# How to Install OpenCV on Your PC

ELEC 474 – Machine Vision

Fall 2019

**Introduction:**

This document describes step-by-step instructions to install OpenCV on a PC. The process described is derived from https://www.deciphertechnic.com/install-opencv-with-visual-studio/.

While there are many different versions of OpenCV that run on different operating systems and IDEs, we will focus here on OpenCV 4.1.1 running C++ on Windows10 using VisualStudio19. This is the version that will be used throughout the lab, and all software is available without cost. You are certainly free to use other operating systems or IDEs, but this configuration is the most commonly used. *TAs will exclusively provide support for the configuration described in this document. Note that versions of OpenCV prior to release 4.0.0 may have different compilation instructions, capabilities and syntax.*

**Prerequisites:**

The assumptions are that you are running Windows 10, and that you have suitable privileges on your PC to install software and change some environment variables.

**Step 1- Install Visual Studio:**

1) Download and install Visual Studio Community 2019 from:

   https://visualstudio.microsoft.com/vs/community/

2) When prompted, add the "Desktop development with C++" Workload.

3) The complete installation will require ~7 GB of disk space. If you need to save space, you can uncheck all of the optional installation packages, and reduce the footprint to ~2 GB.

4) Reboot your computer when prompted.

**Step 2 - Install OpenCV:**

1) Download the Windows release of OpenCV 4.1.1 from: https://opencv.org/releases/

2) Save and run install file opencv-4.1.1-vc14_vc15

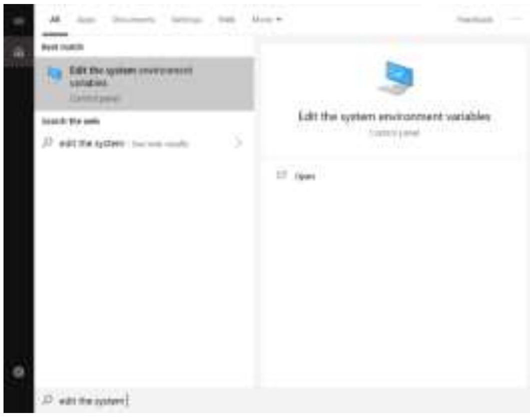
3) Select location to install ('C:\' is preferred)

   ***Do not* install to path with spaces!** *Ex: C:\Program Files*

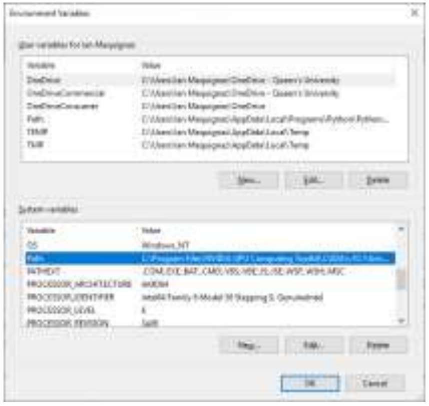
4) Choose “yes to all” when prompted

### Step 3: Add OpenCV to system variables path

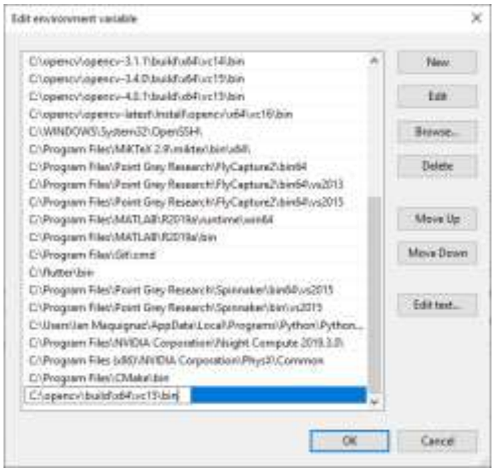
1) From the start menu, search for ‘*Edit the system environment variables*’



2) Under the ‘*Advanced*’ tab, select ‘*Environment Variables*’



3) Under ‘*System variables*’, double-click the ‘*Path*’ variable
4) Click on ‘*New*’ to create a variable and insert the path of *bin* folder inside OpenCV package. The path should be similar to *C:\opencv\build\x64\vc15\bin*. ***Copy-paste the path to ensure it’s correct!***



```
C:\opencv\opencv-3.1.1\build\x64\vc14\bin
C:\opencv\opencv-3.4.0\build\x64\vc15\bin
C:\opencv\opencv-4.0.1\build\x64\vc15\bin
C:\opencv\opencv-latest\install\opencv\x64\vc16\bin
C:\WINDOWS\System32\OpenSSH\
C:\Program Files\MiKTeX 2.9\miktex\bin\x64\
C:\Program Files\Point Grey Research\FlyCapture2\bin64
C:\Program Files\Point Grey Research\FlyCapture2\bin64\vs2013
C:\Program Files\Point Grey Research\FlyCapture2\bin64\vs2015
C:\Program Files\MATLAB\R2019a\runtime\win64
C:\Program Files\MATLAB\R2019a\bin
C:\Program Files\Git\cmd
C:\flutter\bin
C:\Program Files\Point Grey Research\Spinnaker\bin64\vs2015
C:\Program Files\Point Grey Research\Spinnaker\bin\vs2015
C:\Users\Ian Maquignaz\AppData\Local\Programs\Python\Python...
C:\Program Files\NVIDIA Corporation\Nsight Compute 2019.3.0\
C:\Program Files (x86)\NVIDIA Corporation\PhysX\Common
C:\Program Files\CMake\bin
C:\opencv\build\x64\vc15\bin
```

5) Press ‘*OK*’ and exit the environment variable dialog by clicking ‘*OK*’ again

6) Restart your computer to let the updated Path variable take effect.

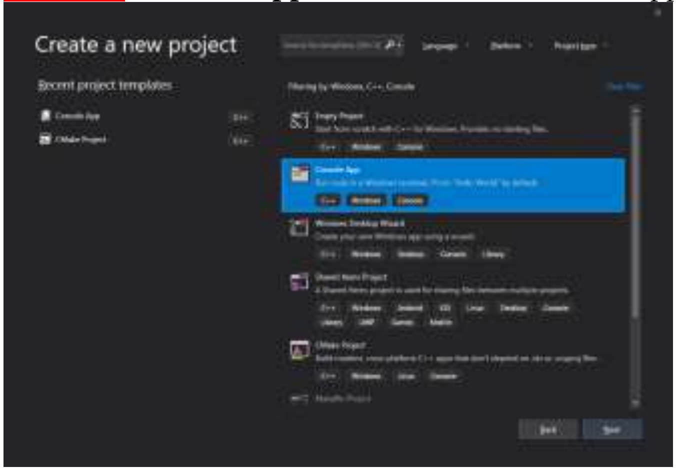
**Step 4: Create a new empty console application**

**[Optional]** Download the OpenCV-Visual Studios template project from OnQ. It should work out of the box if you installed OpenCV to ‘C:\’. If it fails to build, debug by following the instructions below from step 7 onward.

1. Launch Visual Studio 2019.
   1.1. If you are prompted to login, use your Queen’s credentials.

2. Select ‘*Create a new project*’
   2.1. If prompted, under ‘*Development Settings:*’, choose ‘*Visual C++*’
      2.1.1. Select ‘*Start Visual Studio*’

3. Select ‘*Console App*’ and hit ‘*Next’*. **Make sure you select the project template for a *Windows* Console Application.** Linux ‘*Console App*’ may be an option on your system…



4. Give the project a name such as “OpenCV_HelloWorld” and select a location for it to be saved (the default is OK)

**Warning!**
* It is not recommended to save projects to a path with spaces or special characters. It might work, but it also might not. Example *C:\Users\Bob the OpenCV-Builder*
*** Note that Visual Studios has a ‘*feature*’ where files with a path longer than 260 characters cannot be compiled.**
* For the reason above, **DO NOT save your project to the default OneDrive folder.**

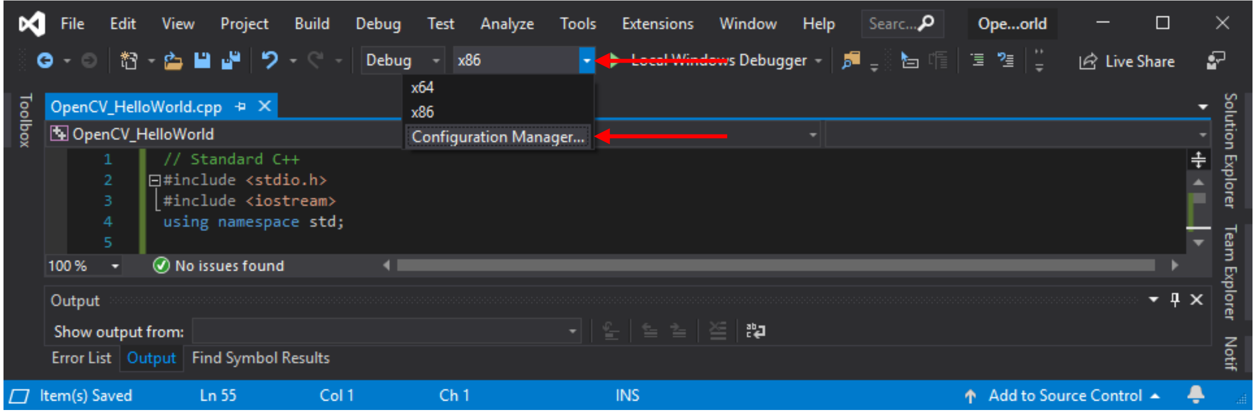
5. Select ‘*Create*’ to populate your project

6. Replace the automatically generated “Hello World” code in ‘*main*.cpp’ with the following:

```
// Standard C++
#include <stdio.h>
#include <iostream>
using namespace std;

// OpenCV Imports
#include <opencv/cv.hpp>
#include <opencv2/opencv.hpp>
#include <opencv2/core/core.hpp> // OpenCV Core Functionality
#include <opencv2/highgui/highgui.hpp> // High-Level Graphical User Interface


// [Optional] Use OpenCV namespace
// NOTE! If not included, all OpenCV functions will need to be prepended with "cv::"
// EX: "Mat image;" -> "cv:Mat image;"
using namespace cv;

int main(int argc, char** argv)
{
        printf("hello world\n");
        cout << "Hello World!" << "\t" << "HELLO WORLD!!" << endl;

        // Create an image
        cv::Mat image;

        // Create 100x100 image of zeros
        image = cv::Mat::zeros(100, 100, CV_16UC3);

        // Import an image from a JPG
        // Requires you to save an image named “image.jpg” inside your project.
        //image = cv::imread("image.jpg", 0);

        if (!image.data)
        {
                printf("No image data \n");
                return -1;
        }


        std::cout << image.cols << "," << image.rows << endl;

        // [Optional] Create a display window
        cv::namedWindow("Display Window", CV_WINDOW_AUTOSIZE);

        // Display the image
        cv::imshow("Display Window", image);
        // Wait for a keystroke in the window
        // NOTE! If you don't add this the window will close immediately!
        cv::waitKey();

        // Terminate the program
        return 0;
}
```

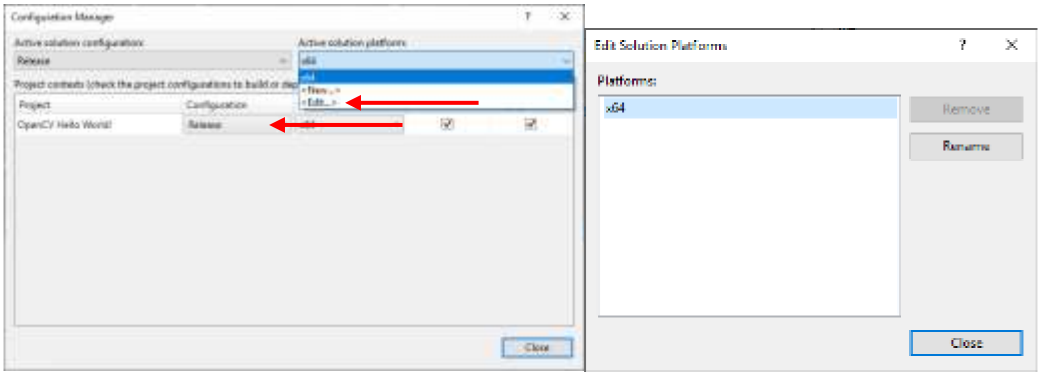
7. OpenCV 4.1.1 will only compile on x64 platforms.

7.1. To remove other platforms auto-populated by visual studios, select the arrow to the right of '*x64*' and select '*Configuration Manager*' from the drop-down list.

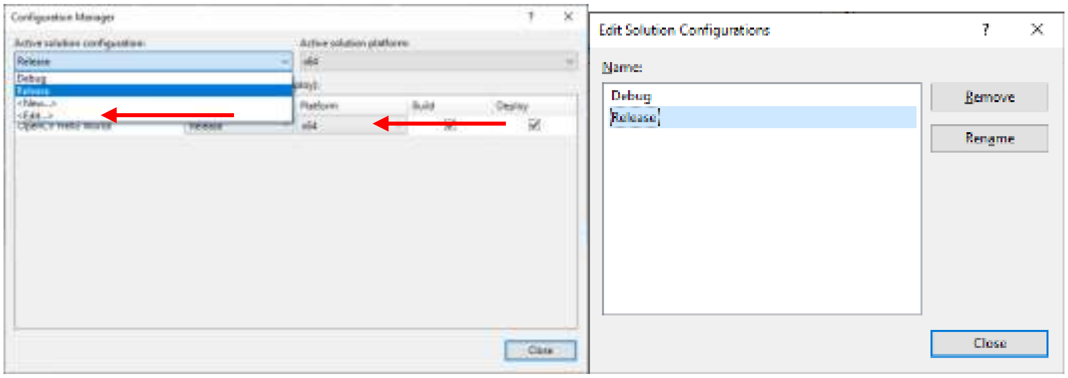


```
// Standard C++
#include <stdio.h>
#include <iostream>
using namespace std;
```

7.2. Select the dropdown below '*Active solution platform*' and click 'Edit'. Remove all Solution Platforms except for 'x64'. Do the same for the dropdown below '*Configuration*'.



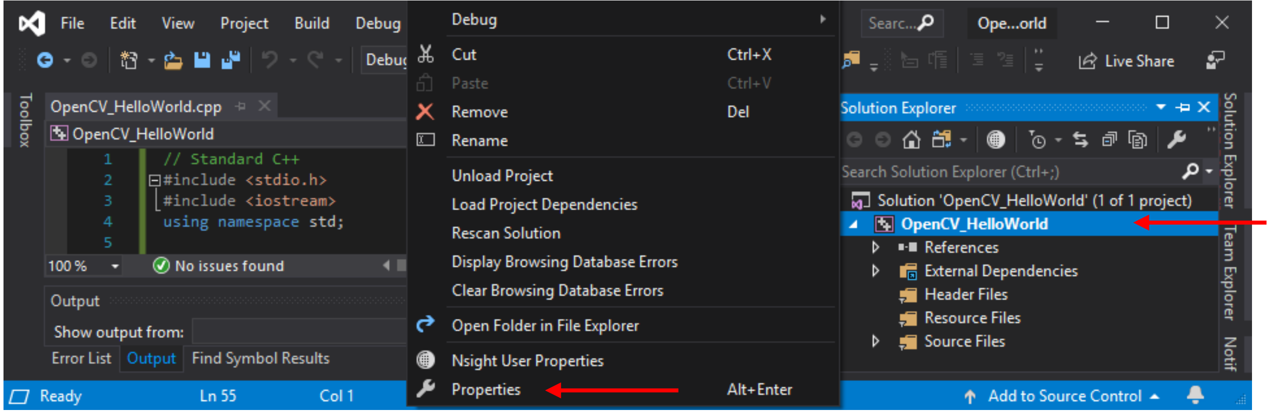
7.3. Select the dropdown below '*Active solution configuration*' and click 'Edit'. Remove all Solution Configurations except for 'Debug' and 'Release'. Do the same for the dropdown below '*Platform'*



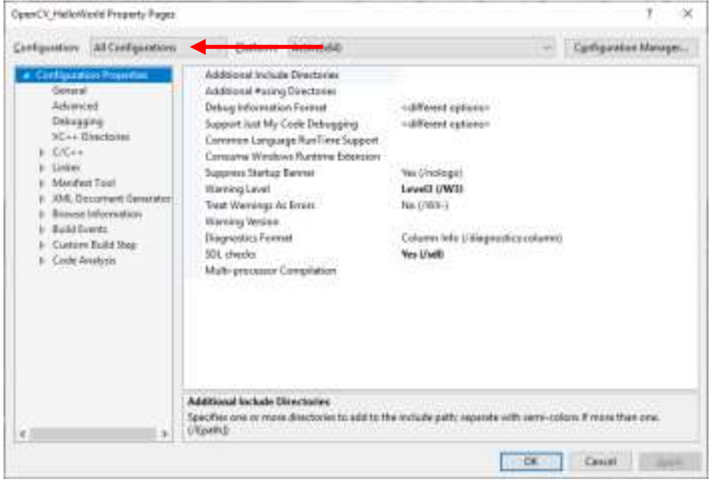
7.4. Close the ‘*Configuration Manager*’ dialogue box

8. Linking Project Configurations to OpenCV

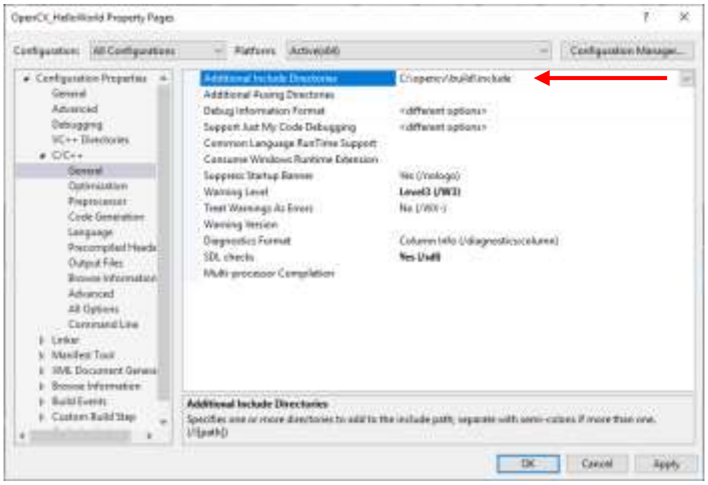
8.1. From the ‘*Solution Explorer*’, right-click the ‘*Solution*’ and select ‘*Properties*’ from the drop-down menu.



8.2. Select the drop-down list right of ‘*Configuration*’ and set the configuration to ‘*All Configurations*’



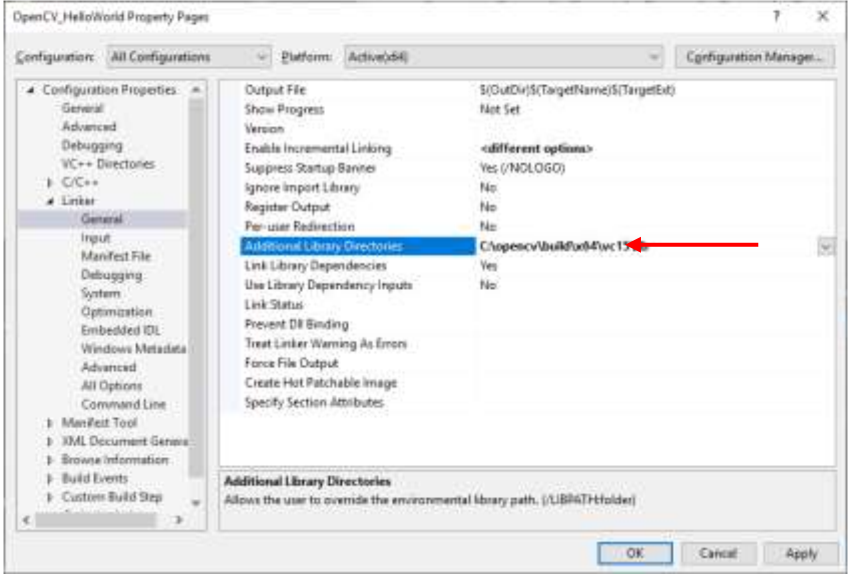
8.3. Go to *Configuration Properties* → *C/C++* → *General*. Copy the path to the include folder of OpenCV and paste it inside ‘*Additional Include Directories*’. The path will look similar to *C:\opencv\build\include*. Then, click Apply.



Additional Include Directories C:\opencv\build\include
Additional #using Directories
Debug Information Format <different options>
Support Just My Code Debugging <different options>
Common Language RunTime Support
Consume Windows Runtime Extension
Suppress Startup Banner Yes (/nologo)
Warning Level **Level3 (/W3)**
Treat Warnings As Errors No (/WX-)
Warning Version
Diagnostics Format Column Info (/diagnostics:column)
SDL checks **Yes (/sdl)**
Multi-processor Compilation

**Additional Include Directories**
Specifies one or more directories to add to the include path; separate with semi-colons if more than one. (/I[path])

8.4. Go to *Configuration Properties* → *Linker* → *General*. Copy the path to the folder containing the OpenCV lib files and paste it inside ‘*Additional Library Directories*’. The path will look similar to *C:\opencv\build\x64\vc15\lib*. Then, click Apply.



Output File $(OutDir)$(TargetName)$(TargetExt)
Show Progress Not Set
Version
Enable Incremental Linking **<different options>**
Suppress Startup Banner Yes (/NOLOGO)
Ignore Import Library No
Register Output No
Per-user Redirection No
Additional Library Directories **C:\opencv\build\x64\vc15**
Link Library Dependencies Yes
Use Library Dependency Inputs No
Link Status
Prevent Dll Binding
Treat Linker Warning As Errors
Force File Output
Create Hot Patchable Image
Specify Section Attributes

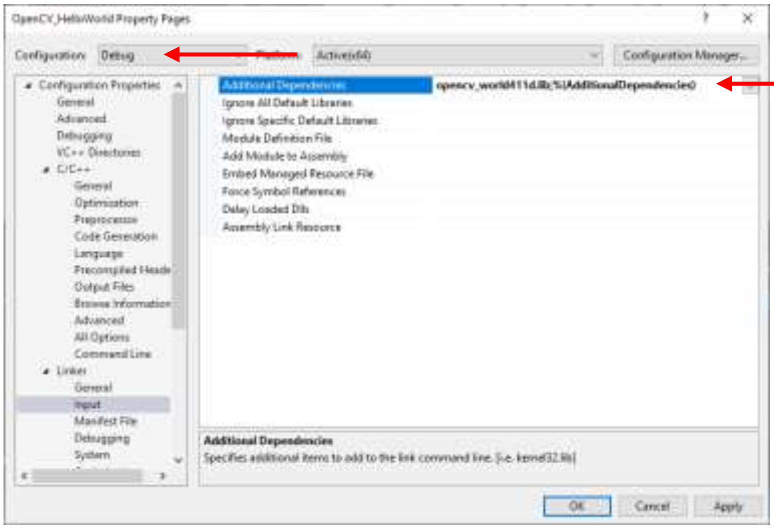
**Additional Library Directories**
Allows the user to override the environmental library path. (/LIBPATH:folder)

8.5. Go to *Configuration Properties* → *Linker* → *Input*. The settings on this page must be set independently for Debug and Release configurations:

8.5.1. Set ‘Configuration’ to ‘***Debug***’.

8.5.1.1. Edit ‘*Additional Dependencies*’ and paste the Debug OpenCV lib file name. For ‘Debug’ this lib should be ‘*opencv_world411d.lib*’

8.5.1.2. Click ‘*Apply*’


OpenCV_HelloWorld Property Pages

Configuration: Debug

Platform: Active(x64)

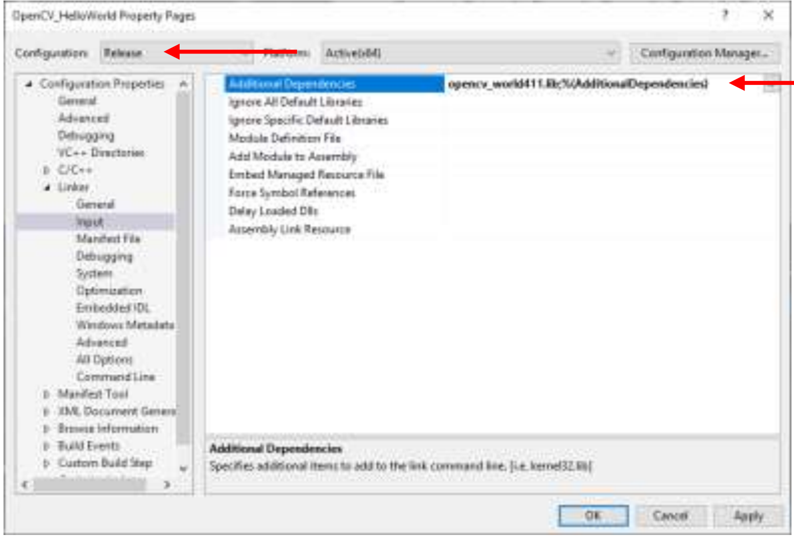
Additional Dependencies: opencv_world411d.lib;%(AdditionalDependencies)

Additional Dependencies
Specifies additional items to add to the link command line. [i.e. kernel32.lib]

8.5.2. Set ‘Configuration’ to ‘Release’

8.5.2.1. Edit ‘Additional Dependencies’ and paste the Release OpenCV lib file name. For ‘Release’ this lib should be ‘opencv_world411.lib’

8.5.2.2. Click ‘*Apply*’


OpenCV_HelloWorld Property Pages

Configuration: Release

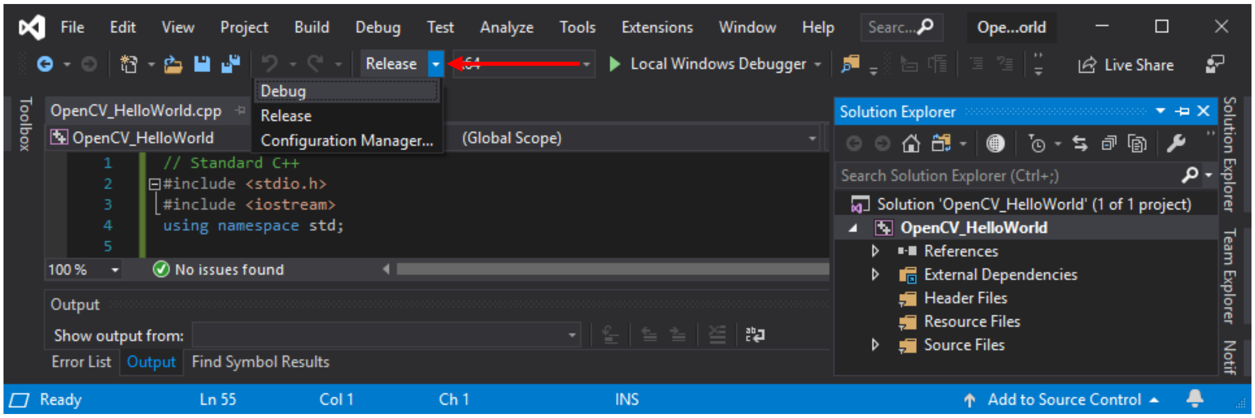
Platform: Active(x64)

Additional Dependencies: opencv_world411.lib;%(AdditionalDependencies)

Additional Dependencies
Specifies additional items to add to the link command line. [i.e. kernel32.lib]

8.5.3. Click ‘*Apply*’ and close the dialogue box.

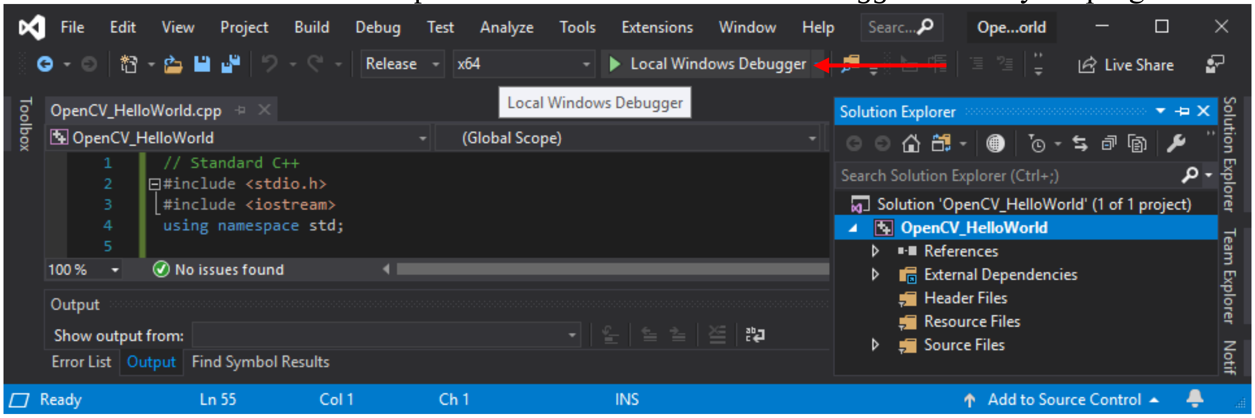
***Note!*** What’s the difference between ‘*Debug*’ and ‘*Release*’ configurations? ‘*Release*’ is a lot faster, but if you make a mistake, the error message (if any) will probably be useless. When using the ‘*Debug*’ configuration, error messages may have helpful information that will speed up solving the problem. Select between ‘Debug’ and ‘Release’ by using the drop-down menu shown in the image below.



```
// Standard C++
#include <stdio.h>
#include <iostream>
using namespace std;
```

8.6. Exit the Properties by clicking ‘*OK*’.

9. Your installation should be complete! Click ‘*Local Windows Debugger*’ to run your program.



```
// Standard C++
#include <stdio.h>
#include <iostream>
using namespace std;
```