Teledyne DALSA • 880 Rue McCaffrey • St-Laurent, Québec, H4T 2C7 • Canada
http://www.teledynedalsa.com/Z-Trak

3D-L-AN0002: Using Z-Trak with MVTEC HALCON HDevelop

Using Z-Trak with MVTec HALCON HDevelop

All Z-Trak Laser Profile Cameras.

Overview

Z-Trak laser profile cameras are GigE Vision compliant. As such, cameras can be used with any image processing software that supports 16-bit image acquisition using the GigE Vision protocol. Camera features follow the GenICam standard v3.0 (SFNC v2.3). This application note demonstrates how to acquire range images from a Z-Trak camera using Halcon HDevelop and display the resulting 3D object model.

Prerequisites

The following table lists the recommended Z-Trak firmware and software for this camera model.

Z-Trak Firmware Design	Software SDK
1.01 and greater	MVTec HALCON (17.12 or higher recommended)
	Sapera LT 8.40 or higher (optional, for camera configuration via CamExpert X program)

Software

MVTec HALCON (full version) machine vision software. Available from the MVTec website:

https://www.mvtec.com/products/halcon/

Refer to MVTec documentation for more information.

The Z-Trak HALCON demo program is available for download from the Teledyne DALSA website:

http://www.teledynedalsa.com/en/products/imaging/3d-cameras/

Sapera LT SDK (full version), the image acquisition and control software development kit (SDK) for Teledyne DALSA cameras is available for download from the Teledyne DALSA website:

http://teledynedalsa.com/imaging/support/downloads/sdks/

Sapera LT includes the CamExpert X application, which provides a graphical user interface to access camera features for configuration and setup.

HALCON Demo Program Overview

The Z-Trak HALCON demo program demonstrates how to acquire a range image from the Z-Trak camera and convert it to a 3D model ready for processing by Halcon 3D point cloud functions. Invalid points are removed from the image to improve image display.

Depending on the object being scanned, Z-Trak settings may need to be adjusted from default values to obtain high quality profiles and reduce noise. The image height should contain enough profiles to image the object. Accurate Y scale is dependent on the speed of the object being scanned and the profile rate.

Refer to the Z-Trak user manual for information on how to optimize Z-Trak parameters for profile acquisition.

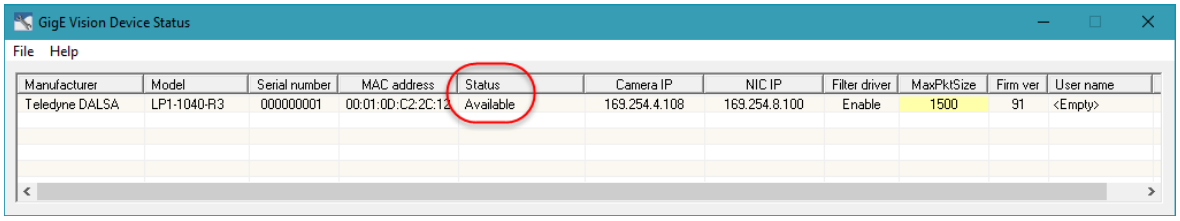
The program performs the following operations:

1. Connects to the Z-Trak and reads features from the device related to image information (*Width*, *Height*, *PixelFormat*, *AcquisitionLineRate*) and X and Z (A and C) coordinate scaling factors (*Scan3dCoordinateScale* and *Scan3dCoordinateOffset*)
2. Sets the minimum timeout for acquiring a full scan (to avoid a timeout).
3. Checks the pixel format and grabs a single image; the demo program supports the Coord3D_C16 pixel format.
4. Removes invalid pixels from the acquired image using a threshold operation to improve visualization.
5. Uses the scaling factors retrieved from the camera to create X, Y, and Z plane images to generate a Halcon ObjectModel3D object for 3D visualization and further processing.

Connecting to the Z-Trak

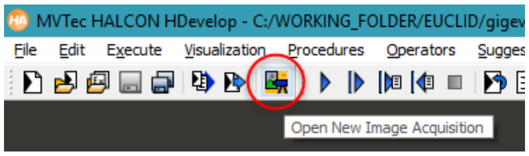
The demo program uses the first GigE Vision device found therefore only one Z-Trak camera must be connected to the computer running the application. However, if necessary, the source code can be modified to select the required device.

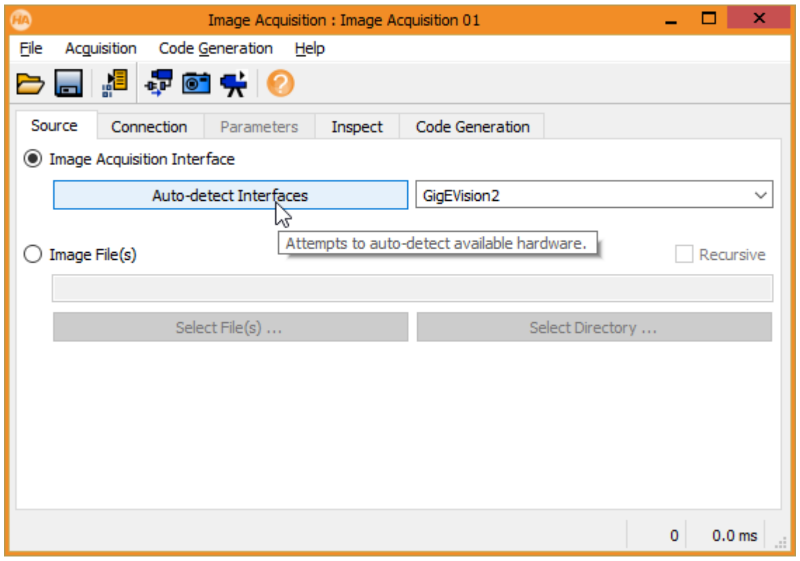
The Teledyne DALSA GigE Vision Device Status (installed with Sapera LT) can be used to verify if the device is available:



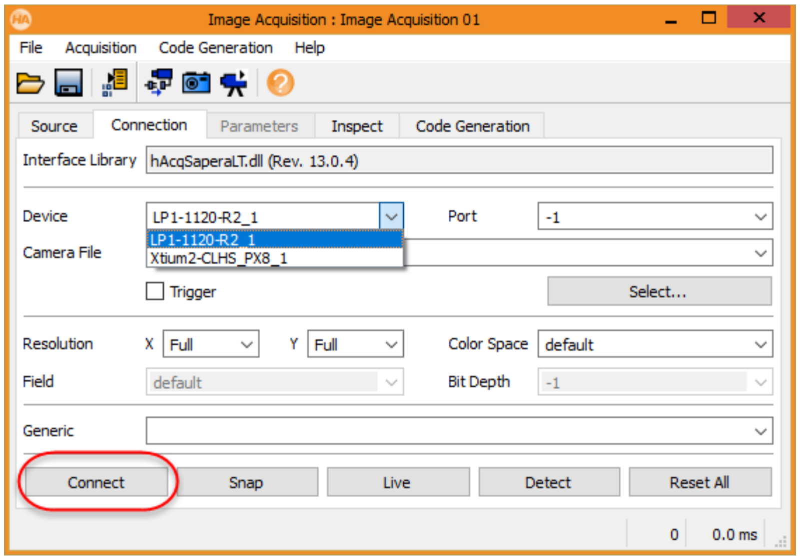
Manufacturer	Model	Serial number	MAC address	Status	Camera IP	NIC IP	Filter driver	MaxPktSize	Firm ver	User name
Teledyne DALSA	LP1-1040-R3	000000001	00:01:0D:C2:2C:12	Available	169.254.4.108	169.254.8.100	Enable	1500	91	<Empty>

Alternatively, the HALCON Image Acquisition Assistant, available from the HALCON menu bar, can be used to detect devices on the GigEVision2 interface:


MVTec HALCON HDevelop - C:/WORKING_FOLDER/EUCLID/gigev
File
Edit
Execute
Visualization
Procedures
Operators
Sugges
Open New Image Acquisition



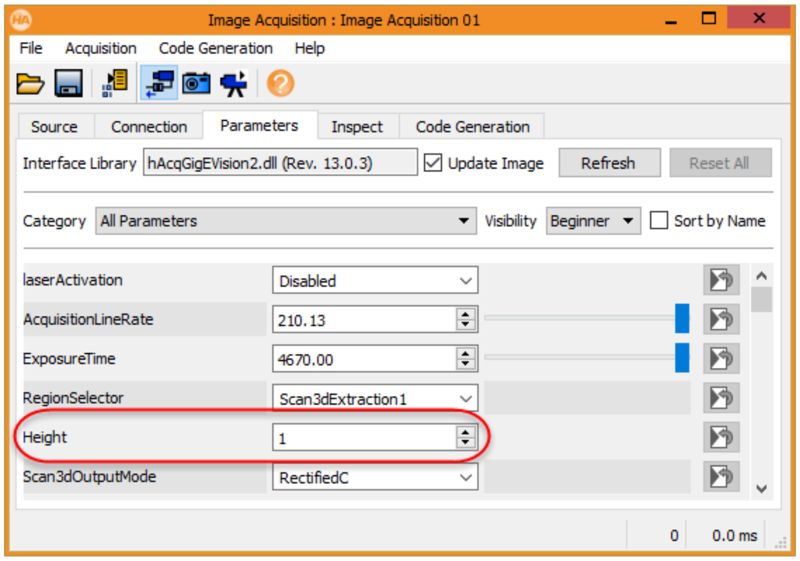
The required device can be selected from the Connection tab from the available devices; click **Connect** to establish communication with the device.



Interface Library: hAcqSaperaLT.dll (Rev. 13.0.4)
Device: LP1-1120-R2_1 (LP1-1120-R2_1, Xtium2-CLHS_PX8_1)
Port: -1
Camera File:
Trigger: ☐
Resolution: X Full, Y Full
Color Space: default
Field: default
Bit Depth: -1
Generic:
0 0.0 ms

The Z-Trak feature settings can be modified and are available in the Parameters tab.

The scan Height must be adjusted to acquire more than 1 profile; set this value to the required number of profiles for a range image:



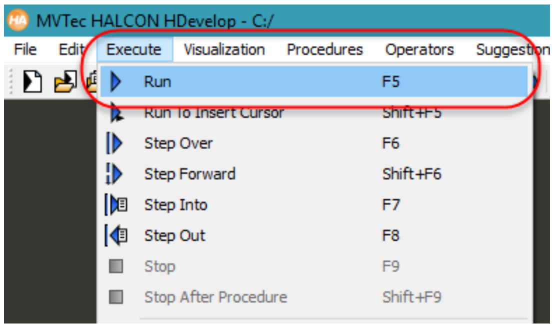
Interface Library: hAcqGigEVision2.dll (Rev. 13.0.3)
Update Image: ☑
Category: All Parameters
Visibility: Beginner
Sort by Name: ☐

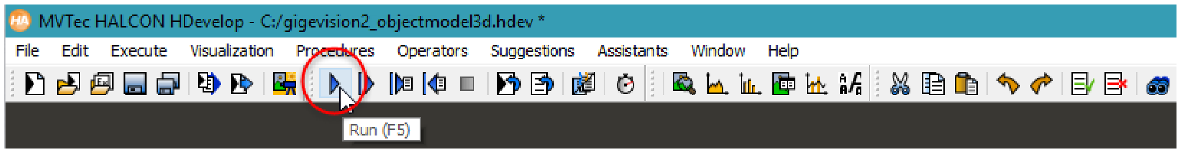
laserActivation	Disabled
AcquisitionLineRate	210.13
ExposureTime	4670.00
RegionSelector	Scan3dExtraction1
Height	1
Scan3dOutputMode	RectifiedC

0 0.0 ms

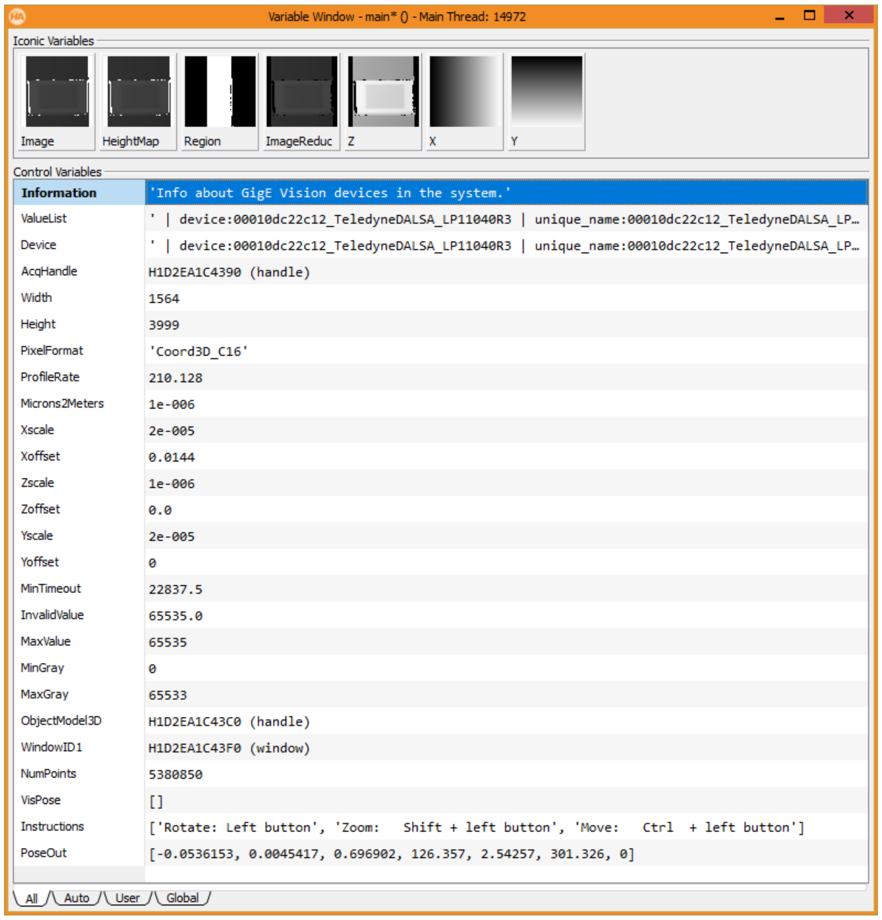
Running the Demo Program

To run the Z-Trak HALCON demo program, load the program and use the **Execute > Run** menu command, click the run icon in the menu bar or use **F5**.



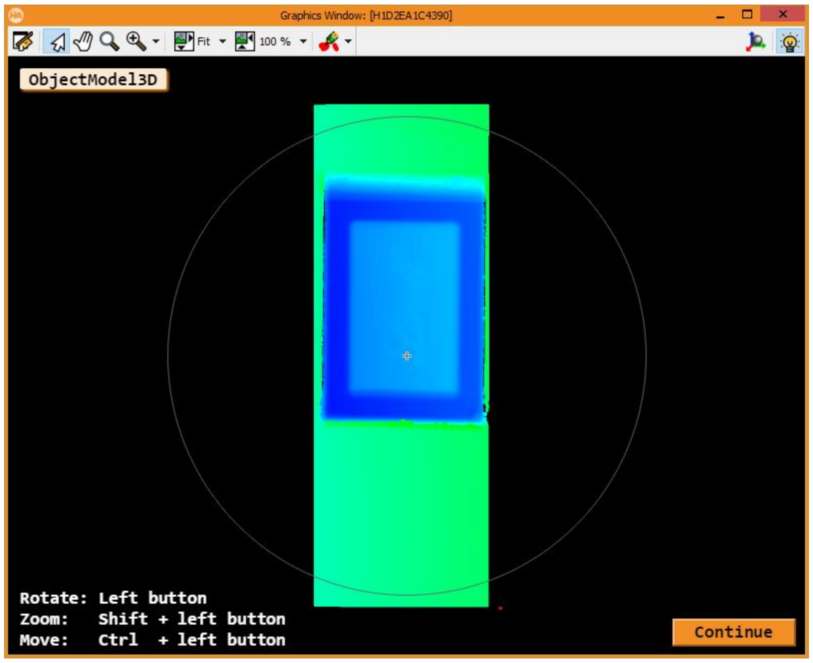


Upon execution the HALCON Iconic and Control variables are populated in the Variable Window.

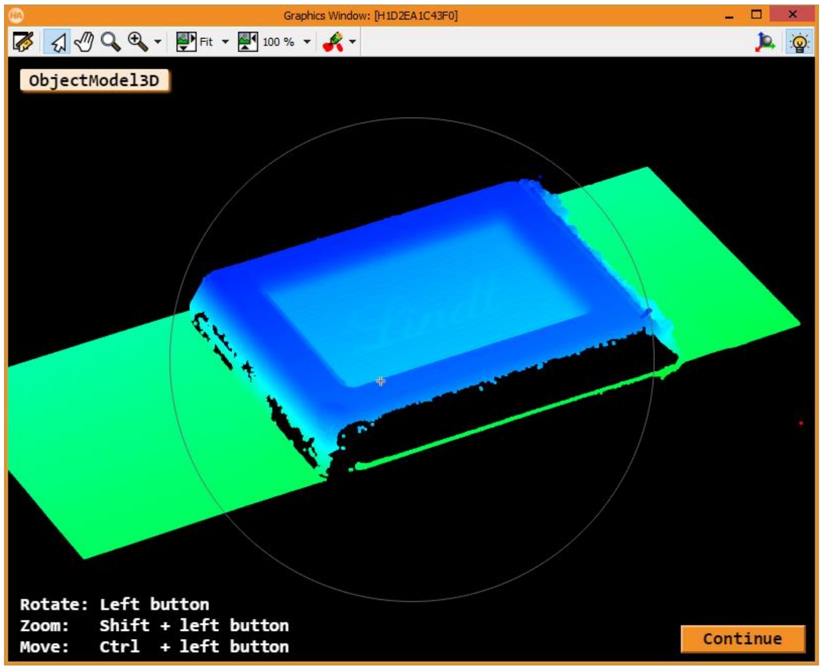


Information	'Info about GigE Vision devices in the system.'
ValueList	' | device:00010dc22c12_TeledyneDALSA_LP11040R3 | unique_name:00010dc22c12_TeledyneDALSA_LP…
Device	' | device:00010dc22c12_TeledyneDALSA_LP11040R3 | unique_name:00010dc22c12_TeledyneDALSA_LP…
AcqHandle	H1D2EA1C4390 (handle)
Width	1564
Height	3999
PixelFormat	'Coord3D_C16'
ProfileRate	210.128
Microns2Meters	1e-006
Xscale	2e-005
Xoffset	0.0144
Zscale	1e-006
Zoffset	0.0
Yscale	2e-005
Yoffset	0
MinTimeout	22837.5
InvalidValue	65535.0
MaxValue	65535
MinGray	0
MaxGray	65533
ObjectModel3D	H1D2EA1C43C0 (handle)
WindowID1	H1D2EA1C43F0 (window)
NumPoints	5380850
VisPose	[]
Instructions	['Rotate: Left button', 'Zoom: Shift + left button', 'Move: Ctrl + left button']
PoseOut	[-0.0536153, 0.0045417, 0.696902, 126.357, 2.54257, 301.326, 0]

The acquired image is rendered as an ObjectModel3D object in the Graphics Window.

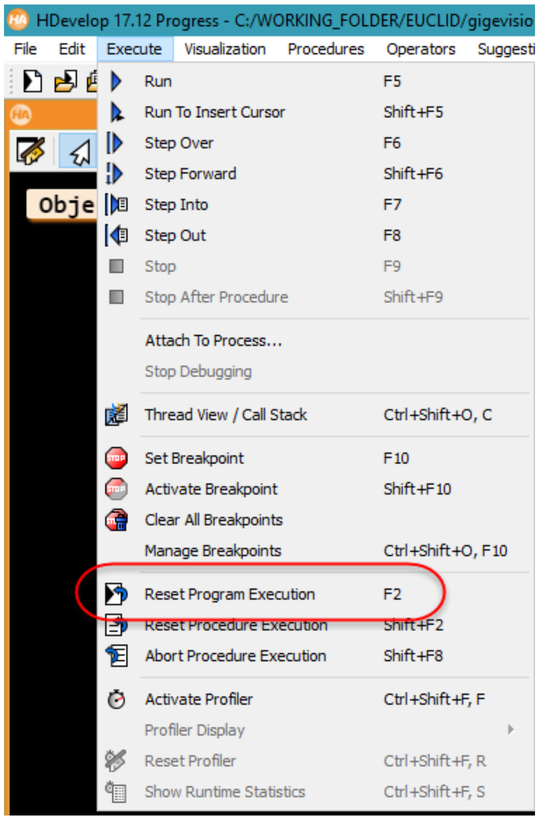


Use the mouse and keyboard combinations to rotate, zoom and move the 3D view.



Click **Continue** in the Graphics Window to end the program execution.

To execute the program again, first use the **Execute > Reset Program Execution** menu command or press **F2** before running the program.


HDevelop 17.12 Progress - C:/WORKING_FOLDER/EUCLID/gigevisio
File
Edit
Execute
Visualization
Procedures
Operators
Suggesti
Run F5
Run To Insert Cursor Shift+F5
Step Over F6
Step Forward Shift+F6
Step Into F7
Step Out F8
Stop F9
Stop After Procedure Shift+F9
Attach To Process...
Stop Debugging
Thread View / Call Stack Ctrl+Shift+O, C
Set Breakpoint F10
Activate Breakpoint Shift+F10
Clear All Breakpoints
Manage Breakpoints Ctrl+Shift+O, F10
Reset Program Execution F2
Reset Procedure Execution Shift+F2
Abort Procedure Execution Shift+F8
Activate Profiler Ctrl+Shift+F, F
Profiler Display
Reset Profiler Ctrl+Shift+F, R
Show Runtime Statistics Ctrl+Shift+F, S
Obje

HDevelop Source Code

```
* Image Acquisition 01: Code generated by Image Acquisition 01
info_framegrabber('GigEVision2', 'device', Information, ValueList)
Device : = ValueList[0]
open_framegrabber('GigEVision2', 0, 0, 0, 0, 0, 0, 'progressive', -1, 'default', -1, 'false',
'default', Device, 0, -1, AcqHandle)

* Get image information from device
get_framegrabber_param(AcqHandle, 'Width', Width)
get_framegrabber_param(AcqHandle, 'Height', Height)
get_framegrabber_param(AcqHandle, 'PixelFormat', PixelFormat)
get_framegrabber_param(AcqHandle, 'AcquisitionLineRate', ProfileRate)

* Get scaling factors from device
Microns2Meters : = 1e-6
set_framegrabber_param(AcqHandle, 'Scan3dCoordinateSelector', 'CoordinateA')
get_framegrabber_param(AcqHandle, 'Scan3dCoordinateScale', Xscale)
get_framegrabber_param(AcqHandle, 'Scan3dCoordinateOffset', Xoffset)
Xscale : = Xscale * Microns2Meters
Xoffset : = Xoffset * Microns2Meters
set_framegrabber_param(AcqHandle, 'Scan3dCoordinateSelector', 'CoordinateC')
get_framegrabber_param(AcqHandle, 'Scan3dCoordinateScale', Zscale)
get_framegrabber_param(AcqHandle, 'Scan3dCoordinateOffset', Zoffset)
Zscale : = Zscale * Microns2Meters
Zoffset : = Zoffset * Microns2Meters
* Y scale is assumed to be same as X scale: adjust according to your application setup
* Accurate Y scale is dependent on the speed of the object being scanned and the profile rate
Yscale : = Xscale
Yoffset : = 0

* Set minimum timeout for acquiring a full scan(computed in milliseconds with 20 % extra)
MinTimeout : = (Height / ProfileRate) * 1000 * 1.2
set_framegrabber_param(AcqHandle, 'grab_timeout', MinTimeout)

* Check pixel format
if (PixelFormat == 'Coord3D_C16')
* Grab one image from 3D scanner
   grab_image(Image, AcqHandle)
   * Get height map from image(in this case the image itself)
   HeightMap : = Image
else
* TODO: add other formats
endif

* Remove invalid pixels from height map by reducing grayscale domain
set_framegrabber_param(AcqHandle, 'Scan3dCoordinateSelector', 'CoordinateC')
get_framegrabber_param(AcqHandle, 'Scan3dInvalidDataValue', InvalidValue)
MaxValue : = lsh(1, 16) - 1
if (InvalidValue = 0)
   * Removing 0 from domain
   MinGray : = 1
   MaxGray : = MaxValue - 1
   elseif(InvalidValue = MaxValue)
   * Removing max value from domain
   MinGray : = 0
   MaxGray : = MaxValue - 2
else
   * Invalid value not supported.Skipping domain reduction
   MinGray : = 0
   MaxGray : = MaxValue - 1
endif
threshold(HeightMap, Region, MinGray, MaxGray)
reduce_domain(HeightMap, Region, ImageReduced)

* Create Z image with correct scaling in meters
```

```
convert_image_type(ImageReduced, Z, 'real')
scale_image(Z, Z, Zscale, Zoffset)

* Create X image with correct scaling in meters
gen_image_surface_first_order(X, 'real', 0, Xscale, Xoffset, 0, 0, Width, Height)
* Create Y image with correct scaling in meters
gen_image_surface_first_order(Y, 'real', Yscale, 0, Yoffset, 0, 0, Width, Height)
* Generate an ObjectModel3D
xyz_to_object_model_3d(X, Y, Z, ObjectModel3D)

dev_open_window(0, 0, 800, 600, 'black', WindowID1)
set_display_font(WindowID1, 16, 'mono', 'true', 'false')
get_object_model_3d_params(ObjectModel3D, 'num_points', NumPoints)
if (NumPoints > 0)
   VisPose : = []
   Instructions[0] : = 'Rotate: Left button'
   Instructions[1] : = 'Zoom:   Shift + left button'
   Instructions[2] : = 'Move:   Ctrl  + left button'
   visualize_object_model_3d(WindowID1, ObjectModel3D, [], VisPose, ['color_attrib', 'lut'],
['coord_z', 'rainbow'], 'ObjectModel3D', [], Instructions, PoseOut)
else
   disp_message(WindowID1, 'No 3D points in the 3D object model!', 'window', 250, 200, 'black',
'true')
   stop()
endif

* Clear the ObjectModel3D
clear_object_model_3d(ObjectModel3D)

* Close device and exit program
close_framegrabber(AcqHandle)
```