OpenCV

Windows Installation Manual

e-con Systems

Your Product Development Partner

Version 1.6

e-con Systems

8/15/2020

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).

  Using git

  Click on code option and copy the URL

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

  Direct Download

  For 3.3.1 (https://github.com/opencv/opencv/archive/3.3.1.zip),
  For 3.4.1 (https://github.com/opencv/opencv/archive/3.4.1.zip).

- 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 2017.

Description

The following steps have been tested on Windows 10. It relatively works on other versions 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.

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.

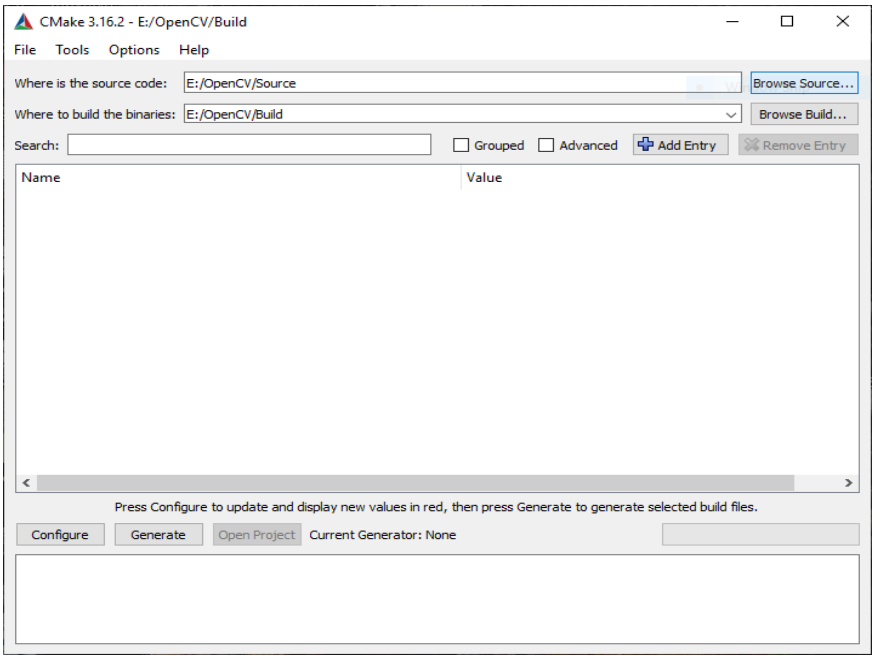


Figure 1: CMake Source and Build Directory Specification Window

Step 2 – Visual Studio Version Selection

A window prompting to select current Visual Studio version (VS2017) in your PC and x32 and x64 version appears. Select the appropriate options as shown below.

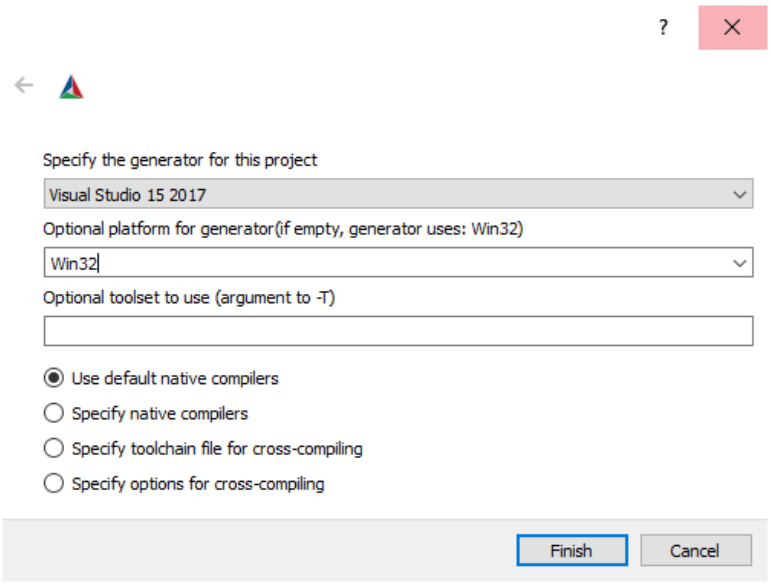
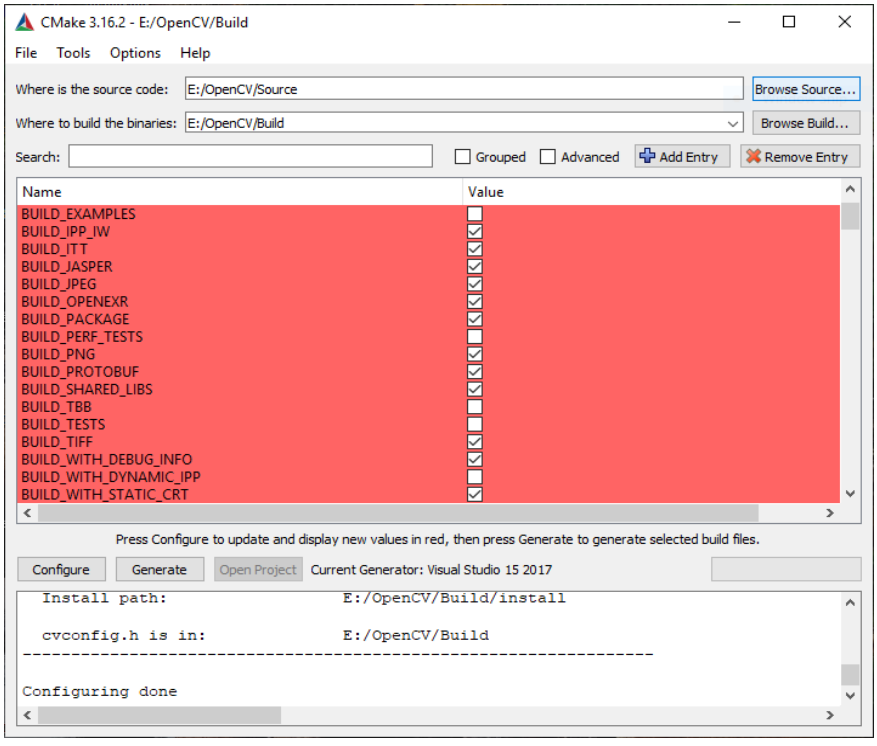


Figure 2: Visual Studio Version selection in CMake

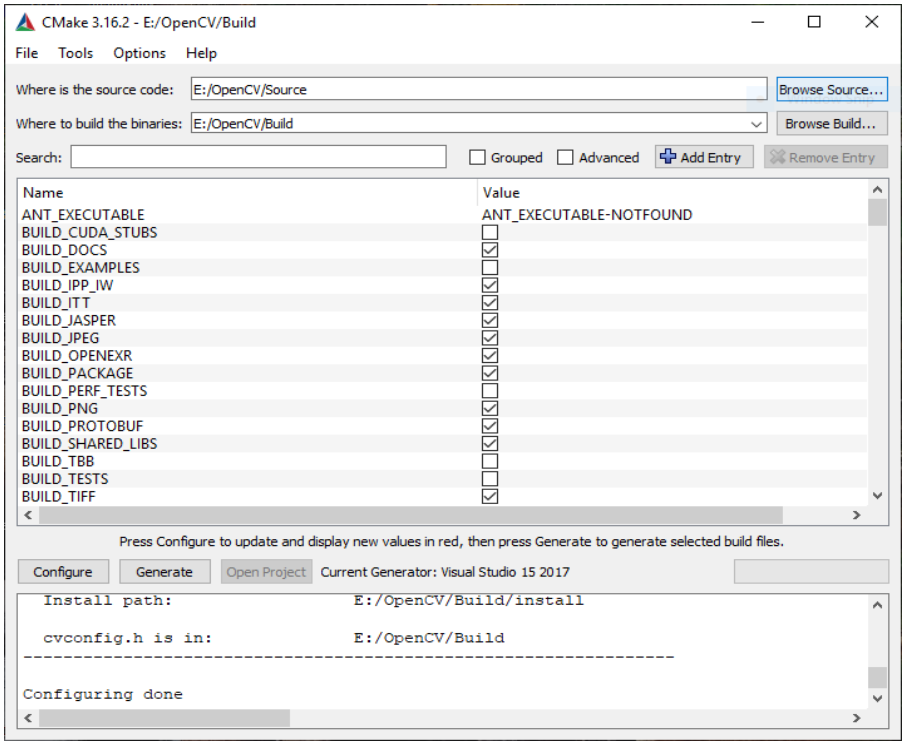
Step 3 – Configure and Generate CMake

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. Check the **BUILD_opencv_world** to build single binary, including all the modules instead of a collection of separate binaries.
4. Click **Configure** till all red flag goes off.



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

Step 4 – Videoio File Replace

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

Step 5 – Building BaseClasses

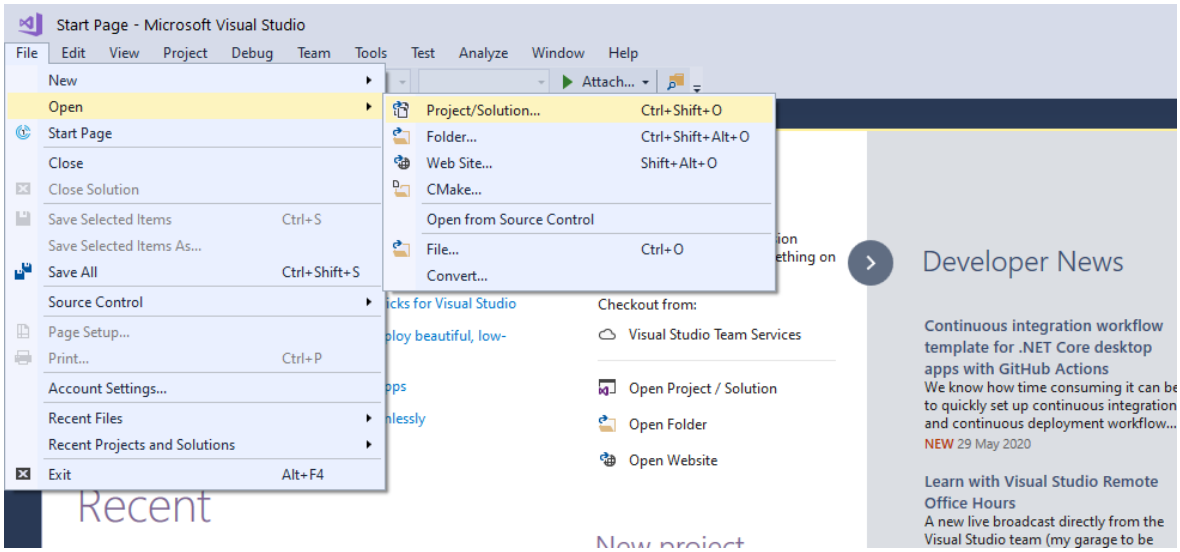
The prebuild binaries of the BaseClasses Library is also available in the below link. The BaseClasses library can either be build or referenced as directly. (https://github.com/econsystems/opencv/tree/master/Binary/Lib)

To Build BaseClasses library files:

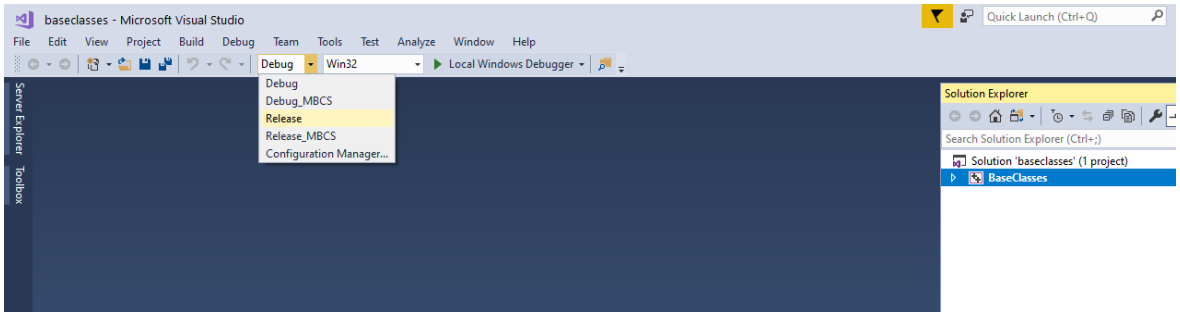
Download BaseClasses project e-con's GitHub (https://github.com/econsystems/opencv/tree/master/Source)

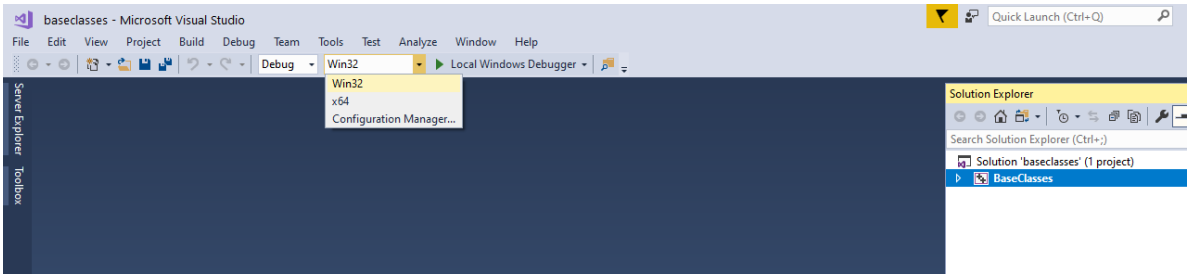
1. Open and Configuring the **baseclasses** project:

   a) Open the new instance of visual studio (2017).
   b) Click **File-> Open -> Project/Solution**

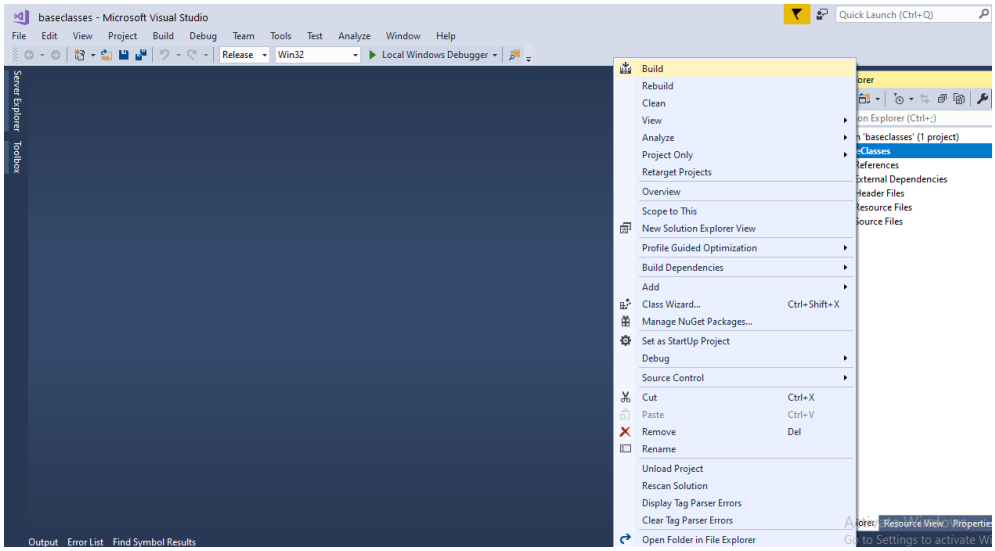


c) Browse the baseclasses project and select **baseclasses.sln**.
d) Choose Solution configuration (Debug / Release) and Solution Platform (Win32 or x64) (based on your requirement).

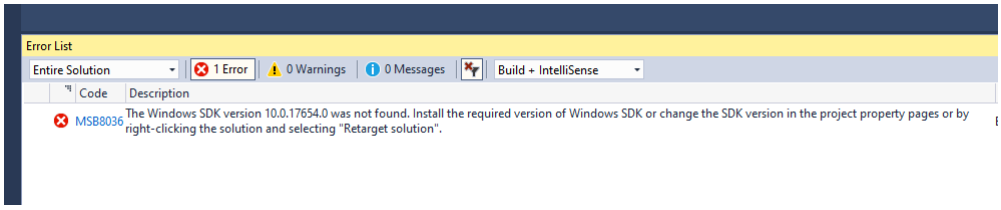




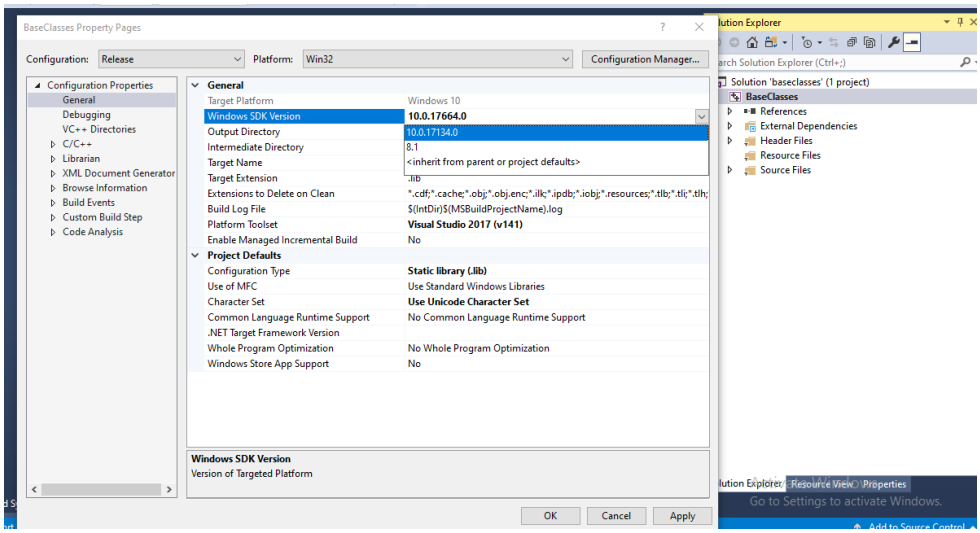
2. Build the baseclasses project:
   a) Right Click on baseclassess solution and select Build Solution (or Rebuild Solution).



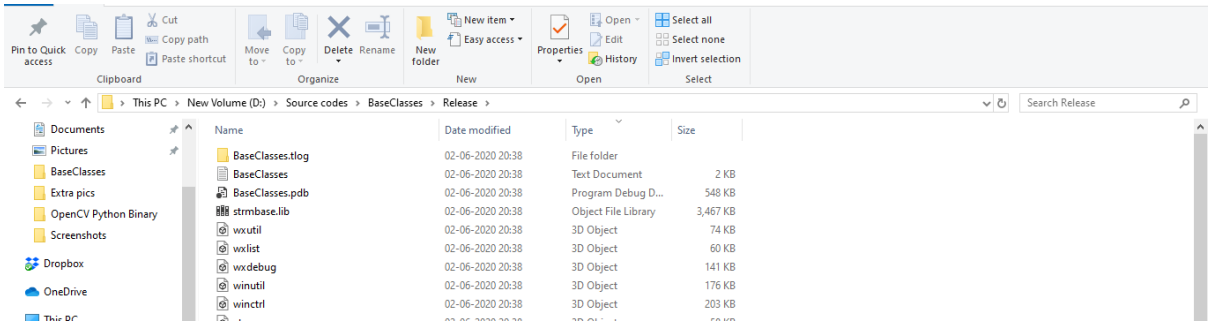
b) While building the solution, if you get the following error change the Configuration Properties as stated below.

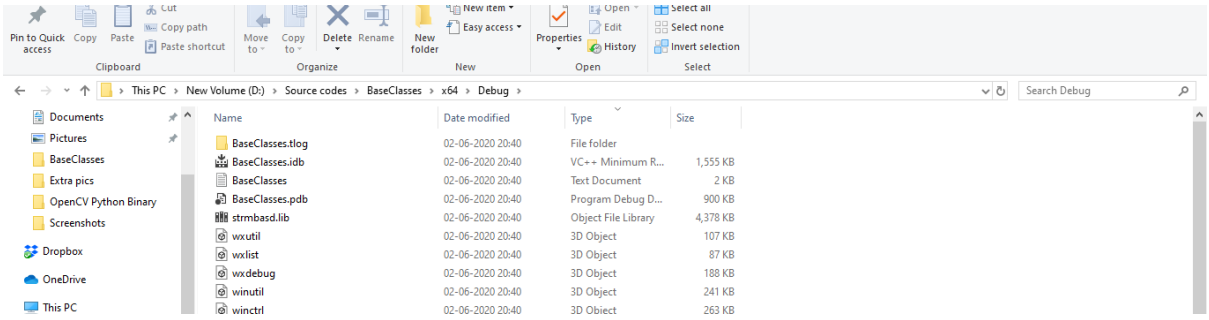


c) **BaseClasses -> Properties -> Configuration Properties -> Genaral -> Windows SDK Version** choose available SDK version.



d) Build the solution.
e) After Building you can find the strmbase.lib (for Release), strmbasd.lib (for Debug).





f) The **strmbase.lib** / **strmbasd.lib** has to be linked in the OpenCV Project.

Step 6 – Build 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_world properties tab under Linker > Input> Additional Dependencies.
3. Add **strmbase.lib** for **release** mode and **strmbasd.lib** for **debug** mode from the **BaseClasses** solution directory, in modules/opencv_world properties tab under **Linker->General->Additional Library directories** and mention the lib name in **Linker->Input->Additional dependencies**.
4. Build **CMakeTargets/All Build** and **CMakeTargets/Install** separately in both the Debug/Release Configuration of the Visual Studio.

Building Y8GrabberFilter

NOTE:

1. Y8Grabber only supports Y8 formats.
2. This Grabber also give Output as YUY2 format.
3. If you want to do any modifications in the Y8Grabber code, you can follow the steps to build the Y8Grabber.

Already Provided the binaries of Y8GrabberFilter (32 and 64 bit) you can make use of it.

(https://github.com/econsystems/opencv/tree/master/Binary/Y8GrabberFilter **)**

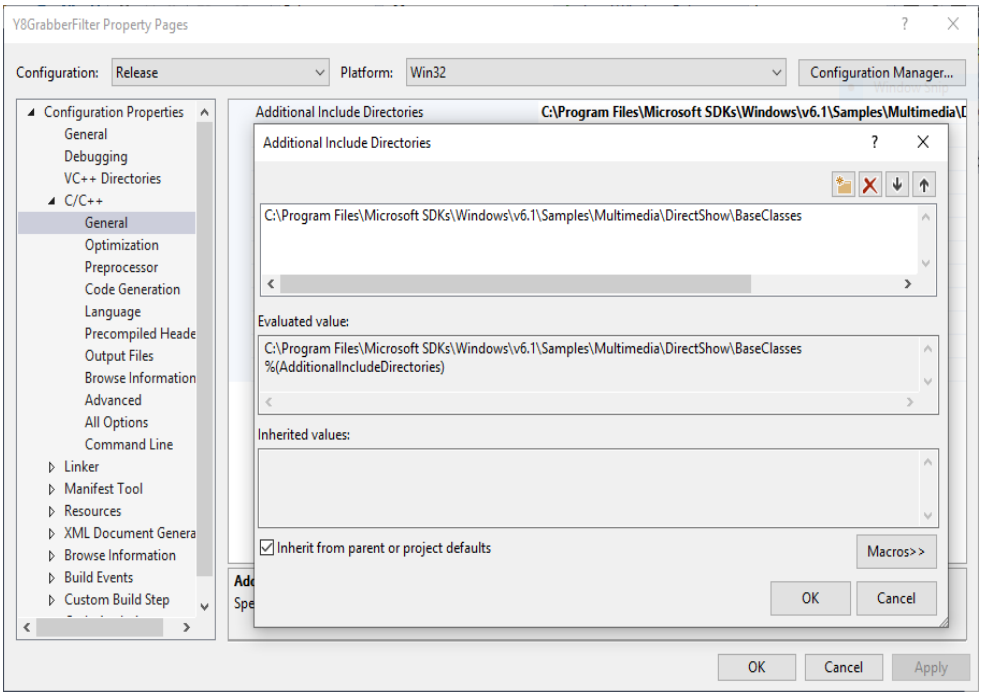
Register Y8GrabberFilter.dll

1. Open the command prompt in administrator mode.
2. Go to the Y8GrabberFilter.dll (32 or 64 bit) path (using the cd command).
3. Register the Y8GrabberFilter.dll (with command **regsvr32 Y8GrabberFilter.dll**).
4. After clicking the enter you can see the following message box.

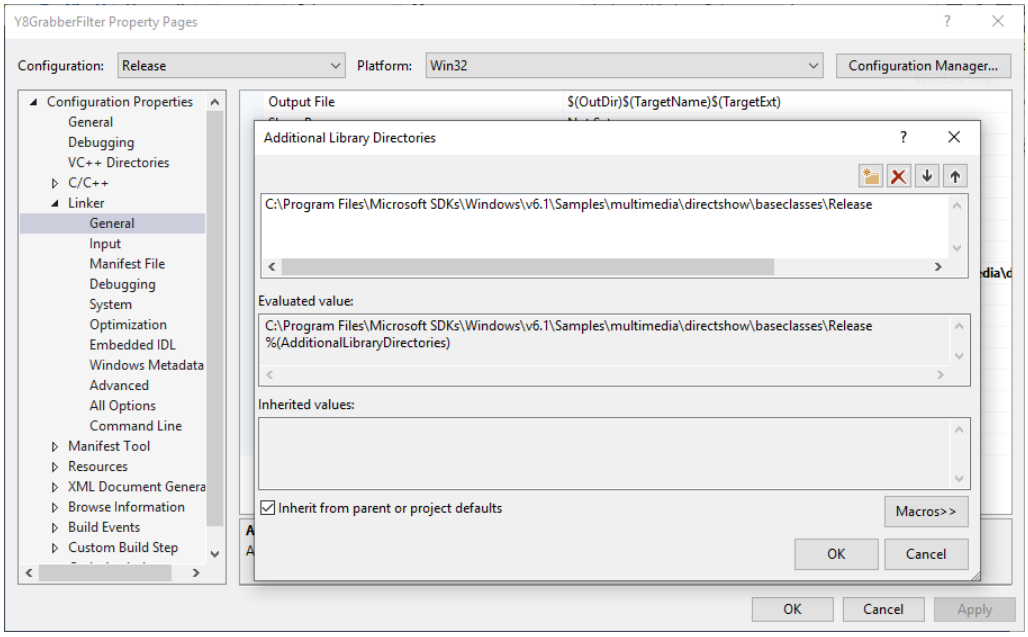
Build Y8GraberFilter in Visual Studio

1. Open the Y8GrabberFilter.sln (Provided in https://github.com/econsystems/opencv/tree/master/Source/Y8GrabberFilter) in visual studio VS2017.
2. Before building the Y8GrabberFilter you should build the **BaseClasses** project or you can use the provided strmbase.lib (or strmbasd.lib) files in the package.

3. Add the **BaseClasses** solution directory in Y8GrabberFilter properties tab under **C/C++->General->Additional Include directories**.



4. Add **strmbase.lib and winmm.lib** for **release** mode and **strmbasd.lib and winmm.lib** for **debug** mode from the **BaseClasses** solution directory, in Y8GrabberFilter properties tab under **Linker->General->Additional Library directories** and mention the lib name in **Linker->Input->Additional dependencies**.



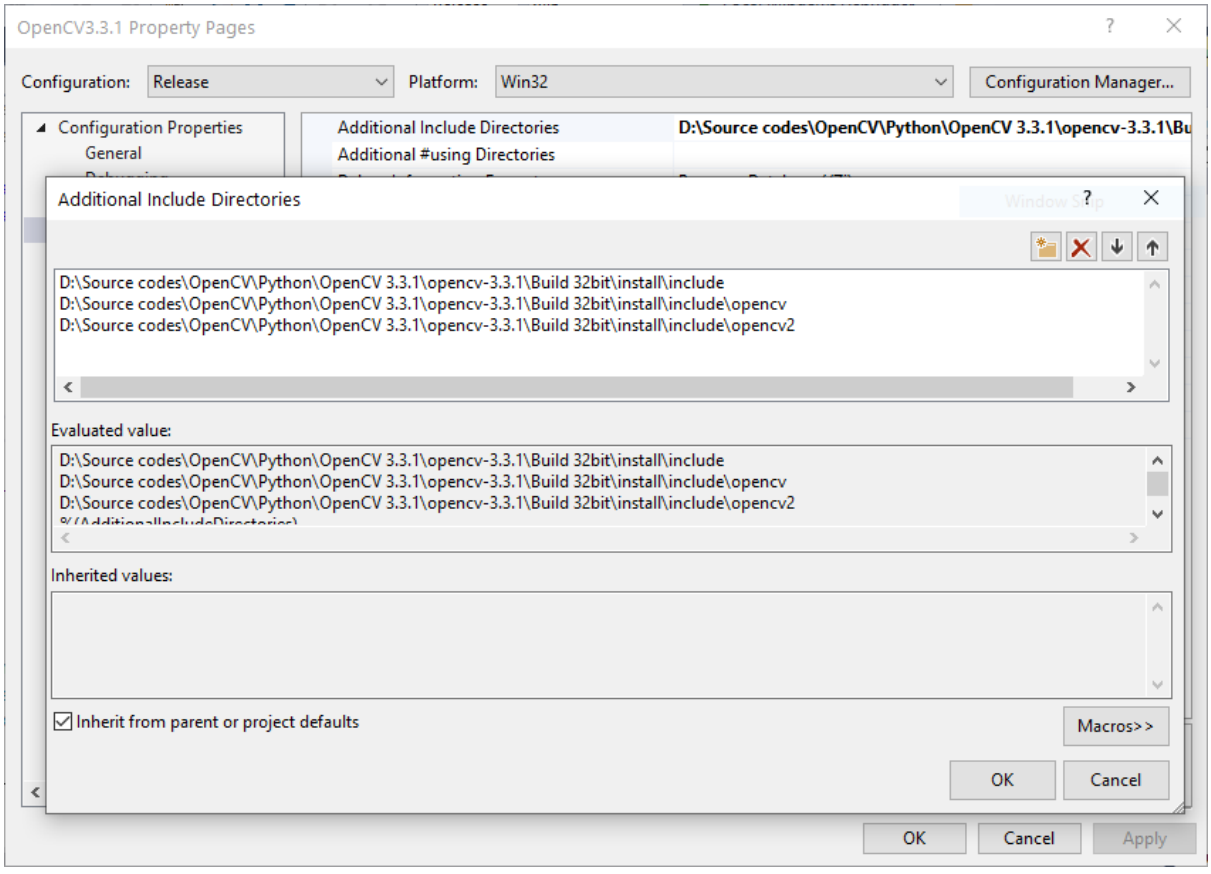
5. Build the Y8GrabberFilter project and you can find the binaries in respective configurations (Release or Debug).

Building Sample Code

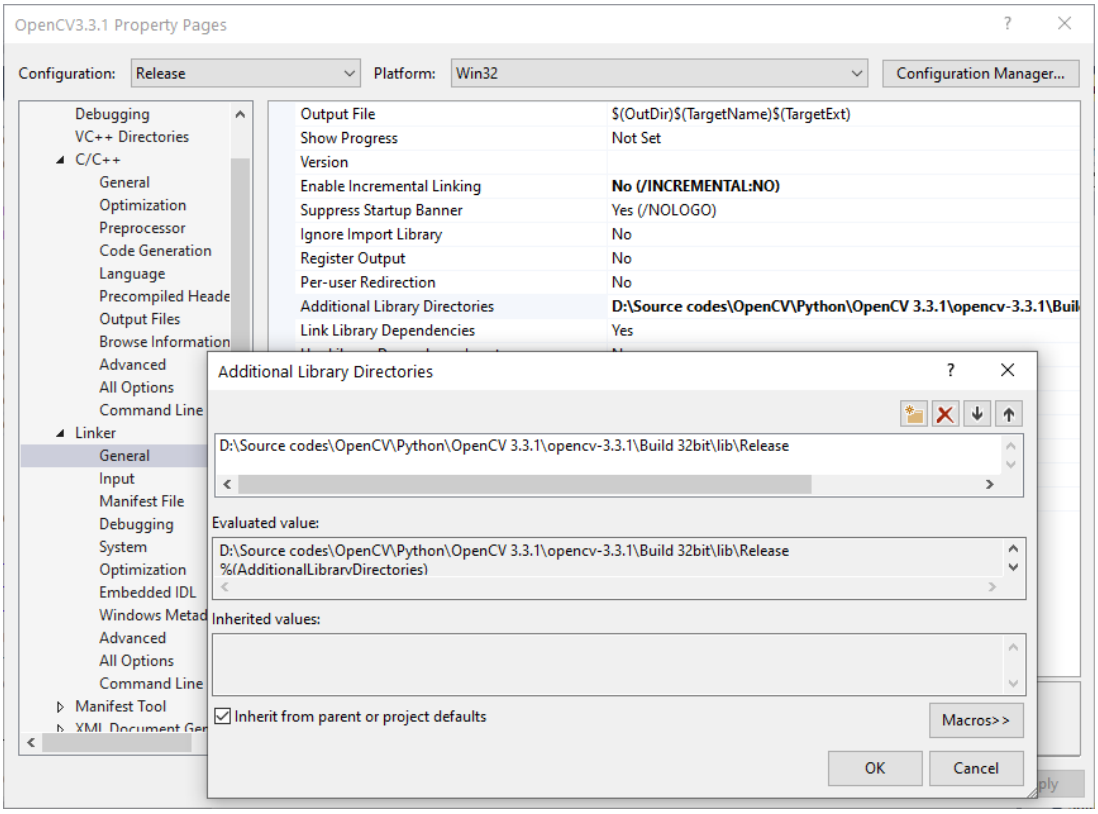
This section describes about how to build the sample code.

The steps to build the sample code are as follows:

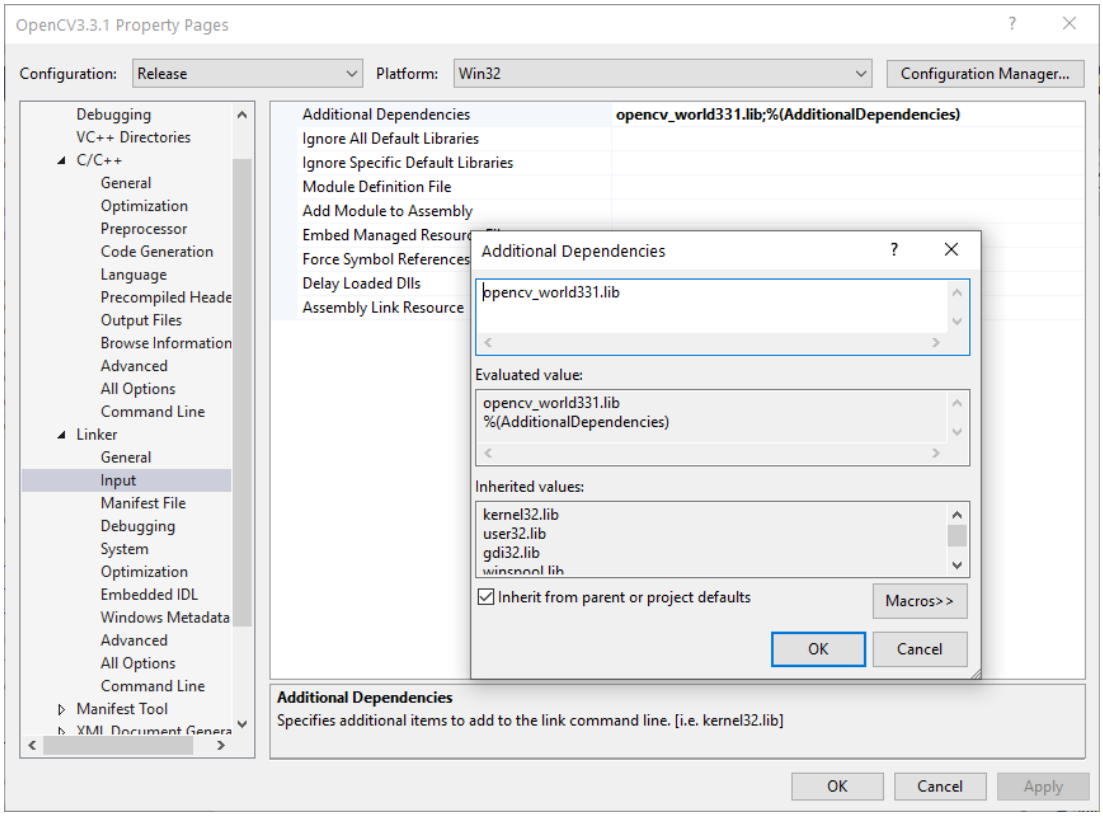
1. Create a new console application in Visual Studio.
2. Copy the OpenCVCam source code in to the ***.cpp** file in the application. (https://github.com/econsystems/opencv/tree/master/Source/OpenCVCam).
3. Change **Application -> Configuration Properties -> General -> project Defaults -> Character Set -> Use Unicode Character Set**
4. 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



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



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



7. 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.
8. **"eCAMFwSw.dll"** is provided in the package binary folder with respected configuration (x86 or x64).

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

- opencv_world331.dll
- eCAMFwSw.dll

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

- opencv_world331d.dll
- eCAMFwSw.dll

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

- opencv_world341.dll
- eCAMFwSw.dll

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

- opencv_world341d.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_world 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 BaseClasses project path 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 2017.

If git is not recognized as an internal or external command error

You need to download and install the git exe from internet with the following link.
https://gitforwindows.org/

Support

Contact Us

If you need any support on OpenCV product, 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	06-April-2018	Initial Draft	Chandra Sekhar. V
1.1	27-July-2018	Added support for the FPS variant in OpenCV for windows	Chandra Sekhar. V
1.2	02-June-2020	Added steps to build and generate base class libraries.	Murali Mohan. M
1.3	12-June-2020	Added Installation Process for Y8Grabber and Python in windows.	Murali Mohan. M
1.4	25-July-2020	Added changes in build procedure.	Murali Mohan. M
1.5	13-August-2020	Removed the Python installation steps.	Murali Mohan. M
1.6	15-August-2020	Added changes.	Murali Mohan. M