

Region settings are often found within expandable feature sections in the tool's panel.

To configure regions:

1. Go to the **Measure** page by clicking on the **Measure** icon.

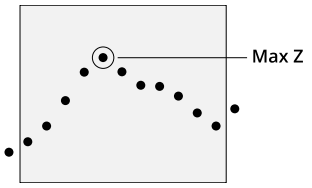
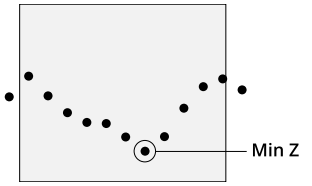
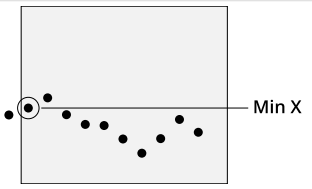
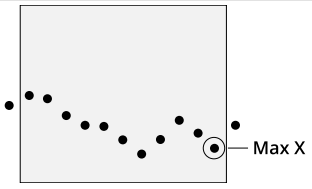
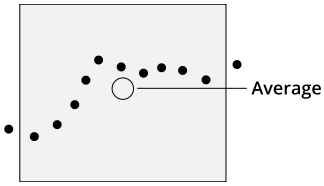
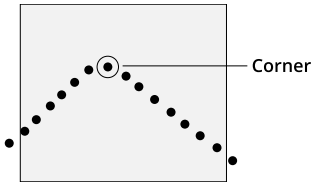
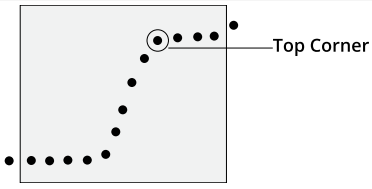
   > The scan mode must be set to the type of measurement you need to configure. Otherwise, the wrong tools, or no tools, will be listed on the **Measure** page.

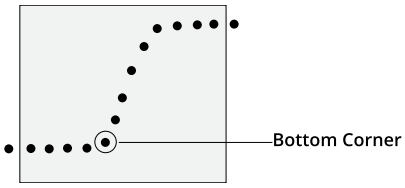
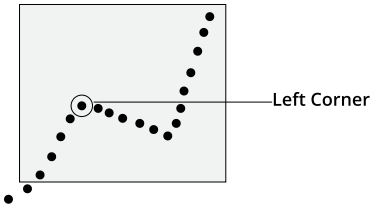
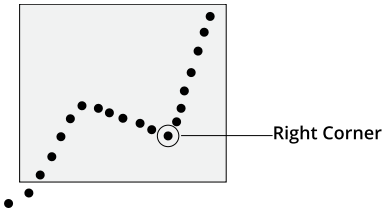
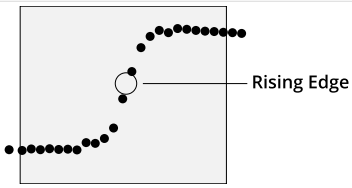
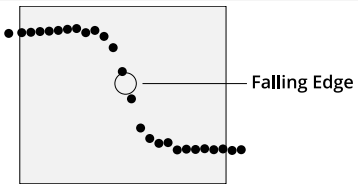
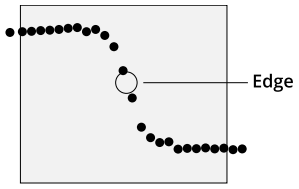
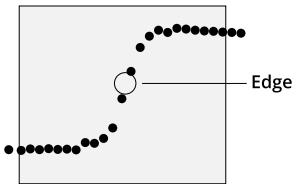
2. In the **Tools** panel, click on a tool in the tool list.
3. Configure the region using the mouse in the data viewer.
   You can also configure regions manually by clicking the expand button (☰) and entering values in the fields. This is useful if you need to set precise values.

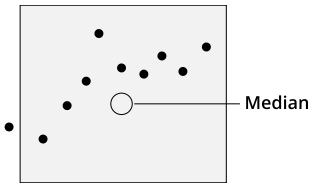
Feature Points

Dimensional and positional measurements detect *feature points* found within the defined measurement region and then compare measurement values taken at the selected point with minimum and maximum thresholds to produce a *decision*. Feature points are selected in one or more **Feature** dropdowns in a tool and are used for all of the tool's measurements.

The following types of points can be identified in a measurement region.

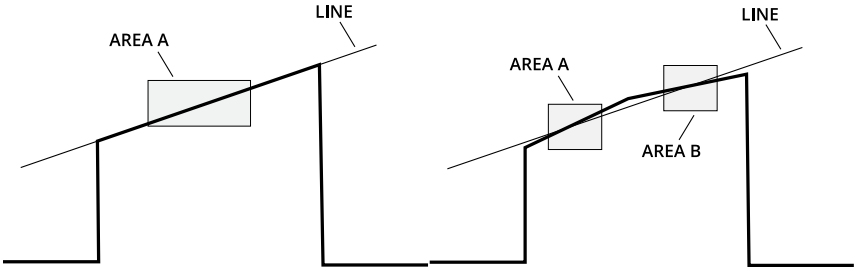
Point Type	Examples
Max Z Finds the point with the maximum Z value in the region of interest.	
Min Z Finds the point with the minimum Z value in the region of interest.	
Min X Finds the point with the minimum X value in the region of interest.	
Max X Finds the point with the maximum X value in the region of interest.	
Average Determines the average location of points in the region of interest.	
Corner Finds a dominant corner in the region of interest, where corner is defined as a change in profile slope.	
Top Corner Finds the top-most corner in the region of interest, where corner is defined as a change in profile shape.	

Point Type	Examples
Bottom Corner Finds the bottom-most corner in the region of interest, where corner is defined as a change in profile shape.	
Left Corner Finds the left-most corner in the region of interest, where corner is defined as a change in profile shape.	
Right Corner Finds the right-most corner in the region of interest, where corner is defined as a change in profile shape.	
Rising Edge Finds a rising edge in the region of interest (moving from left to right).	
Falling Edge Finds a falling edge in the region of interest (moving from left to right).	
Any Edge Finds a rising or falling edge in the region of interest.	 

Point Type	Examples
Median Determines the median location of points in the region of interest.	

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 to generate *pass* / *fail* decisions. The decision state is *pass* if a measurement value is between the minimum and maximum threshold. In the data viewer and next to the measurement, these values are displayed in green. Otherwise, the decision state is *fail*. In the user interface, these values are displayed in red.

All measurements provide decision settings under the **Output** tab.

Along with measurement values, decisions can be sent to external programs and devices.

To configure decisions:

1. Go to the **Measure** page by clicking on the **Measure** icon.

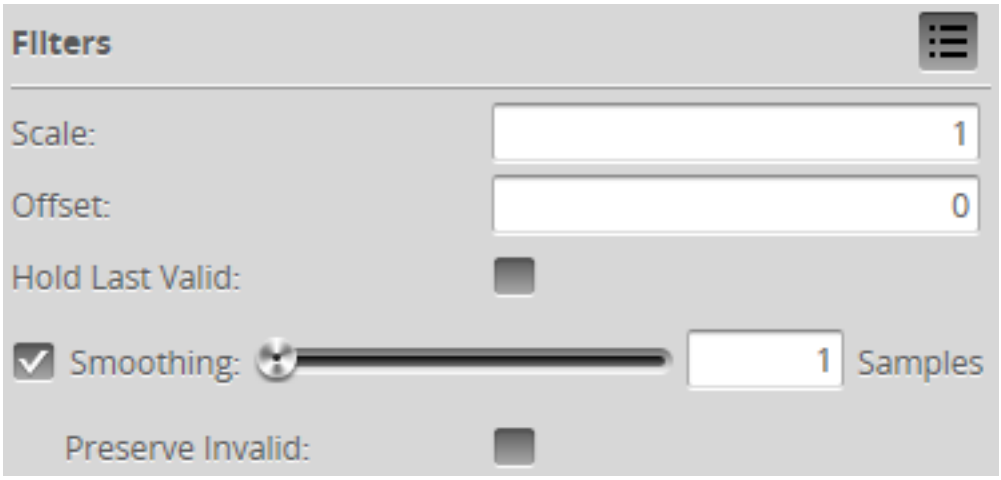
   > The scan mode must be set to the type of measurement you need to configure. Otherwise, the wrong tools, or no tools, will be listed on the **Measure** page.

2. In the **Tools** panel, click on a tool in the tool list.
3. In the measurement list, select a measurement.
   To select a measurement, it must be enabled. See *Enabling and Disabling Measurements* on page 155 for instructions on how to enable a measurement.
4. Click on the **Output** tab.
   For some measurements, only the **Output** tab is displayed.

5. Enter values in the **Min** and **Max** fields.

Filters

Filters can be applied to measurement values before they are output from the Gocator sensors.



All measurements provide filter settings under the **Output** tab. The following settings are available.

Filter	Description
Scale and Offset	The **Scale** and **Offset** settings are applied to a measurement value according to the following formula: Scale * Value + Offset **Scale** and **Offset** can be used to transform the output without the need to write a script. For example, to convert the measurement value from millimeters to thousands of an inch, set **Scale** to 39.37. To convert from radius to diameter, set **Scale** to 2. For more information on scripts, see *Scripts* on page 165.
Hold Last Valid	Holds the last valid value when the measurement is invalid.
Smoothing	Averages the *valid* measurements in the number of preceding frames specified in **Samples**. Use this to reduce the impact of random noise on a measurement's output. If **Hold Last Valid** is enabled, the smoothing filter uses the last valid measurement value until a valid value is encountered.
Preserve Invalid	When enabled, smoothing is only applied to valid measurements and not to invalid results: invalid results are not modified and are sent to output as is. When disabled, smoothing is applied to both valid and invalid results. (This setting is only visible when **Smoothing** is enabled.) If **Hold Last Valid** is enabled, results will always be valid, in which case this setting does nothing.

To configure the filters:

1. Go to the **Measure** page by clicking on the **Measure** icon.

   > The scan mode must be set to the type of measurement you need to configure. Otherwise, the wrong tools, or no tools, will be listed on the **Measure** page.

2. In the **Tools** panel, click on a tool in the tool list.
3. In the measurement list, select a measurement.
   To select a measurement, it must be enabled. See *Enabling and Disabling Measurements* on the next page for instructions on how to enable a measurement.
4. Click on the **Output** tab.
   For some measurements, only the **Output** tab is displayed.
5. Expand the **Filters** panel by clicking on the panel header or the ⊕ button.
6. Configure the filters.
   Refer to the table above for a list of the filters.

Measurement Anchoring

To anchor a tool to a measurement:

1. Place a representative target object in the field of view.
   In Profile mode
   a. Use the **Start** or **Snapshot** button to view live profile data to help position the target.
2. On the **Measure** page, add a suitable tool to act as an anchor.
   A suitable tool is one that returns an X, Y, or Z position as a measurement value.
3. Adjust the anchoring tool's settings and measurement region.
   You can adjust the measurement region graphically in the data viewer or manually by expanding the **Regions** area.
   The position and size of the anchoring tool's measurement regions define the zone within which movement will be tracked.
4. Add the tool that you want to anchor.
   Any tool can be anchored.
5. Adjust the tool and measurement settings, as well as the measurement regions, on a scan of the representative target.
6. Click on the tool's **Anchoring** tab.
7. Choose an anchor from one of the drop-down boxes.
   If the sensor is running, the anchored tool's measurement regions are shown in white to indicate the regions are locked to the anchor. The measurement regions of anchored tools cannot be adjusted. The anchored tool's measurement regions are now tracked and will move with the target's position under the sensor, as long as the anchoring measurement produces a valid measurement value. If the anchoring measurement is invalid, for example, if part moves outside its measurement region, the

anchored tool will not show the measurement regions at all and an “Invalid-Anchor” message will be displayed in the tool panel.

8. Verify that the anchored tool works correctly on other scans of targets in which the part has moved slightly.

To remove an anchor from a tool:

1. Click on the anchored tool's Anchoring tab.
   Select **Disabled** in the X, Y, or Z drop-down.

Enabling and Disabling Measurements

All of the measurements available in a tool are listed in the measurement list in the **Tools** panel after a tool has been added. To configure a measurement, you must enable it.

To enable a measurement:

1. Go to the **Scan** page by clicking on the **Scan** icon.
2. Choose mode in the **Scan Mode** panel.
   If is not selected, tools will not be available in the **Measure** panel.
3. Go to the **Measure** page by clicking on the **Measure** icon.
4. In the measurements list, check the box of the measurement you want to enable.
   The measurement will be enabled and selected. The **Output** tab, which contains output settings will be displayed below the measurements list. For some measurements, a **Parameters** tab, which contains measurement-specific parameters, will also be displayed.

To disable a measurement:

1. Go to the **Scan** page by clicking on the **Scan** icon.
2. Choose mode in the **Scan Mode** panel.
3. Go to the **Measure** page by clicking on the **Measure** icon.
4. In the measurement list, uncheck the box of the measurement you want to disable.
   The measurement will be disabled and the **Output** tab (and the **Parameters** tab if it was available) will be hidden.

Editing Tool, Input, or Output Names

You can change the names of tools you add in Gocator. You can also change the names of their measurements. This allows multiple instances of tools and measurements of the same type to be more easily distinguished in the Gocator web interface. The measurement name is also referenced by the Script tool.

To change a tool or measurement name:

1. Go to the **Scan** page by clicking on the **Scan** icon.

2. Choose mode in the **Scan Mode** panel.
   If is not selected, tools will not be available in the **Measure** panel.
3. Go to the **Measure** page by clicking on the **Measure** icon.
4. Do one of the following:
   - **Tool**: In the tool list, double-click the tool name you want to change
   - **Measurement**: In a tool's measurement list, double-click the measurement name you want to change.
5. Type a new name.
6. Press the Tab or Enter key, or click outside the field.
   The name will be changed.

Changing a Measurement ID

The measurement ID is used to uniquely identify a measurement in the Gocator protocol or in the SDK. The value **must** be unique among all measurements.

To edit a measurement ID:

1. Go to the **Scan** page by clicking on the **Scan** icon.
2. Choose mode in the **Scan Mode** panel.
   If is not selected, tools will not be available in the **Measure** panel.
3. Go to the **Measure** page by clicking on the **Measure** icon.
4. In the measurement list, select a measurement.
   To select a measurement, it must be enabled. See *Enabling and Disabling Measurements* on the previous page for instructions on how to enable a measurement.
5. Click in the ID field.
6. Type a new ID number.
   The value must be unique among all measurements.
7. Press the Tab or Enter key, or click outside the ID field.
   The measurement ID will be changed.

Duplicating a Tool

You can quickly create a copy of a previously added tool in Gocator. All settings of the original are copied. This is useful, for example, when you need almost identical tools with only minor variations, such as different Min and Max values.

To duplicate a tool:

1. Go to the **Scan** page by clicking on the **Scan** icon.
2. Choose mode in the **Scan Mode** panel.
   If one of these modes is not selected, tools will not be available in the **Measure** panel.
3. Go to the **Measure** page by clicking on the **Measure** icon.

4. In the tool list, click the Duplicate button ( ) of the tool you want to duplicate.
   A copy of the tool appears below the original.
5. Configure the copy as desired and rename it if necessary.

   For information on renaming a tool, see *Editing Tool, Input, or Output Names* on page 155.

Removing a Tool

Removing a tool removes all of its associated measurements.

To remove a tool:

1. Go to the **Scan** page by clicking on the **Scan** icon.
2. Choose mode in the **Scan Mode** panel.
   If is not selected, tools will not be available in the **Measure** panel.
3. Go to the **Measure** page by clicking on the **Measure** icon.
4. In the tool list, click on the Duplicate button ( ) of the tool you want to duplicate.
   A copy of the tool appears below the original.

Reordering Tools

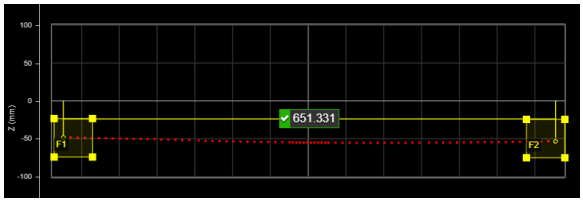
When you add or duplicate a tool, the tool is added to the bottom of the list in the **Tools** panel. You can reorder tools in the web interface to organize tools more logically.

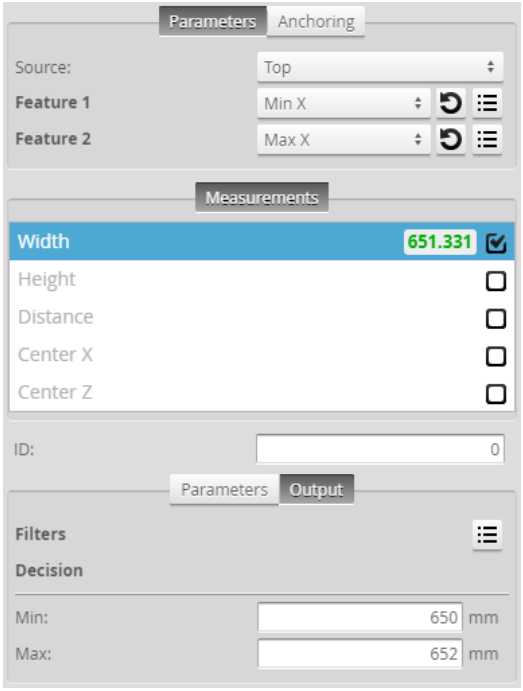
Profile Measurement

This section describes the profile measurement tools available in Gocator sensors.

Dimension

The Dimension tool provides Width, Height, Distance, Center X, and Center Z measurements.

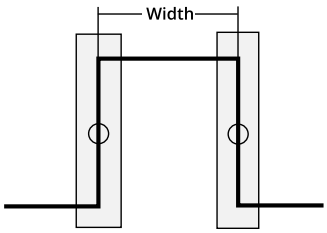
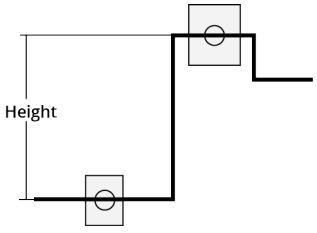


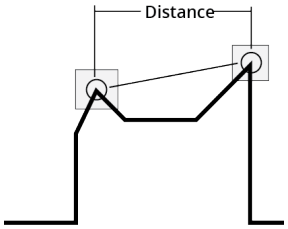
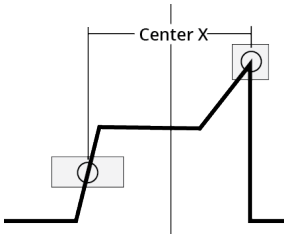
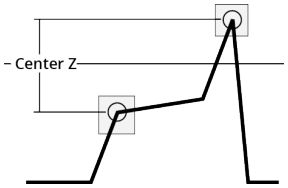


For information on adding, managing, and removing tools and measurements, as well as detailed descriptions of settings common to most tools, see *Tools Panel* on page 148.

Measurements and Settings

Measurements

Measurement	Illustration
Width Determines the difference along the X axis between two feature points. The difference can be calculated as an absolute or signed result. The difference is calculated by: $Width = Feature\ 2_{X\ position} - Feature\ 1_{X\ position}$	
Height Determines the difference along the Z axis between two feature points. The difference can be expressed as an absolute or signed result. The difference is calculated by: $Height = Feature\ 2_{Z\ position} - Feature\ 1_{Z\ position}$	

Measurement	Illustration
Distance Determines the direct, Euclidean distance between two feature points.	
Center X Finds the average location of two features and measures the X axis position of the average location	
Center Z Finds the average location of two features and measures the Z axis position of the average location.	

Parameters

Parameter	Description
Source	The sensor that provides data for the tool's measurements. For more information, see *Source* on page 148.
Feature 1 Feature 2	The **Feature 1** and **Feature 2** settings represent the two features the tool uses to perform measurements. For each, one of the following: • Max Z • Min Z • Max X • Min X • Corner • Average • Rising Edge • Falling Edge • Any Edge • Top Corner • Bottom Corner • Left Corner • Right Corner • Median To set the region of a feature, adjust it graphically in the data viewer, or expand the feature using the expand button (☰) and enter the values in the fields. For more information on regions, see *Regions* on page 149.

Parameter	Description
Absolute *(Width and Height measurements only)*	Determines if the result will be expressed as an absolute or a signed value.
Filters	The filters that are applied to measurement values before they are output. For more information, see *Filters* on page 153.
Decision	The **Max** and **Min** settings define the range that determines whether the measurement tool sends a pass or fail decision to the output. For more information, see *Decisions* on page 152.

Anchoring

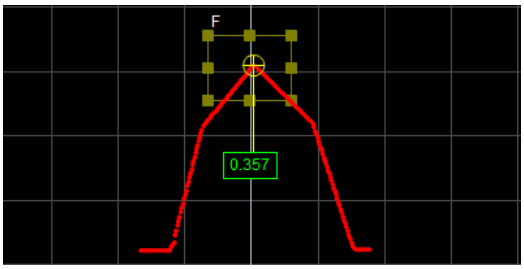
Anchor	Description
X or Z	Lets you choose the X or Z measurement of another tool to use as a positional anchor for this tool.

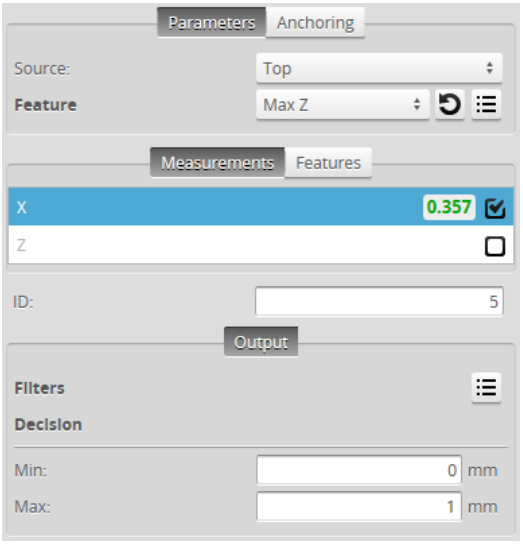
A measurement *must* be enabled in the other tool for it to be available as an anchor. The anchor measurement should also be properly configured before using it as an anchor.

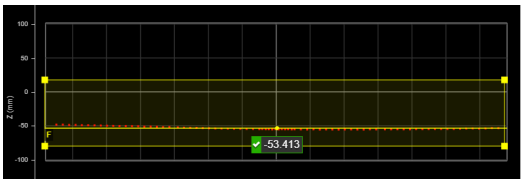
For more information on anchoring, see *Measurement Anchoring* on page 154.

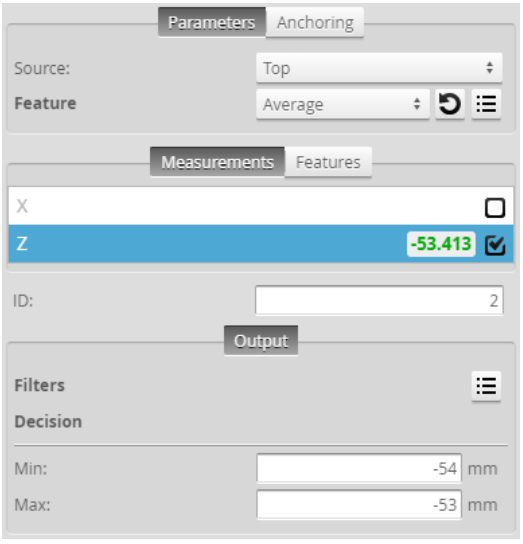
Position

The Position tool finds the X or Z axis position of a feature point. The feature type must be specified and is one of the following: Max Z, Min Z, Max X, Min X, Corner, Average (the mean X and Z of the data points), Rising Edge, Falling Edge, Any Edge, Top Corner, Bottom Corner, Left Corner, Right Corner, or Median (median X and Z of the data points).





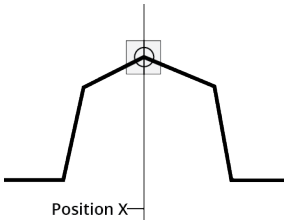
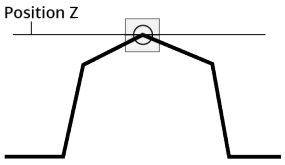




For information on adding, managing, and removing tools and measurements, as well as detailed descriptions of settings common to most tools, see *Tools Panel* on page 148.

Measurements and Settings

Measurements

Measurement	Illustration
X Finds the position of a feature on the X axis.	
Z Finds the position of a feature on the Z axis.	

Parameters

Parameter	Description
Source	The sensor that provides data for the tool's measurements. For more information, see *Source* on page 148.
Feature	The feature the tool uses for its measurements. One of the following: • Max Z • Min Z • Max X • Min X • Corner • Average • Rising Edge • Falling Edge • Any Edge • Top Corner • Bottom Corner • Left Corner • Right Corner • Median To set the region of a feature, adjust it graphically in the data viewer, or expand the feature using the expand button (⋮☰) and enter the values in the fields. For more information on regions, see *Regions* on page 149.
Filters	The filters that are applied to measurement values before they are output. For more information, see *Filters* on page 153.
Decision	The **Max** and **Min** settings define the range that determines whether the measurement tool sends a pass or fail decision to the output. For more information, see *Decisions* on page 152.

Anchoring

Anchor	Description
X or Z	Lets you choose the X or Z measurement of another tool to use as a positional anchor for this tool.

A measurement *must* be enabled in the other tool for it to be available as an anchor. The anchor measurement should also be properly configured before using it as an anchor.

For more information on anchoring, see *Measurement Anchoring* on page 154.

Script

A Script measurement can be used to program a custom measurement using a simplified C-based syntax. A script measurement can produce multiple measurement values and decisions for the output.

See *Scripts* below for more information on the script syntax.

To create or edit a Script measurement:

1. Add a new Script tool or select an existing Script measurement.

2. Edit the script code.

3. Add script outputs using the **Add** button.
   For each script output that is added, an index will be added to the **Output** drop-down and a unique ID will be generated.
   To remove a script output, click on the ✖ button next to it.

4. Click the **Save** button 💾 to save the script code.
   If there is a mistake in the script syntax, the result will be shown as a "Invalid" with a red border in the data viewer when you run the sensor.
   Outputs from multiple measurement tools can be used as inputs to the script. A typical script would take results from other measurement tools using the value and decision function, and output the result using the output function. Stamp information, such as time and encoder stamps, are available in the script, whereas the actual profile3D point cloud data is not. (The script engine is not powerful enough to process the data itself.) Only one script can be created.

Scripts

Scripts use outputs from other measurement tools to produce custom measurements.

Similar to other measurement tools, a script measurement can output multiple measurement values and decisions. Scripts are added, configured, and removed much like other measurement tools; for more information on this, see *Script* under *Profile Measurement* on page 158.

> Scripts must be less than 27,000 characters long.

Scripts use a simplified C-based syntax. The following elements of the C language are supported:

Supported Elements

Elements	Supported
Control Operators	if, while, do, for, switch and return.
Data Types	char, int, unsigned int, float, double, long long (64-bit integer).
Arithmetic and Logical Operator	Standard C arithmetic operators, except ternary operator (i.e., "condition? trueValue: falseValue"). Explicit casting (e.g., int a = (int) a_float) is not supported.
Function Declarations	Standard C function declarations with argument passed by values. Pointers are not supported.

Built-in Script Functions

The script engine provides the following types of functions:

- Measurement
- Output
- Memory
- Runtime variable
- Stamp
- Math

Measurement Functions

Function	Description
int Measurement_Exists(int id)	Determines if a measurement exists by ID. Parameters: id – Measurement ID Returns: 0 – measurement does not exist 1 – measurement exists
int Measurement_Valid(int id)	Determines if a measurement value is valid by its ID. Parameters: id - Measurement ID Returns 0 - Measurement is invalid 1 - Measurement is valid
double Measurement_Value (int id)	Gets the value of a measurement by its ID. Parameters: id - Measurement ID Returns: Value of the measurement 0 – if measurement does not exist 1 – if measurement exists
int Measurement_Decision (int id)	Gets the decision of a measurement by its ID. Parameters: ID - Measurement ID Returns: Decision of the measurement 0 – if measurement decision is false 1 – If measurement decision is true
int Measurement_NameExists(char* toolName, char* measurementName)	Determines if a measurement exist by name. Parameter:

Function	Description
	toolName – Tool name measurementName – Measurement name Returns: 0 – measurement does not exist 1 – measurement exists
int Measurement_Id (char* toolName, char* measurementName)	Gets the measurement ID by the measurement name. Parameters: toolName – Tool name measurementName – Measurement name Returns: -1 – measurement does not exist Other value – Measurement ID

Output Functions

Function	Description
void Output_Set (double value, int decision)	Sets the output value and decision on Output index 0. Only the last output value / decision in a script run is kept and passed to the Gocator output. To output an invalid value, the constant INVALID_VALUE can be used (e.g., Output_SetAt(0, INVALID_VALUE, 0)) Parameters: value - value output by the script decision - decision value output by the script. Can only be 0 or 1
void Output_SetAt(unsigned int index, double value, int decision)	Sets the output value and decision at the specified output index. To output an invalid value, the constant INVALID_VALUE can be used (e.g., Output_SetAt(0, INVALID_VALUE, 0)) Parameters: index – Script output index value – value output by the script decision – decision value output by the script. Can only be 0 or 1
void Output_SetId(int id, double value, int decision)	Sets the output value and decision at the specified script output ID. To output an invalid value, the constant INVALID_VALUE can be used (e.g., Output_SetId(0, INVALID_VALUE, 0)) Parameters: id – Script output ID

Memory Functions

Function	Description
void Memory_Set64s (int id, long long value)	Stores a 64-bit signed integer in persistent memory. Parameters:

Function	Description
	id - ID of the value value - Value to store
long long Memory_Get64s (int id)	Loads a 64-bit signed integer from persistent memory. Parameters: id - ID of the value Returns: value - Value stored in persistent memory
void Memory_Set64u (int id, unsigned long long value)	Stores a 64-bit unsigned integer in the persistent memory Parameters: id - ID of the value value - Value to store
unsigned long long Memory_Get64u (int id)	Loads a 64-bit unsigned integer from persistent memory. Parameters: id - ID of the value Returns: value - Value stored in persistent memory
void Memory_Set64f (int id, double value)	Stores a 64-bit double into persistent memory. Parameters: id - ID of the value value - Value to store
double Memory_Get64f (int id)	Loads a 64-bit double from persistent memory. All persistent memory values are set to 0 when the sensor starts. Parameters: id - ID of the value Returns: value - Value stored in persistent memory
int Memory_Exists (int id)	Tests for the existence of a value by ID. Parameters: id – Value ID Returns: 0 – value does not exist 1 – value exists
void Memory_Clear (int id)	Erases a value associated with an ID. Parameters: id – Value ID
void Memory_ClearAll()	Erases all values from persistent memory

Runtime Variable Functions

Function	Description
int RuntimeVariable_Count()	Returns the number of runtime variables that can be accessed. Returns: The count of runtime variables.
int RuntimeVariable_Get32s(int id)	Returns the value of the runtime variable at the given index. Parameters: Id – ID of the runtime variable Returns: Runtime variable value

Stamp Functions

Function	Description
long long Stamp_Frame()	Gets the frame number of the last frame.
long long Stamp_Time()	Gets the time stamp of the last frame.
long long Stamp_Encoder()	Gets the encoder position of the last frame when the image data was scanned/taken.
long long Stamp_EncoderZ()	Gets the encoder position at the time of the last index pulse of the last frame.
unsigned int Stamp_Inputs()	Gets the digital input state of the last frame. Returns a bit field representing digital input states.

Math Functions

Function	Description
float sqrt(float x)	Calculates square root of x
float sin(float x)	Calculates sin(x) (x in radians)
float cos(float x)	Calculates cos(x) (x in radians)
float tan(float x)	Calculates tan(x) (x in radians)
float asin(float x)	Calculates asin(x)
float acos(float x)	Calculates acos(x)
float atan(float x)	Calculates atan(x)
float pow (float x, float y)	Calculates the exponential value. x is the base, y is the exponent
float fabs(float x)	Calculates the absolute value of x

Output

The following sections describe the **Output** page.

> Most G2 and all G3 and G200 sensors shipped on or after July 18, 2022, no longer support analog output. G1 sensors and vertical G2 sensors will continue to support analog output. For more information, see *Analog Support* on page 22.

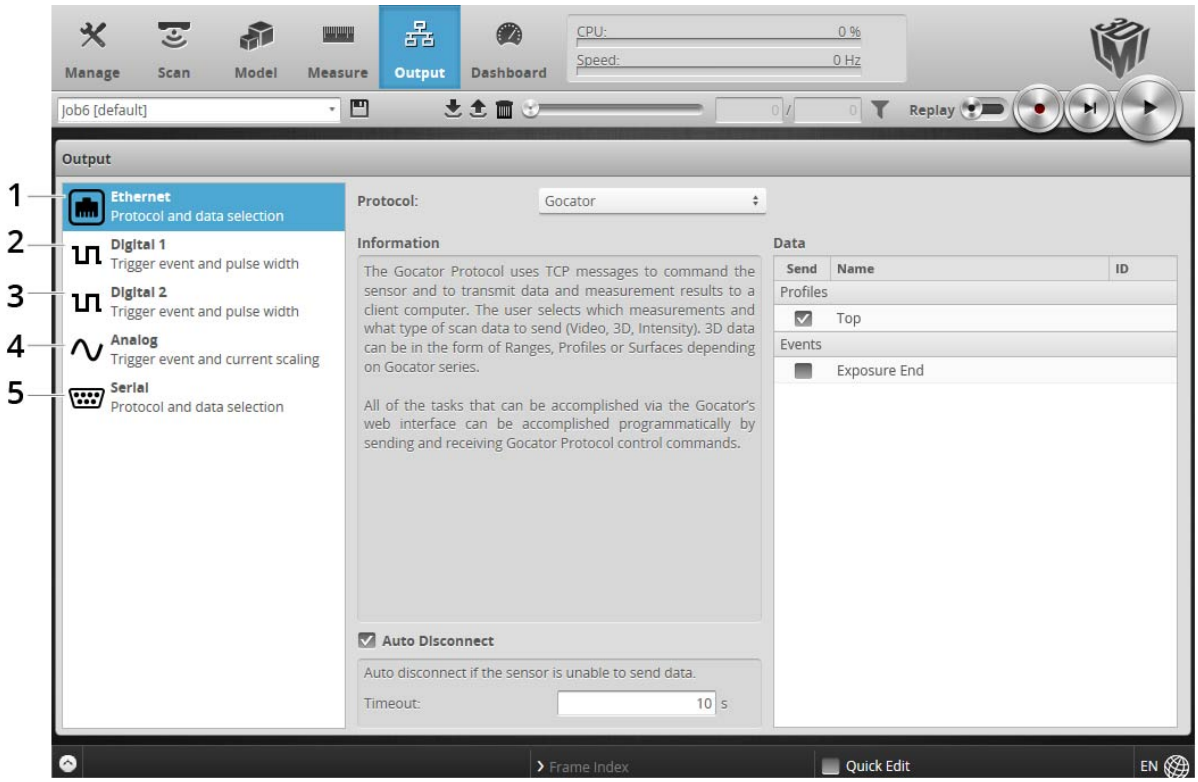
> Currently, output for Gocator 200 series sensors is automatically configured using GoWebScanSDK (for more information, see *GoWebScanSDK* on page 70).

> Gocator 200 series sensors do not support digital, serial, and analog output.

Output Page Overview

Output configuration tasks are performed using the **Output** page. Gocator sensors can transmit data and measurement results to various external devices using several output interface options.

> Gocator 200 sensors only support Ethernet outputs.

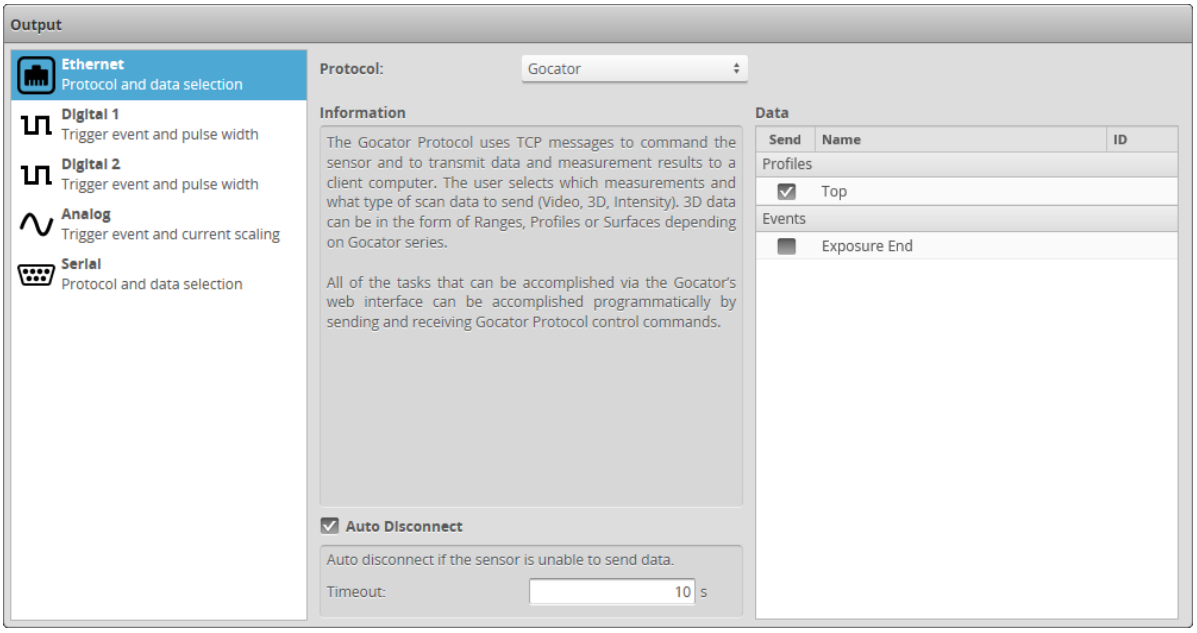


	Category	Description
1	Ethernet	Used to select the data sources that will transmit data via Ethernet. See *Ethernet Output* on the next page.

Ethernet Output

A sensor uses TCP messages (Gocator protocol) to receive commands from client computers, and to send video, , intensity, and measurement results to client computers. The sensor can also receive commands from and send measurement results to a PLC using ASCII, Modbus TCP, PROFINET, or EtherNet/IP protocol. See *Protocols* on page 256 for the specification of these protocols.

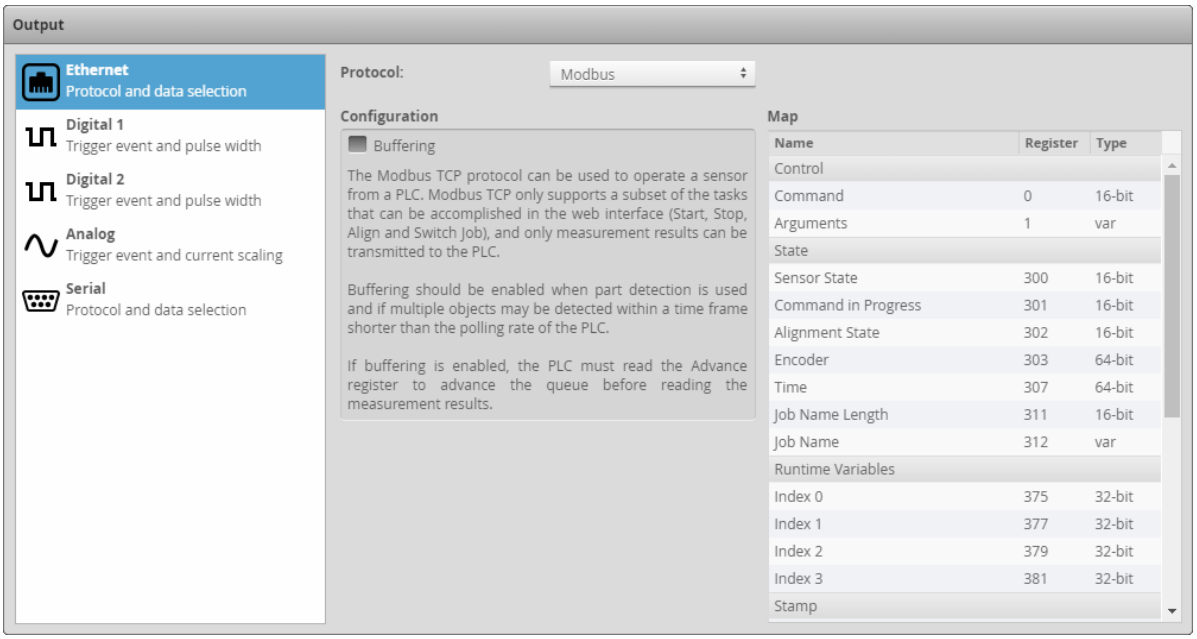
The specific protocols used with Ethernet output are selected and configured within the panel.



To receive commands and send results using Gocator Protocol messages:

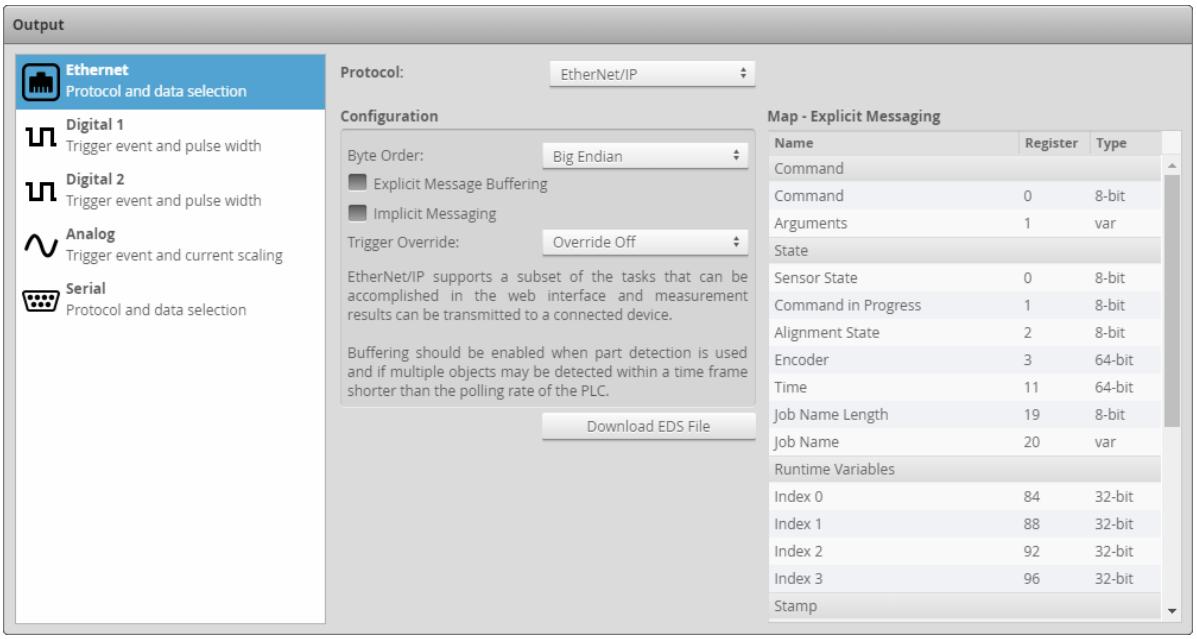
1. Go to the **Output** page.
2. Click on the **Ethernet** category in the **Output** panel.
3. Select **Gocator** as the protocol in the **Protocol** drop-down.
4. Check the video,, intensity, or measurement items to send.
5. (Optional) Uncheck the Auto Disconnect setting.
   By default, this setting is checked, and the timeout is set to 10 seconds.

All of the tasks that can be accomplished with the Gocator's web interface (creating jobs, performing alignment, sending data and health information, and software triggering, etc.) can be accomplished programmatically by sending Gocator protocol control commands.



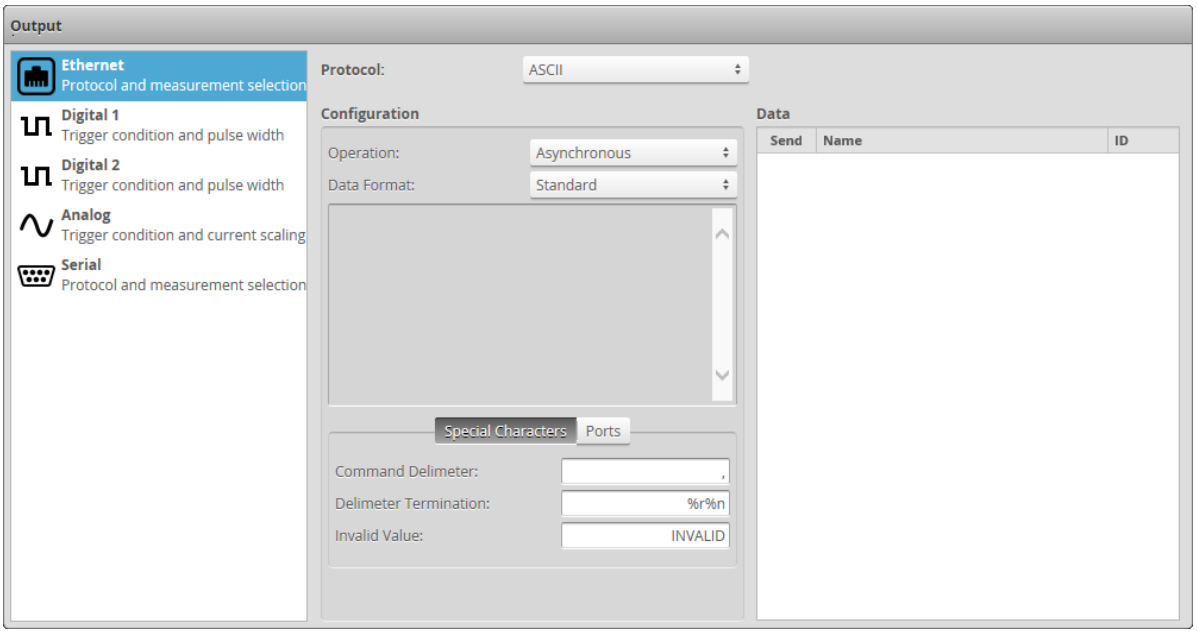
To receive commands and send results using Modbus TCP messages:

1. Go to the **Output** page.
2. Click on **Ethernet** in the **Output** panel.
3. Select **Modbus** as the protocol in the **Protocol** drop-down.
   Unlike the Gocator Protocol, you do not select which measurement items to output. The Ethernet panel will list the register addresses that are used for Modbus TCP communication.
   The Modbus TCP protocol can be used to operate a sensor. Modbus TCP only supports a subset of the tasks that can be performed in the web interface. A sensor can only process Modbus TCP commands when Modbus is selected in the **Protocol** drop-down.
4. Check the **Buffering** checkbox, if needed.
   Buffering is needed, for example, in Surface mode if multiple objects are detected within a time frame shorter than the polling rate of the PLC.
   If buffering is enabled with the Modbus protocol, the PLC must read the Advance register to advance the queue before reading the measurement results.



To receive commands and send results using EtherNet/IP messages:

1. Go to the **Output** page.
2. Click on **Ethernet** in the **Output** panel.
3. Select **EtherNet/IP** in the **Protocol** option.
   Unlike using the Gocator Protocol, you don't select which measurement items to output. The **Ethernet** panel will list the register addresses that are used for EtherNet/IP messages communication.
   The EtherNet/IP protocol can be used to operate a sensor. EtherNet/IP only supports a subset of the tasks that can be accomplished in the web interface. A sensor can only process EtherNet/IP commands when the EtherNet/IP is selected in the **Protocol** option.
4. Check the **Explicit Message Buffering** option, if needed.
   Buffering is needed, for example, in Surface mode if multiple objects are detected within a time frame shorter than the polling 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 (*Sample State Assembly* on page 321).
5. Check the **Implicit Messaging** option, if needed.
   Implicit messaging uses UDP and is faster than explicit messaging, so it is intended for time-critical applications. However, implicit messaging is layered on top of UDP. UDP is connectionless and data delivery is not guaranteed. For this reason, implicit messaging is only suitable for applications where occasional data loss is acceptable.
   For more information on setting up implicit messaging, see http://lmi3d.com/sites/default/files/APPNOTE_Implicit_Messaging_with_Allen-Bradley_PLCs.pdf.
6. Choose the byte order in the **Byte Order** dropdown.
7. Click the **Download EDS File** button to download an EDS file for use with your IDE.



To receive commands and send results using ASCII messages:

1. Go to the **Output** page.
2. Click on **Ethernet** in the **Output** panel.
3. Select **ASCII** as the protocol in the **Protocol** drop-down.
4. Set the operation mode in the **Operation** drop-down.
   In asynchronous mode, the data results are transmitted when they are available. In polling mode, users send commands on the data channel to request the latest result. See *Polling Operation Commands (Ethernet Only)* on page 326 for an explanation of the operation modes.
5. Select the data format from the **Data Format** drop-down.
   Standard: The default result format of the ASCII protocol. Select the measurement to send by placing a check in the corresponding checkbox. See *Standard Result Format* on page 334 for an explanation of the standard result mode.
   Standard with Stamp: Select the measurement to send by placing a check in the corresponding checkbox. See *Standard Result Format* on page 334 for an explanation of the standard result mode.
   Custom: Enables the custom format editor. Use the replacement patterns listed in **Replacement Patterns** to create a custom format in the editor. 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.
6. Set the special characters in the **Special Characters** tab.
   Set the command delimiter, delimiter termination, and invalid value characters. Special characters are used in commands and standard-format data results.
7. Set the TCP ports in the **Ports** tab.
   Select the TCP ports for the control, data, and health channels. If the port numbers of two channels are the same, the messages for both channels are transmitted on the same port.

Dashboard

The following sections describe the **Dashboard** page.

Dashboard Page Overview

The **Dashboard** page summarizes sensor health information. Use this information to troubleshoot your system.

	Element	Description
1	System	Displays sensor state and health information. See *State and Health Information* below.
2	Tool Stats	Not currently used by Gocator 200 series sensors.

State and Health Information

The following state and health information is available in the **System** panel on the **Dashboard** page:

Dashboard General System Values

Name	Description
Sensor State	Current sensor state (Conflict, Ready, or Running).
Application Version	Sensor firmware version.
Laser Safety	Whether Safety is enabled.
Uptime	Length of time since the sensor was power-cycled or reset.
CPU Usage	Sensor CPU utilization.
Current Speed	Current speed of the sensor.
Encoder Value	Current encoder value (ticks).
Encoder Frequency	Current encoder frequency (Hz).
Memory Usage	Sensor memory utilization (MB used / MB total available).

Name	Description
Storage Usage	Sensor flash storage utilization (MB used / MB total available).
Ethernet Link Speed	Speed of the Ethernet link (Mbps).
Ethernet Traffic	Network output utilization (MB/sec).
Internal Temperature	Internal sensor temperature.
Processing Latency	Last delay from camera exposure start to when the results are ready for output.
Processing Latency Peak	Peak delay from camera exposure start to when the results are ready for output.
Alignment State	Whether the sensor or sensor system has been aligned.
Over Temperature State	Whether the internal temperature of the sensor is over a predetermined level.

Dashboard History Values

Name	Description
Scan Count	Number of scans performed since sensor state last changed to Running.
Trigger Drop	Count of camera frames dropped due to excessive trigger speed.
Processing Drop	Count of frame drops due to excessive CPU utilization.
Ethernet Output Drop	Count of frame drops due to slow Ethernet link.
Analog Output Drop	Not used.
Serial Output Drop	Not used.
Digital Output 1 Drop	Not used.
Digital Output 2 Drop	Not used.
Digital Output 1 High Count	Not used.
Digital Output 2 High Count	Not used.
Digital Output 1 Low Count	Not used.
Digital Output 2 Low Count	Not used.
Anchor Invalid Count	Count of invalid anchors.
Valid Spot Count	Count of valid spots detected in the last frame.
Max Spot Count	Maximum number of spots detected since sensor was started.
Camera Search Count	Not applicable to these sensors.

Gocator Emulator

The emulator is a stand-alone application that lets you run a "virtual" sensor, encapsulated in a "scenario." When running a scenario, you can test jobs, evaluate data, and even learn more about new features, rather than take a physical device off the production line to do this. You can also use a scenario to familiarize yourself with the overall interface if you are new to Gocator.

System Requirements

The following are the system requirements for the software:

PC

- Processor: Intel Core i3 or equivalent (64-bit)
- RAM: 4 GB
- Hard drive: 500 GB
- Operating system: Windows 7 or higher (64-bit)

Limitations

In most ways, a scenario behaves like a real sensor, especially when visualizing data, setting up models and part matching, and adding and configuring measurement tools. The following are some of the limitations:

- Changes to job files in the emulator are *not* persistent (they are lost when you close or restart the emulator). However, you can keep a modified job by first saving it and then downloading it from the **Jobs** list on the **Manage** page to a client computer. The job file can then be loaded into the emulator at a later time or even onto a physical sensor for final testing.
- Performing alignment in the emulator has no effect and will never complete.
- The emulator does not support the PROFINET protocol.

For information on saving and loading jobs in the emulator, see *Creating, Saving, and Loading Jobs* on page 182.

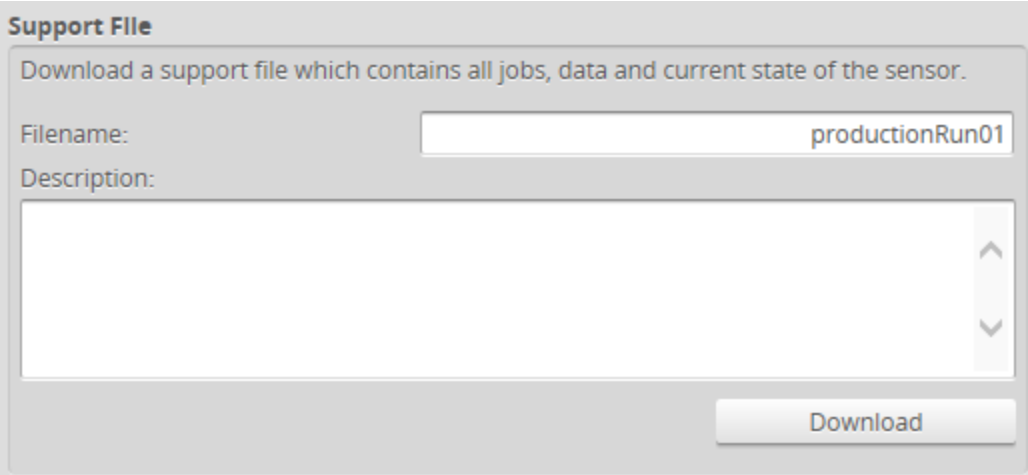
For information on uploading and downloading jobs between the emulator and a computer, and performing other job file management tasks, see *Downloading and Uploading Jobs* on page 186.

Downloading a Support File

The emulator provides several preinstalled scenarios.

You can also create scenarios yourself by downloading a support file from a physical sensor and then adding it to the emulator.

Support files can contain jobs, letting you configure systems and add measurements in an emulated sensor. Support files can also contain replay data, letting you test measurements and some configurations on real data. Dual-sensor systems are supported.



To download a support file:

1. Go to the **Manage** page and click on the **Support** category.
2. In **Filename**, type the name you want to use for the support file.

   When you create a scenario from a support file in the emulator, the filename you provide here is displayed in the emulator's scenario list.

   Support files end with the .gs extension, but you do not need to type the extension in **Filename**.
3. (Optional) In **Description**, type a description of the support file.

   When you create a scenario from a support file in the emulator, the description is displayed below the emulator's scenario list.
4. Click **Download**, and then when prompted, click **Save**.

   ⚠ Downloading a support file stops the sensor.

Running the Emulator

The emulator is contained in the utilities package (14405-x.x.x.x_SOFTWARE_GO_Utilities.zip). To get the package, go to http://lmi3d.com/support, choose your product from the Product Downloads section, and download the package from the Download Center.

To run the emulator, unzip the package and double-click the *GoEmulator* link in the unzipped *Emulator and Accelerator* subfolder.

Emulator launch screen

You can change the language of the emulator's interface from the launch screen. To change the language, choose a language option from the top drop-down:

Selecting the emulator interface language

Adding a Scenario to the Emulator

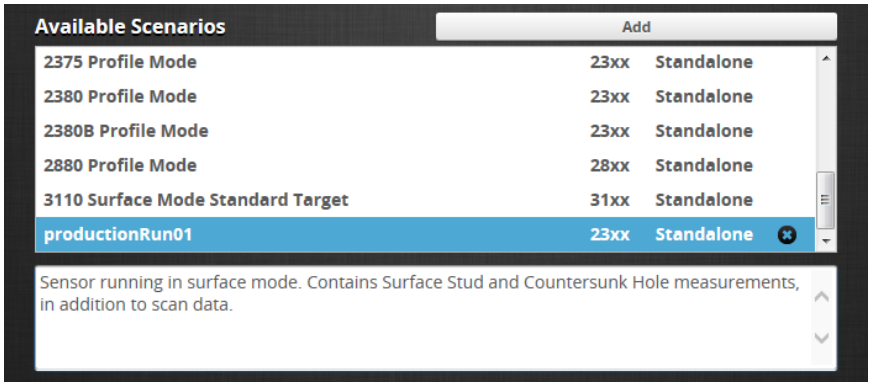
To simulate a physical sensor using a support file downloaded from a sensor, you must add it as a scenario in the emulator.

> You can add support files downloaded from any series of Gocator sensors to the emulator.

To add a scenario:

1. Launch the emulator if it isn't running already.
2. Click the **Add** button and choose a previously saved support file (.gs extension) in the **Choose File to Upload** dialog.

3. (Optional) In **Description**, type a description.



> You can only add descriptions for user-added scenarios.

Running a Scenario

After you have added a virtual sensor by uploading a support file to the emulator, you can run it from the **Available Scenarios** list on the emulator launch screen. You can also run any of the scenarios included in the installation.

To run a scenario:

1. If you want to filter the scenarios listed in **Available Scenarios**, do one or both of the following:
   - Choose a model family in the **Model** drop-down.
   - Choose **Standalone** or **Buddy** to limit the scenarios to single-sensor or dual-/multi-sensor scenarios, respectively.
2. Select a scenario in the **Available Scenarios** list and click **Start**.

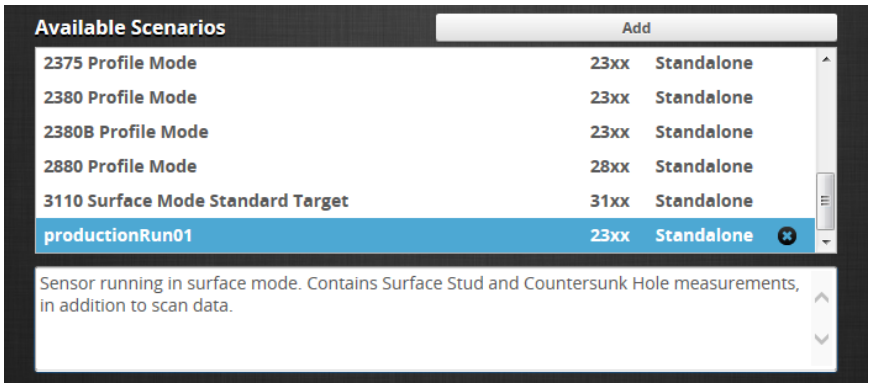
Removing a Scenario from the Emulator

You can easily remove a scenario from the emulator.

> You can only remove user-added scenarios.

To remove a scenario:

1. If the emulator is running a scenario, click Stop Emulation to stop it.
2. In the **Available Scenarios** list, scroll to the scenario you want to remove.

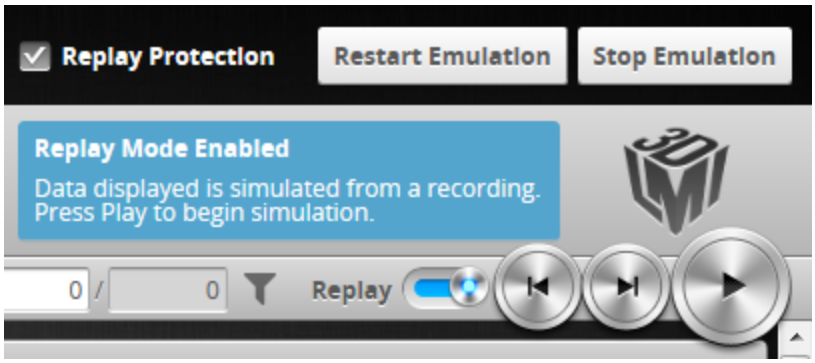


3. Click the ⊗ button next to the scenario you want to remove.

   The scenario is removed from the emulator.

Using Replay Protection

Making changes to certain settings on the **Scan** page causes the emulator to flush replay data. The **Replay Protection** option protects replay data by preventing changes to settings that affect replay data. Settings that do not affect replay data can be changed.



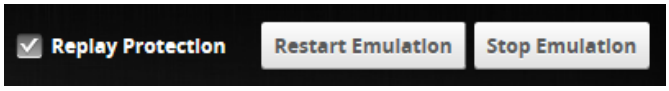
If you try to uncheck **Replay Protection**, you must confirm that you want to disable it.

Replay Protection is on by default.

Stopping and Restarting the Emulator

To stop the emulator:

- Click **Stop Emulation**.

  Stopping the emulator returns you to the launch screen.

To restart the emulator when it is running:

- Click **Restart Emulation**.

  Restarting the emulator restarts the currently running simulation.

Running the Emulator in Default Browser

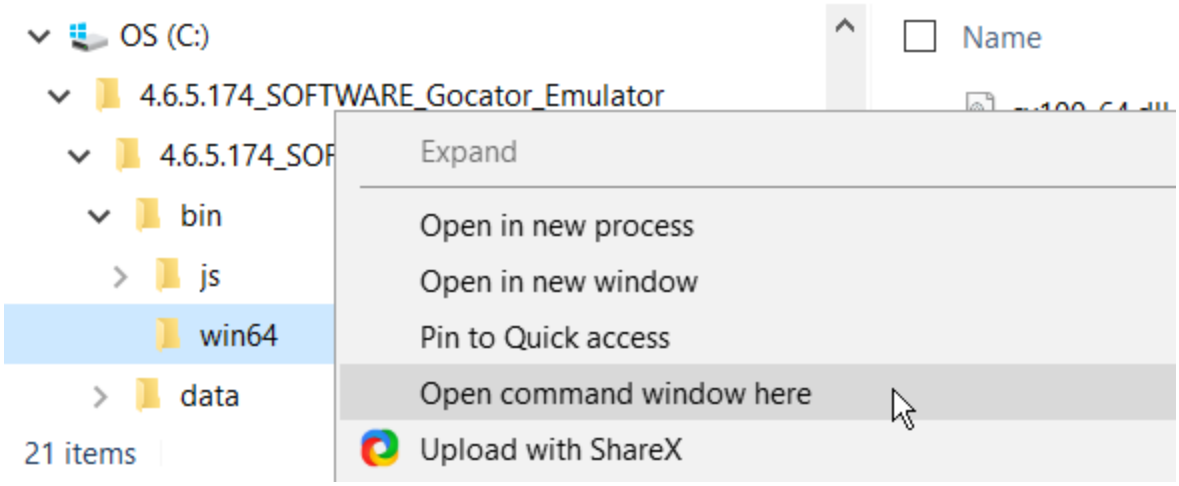
When you use the `/browser` command line parameter, the emulator application launches normally but also launches in your default browser. This provides additional flexibility when using the emulator. For example, you can resize the emulator running in a browser window.

To run the emulator in your default browser:

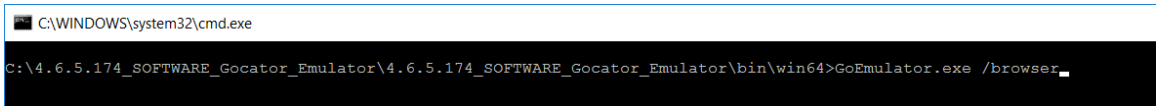
1. In Windows Explorer (Windows 7) or File Explorer (Windows 8 or 10), browse to the location of the emulator.

   The emulator is under `bin\win64`, in the location in which you installed the emulator.

2. Press and hold `Shift`, right-click the `win64` folder containing the emulator, and choose **Open command window here** (or **Open PowerShell window here**).



3. In the command prompt, type `GoEmulator.exe /browser` (or `.\GoEmulator.exe /browser` for PowerShell).



After the emulator application starts, the emulator also launches in your default browser.

Working with Jobs and Data

The following topics describe how to work with jobs and replay data (data recorded from a physical sensor) in a scenario running on the emulator.

Creating, Saving, and Loading Jobs

Changes saved to job files in the emulator are *not* persistent (they are lost when you close or restart the emulator). To keep jobs permanently, you must first save the job in the emulator and then download the job file to a client computer. See below for more information on creating, saving, and switching jobs. For information on downloading and uploading jobs between the emulator and a computer, see *Downloading and Uploading Jobs* on page 186.

The job drop-down list in the toolbar shows the jobs available in the emulator. The job that is currently active is listed at the top. The job name will be marked with "[unsaved]" to indicate any unsaved changes.

To create a job:

1. Choose **[New]** in the job drop-down list and type a name for the job.

2. Click the **Save** button or press **Enter** to save the job.
   The job is saved to the emulator using the name you provided.

To save a job:

- Click the **Save** button.

  The job is saved to the emulator.

To load (switch) jobs:

- Select an existing file name in the job drop-down list.

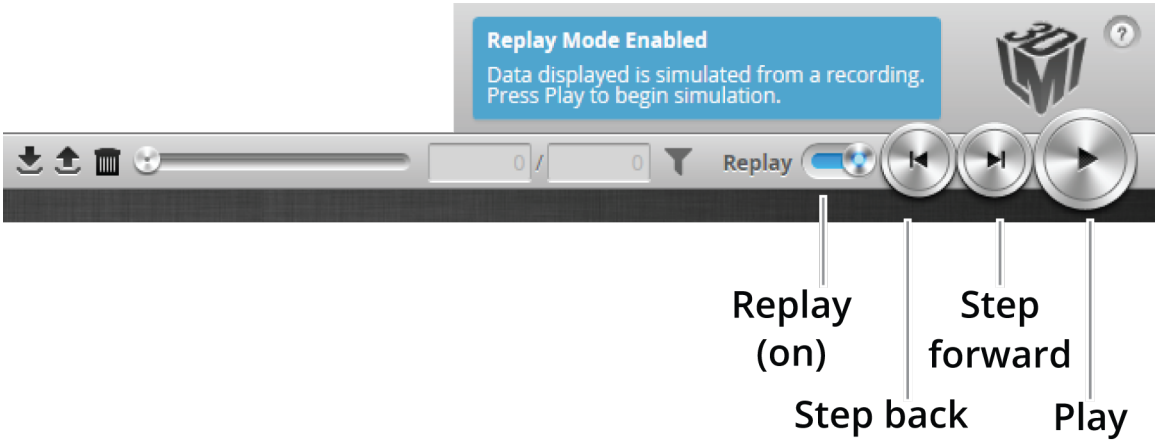
  The job is activated. If there are any unsaved changes in the current job, you will be asked whether you want to discard those changes.

Playback and Measurement Simulation

The emulator can replay scan data previously recorded by a physical sensor, and also simulate measurement tools on recorded data. This feature is most often used for troubleshooting and fine-tuning measurements, but can also be helpful during setup.

Playback is controlled using the toolbar controls.

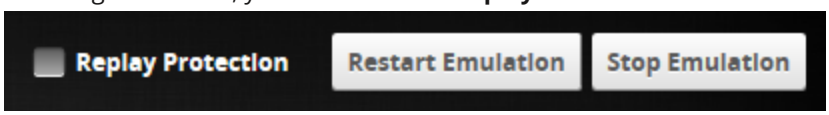
	Recording is not functional in the emulator.



Playback controls when replay is on

To replay data:

1. Toggle **Replay** mode on by setting the slider to the right in the **Toolbar**.
   The slider's background turns blue.
   To change the mode, you must uncheck **Replay Protection**.



2. Use the **Replay** slider or the **Step Forward**, **Step Back**, or **Play** buttons to review data.
   The **Step Forward** and **Step Back** buttons move the current replay location forward and backward by a single frame, respectively.
   The **Play** button advances the replay location continuously, animating the playback until the end of the replay data.
   The **Stop** button (replaces the **Play** button while playing) can be used to pause the replay at a particular location.
   The **Replay** slider (or **Replay Position** box) can be used to go to a specific replay frame.

To simulate measurements on replay data:

1. Toggle **Replay** mode on by setting the slider to the right in the **Toolbar**.
   The slider's background turns blue.
   To change the mode, **Replay Protection** must be unchecked.
2. Go to the **Measure** page.
   Modify settings for existing measurements, add new measurement tools, or delete measurement tools as desired. For information on adding and configuring measurements, see *Measurement and Processing* on page 146.
3. Use the **Replay Slider**, **Step Forward**, **Step Back**, or **Play** button to simulate measurements.
   Step or play through recorded data to execute the measurement tools on the recording.
   Individual measurement values can be viewed directly in the data viewer. Statistics on the measurements that have been simulated can be viewed in the **Dashboard** page; for more information on the dashboard, see *Dashboard* on page 175.

To clear replay data:

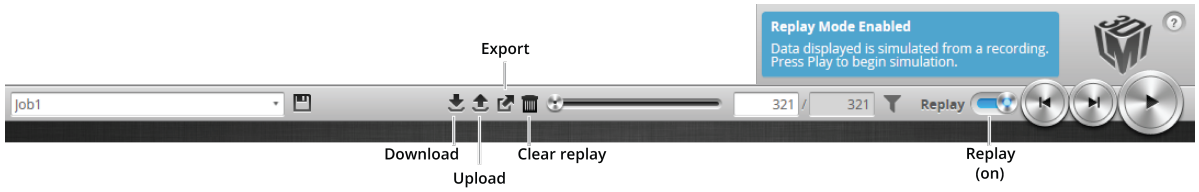
- Click the **Clear Replay Data** button .

Downloading, Uploading, and Exporting Replay Data

Replay data (recorded scan data) can be downloaded from the emulator to a client computer, or uploaded from a client computer to the emulator.

Data can also be exported from the emulator to a client computer in order to process the data using third-party tools.

> You can only upload replay data to the same sensor model that was used to create the data.



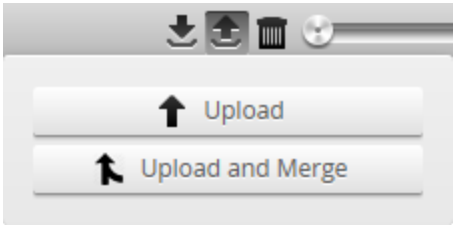
> Replay data is not loaded or saved when you load or save jobs.

To download replay data:

1. Click the Download button .
2. In the **File Download** dialog, click **Save**.
3. In the **Save As...** dialog, choose a location, optionally change the name, and click **Save**.

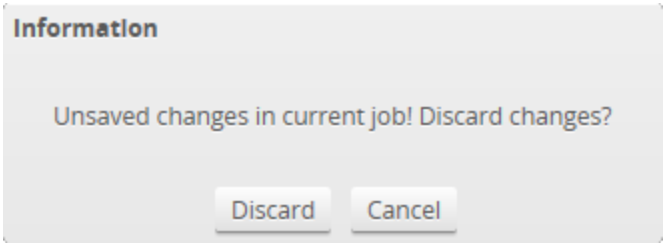
To upload replay data:

1. Click the Upload button .
   The Upload menu appears.



2. In the Upload menu, choose one of the following:
   - **Upload**: Unloads the current job and creates a new unsaved and untitled job from the content of the replay data file.
   - **Upload and merge**: Uploads the replay data and merges the data's associated job with the current job. Specifically, the settings on the **Scan** page are overwritten, but all other settings of the current job are preserved, including any measurements.

   If you have unsaved changes in the current job, the firmware asks whether you want to discard the changes.



3. Do one of the following:
   - Click **Discard** to discard any unsaved changes.
   - Click **Cancel** to return to the main window to save your changes.
4. If you clicked **Discard**, navigate to the replay data to upload from the client computer and click **OK**.
   The replay data is loaded, and a new unsaved, untitled job is created.

Replay data can be exported using the CSV format.

To export replay data in the CSV format:

1. In the **Scan Mode** panel, switch to .
2. Switch to Replay mode.
3. Click the Export button and select **All Data as CSV**.

   data at the current replay location is exported.
   Use the playback control buttons to move to a different replay location; for information on playback, see *To replay data* in *Playback and Measurement Simulation* on page 183.
4. (Optional) Convert exported data to another format using the CSV Converter Tool. For information on this tool, see *CSV Converter Tool* on page 338.

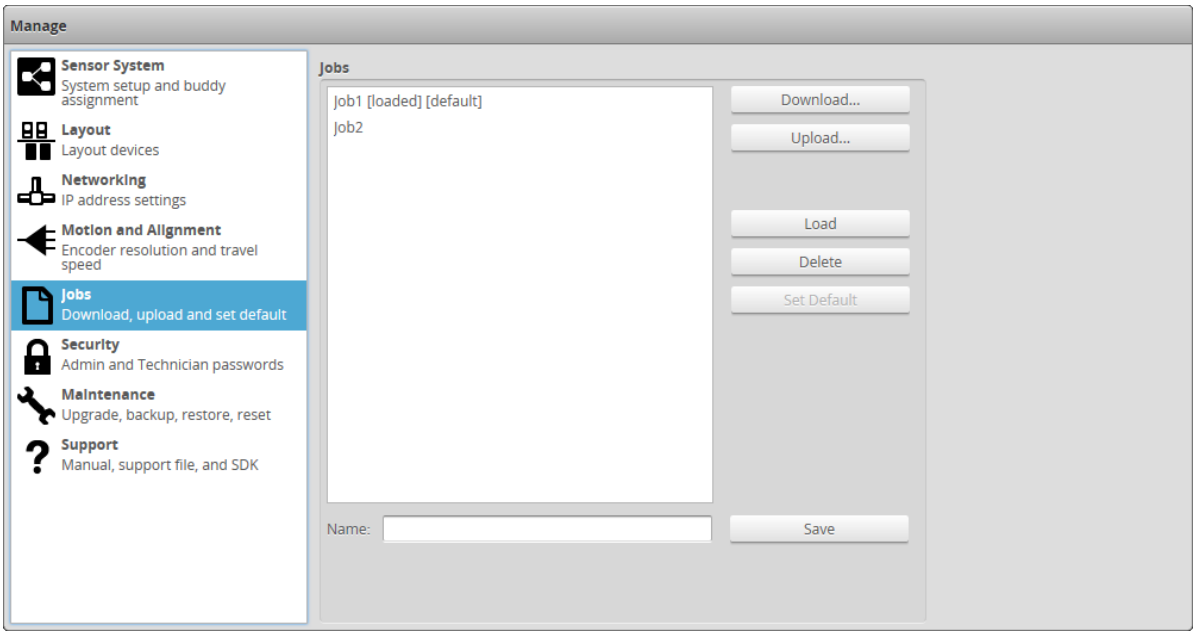
	The decision values in the exported data depend on the *current* state of the job, not the state during recording. For example, if you record data when a measurement returns a *pass* decision, change the measurement's settings so that a *fail* decision is returned, and then export to CSV, you will see a *fail* decision in the exported data.

To export video data to a BMP file:

1. In the **Scan Mode** panel, switch to Video mode.
   Use the playback control buttons to move to a different replay location; for information on playback, see *To replay data* in *Playback and Measurement Simulation* on page 183.
2. Switch to Replay mode.
3. Click the Export button and select **Video data as BMP**.

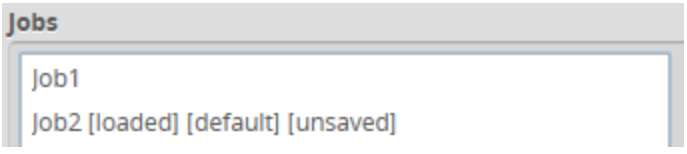
Downloading and Uploading Jobs

The **Jobs** category on the **Manage** page lets you manage the jobs in the emulator.



Element	Description
Name field	Used to provide a job name when saving files.
Jobs list	Displays the jobs that are currently saved in the emulator.
Save button	Saves current settings to the job using the name in the **Name** field. Changes to job files are not persistent in the emulator. To keep changes, first save changes in the job file, and then download the job file to a client computer. See the procedures below for instructions.
Load button	Loads the job that is selected in the job list. Reloading the current job discards any unsaved changes.
Delete button	Deletes the job that is selected in the job list.
Set as Default button	Setting a different job as the default is not persistent in the emulator. The job set as default when the support file (used to create a virtual sensor) was downloaded is used as the default whenever the emulator is started.
Download... button	Downloads the selected job to the client computer.
Upload... button	Uploads a job from the client computer.

Unsaved jobs are indicated by "[unsaved]".



Changes to job files in the emulator are *not* persistent (they are lost when you close or restart the emulator). However, you can keep modified jobs by first saving them and then downloading them to a client computer.

To save a job:

1. Go to the **Manage** page and click on the **Jobs** category.
2. Provide a name in the **Name** field.
   To save an existing job under a different name, click on it in the **Jobs** list and then modify it in the **Name** field.
3. Click on the **Save** button or press **Enter**.

To download, load, or delete a job, or to set one as a default, or clear a default:

1. Go to the **Manage** page and click on the **Jobs** category.
2. Select a job in the **Jobs** list.
3. Click on the appropriate button for the operation.

Scan, Model, and Measurement Settings

The settings on the **Scan** page related to actual scanning will clear the buffer of any scan data that is uploaded from a client computer, or is part of a support file used to create a virtual sensor. If **Replay Protection** is checked, the emulator will indicate in the log that the setting can't be changed because the change would clear the buffer. For more information on Replay Protection, see *Using Replay Protection* on page 180.

Other settings on the **Scan** page related to the post-processing of data can be modified to test their influence on scan data, without modifying or clearing the data,

For information on adding and configuring measurement tools, see *Measurement and Processing* on page 146.

Calculating Potential Maximum Frame Rate

You can use the emulator to calculate the potential maximum frame rate you can achieve with different settings.

For example, when you reduce the active area, in the **Active Area** tab on the **Sensor** panel, the maximum frame rate displayed on the **Trigger** panel is updated to reflect the increased speed that would be available in a physical sensor. (See *Active Area* on page 133 for more information on active area.)

Similarly, you can adjust exposure on the **Exposure** tab on the **Sensor** panel to see how this affects the maximum frame rate. (See *Exposure* on page 136 for more information on exposure.)

To adjust active area in the emulator, **Replay Protection** must be turned off. See *Using Replay Protection* on page 180 for more information.

Saving changes to active area causes replay data to be flushed.

Protocol Output

The emulator simulates output for all of Gocator's Ethernet-based protocols, with the exception of PROFINET.

- Gocator
- ASCII
- Modbus
- EtherNet/IP

Clients (such as PLCs) can connect to the emulator to access the simulated output and use the protocols as they would with a physical sensor.

The emulator allows connections to emulated sensors on localhost (127.0.0.1). You can also allow connections to emulated sensors on your computer's network card; for more information, see *Remote Operation* below.

Remote Operation

You can specify the IP address of one of your computer's network cards to allow clients to connect remotely to an emulated sensor using the `/ip` command line parameter. When the `/ip` parameter is not used, emulated sensors are only available on the local machine (that is, 127.0.0.1 or localhost).

> Clients can only connect to emulated sensors, not to the emulator's launch page.

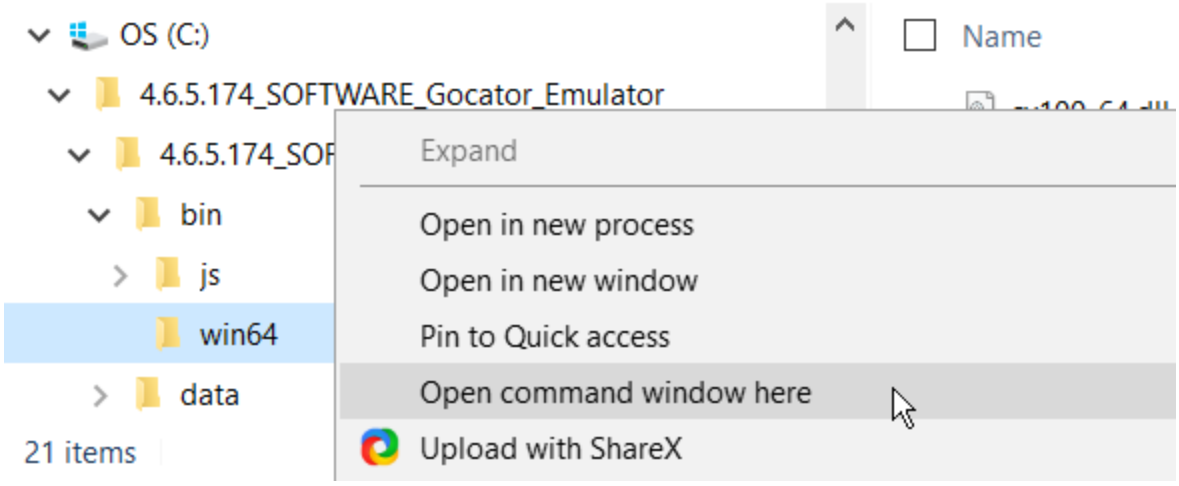
> You may need to contact your network administrator to allow connections to the computer running the emulated sensor.

To allow remote connections to an emulated sensor:

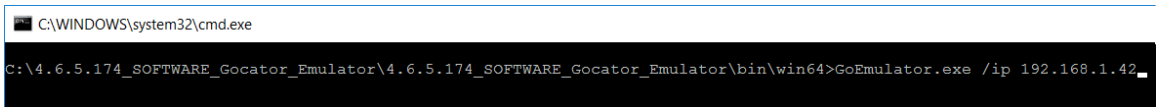
1. In Windows Explorer (Windows 7) or File Explorer (Windows 8 or 10), browse to the location of the emulator.

   The emulator is under `bin\win64`, in the location in which you installed the emulator.

2. Press and hold `Shift`, right-click the `win64` folder containing the emulator, and choose **Open command window here** (or **Open PowerShell window here**).



3. In the command prompt, type `GoEmulator.exe /ip`, followed by a valid IPV4 address on your network.



The emulator application starts.

> The emulator does not check that the IP address is valid.

4. From the emulator launch page, start a scenario.

   For more information, see *Running a Scenario* on page 180.

5. Provide the IP address you used with the `/ip` parameter, followed by port number 3191, to users who want to connect to the emulated sensor, for example:

   ```
   192.168.1.42:3191
   ```

Sensor Device Files

This section describes the user-accessible device files stored on a sensor.

> With the exception of tracheid threshold settings, any changes you make through the Gocator interface or through GoSDK are overridden or ignored by GoWebScanSDK. You *must* set tracheid threshold settings on each sensor, ideally using GoSDK.

Live Files

Various "live" files stored on a sensor represent the sensor's active settings and transformations (represented together as "job" files), the active replay data (if any), and the sensor log.

By changing the live job file, you can change how the sensor behaves. For example, to make settings and transformations active, write to or copy to the _live.job file. You can also save active settings or transformations to a client computer, or to a file on the sensor, by reading from or copying these files, respectively.

> The live files are stored in volatile storage. Only user-created job files are stored in non-volatile storage.

The following table lists the live files:

Live Files

Name	Read/Write	Description
_live.job	Read/Write	The active job. This file contains a Configuration component containing the current settings. If Alignment Reference in the active job is set to Dynamic, it also contains a Transform component containing transformations. For more information on job files (live and user-created), accessing their components, and their structure, see *Job File Structure* on the next page.
_live.cfg	Read/Write	A standalone representation of the Configuration component contained in _live.job. Used primarily for backwards compatibility.
_live.tfm	Read/Write	**If Alignment Reference of the active job is set to Dynamic:** A copy of the Transform component in _live.job. Used primarily for backwards compatibility. **If Alignment Reference of the active job is set to Fixed:** The transformations that are used for *all* jobs whose Alignment Reference setting is set to Fixed.
_live.log	Read	A sensor log containing various messages. For more information on the log file, see *Log File* on the next page.

Name	Read/Write	Description
_live.rec	Read/Write	The active replay simulation data.
ExtendedId.xml	Read	Sensor identification.

Log File

The log file contains log messages generated by the sensor. The root element is *Log*.

To access the log file, use the Read File command, passing "_live.log" to the command. The log file is read-only.

Log Child Elements

Element	Type	Description
@idStart	64s	Identifier of the first log.
@idEnd	64s	Identifier of the final log.
List of (Info | Warning | Error)	List	An ordered list of log entries. This list is empty if idEnd < idStart.

Log/Info | Log/Warning | Log/Error Elements

Element	Type	Description
@time	64u	Log time, in uptime (µs).
@source	32u	The serial number of the sensor the log was produced by.
@id	32u	The Indentifier, or index, of the log
@value	String	Log content; may contain printf-style format specifiers (e.g. %u).
List of (IntArg | FloatArg | Arg)	List	An ordered list of arguments: IntArg – Integer argument FloatArg – Floating-point argument Arg – Generic argument

The arguments are all sent as strings and should be applied in order to the format specifiers found in the content.

Job File Structure

The following sections describe the structure of job files.

Job files, which are stored in a sensor's internal storage, control system behavior when a sensor is running. Job files contain the settings and potentially the transformations associated with the job (if Alignment Reference is set to Dynamic).

There are two kinds of job files:

- A special job file called "_live.job." This job file contains the *active* settings and potentially the transformations associated with the job. It is stored in volatile storage.
- Other job files that are stored in non-volatile storage.

Job File Components

A job file contains components that can be loaded and saved as independent files. The following table lists the components of a job file:

Job File Components

Component	Path	Description
Configuration	config.xml	The job's configurations. This component is always present. For more information, see *Configuration* below.
Transform	transform.xml	Transformation values. Present only if Alignment Reference is set to Dynamic. For more information, see *Transform* on page 253.

Elements in the components contain three types of values: settings, constraints, and properties. Settings are input values that can be edited. Constraints are read-only limits that define the valid values for settings. Properties are read-only values that provide supplemental information related to sensor setup.

When a job file is received from a sensor, it will contain settings, constraints, and properties. When a job file is sent to a sensor, any constraints or properties in the file will be ignored.

Changing the value of a setting can affect multiple constraints and properties. After you upload a job file, you can download the job file again to access the updated values of the constraints and properties.

Accessing Files and Components

Job file components can be accessed individually as XML files using path notation. For example, the configurations in a user-created job file called *productionRun01.job* can be read by passing "productionRun01.job/config.xml" to the Read File command. In the same way, the configurations in the active job could be read using "_live.job/config.xml".

> If Alignment Reference is set to Fixed, the active job file (_live.job) will not contain transformations. To access transformations in this case, you must access them via _live.tfm.

> The following sections correspond to the XML structure used in job file components.

Configuration

The Configuration component of a job file contains settings that control how a sensor behaves.

You can access the Configuration component of the active job as an XML file, either using path notation, via "_live.job/config.xml", or directly via "_live.cfg".

You can access the Configuration component in user-created job files in non-volatile storage, for example, "productionRun01.job/config.xml". You can only access configurations in user-created job files using path notation.

See the following sections for the elements contained in this component.

All sensors share a common job file structure and settings for all features are included in job files, regardless of the model.

> If a setting in a job file is not used by a sensor, the setting's *used* property is set to 0.

Configuration Child Elements

Element	Type	Description
@version	32u	Configuration version (101).
@versionMinor	32u	Configuration minor version (9).
Setup	Section	For a description of the Setup elements, see *Setup* below.
Replay	Section	Contains settings related to recording filtering (see *Replay* on page 217).
Streams	Section	Read-only collection of available data streams (see *Streams/Stream (Read-only)* on page 218).
ToolOptions	Section	List of available tool types and their information. See *ToolOptions* on page 219 for details.
Tools	Collection	Collection of sections. Each section is an instance of a tool and is named by the type of the tool it describes. For more information, see the sections for each tool under *Tools* on page 221.
Tools.options	String (CSV)	Deprecated. Replaced by ToolOptions.
Outputs	Section	For a description of the Output elements, see *Output* on page 250.

Setup

The Setup element contains settings related to system and sensor setup.

Setup Child Elements

Element	Type	Description
TemperatureSafetyEnabled	Bool	Enables laser temperature safety control. Only applies to certain laser-based sensors.
TemperatureSafetyEnabled.used	Bool	Whether or not this property is used.
ScanMode	32s	The default scan mode.
ScanMode options	String (CSV)	List of available scan modes.
OcclusionReductionEnabled	Bool	Enables occlusion reduction.
OcclusionReductionEnabled.used	Bool	Whether or not property is used.
OcclusionReductionEnabled.value	Bool	Actual value used if not configurable.
OcclusionReductionAlg	32s	The Algorithim to use for occlusion reduction: 0 – Standard 1 – High Quality
OcclusionReductionAlg.used	Bool	Whether or not property is used
OcclusionReductionAlg.value	Bool	Actual value used if not configurable
UniformSpacingEnabled	Bool	Enables uniform spacing.

Element	Type	Description
UniformSpacingEnabled.used	Bool	Whether or not property is used.
UniformSpacingEnabled.readonly	Bool	Whether or not property can be modified.
UniformSpacingEnabled.value	Bool	Actual value used if not configurable.
IntensityEnabled	Bool	Enables intensity data collection.
IntensityEnabled.used	Bool	Whether or not property is used.
IntensityEnabled.value	Bool	Actual value used if not configurable.
FlickerFreeModeEnabled	Bool	Enables flicker-free operation.
FlickerFreeModeEnabled.used	Bool	Whether flicker-free operation can be used on this sensor.
ExternalInputZPulseEnabled	Bool	Not used.
ExternalInputZPulseIndex	32u	Input index to use for the input triggered z pulse feature.
ExternalInputZPulseEnabled.used	Bool	Whether the index can be set.
TransmitLimit	32u	Transmit Limit Percentage (1 - 100)
BackgroundSuppression	Section	See *BackgroundSuppression* below.
Filters	Section	See *Filters* below.
Trigger	Section	See *Trigger* on page 198.
Layout	Section	See *Layout* on page 200.
Alignment	Section	See *Alignment* on page 201.
Devices	Collection	A collection of two Device sections (with roles main and buddy). See *Devices / Device* on page 203.
SurfaceGeneration	Section	See *SurfaceGeneration* on page 210.
SurfaceSections	Section	See *SurfaceSections* on page 211.
ProfileGeneration	Section	See *ProfileGeneration* on page 212.
PartDetection	Section	See *PartDetection* on page 213.
PartMatching	Section	See *PartMatching* on page 215.
Custom	Custom	Used by specialized sensors.

BackgroundSuppression

The BackgroundSuppression element contains settings related to background suppression.

BackgroundSuppression Child Elements

Element	Type	Description
Enabled	Bool	Enables background suppression.
FrameRatio	64f	Ratio of background frames to calibration frames

Filters

The Filters element contains settings related to post-processing profiles before they are output or used by measurement tools.

XSmoothing

XSmoothing Child Elements

Element	Type	Description
@used	Bool	Whether or not this field is used
Enabled	Bool	Enables filtering.
Window	64f	Window size (mm).
Window.min	64f	Minimum window size (mm).
Window.max	64f	Maximum window size (mm).

YSmoothing

YSmoothing Child Elements

Element	Type	Description
@used	Bool	Whether or not this field is used
Enabled	Bool	Enables filtering.
Window	64f	Window size (mm).
Window.min	64f	Minimum window size (mm).
Window.max	64f	Maximum window size (mm).

XGapFilling

XGapFilling Child Elements

Element	Type	Description
@used	Bool	Whether or not this field is used
Enabled	Bool	Enables filtering.
Window	64f	Window size (mm).
Window.min	64f	Minimum window size (mm).
Window.max	64f	Maximum window size (mm).

YGapFilling

YGapFilling Child Elements

Element	Type	Description
@used	Bool	Whether or not this field is used
Enabled	Bool	Enables filtering.
Window	64f	Window size (mm).
Window.min	64f	Minimum window size (mm).
Window.max	64f	Maximum window size (mm).

XMedian

XMedian Child Elements

Element	Type	Description
@used	Bool	Whether or not this field is used
Enabled	Bool	Enables filtering.
Window	64f	Window size (mm).
Window.min	64f	Minimum window size (mm).
Window.max	64f	Maximum window size (mm).

YMedian

YMedian Child Elements

Element	Type	Description
@used	Bool	Whether or not this field is used
Enabled	Bool	Enables filtering.
Window	64f	Window size (mm).
Window.min	64f	Minimum window size (mm).
Window.max	64f	Maximum window size (mm).

XDecimation

XDecimation Child Elements

Element	Type	Description
@used	Bool	Whether or not this field is used
Enabled	Bool	Enables filtering.
Window	64f	Window size (mm).
Window.min	64f	Minimum window size (mm).
Window.max	64f	Maximum window size (mm).

YDecimation

YDecimation Child Elements

Element	Type	Description
@used	Bool	Whether or not this field is used
Enabled	Bool	Enables filtering.
Window	64f	Window size (mm).
Window.min	64f	Minimum window size (mm).
Window.max	64f	Maximum window size (mm).

XSlope

This filter is only available on displacement sensors.

XSlope Child Elements

Element	Type	Description
@used	Bool	Whether or not this field is used
Enabled	Bool	Enables filtering.
Window	64f	Window size (mm).
Window.min	64f	Minimum window size (mm).
Window.max	64f	Maximum window size (mm).

YSlope

This filter is only available on displacement sensors.

YSlope Child Elements

Element	Type	Description
@used	Bool	Whether or not this field is used
Enabled	Bool	Enables filtering.
Window	64f	Window size (mm).
Window.min	64f	Minimum window size (mm).
Window.max	64f	Maximum window size (mm).

Trigger

The Trigger element contains settings related to trigger source, speed, and encoder resolution.

Gocator 200 scanners do not support direct encoder input.

Trigger Child Elements

Element	Type	Description
Source	32s	Trigger source: 0 – Time 3 – Software
Source.options	32s (CSV)	List of available source options.
ExternalInputIndex	32s	Index of external input when Source (above) is set to 2 – Digital Input and connected to a Master. 0 – first digital input 1 – second digital input 2 – third digital input 3 – fourth digital input
ExternalInputIndex.options	32s (CSV)	List of available external input indices.
ExternalInputIndex.used	Bool	Whether the external input index used.
Units	32s	Sensor triggering units when source is not clock or encoder: 0 – Time

Element	Type	Description
		1 – Encoder
FrameRate	64f	Frame rate for time trigger (Hz).
FrameRate.min	64f	Minimum frame rate (Hz).
FrameRate.max	64f	Maximum frame rate (Hz).
FrameRate.maxSource	32s	Source of maximum frame rate limit: 0 – Imager 1 – Surface generation
TracheidRate	64f	The frame rate of Tracheid data (Read Only)
TracheidRate.used	Bool	Whether the sensor has a Tracheid data rate.
FrameDataRate	64f	The frame rate of normal (range/profile/surface) data (Read Only)
FrameDataRate.used	Bool	Whether the sensor has a separate FrameDataRate
EncoderSpacing.min	64f	Minimum encoder spacing (mm).
EncoderSpacing.max	64f	Maximum encoder spacing (mm).
EncoderSpacing.minSource	32s	Source of minimum encoder spacing: 0 – Resolution 1 – Surface generation
EncoderSpacing.used	Bool	Whether or not this parameter is configurable.
EncoderTriggerMode	32s	Encoder triggering mode: 0 – Tracking backward 1 – Bidirectional 2 – Ignore backward
Delay	64f	Trigger delay (µs or mm).
Delay.min	64f	Minimum trigger delay (µs or mm).
Delay.max	64f	Maximum trigger delay (µs or mm).
GateEnabled	Bool	Enables digital input gating.
GateEnabled.used	Bool	True if this parameter can be configured.
GateEnabled.value	Bool	Actual value if the parameter cannot be configured.
BurstEnabled	Bool	Enables burst triggering.
BurstEnabled.Used	Bool	Whether or not this parameter is configurable.
BurstCount	32u	Number of scans to take during burst triggering.
BurstCount.used	Bool	Whether or not this parameter is configurable.
BurstCount.max	32u	Maximum burst count.
ReversalDistanceAutoEnabled	Bool	Whether or not to use auto-calculated value.
ReversalDistanceAutoEnabled.used	Bool	Whether or not this parameter can be configured.
ReversalDistance	64f	Encoder reversal threshold (for jitter handling)

Element	Type	Description
ReversalDistance.used	Bool	Whether or not this parameter is used.
ReversalDistance.value	64f	Actual value.
LaserSleepMode.used	Bool	Whether or not this feature can be configured.
LaserSleepMode/Enabled	Bool	Enables or disables the feature.
LaserSleepMode/IdleTime	64u	Idle time before laser is turned off (µs).
LaserSleepMode/WakupEncoderTravel	64u	Minimum amount of encoder movement before laser turns on (mm).

Layout

Layout Child Elements

Element	Type	Description
DataSource	32s	Data source of the layout output (read-only): 0 – Top 1 – Bottom 2 – Top left 3 – Top right 4 – Top Bottom 5 – Left Right
XSpacingCount	32u	Number of points along X when data is resampled.
YSpacingCount	32u	Number of points along Y when data is resampled.
TransformedDataRegion	Region3D	Transformed data region of the layout output.
Orientation	32s	Sensor orientation: 0 – Normal (single-sensor system) / Wide (dual-sensor system) 1 – Opposite 2 – Reverse 3 – Grid
Grid	Grid	Grid representation of the multi-sensor layout.
Orientation.options	32s (CSV)	List of available orientation options.
Orientation.value	32s	Actual value used if not configurable.
MultiplexBuddyEnabled	Bool	Enables multiplexing for buddies.
MultiplexSingleEnabled	Bool	Enables multiplexing for a single sensor configuration.
MultiplexSingleExposureDuration	64f	Exposure duration in µs (currently rounded to integer when read by the sensor)
MultiplexSingleDelay	64f	Delay in µs. (Currently gets rounded up when read by the sensor.)
MultiplexSinglePeriod	64f	Period in µs. (Currently gets rounded up when read by the sensor.)
MultiplexSinglePeriod.min	64f	Minimum period in µs.

Region3D Child Elements

Element	Type	Description
X	64f	X start (mm).
Y	64f	Y start (mm).
Z	64f	Z start (mm).
Width	64f	X extent (mm).
Length	64f	Y extent (mm).
Height	64f	Z extent (mm).
ZAngle	64f	Z Angle start (degrees).
ZAngle.used	Bool	Whether or not this property is used.

Grid Elements

Element	Type	Description
ColumnCount	32u	Column count.
ColumnCount.value	32u	Column count value.

Alignment

The Alignment element contains settings related to alignment.

Alignment Child Elements

Element	Type	Description
@used	Bool	Whether or not this field is used
InputTriggerEnabled	Bool	Enables digital input-triggered alignment operation.
InputTriggerEnabled.used	Bool	Whether or not this feature can be enabled. This feature is available only on some sensor models.
InputTriggerEnabled.value	Bool	Actual feature status.
Type	32s	Type of alignment operation: 0 – Stationary 1 – Moving
Type.options	32s (CSV)	List of available alignment types.
StationaryTarget	32s	Stationary alignment target: 0 – None 1 – Disk 2 – Bar 3 – Plate
StationaryTarget.options	32s (CSV)	List of available stationary alignment targets.
MovingTarget	32s	Moving alignment target: 1 – Disk 2 – Bar
MovingTarget.options	32s (CSV)	List of available moving alignment targets.

Element	Type	Description
EncoderCalibrateEnabled	Bool	Not used.
Disk	Section	See *Disk* below.
Bar	Section	See *Bar* below.
Plate	Section	See *Plate* below.
Polygon	Section	See *Polygon* on the next page.

Disk

Disk Child Elements

Element	Type	Description
Diameter	64f	Disk diameter (mm).
Height	64f	Disk height (mm).

Bar

Bar Child Elements

Element	Type	Description
Width	64f	Bar width (mm).
Height	64f	Bar height (mm).
HoleCount	32u	Number of holes.
HoleCount.value	32u	Actual number of holes expected by system.
HoleCount.used	Bool	Whether the hole count with be used in the bar alignment proceudure.
HoleDistance	64f	Distance between holes (mm).
HoleDistance.used	Bool	Whether the hole distance will be used in the bar alignment procedure.
HoleDiameter	64f	Diameter of holes (mm).
HoleDiameter.used	Bool	Whether the hold diameter will be used in the bar alignment procedure.
DegreesOfFreedom	32s	Degrees of freedom (DOF) to align: 42 – 3 DOF: x, z, y angle 58 – 4 DOF: x, y, z, y angle 59 – 5 DOF: x, y, z, y angle, z angle

Plate

Plate Child Elements

Element	Type	Description
Height	64f	Plate height (mm).
HoleCount	32u	Number of holes.
RefHoleDiameter	64f	Diameter of reference hole (mm).
SecHoleDiameter	64f	Diameter of secondary hole(s) (mm).

Polygon

Polygon Child Elements

Element	Type	Description
Corners	List	Contains a list of Corners (described below).
Corners.minCount	32s	Minimum number of corners.

Polygon/Corner

Corner Child Elements

Element	Type	Description
X	64f	X Position
Y	64f	Y Position
Devices	List of 32u	List of devices this corner is assigned to.
Devices.options	List of 32u	List of valid options for this field.

Devices / Device

Devices / Device Child Elements

Element	Type	Description
@index	32u	Ordered index of devices in device list.
@role	32s	Sensor role: 0 – Main
Layout	Layout	Multiplexing bank settings.
DataSource	32s	Data source of device output (read-only): 0 – Top
XSpacingCount	32u	Number of resampled points along X (read-only).
YSpacingCount	32u	Number of resampled points along Y (read-only).
ActiveArea	Region3D	Active area. (Contains min and max attributes for each element.)
TransformedDataRegion	Region3D	Active area after transformation (read-only).
FrontCamera	Window	Front camera window (read-only).
BackCamera	Window	Back camera window (read-only).
BackCamera.used	Bool	Whether or not this field is used.
PatternSequenceType	32s	The projector pattern sequence to display when a projector equipped device is running. The following types are possible: -1 – None 0 – Default 100 – Nine Lines 101 – Focus 102 – Standard Sequence
PatternSequenceType.options	32s	List of available pattern sequence types.

Element	Type	Description
PatternSequenceType.used	Bool	Whether or not this field is used.
PatternSequenceIndex	32u	The index of the pattern sequence to display. Choose the pattern that produces the best data. The indices represent Phase Pattern Sequences, followed by Stripe Pattern Sequences in reverse order. The lower indices are the higher frequency phase code patterns, and the higher indices are the lower frequency binary patterns. Index 1 [Phase Pattern Sequence Image 5]: Highest frequency sinusoid. Index 2 [Phase Pattern Sequence Image 4] [...] Index 5 [Phase Pattern Sequence Image 1]: Lowest frequency sinusoid. Index 6 [Stripe Pattern Sequence Image 7]: Highest bar count. Index 7 [Stripe Pattern Sequence Image 6] [...] Index 12 [Stripe Pattern Sequence Image 1]: Lowest bar count) Index 13 [Reference Image 1]
PatternSequenceIndex.min	32u	The minimum index (inclusive)
PatternSequenceIndex.max	32u	The maximum index (inclusive)
PatternSequenceIndex.used	Bool	Whether or not the pattern sequence index should be displayed
PatternSequenceIndex	32u	The index of the pattern sequence to display.
PatternSequenceIndex.min	32u	The minimum index (inclusive).
PatternSequenceIndex.max	32u	The maximum index (inclusive).
PatternSequenceIndex.used	Bool	Whether or not the pattern sequence index should be displayed.
PatternSequenceCount	32u	Number of frames in the active sequence (read-only).
ExposureMode	32s	Exposure mode: 0 – Single exposure
ExposureMode.options	32s (CSV)	List of available exposure modes.
Exposure	64f	Single exposure (µs).
Exposure.min	64f	Minimum exposure (µs).
Exposure.max	64f	Maximum exposure (µs).
Exposure.used	Bool	Whether or not this field is used.
DynamicExposureMin	64f	Dynamic exposure range minimum (µs).
DynamicExposureMax	64f	Dynamic exposure range maximum (µs).
ExposureSteps	64f (CSV)	Mutiple exposure list (µs).

Element	Type	Description
ExposureSteps.countMin	32u	Minimum number of exposure steps.
ExposureSteps.countMax	32u	Maximum number of exposure steps.
IntensitySource	32s	Intensity source: 0 – Both cameras 1 – Front camera 2 – Back camera
IntensitySource.options	32s (CSV)	List of available intensity sources.
IntensityMode	32s	Intensity Mode: 0 – Auto 1 - Preserve
IntensityMode.used	Bool	Whether intensity mode is used
ZSubsampling	32u	Subsampling factor in Z.
ZSubsampling.options	32u (CSV)	List of available subsampling factors in Z.
SpacingInterval	64f	Uniform spacing interval (mm).
SpacingInterval.min	64f	Minimum spacing interval (mm).
SpacingInterval.max	64f	Maximum spacing interval (mm).
SpacingInterval.used	Bool	Whether or not field is used.
SpacingInterval value	64f	Actual value used.
SpacingIntervalType	32s	Spacing interval type: 0 – Maximum resolution 1 – Balanced 2 – Maximum speed 3 – Custom
SpacingIntervalType.used	Bool	Whether or not this field is used.
Tracking	Section	See *Tracking Child Elements* on the next page.
Material	Section	See *Material Child Elements* on page 207.
Tracheid	Section	See *Tracheid Child Elements* on page 210.
IndependentExposures	Section	See *IndependentExposures Child Elements* on page 209
Custom	Custom	Used by specialized sensors.

Region3D Child Elements

Element	Type	Description
X	64f	X start (mm).
Y	64f	Y start (mm).
Z	64f	Z start (mm).
Width	64f	X extent (mm).
Length	64f	Y extent (mm).

Element	Type	Description
Height	64f	Z extent (mm).
ZAngle	64f	Z Angle start (degrees).
ZAngle.used	Bool	Whether or not this property is used.

Window Child Elements

Element	Type	Description
X	32u	X start (pixels).
Y	32u	Y start (pixels).
Width	32u	X extent (pixels).
Height	32u	Y extent (pixels).

Layout Child Elements

Element	Type	Description
Grid	Grid	Layout grid information.
MultiplexingBank	32u	Multiplexing bank ID
MultiplexingBank.used	32u	Whether or not this field can be specified
MultiplexingBank.value	32u	Actual value used by system

Grid Child Elements

Element	Type	Description
@used	Bool	Whether or not this section is used.
Row	32s	Device row position in grid layout.
Row.value	32s	Value in use by the sensor, useful for determining value when used is false.
Column	32s	Device column position in grid layout.
Column.value	32s	Value in use by the sensor, useful for determining value when used is false.
Direction	32s	Sensor orientation direction.
Direction.value	32s	Value in use by the sensor, useful for determining value when used is false.

Tracking is only available on Gocator 2300 and 2400 series sensors.

Tracking Child Elements

Element	Type	Description
Enabled	Bool	Enables tracking.
Enabled.used	Bool	Whether or not this field is used.
SearchThreshold	64f	Percentage of spots that must be found to remain in track.
Height	64f	Tracking window height (mm).
Height.min	64f	Minimum tracking window height (mm).
Height.max	64f	Maximum tracking window height (mm).

Material Child Elements

Element	Type	Description
Type	32s	Type of Material settings to use. 0 – Custom 1 – Diffuse 3 – Reflective
Type.used	Bool	Determines if the setting's value is currently used.
Type.value	32s	Value in use by the sensor, useful for determining value when used is false.
Type.options	32u (CSV)	List of available material types.
SpotThreshold	32s	Spot detection threshold.
SpotThreshold.min	32s	The minimum spot detection threshold possible.
SpotThreshold.max	32s	The maximum spot detection threshold possible.
SpotThreshold.used	Bool	Determines if the setting's value is currently used.
SpotThreshold.value	32s	Value in use by the sensor, useful for determining value when used is false.
SpotThreshold.readonly	Bool	Whether or not property can be modified. If set, the value should be considered read-only by the client. Only has meaning if "used" is also set.
SpotWidthMax	32s	Spot detection maximum width.
SpotWidthMax.used	Bool	Determines if the setting's value is currently used.
SpotWidthMax.value	32s	Value in use by the sensor, useful for determining value when used is false.
SpotWidthMax.min	32s	Minimum allowed spot detection maximum value.
SpotWidthMax.max	32s	Maximum allowed spot detection maximum value.
SpotSelectionType	32s	Spot selection type 0 – Best. Picks the strongest spot in a given column. 1 – Top. Picks the spot which is most Top/Left on the imager 2 – Bottom. Picks the spot which is most Bottom/Right on the imager 3 – None. All spots are available. This option may not be available in some configurations. 4 – Continuity. Picks the most continuous spot.
SpotSelectionType.used	Bool	Determines if the setting's value is currently used.
SpotSelectionType.value	32s	Value in use by the sensor, useful for determining value when used is false.
SpotSelectionType.options	32s (CSV)	List of available spot selection types.
CameraGainAnalog	64f	Analog camera gain factor.

Element	Type	Description
CameraGainAnalog.used	Bool	Determines if the setting's value is currently used.
CameraGainAnalog.value	64f	Value in use by the sensor, useful for determining value when used is false.
CameraGainAnalog.min	64f	Minimum value.
CameraGainAnalog.max	64f	Maximum value.
CameraGainDigital	64f	Digital camera gain factor.
CameraGainDigital.used	Bool	Determines if the setting's value is currently used.
CameraGainDigital.value	64f	Value in use by the sensor, useful for determining value when used is false.
CameraGainDigital.min	64f	Minimum value.
CameraGainDigital.max	64f	Maximum value.
DynamicSensitivity	64f	Dynamic exposure control sensitivity factor. This can be used to scale the control setpoint.
DynamicSensitivity.used	Bool	Determines if the setting's value is currently used.
DynamicSensitivity.value	64f	Value in use by the sensor, useful for determining value when used is false.
DynamicSensitivity.min	64f	Minimum value.
DynamicSensitivity.max	64f	Maximum value.
DynamicThreshold	32s	Dynamic exposure control threshold. If the detected number of spots is fewer than this number, the exposure will be increased.
DynamicThreshold.used	Bool	Determines if the setting's value is currently used.
DynamicThreshold.value	32s	Value in use by the sensor, useful for determining value when used is false.
DynamicThreshold.min	32s	Minimum value.
DynamicThreshold.max	32s	Maximum value.
SensitivityCompensationEnabled	Bool	Sensitivity compensation toggle. Used in determining analog and digital gain, along with exposure scale.
SensitivityCompensationEnabled.used	Bool	Determines if the setting's value is currently used.
SensitivityCompensationEnabled.value	Bool	Value in use by the sensor, useful for determining value when used is false.
GammaType	32s	Gamma type.
GammaType used	Bool	Determines if the setting's value is currently used.
GammaType value	32s	Value in use by the sensor. Useful for determining value when used is false.
SpotContinuitySorting	Section	See *SpotContinuitySorting Child Elements* on the next page.
SurfaceEncoding	32s	Surface encoding type: 0 – Standard

Element	Type	Description
		1 – Interreflection (advanced use only)
SurfaceEncoding.used	Bool	Determines if the setting's value is currently used.
SurfaceEncoding.value	Bool	Value in use by the sensor, useful for determining value when used is false.
SurfaceEncoding.readonly	Bool	Whether or not property can be modified. If set, the value should be considered read-only by the client. Only has meaning if "used" is also set.
SurfacePhaseFilter	32s	Surface phase filter (correction type) 0 – None 1 – Reflective 2 - Translucent
SurfacePhaseFilter.used	Bool	Determines if the setting's value is currently used.
SurfacePhaseFilter.value	Bool	Value in use by the sensor, useful for determining value when used is false.
ContrastThreshold	32s	Contrast detection threshold.
ContrastThreshold.min	32s	The minimum contrast detection threshold possible.
ContrastThreshold.max	32s	The maximum contrast detection threshold possible.
ContrastThreshold.used	Bool	Determines if the setting's value is currently used.
ContrastThreshold.value	32s	Value in use by the sensor, useful for determining value when used is false.
ContrastThreshold.readonly	Bool	Whether or not property can be modified. If set, the value should be considered read-only by the client. Only has meaning if "used" is also set.

SpotContinuitySorting Child Elements

Element	Type	Description
MinimumSegmentSize	32u	Smallest continuous segment considered in continuity sorting.
SearchWindow/X	32u	X component of continuity sorting search window size.
SearchWindow/Y	32u	Y component of continuity sorting search window size.

IndependentExposures settings are only supported by 3x00 series sensors.

IndependentExposures Child Elements

Element	Type	Description
@used	Bool	Whether this field is used
Enabled	Bool	Whether to allow using separate exposure values for each camera
FrontCameraExposure	64f	The exposure value to use for the front camera
FrontCameraExposure.min	64f	The minimum exposure value possible for front camera

Element	Type	Description
FrontCameraExposure.max	64f	The maximum exposure value possible for back camera
BackCameraExposure	64f	The exposure value to use for the front camera
BackCameraExposure.min	64f	The minimum exposure value possible for front camera
BackCameraExposure.max	64f	The maximum exposure value possible for back camera

Tracheid settings are only supported by Gocator 200 series multi-point sensors.

Tracheid Child Elements

Element	Type	Description
@used	Bool	Whether this field is used
TracheidExposureEnabled	Bool	Whether to use a unique exposure for tracheid capture
TracheidExposure	64f	The exposure value to use for tracheid measurements
TracheidExposure.min	64f	The minimum exposure value possible tracheid measurements
TracheidExposure.max	64f	The maximum exposure value possible for tracheid measurements
Camera0Threshold	32u	The tracheid threshold for camera 0
Camera1Threshold	32u	The tracheid threshold for camera 1

SurfaceGeneration

The SurfaceGeneration element contains settings related to surface generation.

SurfaceGeneration Child Elements

Element	Type	Description
Type	32s	Surface generation type: 0 – Continuous 1 – Fixed length 2 – Variable length 3 – Rotational
Type.options	32s (CSV)	List of available generation types
Type.value	32s	Value in use by the sensor
FixedLength	Section	See *FixedLength* below.
VariableLength	Section	See *VariableLength* on the next page.
Rotational	Section	See *Rotational* on the next page.

FixedLength

FixedLength Child Elements

Element	Type	Description
StartTrigger	32s	Start trigger condition:

Element	Type	Description
		0 – Sequential 1 – Digital input 2 – Software triggered
ExternalInputIndex	32s	Index of external input when Source (above) is set to 1 – Digital Input and connected to a Master. 0 – first digital input 1 – second digital input 2 – third digital input 3 – fourth digital input
ExternalInputIndex.options	32s (CSV)	List of available external input indices.
ExternalInputIndex.used	Bool	Is the external input index in use.
Length	64f	Surface length (mm).
Length.min	64f	Minimum surface length (mm).
Length.max	64f	Maximum surface length (mm).

VariableLength

VariableLength Child Elements

Element	Type	Description
MaxLength	64f	Maximum surface length (mm).
MaxLength.min	64f	Minimum value for maximum surface length (mm).
MaxLength.max	64f	Maximum value for maximum surface length (mm).

Rotational

Rotational Child Elements

Element	Type	Description
Circumference	64f	Circumference (mm).
Circumference.min	64f	Minimum circumference (mm).
Circumference.max	64f	Maximum circumference (mm).

SurfaceSections

SurfaceSections Child Elements

Element	Type	Description
@used	Bool	Whether surface sectioning is enabled.
@xMin	64f	The minimum valid X value to be used for section definition.
@xMax	64f	The maximum valid X value to be used for section definition.
@yMin	64f	The minimum valid Y value to be used for section definition.
@yMax	64f	The maximum valid Y value to be used for section definition.
Section	Collection	A series of Section elements.

Section Child Elements

Element	Type	Description
@id	32s	The ID assigned to the surface section.
@name	String	The name associated with the surface section.
StartPoint	Point64f	The beginning point of the surface section.
EndPoint	Point64f	The end point of the surface section.
CustomSpacingIntervalEnabled	Bool	Indicates whether a user specified custom spacing interval is to be used for the resulting section.
SpacingInterval	64f	The user specified spacing interval.
SpacingInterval.min	64f	The spacing interval limit minimum.
SpacingInterval.max	64f	The spacing interval limit maximum.
SpacingInterval.value	64f	The current spacing interval used by the system.

ProfileGeneration

The ProfileGeneration element contains settings related to profile generation.

ProfileGeneration Child Elements

Element	Type	Description
Type	32s	Profile generation type: 0 – Continuous 1 – Fixed length 2 – Variable length 3 – Rotational
Type.options	32s (CSV)	List of available generation types
Type.value	32s	Value in use by the sensor
FixedLength	Section	See *FixedLength* below.
VariableLength	Section	See *VariableLength* on the next page.
Rotational	Section	See *Rotational* on the next page.

FixedLength

FixedLength Child Elements

Element	Type	Description
StartTrigger	32s	Start trigger condition: 0 – Sequential 1 – Digital input 2 – Software triggered
ExternalInputIndex	32s	Index of external input when Source (above) is set to 1 – Digital Input and connected to a Master. 0 – first digital input 1 – second digital input

Element	Type	Description
		2 – third digital input
		3 – fourth digital input
ExternalInputIndex.options	32s (CSV)	List of available external input indices.
ExternalInputIndex.used	Bool	Is the external input index in use.
Length	64f	Profile length (mm).
Length.min	64f	Minimum profile length (mm).
Length.max	64f	Maximum profile length (mm).

VariableLength

VariableLength Child Elements

Element	Type	Description
MaxLength	64f	Maximum surface length (mm).
MaxLength.min	64f	Minimum value for maximum profile length (mm).
MaxLength.max	64f	Maximum value for maximum profile length (mm).

Rotational

Rotational Child Elements

Element	Type	Description
Circumference	64f	Circumference (mm).
Circumference.min	64f	Minimum circumference (mm).
Circumference.max	64f	Maximum circumference (mm).

PartDetection

PartDetection Child Elements

Element	Type	Description
Enabled	Bool	Enables part detection.
Enabled.used	Bool	Whether or not this field is used.
Enabled value	Bool	Actual value used if not configurable.
MinArea	64f	Minimum area (mm^2).
MinArea.min	64f	Minimum value of minimum area.
MinArea.max	64f	Maximum value of minimum area.
MinArea.used	Bool	Whether or not this field is used.
GapWidth	64f	Gap width (mm).
GapWidth.min	64f	Minimum gap width (mm).
GapWidth.max	64f	Maximum gap width (mm).
GapWidth.used	Bool	Whether or not this field is used.
GapLength	64f	Gap length (mm).

Element	Type	Description
GapLength.min	64f	Minimum gap length (mm).
GapLength.max	64f	Maximum gap length (mm).
GapLength.used	Bool	Whether or not this field is used.
PaddingWidth	64f	Padding width (mm).
PaddingWidth.min	64f	Minimum padding width (mm).
PaddingWidth.max	64f	Maximum padding width (mm).
PaddingWidth.used	Bool	Whether or not this field is used.
PaddingLength	64f	Padding length (mm).
PaddingLength.min	64f	Minimum padding length (mm).
PaddingLength.max	64f	Maximum padding length (mm).
PaddingLength.used	Bool	Whether or not this field is used.
MinLength	64f	Minimum length (mm).
MinLength.min	64f	Minimum value of minimum length (mm).
MinLength.max	64f	Maximum value of minimum length (mm).
MinLength.used	Bool	Whether or not this field is used.
MaxLength	64f	Maximum length (mm).
MaxLength.min	64f	Minimum value of maximum length (mm).
MaxLength.max	64f	Maximum value of maximum length (mm).
MaxLength.used	Bool	Whether or not this field is used.
Threshold	64f	Height threshold (mm).
Threshold.min	64f	Minimum height threshold (mm).
Threshold.max	64f	Maximum height threshold (mm).
ThresholdDirection	32u	Threshold direction: 0 – Above 1 – Below
FrameOfReference	32s	Part frame of reference: 0 – Sensor 1 – Scan 2 – Part
FrameOfReference.used	Bool	Whether or not this field is used.
FrameOfReference.value	32s	Actual value.
IncludeSinglePointsEnabled	Bool	Enables preservation of single data points in Top+Bottom layout
IncludeSinglePointsEnabled.used	Bool	Whether or nto this field is available to be modified
EdgeFiltering	Section	See *EdgeFiltering* on the next page.

EdgeFiltering

EdgeFiltering Child Elements

Element	Type	Description
@used	Bool	Whether or not this section is used.
Enabled	Bool	Enables edge filtering.
PreserveInteriorEnabled	Bool	Enables preservation of interior.
ElementWidth	64f	Element width (mm).
ElementWidth.min	64f	Minimum element width (mm).
ElementWidth.max	64f	Maximum element width (mm).
ElementLength	64f	Element length (mm).
ElementLength.min	64f	Minimum element length (mm).
ElementLength.max	64f	Maximum element length (mm).

PartMatching

The PartMatching element contains settings related to part matching.

PartMatching Child Elements

Element	Type	Description
Enabled	Bool	Enables part matching.
Enabled.used	Bool	Whether or not this field is used.
MatchAlgo	32s	Match algorithm. 0 – Edge points 1 – Bounding Box 2 – Ellipse
Edge	Section	See *Edge* below.
BoundingBox	Section	See *BoundingBox* below.
Ellipse	Section	See *Ellipse* on the next page.

Edge

Edge Child Elements

Element	Type	Description
ModelName	String	Name of the part model to use. Does not include the .mdl extension.
Acceptance/Quality/Min	64f	Minimum quality value for a match.

BoundingBox

BoundingBox Child Elements

Element	Type	Description
ZAngle	64f	Z rotation to apply to bounding box (degrees).
AsymmetryDetectionType	32s	Determine whether to use asymmetry detection and, if enabled, which dimension is the basis of detection. The

Element	Type	Description
		possible values are: 0 – None 1 – Length 2 - Width
Acceptance/Width/Min	64f	Minimum width (mm).
Acceptance/Width/Max	64f	Maximum width (mm).
Acceptance/Width/Tolerance	64f	Width acceptance tolerance value
Acceptance/Width/Tolerance.deprecated	Bool	Whether this tolerance field is deprecated
Acceptance/Length/Min	64f	Minimum length (mm).
Acceptance/Length/Max	64f	Maximum length (mm).
Acceptance/Length/Tolerance	64f	Length acceptance tolerance value
Acceptance/Length/Tolerance.deprecated	Bool	Whether this tolerance field is deprecated
X	64f	X value
X.deprecated	Bool	Whether this X field is deprecated
Y	64f	Y value
Y.deprecated	Bool	Whether this Y field is deprecated
Width	64f	Width value
Width.deprecated	Bool	Whether this width field is deprecated
Length	64f	Length value
Length.deprecated	Bool	Whether this length field is deprecated

Ellipse

Ellipse Child Elements

Element	Type	Description
ZAngle	64f	Z rotation to apply to ellipse (degrees).
AsymmetryDetectionType	32s	Determine whether to use asymmetry detection and, if enabled, which dimension is the basis of detection. The possible values are: 0 – None 1 – Major 2 - Minor
Acceptance/Major/Min	64f	Minimum major length (mm).
Acceptance/Major/Max	64f	Maximum major length (mm).
Acceptance/Major/Tolerance	64f	Major acceptance tolerance value
Acceptance/Major/Tolerance.deprecated	Bool	Whether this tolerance field is deprecated
Acceptance/Minor/Min	64f	Minimum minor length (mm).
Acceptance/Minor/Max	64f	Maximum minor length (mm).

Element	Type	Description
Acceptance/Minor/Tolerance	64f	Minor acceptance tolerance value
Acceptance/Minor/Tolerance.deprecated	Bool	Whether this tolerance field is deprecated
X	64f	X value
X.deprecated	Bool	Whether this X field is deprecated
Y	64f	Y value
Y.deprecated	Bool	Whether this Y field is deprecated
Width	64f	Width value
Width.deprecated	Bool	Whether this width field is deprecated
Length	64f	Length value
Length.deprecated	Bool	Whether this length field is deprecated

Replay

Contains settings related to recording filtering.

RecordingFiltering

RecordingFiltering Child Elements

Element	Type	Description
ConditionCombineType	32s	0 – Any: If any enabled condition is satisfied, the current frame is recorded. 1 – All: All enabled conditions must be satisfied for the current frame to be recorded.
Conditions	Collection	A collection of AnyMeasurement, AnyData, or Measurement conditions.

Conditions/AnyMeasurement

Conditions/AnyMeasurement Elements

Element	Type	Description
Enabled	Bool	Indicates whether the condition is enabled.
Result	32s	The measurement decision criteria to be included in the filter. Possible values are: 0 – Pass 1 – Fail 2 – Valid 3 – Invalid

Conditions/AnyData

Conditions/AnyData Elements

Element	Type	Description
Enabled	Bool	Indicates whether the condition is enabled.
RangeCountCase	32s	The case under which to record data: 0 – Range count at or above threshold of valid data points. 1 – Range count below threshold.
RangeCountThreshold	32u	The threshold for the number of range points that are valid.

Conditions/Measurement

Conditions/Measurement Elements

Element	Type	Description
Enabled	Bool	Indicates whether the condition is enabled.
Result	32s	The measurement decision criteria for the selected ID to be included in the filter. Possible values are: 0 – Pass 1 – Fail 2 – Valid 3 – Invalid
Ids	32s	The ID of the measurement to filter.

Streams/Stream (Read-only)

Streams/Stream Child Elements

Element	Type	Description
Step	32s	The data step of the stream being described. Possible values are: 1 – Video 2 – Range 3 – Surface 4 – Section
Id	32u	The stream ID.
CadenceId	32u	Represents a stage in the data processing pipeline. The greater the number, the farther removed from the initial acquisition stage. One of the following: 0 – Primary 1 – Auxiliary 10 - Diagnostic
DataType	32s	The stream data type 0 – None 4 – Uniform Profile

Element	Type	Description
		16 – Uniform Surface
ColorEncoding	32s	The color encoding type. Only appears for Video stream steps (1). 0 – None 1 – Bayer BGGR 2 – Bayer GBRG 3 – Bayer RGGB 4 – Bayer GRBG
IntensityEnabled	Bool	Whether the stream includes intensity data
Sources	Collection	A collection of Source elements as described below.

Source Child Elements

Element	Type	Description
Id	32s	The ID of the data source. Possible values are: 0 – Top 1 – Bottom 2 – Top Left 3 – Top Right 4 – Top Bottom 5 – Left Right
Capability	32s	The capability of the data stream source. Possible values are: 0 – Full 1 – Diagnostic only 2 - Virtual
Region	Region3d	The region of the given stream source.
AdditionalRegions	Collection	Collection of additional regions (for example, for the second camera).
AdditionalRegions/Region	Region3d	Additional regions.

ToolOptions

The ToolOptions element contains a list of available tool types, their measurements, and settings for related information.

ToolOptions Child Elements

Element	Type	Description
<Tool Names>	Collection	A collection of tool name elements. An element for each tool type is present.

Tool Name Child Elements

Element	Type	Description
@displayName	String	Display name of the tool.
@isCustom	Bool	Reserved for future use.

Element	Type	Description
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
MeasurementOptions	Collection	See *MeasurementOptions* below
FeatureOptions	Collection	See *FeatureOptions* below.
StreamOptions	Collection	See *StreamOptions* on the next page.

MeasurementOptions

MeasurementOptions Child Elements

Element	Type	Description
<Measurement Names>	Collection	A collection of measurement name elements. An element for each measurement is present.

<Measurement Name> Child Elements

Element	Type	Description
@displayName	String	Display name of the tool.
@minCount	32u	Minimum number of instances in a tool.
@maxCount	32u	Maximum number of instances in a tool.

FeatureOptions

FeatureOptions Child Elements

Element	Type	Description
<Feature Names>	Collection	A collection of feature name elements. An element for each measurement is present.

<Feature Name> Child Elements

Element	Type	Description
@displayName	String	Display name of the feature.
@minCount	32u	Minimum number of instances in a tool.
@maxCount	32u	Maximum number of instances in a tool.
@dataType	String	The data type of the feature. One of: – PointFeature – LineFeature – CircleFeature – PlaneFeature

StreamOptions

StreamOptions Child Elements

Element	Type	Description
@step	32s	The data step of the stream being described. Possible values are: 1 – Video 2 – Range 3 – Surface 4 – Section
@ids	CSV	A list representing the available IDs associated with the given step.

Tools

The Tools element contains measurement tools. The following sections describe each tool and its available measurements.

Tools Child Elements

Element	Type	Description
@options	String (CSV)	A list of the tools available in the currently selected scan mode.
<ToolType>	Section	An element for each added tool.

Profile Types

The following types are used by various measurement tools.

ProfileFeature

An element of type ProfileFeature defines the settings for detecting a feature within an area of interest.

ProfileFeature Child Elements

Element	Type	Description
Type	32s	Determine how the feature is detected within the area: 0 – Max Z 1 – Min Z 2 – Max X 3 – Min X 4 – Corner 5 – Average 6 – Rising Edge 7 – Falling Edge 8 – Any Edge 9 – Top Corner 10 – Bottom Corner 11 – Left Corner

Element	Type	Description
		12 – Right Corner 13 – Median
RegionEnabled	Bool	Indicates whether feature detection applies to the defined Region or to the entire active area.
Region	ProfileRegion2D	Element for feature detection area.

ProfileLine

An element of type ProfileLine defines measurement areas used to calculate a line.

ProfileLine Child Elements

Element	Type	Description
RegionCount	32s	Count of the regions.
Regions	(Collection)	The regions used to calculate a line. Contains one or two Region elements of type ProfileRegion2D, with RegionEnabled fields for each.

ProfileRegion2d

An element of type ProfileRegion2d defines a rectangular area of interest.

ProfileRegion2d Child Elements

Element	Type	Description
X	64f	Setting for profile region X position (mm).
Z	64f	Setting for profile region Z position (mm).
Width	64f	Setting for profile region width (mm).
Height	64f	Setting for profile region height (mm).

Geometric Feature Types

The Geometric Feature type is used by various measurement tools.

Feature Child Elements

Element	Type	Description
@id	32s	The identifier of the geometric feature. -1 if unassigned.
@dataType	String	The data type of the feature. One of: – PointFeature – LineFeature
@type	String	Type name of feature.
Name	String	The display name of the feature.
Enabled	Bool	Whether the given feature output is enabled.
Parameters	Collection	Collection of GdkParam elements.

Parameter Types

The following types are used by internal and custom (user-created) GDK-based tools.

GDK Parameter Child Elements

Element	Type	Description
@label	String	Parameter label.
@type	String	Type of parameter. It is one of the following (see tables below for elements found in each type): - Bool - Int - Float - ProfileRegion - SurfaceRegion2d - SurfaceRegion3d - GeometricFeature
@options	Variant (CSV)	Options available for this parameter.
@optionNames	String (CSV)	Names

GDK Parameter Bool Type

Element	Type	Description
	Bool	Boolean value of parameter.

GDK Parameter Int Type

Element	Type	Description
	32s	Integer value of parameter of integer type.

GDK Parameter Float Type

Element	Type	Description
	64f	Floating point value of parameter.

GDK Parameter String Type

Element	Type	Description
	String	String value of parameter.

GDK Parameter Profile Region Type

Element	Type	Description
X	64f	X value of region.
Z	64f	Z value of region.
Width	64f	Width value of region.
Height	64f	Height value of region.

GDK Parameter Surface Region 2D Type

Element	Type	Description
X	64f	X value of region.
X	64f	X value of region.

Element	Type	Description
Y	64f	Y value of region.
Width	64f	Width value of region.
Length	64f	Length value of region.

GDK Parameter Surface Region 3D Type

Element	Type	Description
X	64f	X value of region.
Y	64f	Y value of region.
Z	64f	Z value of region.
Width	64f	Width value of region.
Length	64f	Length value of region.
Height	64f	Height value of region.
ZAngle	64f	ZAngle value of region.

GDK Parameter Geometric Feature Type

Element	Type	Description
	32s	Geometric feature Id for parameter.

ProfileArea

A ProfileArea element defines settings for a profile area tool and one or more of its measurements.

ProfileArea Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.
Name	String	Tool name.
Features	Collection	Collection of geometric feature outputs available in the tool. See *ProfileArea* above.
Source	32s	Profile source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
StreamOptions	Collection	A collection of StreamOptions elements.
Stream\Step	32s	The stream source step. Possible values are:

Element	Type	Description
		1 – Video 2 – Range 3 – Surface 4 – Section
Stream\Id	32u	The stream source ID.
Type	Boolean	Area to measure: 0 – Object (convex shape above the baseline) 1 – Clearance (concave shape below the baseline)
Type.used	Boolean	Whether or not field is used.
Baseline	Boolean	Baseline type: 0 – X-axis 1 – Line
Baseline.used	Boolean	Whether or not field is used.
RegionEnabled	Boolean	If enabled, the defined region is used for measurements. Otherwise, the full active area is used.
Region	ProfileRegion2d	Measurement region.
Line	ProfileLine	Line definition when Baseline is set to Line.
Measurements\Area	Area tool measurement	Area measurement.
Measurements\CentroidX	Area tool measurement	CentroidX measurement.
Measurements\CentroidZ	Area tool measurement	CentroidZ measurement.
Features\CenterPoint	GeometricFeature	CenterPoint PointFeature.

Area Tool Measurement

Element	Type	Description
@id	32s	Measurement ID. Optional (measurement disabled if not set).
Name	String	Measurement name.
Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state 0 – Disable

Element	Type	Description
		1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.
DecisionMax	64f	Maximum decision threshold.

ProfileBoundingBox

A ProfileBoundingBox element defines settings for a profile bounding box tool and one or more of its measurements.

ProfileBoundingBox Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.
Name	String	Tool name.
Features	Collection	Collection of geometric feature outputs available in the tool. See *ProfileBoundingBox* above.
Source	32s	Profile source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
StreamOptions	Collection	A collection of StreamOptions elements.
Stream\Step	32s	The stream source step. Possible values are: 1 – Video 2 – Range 3 – Surface 4 – Section
Stream\Id	32u	The stream source ID.
RegionEnabled	Bool	Whether or not to use the region. If the region is disabled, all available data is used.
Region	ProfileRegion2d	Measurement region.
Measurements\X	Bounding Box tool	X measurement.

Element	Type	Description
	measurement	
Measurements\Z	Bounding Box tool measurement	Z measurement.
Measurements\Width	Bounding Box tool measurement	Width measurement.
Measurements\Height	Bounding Box tool measurement	Height measurement.
Measurements\GlobalX	Bounding Box tool measurement	GlobalX measurement
Measurements\GlobalY	Bounding Box tool measurement	GlobalY measurement
Measurements\GlobalAngle	Bounding Box tool measurement	GlobalAngle measurement
Features\CenterPoint	GeometricFeature	CenterPoint PointFeature.
Features\CornerPoint	GeometricFeature	CornerPoint PointFeature.

Bounding Box Tool Measurement

Element	Type	Description
@id	32s	Measurement ID. Optional (measurement disabled if not set).
Name	String	Measurement name.
Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state 0 – Disable 1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.
DecisionMax	64f	Maximum decision threshold.

ProfileCircle

A ProfileCircle element defines settings for a profile circle tool and one or more of its measurements.

ProfileCircle Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.
Name	String	Tool name.
Features	Collection	Collection of geometric feature outputs available in the tool. See *ProfileCircle* above.
Source	32s	Profile source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
StreamOptions	Collection	A collection of StreamOptions elements.
Stream\Step	32s	The stream source step. Possible values are: 1 – Video 2 – Range 3 – Surface 4 – Section
Stream\Id	32u	The stream source ID.
RegionEnabled	Bool	Whether or not to use the region. If the region is disabled, all available data is used.
Region	ProfileRegion2d	Measurement region.
Measurements\X	Circle tool measurement	X measurement.
Measurements\Z	Circle tool measurement	Z measurement.
Measurements\Radius	Circle tool measurement	Radius measurement.
Measurements\StdDev	CircleMeasurement	Standard deviation measurement
Measurements\MinError	CircleMeasurement	Minimum error measurement
Measurements\MinErrorX	CircleMeasurement	Minimum error X measurement
Measurements\MinErrorZ	CircleMeasurement	Minimum error Z measurement
Measurements\MaxError	CircleMeasurement	Maximum error measurement
Measurements\MaxErrorX	CircleMeasurement	Maximum error X measurement

Element	Type	Description
Measurements\MaxErrorZ	CircleMeasurement	Maximum error Z measurement
Features\CenterPoint	GeometricFeature	CenterPoint PointFeature.

Circle Tool Measurement

Element	Type	Description
@id	32s	Measurement ID. Optional (measurement disabled if not set).
Name	String	Measurement name.
Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state 0 – Disable 1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.
DecisionMax	64f	Maximum decision threshold.

ProfileDimension

A ProfileDimension element defines settings for a profile dimension tool and one or more of its measurements.

ProfileDimension Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.

Element	Type	Description
Name	String	Tool name.
Features	Collection	Not used.
Source	32s	Profile source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
StreamOptions	Collection	A collection of StreamOptions elements.
Stream\Step	32s	The stream source step. Possible values are: 1 – Video 2 – Range 3 – Surface 4 – Section
Stream\Id	32u	The stream source ID.
RefFeature	ProfileFeature	Reference measurement region.
Feature	ProfileFeature	Measurement region.
Measurements\Width	Dimension tool measurement	Width measurement.
Measurements\Height	Dimension tool measurement	Height measurement.
Measurements\Distance	Dimension tool measurement	Distance measurement.
Measurements\CenterX	Dimension tool measurement	CenterX measurement.
Measurements\CenterZ	Dimension tool measurement	CenterZ measurement.

Dimension Tool Measurement

Element	Type	Description
@id	32s	Measurement ID. Optional (measurement disabled if not set).
Name	String	Measurement name.
Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable

Element	Type	Description
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state 0 – Disable 1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.
DecisionMax	64f	Maximum decision threshold.
Absolute *(Width and Height measurements only)*	Boolean	Setting for selecting absolute or signed result: 0 – Signed 1 – Absolute

ProfileGroove

A ProfileGroove element defines settings for a profile groove tool and one or more of its measurements.

The profile groove tool is dynamic, meaning that it can contain multiple measurements of the same type in the Measurements element.

ProfileGroove Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.
Name	String	Tool name.
Features	Collection	Not used.
Source	32s	Profile source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
StreamOptions	Collection	A collection of StreamOptions elements.
Stream\Step	32s	The stream source step. Possible values are:

Element	Type	Description
		1 – Video 2 – Range 3 – Surface 4 – Section
Stream\Id	32u	The stream source ID.
Shape	32s	Shape: 0 – U-shape 1 – V-shape 2 – Open
MinDepth	64f	Minimum depth.
MinWidth	64f	Minimum width.
MaxWidth	64f	Maximum width.
RegionEnabled	Bool	Whether or not to use the region. If the region is disabled, all available data is used.
Region	ProfileRegion2d	Measurement region.
Measurements\X	Groove tool measurement	X measurement.
Measurements\Z	Groove tool measurement	Z measurement.
Measurements\Width	Groove tool measurement	Width measurement.
Measurements\Depth	Groove tool measurement	Depth measurement.

Groove Tool Measurement

Element	Type	Description
@id	32s	Measurement ID. Optional (measurement disabled if not set).
Name	String	Measurement name.
Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state

Element	Type	Description
		0 – Disable 1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.
DecisionMax	64f	Maximum decision threshold.
SelectType	32s	Method of selecting a groove when multiple grooves are found: 0 – Max depth 1 – Ordinal, from left 2 – Ordinal, from right
SelectIndex	32s	Index when SelectType is set to 1 or 2.
Location *(X and Z measurements only)*	32s	Setting for groove location to return from: 0 – Bottom 1 – Left corner 2 – Right corner

ProfileIntersect

A ProfileIntersect element defines settings for a profile intersect tool and one or more of its measurements.

ProfileIntersect Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.
Name	String	Tool name.
Features	Collection	Collection of geometric feature outputs available in the tool. See *ProfileIntersect* above.
Source	32s	Profile source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.

Element	Type	Description
RefType	32s	Reference line type: 0 – Fit 1 – X Axis
StreamOptions	Collection	A collection of StreamOptions elements.
Stream\Step	32s	The stream source step. Possible values are: 1 – Video 2 – Range 3 – Surface 4 – Section
Stream\Id	32u	The stream source ID.
RefLine	ProfileLine	Definition of reference line. Ignored if RefType is not 0.
Line	ProfileLine	Definition of line.
Measurements\X	Intersect tool measurement	X measurement.
Measurements\Z	Intersect tool measurement	Z measurement.
Measurements\Angle	Intersect tool measurement	Angle measurement.
Features\IntersectPoint	GeometricFeature	IntersectPoint PointFeature.
Features\Line	GeometricFeature	Line LineFeature.
Features\BaseLine	GeometricFeature	BaseLine LineFeature.

Intersect Tool Measurement

Element	Type	Description
@id	32s	Measurement ID. Optional (measurement disabled if not set).
Name	String	Measurement name.
Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state 0 – Disable

Element	Type	Description
		1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.
DecisionMax	64f	Maximum decision threshold.
Absolute *(Angle measurement only)*	Boolean	Setting for selecting the angle range: 0 – A range of -90 to 90 degrees is used. 1 – A range of 0 to 180 degrees is used.

ProfileLine

A ProfileLine element defines settings for a profile line tool and one or more of its measurements.

ProfileLine Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.
Name	String	Tool name.
Features	Collection	Collection of geometric feature outputs available in the tool. See *ProfileLine* above.
Source	32s	Profile source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
StreamOptions	Collection	A collection of StreamOptions elements.
Stream\Step	32s	The stream source step. Possible values are: 1 – Video 2 – Range 3 – Surface 4 – Section
Stream\Id	32u	The stream source ID.
RegionEnabled	Bool	Whether or not to use the region. If the region is disabled, all available data is used.

Element	Type	Description
Region	ProfileRegion2d	Measurement region.
FittingRegions	ProfileLine	ProfileLine describing up to 2 regions to fit to.
FittingRegionsEnabled	Bool	Whether the fitting regions are enabled.
Measurements\StdDev	Line tool measurement	StdDev measurement.
Measurements\MaxError	Line tool measurement	MaxError measurement.
Measurements\MinError	Line tool measurement	MinError measurement.
Measurements\Percentile	Line tool measurement	Percentile measurement.
Measurements\Offset	Line tool measurement	Offset measurement.
Measurements\Angle	Line tool measurement	Angle measurement.
Measurements\MinErrorX	Line tool measurement	Minimum Error in Z measurement.
Measurements\MinErrorZ	Line tool measurement	Minimum Error in Z measurement.
Measurements\MaxErrorX	Line tool measurement	Maximum Error in X measurement.
Measurements\MaxErrorZ	Line tool measurement	Maximum Error in Z measurement.
Features\Line	GeometricFeature	Line LineFeature.
Features\ErrorMinPoint	GeometricFeature	ErrorMinPoint PointFeature.
Features\ErrorMaxPoint	GeometricFeature	ErrorMaxPoint PointFeature.

Line Tool Measurement

Element	Type	Description
@id	32s	Measurement ID. Optional (measurement disabled if not set).
Name	String	Measurement name.
Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state 0 – Disable 1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.

Element	Type	Description
DecisionMax	64f	Maximum decision threshold.
Percent	64f	Error percentile.
(Percentile measurement only)		

ProfilePanel

A ProfilePanel element defines settings for a profile panel tool and one or more of its measurements.

ProfilePanel Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.
Name	String	Tool name.
Features	Collection	Not used.
Source	32s	Profile source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
StreamOptions	Collection	A collection of StreamOptions elements.
Stream\Step	32s	The stream source step. Possible values are: 1 – Video 2 – Range 3 – Surface 4 – Section
Stream\Id	32u	The stream source ID.
RefSide	32s	Setting for reference side to use.
MaxGapWidth	64f	Setting for maximum gap width (mm).
LeftEdge	ProfilePanelEdge	Element for left edge configuration.
RightEdge	ProfilePanelEdge	Element for right edge configuration.
Measurements\Gap	Gap/Flush measurement	Gap measurement.
Measurements\Flush	Gap/Flush measurement	Flush measurement.

Element	Type	Description
Measurements\LeftGapX	Gap/Flush measurement	Left Gap X measurement.
Measurements\LeftGapZ	Gap/Flush measurement	Left Gap Z measurement.
Measurements\LeftFlushX	Gap/Flush measurement	Left Flush X measurement.
Measurements\LeftFlushZ	Gap/Flush measurement	Left Flush Z measurement.
Measurements\LeftSurfaceAngle	Gap/Flush measurement	Left Surface Angle measurement.
Measurements\RightGapX	Gap/Flush measurement	Right Gap X measurement.
Measurements\RightGapZ	Gap/Flush measurement	Right Gap Z measurement.
Measurements\RightFlushX	Gap/Flush measurement	Right Flush X measurement.
Measurements\RightFlushZ	Gap/Flush measurement	Right Flush Z measurement.
Measurements\RightSurfaceAngle	Gap/Flush measurement	Right Surface Angle measurement.

ProfilePanelEdge

Element	Type	Description
EdgeType	32s	Edge type: 0 – Tangent 1 – Corner
MinDepth	64f	Minimum depth.
MaxVoidWidth	64f	Maximum void width.
SurfaceWidth	64f	Surface width.
SurfaceOffset	64f	Surface offset.
NominalRadius	64f	Nominal radius.
EdgeAngle	64f	Edge angle.
RegionEnabled	Bool	Whether or not to use the region. If the region is disabled, all available data is used.
Region	ProfileRegion2d	Edge region.

Gap/Flush Measurement

Element	Type	Description
@id	32s	Measurement ID. Optional (measurement disabled if not set).

Element	Type	Description
Name	String	Measurement name.
Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state 0 – Disable 1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.
DecisionMax	64f	Maximum decision threshold.
Axis *(Gap measurement only)*	32s	Measurement axis: 0 – Edge 1 – Surface 2 – Distance
Absolute *(Flush measurement only)*	Boolean	Setting for selecting absolute or signed result: 0 – Signed 1 – Absolute

ProfilePosition

A ProfilePosition element defines settings for a profile position tool and one or more of its measurements.

ProfilePosition Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool

Element	Type	Description
@id	32s	The tool's ID.
Name	String	Tool name.
Features	Collection	Collection of geometric feature outputs available in the tool. See *ProfilePosition* on the previous page.
Source	32s	Profile source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
StreamOptions	Collection	A collection of StreamOptions elements.
Stream\Step	32s	The stream source step. Possible values are: 1 – Video 2 – Range 3 – Surface 4 – Section
Stream\Id	32u	The stream source ID.
Feature	ProfileFeature	Element for feature detection.
Measurements\X	Position tool measurement	X measurement.
Measurements\Z	Position tool measurement	Z measurement.
Features\Point	GeometricFeature	Point PointFeature

Position Tool Measurement

Element	Type	Description
id (attribute)	32s	Measurement ID. Optional (measurement disabled if not set).
Name	String	Measurement name.
Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state

Element	Type	Description
		0 – Disable 1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.
DecisionMax	64f	Maximum decision threshold.

ProfileRoundCorner

A ProfileRoundCorner element defines settings for a profile round corner tool and one or more of its measurements.

ProfileRoundCorner Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.
Name	String	Tool name.
Features	Collection	Not used.
Source	32s	Profile source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
StreamOptions	Collection	A collection of StreamOptions elements.
Stream\Step	32s	The stream source step. Possible values are: 1 – Video 2 – Range 3 – Surface 4 – Section
Stream\Id	32u	The stream source ID.
RefDirection	32s	Setting for reference side to use: 0 – Left 1 – Right

Element	Type	Description
Edge	ProfilePanelEdge	Element for edge configuration
Measurements\X	Round Corner tool measurement	X measurement.
Measurements\Z	Round Corner tool measurement	Z measurement.
Measurements\Angle	Round Corner tool measurement	Angle measurement.
Features\CenterPoint	Geometric Feature	Circle Center PointFeature.
Features\EdgePoint	Geometric Feature	Edge PointFeature.

ProfilePanelEdge

Element	Type	Description
EdgeType	32s	Edge type: 0 – Tangent 1 – Corner
MinDepth	64f	Minimum depth.
MaxVoidWidth	64f	Maximum void width.
SurfaceWidth	64f	Surface width.
SurfaceOffset	64f	Surface offset.
NominalRadius	64f	Nominal radius.
EdgeAngle	64f	Edge angle.
RegionEnabled	Bool	Whether or not to use the region. If the region is disabled, all available data is used.
Region	ProfileRegion2d	Edge region.

Round Corner Tool Measurement

Element	Type	Description
id (attribute)	32s	Measurement ID. Optional (measurement disabled if not set).
Name	String	Measurement name.
Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state 0 – Disable 1 – Enable

Element	Type	Description
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.
DecisionMax	64f	Maximum decision threshold.

ProfileStrip

A ProfileStrip element defines settings for a profile strip tool and one or more of its measurements.

The profile strip tool is dynamic, meaning that it can contain multiple measurements of the same type in the Measurements element.

ProfileStrip Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.
Name	String	Tool name.
Features	Collection	Not used.
Source	32s	Profile source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
StreamOptions	Collection	A collection of StreamOptions elements.
Stream\Step	32s	The stream source step. Possible values are: 1 – Video 2 – Range 3 – Surface 4 – Section
Stream\Id	32u	The stream source ID.
BaseType	32s	Setting for the strip type:

Element	Type	Description
		0 – None 1 – Flat
LeftEdge	Bitmask	Setting for the left edge conditions: 1 – Raising 2 – Falling 4 – Data End 8 – Void
RightEdge	Bitmask	Setting for the right edge conditions: 1 – Raising 2 – Falling 4 – Data End 8 – Void
TiltEnabled	Boolean	Setting for tilt compensation: 0 – Disabled 1 – Enabled
SupportWidth	64f	Support width of edge (mm).
TransitionWidth	64f	Transition width of edge (mm).
MinWidth	64f	Minimum strip width (mm).
MinHeight	64f	Minimum strip height (mm).
MaxVoidWidth	64f	Void max (mm).
Region	ProfileRegion2d	Region containing the strip.
Measurements\X	Strip tool measurement	X measurement.
Measurements\Z	Strip tool measurement	Z measurement.
Measurements\Width	Strip tool measurement	Width measurement.
Measurements\Height	Strip tool measurement	Width measurement.

Strip Tool Measurement

Element	Type	Description
@id	32s	Measurement ID. Optional (measurement disabled if not set).
Name	String	Measurement name.
Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable

Element	Type	Description
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state 0 – Disable 1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.
DecisionMax	64f	Maximum decision threshold.
SelectType	32s	Method of selecting a groove when multiple grooves are found: 0 – Best 1 – Ordinal, from left 2 – Ordinal, from right
SelectIndex	32s	Index when SelectType is set to 1 or 2.
Location *(X, Z, and Height measurements only)*	32s	Setting for groove location to return from: 0 – Left 1 – Right 2 – Center

Script

A Script element defines settings for a script measurement.

Script Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.
Name	String	Tool name.
Code	String	Script code.
Measurements\Output	(Collection)	Dynamic list of Output elements.

Output

Element	Type	Description
@id	32s	Measurement ID. Optional (measurement disabled if not set).
Name	String	Measurement name.

Tool (type FeatureDimension)

A Tool element of type FeatureDimension defines settings for a feature dimension tool and one or more of its measurements.

Tool Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.
@type	String	Type name of the tool.
@version	String	Version string for custom tool.
Name	String	Tool name.
Source	32s	Surface source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Y	String (CSV)	The Y measurements (IDs) used for anchoring.
Anchor\Y.options	String (CSV)	The Y measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
Parameters\RefPoint	GdkParamGeometricFeature	Reference point feature.
Parameters\Feature	GdkParamGeometricFeature	Reference feature.
Measurements\Measurement @type=Width	Dimension Measurement	Width measurement.
Measurements\Measurement @type=Length	Dimension Measurement	Length measurement.
Measurements\Measurement @type=Height	Dimension Measurement	Width measurement.

Element	Type	Description
Measurements\Measurement @type=Distance	Dimension Measurement	Distance measurement.
Measurements\Measurement @type=PlaneDistance	Dimension Measurement	Plane distance measurement.

Dimension Measurement Child Elements

@id	32s	Measurement ID. Optional (measurement disabled if not set).
@type	String	Type name of measurement.
Name	String	Measurement name.
Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state 0 – Disable 1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.
DecisionMax	64f	Maximum decision threshold.
Parameters\WidthAbsolute *(Width measurement only)*	GdkParamBool	Absolute width enabled boolean.
Parameters\LengthAbsolute *(Length measurement only)*	GdkParamBool	Absolute length enabled boolean.
Parameters\HeightAbsolute *(Height measurement only)*	GdkParamBool	Absolute height enabled boolean.

Tool (type FeatureIntersect)

A Tool element of type FeatureIntersect defines settings for a feature intersection tool and one or more of its measurements.

Tool Child Elements

Element	Type	Description
@isCustom	Bool	Reserved for future use.
@format	32s	Format type of the tool: 0 – Standard built-in tool 1 – GDK user-defined tool 2 – Internal GDK tool
@id	32s	The tool's ID.
@type	String	Type name of the tool.
@version	String	Version string for custom tool.
Name	String	Tool name.
Source	32s	Surface source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Y	String (CSV)	The Y measurements (IDs) used for anchoring.
Anchor\Y.options	String (CSV)	The Y measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
Parameters\Line	GdkParamGeometricFeature	Line feature input.
Parameters\RefLine	GdkParamGeometricFeature	Reference line feature input.
Measurements\Measurement @type=\X	Intersect Measurement	X measurement.
Measurements\Measurement @type=Y	Intersect Measurement	Y measurement.
Measurements\Measurement @type=Z	Intersect Measurement	Z measurement.
Measurements\Measurement @type=Angle	Intersect Measurement	Angle measurement.
Features\IntersectPoint	GDK Feature	Intersect point feature.

Intersect Measurement Child Elements

@id	32s	Measurement ID. Optional (measurement disabled if not set).
@type	String	Type name of measurement.
Name	String	Measurement name.

Enabled	Boolean	Measurement enable state: 0 – Disable 1 – Enable
HoldEnabled	Boolean	Output hold enable state: 0 – Disable 1 – Enable
SmoothingEnabled	Boolean	Smoothing enable state: 0 – Disable 1 – Enable
PreserveInvalidsEnabled	Boolean	Preserve invalid measurements enable state 0 – Disable 1 – Enable
SmoothingWindow	32u	Smoothing window.
Scale	64f	Output scaling factor.
Offset	64f	Output offset factor.
DecisionMin	64f	Minimum decision threshold.
DecisionMax	64f	Maximum decision threshold.
Parameters\AngleRange	GdkParamInt	Angle range option choice. Is one of: 0 – -180 To 180 1 – 0 To 360

Custom

A Custom element defines settings for a user-created GDK-based tool and one or more of its measurements.

Custom Child Elements

Element	Type	Description
@type	String	Type name of the tool.
@version	String	Version string for custom tool.
Name	String	Tool name.
Source	32s	Surface source.
Anchor\X	String (CSV)	The X measurements (IDs) used for anchoring.
Anchor\X.options	String (CSV)	The X measurements (IDs) available for anchoring.
Anchor\Y	String (CSV)	The Y measurements (IDs) used for anchoring.
Anchor\Y.options	String (CSV)	The Y measurements (IDs) available for anchoring.
Anchor\Z	String (CSV)	The Z measurements (IDs) used for anchoring.
Anchor\Z.options	String (CSV)	The Z measurements (IDs) available for anchoring.
Parameters	GDK Parameter	Collection of parameters. The element name in the job file is the name of the parameter.

Element	Type	Description
Measurements	GDK Measurement	Collection of measurements.
Features	GDK Feature	Collection of features.

Output

The Output element contains an Ethernet sub-element.

For all sub-elements, the source identifiers used for measurement outputs correspond to the measurement identifiers defined in each tool's Measurements element. For example, in the following XML, in the options attribute of the Measurements element, 2 and 3 are the identifiers of measurements that are enabled and available for output. The value of the Measurements element (that is, 2) means that only the measurement with id 2 () will be sent to output.

```
<Output>
  <Ethernet>     ...
    <Measurements options="2,3">2</Measurements>
```

Ethernet

The Ethernet element defines settings for Ethernet output.

In the Ethernet element, the source identifiers used for video, range, profile, and surface output, as well as range, profile, and surface intensity outputs, correspond to the *sensor* that provides the data. For example, in the XML below, the *options* attribute of the element shows that only two sources are available (see the table below for the meanings of these values). The value in this element—0—indicates that only data from that source will be sent to output.

```
    ...
```

Ethernet Child Elements

Element	Type	Description
Ethernet.used	Boolean	Indicates if the output is available on the sensor.
Protocol	32s	Ethernet protocol: 0 – Gocator 1 – Modbus 2 – EtherNet/IP 3 – ASCII 4 – PROFINET
Protocol.options	32s (CSV)	List of available protocol options.
TimeoutEnabled	Boolean	Enable or disable auto-disconnection timeout. Applies only to the Gocator protocol.
Timeout	64f	Disconnection timeout (seconds). Used when TimeoutEnabled is true and the Gocator protocol is selected.

Element	Type	Description
Ascii	Section	See *Ascii* on the next page.
EIP	Section	See *EIP* on page 253.
Modbus	Section	See *Modbus* on page 253.
Profinet	Section	See *Ethernet* on the previous page.
Ptp	Section	Enable or disable Precision Time Protocol support.
Videos	32s (CSV)	Selected video sources: 0 – Top 1 – Bottom 2 – Top left 3 – Top right
Videos.options	32s (CSV)	List of available video sources (see above).
Ranges	32s (CSV)	Selected range sources: 0 – Top 1 – Bottom 2 – Top left 3 – Top right
Ranges.options	32s (CSV)	List of available range sources (see above).
Profiles	32s (CSV)	Selected profile sources: 0 – Top 1 – Bottom 2 – Top left 3 – Top right
Profiles.options	32s (CSV)	List of available profile sources (see above).
Surfaces	32s (CSV)	Selected surface sources: 0 – Top 1 – Bottom 2 – Top left 3 – Top right
Surfaces.options	32s (CSV)	List of available surface sources (see above).
SurfaceSections	32s (CSV)	Selected surface section sources.
SurfaceSections.options	32s (CSV)	List of available surface section sources.
RangeIntensities	32s (CSV)	Selected range intensity sources. 0 – Top 1 – Bottom 2 – Top left 3 – Top right

Element	Type	Description
RangeIntensities.options	32s (CSV)	List of available range intensity sources (see above).
ProfileIntensities	32s (CSV)	Selected profile intensity sources. 0 – Top 1 – Bottom 2 – Top left 3 – Top right
ProfileIntensities.options	32s (CSV)	List of available profile intensity sources (see above).
SurfaceIntensities	32s (CSV)	Selected surface intensity sources.
SurfaceIntensities.options	32s (CSV)	List of available surface intensity sources (see above).
SurfaceSectionIntensities	32s (CSV)	Selected surface section intensity sources
SurfaceSectionIntensities.options	32s (CSV)	List of available surface section intensity sources.
Tracheids	32s (CSV)	Selected tracheid sources.
Tracheids.options	32s (CSV)	List of available tracheid sources.
Measurements	32u (CSV)	Selected measurement sources.
Measurements.options	32u (CSV)	List of available measurement sources.
Events	32u (CSV)	Selected events
Events.Options	32u (CSV)	CSV list of possible event options: 0 – Exposure Begins 1 – Exposure Ends
Features	32u (CSV)	Selected feature sources.
Features.options	32u (CSV)	List of available feature sources.
ToolData	32u (CSV)	Selected tool data sources.
ToolData.options	32u (CSV)	List of available tool data sources.

Ascii

Ascii Child Elements

Element	Type	Description
Operation	32s	Operation mode: 0 – Asynchronous 1 – Polled
ControlPort	32u	Control service port number.
HealthPort	32u	Health service port number.
DataPort	32u	Data service port number.
Delimiter	String	Field delimiter.
Terminator	String	Line terminator.
InvalidValue	String	String for invalid output.

Element	Type	Description
CustomDataFormat	String	Custom data format.
CustomFormatEnabled	Bool	Enables custom data format.
StandardFormatMode	32u	The formatting mode used if not a custom format: 0 – Standard 1 – Standard with Stamp

EIP

EIP Child Elements

Element	Type	Description
BufferEnabled	Bool	Enables EtherNet/IP output buffering.
EndianOutputType	32s	Endian output type: 0 – Big endian 1 – Little endian
ImplicitOutputEnabled	Bool	Enables Implict (I/O) Messaging.
ImplicitTriggerOverride	32s	Override requested trigger type by client: 0 – No override 1 – Cyclic 2 – Change of State

Modbus

Modbus Child Elements

Element	Type	Description
BufferEnabled	Bool	Enables Modbus output buffering.

Transform

The transformation component contains information about the physical system setup that is used to:

- Transform data from sensor coordinate system to another coordinate system (e.g., world)
- Define the travel offset (Y offset) between sensors for staggered operation

> Gocator 200 sensors do not support direct encoder input.

You can access the Transform component of the active job as an XML file, either using path notation, via "_live.job/transform.xml", or directly via "_live.tfm".

You can access the Transform component in user-created job files in non-volatile storage, for example, "productionRun01.job/transform.xml". You can only access transformations in user-created job files using path notation.

See the following sections for the elements contained in this component.

Transformation Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<Transform version="100">
  <EncoderResolution>1</EncoderResolution>
  <Speed>100</Speed>
  <Devices>
    <Device role="0">
      <X>-2.3650924829</X>
      <Y>0.0</Y>
      <Z>123.4966803469</Z>
      <XAngle>5.7478302588</XAngle>
      <YAngle>3.7078302555</XAngle>
      <ZAngle>2.7078302556</XAngle>
    </Device>
    <Device id="1">
      <X>0</X>
      <Y>0.0</Y>
      <Z>123.4966803469</Z>
      <XAngle>5.7478302588</XAngle>
      <YAngle>3.7078302555</XAngle>
      <ZAngle>2.7078302556</XAngle>
    </Device>
  </Devices>
</Transform>
```

The Transform element contains the alignment record for sensor.

Transform Child Elements

Element	Type	Description
@version	32u	Major transform version (100).
@versionMinor	32u	Minor transform version (0).
EncoderResolution	64f	Encoder Resolution (mm/tick).
Speed	64f	Travel Speed (mm/s).
Devices	(Collection)	Contains two Device elements.

Device

A Device element defines the transformation for a sensor. There is one entry element per sensor, identified by a unique role attribute (0 for main and 1 for buddy):

Device Child Elements

Element	Type	Description
@role	32s	Role of device described by this section: 0 – Main

Element	Type	Description
		1 – Buddy
X	64f	Translation on the X axis (mm).
Y	64f	Translation on the Y axis (mm).
Z	64f	Translation on the Z axis (mm).
XAngle	64f	Rotation around the X axis (degrees).
YAngle	64f	Rotation around the Y axis (degrees).
ZAngle	64f	Rotation around the Z axis (degrees).

The rotation (counter-clockwise in the X-Z plane) is performed before the translation.

Protocols

Gocator supports protocols for communicating with sensors over Ethernet (TCP/IP) and serial output. For a protocol to output data, it must be enabled and configured in the active job.

> If you switch jobs or make changes to a job using the SDK or a protocol (from a PLC), the switch or changes are not automatically displayed in the web interface: you must refresh the browser to see these.

Protocols available over Ethernet

- Gocator
- Modbus
- EtherNet/IP
- ASCII

For an overview of the Ethernet ports used by sensors, see *Required Ports* on page 55.

Gocator Protocol

This section describes the TCP and UDP commands and data formats used by a client computer to communicate with Gocator sensors using the Gocator protocol. It also describes the connection types (Discovery, Control, Upgrade, Data, and Health), and data types. The protocol enables the client to:

- Send commands to run sensors, provide software triggers, read/write files, etc.
- Receive data, health, and diagnostic messages.
- Upgrade firmware.

> The Gocator 4.x/5.x firmware uses mm, mm^2, mm^3, and degrees as standard units. In all protocols, values are scaled by 1000, as values in the protocols are represented as integers. This results in effective units of mm/1000, mm^2/1000, mm^3/1000, and deg/1000 in the protocols.

To use the protocol, it must be enabled and configured in the active job.

> Sensors send UDP broadcasts over the network over the Internal Discovery channel (port 2016) at regular intervals during operation to perform peer discovery.

> The Gocator SDK provides open source C language libraries that implement the network commands and data formats defined in this section. For more information, see *GoSDK* on page 64.

For information on configuring the protocol using the web interface, see *Ethernet Output* on page 171.

For information on job file structures (for example, if you wish to create job files programmatically), see *Job File Structure* on page 192.

Data Types

The table below defines the data types and associated type identifiers used in this section.

All values except for IP addresses are transmitted in little endian format (least significant byte first) unless stated otherwise. The bytes in an IP address "a.b.c.d" will always be transmitted in the order a, b, c, d (big endian).

Data Types

Type	Description	Null Value
char	Character (8-bit, ASCII encoding)	-
byte	Byte.	-
8s	8-bit signed integer.	-128
8u	8-bit unsigned integer.	255U
16s	16-bit signed integer.	-32768 (0x8000)
16u	16-bit unsigned integer.	65535 (0xFFFF)
32s	32-bit signed integer.	-2147483648 (0x80000000)
32u	32-bit unsigned integer.	4294967295 (0xFFFFFFFF)
64s	64-bit signed integer.	-9223372036854775808 (0x8000000000000000)
64u	64-bit unsigned integer.	18446744073709551615 (0xFFFFFFFFFFFFFFFF)
64f	64-bit floating point	-1.7976931348623157e+308
Point16s	Two 16-bit signed integers	-
Point64f	Two 64-bit floating point values	-
Point3d64f	Three 64-bit floating point values	-
Rect64f	Four 64-bit floating point values	-
Rect3d64f	Eight 64-bit floating point values	-

Commands

The following sections describe the commands available on the Discovery (page 258), Control (page 261), and Upgrade (page 294) channels.

When a client sends a command over the Control or Upgrade channel, the sensor sends a reply whose identifier is the same as the command's identifier. The identifiers are listed in the tables of each of the commands.

Status Codes

Each reply on the Discovery, Control, and Upgrade channels contains a *status* field containing a status code indicating the result of the command. The following status codes are defined:

Status Codes

Label	Value	Description
OK	1	Command succeeded.
Failed	0	Command failed.
Invalid State	-1000	Command is not valid in the current state.
Item Not Found	-999	A required item (e.g., file) was not found.
Invalid Command	-998	Command is not recognized.
Invalid Parameter	-997	One or more command parameters are incorrect.
Not Supported	-996	The operation is not supported.
Simulation Buffer Empty	-992	The simulation buffer is empty.

Discovery Commands

Sensors ship with the following default network configuration:

Setting	Default
DHCP	0 (disabled)
IP Address	192.168.1.10
Subnet Mask	255.255.255.0
Gateway	0.0.0.0 (disabled)

Use the Get Address and Set Address commands to modify a sensor's network configuration. These commands are UDP broadcast messages:

Destination Address	Destination Port
255.255.255.255	3220

When a sensor accepts a discovery command, it will send a UDP broadcast response:

Destination Address	Destination Port
255.255.255.255	Port of command sender.

The use of UDP broadcasts for discovery enables a client computer to locate a sensor when the sensor and client are configured for different subnets. All you need to know is the serial number of the sensor in order to locate it on an IP network.

Get Address

The Get Address command is used to discover sensors across subnets.

Command

Field	Type	Offset	Description
length	64s	0	Command length.
type	64s	8	Command type (0x1).

Field	Type	Offset	Description
signature	64s	16	Message signature (0x0000504455494D4C)
deviceId	64s	24	Serial number of the device whose address information is queried. 0 selects all devices.

Reply

Field	Type	Offset	Description
length	64s	0	Reply length.
type	64s	8	Reply type (0x1001).
status	64s	16	Operation status.
signature	64s	24	Message signature (0x0000504455494D4C)
deviceId	64s	32	Serial number.
dhcpEnabled	64s	40	0 – Disabled 1 – Enabled
reserved[4]	byte	48	Reserved.
address[4]	byte	52	The IP address in left to right order.
reserved[4]	byte	56	Reserved.
subnetMask[4]	byte	60	The subnet mask in left to right order.
reserved[4]	byte	64	Reserved.
gateway[4]	byte	68	The gateway address in left to right order.
reserved[4]	byte	72	Reserved.
reserved[4]	byte	76	Reserved.

Set Address

The Set Address command modifies the network configuration of a sensor. On receiving the command, the sensor will perform a reset. You should wait 30 seconds before re-connecting to the sensor.

Command

Field	Type	Offset	Description
length	64s	0	Command length.
type	64s	8	Command type (0x2).
signature	64s	16	Message signature (0x0000504455494D4C)
deviceId	64s	24	Serial number of the device whose address information is queried. 0 selects all devices.
dhcpEnabled	64s	32	0 – Disabled 1 – Enabled
reserved[4]	byte	40	Reserved.
address[4]	byte	44	The IP address in left to right order.
reserved[4]	byte	48	Reserved.
subnetMask[4]	byte	52	The subnet mask in left to right order.
reserved[4]	byte	56	Reserved.

Field	Type	Offset	Description
gateway[4]	byte	60	The gateway address in left to right order.
reserved[4]	byte	64	Reserved.
reserved[4]	byte	68	Reserved.

Reply

Field	Type	Offset	Description
length	64s	0	Reply length.
type	64s	8	Reply type (0x1002).
status	64s	16	Operation status. For a list of status codes, see *Commands* on page 257.
signature	64s	24	Message signature (0x0000504455494D4C).
deviceId	64s	32	Serial number.

Get Info

The Get Info command is used to retrieve sensor information.

Command

Field	Type	Offset	Description
length	64s	0	Command length.
type	64s	8	Command type (0x5).
signature	64s	16	Message signature (0x0000504455494D4C).
deviceId	64s	24	Serial number of the device whose address information is queried. 0 selects all devices.

Reply

Field	Type	Offset	Description
length	64s	0	Reply length.
type	64s	8	Reply type (0x1005).
status	64s	16	Operation status. For a list of status codes, see *Commands* on page 257.
signature	64s	24	Message signature (0x0000504455494D4C).
attrCount	16u	32	Byte count of the attributes (begins after this field and ends before propertyCount).
id	32u	34	Serial number.
version	32u	38	Version as a 4-byte integer (encoded in little-endian).
uptime	64u	42	Sensor uptime (microseconds).
ipNegotiation	byte	50	IP negotiation type: 0 – Static 1 – DHCP

Field	Type	Offset	Description
addressVersion	byte	51	IP address version (always 4).
address[4]	byte	52	IP address.
reserved[12]	byte	56	Reserved.
prefixLength	32u	68	Subnet prefix length (in number of bits).
gatewayVersion	byte	72	Gateway address version (always 4).
gatewayAddress[4]	byte	73	Gateway address.
reserved[12]	byte	77	Reserved.
controlPort	16u	89	Control channel port.
upgradePort	16u	91	Upgrade channel port.
healthPort	16u	93	Health channel port.
dataPort	16u	95	Data channel port.
webPort	16u	97	Web server port.
propertyCount	8u	99	Number of sensor ID properties.
properties[propertyCount]	Property	100	List of sensor ID properties.

Property

Field	Type	Description
nameLength	8u	Length of the name.
name[nameLength]	char	Name string.
valueLength	8u	Length of the value.
value[valueLength]	char	Value string.

Control Commands

A client sends control commands for most operations over the Control TCP channel (port 3190).

The Control channel and the Upgrade channel (port 3192) can be connected simultaneously. For more information on Upgrade commands, see *Upgrade Commands* on page 294.

States

A sensor system can be in one of states: . The client sends the Start and Stop control commands to change the system's current state to Running and Ready, respectively. The sensor can also be configured to boot in either the Ready or Running state, by enabling or disabling autostart, respectively, using the Set Auto Start Enabled command.

In the Ready state, a sensor can be configured. In the Running state, a sensor responds to input signals, performs measurements, drives its outputs, and sends data messages to the client.

The state of the sensor can be retrieved using the Get States or Get System Info command.

Progressive Reply

Some commands send replies progressively, as multiple messages. This allows the sensor to stream data without buffering it first, and allows the client to obtain progress information on the stream.

A progressive reply begins with an initial, standard reply message. If the *status* field of the reply indicates success, the reply is followed by a series of "continue" reply messages.

A continue reply message contains a block of data of variable size, as well as status and progress information. The series of continue messages is ended by either an error, or a continue message containing 0 bytes of data.

Protocol Version

The Protocol Version command returns the protocol version of the connected sensor.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4511)

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4511).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
majorVersion	8u	10	Major version.
minorVersion	8u	11	Minor version.

Get Address

The Get Address command is used to get a sensor address.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x3012)

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x3012).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
dhcpEnabled	byte	10	0 – DHCP not used 1 – DHCP used
address[4]	byte	11	IP address (most significant byte first).
subnetMask[4]	byte	15	Subnet mask.
gateway[4]	byte	19	Gateway address.

Set Address

The Set Address command modifies the network configuration of a sensor. On receiving the command, the sensor will perform a reset. You should wait 30 seconds before re-connecting to the sensor.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x3013)
dhcpEnabled	byte	6	0 – DHCP not used 1 – DHCP used
address[4]	byte	7	IP address (most significant byte first).
subnetMask[4]	byte	11	Subnet mask.
gateway[4]	byte	15	Gateway address.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x3013).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Get System Info V2

The Get System Info command reports information about the local node, remote nodes and assigned buddies.

Firmware version refers to the version of the sensor's firmware installed on each individual sensor. The client can upgrade the sensor's firmware by sending the Start Upgrade command (see *Start Upgrade* on page 294). Firmware upgrade files are available from the downloads section under the support tab on the LMI web site. For more information on getting the latest firmware, see *Firmware Upgrade* on page 125.

Every sensor contains factory backup firmware. If a firmware upgrade command fails (e.g., power is interrupted), the factory backup firmware will be loaded when the sensor is reset or power cycled. In this case, the sensors will fall back to the factory default IP address. To avoid IP address conflicts in a multi-sensor system, connect to one sensor at a time and re-attempt the firmware upgrade.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4010)

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4010).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
localInfoSize	16u	10	Size of localInfo structure. Current: 116.
localInfo	Local Info	12	Info for this device.
remoteCount	32u	-	Number of discovered sensors.
remoteInfoSize	16u	-	Size of remoteInfo structure. Current 124.
remoteInfo[remoteCount]	Remote Info	-	List of info for discovered sensors.
buddyInfoCount	32u	-	Number of buddies assigned (can be 0).
buddyInfoSize	16u	-	Size of buddyInfo structure. Current: 8.
Buddies[buddyCount]	Buddy Info	-	List of info for the assigned buddies.

Local Info

Field	Type	Offset	Description
deviceId	32u	0	Serial number of the device.
address[4]	byte	4	IP address (most significant byte first).
modelName[32]	char	8	Model name; "part number" starting with GoSdk 5.3.17.23. Should not be parsed.
firmwareVersion[4]	byte	40	Firmware version (most significant byte first).
state	32s	44	Sensor state 0 – Ready 1 – Running For more information on states, see *Control Commands* on page 261.
role	32s	48	Sensor role 0 – Main
modelNumber[32]	char	52	Model number that can be parsed.
modelDisplayName[32]	char	84	User-friendly model display name that can be used to rename sensors more appropriately for custom-branding naming.

Remote Info

Field	Type	Offset	Description
deviceId	32u	0	Serial number of the remote device.
address[4]	byte	4	IP address (most significant byte first).
modelName[32]	char	8	Remote model name; "remote part number" starting with GoSdk 5.3.17.23.

Field	Type	Offset	Description
firmwareVersion[4]	byte	40	Remote firmware version (most significant byte first).
state	32s	44	Remote sensor state 0 – Ready 1 – Running For more information on states, see *Control Commands* on page 261.
role	32s	48	Sensor role 0 – Main
mainId	32u	52	Serial number of the main device, or zero.
buddyableStatus	32s	56	Whether or not the device can be buddied: 1 – Can be buddied Errors: 0 – Unbuddiable (General Error) -100 – Already buddied -99 – Invalid State (e.g. running) -98 – Version Mismatch -97 – Model Mismatch
modelNumber[32]	char	60	Model number that can be parsed.
modelDisplayName[32]	char	92	Remote user-friendly model display name that can be used to rename sensors more appropriately for custom-branding naming.

Buddy Info

Field	Type	Offset	Description
deviceId	32u	2	Serial number of the device.
state	k32s	6	Buddy state 2 - Connecting 1 – Connected Errors: 0 – Unbuddiable (General Error) -100 – Already buddied -99 – Invalid State (e.g. running) -98 – Version Mismatch -97 – Model Mismatch -95 – Device Missing -92 – Standalone Sensor -91 – Restricted Sensor Mismatch

Get System Info

> This version of the Get System Info command is deprecated. Use Get System Info (v2) instead.

The Get System Info command reports information for sensors that are visible in the system.

Firmware version refers to the version of the sensor's firmware installed on each individual sensor. The client can upgrade the sensor's firmware by sending the Start Upgrade command (see *Start Upgrade* on page 294). Firmware upgrade files are available from the downloads section under the support tab on the LMI web site. For more information on getting the latest firmware, see *Firmware Upgrade* on page 125.

Every sensor contains factory backup firmware. If a firmware upgrade command fails (e.g., power is interrupted), the factory backup firmware will be loaded when the sensor is reset or power cycled. In this case, the sensors will fall back to the factory default IP address. To avoid IP address conflicts in a multi-sensor system, connect to one sensor at a time and re-attempt the firmware upgrade.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4002)

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4002).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
localInfo	Sensor Info	10	Info for this device.
remoteCount	32u	66	Number of discovered sensors.
remoteInfo[remoteCount]	Sensor Info	70	List of info for discovered sensors.

Sensor Info

Field	Type	Offset	Description
deviceId	32u	0	Serial number of the device.
address[4]	byte	4	IP address (most significant byte first).
modelName[32]	char	8	Model name.
firmwareVersion[4]	byte	40	Firmware version (most significant byte first).
state	32s	44	Sensor state 0 – Ready 1 – Running For more information on states, see *Control Commands* on page 261.
role	32s	48	Sensor role

Field	Type	Offset	Description
			0 – Main

Get States

The Get States command returns various system states.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4525)

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4525).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
count	32u	10	Number of state variables.
sensorState	32s	14	Sensor state 0 – Ready 1 – Running For more information on states, see *Control Commands* on page 261.
loginState	32s	18	Device login state 0 – No user 1 – Administrator 2 – Technician
alignmentReference	32s	22	Alignment reference 0 – Fixed 1 – Dynamic
alignmentState	32s	26	Alignment state 0 – Unaligned 1 – Aligned
recordingEnabled	32s	30	Whether or not recording is enabled 0 – Disabled 1 – Enabled
playbackSource	32s	34	Playback source 0 – Live data 1 – Recorded data

Field	Type	Offset	Description
uptimeSec	32su	38	Uptime (whole seconds component)
uptimeMicrosec	32u	42	Uptime (remaining microseconds component)
playbackPos	32u	46	Playback position
playbackCount	32u	50	Playback frame count
autoStartEnabled	32u	54	Auto-start enable (boolean)
isAccelerator	32u	58	Is the device an accelerator instance?
voltage	32u	62	Voltage setting 0 – 48V 1 – 24V
cableLength	32u	66	Cable length (maximum Is 60.0 meters, default is 5.0 meters)
quickEditEnabled	32u	70	Quick Edit state
securityLevel	32s	74	Security Level 0 – No security, any user type can access system. 1 – Basic security level, only authorized user types can access system.
brandingType	32s	78	Branding Type 0 – None/Gocator (default) 1 – White Label 2 – Custom

Log In/Out

The Log In/Out command is used to log in or out of a sensor.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4003).
userType	32s	6	Defines the user type 0 – None (log out) 1 – Administrator 2 – Technician
password[64]	char	10	Password (required for log-in only).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4003).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Change Password

The Change Password command is used to change log-in credentials for a user.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4004).
user type	32s	6	Defines the user type 0 – None (log out) 1 – Administrator 2 – Technician
password[64]	char	10	New password.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4004).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Passwords can only be changed if a user is logged in as an administrator.

Set Buddy

The Set Buddy command is used to assign or unassign a Buddy sensor.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4005).
buddyId	32u	6	Id of the sensor to acquire as buddy. Set to 0 to remove buddy.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4005).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

List Files

The List Files command returns a list of the files in the sensor's file system.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x101A).
extension[64]	char	6	Specifies the extension used to filter the list of files (does not include the "."). If an empty string is used, then no filtering is performed.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x101A).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
count	32u	10	Number of file names.
fileNames[count][64]	char	14	File names.

Copy File

The Copy File command copies a file from a source to a destination within the connected sensor (a .job file, a component of a job file, or another type of file; for more information, see *Job File Structure* on page 192).

To make a job active (to load it), copy a saved job to "_live.job".

To "save" the active job, copy from "_live.job" to another file.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x101B).
source[64]	char	6	Source file name.
destination[64]	char	70	Destination file name.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x101B).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Read File

Downloads a file from the connected sensor (a .job file, a component of a job file, or another type of file; for more information, see *Job File Structure* on page 192).

To download the live configuration, pass "_live.job" in the *name* field.

To read the configuration of the live configuration only, pass "_live.job/config.xml" in the *name* field.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x1007).
name[64]	char	6	Source file name.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x1007).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
length	32u	10	File length.
data[length]	byte	14	File contents.

Write File

The Write File command uploads a file to the connected sensor (a .job file, a component of a job file, or another type of file; for more information, see *Job File Structure* on page 192).

To make a job file live, write to "_live.job". Except for writing to the live file, the file is permanently stored on the sensor.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x1006).
name[64]	char	6	Source file name.
length	32u	70	File length.
data[length]	byte	74	File contents.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x1006).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Delete File

The Delete File command removes a file from the connected sensor (a .job file, a component of a job file, or another type of file; for more information, see *Job File Structure* on page 192).

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x1008).
name[64]	char	6	Source file name.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x1008).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

User Storage Used

The User Storage Used command returns the amount of user storage that is used.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x1021).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x1021).
status	32s	6	Reply status.
spaceUsed	64u	10	The used storage space in bytes.

User Storage Free

The User Storage Free command returns the amount of user storage that is free.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x1022).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x1022).
status	32s	6	Reply status.
spaceFree	64u	10	The free storage space in bytes.

Get Default Job

The Get Default Job command gets the name of the job the sensor loads when it powers up.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4100).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4100).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
name[64]	char	10	The file name (null-terminated) of the job the sensor loads when it powers up.

Set Default Job

The Set Default Job command sets the job the sensor loads when it powers up.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4101).
fileName[64]	char	6	File name (null-terminated) of the job the sensor loads when it powers up.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4101).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Get Loaded Job

The Get Loaded Job command returns the name and modified status of the currently loaded file.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4512).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4512).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
fileName[64]	char	10	Name of the currently loaded job.
changed	8u	74	Whether or not the currently loaded job has been changed (1: yes; 0: no).

Get Alignment Reference

The Get Alignment Reference command is used to get the sensor's alignment reference.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4104).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4104).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
reference	32s	10	Alignment reference 0 – Fixed 1 – Dynamic

Set Alignment Reference

The Set Alignment Reference command is used to set the sensor's alignment reference.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4103).
reference	32s	6	Alignment reference 0 – Fixed

Field	Type	Offset	Description
			1 – Dynamic

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4103).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Clear Alignment

The Clear Alignment command clears sensor alignment.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4102).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4102).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Get Timestamp

The Get Timestamp command retrieves the sensor's timestamp, in clock ticks. All devices in a system are synchronized with the system clock; this value can be used for diagnostic purposes, or used to synchronize the start time of the system.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x100A).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x100A).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
timestamp	64u	10	Timestamp, in clock ticks.

Get Encoder

This command retrieves the current system encoder value.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x101C).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x101C).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
encoder	64s	10	Current encoder position, in ticks.

Reset Encoder

The Reset Encoder command is used to reset the current encoder value.

The encoder value can be reset only when the encoder is connected directly to a sensor. When the encoder is connected to the master, the value cannot be reset via this command.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x101E).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x101E).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Start

The Start command starts the sensor system (system enters the Running state). For more information on states, see *Control Commands* on page 261.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x100D).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x100D).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Scheduled Start

The scheduled start command starts the sensor system (system enters the Running state) at target time or encoder value (depending on the trigger mode). For more information on states, see *Control Commands* on page 261.

Command

Field	Type	Offset	Description
length	32u	0	Command size – in bytes.
id	16u	4	Command identifier (0x100F).
target	64s	6	Target scheduled start value (in ticks or µs, depending on the trigger type).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size – in bytes.
id	16u	4	Reply identifier (0x100F).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Stop

The Stop command stops the sensor system (system enters the Ready state). For more information on states, see *Control Commands* on page 261.

Command

Field	Type	Type	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x1001).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x1001).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Get Auto Start Enabled

The Get Auto Start Enabled command returns whether the system automatically starts after booting.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x452C).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x452C).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
enable	8u	10	0: disabled 1: enabled

Set Auto Start Enabled

The Set Auto Start Enabled command sets whether the system automatically starts after booting (enters Running state; for more information on states, see *Control Commands* on page 261).

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x452B).
enable	8u	6	0: disabled 1: enabled

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x452B).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Start Alignment

The Start Alignment command is used to start the alignment procedure on a sensor.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4600).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4600).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
opId	32u	10	Operation ID. Use this ID to correlate the command/reply on the Command channel with the correct Alignment Result message on the Data channel. A unique ID is returned each time the client uses this command.

Start Exposure Auto-set

The Start Exposure Auto-set command is used to start the exposure auto-set procedure on a sensor.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4601).
role	32s	6	Role of sensors to auto-set. 0 – Main 1 – Buddy

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4601).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
opId	32u	10	Operation ID. Use this ID to correlate the command/reply on the Command channel with the correct Exposure Calibration Result message on the Data channel. A unique ID is returned each time the client uses this command.

Software Trigger

The Software Trigger command causes the sensor to take a snapshot while in software mode and in the Running state.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4510).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4510).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Ping

The Ping command can be used to test the control connection. This command has no effect on sensors.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x100E).
timeout	64u	6	Timeout value (microseconds).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x100E).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

If a non-zero value is specified for timeout, the client must send another ping command before the timeout elapses; otherwise the server would close the connection. The timer is reset and updated with every command.

Reset

The Reset command reboots the Main sensor and any Buddy sensors. All sensors will automatically reset 3 seconds after the reply to this command is transmitted.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4300).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4300).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Backup

The Backup command creates a backup of all files stored on the connected sensor and downloads the backup to the client.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x1013).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x1013).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
length	32u	10	Data length.
data[length]	byte	14	Data content.

Restore

The Restore command uploads a backup file to the connected sensor and then restores all sensor files from the backup.

The sensor must be reset or power-cycled before the restore operation can be completed.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x1014).
length	32u	6	Data length.
data[length]	byte	10	Data content.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x1014).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Restore Factory

The Restore Factory command restores the connected sensor to factory default settings.

The command erases the non-volatile memory of the main device.

This command has no effect on connected Buddy sensors.

Note that the sensor must be reset or power-cycled before the factory restore operation can be completed.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4301).
resetIp	8u	6	Specifies whether IP address should be restored to default: 0 – Do not reset IP 1 – Reset IP

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4301).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Get Recording Enabled

The Get Recording Enabled command retrieves whether recording is enabled.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4517).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4517).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
enable	8u	10	0: disabled; 1: enabled.

Set Recording Enabled

The Set Recording Enabled command enables recording for replay later.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4516).
enable	8u	6	0: disabled; 1: enabled.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4516).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Clear Replay Data

The Clear Replay Data command clears the sensors replay data..

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4513).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4513).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Get Playback Source

The Get Playback Source command gets the data source for data playback.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4524).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4524).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
source	32s	10	Source 0 – Live 1 – Replay buffer

Set Playback Source

The Set Playback Source command sets the data source for data playback.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4523).
source	32s	6	Source 0 – Live 1 – Replay buffer

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4523).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Simulate

The Simulate command simulates the last frame if playback source is live, or the current frame if playback source is the replay buffer.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4522).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4522).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
bufferValid	8u	10	Whether or not the buffer is valid.

A reply status of -996 means that the current configuration (mode, sensor type, etc.) does not support simulation.

A reply status of -992 means that the simulation buffer is empty. Note that the buffer can be valid even if the simulation buffer is actually empty due to optimization choices. This scenario means that the simulation buffer would be valid if data were recorded.

Seek Playback

The Seek Playback command seeks to any position in the current playback dataset. The frame is then sent.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4503).
frame	32u	6	Frame index.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4503).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Step Playback

The Step Playback command advances playback by one frame.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4501).
direction	32s	6	Define step direction 0 – Forward 1 – Reverse

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4501).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

When the system is running in the Replay mode, this command advances replay data (playback) by one frame. This command returns an error if no live playback data set is loaded. You can use the Copy File command to load a replay data set to _live.rec.

Playback Position

The Playback Position command retrieves the current playback position.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4502).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4502).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
Frame Index	32u	10	Current frame index (starts from 0).
Frame Count	32u	14	Total number of available frames/objects.

Clear Measurement Stats

The Clear Measurement Stats command clears the sensor's measurement statistics.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4526).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4526).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Read Live Log

The Read Live Log command returns an XML file containing the log messages between the passed start and end indexes.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x101F).
Start	32u	6	First log to read
End	32u	10	Last log to read

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x101F).
status	32s	6	Reply status.
length	32u	10	File length
data[length]	byte	14	XML Log File

Clear Log

The Clear Log command clears the sensor's log.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x101D).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x101D).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Simulate Unaligned

The Simulate Unaligned command simulates data before alignment transformation.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x452A).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x452A).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Acquire

The Acquire command acquires a new scan.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4528).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4528).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

The command returns after the scan has been captured and transmitted.

Acquire Unaligned

The Acquire Unaligned command acquires a new scan without performing alignment transformation.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4527).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4527).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

The command returns after the scan has been captured and transmitted.

Create Model

The Create Model command creates a new part model from the active simulation scan.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4602).
modelName[64]	char	6	Name of the new model (without .mdl extension)

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4602).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Detect Edges

The Detect Edges command detects and updates the edge points of a part model.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.

Field	Type	Offset	Description
id	16u	4	Command identifier (0x4604).
modelName[64]	char	6	Name of the model (without .mdl extension)
sensitivity	16u	70	Sensitivity (in thousandths).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4604).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Add Tool

The Add Tool command adds a tool to the live job.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4530).
typeName[64]	char	6	Type name of the tool (e.g., ProfilePosition)
name[64]	char	70	User-specified name for tool instance

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4530).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

Add Measurement

The Add Measurement command adds a measurement to a tool instance.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4531).
toolIndex	32u	6	Index of the tool instance the new measurement is added to.
typeName[64]	char	10	Type name of the measurement (for example, X).
name[64]	char	74	User-specified name of the measurement instance.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4531).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.

> This command can only be used with dynamic tools (tools with a dynamic list of measurements). The maximum number of instances for a given measurement type can be found in the ToolOptions node. For dynamic tools, the maximum count is greater than one, while for static tools it is one.

Read File (Progressive)

The progressive Read File command reads the content of a file as a stream.

This command returns an initial reply, followed by a series of "continue" replies if the initial reply's *status* field indicates success. The continue replies contain the actual data, and have 0x5000 as their identifier.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4529).
name[64]	char	6	Source file name.

Initial Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4529).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
progressTotal	32u	10	Progress indicating completion (100%).
progress	32u	14	Current progress.

Continue Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x5000).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
progressTotal	32u	10	Progress indicating completion (100%).
progress	32u	14	Current progress.

Field	Type	Offset	Description
size	32u	18	Size of the chunk in bytes.
data[size]	byte	22	Chunk data.

Export CSV (Progressive)

The progressive Export CSV command exports replay data as a CSV stream.

This command returns an initial reply, followed by a series of "continue" replies if the initial reply's *status* field indicates success. The continue replies contain the actual data, and have 0x5000 as their identifier.

Command

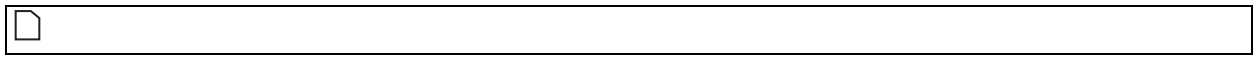
Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4507).

Initial Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4507).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
progressTotal	32u	10	Progress indicating completion (100%).
progress	32u	14	Current progress.

Continue Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x5000).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
progressTotal	32u	10	Progress indicating completion (100%).
progress	32u	14	Current progress.
size	32u	18	Size of the chunk in bytes.
data[size]	byte	22	Chunk data.



Export Bitmap (Progressive)

The progressive Export Bitmap command exports replay data as a bitmap stream.

This command returns an initial reply, followed by a series of "continue" replies if the initial reply's *status* field indicates success. The continue replies contain the actual data, and have 0x5000 as their identifier.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4508).
type	32s	6	Data type: 0 – Range or video 1 – Intensity
source	32s	10	Data source to export.

Initial Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4508).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
progressTotal	32u	10	Progress indicating completion (100%).
progress	32u	14	Current progress.

Continue Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x5000).
status	32s	6	Reply status. For a list of status codes, see *Commands* on page 257.
progressTotal	32u	10	Progress indicating completion (100%).
progress	32u	14	Current progress.
size	32u	18	Size of the chunk in bytes.
data[size]	byte	22	Chunk data.

Get Runtime Variable Count

The Get Runtime Variable Count command gets the number of runtime variables that can be accessed.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4537).

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4537).
status	32s	6	Reply status.
valueLength	32u	10	The count of runtime variables.

Set Runtime Variables

The Set Runtime Variables command sets the runtime variables at the given index for the given length.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4536).
index	32u	6	The starting index of the variables to set.
length	32u	10	The number of values to set from the starting index.
values[length]	32s	14	The runtime variable values to set.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4536).
status	32s	6	Reply status.

Get Runtime Variables

The Get Runtime Variables command gets the runtime variables for the given index and length.

Command

Field	Type	Offset	Description
length	32u	0	Command size including this field, in bytes.
id	16u	4	Command identifier (0x4535).
index	32u	6	The starting index of the variables to retrieve.
length	32u	10	The number of values to retrieve from the starting index.

Reply

Field	Type	Offset	Description
length	32u	0	Reply size including this field, in bytes.
id	16u	4	Reply identifier (0x4535).
status	32s	6	Reply status.
index	32u	10	The starting index of the variables being returned.

Field	Type	Offset	Description
length	32u	14	The number of values being returned.
values[length]	32s	18	The runtime variable values.

Upgrade Commands

A client sends firmware upgrade commands over the Upgrade TCP channel (port 3192).

The Control channel (port 3190) and the Upgrade channel can be connected simultaneously. For more information on Control commands, see *Control Commands* on page 261.

After connecting to a sensor, you can use the Protocol Version command to retrieve the protocol version. Protocol version refers to the version of the Gocator Protocol supported by the *connected sensor* (the sensor to which a command connection is established), and consists of major and minor parts. The minor part is updated when backward-compatible additions are made to the protocol. The major part is updated when breaking changes are made to the protocol.

Start Upgrade

The Start Upgrade command begins a firmware upgrade for the sensors in a system. All sensors automatically reset 3 seconds after the upgrade process is complete.

Command

Field	Type	Offset	Description
length	64s	0	Command size including this field, in bytes.
id	64s	8	Command identifier (0x0000).
length	64s	16	Length of the upgrade package (bytes).
data[length]	byte	24	Upgrade package data.

Reply

Field	Type	Offset	Description
length	64s	0	Reply size including this field, in bytes.
id	64s	8	Reply identifier (0x0000).
status	64s	16	Reply status. For a list of status codes, see *Commands* on page 257.

Start Upgrade Extended

The Start Upgrade Extended command begins a firmware upgrade for the sensors in a system. All sensors automatically reset 3 seconds after the upgrade process is complete.

Command

Field	Type	Offset	Description
length	64s	0	Command size including this field, in bytes.
id	64s	8	Command identifier (0x0003).
skipValidation	64s	16	Whether or not to skip validation (0 – do not skip, 1 – skip).

Field	Type	Offset	Description
length	64s	24	Length of the upgrade package (bytes).
data[length]	byte	32	Upgrade package data.

Reply

Field	Type	Offset	Description
length	64s	0	Reply size including this field, in bytes.
id	64s	8	Reply identifier (0x0003).
status	64s	16	Reply status. For a list of status codes, see *Commands* on page 257.

Get Upgrade Status

The Get Upgrade Status command determines the progress of a firmware upgrade.

Command

Field	Type	Offset	Description
length	64s	0	Command size including this field, in bytes.
id	64s	8	Command identifier (0x1)

Reply

Field	Type	Offset	Description
length	64s	0	Reply size including this field, in bytes.
id	64s	8	Reply identifier (0x1).
status	64s	16	Reply status. For a list of status codes, see *Commands* on page 257.
state	64s	24	Upgrade state: -1 – Failed 0 – Completed 1 – Running 2 – Completed, but should run again
progress	64s	32	Upgrade progress (valid when in the Running state)

Get Upgrade Log

The Get Upgrade Log command can retrieve an upgrade log in the event of upgrade problems.

Command

Field	Type	Offset	Description
length	64s	0	Command size including this field, in bytes.
id	64s	8	Command identifier (0x2)

Reply

Field	Type	Offset	Description
length	64s	0	Reply size including this field, in bytes.
id	64s	8	Reply identifier (0x2).
status	64s	16	Reply status. For a list of status codes, see *Commands* on page 257.
length	64s	24	Length of the log (bytes).
log[length]	char	32	Log content.

Results

The following sections describe the results (data and health) that a sensor sends.

Health Results

A client can receive health messages from a sensor by connecting to the Health TCP channel (port 3194).

The Data channel (port 3196) and the Health channel can be connected at the same time. The sensor accepts multiple connections on each port. For more information on the Data channel, see *Data Results* on page 301.

Messages that are received on the Data and Health channels use a common structure, called Gocator Data Protocol (GDP). Each message consists of a 6-byte header, containing *size* and *control* fields, followed by a variable-length, message-specific content section. The structure of the GDP message is defined below.

Gocator Data Protocol

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last Message flag Bits 0-14: Message type identifier. (See individual data result sections.)

Messages are always sent in groups. The Last Message flag in the *control* field is used to indicate the final message in a group. If there is only one message per group, this bit will be set in each message.

A Health Result contains a single data block for health *indicators*. Each indicator reports the current status of some aspect of the sensor system, such as CPU usage or network throughput.

Health Result

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last message flag. Bits 0-14: Message type identifier. Always 0.
count (C)	32u	6	Count of indicators in this message.

Field	Type	Offset	Description
source	8u	10	Source (0 – Main, 1 – Buddy).
reserved[3]	8u	11	Reserved
indicators[C]	Indicator	14	Array of indicators (see format below).

The indicators block contains a 2-dimensional array of indicator data. Each row in the array has the following format:

Indicator Format

Field	Type	Offset	Description
id	32u	0	Unique indicator identifier (see *Indicator identifiers* below table below).
instance	32u	4	Indicator instance.
value	64s	8	Value (identifier-specific meaning).

The following health indicators are defined for sensor systems.

When a sensor is accelerated, some health indicators report values from the *PC* that is accelerating the sensor, or a combination of both. In the table below, values are reported from the sensor unless otherwise indicated.

Undocumented indicators may be included in addition to the indicators defined below.

Indicator identifiers

Indicator	ID	Instance	Value
Encoder Value	1003	-	Current system encoder tick.
Encoder Frequency	1005	-	Current system encoder frequency (ticks/s).
Laser Safety	1010	-	Laser safety status. 0: laser is disabled 1: laser is enabled
App Version	2000	-	Firmware application version.
Internal Temperature	2002	-	Internal temperature (centidegrees Celsius).
Uptime	2017	-	Time elapsed since node boot-up or reset (seconds).
Projector Temperature	2404	-	Projector module temperature (centidegrees Celsius). Only available on projector based devices.
Control Temperature	2028	-	Control module temperature (centidegrees Celsius). Available only on 3B-class devices.
Memory Usage	2003	-	Amount of memory currently used (bytes).

Indicator	ID	Instance	Value
Memory Capacity	2004	-	Total amount of memory available (bytes).
Storage Usage	2005	-	Amount of non-volatile storage used (bytes).
Storage Capacity	2006	-	Total amount of non-volatile storage available (bytes).
Alignment State	20008	-	Alignment state: 0 – not aligned 1 - aligned
CPU Usage	2007	-	CPU usage (percentage of maximum).
Net Out Capacity	2009	-	Total available outbound network throughput (bytes/s).
Net Out Link Status	2034	-	Current Ethernet link status.
Sync Source*	2043	-	Synchronization source. 1 - Master device 2 - Sensor
Digital Inputs*	2024	-	Current digital input status (one bit per input).
Event Count	2102	-	Total number of events triggered.
Camera Trigger Drops	2201	-	Number of dropped triggers.
Analog Output Drops	21014 (previously 2501)	Output Index	Not used.
Digital Output Drops	21015 (previously 2601)	Output Index	Not used.
Serial Output Drops	21016 (previously 2701)	Output Index	Not used.
Sensor State*	20000	-	Sensor state. -1 – Conflict 0 – Ready 1 – Running
Current Sensor Speed*	20001	-	Current sensor speed. (Hz)
Maximum Speed*	20002	-	The sensor’s maximum speed.
Spot Count*	20003	-	Number of spots found in the last unresampled profile/surface.
Max Spot Count*	20004	-	Maximum number of spots that can be found.
Scan Count*	20005	-	Number of surfaces detected from a top device.
Master Status*	20006	0 for main	Master connection status:

Indicator	ID	Instance	Value
		1 for buddy	0 – Not connected 1 – Connected The indicator with instance = buddy does not exist if the buddy is not connected.
Cast Start State*	20007		The state of the second digital input. (NOTE: Only available on XLine capable licensed devices)
Point Count	20015	-	Number of points found in last resampled Profile/Surface.
Max Point Count	20016	-	Maximum number of points that can be found.
Laser Overheat*	20020	-	Indicates whether laser overheat has occurred. 0 – Has not overheated 1 – Has overheated Only available on certain 3B laser devices.
Laser Overheat Duration*	20021	-	The length of time in which the laser overheating state occurred. Only available on certain 3B laser devices.
Playback Position*	20023	-	The current replay playback position.
Playback Count*	20024	-	The number of frames present in the replay.
FireSync Version	20600	-	The FireSync version used by the Gocator build. The low-level firmware version used by the sensor.
Processing Drops**	21000	-	Number of dropped frames. The sum of various processing drop related indicators.
Last Processing Latency	21001	-	Last delay from camera exposure to availability of all results.
Max Processing Latency	21002	-	Maximum value of processing latency.
Ethernet Output	21003	-	Number of bytes transmitted.
Ethernet Rate	21004	-	The average number of bytes per second being transmitted.
Ethernet Drops	21005	-	Number of dropped Ethernet packets.
Trigger Drops**	21010		Number of dropped triggers. The sum of various triggering-related drop indicators.
Output Drops**	21011		Not all outputs are supported by Gocator 200 scanners.
Controlled Trigger Drops	21017		Trigger drops from the Controlled Triggering System (Grouped with “Trigger Drops” indicator)
Surface Processing Time	21018		Processing Time of Frame on 35XX/32XX (microseconds)

Indicator	ID	Instance	Value
Max Frame Rate	21019		Max Configurable frame rate given above Processing Time (scaled by 1x10-6)
Range Valid Count**	21100	-	Number of valid ranges.
Range Invalid Count**	21101	-	Number of invalid ranges.
Anchor Invalid Count**	21200	-	Number of frames with anchoring invalid.
Light Operational Time	21201	-	Total running time of G2 laser or G3 projector light (on Gocator firmware 5.3 or later), in minutes.
First Log Id	21301		ID of the first available log entry.
Last Log Id	21300		ID of the last available log entry. It is inclusive: for example, if first = 3 and last = 5, the available log IDs are 3, 4, 5. If no log is available, the last ID is less than the first ID.
Z-Index Drop Count	22000	-	The number of dropped surfaces due to a lack of z-encoder pulse during rotational part detection.
Tool Run Time	22004	Tool Index	The most recent time taken to execute the tool.
Part Total Emitted	22006	-	Total number of parts emitted by profile part detection.
Part Length Limit	22007	-	Number of parts emitted due to reaching the length limit.
Part Min Area Drops	22008	-	Number of parts dropped due to being smaller than the minimum area.
Part Backtrack Drops	22009	-	Number of parts dropped due to backtracking.
Parts Currently Active	22010	-	Number of parts currently being tracked.
Part Length	22011	-	Length of largest active part.
Part Start Y	22012	-	Start Y position of the largest active part.
Part Tracking State	22013	-	Tracking state of the largest active part.
Part Capacity Exceeded	22014	-	Part detection part or run capacity has been exceeded.
Part X Position	22015	-	Center X position of the largest active part.
Tool Runtime Minimum	22016	-	Minimum time spent for tool to process a sample
Tool Runtime Maximum	22017	-	Maximum time spent for tool to process a sample
Tool Runtime Average	22018	-	Average time for tool to process a sample
Tool Runtime Percent Average	22019	-	Average percentage of total time spent running this tool
Bar Alignment Status	22020	-	Status of the buffered bar alignment when aligning:

Indicator	ID	Instance	Value
			1 – buffer leveling in progress 2 – buffer searching in progress 3 – buffer scanning in progress 4 – buffer padding in progress 5 – buffering complete; processing alignment on buffered data 11 – alignment leveling in progress 12 – alignment searching in progress 13 – alignment fitting in progress 14 – alignment complete 15 – alignment completed but failed 16 – alignment cancelled
Value	30000	Measurement ID	Measurement Value.
Pass	30001	Measurement ID	Number of pass decision.
Fail	30002	Measurement ID	Number of fail decision.
Min	30003	Measurement ID	Minimum measurement value.
Max	30004	Measurement ID	Maximum measurement value.
Average	30005	Measurement ID	Average measurement value.
Std. Dev.	30006	Measurement ID	Measurement value standard deviation.
Invalid Count	30007	Measurement ID	Number of invalid values.
Overflow	30008	Measurement ID	Number of times this measurement has overflown on any output. Multiple simultaneous overflows result in only a single increment to this counter. Overflow conditions include: -Value exceeds bit representation available for given protocol When a measurement value overflow occurs, the value is set to the null value appropriate for the given protocol's measurement value output type. The Overflow health indicator increments.

* When the sensor is accelerated, the indicator's value is reported from the accelerating PC.

** When the sensor is accelerated, the indicator's value is the sum of the values reported from the sensor and the accelerating PC.

Data Results

A client can receive data messages from a sensor by connecting to the Data TCP channel (port 3196).

The Data channel and the Health channel (port 3194) can be connected at the same time. The sensor accepts multiple connections on each port. For more information on the Health channel, see *Health Results* on page 296.

Messages that are received on the Data and Health channels use a common structure, called Gocator Data Protocol (GDP). Each message consists of a 6-byte header, containing *size* and *control* fields, followed by a variable-length, message-specific content section. The structure of the GDP message is defined below.

Gocator Data Protocol

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last Message flag Bits 0-14: Message type identifier. (See individual data result sections.)

Messages are always sent in groups. The Last Message flag in the *control* field is used to indicate the final message in a group. If there is only one message per group, this bit will be set in each message.

Stamp

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last message flag. Bits 0-14: Message type identifier. For this message, set to 1.
count (C)	32u	6	Count of stamps in this message.
size	16u	10	Stamp size, in bytes (min: 56, current: 56).
source	8u	12	Source (0 – Main).
reserved	8u	13	Reserved.
stamps[C]	Stamp	14	Array of stamps (see below).

Stamp

Field	Type	Offset	Description
frameIndex	64u	0	Frame index (counts up from zero).
timestamp	64u	8	Timestamp (µs).
encoder	64s	16	Current encoder value (ticks).
encoderAtZ	64s	24	Encoder value latched at z/index mark (ticks).
status	64u	32	Bit field containing various frame information: Bit 0: sensor digital input state Bit 4: master digital input state Bit 8-9: inter-frame digital pulse trigger. (Master digital input if master is connected, otherwise sensor digital input. Value is cleared after each frame and clamped at 3 if more than 3

Field	Type	Offset	Description
			pulses are received).
serialNumber	32u	40	Sensor serial number. (In a dual-sensor system, the serial number of the main sensor.)
reserved[5]	32u	44	Reserved.

Video

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last message flag. Bits 0-14: Message type identifier. For this message, set to 2.
attributesSize	16u	6	Size of attributes, in bytes (min: 20, current: 20).
height (H)	32u	8	Image height, in pixels.
width (W)	32u	12	Image width, in pixels.
pixelSize	8u	16	Pixel size, in bytes.
pixelFormat	8u	17	Pixel format: 1 – 8-bit greyscale 2 – 8-bit color filter 3 – 8-bits-per-channel color (B, G, R, X)
colorFilter	8u	18	Color filter array alignment: 0 – None 1 – Bayer BG/GR 2 – Bayer GB/RG 3 – Bayer RG/GB 4 – Bayer GR/BG
source	8u	19	Source 0 – Top 1 – Bottom 2 – Top Left 3 – Top Right
cameraIndex	8u	20	Camera index.
exposureIndex	8u	21	Exposure index.
exposure	32u	22	Exposure (ns).
flippedX	8u	26	Indicates whether the video data must be flipped horizontally to match up with profile data.
flippedY	8u	27	Indicates whether the video data must be flipped vertically to match up with profile data.
streamStep	32s	28	Data stream step number. For video, values are:

Field	Type	Offset	Description
			0 – video stream step 8 – tool data stream step
streamStepId	32s	32	Data stream step identifier within the stream step.
pixels[H][W]	(Variable)	36	Image pixels. (Depends on pixelSize above.)

Profile Point Cloud

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last message flag. Bits 0-14: Message type identifier. For this message, set to 5.
attributeSize	16u	6	Size of attributes, in bytes (min: 32, current: 32).
count (C)	32u	8	Number of profile arrays.
width (W)	32u	12	Number of points per profile array.
xScale	32u	16	X scale (nm).
zScale	32u	20	Z scale (nm).
xOffset	32s	24	X offset (µm).
zOffset	32s	28	Z offset (µm).
Source	8u	32	Source 0 – Top 1 – Bottom 2 – Top Left 3 – Top Right
exposure	32u	33	Exposure (ns).
cameraIndex	8u	37	Camera index.
reserved[2]	8u	38	Reserved.
streamStep	32s	40	Stream step
streamStepId	32s	44	Data stream step identifier within the stream step.
ranges[C][W]	Point16s	48	Profile ranges.

Measurement

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last message flag. Bits 0-14: Message type identifier. For this message, set to 10.

Field	Type	Offset	Description
count (C)	32u	6	Count of measurements in this message.
reserved[2]	8u	10	Reserved.
id	16u	12	Measurement identifier.
measurements[C]	Measurement	14	Array of measurements (see below).

Measurement

Field	Type	Offset	Description
value	32s	0	Measurement value.
decision	8u	4	Measurement decision bitmask. Bit 0: 1 – Pass 0 – Fail Bits 1-7: 0 – Measurement value OK 1 – Invalid value 2 – Invalid anchor
reserved[3]	8u	5	Reserved.

Alignment Result

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last message flag. Bits 0-14: Message type identifier. For this message, set to 11.
attributesSize	16u	6	Size of attributes, in bytes (min: 8, current: 8).
opId	32u	8	Operation ID.
status	32s	12	Operation status. 1 – OK 0 – General failure -1 – No data in the field of view for stationary alignment -2 – No profiles with sufficient data for line fitting for travel alignment -3 – Invalid target detected. Examples include: - Calibration disk diameter too small. - Calibration disk touches both sides of the field of view. - Too few valid data points after outlier rejection. -4 – Target detected in an unexpected position. -5 – No reference hole detected in bar alignment.

Field	Type	Offset	Description
			-6 – No change in encoder value during travel calibration -988 – User aborted -993 – Timed out -997 – Invalid parameter

Exposure Calibration Result

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last message flag. Bits 0-14: Message type identifier. For this message, set to 12.
attributesSize	16u	6	Size of attributes, in bytes (min: 8, current: 8).
opId	32u	8	Operation ID.
status	32s	12	Operation status.
exposure	32u	16	Exposure result (ns).

Event

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last message flag. Bits 0-14: Message type identifier. For this message, set to 22.
attributesSize	16u	6	Size of attributes, in bytes (min: 8, current: 8).
eventType	32u	8	The type of event: 0 – Exposure Begin 1 – Exposure End
length	32u	12	The number of bytes containing additional data.
data[length]	8u	16	Additional data.

Tracheid

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last message flag. Bits 0-14: Message type identifier. For this message, set to 23.
attributesSize	16u	6	Size of attributes, in bytes (min: 10, current: 10).
frameCount	32u	8	The number of data frames.
pointCount	32u	12	The number of spots.

Field	Type	Offset	Description
source	8u	16	Source: 0 – Top 1 – Bottom 2 – Top Left 3 – Top Right
cameraIndex	8u	17	The camera the data came from.
momentData[frameCount, pointCount * 6]	32s	18	Tracheid moments for each point.

Feature Point

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last message flag. Bits 0-14: Message type identifier. For this message, set to 24.
id	16u	6	Feature Id
Point.x	64s	8	X Coordinate of Point (Scaled by 10^6)
Point.y	64s	16	Y Coordinate of Point (Scaled by 10^6)
Point.z	64s	24	Z Coordinate of Point (Scaled by 10^6)

Feature Line

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last message flag. Bits 0-14: Message type identifier. For this message, set to 25.
id	16u	6	Feature Id
Point.x	64s	8	X Coordinate of Point (Scaled by 10^6)
Point.y	64s	16	Y Coordinate of Point (Scaled by 10^6)
Point.z	64s	24	Z Coordinate of Point (Scaled by 10^6)
Direction.x	64s	32	X Component of Direction Vector (Scaled by 10^6)
Direction.y	64s	40	Y Component of Direction Vector (Scaled by 10^6)
Direction.z	64s	48	Z Component of Direction Vector (Scaled by 10^6)

Feature Plane

Field	Type	Offset	Description
size	32u	0	Count of bytes in message (including this field).
control	16u	4	Bit 15: Last message flag.

Field	Type	Offset	Description
			Bits 0-14: Message type identifier. For this message, set to 26.
id	16u	6	Feature Id
Normal.x	64s	8	X Component of Normal Vector (Scaled by 10^6)
Normal.y	64s	16	Y Component of Normal Vector (Scaled by 10^6)
Normal.z	64s	24	Z Component of Normal Vector (Scaled by 10^6)
originDistance	64s	32	Distance to Origin (Scaled by 10^6)

Modbus Protocol

Modbus is designed to allow industrial equipment such as Programmable Logic Controllers (PLCs), sensors, and physical input/output devices to communicate over an Ethernet network.

Modbus embeds a Modbus frame into a TCP frame in a simple manner. This is a connection-oriented transaction, and every query expects a response.

This section describes the Modbus TCP commands and data formats. Modbus TCP communication lets the client:

- Switch jobs.
- Align and run sensors.
- Receive measurement results, sensor states, and stamps.

To use the Modbus protocol, it must be enabled and configured in the active job. For information on configuring the protocol using the Web interface, see *Ethernet Output* on page 171.

> The Gocator 4.x/5.x firmware uses mm, mm^2, mm^3, and degrees as standard units. In all protocols, values are scaled by 1000, as values in the protocols are represented as integers. This results in effective units of mm/1000, mm^2/1000, mm^3/1000, and deg/1000 in the protocols.

If buffering is enabled with the Modbus protocol, the PLC must read the Buffer Advance output register (see *State* on page 312) to advance the queue before reading the measurement results.

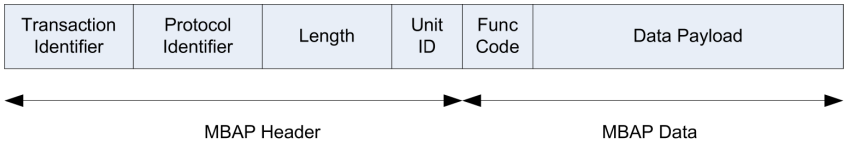
Concepts

A PLC sends a command to start each sensor. The PLC then periodically queries each sensor for its latest measurement results. In Modbus terminology, the PLC is a Modbus Client. Each sensor is a Modbus Server which serves the results to the PLC.

The Modbus protocol uses TCP for connection and messaging. The PLC makes a TCP connection to the sensor on port 502. Control and data messages are communicated on this TCP connection. Up to eight clients can be connected to the sensor simultaneously. A connection closes after 10 minutes of inactivity.

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

Registers

Modbus registers are 16 bits wide and are either control registers or output registers.

Control registers are used to control the sensor states (e.g., start, stop, or calibrate a sensor).

The output registers report the sensor states, stamps, and measurement values and decisions. You can read multiple output registers using a single Read Holding Registers or a single Read Input Registers command. Likewise, you can control the state of the sensor using a single Write Multiple Register command.

Control registers are write-only, and output registers are read-only.

Register Map Overview

Register Address	Name	Read/Write	Description
0 - 124	Control Registers	WO	Registers for Modbus commands. See *Control Registers* below for detailed descriptions.
300 - 899	Sensor States	RO	Report sensor states. See *State* on the next page for detailed descriptions.
900 - 999	Stamps	RO	Return stamps associated with each. See *State* on the next page for detailed descriptions.
1000 - 1998	Measurements & Decisions	RO	333 measurement and decision pairs. See *Measurement Registers* on page 315 for detailed descriptions.

Control Registers

Control registers are used to operate the sensor. Register 0 stores the command to be executed. Subsequent registers contain parameters for the commands if applicable. The sensor executes a command when the value in register 0 is changed. To set the parameters before a command is executed, you should set up the parameters and the command using a single Multiple Write register command.

Control Register Map

Register Address	Name	Read/Write	Description
0	Command Register	WO	Takes a 16-bit command. For a list of the available commands, see table below.
1 – 64	Command Parameters	WO	For **Load Job** (5) command: Null-terminated filename. Each 16-bit register holds a single character. Specifies the filename. If the file extension ".job" is missing, it is automatically appended to the filename. For **Set Runtime Variables** (6) command: Registers 1-8 are used to set the values of the runtime variables.

The 16-bit values used for Command Register are described below.

Command Register Values

Value	Name	Description
0	Stop Running	Stops the sensor. No effect if sensor is already stopped.
1	Start Running	Starts the sensor. No effect if sensor is already started.
2	Align (stationary target)	Starts the stationary alignment process. State register 301 will be set to 1 (busy). When the alignment process is complete, the register is set back to zero.
3	Align (moving target)	Starts moving alignment process and also calibrate encoder resolution. State register 301 will be set to 1 (busy). When the alignment process is complete, the register is set back to zero.
4	Clear Alignment	Clears the alignment.
5	Load Job	Activates the specified job file. Set registers 1-64 to the null-terminated filename, one filename character per 16-bit register, including the null terminator character. The ".job" extension is optional; if it is missing, it is automatically appended to the file name.
6	Set Runtime Variables	Sets the runtime variables. Set registers 1 through 8 to the values of all four 32-bit runtime variables.
7	Software trigger	Software trigger the sensor to capture one frame. The sensor must already be running, in trigger mode “Software”. Otherwise, software trigger has no effect.

Output Registers

Output registers are used to output states, stamps, and measurement results. Each register address holds a 16-bit data value.

State

State registers report the current sensor state.

State Register Map

Register Address	Name	Type	Description
300	Sensor State	16u	Sensor State: 0 - Stopped 1 - Running
301	Modbus Command in Progress	16u	1 when the sensor is busy performing the last command, 0 when done. Registers 302 and 311-371 below are only valid when there is no command in progress.
302	Alignment State	16u	Current Alignment State:

Register Address	Name	Type	Description
			0 - Not aligned 1- Aligned (Valid when register 301 = 0.)
303	Encoder Position High	64u	Current encoder position (64-bit value, requiring four 16-bit registers)
304	Encoder		
305	Encoder		
306	Encoder Low		
307	Time High	64s	Uptime timestamp (64-bit value, requiring four 16-bit registers)
308	Time		
309	Time		
310	Time Low		
311	Job File Name Length	16u	Number of characters in the current job file name. (Valid when register 301 = 0.)
312 – 371	Live Job Name	16u	Name of currently loaded job file. Does not include the extension. Each 16-bit register contains a single character. (Valid when register 301 = 0.)
375	Runtime Variable 0 High	32s	Runtime variable value stored in two register locations.
376	Runtime Variable 0 Low		
...	...	...	...
381	Runtime Variable 3 High	32s	Runtime variable value stored in two register locations.
382	Runtime Variable 3 Low		

Stamp

Stamps contain trigger timing information used for synchronizing a PLC's actions. A PLC can also use this information to match up data from multiple sensors.

Stamp Register Map

Register Address	Name	Type	Description
960-975	reserved		Not used.
976	Buffer Advance Register	16u	If buffering is enabled, this address must be read by the PLC Modbus client first to advance the buffer. After the buffer advance read operation, the Modbus client can read the updated Measurements & Decisions in addresses 1000-

Register Address	Name	Type	Description
			1059.
977	Buffer Count	16u	Number of buffered messages currently in the queue.
978	Buffer Overflow Flag	16u	Buffer Overflow Indicator: 0 - No overflow 1 - Overflow. (Indicates data is being lost.)
979	Inputs	16u	Digital input state of the last frame.
980	zPosition High	64u	Encoder position at time of last index pulse. 64-bit value, requiring four 16-bit registers.
981	zPosition		
982	zPosition		
983	zPosition Low		
984	Exposure High	32u	Laser exposure (µs) of the last frame. Stored in two register locations.
985	Exposure Low		
986	Temperature High	32u	Sensor temperature in degrees Celcius * 100 (centidegrees) of the last frame. Stored in two register locations.
987	Temperature Low		
988	Encoder Position High	64u	Encoder position of the last frame when the image data was scanned/taken. 64-bit value, requiring four 16-bit registers.
989	Encoder Position		
990	Encoder Position		
991	Encoder Position Low		
992	Time High	64u	Time stamp in microseconds of the last frame. 64-bit value, requiring four 16-bit registers.
993	Time		
994	Time		
995	Time Low		
996	Frame Index High	64u	The frame number of the last frame. 64-bit value, requiring four 16-bit registers.
997	Frame Index		
998	Frame Index		
999	Frame Index Low		

Measurement Registers

Measurement results are reported in pairs of values and decisions. Measurement values are 32 bits wide and decisions are 8 bits wide.

The measurement ID is used to find the register address of each pair. The register address of the first word can be calculated as (1000 + 3 * ID). For example, a measurement with ID set to 4 can be read from registers 1012 (high word) and 1013 (low word), and the decision at 1015.

Measurement Register Map

Register Address	Name	Type	Description
1000	Measurement 0 High	32u	Measurement value in µm (0x80000000 if invalid)
1001	Measurement 0 Low		
1002	Decision 0	16u	Measurement decision. A bit mask, where: Bit 0: 1 - Pass 0 - Fail Bits 1-7: 0 - Measurement value OK 1 - Invalid value 2 - Invalid anchor
1003	Measurement 1 High		
1004	Measurement 1 Low		
1005	Decision 1		
1006	Measurement 2 High		
1007	Measurement 2 Low		
1008	Decision 2		
...	...	...	...
1996	Measurement 332 High		
1997	Measurement 332 Low		
1998	Decision 332		

EtherNet/IP Protocol

EtherNet/IP is an industrial protocol that allows bidirectional data transfer with PLCs. It encapsulates the object-oriented Common Industrial Protocol (CIP). EtherNet/IP communication enables the client to:

- Switch jobs.
- Align and run sensors.
- Receive sensor states, stamps, and measurement results.
- Set and retrieve runtime variables.

This section describes the EtherNet/IP messages and data formats.

Note that in firmware version 5.2, the identity information was updated as follows:

Attribute	Before Firmware 5.2	Firmware 5.2 and later
Product Code	Was 1000, 2000, or 3000 depending on the model.	Now 1.
Major Revision	Matched firmware major version.	Now 1.
Minor Revision	Matched firmware minor version.	Now 1.

This update may require a change on a device attempting to connect to a sensor via EtherNet/IP. A compatible EDS file can be downloaded from the web interface. If the existing EDS must be maintained, the device can be configured to disable electronic keying, ignoring the product code and version numbers.

To use the EtherNet/IP protocol, it must be enabled and configured in the active job. For information on configuring the protocol using the Web interface, see *Ethernet Output* on page 171.

> The Gocator 4.x/5.x firmware uses mm, mm^2, mm^3, and degrees as standard units. In all protocols, values are scaled by 1000, as values in the protocols are represented as integers. This results in effective units of mm/1000, mm^2/1000, mm^3/1000, and deg/1000 in the protocols.

Sensors support unconnected or connected explicit messaging (with TCP), as well as implicit (or I/O) messaging. For information on explicit messaging assemblies and objects, see *Explicit Messaging* below. For information on implicit messaging assemblies and objects, see *Implicit Messaging* on page 323.

Explicit Messaging

To EtherNet/IP-enabled devices on the network, the sensor information is seen as a collection of objects, which have attributes that can be queried.

Sensors support all required objects for explicit messaging, such as the Identity object, TCP/IP object, and Ethernet Link object. In addition, an Assembly object is used for sending sensor and sample data and receiving commands. The Assembly object contains four assemblies: the command assembly (32 bytes), the runtime variable configuration assembly (64 bytes), the sensor state assembly (100 bytes), and the sample state assembly object (380 bytes). The data attribute (0x03) of the assembly objects is a byte array containing information about the sensor. The data attribute can be accessed with the Get Attribute and Set Attribute commands.

The PLC sends a command to start a sensor. The PLC then periodically queries the attributes of the assembly objects for its latest measurement results. In EtherNet/IP terminology, the PLC is a scanner and the sensor is an adapter.

The following sections describe the explicit messaging assemblies and objects.

Identity Object (Class 0x01)

Attribute	Name	Type	Value	Description	Access
1	Vendor ID	UINT	1256	ODVA-provided vendor ID	Get
2	Device Type	UINT	43	Device type	Get
3	Product Code	UINT	1	Product code	Get
4	Revision	USINT	1.1	Byte 0 - 1 Byte 1 - 1	Get
6	Serial number	UDINT	32-bit value	Sensor serial number	Get
7	Product Name	SHORT STRING 32	"Gocator"	Gocator product name	Get

TCP/IP Object (Class 0xF5)

The TCP/IP Object contains read-only network configuration attributes such as IP Address. TCP/IP configuration via Ethernet/IP is not supported. See Volume 2, Chapter 5-3 of the CIP Specification for a complete listing of TCP/IP object attributes.

Attribute	Name	Type	Value	Description	Access
1	Status	UDINT	0	TCP interface status	Get
2	Configuration Capability	UINT	0		Get
3	Configuration Control	UINT	0	Product code	Get
4	Physical Link Object	Structure (See description)		See 5.3.3.2.4 of CIP Specification Volume 2: Path size (UINT) Path (Padded EPATH)	Get
5	Interface Configuration	Structure (See description)		See 5.3.3.2.5 of CIP Specification Volume	Get

Attribute	Name	Type	Value	Description	Access
				2: IP address (UDINT) Network mask (UDINT) Gateway address (UDINT) Name server (UDINT) Secondary name (UDINT) Domain name (UDINT)	

Ethernet Link Object (Class 0xF6)

The Ethernet Link Object contains read-only attributes such as MAC Address (Attribute 3). See Volume 2, Chapter 5-4 of the CIP Specification for a complete listing of Ethernet Link object attributes.

Attribute	Name	Type	Value	Description	Access
1	Interface Speed	UDINT	1000	Ethernet interface data rate (mbps)	Get
2	Interface Flags	UDINT		See 5.4.3.2.1 of CIP Specification Volume 2: Bit 0: Link Status 0 – Inactive 1 - Active Bit 1: Duplex 0 – Half Duplex 1 – Full Duplex	Get
3	Physical Address	Array of 6 USINTs		MAC address (for example: 00 16 20 00 2E 42)	Get

Assembly Object (Class 0x04)

For explicit messaging, the Ethernet/IP object model includes the following assemblies: command, runtime variable configuration, sensor state, and sample state.

All assembly object instances are static. Data in a data byte array in an assembly object are stored in the big endian format.

Command Assembly

The command assembly object is used to start, stop, and align the sensor, and also to switch jobs on the sensor.

Command Assembly

Information	Value
Class	0x4
Instance	0x310
Attribute Number	3
Length	32 bytes
Supported Service	0x10 (SetAttributeSingle)

Attributes 1 and 2 are not implemented, as they are not required for the static assembly object.

Attribute 3

Attribute	Name	Type	Value	Description	Access
3	Command	Byte Array	See Below	Command parameters Byte 0 - Command. See table below for specification of the values.	Get, Set

Command Definitions

Value	Name	Description
0	Stop running	Stop the sensor. No action if the sensor is already stopped
1	Start Running	Start the sensor. No action if the sensor is already started.
2	Stationary Alignment	Start the stationary alignment process. Byte 1 of the sensor state assembly will be set to 1 (busy) until the alignment process is complete, then back to zero.
4	Clear Alignment	Clear the alignment.
5	Load Job	Load the job. Set bytes 1-31 to the file name (one character per byte. File name must be null-terminated. The job name and extension are case-sensitive. If the extension ".job" is missing, it is automatically appended to the file name.
6	Reserved	Do not use.
7	Software trigger	Sends a software trigger to the sensor to capture one frame. The sensor must already be running, and its trigger mode must be set to "Software". Otherwise, software trigger has no effect.

Runtime Variable Configuration Assembly

The runtime variable configuration assembly object contains the sensor's intended runtime variables.

Runtime Variable Configuration Assembly

Information	Value
Class	0x04
Instance	0x311
Attribute Number	3
Length	64 bytes
Supported Service	0x10 (SetAttributeSingle)

Attribute 3

Attribute	Name	Type	Value	Description	Access
3	Command	Byte Array	See below	Runtime variable configuration information. See below for more details.	Get

Sensor State Information

Byte	Name	Type	Description
0-3	Runtime Variable 0	32s	Stores the intended value of the Runtime Variable at index 0.
4-7	Runtime Variable 1	32s	Stores the intended value of the Runtime Variable at index 1.
8-11	Runtime Variable 2	32s	Stores the intended value of the Runtime Variable at index 2.
12-15	Runtime Variable 3	32s	Stores the intended value of the Runtime Variable at index 3.
16-63	Reserved		

Sensor State Assembly

The sensor state assembly object contains the sensor's states, such as the current sensor temperature, frame count, and encoder values.

Sensor State Assembly

Information	Value
Class	0x04
Instance	0x320
Attribute Number	3
Length	100 bytes
Supported Service	0x0E (GetAttributeSingle)

Attributes 1 and 2 are not implemented, as they are not required for the static assembly object.

Attribute 3

Attribute	Name	Type	Value	Description	Access
3	Command	Byte Array	See below	Sensor state information. See below for more details.	Get

Sensor State Information

Byte	Name	Type	Description
0	Sensor State		Sensor state: 0 - Stopped 1 - Running
1	EtherNet/IP Command in Progress		Command busy status: 0 - Not busy 1 - Busy performing the last command Bytes 2 and 19-83 below are only valid when there is no command in progress.
2	Alignment State		Alignment status: 0 - Not aligned

Byte	Name	Type	Description
			1 - Aligned The value is only valid when byte1 is set to 0.
3-10	Encoder	64s	Current encoder position
11-18	Time	64s	Current timestamp
19	Current Job Filename Length	8u	Number of characters in the current job filename. (e.g., 11 for "current.job"). The length includes the .job extension. Valid when byte 1 = 0.
20-83	Current Job Filename		Name of currently loaded job, including the ".job" extension. Each byte contains a single character. Valid when byte 1 = 0.
84-87	Runtime Variable 0	32s	Runtime variable value at index 0
...	...		
96-99	Runtime Variable 3	32s	Runtime variable value at index 3

Sample State Assembly

The sample state object contains measurements and their associated stamp information.

Sample State Assembly

Information	Value
Class	0x04
Instance	0x321
Attribute Number	3
Length	380 bytes
Supported Service	0x0E (GetAttributeSingle)

Attribute 3

Attribute	Name	Type	Value	Description	Access
3	Command	Byte Array	See below	Sample state information. See below for more details.	Get

Sample State Information

Byte	Name	Type	Description
0-1	Inputs	16u	Digital input state of the last frame.
2-9	Z Index Position	64u	Encoder position at time of last index pulse of the last frame.
14-17	Temperature	32u	Sensor temperature in degrees Celsius * 100 (centidegrees) of the last frame.
18-25	Encoder Position	64u	Encoder position of the last frame when the

Byte	Name	Type	Description
			image data was scanned/taken.
26-33	Time	64u	Time stamp in microseconds of the last frame.
34-41	Frame Counter	64u	The frame number of the last frame.
42	Buffer Count	8u	Represents the number of frames waiting to be output if buffering is enabled.
43	Buffer Overflowing	8u	Indicates whether the output buffer has overflowed: 0 - No overflow 1 - Overflow
44 - 79	Reserved		Reserved bytes.
80-83	Measurement 0	32s	Measurement value in µm (0x80000000 if invalid).
84	Decision 0	8u	Measurement decision. A bit mask, where: Bit 0: 1 - Pass 0 - Fail Bits 1-7: 0 - Measurement value OK 1 - Invalid value 2 - Invalid anchor
...	...		
375-378	Measurement 59	32s	Measurement value in µm (0x80000000 if invalid).
379	Decision 59	8u	Measurement decision. A bit mask, where: Bit 0: 1 - Pass 0 - Fail Bits 1-7: 0 - Measurement value OK 1 = Invalid value 2 = Invalid anchor

Measurement results are reported in pairs of values and decisions. Measurement values are 32 bits wide and decisions are 8 bits wide.

The measurement ID defines the byte position of each pair within the state information. The position of the first word can be calculated as (80 + 5 * ID). For example, a measurement with ID set to 4 can be read from byte 100 (high word) to 103 (low word) and the decision at 104.

If buffering is enabled in the Ethernet Output panel, reading the Extended Sample State Assembly Object automatically advances the buffer. See *Ethernet Output* on page 171 for information on the **Output** panel.

Implicit Messaging

Implicit messaging uses UDP and is faster than explicit messaging, and is ideal for time-critical applications. However, implicit messaging is layered on top of UDP. UDP is connectionless and data delivery is not guaranteed. For this reason, implicit messaging is only suitable for applications where occasional data loss is acceptable.

For detailed information on setting up implicit messaging using Allen-Bradley PLCs, see http://lmi3d.com/sites/default/files/APPNOTE_Implicit_Messaging_with_Allen-Bradley_PLCs.pdf.

The following sections describe the implicit messaging assemblies.

Assembly Object (Class 0x04)

For implicit messaging, the Ethernet/IP object model includes the following assemblies: implicit messaging command and implicit messaging output.

All assembly object instances are static. Data in a data byte array in an assembly object are stored in the big endian format.

Implicit Messaging Command Assembly

Implicit Messaging Command Assembly

Information	Value
Class	0x04
Instance	0x64
Attribute Number	3
Length	32 bytes

Implicit Messaging Command Assembly Information

Byte	Name	Type	Description
0	Command	8u	A bit mask where setting the following bits will only perform the action with highest priority*: 1 – Stop sensor 2 – Start sensor 4 – Perform stationary alignment 8 – Perform moving alignment 16 – Clear alignment 32 – Set runtime variables 64 – Load job file 128 – Software trigger

Byte	Name	Type	Description
			*The priority of commands is currently as follows: 1. Stop sensor 2. Start sensor 3. Perform stationary alignment 4. Perform moving alignment 5. Clear alignment 6. Set runtime variables 7. Load job file 8. Software trigger
1-31	Reserved (except for configuring runtime variables and loading job file)		If you are setting the runtime variables, use bytes 4-19 to define the values of each of the four runtime variables in little endian format. If you are loading job file, use bytes 1-31 for the filename, one character per byte. The job name and extension are case-sensitive. The filename must be null terminated and must end with ".job".

Implicit Messaging Output Assembly

Implicit Messaging Output Assembly

Information	Value
Class	0x04
Instance	0x322
Attribute Number	3
Length	376 bytes

Implicit Messaging Output Assembly Information

Byte	Name	Type	Description
0	Sensor State	8u	Sensor state is a bit mask where: Bit 0: 1 – Running 0 – Stopped Bit 1: 1 – Conflict due to unreachable buddy 0 – No conflict Bits [2-7]: Not used.
1	Alignment and Command	8u	A bit mask where:

Byte	Name	Type	Description
	state		Bit 0: 1 – Explicit or Implicit Command in progress 0 – No Explicit or Implicit command is in progress Bit 1 1 – Aligned 0 – Not aligned
2-3	Inputs	16u	Digital input state of the last frame.
4-11	Z Index Position	64u	Encoder position at time of last index pulse of the last frame.
12-15	Exposure	32u	Exposure in µs of the last frame.
16-19	Temperature	32u	Sensor temperature in degrees celsius * 100 (centidegrees) of the last frame.
20-27	Encoder Position	64s	Encoder position of the last frame when the image data was scanned/taken.
28-35	Time	64u	Time stamp in microseconds of the last frame.
36-43	Frame Index	64u	The frame number of the last frame.
44-55	Reserved		
56	Decision 0	8u	Measurement decision is a bit mask where: Bit 0: 1 – Pass 0 – Fail Bits [1-7]: 0 – Measurement value OK 1 – Invalid Value 2 – Invalid Anchor
…	…		
119	Decision 63	8u	Measurement decision is a bit mask where: Bit 0: 1 – Pass 0 – Fail Bits [1-7]: 0 – Measurement value OK 1 – Invalid Value 2 – Invalid Anchor
120-123	Measurement 0	32s	Measurement value in µm.

Byte	Name	Type	Description
			(0x80000000 if invalid)
...	...		
372-375	Measurement 63	32s	Measurement value in µm. (0x80000000 if invalid)

ASCII Protocol

This section describes the Ethernet ASCII protocol.

The protocol communicates using ASCII strings. The output result format from the sensor is user-configurable.

To use the ASCII protocol, it must be enabled and configured in the active job.

> The Gocator 4.x/5.x firmware uses mm, mm^2, mm^3, and degrees as standard units. In all protocols, values are scaled by 1000, as values in the protocols are represented as integers. This results in effective units of mm/1000, mm^2/1000, mm^3/1000, and deg/1000 in the protocols.

For information on configuring the protocol with the Web interface (when using the protocol over Ethernet), see *Ethernet Output* on page 171.

Connection Settings

Ethernet Communication

With Ethernet ASCII output, you can set the connection port numbers of the three channels used for communication (Control, Data, and Health):

Ethernet Ports for ASCII

Name	Description	Default Port
Control	To send commands to control the sensor.	8190
Data	To retrieve measurement output.	8190
Health	To retrieve specific health indicator values.	8190

Channels can share the same port or operate on individual ports. The following port numbers are reserved for sensor internal use: 2016, 2017, 2018, and 2019. Each port can accept multiple connections, up to a total of 16 connections for all ports.

Polling Operation Commands (Ethernet Only)

On the Ethernet output, the Data channel can operate asynchronously or by polling.

Under asynchronous operation, measurement results are automatically sent on the Data channel when the sensor is in the running state and results become available. The result is sent on all connected data channels.

Under polling operation, a client can:

- Switch to a different job.
- Align, run, and trigger sensors.
- Receive sensor states, health indicators, stamps, and measurement results

A sensor sends Control, Data, and Health messages over separate channels. The Control channel is used for commands such as starting and stopping the sensor, loading jobs, and performing alignment (see *Command Channel* on the next page).

The Data channel is used to receive and poll for measurement results. When the sensor receives a Result command, it will send the latest measurement results on the same data channel that the request is received on. See *Data Channel* on page 331 for more information.

The Health channel is used to receive health indicators (see *Health Channel* on page 333).

Command and Reply Format

Commands are sent from the client to the sensor. Command strings are not 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.

The delimiter and termination characters are configured in the Special Character settings.

Special Characters

The ASCII Protocol has three special characters.

Special Characters

Special Character	Explanation
Delimiter	Separates input arguments in commands and replies, or data items in results. Default value is ",".
Terminator	Terminates both commands and result output. Default value is "%r%n".
Invalid	Represents invalid measurement results. Default value is "INVALID"

The values of the special characters are defined in the Special Character settings. In addition to normal ASCII characters, the special characters can also contain the following format values.

Format values for Special Characters

Format Value	Explanation
%t	Tab
%n	New line
%r	Carriage return
%%	Percentage (%) symbol

Command Channel

The following sections list the actions available on the command channel.

Optional parameters are shown in italic. The placeholder for data is surrounded by brackets (<>). In the examples, the delimiter is set to ','.

Start

The Start command starts the sensor system (causes it to enter the Running state). This command is only valid when the system is in the Ready state. If a start target is specified, the sensor starts at the target time or encoder (depending on the trigger mode).

Formats

Message	Format
Command	Start,start target The start target (optional) is the time or encoder position at which the sensor will be started. The time and encoder target value should be set by adding a delay to the time or encoder position returned by the Stamp command. The delay should be set such that it covers the command response time of the Start command.
Reply	OK or ERROR, <Error Message>

Examples:

```
Command: Start
Reply: OK
Command: Start,1000000
Reply: OK
Command: Start
Reply: ERROR, Could not start the sensor
```

Stop

The stop command stops the sensor system (causes it to enter the Ready state). This command is valid when the system is in the Ready or Running state.

Formats

Message	Format
Command	Stop
Reply	OK or ERROR, <Error Message>

Examples:

```
Command: Stop
Reply: OK
```

Trigger

The Trigger command triggers a single frame capture. This command is only valid if the sensor is configured in the Software trigger mode and the sensor is in the Running state.

Formats

Message	Format
Command	Trigger
Reply	OK or ERROR, <Error Message>

Examples:

```
Command: Trigger
Reply: OK
```

LoadJob

The Load Job command switches the active sensor configuration.

Formats

Message	Format
Command	LoadJob,job file name If the job file name is not specified, the command returns the current job name. An error message is generated if no job is loaded. ".job" is appended if the filename does not have an extension.
Reply	OK or ERROR, <Error Message>

Examples:

```
Command: LoadJob,test.job
Reply: OK,test.job loaded successfully
Command: LoadJob
Reply: OK,test.job
Command: LoadJob,wrongname.job
Reply: ERROR, failed to load wrongname.job
```

Stamp

The Stamp command retrieves the current time, encoder, and/or the last frame count.

Formats

Message	Format
Command	Stamp,time,encoder,frame If no parameters are given, time, encoder, and frame will be returned. There could be more than one selection.
Reply	If no arguments are specified: OK, time, <time value>, encoder, <encoder position>, frame, <frame count> ERROR, <Error Message> If arguments are specified, only the selected stamps will be returned.

Examples:

```
Command: Stamp
Reply: OK,Time,9226989840,Encoder,0,Frame,6
Command: Stamp,frame
Reply: OK,6
```

Stationary Alignment

The Stationary Alignment command performs an alignment based on the settings in the sensor's live job file. A reply to the command is sent when the alignment has completed or failed. The command is timed out if there has been no progress after one minute.

Formats

Message	Format
Command	StationaryAlignment
Reply	If no arguments are specified OK or ERROR, <Error Message>

Examples:

```
Command: StationaryAlignment
Reply: OK
Command: StationaryAlignment
Reply: ERROR,ALIGNMENT FAILED
```

Clear Alignment

The Clear Alignment command clears the alignment record generated by the alignment process.

Formats

Message	Format
Command	ClearAlignment
Reply	OK or ERROR, <Error Message>

Examples:

```
Command: ClearAlignment
Reply: OK
```

Set Runtime Variables

The Set Runtime Variables command sets the runtime variables, using the specified index, length, and data. Values are integers.

Formats

Message	Format
Command	setvars,index,length,data Where *data* is the delimited integer values to be set.
Reply	OK or ERROR

Examples:

```
Command: setvars,0,4,1,2,3,4
Reply: OK
```

Get Runtime Variables

The Get Runtime Variables command gets the runtime variables, using the specified index and length.

Formats

Message	Format
Command	setvars,index,length
Reply	OK,data Where *data* is the delimited data for the passed length.

Examples:

```
Command: getvars,0,4
Reply: OK,1,2,3,4
```

Data Channel

The following sections list the actions available on the data channel.

Optional parameters are shown in italic. The placeholder for data is surrounded by brackets (<>). In the examples, the delimiter is set to ','.

Result

The Result command retrieves measurement values and decisions.

Formats

Message	Format
Command	Result,measurement ID,measurement ID...
Reply	If no arguments are specified, the custom format data string is used.

Message	Format
	OK, <custom data string> ERROR, <Error Message>
	If arguments are specified,
	OK, <data string in standard format>
	ERROR, <Error Message>

Examples:

Standard data string for measurements ID 0 and 1:

```
Result,0,1
```

```
OK,M00,00,V151290,D0,M01,01,V18520,D0
```

Standard formatted measurement data with a non-existent measurement of ID 2:

```
Result,2
```

```
ERROR,Specified measurement ID not found. Please verify your input
```

Custom formatted data string (%time, %value[0], %decision[0]):

```
Result
```

```
OK,1420266101,151290,0
```

Value

The Value command retrieves measurement values.

Formats

Message	Format
Command	Value,measurement ID,measurement ID...
Reply	If no arguments are specified, the custom format data string is used.
	OK, <custom data string> ERROR, <Error Message>
	If arguments are specified,
	OK, <data string in standard format, except that the decisions are not sent> ERROR, <Error Message>

Examples:

Standard data string for measurements ID 0 and 1:

```
Value,0,1
```

```
OK,M00,00,V151290,M01,01,V18520
```

Standard formatted measurement data with a non-existent measurement of ID 2:

```
Value,2
```

```
ERROR,Specified measurement ID not found. Please verify your input
```

Custom formatted data string (%time, %value[0]):

```
Value
```

```
OK, 1420266101, 151290
```

Decision

The Decision command retrieves measurement decisions.

Formats

Message	Format
Command	Decision,measurement ID,measurement ID...
Reply	If no arguments are specified, the custom format data string is used. OK, <custom data string> ERROR, <Error Message> If arguments are specified, OK, <data string in standard format, except that the values are not sent> ERROR, <Error Message>

Examples:

Standard data string for measurements ID 0 and 1:

```
Decision,0,1
```

```
OK,M00,00,D0,M01,01,D0
```

Standard formatted measurement data with a non-existent measurement of ID 2:

```
Decision,2
```

```
ERROR,Specified measurement ID not found. Please verify your input
```

Custom formatted data string (%time, %decision[0]):

```
Decision
```

```
OK,1420266101, 0
```

Health Channel

The following sections list the actions available on the health channel.

Optional parameters are shown in italic. The placeholder for data is surrounded by brackets (<>). In the examples, the delimiter is set to ','.

Health

The Health command retrieves health indicators. See *Health Results* on page 296 for details on health indicators.

Formats

Message	Format
Command	Health,health indicator ID.Optional health indicator instance ... More than one health indicator can be specified. Note that the health indicator instance is optionally attached to the indicator ID with a '.'. If the health indicator instance field is used the delimiter cannot be set to '.'.
Reply	OK, <health indicator of first ID>, <health indicator of second ID> ERROR, <Error Message>

Examples:

```
health,2002,2017
```

```
OK,46,1674
```

```
Health
```

```
ERROR,Insufficient parameters.
```

Standard Result Format

A sensor can send measurement results either in the standard format or in a custom format. In the standard format, you select in the web interface which measurement values and decisions to send. For each measurement the following message is transmitted:

M	t_n	,	i_n	,	V	v_n	,	D	d_1	CR

Field	Shorthand	Length	Description
MeasurementStart	M	1	Start of measurement frame.
Type	t_n	n	Hexadecimal value that identifies the type of measurement. The measurement type is the same as defined elsewhere (see *Data Results* on page 301).
Id	i_n	n	Decimal value that represents the unique identifier of the measurement.
ValueStart	V	1	Start of measurement value.
Value	v_n	n	Measurement value, in decimal. The unit of the value is measurement-specific.
DecisionStart	D	1	Start of measurement decision.
Decision	d_1	1	Measurement decision, a bit mask where:

Field	Shorthand	Length	Description
			Bit 0: 1 – Pass 0 – Fail Bits 1-7: 0 – Measurement value OK 1 – Invalid value 2 - Invalid anchor

Custom Result Format

In the custom format, you enter a format string with place holders to create a custom message. The default format string is "%time, %value[0], %decision[0]".

Result Placeholders

Format Value	Name	Explanation
%time	Time	Timestamp in microseconds of the last frame.
%encoder	Encoder Position	Encoder position of the last frame when the image data was scanned/taken.
%frame	Frame Index	Frame number of the last frame.
%value[Measurement ID]	Value	Measurement value of the specified measurement ID. The ID must correspond to an existing measurement. The value output will be displayed as an integer in micrometers.
%decision [Measurement ID]	Decision	Measurement decision, where the selected measurement ID must correspond to an existing measurement. Measurement decision is a bit mask where: Bit 0: 1 – Pass 0 – Fail Bits 1-7: 0 – Measurement value OK 1 – Invalid value 2 - Invalid anchor

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.

Selcom Protocol

Gocator 200 sensors do not use the Selcom protocol.

Tools

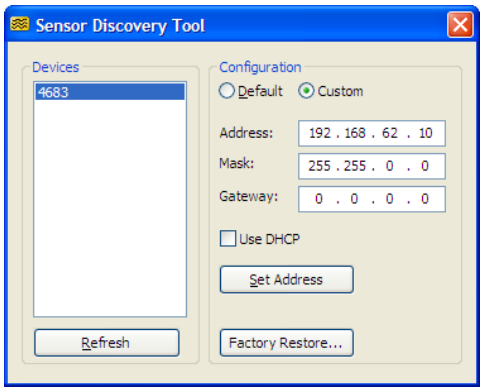
The following sections describe some of the tools provided with a Gocator sensor, as well as the CSV format that a sensor can export. For information on the integrations available with a sensor, see *Protocols* on page 256.

- Bandwidth Tool: Use this tool to dia gnose bandwidth-related issues.
- CSV Converter Tool: Used to convert CSV data exported from a sensor to several formats. See *CSV Converter Tool* on the next page.
- Discovery Tool: Used to find sensors on a network. See *Sensor Discovery Tool* below.

Sensor Discovery Tool

If a sensor's network address or administrator password is forgotten, the sensor can be discovered on the network and/or restored to factory defaults by using the Sensor Discovery software tool. This tool can be obtained from the downloads area of the LMI Technologies website: http://www.lmi3D.com.

After downloading the utility package [14405-x.x.x.x_SOFTWARE_GO_Utilities.zip], unzip the file and run the Sensor Discovery Tool [Tools > Discovery > kDiscovery.exe].



Any sensors that are discovered on the network will be displayed in the Devices list.

To change the network address of a sensor:

1. Select the **Custom** option.
2. Enter the new network address information.
3. Click **Set Address**.

To restore a sensor to factory defaults:

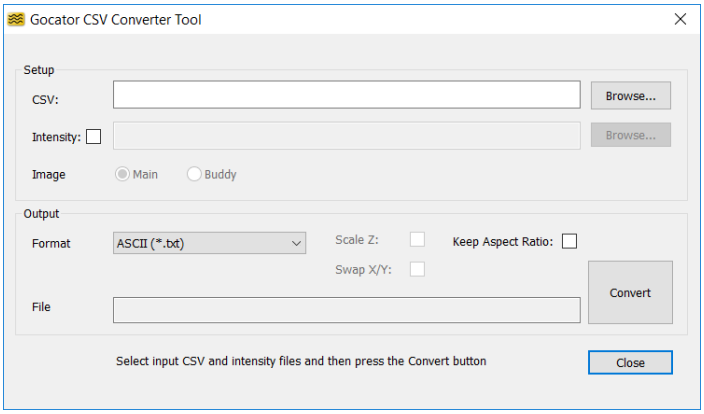
1. Select the sensor serial number in the **Devices** list.
2. Press the **Factory Restore...** button.
   Confirm when prompted.

> The Sensor Discovery tool uses UDP broadcast messages to reach sensors on different subnets. This enables the Sensor Discovery tool to locate and re-configure sensors even when the sensor IP address or subnet configuration is unknown.

CSV Converter Tool

The CSV Converter tool lets you convert data exported from a Gocator sensor in the CSV format to several formats (see table below). For more information on exporting recorded data, see *Downloading, Uploading, and Exporting Replay Data* on page 110.

For information on the CSV file format that the sensor exports, see the next section.



> The tool supports data exported from Profile mode.

To get the utility package (), go to http://lmi3d.com/support, choose your product from the Product Downloads section, and download it from the Download Center.

After downloading the tool package, unzip the file and run the Gocator CSV Converter tool [Tools > CSV Converter > kCsvConverter.exe].

The tool supports the following output formats:

Output formats

Format	Description
ASCII (XYZI)	Comma-separated points in X, Y, Z, Intensity (if available) format.
16-bit BMP	Heightmap with 16bit height values in a 5-5-5 RGB image. Not intended for visualization.

Format	Description
16-bit TIFF	Heightmap as grayscale image.
16-bit PNG	Heightmap as grayscale image.
GenTL RGB	Not intended for use with this sensor family.
GenTL Mono	Not intended for use with this sensor family.
Raw CSV	LMI Gocator CSV format for a single frame.
HexSight HIG	LMI HexSight heightmap.
STL ASCII	Mesh in standard STL text format (can become very large).
STL Binary	Mesh in binary STL format.
Wavefront OBJ	Mesh with comma-separated vertices and facets in text format.
ODSCAD OMC	ODSCAD heightmap.
MountainsMap SUR	DigitalSurf MountainsMap heightmap.
24-bit Spectrum	Color spectrum bitmap for visualization of heightmap. Does not contain height values.

With some formats, one or more of the following options are available:

Output options

Option	Description
Scale Z	Resamples the Z values to use the full value range.
Swap X/Y	Swaps the X and Y axes to obtain a right-handed coordinate system.
Keep Aspect Ratio	Resamples the X and Y axes to obtain the proper aspect ratio.

To convert exported CSV into different formats:

1. Select the CSV file to convert in the **CSV** field.
2. (Optional) If intensity information is required, check the **Intensity** box and select the intensity bitmap.
   Intensity information is only used when converting to ASCII or GenTL format. If intensity is not selected, the ASCII format will only contain the point coordinates (XYZ).
3. If a dual-sensor system was used, choose the source sensor next to **Image**.
4. Select the output format.
   For more information on output formats, see *Output formats* on the previous page.
5. (Optional) Set the **Scale Z**, **Swap X/Y**, and **Keep Aspect Ratio** options.
   Availability of these options depends on the output format you have chosen. For more information, see *Output options* above.
6. Click **Convert**.
   The converter converts the input files.

The converted file will be in the same directory as the input file. It will also have the same name as the input file but with a different file extension. The converted file name is displayed in the **Output File** field.

Troubleshooting

Review the guidance in this chapter if you are experiencing difficulty with a sensor system.

If the problem that you are experiencing is not described in this section, see *Return Policy* on page 370.

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.
- Use the Sensor Recovery tool to verify that the sensor has the correct network settings. See *Sensor Discovery Tool* on page 337 for more information.

When attempting to log in, the password is not accepted.

- Use the Sensor Recovery tool. See *Sensor Discovery Tool* on page 337 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 130 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.

Specifications

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

Gocator 200 Series

The Gocator 200 series consists of the following models:

MODEL	205	210	230	240	250
Measurement Range (MR)[1]	11” 279.4 mm	14” 355.6 mm	8” / 203.2 mm or 10” / 254 mm	8” / 203.2 mm or 10” / 254 mm	8” / 203.2 mm or 10” / 254 mm
Clearance Distance (CD)[1]	20” / 508 mm or 19” / 482 mm	17” 431.8 mm	20” / 508.0 mm or 19” / 482 mm	20” / 508.0 mm or 19” / 482 mm	20” / 508.0 mm or 19” / 482 mm
Scan/Profile Speed (kHz) [2]	4 kHz or 3 kHz	2 kHz	4 kHz or 3 kHz	4 kHz or 3 kHz	4 kHz or 3 kHz
Tracheid Speed (kHz)[2]	N/A	N/A	N/A	N/A[3]	2 kHz or 1.5 kHz
Field of View (FOV)	24” / 609.6 mm	24” / 609.6 mm	24” / 609.6 mm	24” / 609.6 mm	24” / 609.6 mm
Number of Points	N/A	30	76	76	76
X Resolution (at mid-range)	N/A	1.1” / 27.94 mm	0.333” / 8.5 mm	0.333” / 8.5 mm	0.333” / 8.5 mm
Z Resolution	N/A	0.008” / 0.203 mm	0.005” / 0.127 mm	0.005” / 0.127 mm	0.005” / 0.127 mm
XY Resolution (color vision)	0.01” x 0.01” / 0.25 mm x 0.25 mm	N/A	N/A	N/A	N/A
Weight (kg)	1.35	5.5			
Dimensions	For dimensions, see the model-specific sections below.				

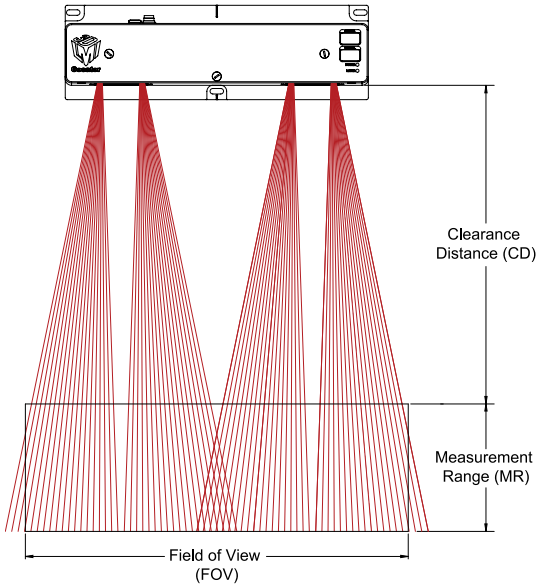
[1] Distance to center of MR is consistent between 8” and 10” configuration.

[2] 4 kHz profile and 2 kHz tracheid at 8” MR and lower.

[3] Tracheid detection available as a software upgrade.

For light bar specifications, see *Light Bars* on page 355.

The following diagram illustrates some of the terms used in the table above (Gocator 230/250 shown, 8” measurement range).



Resolution Z is the maximum variability of height measurements across multiple frames, with 95% confidence.

Resolution X is the distance between data points along the laser line.

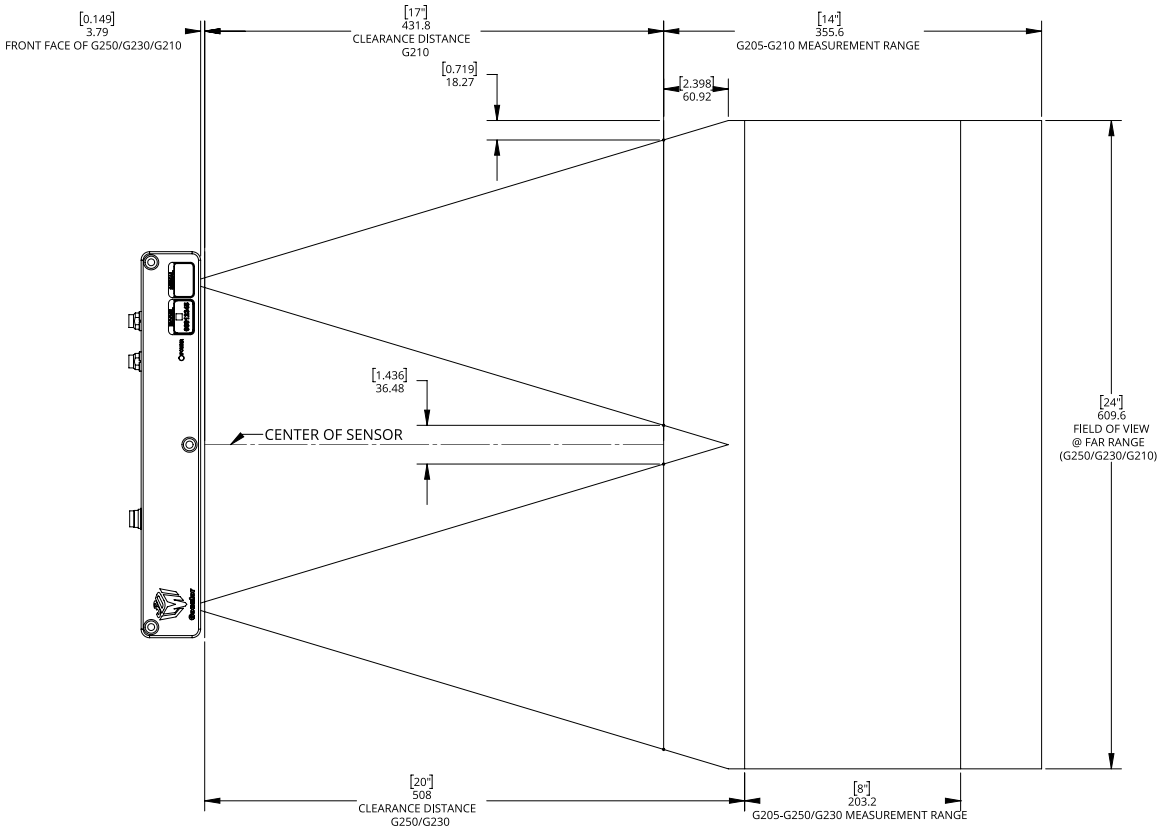
ALL 200 SERIES MODELS	
Interface	Gigabit Ethernet
Inputs	Laser Safety Enable, Trigger
Input Voltage (Power)	+48 VDC Gocator 210 / 230 / 250: 25 Watts; Gocator 205: up to 78 Watts); RIPPLE +/- 10%
Housing	Gasketed aluminum enclosure, IP67
Operating Temp.	0 to 50 °C
Storage Temp.	-30 to 70 °C
Vibration Resistance	10 to 55 Hz, 1.5 mm double amplitude in X, Y and Z directions, 2 hours per direction
Shock Resistance	15 g, half sine wave, 11 ms, positive and negative for X, Y and Z directions
Scanning Software	Browser-based GUI and open source SDK for configuration, real-time 3D visualization, and reference multi-sensor board state machine design. Industrial protocols for integration with PLCs.

Mechanical dimensions, CD/FOV/MR, and the envelope for each sensor model are illustrated on the following pages.

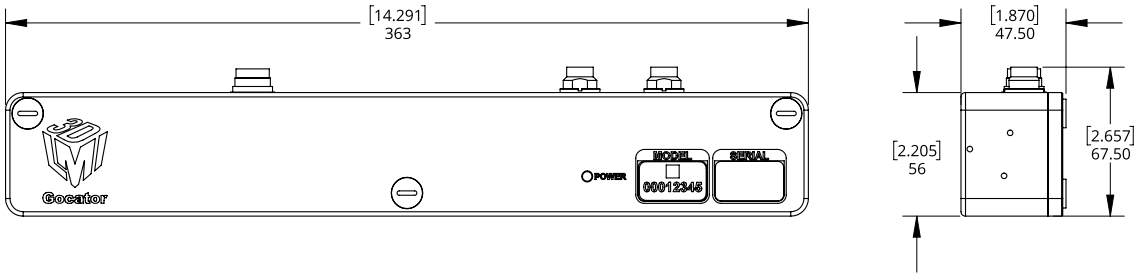
Gocator 205 (Color Vision Module)

For the Gocator 205 envelope, see *Envelope: Scanner + Camera Module + Light Bar* on page 352.

Field of View / Measurement Range

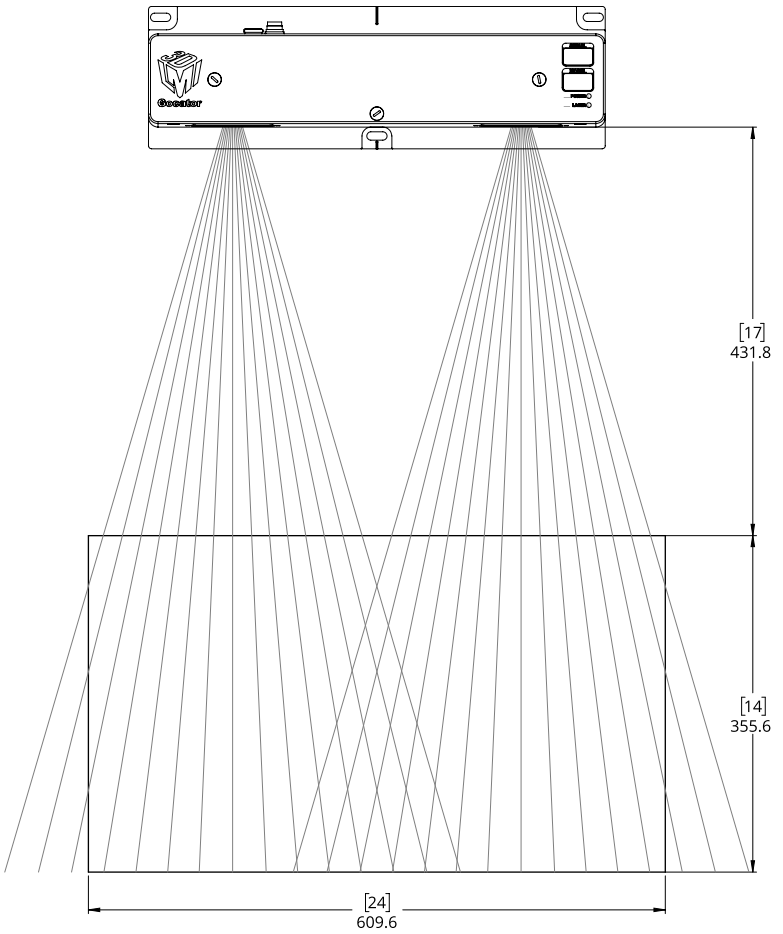


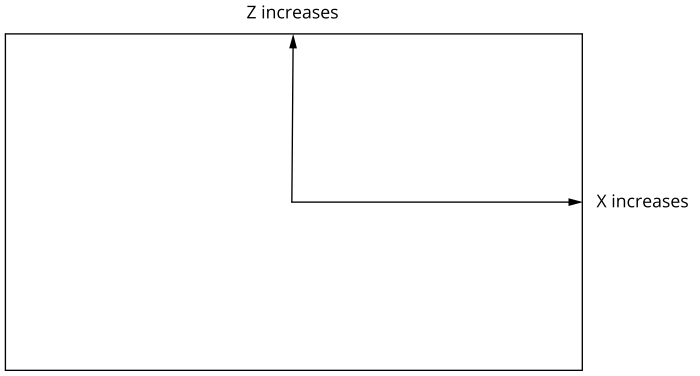
Dimensions



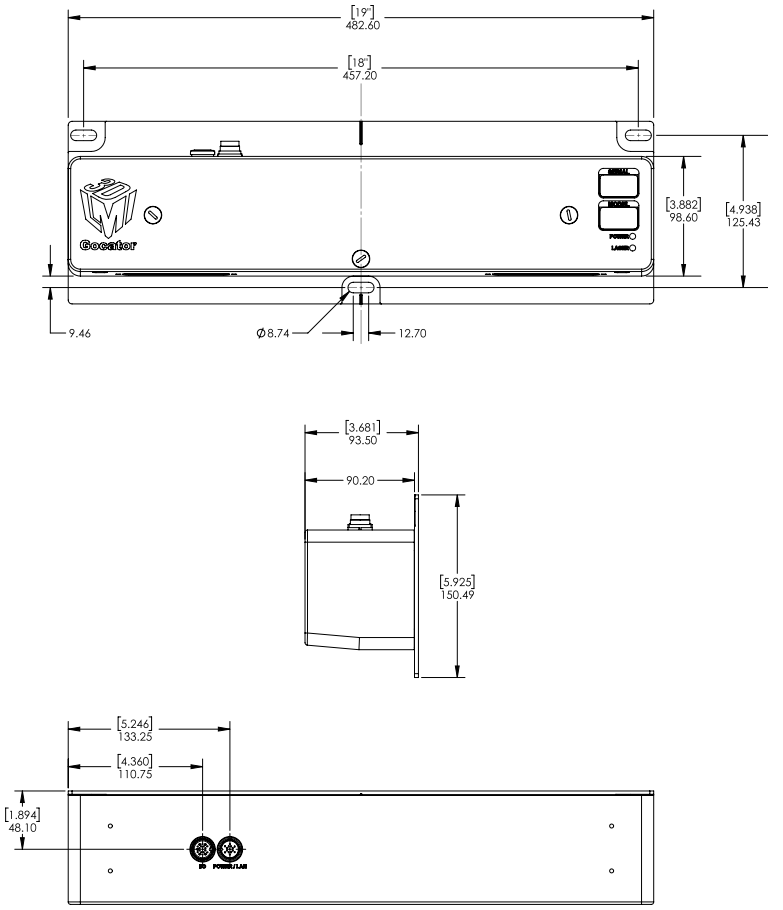
Gocator 210

Field of View / Measurement Range / Coordinate System Orientation

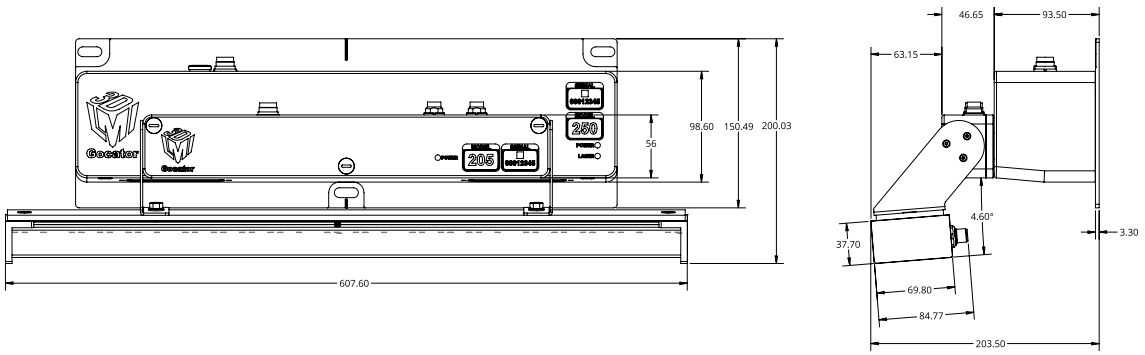




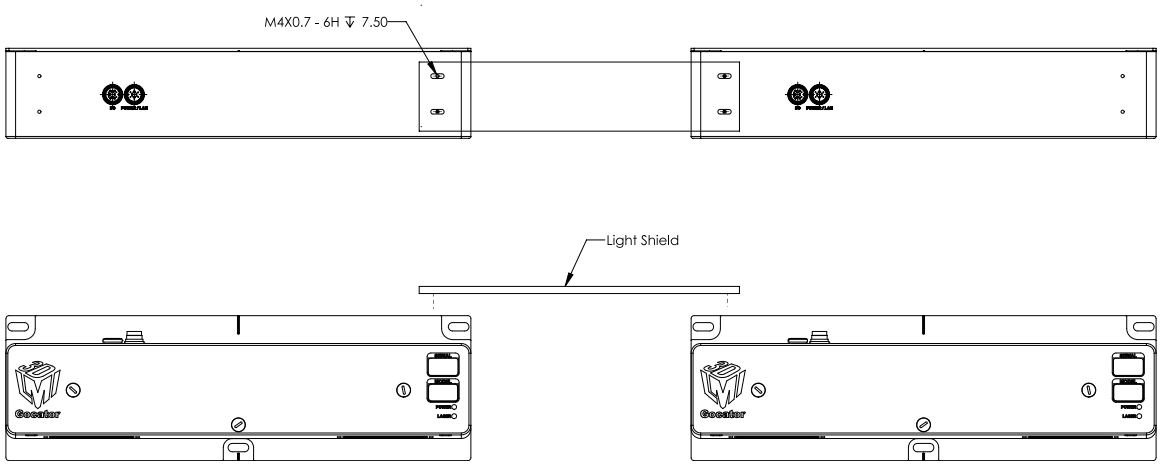
Dimensions



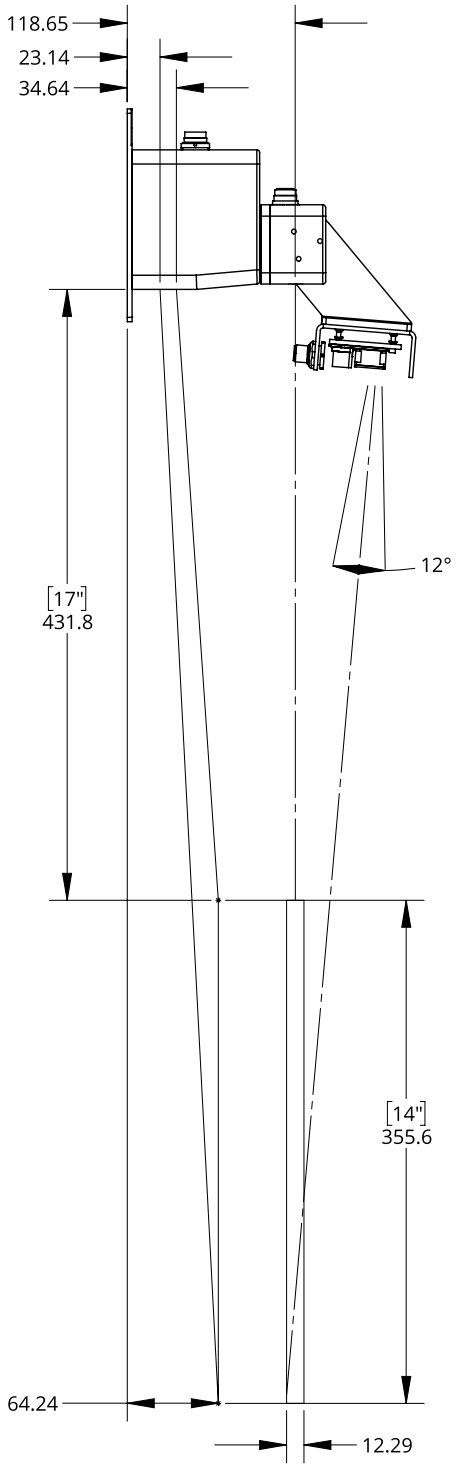
Dimensions: Scanner + Camera Module + Light Bar



Light Shield Hole Specifications



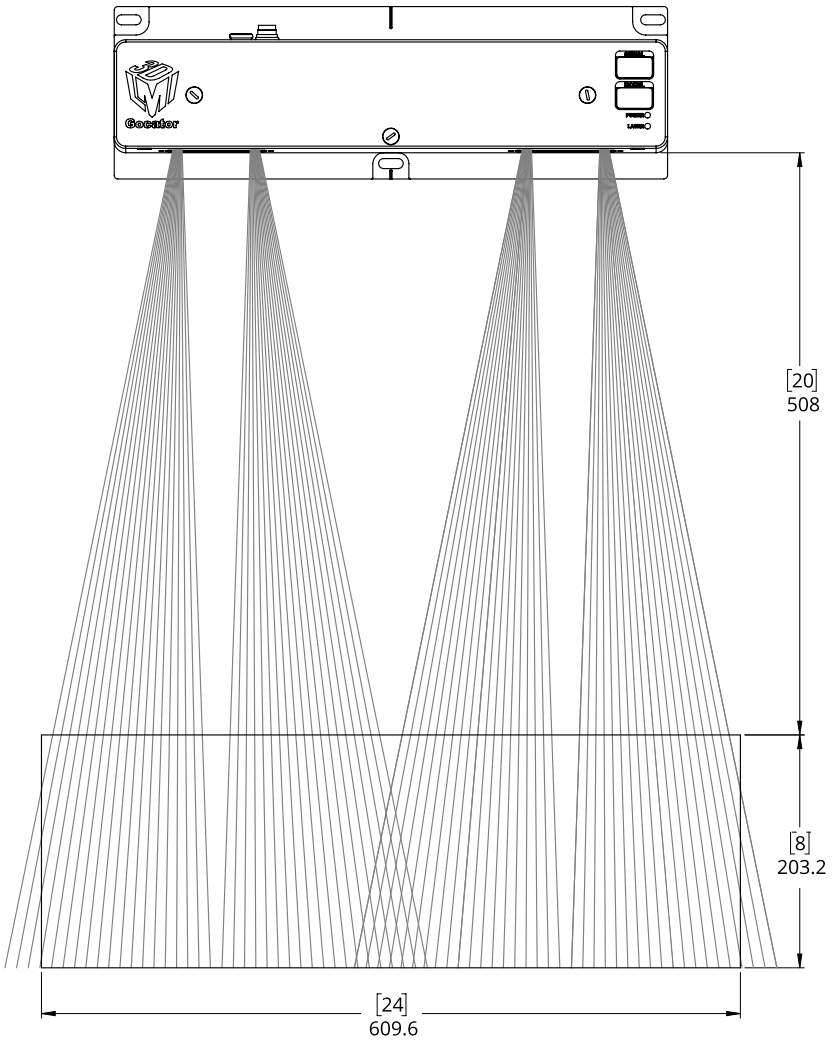
Envelope: Scanner + Camera Module + Light Bar

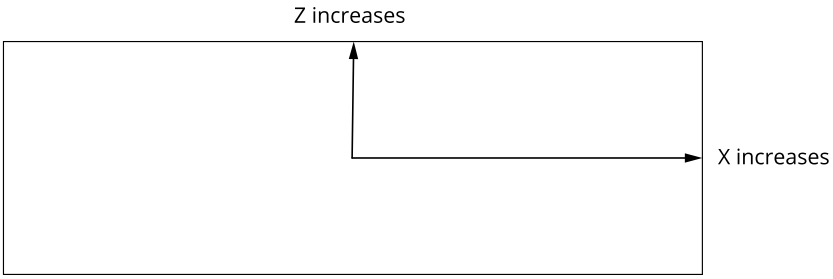


Gocator 230 / 240 / 250

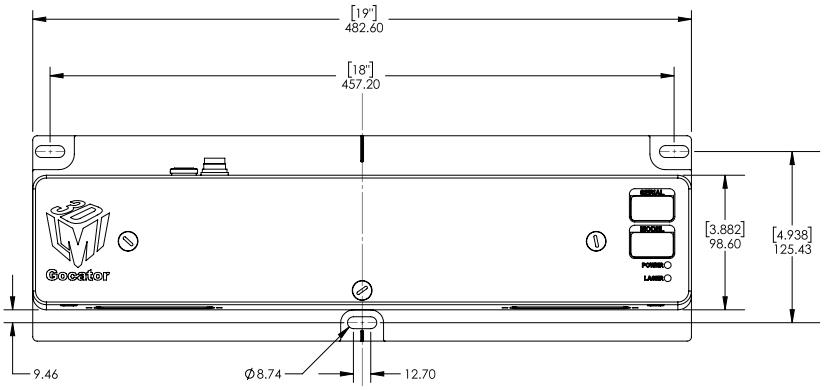
Only the 8" measurement range option is shown below.

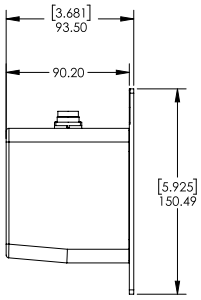
Field of View / Measurement Range / Coordinate System Orientation

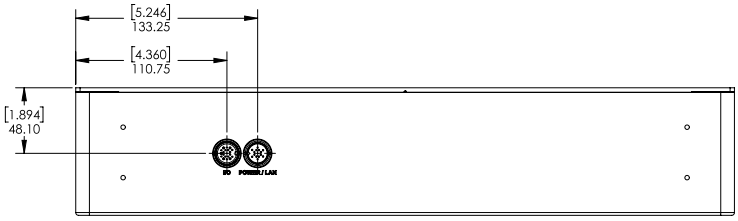




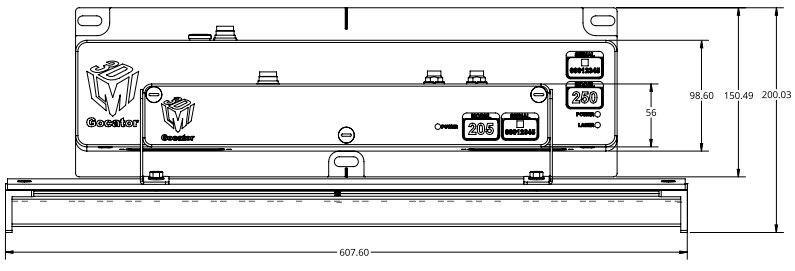
Dimensions: Scanner

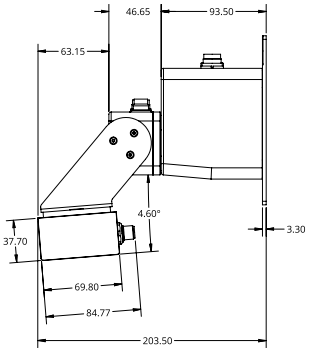




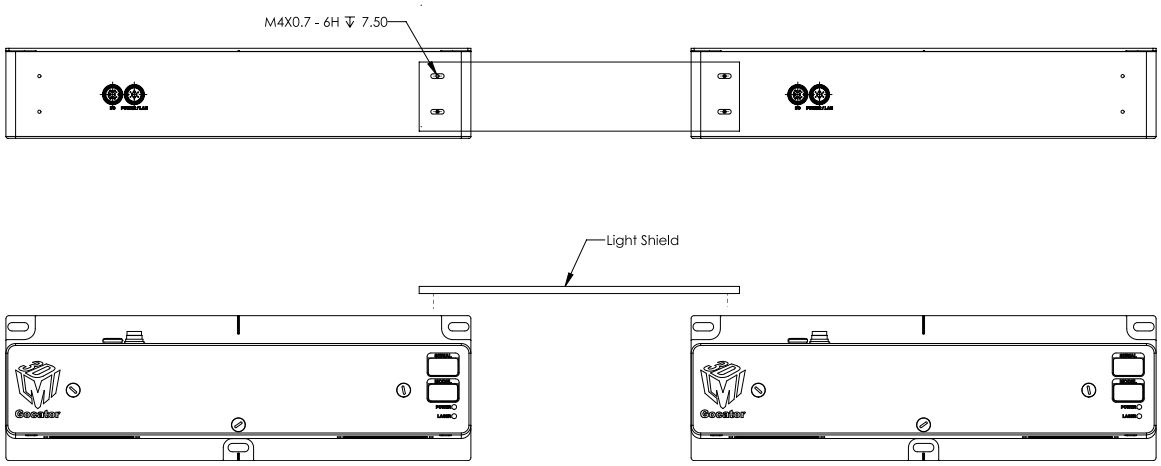


Dimensions: Scanner + Camera Module + Light Bar

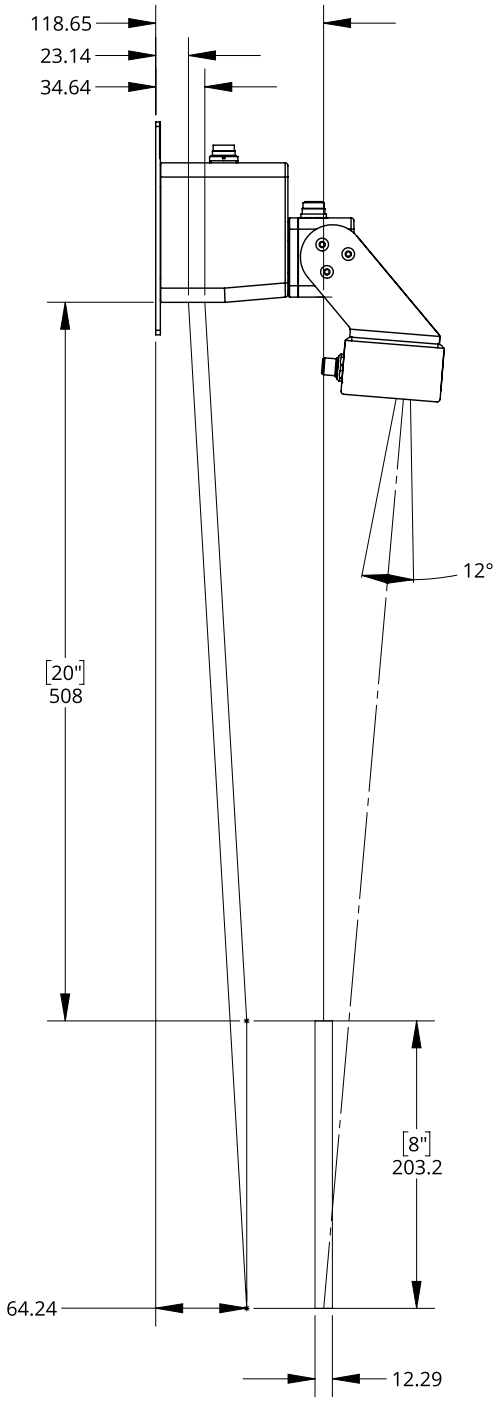




Light Shield Hole Specifications



Envelope: Scanner + Camera Module + Light Bar



Connectors

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

Gocator Power/LAN Connector

The Power/LAN connector is a 14 pin, M16 style connector that provides power input, laser safety input, and Ethernet.

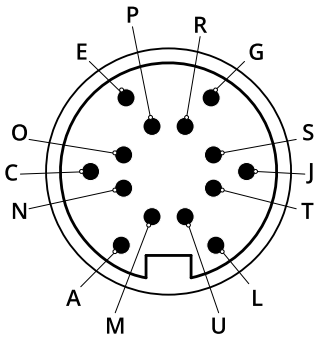
This connector is rated IP67 only when a cable is connected or when a protective cap is used.

Some sensors require a minimum input voltage of 48 VDC. Verify the accepted input voltage for your sensor in the sensor's specifications; for specifications, see *Gocator 200 Series* on page 342.

This section defines the electrical specifications for Power/LAN Connector pins, organized by function.

Gocator Power/LAN Connector Pins

Function	Pin	Lead Color on Standard Cordsets	Lead Color on High Flex Cordsets
GND_24-48V*	L	White/Orange & Black	Orange/Red
GND_24-48V*	L	Orange/Black	Orange/Black
DC_24-48V*	A	White/Green & Black	Green/Red
DC_24-48V*	A	Green/Black	Green/Black
Safety-	G	White/Blue & Black	Blue/Black
Safety+	J	Blue/Black	Blue/Red
Sync+**	E	White/Brown & Black	Brown/Red
Sync-**	C	Brown/Black	Brown/Black
Ethernet MX1+	M	White/Orange	White/Orange
Ethernet MX1-	N	Orange	Orange
Ethernet MX2+	O	White/Green	White/Green
Ethernet MX2-	P	Green	Green
Ethernet MX3-	S	White/Blue	White/Blue
Ethernet MX3+	R	Blue	Blue
Ethernet MX4+	T	White/Brown	White/Brown
Ethernet MX4-	U	Brown	Brown



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

*Both wires must be connected to the power supply. The leads are connected to the same connector pin, and both are required to support the maximum possible current draw.

**The Sync leads are not connected in the open wire versions of the Power/LAN cordsets.

Grounding Shield

The grounding shield should be mounted to the earth ground.

Power

Apply positive voltage to DC_24-48V.

> Some sensors require a minimum input voltage of 48 VDC. Verify the accepted input voltage for your sensor in the sensor's specifications; for specifications, see *Gocator 200 Series* on page 342.

Power requirements

Function	Pins	Min	Max
DC_24-48V	A	24 V (Some models require a minimum of 48 V.)	48 V
GND_24-48VDC	L	0 V	0 V

Safety Input

The Safety_in+ signal should be connected to a voltage source in the range listed below. The Safety_in- signal should be connected to the ground/common of the source supplying the Safety_in+.

> You should not use the Laser Safety input start and stop the lasers. Instead, use the SDKs. For more information, *GoSDK* on page 64.

Laser safety requirements

Function	Pins	Min	Max
Safety_in+	J	24 V	48 V
Safety_in-	G	0 V	0 V

> ⚠ Confirm the wiring of Safety_in- before starting the sensor. Wiring DC_24-48V into Safety_in- may damage the sensor.

Gocator I/O Connector

Gocator 200 sensors do not use the I/O connector.

Light Bars

The following sections provide the specifications of Gocator light bars, which are intended for use with Gocator 200 series multi-point sensors.

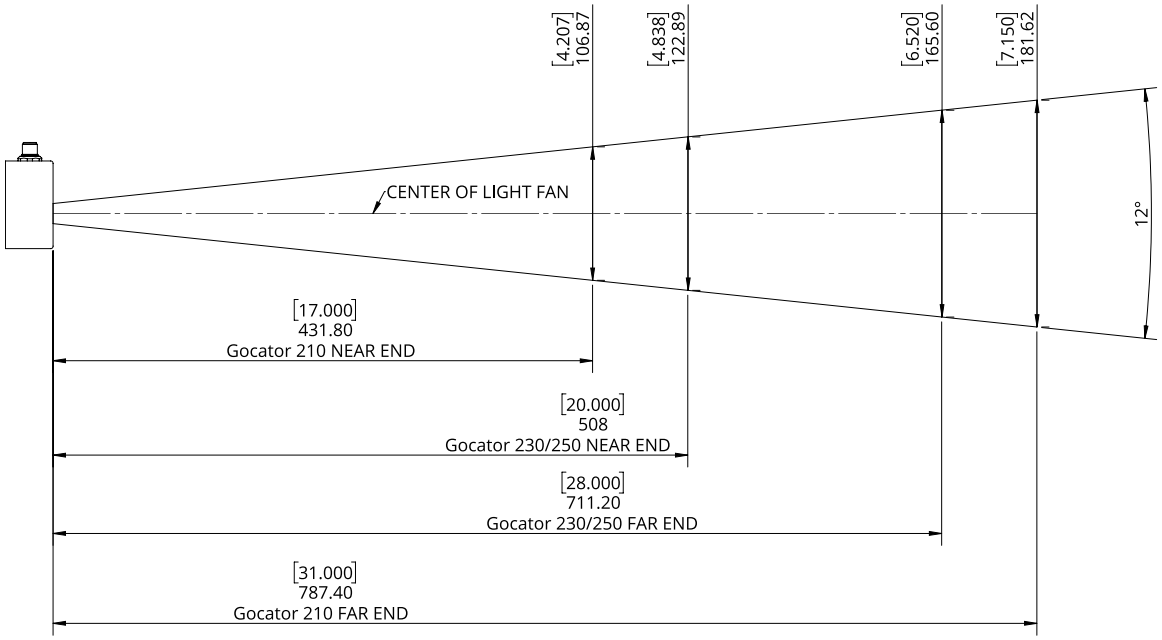
MODEL	LB200	LB210
Clearance Distance (CD) (mm)	17" / 431.8 mm	17" / 431.8 mm
Measurement Range (MR) (mm)	14" / 355.6 mm	14" / 355.6 mm
Field of View (FOV) (mm)	24" / 609.6 mm 12° FWHM	24" / 609.6 mm 30° FWHM

LB210 has a 30-degree fan angle and is recommended for standard on-axis applications, as well as applications with a larger illumination area or with reduced clearance distance.

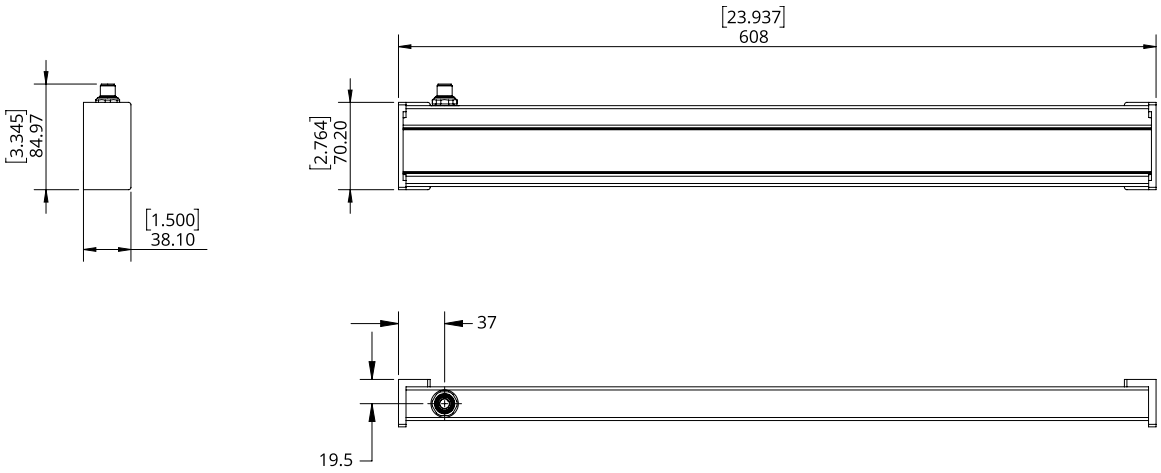
LB200 has a 12-degree fan angle with the same total illumination as LB200 and is recommended for off-axis scanning applications.

LB200 / LB210

Field of View (LB200)



Dimensions (LB200 & LB210)



Master Network Controllers

The following sections provide the specifications of Master network controllers.

Master 810/2410

Master network controllers provide sensor power and laser safety, and broadcast system-wide synchronization information (that is, time, encoder count, encoder index, and digital I/O states) to all devices on a sensor network.

> It is not necessary to power down a sensor's power source such as a Master before unplugging the sensor from the Master. (Sensors can be "hot-swapped.")

The following table summarizes Master 810 and 2410:

Master 810 and 2410	
Input Voltage (Power)	+24-48 VDC (2 Watts)[1]
Total Power	Master 810 / 2410 input power + (sensor input power x number of sensors)
Capacity	Master 810: up to 8 sensors Master 2410: up to 24 sensors
I/O	4 digital inputs[2] Single-Ended Active LOW: 0 to +0.8 VDC Single-Ended Active HIGH: +3.3 to +24 VDC Differential LOW: 0.8 to -24 VDC Differential HIGH: +3.3 to +24 VDC 10-pin Phoenix For more information, see *Electrical Specifications* on page 362.
Encoder	Differential (5 VDC, 12 VDC) Single-ended (5 VDC, 12 VDC)[3] For more information, see *Electrical Specifications* on page 362.
LED Indicators	Safety, power, encoder, input. For more information, see *LED Indicators* on the next page.
Cable	Dual CAT5e cable for power / safety / synchronization / data
Weight (kg)	Master 810: 0.6 Master 2410: 0.9
Storage Temperature	-30 to 70 °C
Operation Temperature	0 to 50 °C

Notes

1. Refer to sensor datasheets for additional power required by sensors.
2. Gocator only supports one digital input.
3. Supports open collector, pull-up resistor, line driver, push-pull, and TTL.

The following table describes the meanings of the encoder and sensor port LED indicators:

LED Indicators

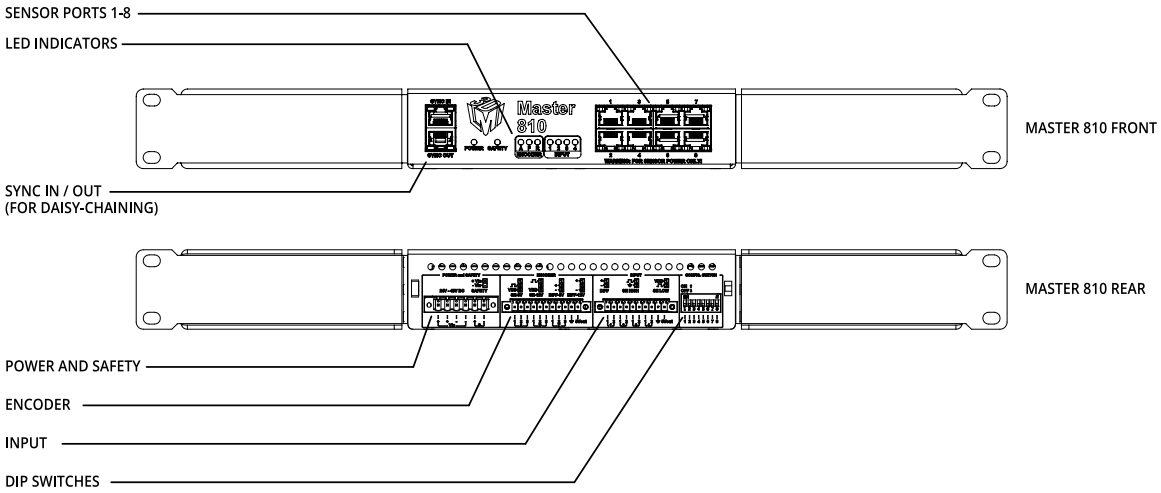
Indicator	Description
Power	Device is on.
Safety	Indicates the status of the Laser Safety circuitry. The “On” state indicates that all sensor light sources are active.
Encoder A	Reserved
Encoder F	**On continuously**: Forward motion with no indexing is detected. **Blinking**: Forward motion with indexing is detected.
Encoder R	**On continuously**: Reverse motion with no indexing is detected. **Blinking**: Reverse motion with indexing is detected.
Input 1-4	Digital input ports 1-4 active.
SYNC IN and SYNC OUT Ports (Green and Orange LEDs)	Reserved.
Sensor Port Green LED	Indicates that a sensor is connected to the port and is powered up.
Sensor Port Orange LED	Not used.

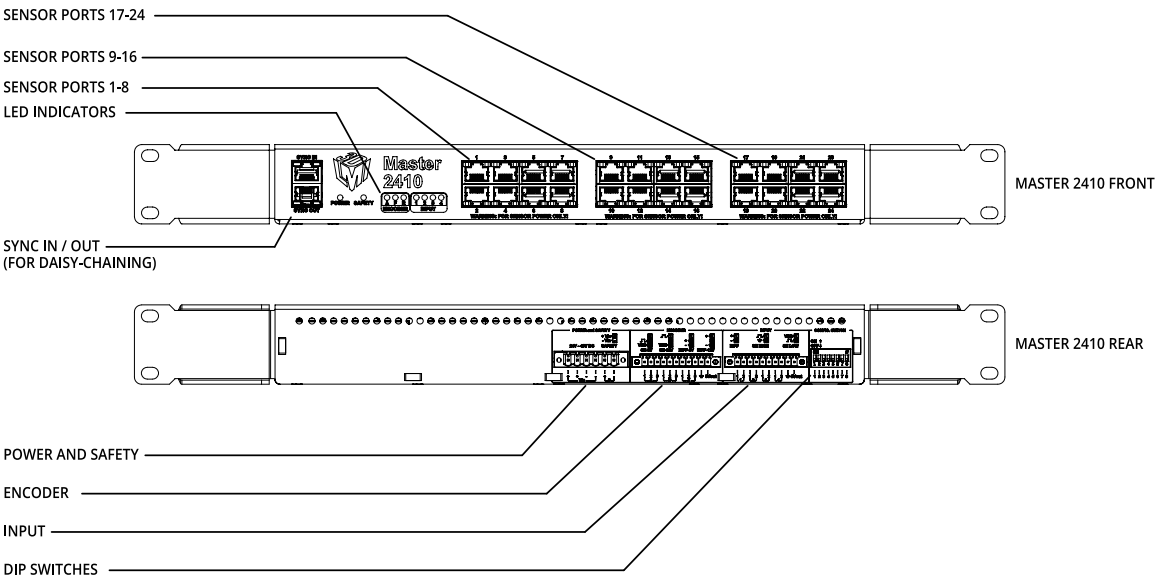
Master 810 and 2410 can be mounted to DIN rails using the provided adapters (for more information, see *Installing DIN Rail Clips: Master 810 or 2410* on page 38). The units are also provided with removable adapters for 1U rack mounting; the mounting holes for this option are compatible with older Master models (400/800/1200/2400).

> The Phoenix connectors on Master 400/800/1200/2400 are not compatible with the connectors on Master 810/2410. For this reason, if you are switching models in your network, you must rewire the connections to the Master.

Master 2410 can currently be used with encoders with a maximum quadrature frequency of 300 kHz.

Master 810 can be configured to work with a maximum encoder quadrature frequency of 6.5 MHz. For more information, see *Configuring Master 810* on page 39.





Power and Safety

Power and Safety (6 pin connector)

Function	Pin
Power In+	1
Power In+	2
Power In-	3
Power In-	4
Safety Control+	5
Safety Control–	6

The 6-pin Power and Safety connector's specifications are as follows:

- CONNECTOR, 6 Position Terminal Block Plug, Female Sockets 0.200" (5.08mm) 180° Free Hanging (In-Line)
- Manufacturer PN: 1912223
- Supplier Part Number 277-11017-ND
- Manufacturer: Phoenix Contact

> The power supply must be isolated from AC ground. This means that AC ground and DC ground are not connected.

Encoder

Encoder (11 pin connector)

Function	Pin
Encoder_A_Pin_1	1
Encoder_A_Pin_2	2

Function	Pin
Encoder_A_Pin_3	3
Encoder_B_Pin_1	4
Encoder_B_Pin_2	5
Encoder_B_Pin_3	6
Encoder_Z_Pin_1	7
Encoder_Z_Pin_2	8
Encoder_Z_Pin_3	9
GND (output for powering external devices)	10
+5VDC (output for powering external devices)	11

For Encoder connection wiring options, see *Encoder* on page 363.

The 11-pin Encoder connector's specifications are as follows:

- CONNECTOR, 11 Position Terminal Block Plug, Female Sockets 0.138" (3.50mm) 180° Free Hanging (In-Line)
- Manufacturer PN: 1847217
- Supplier Part Number 277-8897-ND
- Manufacturer: Phoenix Contact

Input

On earlier revisions of Master 810 and Master 2410, the inputs are labeled 0-3.

Input (10 pin connector)

Function	Pin
Input 1 Pin 1	1
Input 1 Pin 2	2
Input 2 Pin 1*	3
Input 2 Pin 2*	4
Input 3 Pin 1*	5
Input 3 Pin 2*	6
Input 4 Pin 1*	7
Input 4 Pin 2*	8
GND (output for powering other devices)	9
+5VDC (output for powering other devices)	10

*Using inputs 2, 3, and 4 is only possible through GoSDK (see TriggerExternalInputTest in GoSensorFix.cs). When not connected to a Master, a system reverts to using only digital input 1 for triggering

For Input connection wiring options, see *Input* on page 365.

The following are the 10-pin Input connector's specifications:

- CONNECTOR, 10 Position Terminal Block Plug, Female Sockets 0.138" (3.50mm) 180° Free Hanging (In-Line)
- Manufacturer PN: 1847204
- Supplier Part Number 277-6350-ND
- Manufacturer: Phoenix Contact

The Input connector does not need to be wired up for proper operation.

Electrical Specifications

Some sensors require a minimum input voltage of 48 VDC. Verify the accepted input voltage for your sensor in the sensor's specifications; for specifications, see *Gocator 200 Series* on page 342.

Electrical Specifications

Specification	Value
Power Supply Voltage	+24 VDC to +48 VDC
Power Supply Current (Max.)*	Master 810: 9 A Master 2410: 25 A * Fully loaded with 1 A per sensor port.
Power Draw (Min.)	Master 810: 1.7 W Master 2410: 4.8 W
Encoder Signal Voltage	Single-Ended Active LOW: 0 to +0.8 VDC Single-Ended Active HIGH: +3.3 to +24 VDC Differential LOW: 0.8 to -24 VDC Differential HIGH: +3.3 to +24 VDC For more information, see *Encoder* on the next page.
Digital Input Voltage Range	Single-Ended Active LOW: 0 to +0.8 VDC Single-Ended Active HIGH: +3.3 to +24 VDC Differential LOW: 0.8 to -24 VDC Differential HIGH: +3.3 to +24 VDC For more information, see *Input* on page 365. If the input voltage is above 24 V, use an external resistor, using the following formula: R = [(Vin - 1.2V) / 10mA] - 680

When using a Master hub, the chassis must be well grounded.

	The power supply must be isolated from AC ground. This means that AC ground and DC ground are not connected.

	24 VDC power supply is only supported if all connected sensors support an input voltage of 24 VDC.

	The Power Draw specification is based on a Master with no sensors attached. Every sensor has its own power requirements that need to be considered when calculating total system power requirements..

Encoder

Master 810 and 2410 support the following types of encoder signals: Single-Ended (5 to 12 VDC, 12 to 24 VDC) and Differential (5 to 12 VDC, 12 to 24 VDC).

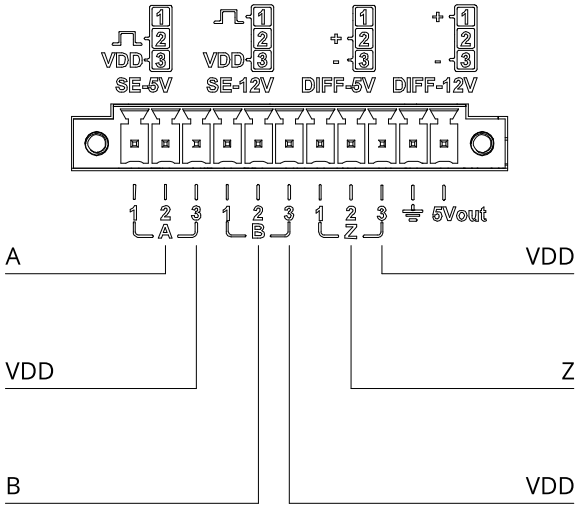
For 5 to 12 VDC operation, pins 2 and 3 of each channel are used.

	The 5-volt encoder input supports up to 12 volts for compatibility with earlier Master network controllers. However, we strongly recommend connecting 12-volt output encoders to the appropriate 12-volt input to attain maximum tolerance.

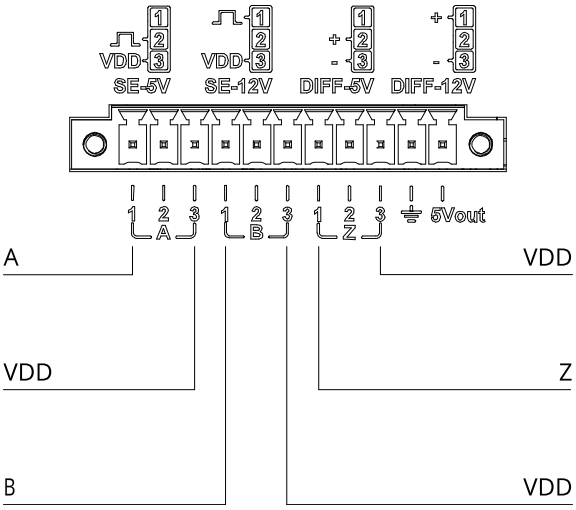
For 12 to 24 VDC operation, pins 1 and 3 of each channel are used.

	Although Master 810 and 2410 are currently labeled as only supporting up to 12 VDC, they can support up to 24 VDC.

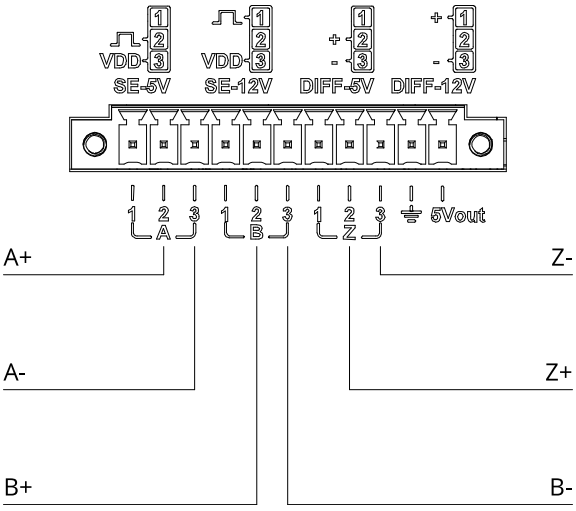
To determine how to wire a Master to an encoder, see the illustrations below.



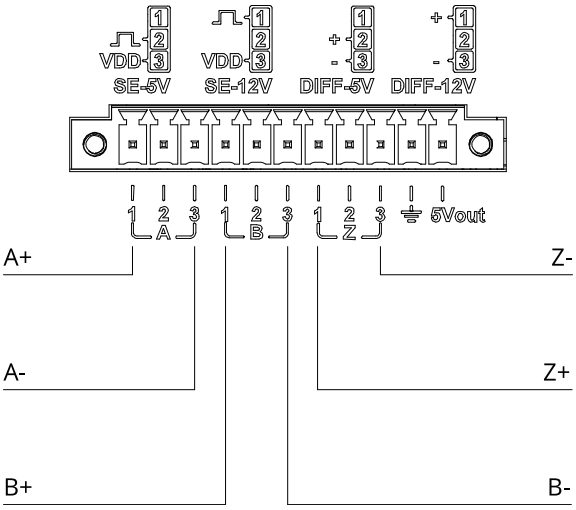
Single-Ended 5 VDC



Single-Ended 12 VDC



Differential 5 VDC



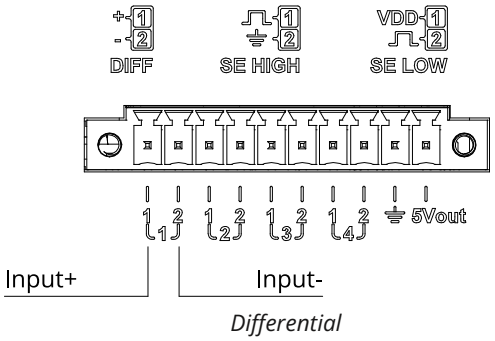
Differential 12 VDC

Input

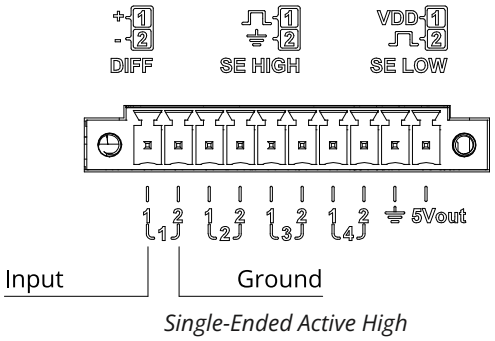
Master 810 and 2410 support the following types of input: Differential, Single-Ended High, and Single-Ended Low.

> Currently, Gocator only supports Input 0.

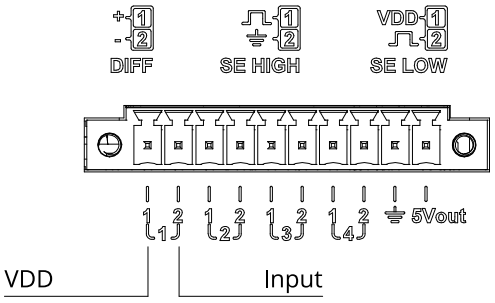
For digital input voltage ranges, see the table below.



Differential



Single-Ended Active High



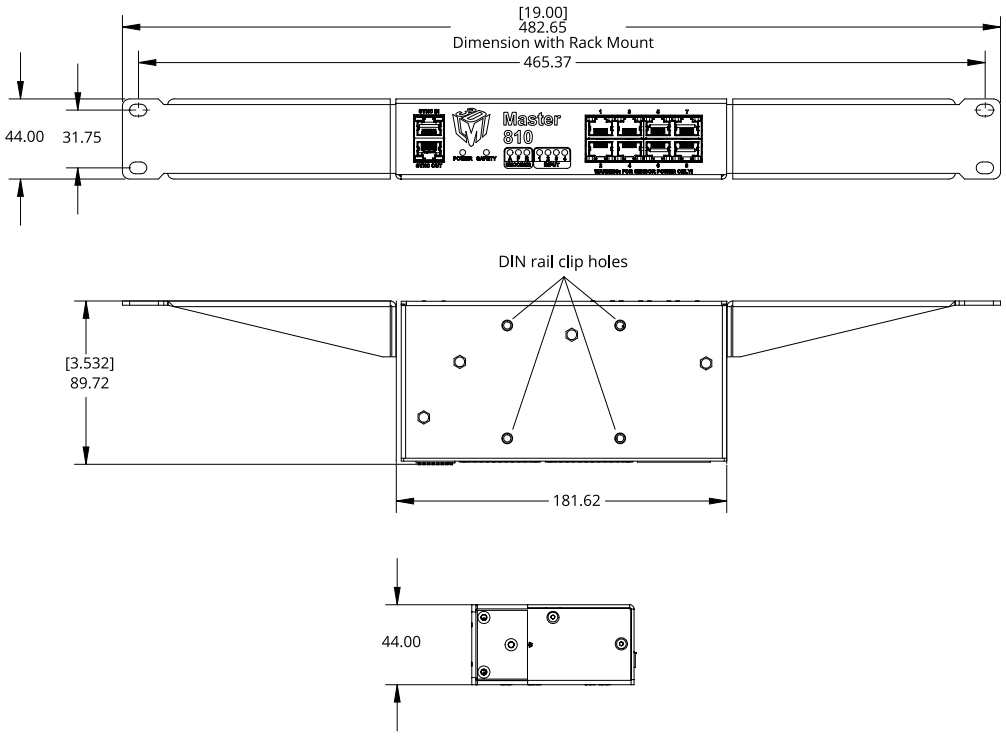
Single-Ended Active Low

Digital Input Voltage Ranges

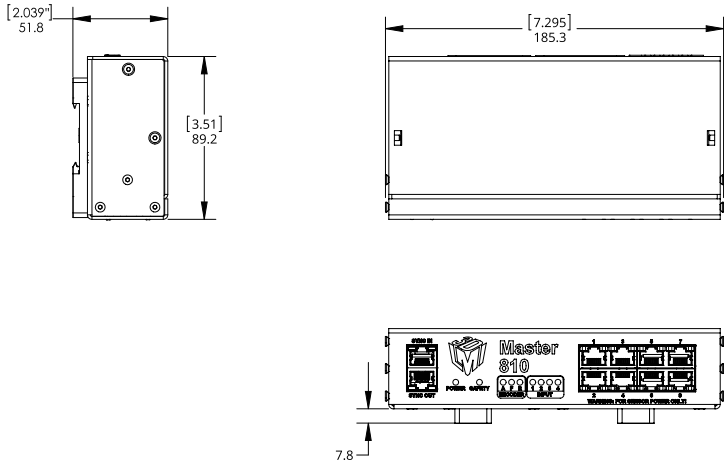
	Input Status	Min (VDC)	Max (VDC)
Single-ended Active High	Off	0	+0.8
	On	+3.3	+24
Single-ended Active Low	Off	(V_{DD} - 0.8)	V_{DD}
	On	0	(V_{DD} - 3.3)
Differential	Off	-24	+0.8
	On	+3.3	+24

Master 810 Dimensions

With 1U rack mount brackets:



With DIN rail mount clips:



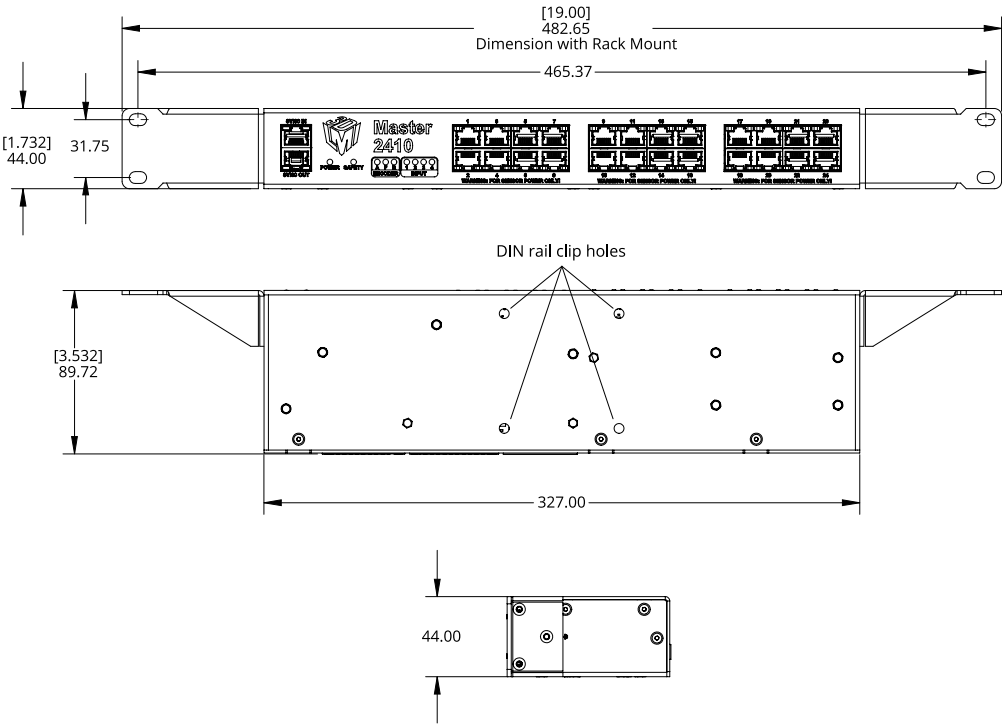
Older revisions of Master 810 and 2410 network controllers use a different configuration for the DIN rail clip holes.

For information on installing DIN rail clips, see *Installing DIN Rail Clips: Master 810 or 2410* on page 38.

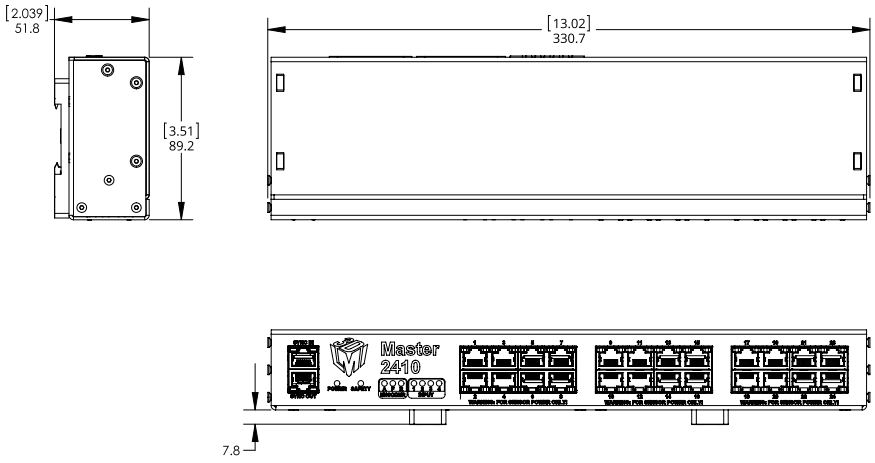
The CAD model of the DIN rail clip is available at https://www.winford.com/products/cad/dinm12-rc.igs.

Master 2410 Dimensions

With 1U rack mount brackets:



With DIN rail mount clips:



Older revisions of Master 810 and 2410 network controllers use a different configuration for the DIN rail clip holes.

For information on installing DIN rail clips, see *Installing DIN Rail Clips: Master 810 or 2410* on page 38.

The CAD model of the DIN rail clip is available at https://www.winford.com/products/cad/dinm12-rc.igs.

Accessories

Masters

Description	Part Number
Master 810 - for networking up to 8 sensors	301114
Master 2410 - for networking up to 24 sensors	301115

Custom cordset lengths and connector orientations are available upon request. Prices depend on length and orientation requested. For standard cordsets, the maximum cable length is 60 m.

Cordsets - Straight

Description	Part Number
0.5m Light Bar to Gocator 205 cordset	301160
2m Power and Ethernet cordset, 1x open wire end, 1x RJ45 end	301176-2m
5m Power and Ethernet cordset, 1x open wire end, 1x RJ45 end	301176-5m
10m Power and Ethernet cordset, 1x open wire end, 1x RJ45 end	301176-10m
2m Power and Ethernet to Master cordset, 2x RJ45 ends	301165-2m
5m Power and Ethernet to Master cordset, 2x RJ45 ends	301165-5m
10m Power and Ethernet to Master cordset, 2x RJ45 ends	301165-10m

Cordsets - 90-degree

Description	Part Number
2m Power and Ethernet cordset, 90-deg, 1x open wire end, 1x RJ45 end	301171-2m
5m Power and Ethernet cordset, 90-deg, 1x open wire end, 1x RJ45 end	301171-5m
10m Power and Ethernet cordset, 90-deg, 1x open wire end, 1x RJ45 end	301171-10m
2m Power and Ethernet to Master cordset, 90-deg, 2x RJ45 ends	301173-2m
5m Power and Ethernet to Master cordset, 90-deg, 2x RJ45 ends	301173-5m
10m Power and Ethernet to Master cordset, 90-deg, 2x RJ45 ends	301173-10m

Light Bars

Description	Part Number
LB200, Light Bar, 12 degrees	30803

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

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

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

For more information on safety and laser classifications, please contact:

U.S. Food and Drug Administration
Center for Devices and Radiological Health
WO66-G609
10903 New Hampshire Avenue
Silver Spring MD 20993-0002
USA

Contact

Americas	EMEAR	ASIA PACIFIC
LMI Technologies (Head Office)	LMI Technologies GmbH	LMI (Shanghai) Trading Co., Ltd.
Burnaby, Canada	Berlin, Germany	Shanghai, China
+1 604 636 1011	+49 (0)3328 9360 0	+86 21 5441 0711

LMI Technologies has sales offices and distributors worldwide. All contact information is listed at lmi3D.com/contact/locations.