Scientific Camera MATLAB .NET Interface Guide

For Models CS895MU, CS895CU, CS505MU, CS505CU, CS505MUP, CS235MU, CS235CU, CS2100, 8050, 8051, 4070, 1500, 1501, and 340

Table of Contents

Chapter 1 Introduction

1.1. Overview

The Thorlabs Scientific Camera MATLAB .NET Manual contains information to help you program your own MATLAB applications to operate Thorlabs scientific cameras. The Thorlabs Scientific Camera .NET interface is part of the Thorlabs Scientific Camera software packages that can be downloaded for free on the Thorlabs website. In addition to drivers and documentations, the software package includes Software Development Kit (SDK) for creating your own applications. Example codes are also provided to make it easier to start programming with your Thorlabs Scientific Camera.

1.2. MATLAB .NET connectivity

The MATLAB .NET connectivity functions allows you to create and interact with the Thorlabs Scientific Camera .NET SDK. For more information on MATLAB .NET connectivity, please refer to MATLAB Manuals.

1.3. Thorlabs Scientific Camera .NET Application Programming Interface (API)

For complete information on programming with Thorlabs Scientific Camera .NET SDK, please refer to the *DotNet, LabVIEW, and MATLAB scientific-camera programming guide.chm* help file.

Chapter 2 Getting Started in MATLAB

2.1. Before you begin

IMPORTANT: In order to properly load all referenced libraries, the library files should be copied to the current folder from which the MATLAB .m script file is being run.

IMPORTANT: TLCameraSDK, TLCamera and color processing related .NET objects need to be properly disposed by explicitly calling their respective Dispose methods after you finished using them and before removing their handles from memory.

2.2. Reference Thorlabs Scientific Camera .NET assembly

Make TLCamera .NET assembly visible to MATLAB:

```
      NET.addAssembly([pwd, '\Thorlabs.TSI.TLCamera.dll']);
```

This points to the "Thorlabs.TSI.TLCamera.dll" in the folder that contains the .m script.

2.3. Create an instance of ITLCameraSDK

You can create an instance of an “ITLCameraSDK” object by calling the static method “Thorlabs.TSI.TLCamera.TLCameraSDK.OpenTLCameraSDK”.

```
      tlCameraSDK = Thorlabs.TSI.TLCamera.TLCameraSDK.OpenTLCameraSDK;
```

2.4. Discover connected Thorlabs scientific cameras (Optional)

Through the Thorlabs.TSI.TLCameraIn.ITLCameraSDK object, discover the serial numbers of any connected cameras by

```
      serialNumbers = tlCameraSDK.DiscoverAvailableCameras;
```

2.5. Open a camera

Through the Thorlabs.TSI.TLCameraIn.ITLCameraSDK object, open a camera through its serial number:

```
      tlCamera = tlCameraSDK.OpenCamera(serialNumbers.Item(0), false);
```

2.6. Set and get camera properties

Camera properties can be set and get as:

```
      tlCamera.ExposureTime_us = 200000;
```

For detailed descriptions of the properties please refer to the API Help.

2.7. Start the camera

Set the ITLCamera.OperationMode to SoftwareTriggered, HardwareTriggered, or Bulb.

```
tlCamera.OperationMode =
Thorlabs.TSI.TLCameraInterfaces.OperationMode.SoftwareTriggered;
```

Then prepare the camera for software or hardware triggers:

```
tlCamera.Arm;
```

Depending on the OperationMode you chose above, either issue software trigger(s) by

```
tlCamera.IssueSoftwareTrigger;
```

Or send hardware trigger(s) to the camera trigger input.

2.8. Get image from camera

Because of the thread model of MATLAB, it is more convenient to poll the camera for images once the camera starts running.

Query in a loop whether an image is available on the camera, and then retrieve the image. This can be done in two ways:

1. Check

```
if (tlCamera.NumberOfQueuedFrames > 0)
```

And then call

```
imageFrame = tlCamera.GetPendingFrameOrNull;
```

to get the image frame data.

2. Directly call

```
imageFrame = tlCamera.GetPendingFrameOrNull;
```

and check

```
if ~isempty(imageFrame)
```

imageFrame contains image data that you can process or save.

2.9. Color and polarization processing of image data

For color or polarization cameras, the unprocessed monochrome Bayer image data from the camera can be processed by stand-alone image processing modules to color or polarization images. Please refer to included code samples for using the image processing modules.

2.10. Stop the camera

When you are finished acquiring images with the camera, call

```
tlCamera.Disarm;
```

2.11. Closing and disposing the camera

When you are done using a camera, dispose the tlCamera object by:

```
tlCamera.Dispose;
```

When you are done using all cameras, dispose the tlCameraSDK object by:

```
tlCameraSDK.Dispose;
```

Chapter 3 MATLAB code samples

The following MATLAB sample codes are installed with the Thorlabs Scientific Camera software package to help you program your own application.

Example	**Description**
SimpleImageAcquisition	Starts the camera and gets images by polling. The sample shows camera initialization, image acquisition and camera closing.
ColorProcessing	Demonstrates how to use a color camera. The sample shows how to set up custom color processing and process color image data from the camera
PolarizationProcessing	Demonstrates how to use a polarization camera. The sample shows how to convert Bayer patterned raw image data from the camera into Azimuth, Degree of Linear Polarization (DoLP) or Intensity images.
SoftwareTrigger	Demonstrates how to use software triggered acquisition.
HardwareTrigger	Demonstrates how to use hardware triggered acquisition.
ThorlabsCameraGUI	A comprehensive code sample using MATLAB GUI.

THORLABS

www.thorlabs.com