# OpenCV

# Windows Installation Manual

e-con Systems

Your Product Development Partner

Version 1.1

e-con Systems

7/27/2018

**Disclaimer**

e-con Systems reserves the right to edit/modify this document without any prior intimation of whatsoever.

# Contents

# Introduction to OpenCV

Open Source Computer Vision Library (OpenCV) is an open source computer vision and machine learning software library. OpenCV libraries are used to communicate with Cameras. APIs introduced in the OpenCV can be supported with all e-con Systems cameras.

This document helps you to install OpenCV in Windows and build a sample code to access the camera with OpenCV.

## Prerequisites

The prerequisites are as follows:

- Click here(https://cmake.org/download/) to download CMake.
- Download OpenCV from here(https://github.com/opencv/opencv).
- Click clone or download option and copy the URL

```
$ git clone <OpenCV_URL>
$ cd opencv
$ git checkout <opencv_version(3.3.1 or 3.4.1)>
```

- Create a source directory in the opencv folder and move all the files to the source folder
- Create a build directory in the opencv/
- Build OpenCV in your PC using Visual Studio

## Description

The following steps have been tested on Windows 10. OpenCV must work on any other relatively modern version of Windows OS.

# Building OpenCV

OpenCV is a sample command line application used to demonstrate some of the features of the e-con Systems cameras with OpenCV APIs.

The steps to build OpenCV are as follows:



## Step 1 - Launching CMake Window

In CMake window, select the OpenCV sources as source folder and OpenCV/build as build folder and click **Configure** button.

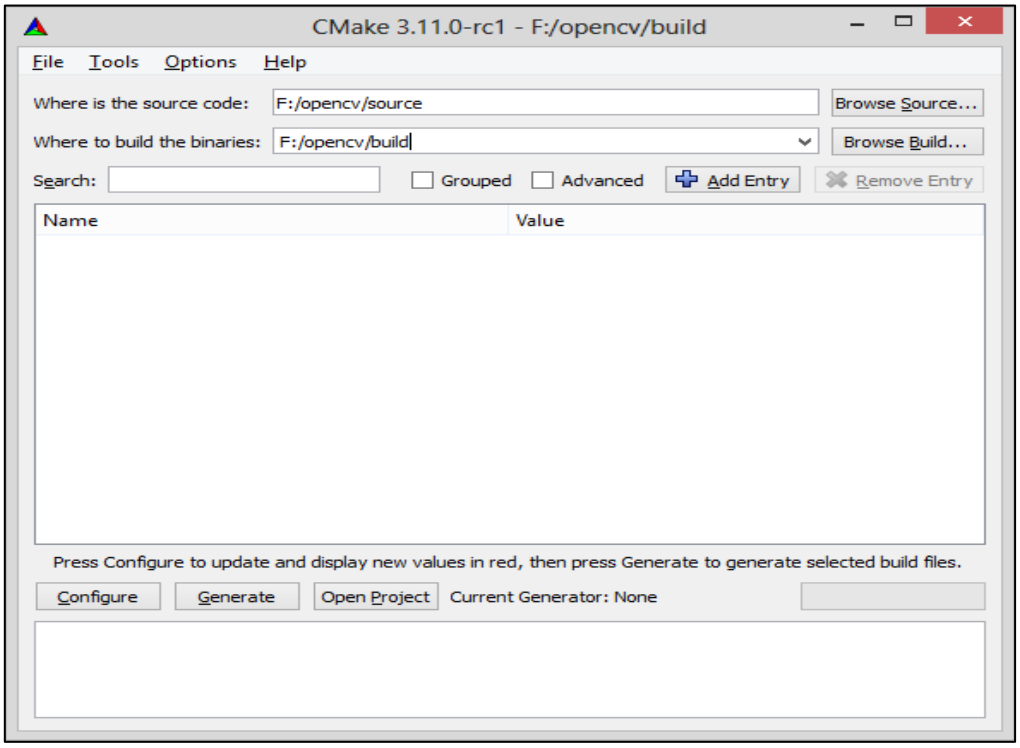



**Figure 1: CMake Source and Build Directory Specification Window**

## Step 2 - Selecting Visual Studio Version

A window prompting to select Visual Studio version along with x32 and x64 version appears. Select the appropriate options as shown below.

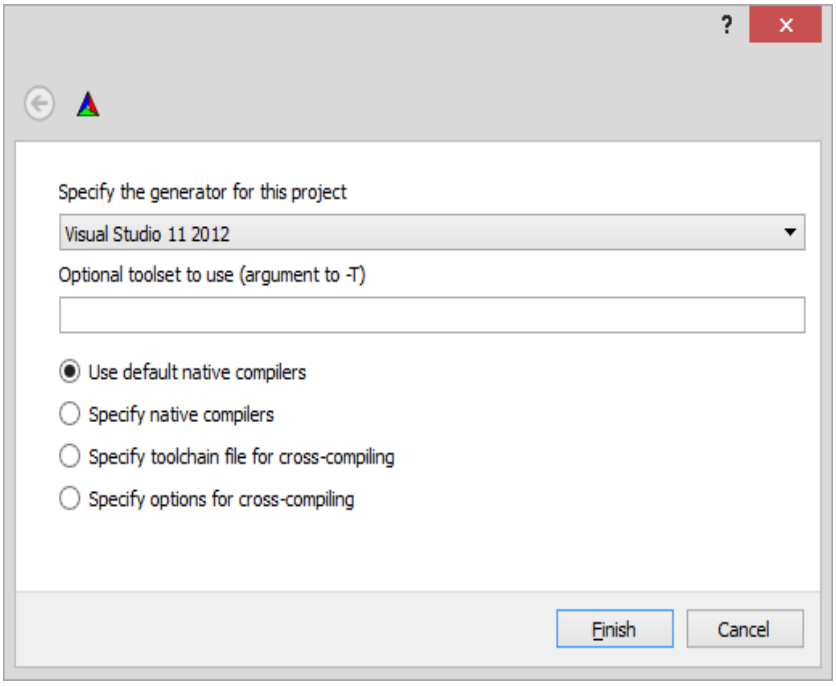
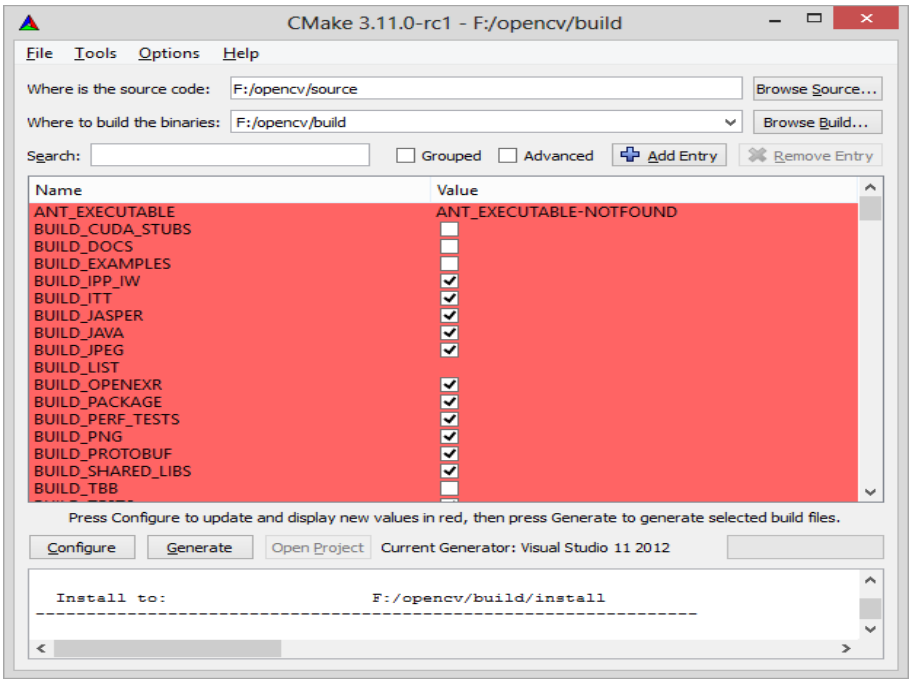



**Figure 2: Selecting Visual Studio Version in CMake**

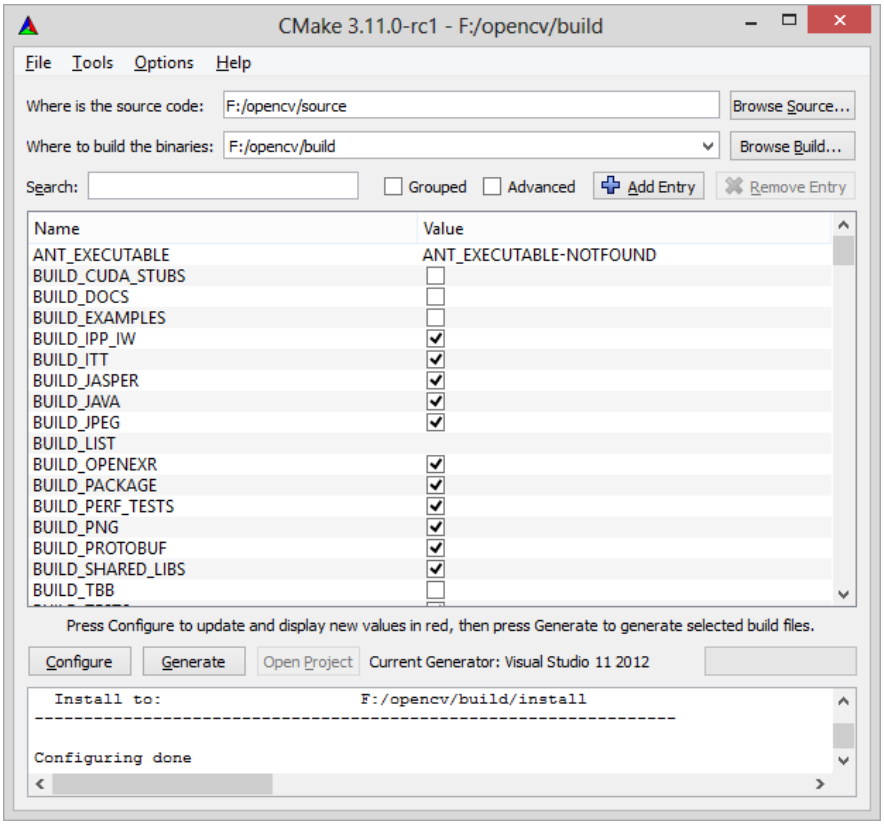
## Step 3 - Configuring and Generating CMake

The steps to configure and generate CMake are as follows:

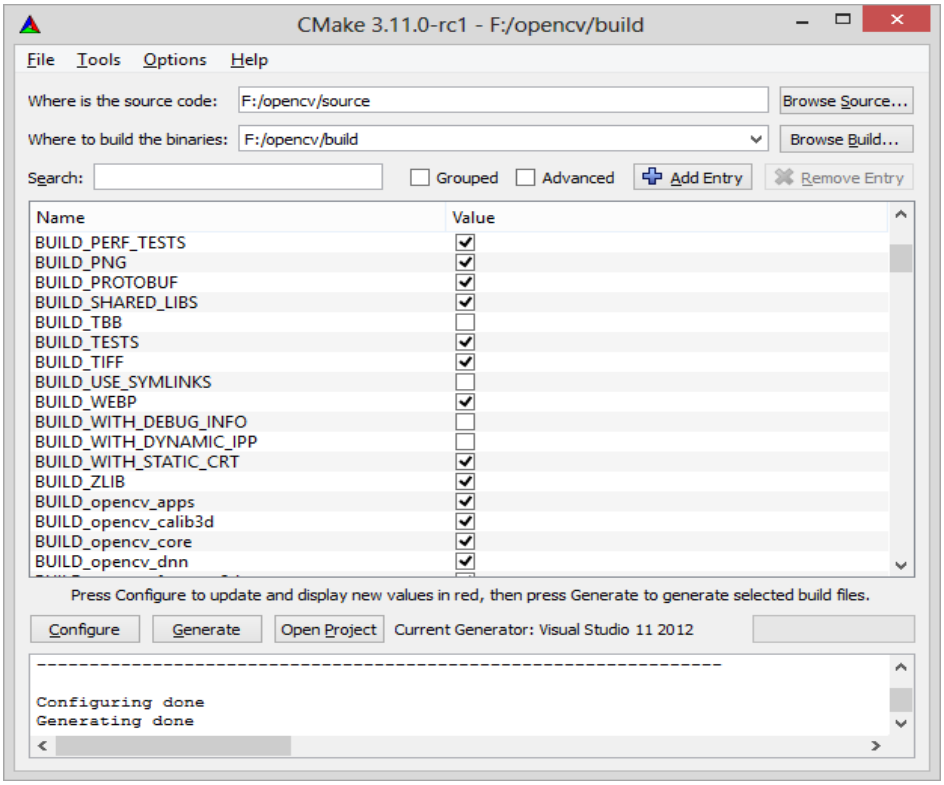
1. Click **Configure** after selecting the Visual Studio.




2. Uncheck BUILD_PERF_TESTS and BUILD_TESTS.
3. Click **Configure** till all red flag goes off.




4. Click **Generate** to create Visual Studio solution file in the OpenCV build directory.




## Step 4 - Replacing Videoio File

Replace the **videoio** folder with the folder downloaded from the [e-con's github(https://github.com/econsystems/opencv/tree/master/sources)](https://github.com/econsystems/opencv/tree/master/sources) with **OpenCV/Sources/modules/** location.

## Step 5 - Building OpenCV in Visual Studio

The steps to build OpenCV in Visual Studio are as follows:

1. Run the **OpenCV.sln** found in the build directory of OpenCV using Visual Studio.
2. Add **setupapi.lib** in modules/opencv_videoio properties tab under Linker > Input> Additional Dependencies.
3. Copy the **strmbase.lib** from the Microsoft SDKs from **c:\Program Files**\..(related to DirectShow) and paste it in the **build/lib/release** directory of OpenCV.

**Note:** Based on the x64 or x86 build of OpenCV, the strmbase.lib from the x86 or x64 should be copied.

4. Build **CMakeTargets/All Build** and **CMakeTargets/Install** separately in both the Debug/Release Configuration of the Visual Studio.

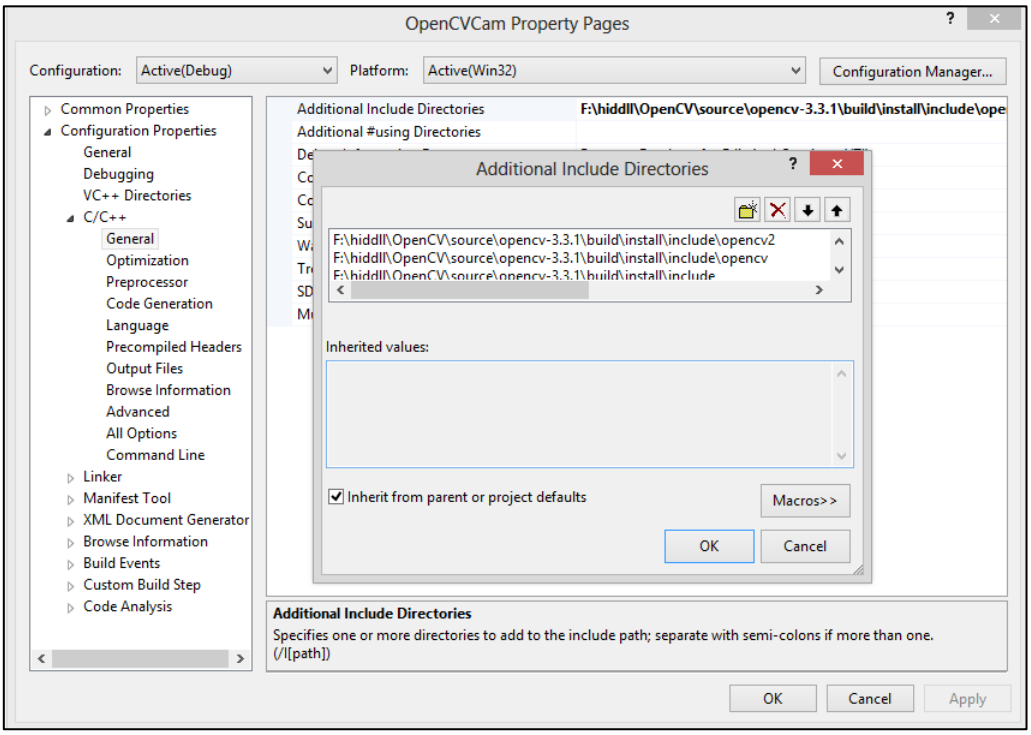
# Building Sample Code

This section describes about how to build the sample code.

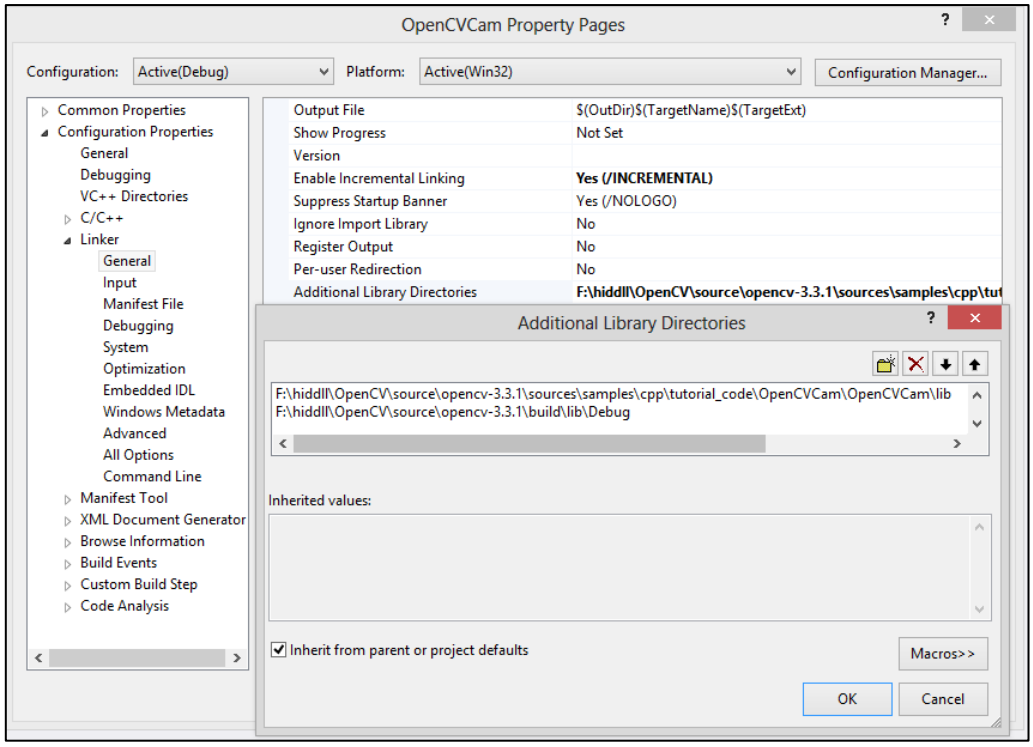
The steps to build the sample code are as follows:

1. Create a new console application in Visual Studio. Add the ***.cpp** file of the application to be built from the blog(https://github.com/econsystems/opencv/tree/master/sources/OpenCVCam) .
2. Change **Application -> Configuration Properties -> General -> project Defaults -> Character Set -> Use Unicode Character Set**
3. Link the OpenCV header under **C/C++ -> General -> Additional Include Directories files** with the following:

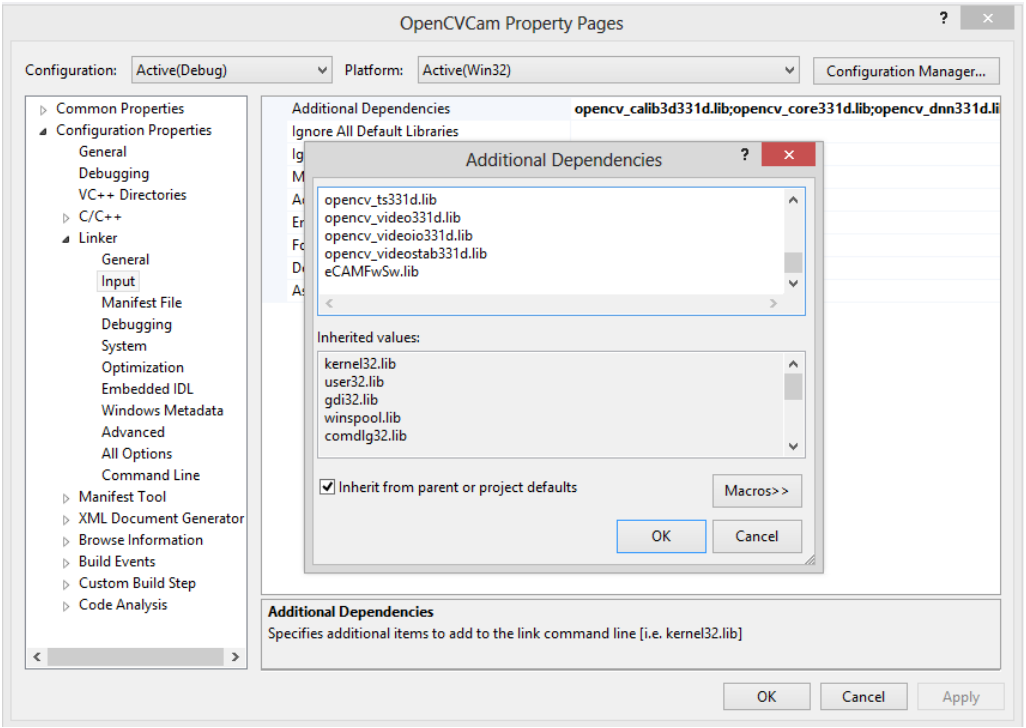
    - OpenCV/build/install/include
    - OpenCV/build/install/include/opencv
    - OpenCV/build/install/include/opencv2




4. Link the library files of OpenCV from the **OpenCV/build/lib/Release** for configuration type release under **Linker > General > Additional Library Directories**.




5. List all the library names linked to the project under **Linker > Input > Additional Dependencies** in the Visual Studio Project Property page.




6. Add runtime libraries in the sample application **root** folder from the **OpenCV/build/bin/Debug** or **OpenCV/build/bin/Release** based on the configuration of the sample application.

The runtime libraries for **OpenCV release version 3.3.1** are:

- opencv_core331.dll
- opencv_highgui331.dll
- opencv_imgcodecs331.dll
- opencv_imgproc331.dll
- opencv_videoio331.dll
- eCAMFwSw.dll

and the runtime libraries for **OpenCV Debug version 3.3.1** are:

- opencv_core331d.dll
- opencv_highgui331d.dll
- opencv_imgcodecs331d.dll
- opencv_imgproc331d.dll
- opencv_videoio331d.dll
- eCAMFwSw.dll

The runtime libraries for **OpenCV release version 3.4.1** are:

- opencv_core341.dll
- opencv_highgui341.dll
- opencv_imgcodecs341.dll
- opencv_imgproc341.dll
- opencv_videoio341.dll
- eCAMFwSw.dll

and the runtime libraries for **OpenCV Debug version 3.4.1** are:

- opencv_core341d.dll
- opencv_highgui341d.dll
- opencv_imgcodecs341d.dll
- opencv_imgproc341d.dll
- opencv_videoio341d.dll
- eCAMFwSw.dll

- Run the OpenCVCam.exe application using **Administrator** Mode.

# Troubleshooting

In this section, you can view the list of commonly occurring issues and their troubleshooting steps.

**Linker issues relating to setupdi* while building.**

Add **setupapi.lib** in the modules/opencv_videoio properties tab under **Linker> Input > Additional dependencies**.

**There is no install folder present in the opencv<version>/build/**

Build the CMakeTargets or **install project** in both Debug and Release configurations.

**HID settings are not shown in the command line application.**

Change Use Unicode Character set in the Application-> configuration properties -> General -> Project defaults -> character set

**In Opencv version 3.4.1, Opencv_test_namespace related errors while building.**

Unload the tests accuracy and tests performance projects from the opencv and start the building process again.

**IAMVIDEOCONTROL related error while building Opencv.**

Copy the **strmbase.lib,** if using release mode or strmbasd.lib, if using debug mode from the c:\Program Files\...(related to directshow) and paste it in the **build/lib/release (or) debug** directory of the OpenCV. Also based on the x86 or x64 architecture, the libs should be copied and pasted. If the strmbase.lib or strmbasd.lib is not present. Then build the **baseclasses.sln** using visual studio which will be present in the **c:\ProgramFiles\Microsoft SDKs\.....** If baseclasses.sln file is not present, Install Microsoft SDK properly in the system.

# Support

**Contact Us**

If you need any support on OpenCV sample application, please contact us using the Live Chat option available on our website - https://www.e-consystems.com/

**Creating a Ticket**

If you need to create a ticket for any type of issue, please visit the ticketing page on our website - https://www.e-consystems.com/create-ticket.asp

**RMA**

To know about our Return Material Authorization (RMA) policy, please visit the RMA Policy page on our website - https://www.e-consystems.com/RMA-Policy.asp

**General Product Warranty Terms**

To know about our General Product Warranty Terms, please visit the General Warranty Terms page on our website - https://www.e-consystems.com/warranty.asp

# Revision History

| Rev | Date | Description | Author |
|---|---|---|---|
| 1.0 | 10-Apr-18 | Initial Draft | Chandra Sekar. V |
| 1.1 | 27-Jul-18 | Added support for the FPS variant in OpenCV for Windows | Chandra Sekar. V |
| | | | |
| | | | |
| | | | |
| | | | |