VsxProtocolDriver

General

The driver VsxProtocolDriver provides full access to the input and output data of the sensor and facilitates integration in a C#-based programming environment. The driver connects to the sensor and handles communication in accordance with the communication protocol. The user can access functions for setting parameters on the sensor, retrieving parameter values from the sensor, and saving and loading entire parameter sets both locally and on the sensor. The user can also receive sensor images. Each function also contains an error object from which information can be obtained in the event of an error in the function.

The driver is implemented in C# and requires .NET 5.0 or higher.

The functions of the driver can be used **synchronously** or **asynchronously**. For this, the required instance must be created using the Init function of the respective classes. The VsxProtocolDriver class provides the asynchronous driver. The VsxProtocolDriverSync class provides the synchronous interface.

For clarity, this manual only describes the most important functions and variables. The SDK contains additional functions that are used for other Pepperl+Fuchs vision sensors. These functions are described in the corresponding product manual. There are several declaration options for some functions. In the following, the preferred functions are marked in bold.

Note

Integrating the NuGet

In order to use the SDK, the NuGet must be integrated. This can be done in Visual Studio using the NuGet package manager, for example. The SDK can be found on the product page of the corresponding sensor from Pepperl+Fuchs in the software folder. The NuGet is located in the ZIP file stored in this folder under the project folder ext.

Synchronous and Asynchronous Functions

The auxiliary classes used in the parameters are described below.

Static Functions

UDP Broadcast

Applicable to:
- SmartRunner
- SmartRunner 3-D

The static function returns a list of the Vsx devices that are found on the network via a UDP broadcast.

Asynchronous Function

```
public static Task<(bool Succ, List<Device> DeviceList, Error ErrorDesc)> UdpDeviceList()
```

Synchronous Function

```
public static (bool Succ, List<Device> DeviceList, Error ErrorDesc) UdpDeviceList()
```

Possible error IDs: VSX_DRIVER_CONNECTION_ERROR

Configure Network Settings via UDP

Applicable to:
- SmartRunner
- SmartRunner 3-D

Network settings on the device are changed via UDP.

Asynchronous Function

```
public static Task<(bool Succ, Error ErrorDesc)> SetNetworkSettingsViaUdp(string macAddress,
string ipAddress, string networkMask, string gateway)
```

Synchronous Function

```
public static (bool Succ, Error ErrorDesc) SetNetworkSettingsViaUdp(string macAddress, string
ipAddress, string networkMask, string gateway)
```

Possible error IDs: None

TCP/IP Connection

Applicable to:
- SmartRunner
- SmartRunner 3-D

Initializes a new driver instance that can be used to communicate with the device via TCP/IP. While the IP address must be specified, the default VSXPORT = 50005 can be used.

Asynchronous Function

```
public static VsxProtocolDriver Init(string ipAddress, int port = VSXPORT, string pluginName = "")
```

Synchronous Function

```
public static VsxProtocolDriverSync Init(string ipAddress, int port = VSXPORT, string pluginName = "")
```

Save an IVsxMessage

Applicable to:
- SmartRunner
- SmartRunner 3-D

An IVsxMessage is a general interface for data exchanged between the PC and the sensor.

Asynchronous Function

```
public static (bool Succ, Error ErrorDesc) SaveData(string filename, IVsxMessage message)
```

Synchronous Function

```
public static (bool Succ, Error ErrorDesc) SaveData(string filename, IVsxMessage message)
```

Possible error IDs: VSX_DRIVER_DATA_ERROR, VSX_DRIVER_INVALID_DATA_ERROR, VSX_DRIVER_SAVE_FILE_ERROR

Save a Point Cloud File

Applicable to:
- SmartRunner 3-D

Saves a point cloud file consisting of planes x, y, and z under the specified file name.

Asynchronous Function

```
public static (bool Succ, Error ErrorDesc) Save3DPointCloudData(string filename, VsxImageData2Message x, VsxImageData2Message y, VsxImageData2Message z)
```

Synchronous Function

```
public static (bool Succ, Error ErrorDesc) Save3DPointCloudData(string filename, VsxImageData2Message x, VsxImageData2Message y, VsxImageData2Message z)
```

Possible error IDs: VSX_DRIVER_DATA_ERROR, VSX_DRIVER_SAVE_FILE_ERROR

Non-Static Functions

Establish a Connection to the Device

Applicable to:
- SmartRunner
- SmartRunner 3-D

Establishes a connection to the device using the parameters set via Init. CONNECTION_TIMEOUT_MS = 1000 can be used as the timeout for opening the connection. A connection to the device must be established in order to use non-static features.

Asynchronous Function

```
public Task<(bool Succ, Error ErrorDesc)> Connect(int timeout = CONNECTION_TIMEOUT_MS)
```

Synchronous Function

```
public (bool Succ, Error ErrorDesc) Connect(int timeout =
VsxProtocolDriver.CONNECTION_TIMEOUT_MS)
```

Possible error IDs: VSX_DRIVER_CONNECTION_ERROR

Disconnect from the Device

Applicable to:
- SmartRunner
- SmartRunner 3-D

Disconnects from the device.

Asynchronous Function

```
public Task<(bool Succ, Error ErrorDesc)> Disconnect()
```

Synchronous Function

```
public (bool Succ, Error ErrorDesc) Disconnect()
```

Possible error IDs: None

Re-establish the Connection with Transferred Parameters

Applicable to:
- SmartRunner
- SmartRunner 3-D

Disconnects the device and re-establishes the connection using the transferred parameters. These functions can be used if the connection parameters need to be changed at runtime.

Asynchronous Function

```
public Task<(bool Succ, Error ErrorDesc)> ReConnect(string ipAddress, int port = VSXPORT)
```

Synchronous Function

```
public (bool Succ, Error ErrorDesc) ReConnect(string ipAddress, int port = VSXPORT)
```

Asynchronous Function

```
public Task<(bool Succ, Error ErrorDesc)> ReConnect(string serialPort, int baudrate,
TheSensor.ConnectionType connectionType)
```

Synchronous Function

```
public (bool Succ, Error ErrorDesc) ReConnect(string serialPort, int baudrate,
TheSensor.ConnectionType connectionType)
```

Possible error IDs: VSX_DRIVER_CONNECTION_ERROR

Device Information

Applicable to:
- SmartRunner
- SmartRunner 3-D

Transmits information about the device (such as MAC address, etc.).

Asynchronous Function

```
public Task<(bool Succ, Device CurrentDevice, Error ErrorDesc)> GetCurrentDeviceInformation()
```

Synchronous Function

```
public (bool Succ, Device CurrentDevice, Error ErrorDesc) GetCurrentDeviceInformation()
```

Possible error IDs: VSX_DRIVER_CONNECTION_ERROR

Feature List

Applicable to:
- SmartRunner
- SmartRunner 3-D

Transmits the list of features available on the device.

Asynchronous Function

```
public Task<(bool Succ, float XmlVersion, Hashtable FeatureList, Error ErrorDesc)>
GetFeatureList()
```

Synchronous Function

```
public (bool Succ, float XmlVersion, Hashtable FeatureList, Error ErrorDesc)
GetFeatureList()
```

Possible error IDs: VSX_DRIVER_CONNECTION_ERROR

Parameter List incl. Detailed Information and Current Values

Applicable to:
- SmartRunner
- SmartRunner 3-D

Transmits a list of all parameters available on the device, including detailed information and their current values.

Asynchronous Function

```
public Task<(bool Succ, List<Parameter> ParameterList, Error ErrorDesc)> GetParameterList()
```

Synchronous Function

```
public (bool Succ, List<Parameter> ParameterList, Error ErrorDesc) GetParameterList()
```

Possible error IDs: VSX_DRIVER_GENERAL_ERROR, VSX_DRIVER_DATA_ERROR, VSX_DRIVER_CONNECTION_ERROR

Transmit the Value of a Single Device Parameter

Applicable to:
- SmartRunner
- SmartRunner 3-D

Transmits the value of a single device parameter.

Asynchronous Function

```
public Task<(bool Succ, object parameterValue, Error ErrorDesc)>
GetSingleParameterValue(Parameter parameter)

public Task<(bool Succ, object parameterValue, Error ErrorDesc)>
GetSingleParameterValue(string parameterId)

public Task<(bool Succ, object parameterValue, Error ErrorDesc)>
GetSingleParameterValue(ushort settingsVersion, ushort configVersion, string configId, string
parameterId)
```

Synchronous Function

```
public (bool Succ, object parameterValue, Error ErrorDesc) GetSingleParameterValue(Parameter parameter)

public (bool Succ, object parameterValue, Error ErrorDesc) GetSingleParameterValue(string parameterId)

public (bool Succ, object parameterValue, Error ErrorDesc) GetSingleParameterValue(ushort settingsVersion, ushort configVersion, string configId, string parameterId)
```

Possible error IDs: VSX_DRIVER_CONNECTION_ERROR, VSX_DRIVER_DATA_ERROR

Set the Value of a Single Device Parameter

Applicable to:
- SmartRunner
- SmartRunner 3-D

Sets the value of a single parameter on the device. The parameter is defined by transferring the function parameters in the same way as for the function *GetSingleParameterValue*. In addition, the desired value is transferred.

Asynchronous Function

```
public Task<(bool Succ, List<Parameter> DependendParameters, Error ErrorDesc)>
SetSingleParameterValue(Parameter parameter, object value)

public Task<(bool Succ, List<Parameter> DependendParameters, Error ErrorDesc)>
SetSingleParameterValue(string parameterId, object value)

public Task<(bool Succ, List<Parameter> DependendParameters, Error ErrorDesc)>
SetSingleParameterValue(ushort settingsVersion, ushort configVersion, string configId, string parameterId, object value)
```

Synchronous Function

```
public (bool Succ, List<Parameter> DependendParameters, Error ErrorDesc)
SetSingleParameterValue(Parameter parameter, object value)

public (bool Succ, List<Parameter> DependendParameters, Error ErrorDesc)
SetSingleParameterValue(string parameterId, object value)

public (bool Succ, List<Parameter> DependendParameters, Error ErrorDesc)
SetSingleParameterValue(ushort settingsVersion, ushort configVersion, string configId, string parameterId, object value)
```

Possible error IDs: VSX_DRIVER_CONNECTION_ERROR, VSX_DRIVER_DATA_ERROR

Change Network Settings

Applicable to:
- SmartRunner
- SmartRunner 3-D

Changes the network settings on the device. The connection to the device is terminated and must be re-established using the "Connect" function (see "Establish a Connection to the Device" on page 3).

Asynchronous Function

```
public Task<(bool Succ, Error ErrorDesc)> SetNetworkSettings(string ipAddress,
string networkMask, string gateway)
```

Synchronous Function

```
public (bool Succ, Error ErrorDesc) SetNetworkSettings(string ipAddress, string networkMask, string gateway)
```

Possible error IDs: VSX_DRIVER_CONNECTION_ERROR

Send the Firmware File to the Device

Applicable to:
- SmartRunner
- SmartRunner 3-D

Sends the firmware file under the specified path and file name to the device. The current status can be read out via *FirmwareStateChannelReader* while the update is in progress.

Asynchronous Function

```
public Task<(bool Succ, Error ErrorDesc)> SendFirmware(string fileName)
```

Synchronous Function

```
public (bool Succ, Error ErrorDesc) SendFirmware(string fileName)
```

Possible error IDs: VSX_DRIVER_CONNECTION_ERROR, VSX_DRIVER_DEVICE_ERROR

Read the Parameter Set

Applicable to:
- SmartRunner
- SmartRunner 3-D

Reads the current parameter set from the device and saves it under the specified path and file name.

Asynchronous Function

```
public Task<(bool Succ, Error ErrorDesc)> DownloadParameterSet(string destinationFileName)
```

Synchronous Function

```
public (bool Succ, Error ErrorDesc) DownloadParameterSet(string destinationFileName)
```

Possible error IDs: VSX_DRIVER_SAVE_FILE_ERROR

Load Parameter Set

Applicable to:
- SmartRunner
- SmartRunner 3-D

Loads a parameter set stored under the specified path and file name and sends it to the device.

Asynchronous Function

```
public Task<(bool Succ, Error ErrorDesc)> UploadParameterSet(string sourceFileName)
```

Synchronous Function

```
public (bool Succ, Error ErrorDesc) UploadParameterSet(string sourceFileName)
```

Possible error IDs: VSX_DRIVER_CONNECTION_ERROR

Save the Parameter Set

Applicable to:
- SmartRunner
- SmartRunner 3-D

Saves the current parameter settings on the device. The specified values are set each time the device is started.

All parameter data is initially stored on the device in a volatile state and can be reset by a restart or by *LoadParameterSetOnDevice*. Only *SaveParameterSetOnDevice* permanently saves the parameters.

Asynchronous Function

```
public Task<(bool Succ, Error ErrorDesc)> SaveParameterSetOnDevice()
```

Synchronous Function

```
public (bool Succ, Error ErrorDesc) SaveParameterSetOnDevice()
```

Possible error IDs: VSX_DRIVER_CONNECTION_ERROR

Load Parameter Settings

Applicable to:
- SmartRunner
- SmartRunner 3-D

Loads the parameter settings saved via *SaveParameterSetOnDevice* on the device. The parameters then have the previously saved values. A current parameter list is transmitted.

Asynchronous Function

```
public Task<(bool Succ, List<Parameter> ParameterList, Error ErrorDesc)>
LoadParameterSetOnDevice()
```

Synchronous Function

```
public (bool Succ, List<Parameter> ParameterList, Error ErrorDesc) LoadParameterSetOnDevice()
```

Possible error IDs: VSX_DRIVER_GENERAL_ERROR, VSX_DRIVER_DATA_ERROR, VSX_DRIVER_CONNECTION_ERROR

Load Factory Settings

Applicable to:
- SmartRunner
- SmartRunner 3-D

Loads the factory settings of all parameters on the device. A current parameter list is transmitted.

Asynchronous Function

```
public Task<(bool Succ, List<Parameter> ParameterList, Error ErrorDesc)>
LoadDefaultParameterSetFromDevice()
```

Synchronous Function

```
public (bool Succ, List<Parameter> ParameterList, Error ErrorDesc)
LoadDefaultParameterSetFromDevice()
```

Possible error IDs: VSX_DRIVER_GENERAL_ERROR, VSX_DRIVER_DATA_ERROR, VSX_DRIVER_CONNECTION_ERROR

Read Out Sensor Data

Applicable to:
- SmartRunner
- SmartRunner 3-D

Starts reading out sensor data. The sensor data for each trigger is packaged in *VsxDynamicContainer*. The data can be called up using the *GetDynamicContainer* function. The *bufferSize* specifies how many containers can be buffered by the driver, the *startCondition* specifies the container from which buffering should be performed, and the *strategy* determines what should happen if the buffer is full. DROP_OLDEST indicates that the oldest stored container is discarded, while DROP_WRITE means the latest container received is discarded.

Asynchronous Function

```
public void ResetDynamicContainerGrabber(int bufferSize,
PF.Foundation.Communication.Vsx.Strategy strategy =
PF.Foundation.Communication.Vsx.Strategy.DROP_OLDEST)
```

Synchronous Function

```
public void ResetDynamicContainerGrabber(int bufferSize,
PF.Foundation.Communication.Vsx.Strategy strategy =
PF.Foundation.Communication.Vsx.Strategy.DROP_OLDEST)
```

Possible error IDs: None

Output of the Oldest Buffered DynamicContainer

Applicable to:
- SmartRunner
- SmartRunner 3-D

Returns the oldest buffered DynamicContainer (see "Read Out Sensor Data" on page 7). This function returns an error after *timeoutMs* millisecons if no new DynamicContainer is available by then.

Asynchronous Function

```
public Task<(bool Succ, IVsxDynamicContainer Container, int NumberOfDiscardedItems, Error
ErrorDesc)> GetDynamicContainer(int timeoutMs = System.Threading.Timeout.Inifinite)
```

Synchronous Function

```
public (bool Succ, IVsxDynamicContainer Container, int NumberOfDiscardedItems, Error
ErrorDesc) GetDynamicContainer(int timeoutMs = Timeout.Infinite)
```

Possible error IDs: VSX_DRIVER_INIT_ERROR, VSX_DRIVER_TIMEOUT_ERROR

Read Out Log Data

Applicable to:
- SmartRunner
- SmartRunner 3-D

Restarts the readout of log data. The log data can be called up using the *GetLogMessage* function. The *bufferSize* specifies how many log data messages can be buffered by the driver. The *typeMask* specifies which log data types should be transferred from the device, and the *strategy* specifies what should happen if the buffer is full. DROP_OLDEST indicates that the oldest stored log message is discarded, while DROP_WRITE means the latest log message received is discarded.

Asynchronous Function

```
public void ResetLogMessageGrabber(int bufferSize, int typeMask,
PF.Foundation.Communication.Vsx.Strategy strategy =
PF.Foundation.Communication.Vsx.Strategy.DROP_OLDEST)
```

Synchronous Function

```
public void ResetLogMessageGrabber(int bufferSize, int typeMask,
PF.Foundation.Communication.Vsx.Strategy strategy =
PF.Foundation.Communication.Vsx.Strategy.DROP_OLDEST)
```

Possible error IDs: None

Output of the Oldest Buffered LogMessage

Applicable to:
- SmartRunner
- SmartRunner 3-D

Returns the oldest buffered *LogMessage* (see "Read Out Log Data" on page 8). This function returns an error after *timeoutMs* millisecons if no new DynamicContainer is available by then.

Asynchronous Function

```
public Task<(bool Succ, VsxLogMessage LogMessage, int NumberOfDiscardedItems, Error
ErrorDesc)> GetLogMessage(int timeoutMs = System.Threading.Timeout.Inifinite)
```

Synchronous Function

```
public (bool Succ, VsxLogMessage LogMessage, int NumberOfDiscardedItems, Error ErrorDesc)
GetLogMessage(int timeoutMs = Timeout.Infinite)
```

Possible error IDs: VSX_DRIVER_INIT_ERROR, VSX_DRIVER_TIMEOUT_ERROR

Properties

Information about Firmware Updates

Applicable to:
- SmartRunner
- SmartRunner 3-D

The *FirmwareStateChannelReader* provides information about the current status of the firmware update

```
public ChannelReader<FirmwareState> FirmwareStateChannelReader
```

Connection Status

Applicable to:
- SmartRunner
- SmartRunner 3-D

Indicates the connection status.

```
public bool Connected { get; }
```

Timeout

Applicable to:
- SmartRunner
- SmartRunner 3-D

Timeout specifies (in ms) how long to wait for a response to a request sent to the device. Depending on the connection type, the default value is DEFAULT_ETHERNET_TIMEOUT_MS or DEFAULT_SERIAL_TIMEOUT_MS.

```
public int WaitTimeout { get; set; }
```

Number of Discarded DynamicContainers or LogMessages

Applicable to:
- SmartRunner
- SmartRunner 3-D

Specifies the number of DynamicContainers or LogMessages discarded since the last reset grabber, once there was no more space in the buffer.

DynamicContainer

```
public int MissingContainerFramesCounter { get; }
```

LogMessages

```
public int MissingLogMessagesCounter { get; }
```

Number of Buffered DynamicContainers or LogMessages

Applicable to:
- SmartRunner
- SmartRunner 3-D

Specifies how many DynamicContainers or LogMessages can be buffered by the driver.

DynamicContainer

```
public int DynamicContainerQueueSize { get; }
```

LogMessages

```
public int LogMessageQueueSize { get; }
```

Which DynamicContainers or LogMessages Are Discarded

Applicable to:
- SmartRunner
- SmartRunner 3-D

Specifies which DynamicContainers or LogMessages should be discarded if the buffer is full. DROP_OLDEST discards the oldest stored container or log message, while DROP_WRITE discards the most recently received.

DynamicContainer

```
public Strategy DynamicContainerGrabberStrategy { get; }
```

LogMessages

```
public Strategy LogMessageGrabberStrategy { get; }
```

Events

Loss of Connection

Applicable to:
- SmartRunner
- SmartRunner 3-D

The event is triggered as soon as the driver detects a loss of connection to the device. The parameters are the IP address of the previously connected device and a message indicating why the connection was lost. This event is only triggered if a TCP/IP connection is used.

```
public event Action<string, string> OnDisconnect
```

Auxiliary Classes

Device Information (PF.Types.Device)

Applicable to:
- SmartRunner
- SmartRunner 3-D

Contains information about the currently connected device.

```
public string PhysicalAddress;

public int PhysicalPort;

public string IpAddress;

public string NetworkMask;

public string Gateway;

public string MacAddress;

public string Identifier;

public string FirmwareVersion;

public string SensorType;
```

Error Information (PF.Types.Error)

Applicable to:
- SmartRunner
- SmartRunner 3-D

Contains information about an error that has occurred.

```
public ErrorId Id;

public string Tag;

public string Message;
```

The Following Error IDs Are Possible

Applicable to:
- SmartRunner
- SmartRunner 3-D

```
VSX_DRIVER_NO_ERROR = 0x0,

VSX_DRIVER_INIT_ERROR = -0x1,

VSX_DRIVER_TIMEOUT_ERROR = -0x2,

VSX_DRIVER_SAVE_FILE_ERROR = -0x3,

VSX_DRIVER_DATA_ERROR = -0x4,

VSX_DRIVER_CONNECTION_ERROR = -0x5,

VSX_DRIVER_INVALID_DATA_ERROR = -0x6,

VSX_DRIVER_DEVICE_ERROR = -0x7,

VSX_DRIVER_GENERAL_ERROR = -0x1000
```

Parameters (PF.VsxDriver.Parameter)

Applicable to:
- SmartRunner
- SmartRunner 3-D

Contains information about a device parameter. Important properties here are the details of the version and IDs, which are required for setting a parameter. Another property is Value, which provides the current parameter value. Not every property is used for every parameter.

```
ushort settingsVersion;

ushort configVersion;

string configId;

string parameterId;

string name;

Vsx.ParameterTypes type;

Vsx.ValueTypes valueType;

bool enable;

bool visible;

object min;

object max;

object step;

string userLevel;

object value;

object defaultValue;

string unit;

List<ItemTuple> items;
```

Information about the Firmware Update (PF.Foundation.Communication.Vsx.Device.FirmwareState)

Applicable to:
- SmartRunner
- SmartRunner 3-D

Contains information about the current status of a firmware update that is running.

```
public int Id;

public string Tag;

public string Message;
```

List of IVsxMessages (PF.Foundation.Communication.Vsx.Message.VsxDynamicContainerMessage: IVsxMessage)

Applicable to:
- SmartRunner
- SmartRunner 3-D

Contains a list of IVsxMessages, which in turn contain data sent by the device. The messages it contains are identified in the list using a string. The possible messages are device-specific. Example of a VsxDynamicContainerMessage.

```
public bool ContainsMessage(string tag)

public IVsxMessage GetMessage(string tag)
```

Image Data (PF.Foundation.Communication.Vsx.Message.VsxImageData2Message: IVsxMessage)

Applicable to:
- SmartRunner
- SmartRunner 3-D

Contains image data of a specific image. Depending on whether the individual image values are bytes or floats, they are stored in the respective ImageData or ImageDataFloats array.

```
public ImageData2Format;

public int Width;

public int Height;

public int LinePitch;

public long FrameCounter;

public double CoordinateScale;

public double CoordinateOffset;

public double AxisMin;

public double AxisMax;

public double InvalidDataValue;

public byte[] ImageData;

public float[] ImageDataFloats;
```

PF.Foundation.Communication.Vsx.Message.VsxDisparityDescriptorMessage: IVsxMessage

Applicable to:
- SmartRunner
- SmartRunner 3-D

```
public double FocalLength;

public double PrincipalPointU;

public double PrincipalPointV;

public double Baseline;
```

PF.Foundation.Communication.Vsx.Message.VsxTransformationMessage: IVsxMessage

Applicable to:
- SmartRunner
- SmartRunner 3-D

```
public double TranslationTX;

public double TranslationTY;

public double TranslationTZ;

public double QuaternionQ0;

public double QuaternionQ1;

public double QuaternionQ2;

public double QuaternionQ3;
```

SmartRunner

DynamicContainer

A DynamicContainer received by the sensor can contain the following messages:

TagName	Type	Status	Description
"Image"	ImageData[Mono8]	Optional	Image data
"Line"	LineMessage	Optional	Line data

Structure of the LineMessage Type

Main Property

Line

Returns a list of points. Each point consists of image and world coordinates, and quality information. The points are of type *Coordinate*. In detail, the point information looks as follows:

- ContentValue: Reserved for future use
- ImageCoordinate: Image coordinates in subpixels
- Intensity: Value that correlates with the grayscale value point
- LineID: Always 0 in single-line systems
- Quality: Point quality as a percentage
- SegmentID: Line segment number if supported, otherwise 0
- Valid: True if point is valid
- WorldCoordinate: World coordinates in mm

Additional Properties (Meta-Information)

Meta-information provides additional information about the data.

MinX

Minimum possible X value of the device family, usually negative in µm.

MaxX

Maximum possible X value of the device family, usually positive in µm.

MinZ

Minimum possible Z value of the device family, positive in µm.

MaxZ

Maximum possible Z value of the device family, positive in µm.

FrameCounter

Current frame counter, related data has the same counter.

ScaleXYZ, ScaleC

Scaling factors for the internal calculation of the line data.

Width

Maximum number of possible line points.

Format

Reserved for future use.

Status

Reserved for future use.

Lines

Returns a list of lines. Usually, the list contains only one line. Each line consists of a list of points, the points of the first line can be called directly via Line.

SmartRunner 3-D Stereo

A DynamicContainer received by the sensor can contain the following messages:

TagName	Type	Status	Description
"LeftRaw"	ImageData2[Mono8]	Optional	Left raw data image
"RightRaw"	ImageData2[Mono8]	Optional	Right raw data image
"LeftRectified"	ImageData2[Mono8]	Optional	Left rectified image
"RightRectified"	ImageData2[Mono8]	Optional	Right rectified image
"DisparityC"	ImageData2[Coord3D_C16]	Optional	Disparity map (1)
"DisparityDesc"	DisparityDescriptor	Optional	Required for disparity map (1)
"CalibratedA"[a]	ImageData2[Coord3D_A32f]	Optional	X level of the calibrated grid
"CalibratedB"[b]	ImageData2[Coord3D_B32f]	Optional	Y level of the calibrated grid
"CalibratedC"[c]	ImageData2[Coord3D_C32f]	Optional	Z level of the calibrated grid
"Transformation"	Transformation	Optional	Transformation of XYZ
"Confidence"	ImageData2[Confidence8]	Optional	Confidence map (2)

a. The three layers "CalibratedA," "CalibratedB," and "CalibratedC" are not sent on the device side. The "ProcessDeviceData" function (VsxDynamicContainerMessage container) generates these three messages and calculates the respective level of the "PointCloud."

b. The three layers "CalibratedA," "CalibratedB," and "CalibratedC" are not sent on the device side. The "ProcessDeviceData" function (VsxDynamicContainerMessage container) generates these three messages and calculates the respective level of the "PointCloud."

c. The three layers "CalibratedA," "CalibratedB," and "CalibratedC" are not sent on the device side. The "ProcessDeviceData" function (VsxDynamicContainerMessage container) generates these three messages and calculates the respective level of the "PointCloud."

SmartRunner 3-D Time-of-Flight

A DynamicContainer received by the sensor can contain the following messages:

TagName	Type	Status	Description
"Amplitude"	ImageData2[Mono16]	Optional	Amplitude of the signal
"Depth"	ImageData2[Mono16]	Optional	Depth image
"CalibratedA"	ImageData2[Coord3D_A16]	Optional	X level of the calibrated grid
"CalibratedB"	ImageData2[Coord3D_B16]	Optional	Y level of the calibrated grid
"CalibratedC"	ImageData2[Coord3D_C16]	Optional	Z level of the calibrated grid
"Transformation"	Transformation	Optional	Transformation of XYZ
"Confidence"	ImageData2[Confidence8]	Optional	Confidence map (1)