1. Installing & Compile OpenCV with GPU Support

Installation of Required Driver and Programs

Installation involves several steps which are,

1- Graphics Card driver installation

2- IDE installation (Visual Studio Community)

3- CUDA and cuDNN installation

4- CMake installation

5- OpenCV installation

Graphics Card Driver Installation

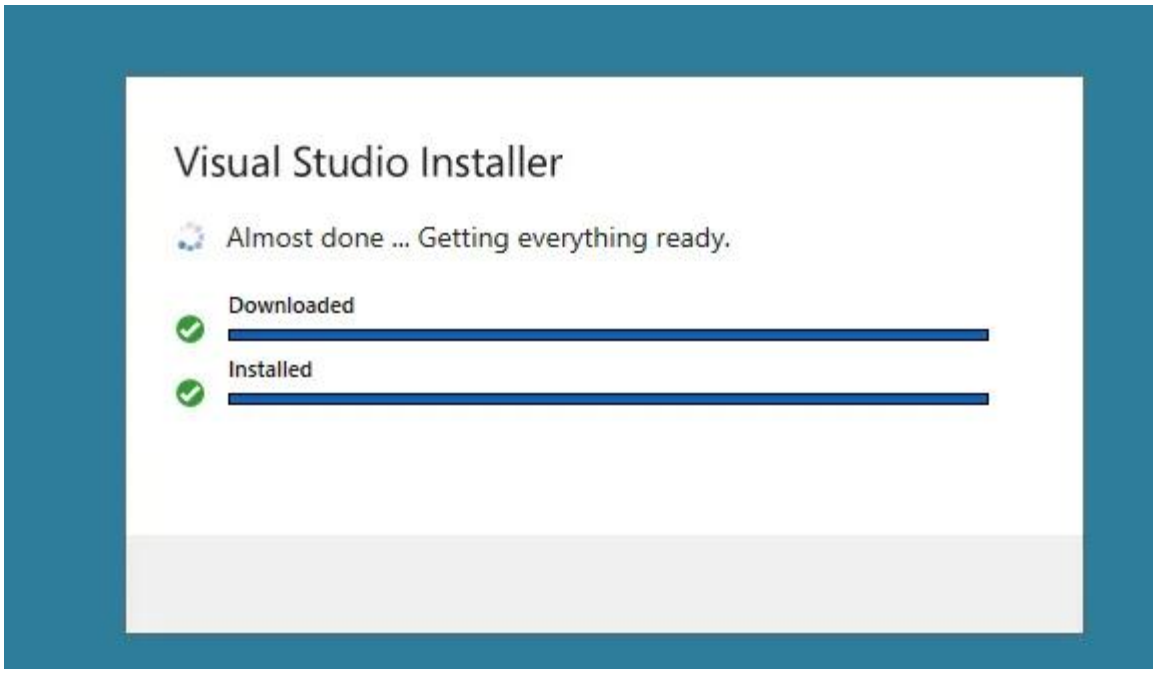
You can select the graphics card that you own and download the drivers from https://www.nvidia.com/Download/index.aspx . In this step, there are no important actions that you need to take. Just download the executable file and run it to install the driver.

IDE Installation

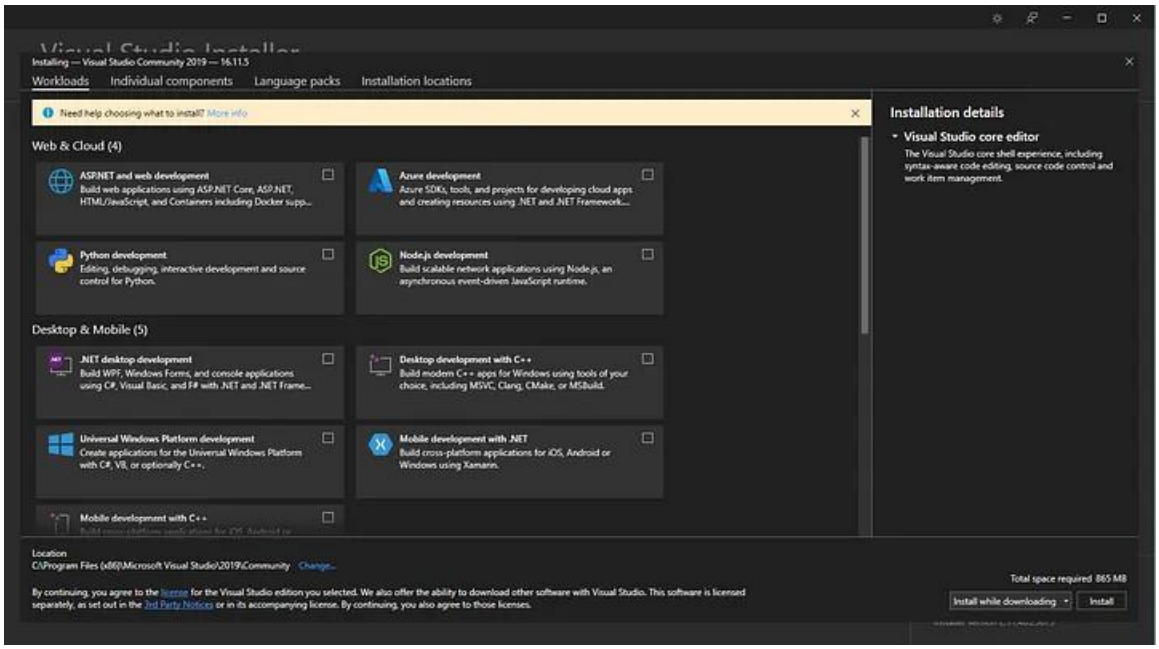
As for the IDE, Visual Studio 2019 Community will be used. It is free and it provides lots of useful tools. You can download it https://visualstudio.microsoft.com/vs/community/.

Run the executable file to proceed.

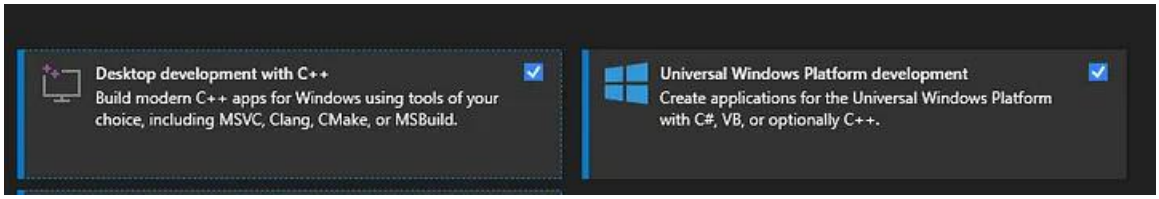
You will see that the installer is getting things ready.



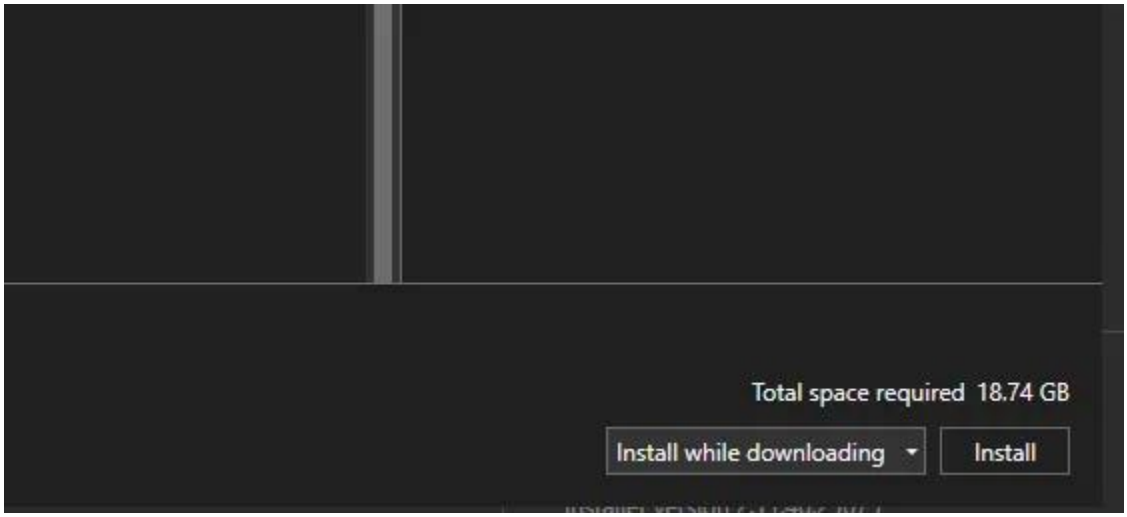
A menu that will allow you to select to which components to install will appear.



You need to select the components we need for C++ development.



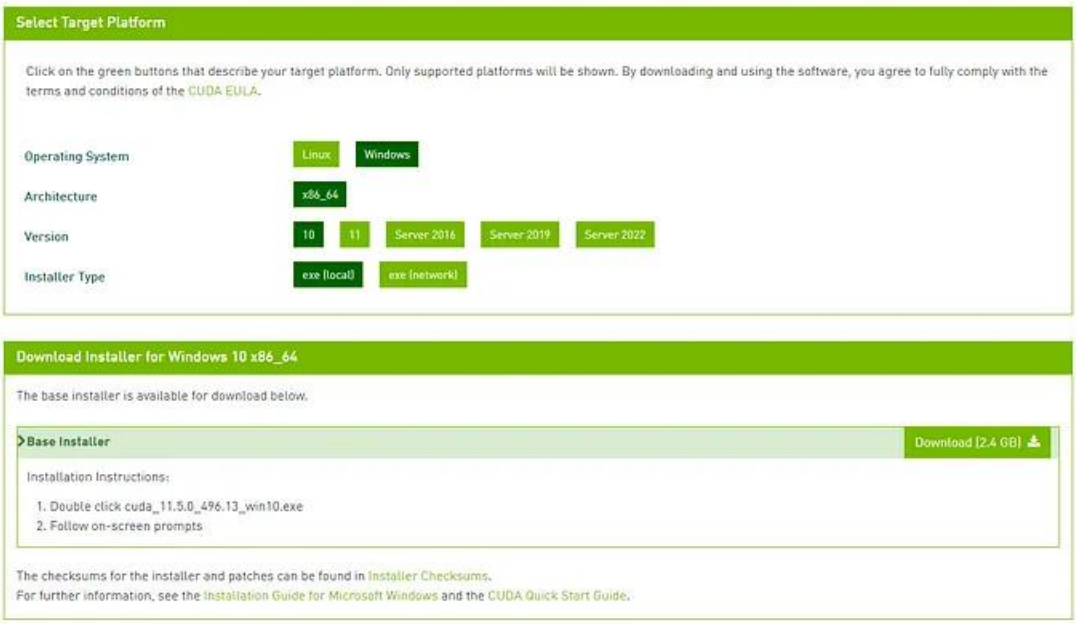
Then click the “Install” button at the bottom right corner.



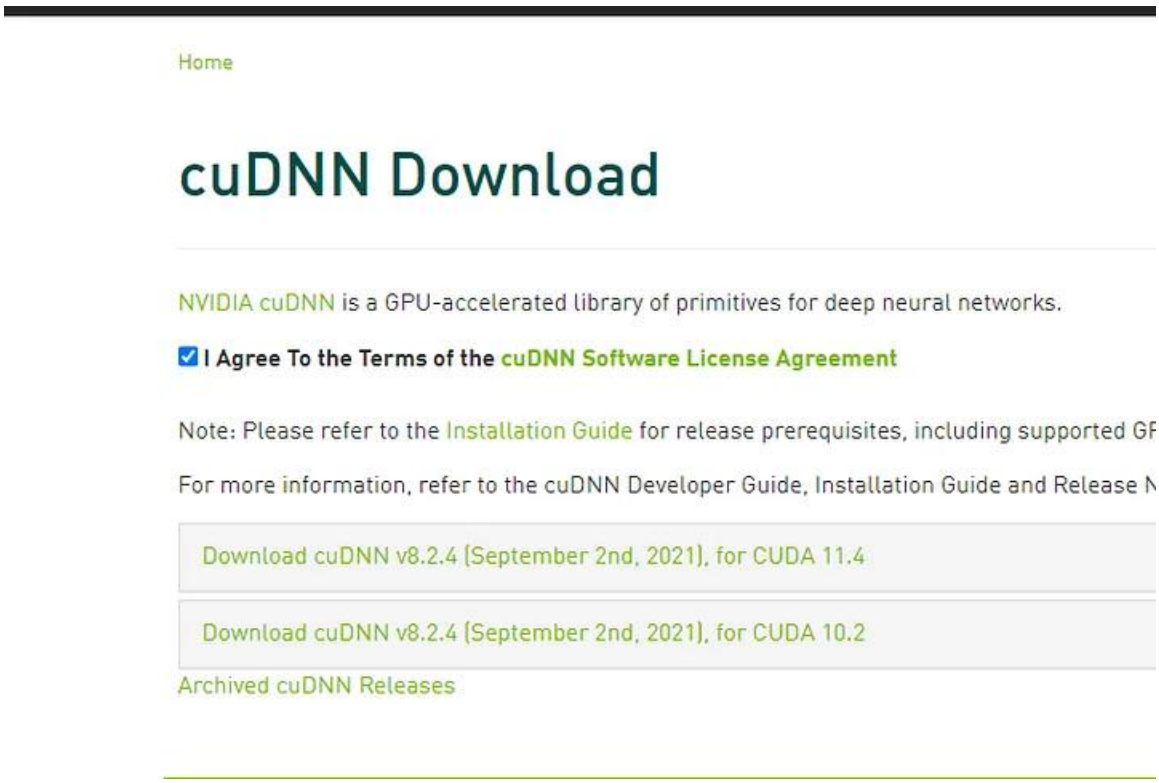
After installation is complete, you need to restart your computer.

<u>CUDA and cuDNN installation</u>

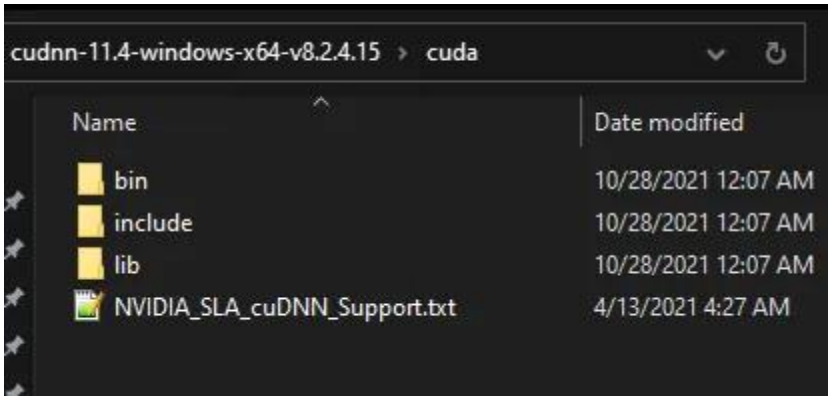
You can select the CUDA version that you want to install and download the executable from https://developer.nvidia.com/cuda-downloads. In this step, there are no important actions that you need to take. Just download the executable file and run it to install the CUDA Toolkit.



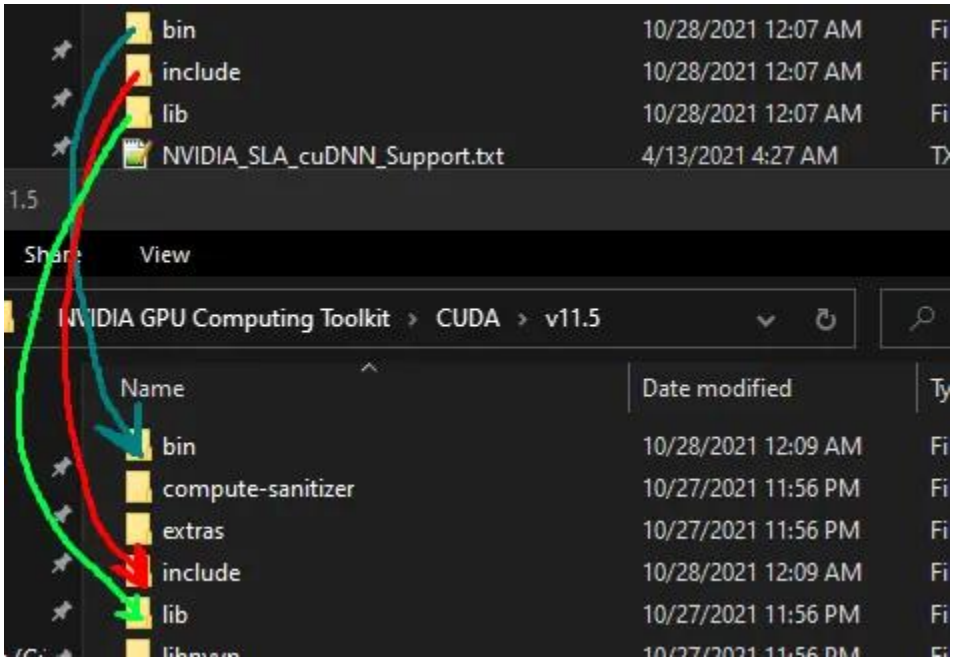
Now, download cuDNN from https://developer.nvidia.com/rdp/cudnn-download.



Extract the file and you'll see some subfolders.

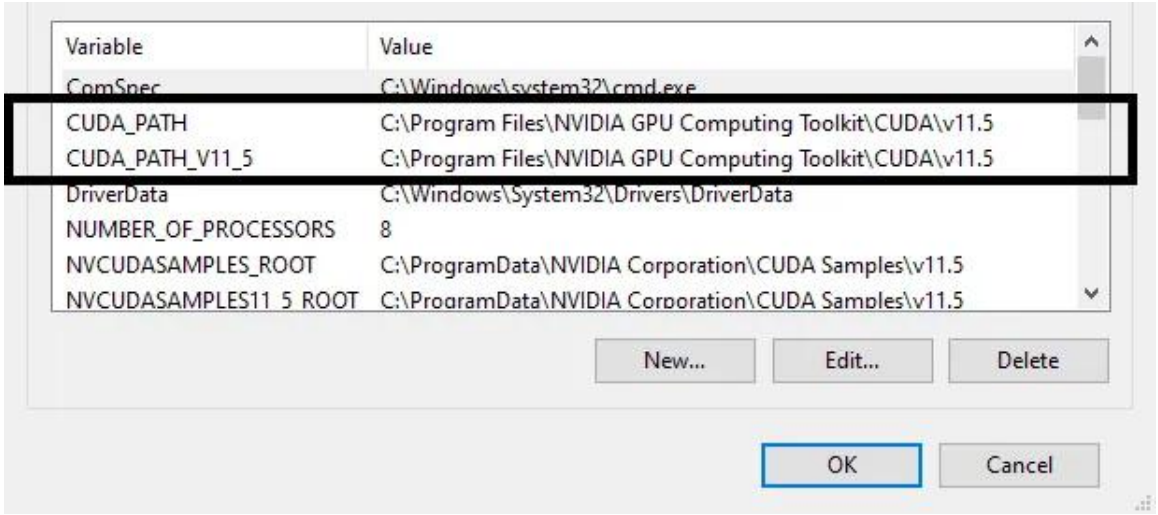


Copy the contents of each folder into corresponding folders at the CUDA folder respectively.



Finally, check if the CUDA folder was added to the path or not. Go to the environment variables options by typing env into the Windows search box and click “Edit the system environment variables”.

If everything was right, you’ll have CUDA_PATH and CUDA_PATH_V11_5 successfully created. This final step ends the CUDA and cuDNN section of this guide.



CMake Installation

You can download the CMake version that you want to install from https://cmake.org/download/. In this step, there are no important actions that you need to take. Just download the executable file and run it to install the CMake.

Platform	Files
Unix/Linux Source (has \n line feeds)	cmake-3.22.0-rc1.tar.gz
Windows Source (has \r\n line feeds)	cmake-3.22.0-rc1.zip

Binary distributions:

Platform	Files
Windows x64 Installer: **Installer tool has changed. Uninstall CMake 3.4 or lower first!**	cmake-3.22.0-rc1-windows-x86_64.msi
Windows x64 ZIP	cmake-3.22.0-rc1-windows-x86_64.zip
Windows i386 Installer: **Installer tool has changed. Uninstall CMake 3.4 or lower first!**	cmake-3.22.0-rc1-windows-i386.msi
Windows i386 ZIP	cmake-3.22.0-rc1-windows-i386.zip
macOS 10.13 or later	cmake-3.22.0-rc1-macos-universal.dmg

OpenCV Installation

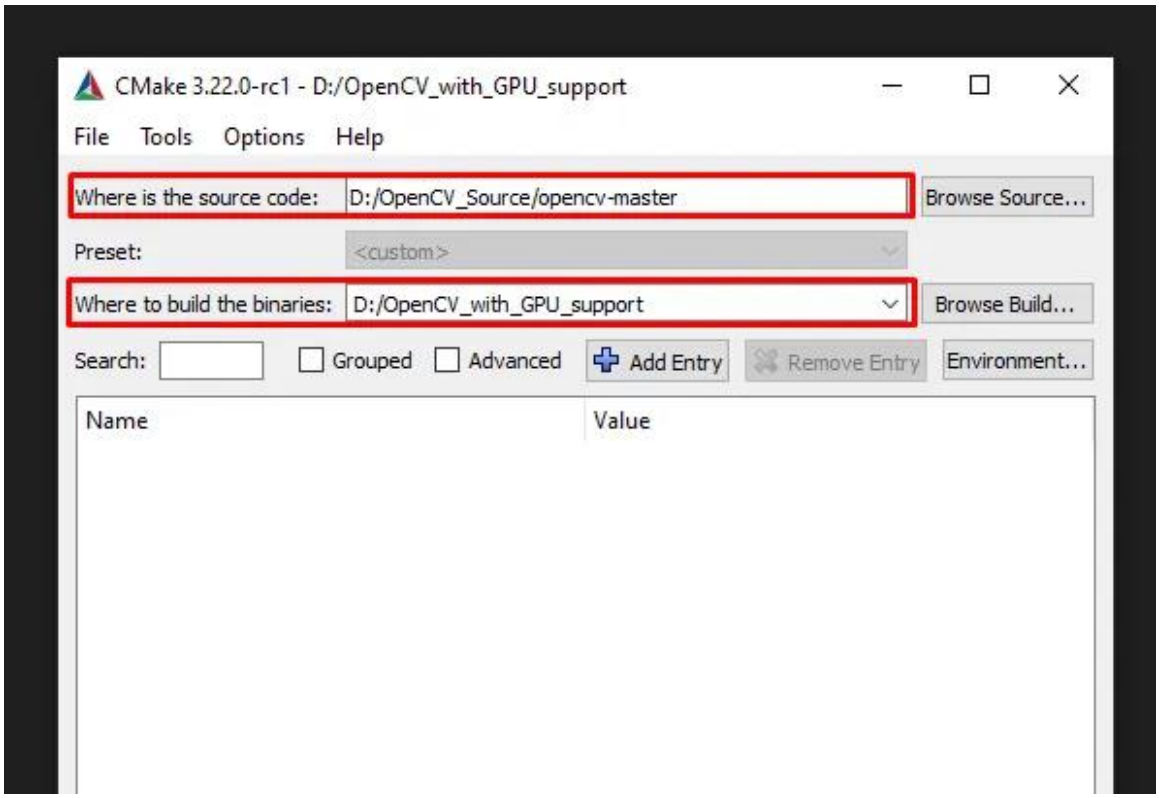
In fact, there is no installation, we will compile the OpenCV from the source and generate library files (.dll and .lib) to use in Visual Studio.

Download the OpenCV 4.5.4 from https://github.com/opencv/opencv/archive/4.5.4.zip

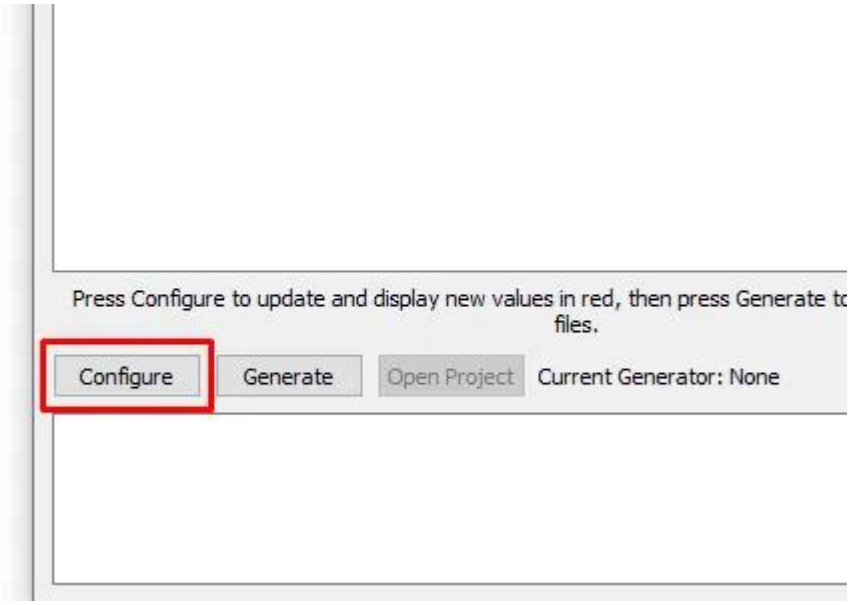
Download the extra modules which contain the GPU module from https://github.com/opencv/opencv_contrib

After the downloads are complete, extract those two files into a proper directory which we will use for the next steps.

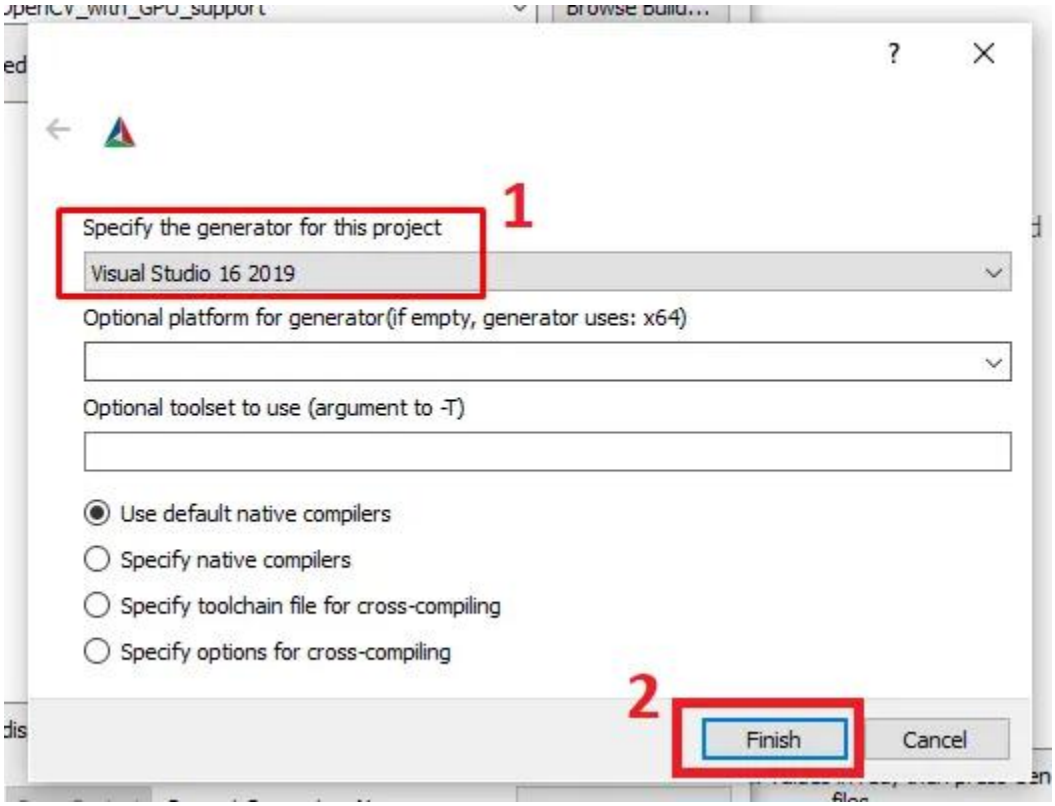
Start CMake, copy and paste the path of opencv-master folder. Do the same thing for the path that you want to extract compiled library.



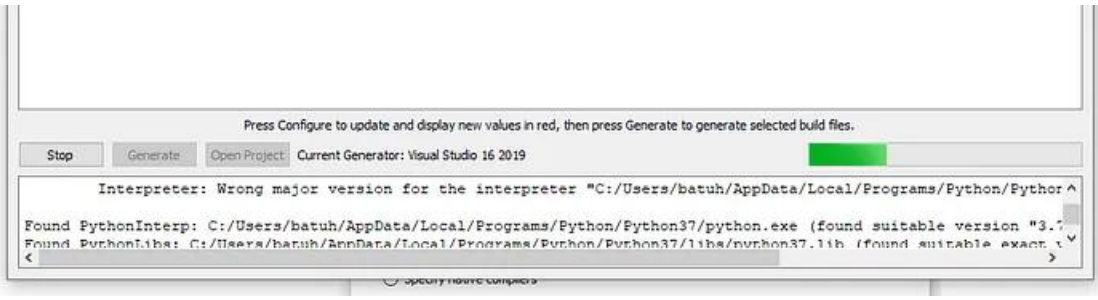
2. Press “Configure” to continue.

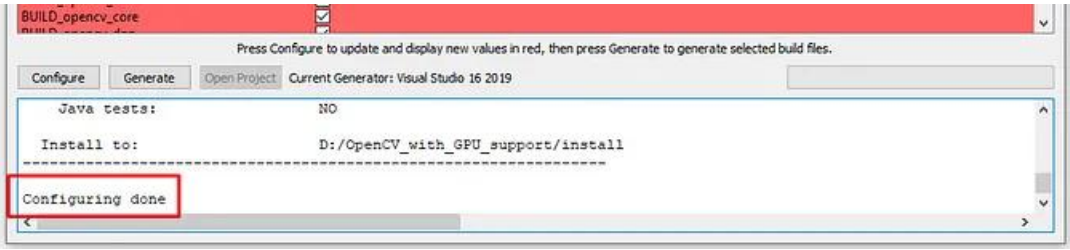


3. Select the proper Visual Studio version (Visual Studio 16 2019 for this guide) and then press “Finish”.



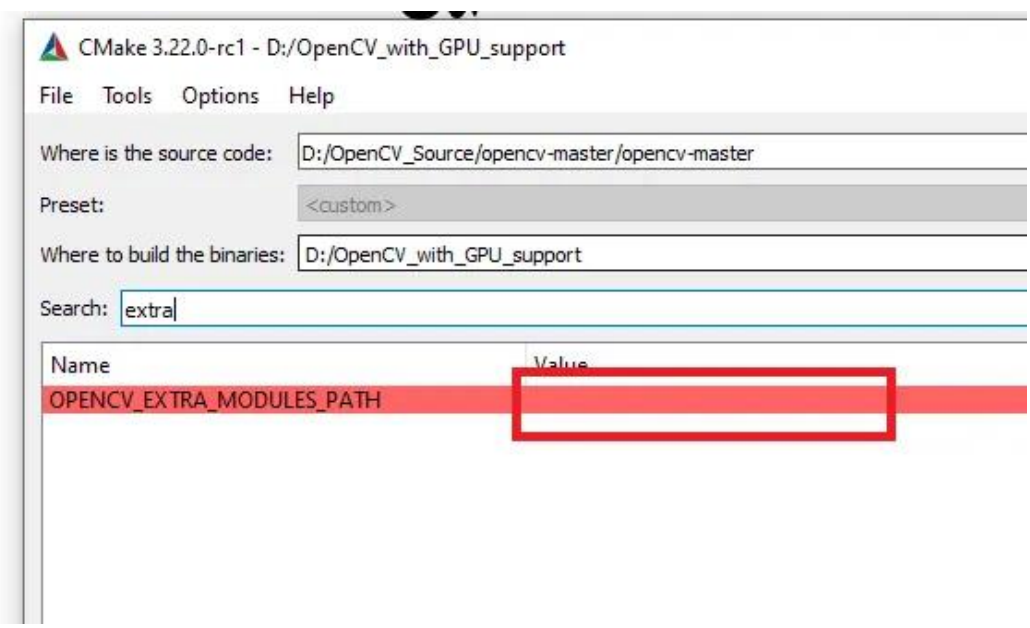
4. There will be a pre-scan/configuration step and if there is not any error, a “Configuring done” message will appear.

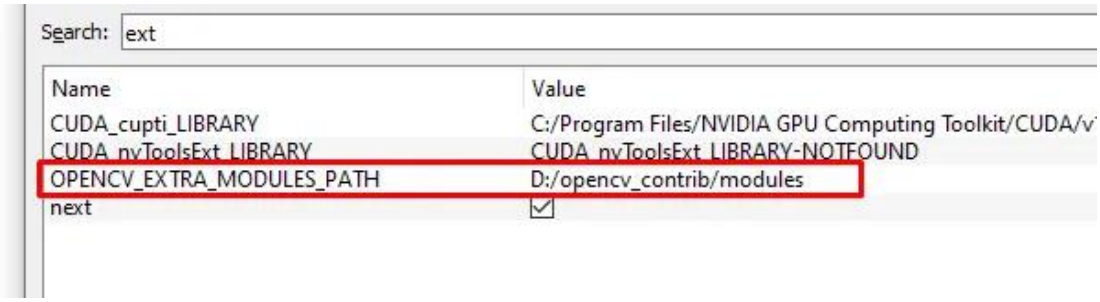




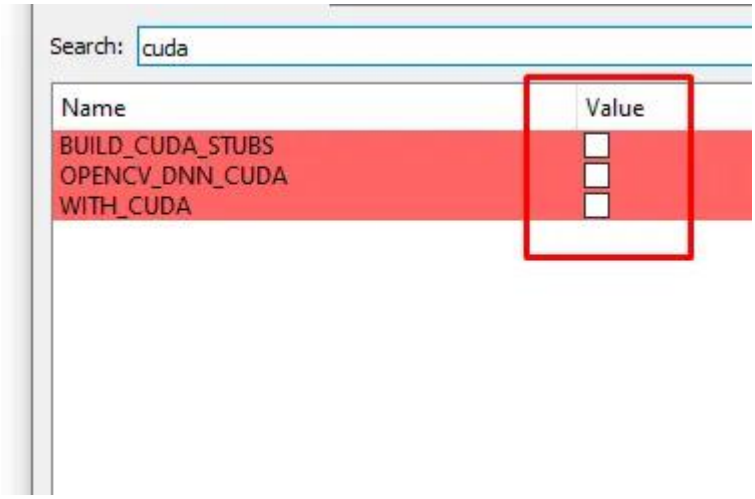
5. You will see a window that shows some options and their status.

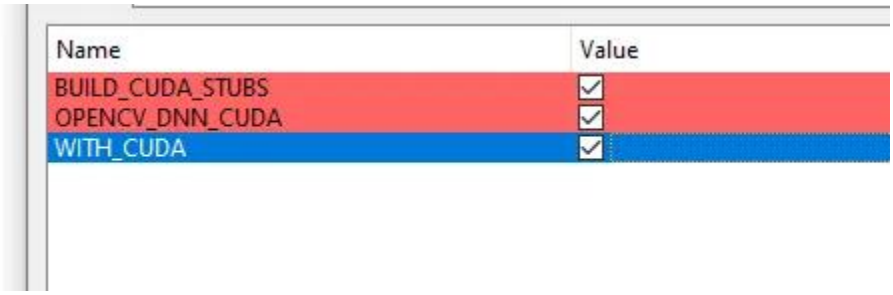
6. Type “extra” into the search box and “OPENCV_EXTRA_MODULES_PATH” will appear. Copy and paste the path of the opencv_contrib folder. Be careful with the “\” character in the path, don’t forget to change it to “/” to avoid errors.



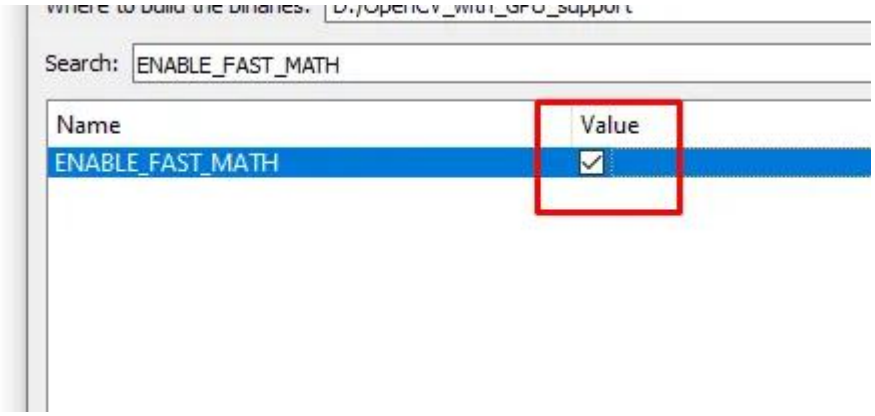


7. Type “CUDA” to the search box and activate the BUILD_CUDA_STUBS, OPENCV_DNN_CUDA, and WITH_CUDA options.

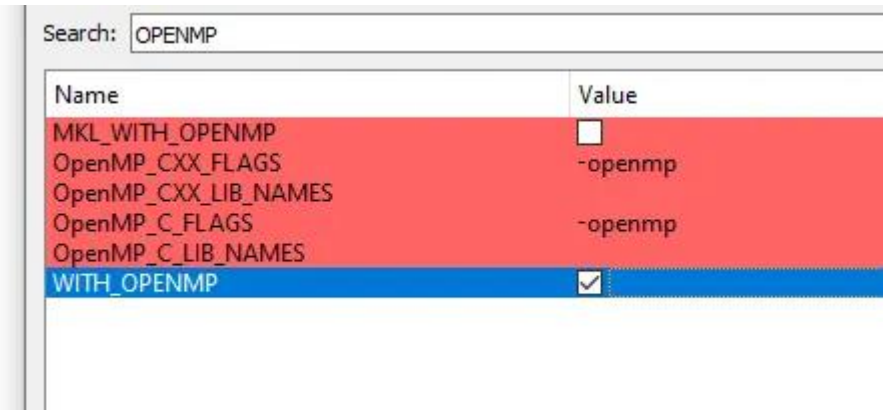




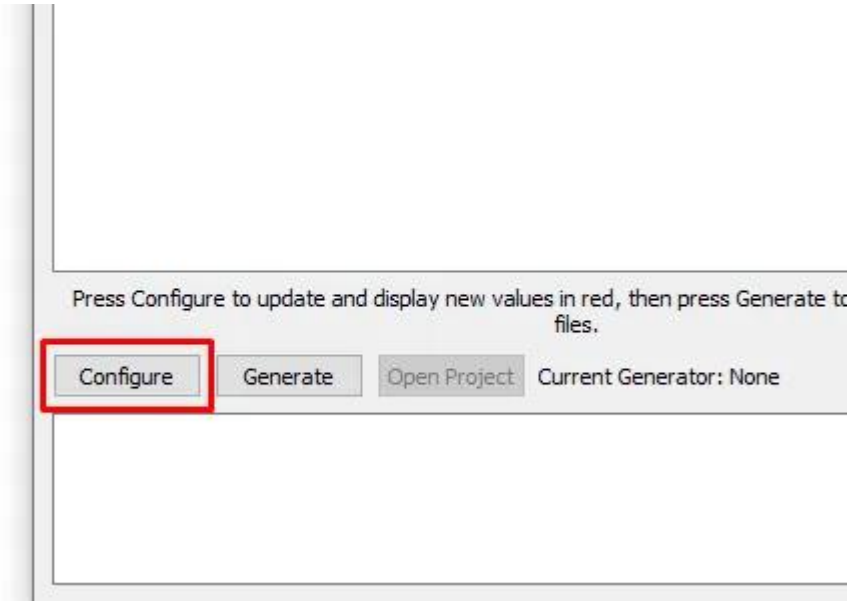
8. Type “ENABLE_FAST_MATH” to the search box and activate the ENABLE_FAST_MATH option.



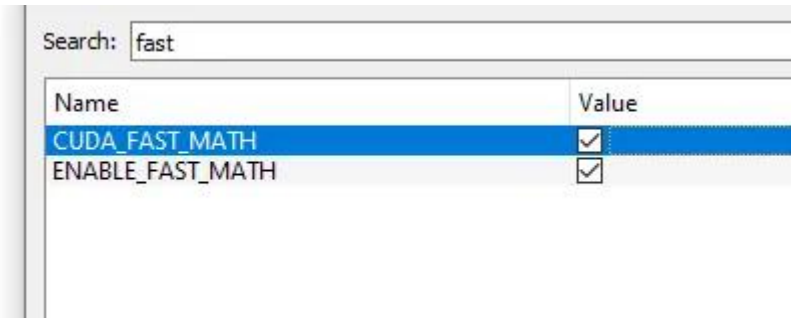
9. Type “OPENMP” to the search box and activate the OPENMP option.



10. Press the “Configure” button.



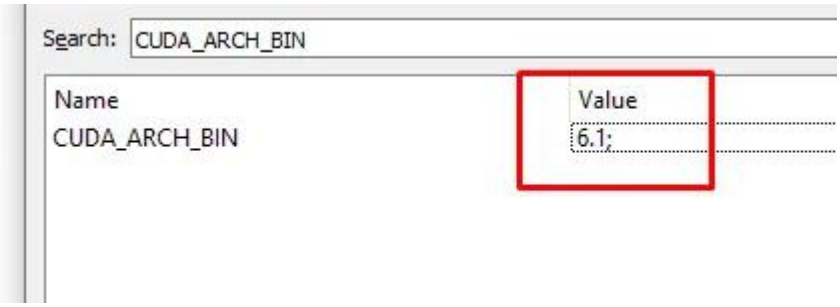
11. Type “CUDA_FAST_MATH” to the search box and activate the CUDA_FAST_MATH option.



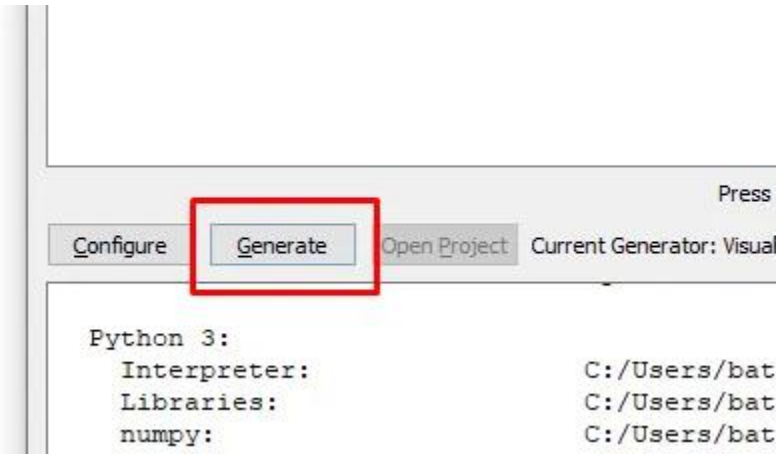
12. As for the last step, we need to choose the CUDA Compute Capability version. Although it is not required to choose it, if we don’t select a proper version, CMake will generate a compiled library for each version, so the installation step will take longer. You can check the CUDA

Compute Capability of your GPU from
https://gist.github.com/standaloneSA/99788f30466516dbcc00338b36ad5acf.

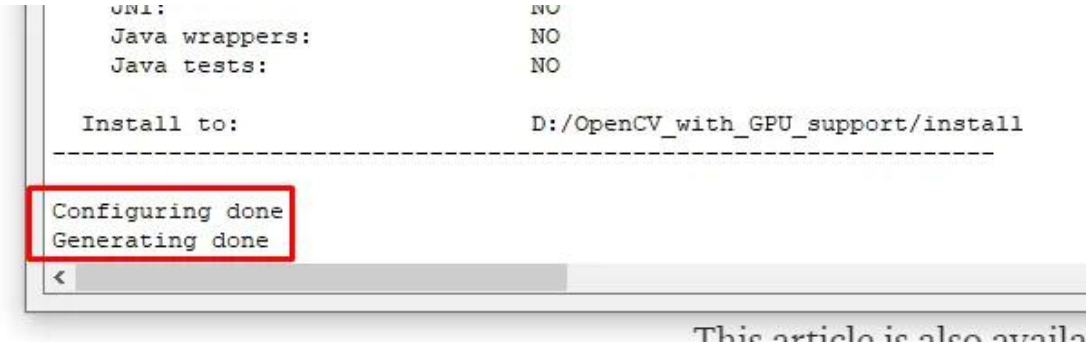
13. Type “CUDA_ARCH_BIN” to the search box and type the proper version for your GPU. Then press the “Configure” button.



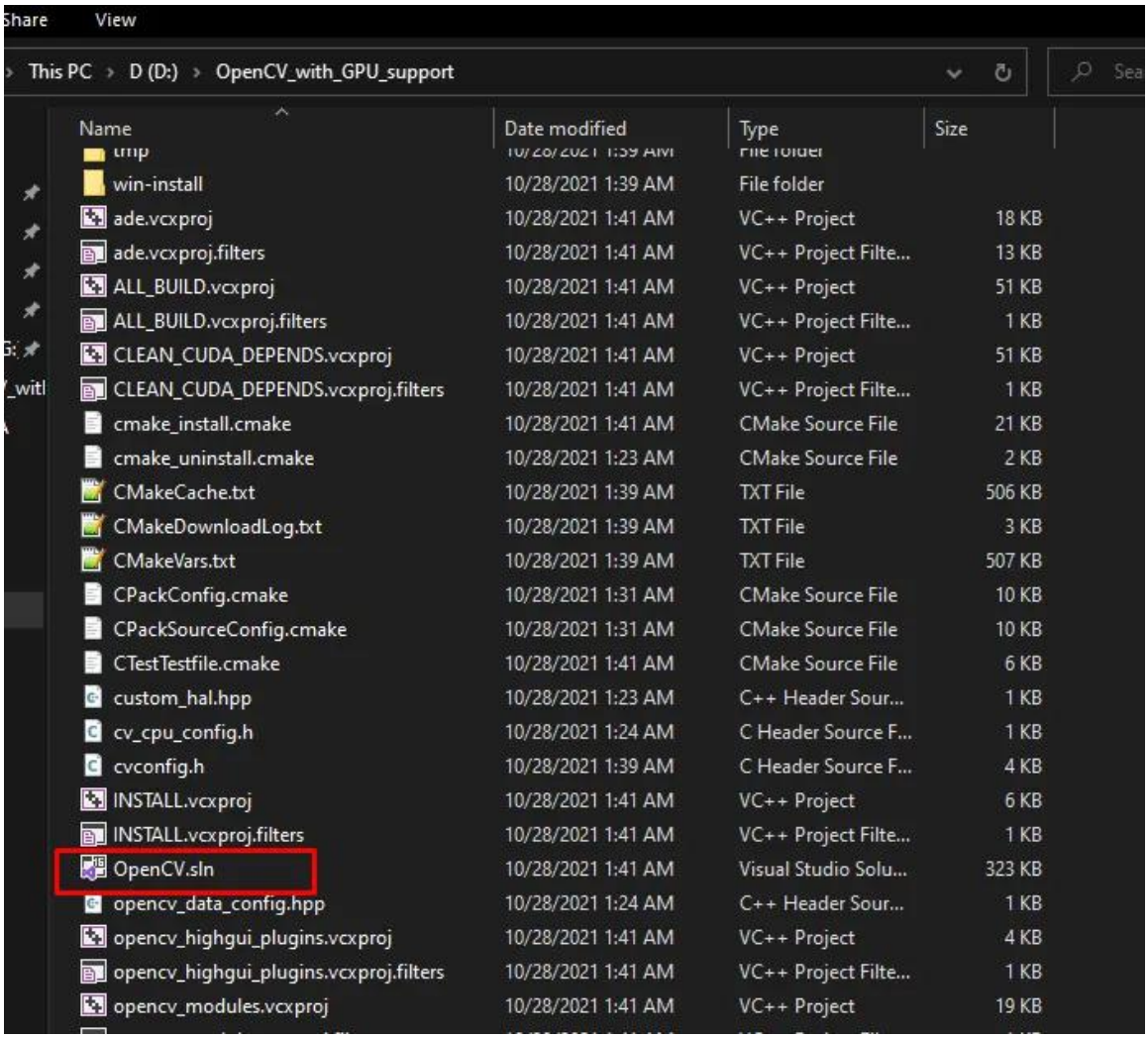
14. For the last step, press the “Generate” button next to “Configure”.



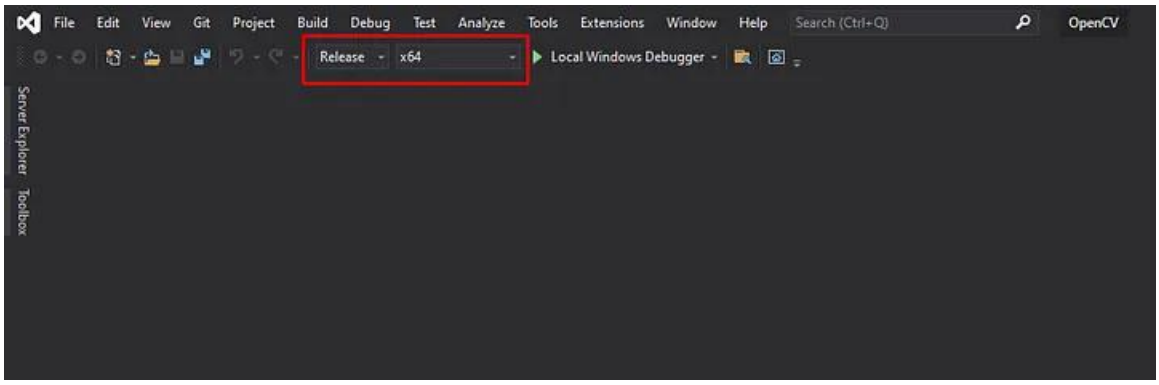
15. If there weren’t any errors, you’ll see a message similar to the one below. Now you can exit the CMake and proceed to the next step.



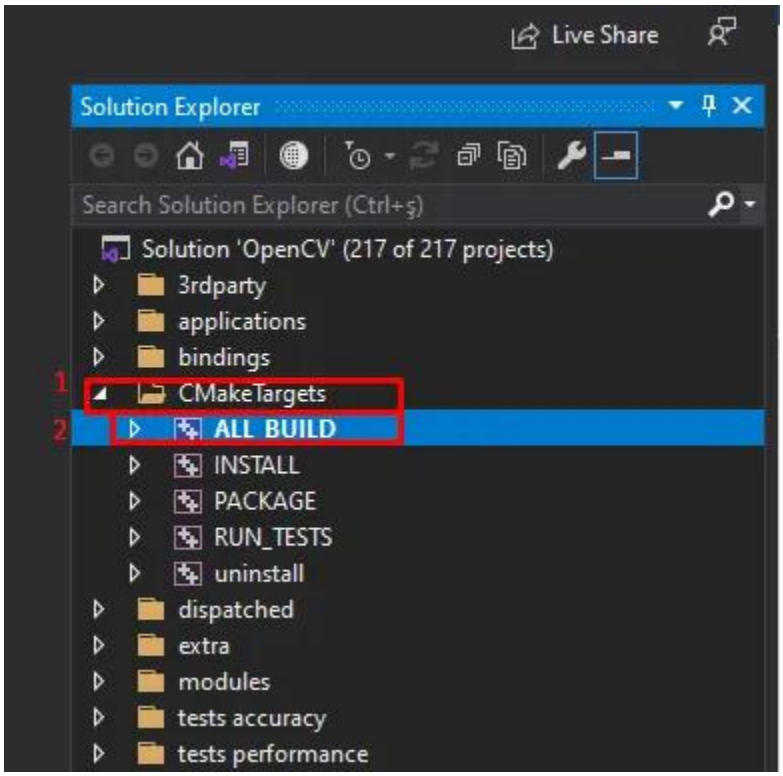
16. Head to the directory where your compiled files are located and open the OpenCV.sln with Visual Studio.

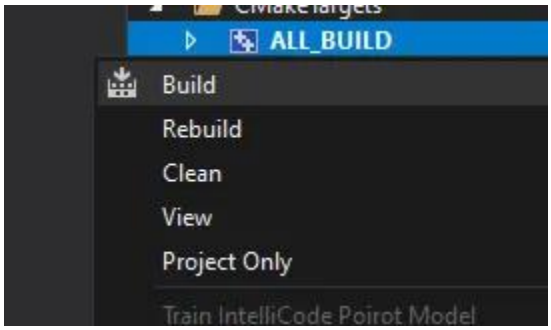


17. You can build libraries for Debug and Release configurations. In this guide, we'll proceed with the Release configuration but the steps for Debug configuration are the same. Select "Release" from the top menu.



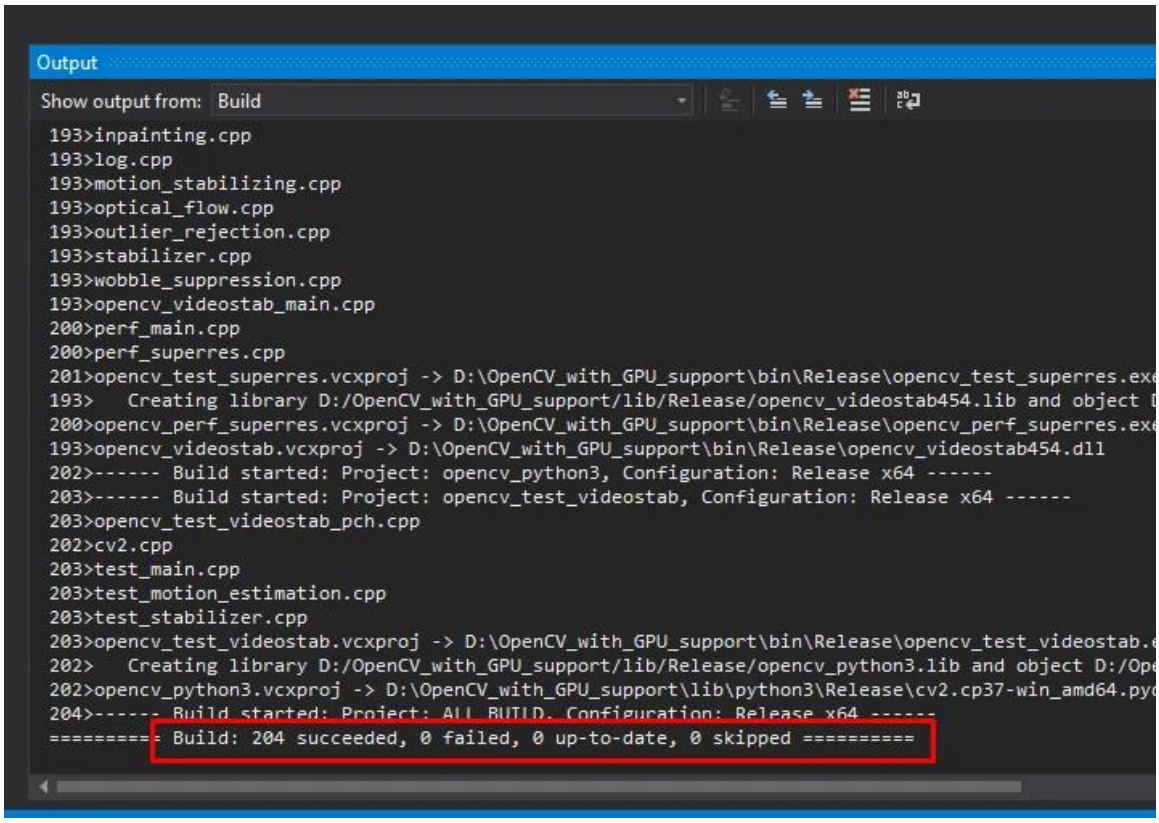
18. Expand the CMakeTargets folder at the Solution Explorer menu and select the ALL_BUILD option. Right-click to ALL_BUILDING and then click Build.



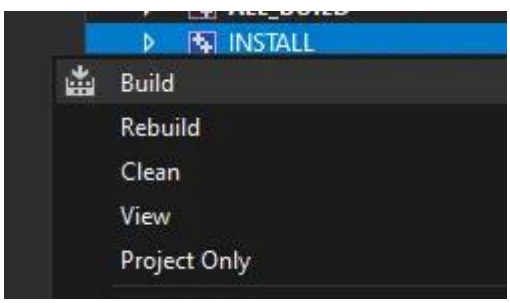


NOTE: This step may take 30 minutes to 1 hour depending on your system.

If there was no error, you'll get a message similar to the one below.



19. Follow Step 18 for the “INSTALL” option in the CMakeTargets folder. This one will take shorter.



If there was no error, you’ll get a message similar to the one below. You can exit the Visual Studio for now.

```
2>-- Installing: D:/OpenCV_with_GPU_support/install/etc/haarcascades/haarcascade_lic
2>-- Installing: D:/OpenCV_with_GPU_support/install/etc/haarcascades/haarcascade_low
2>-- Installing: D:/OpenCV_with_GPU_support/install/etc/haarcascades/haarcascade_pro
2>-- Installing: D:/OpenCV_with_GPU_support/install/etc/haarcascades/haarcascade_rig
2>-- Installing: D:/OpenCV_with_GPU_support/install/etc/haarcascades/haarcascade_rus
2>-- Installing: D:/OpenCV_with_GPU_support/install/etc/haarcascades/haarcascade_smi
2>-- Installing: D:/OpenCV_with_GPU_support/install/etc/haarcascades/haarcascade_upp
2>-- Installing: D:/OpenCV_with_GPU_support/install/etc/lbpcascades/lbpcascade_front
2>-- Installing: D:/OpenCV_with_GPU_support/install/etc/lbpcascades/lbpcascade_front
2>-- Installing: D:/OpenCV_with_GPU_support/install/etc/lbpcascades/lbpcascade_front
2>-- Installing: D:/OpenCV_with_GPU_support/install/etc/lbpcascades/lbpcascade_profi
2>-- Installing: D:/OpenCV_with_GPU_support/install/etc/lbpcascades/lbpcascade_silve
2>-- Installing: D:/OpenCV_with_GPU_support/install/x64/vc16/bin/opencv_annotation.e
2>-- Installing: D:/OpenCV_with_GPU_support/install/x64/vc16/bin/opencv_visualisatio
2>-- Installing: D:/OpenCV_with_GPU_support/install/x64/vc16/bin/opencv_interactive-
2>-- Installing: D:/OpenCV_with_GPU_support/install/x64/vc16/bin/opencv_version.exe
2>-- Installing: D:/OpenCV_with_GPU_support/install/x64/vc16/bin/opencv_version_win3
2>-- Installing: D:/OpenCV_with_GPU_support/install/x64/vc16/bin/opencv_model_diagno
========== Build: 2 succeeded, 0 failed, 203 up-to-date, 0 skipped ==========
```

Build succeeded

2. Installing & Compile OpenCV with CPU Support

Installation of Required Driver and Programs

Installation involves several steps which are

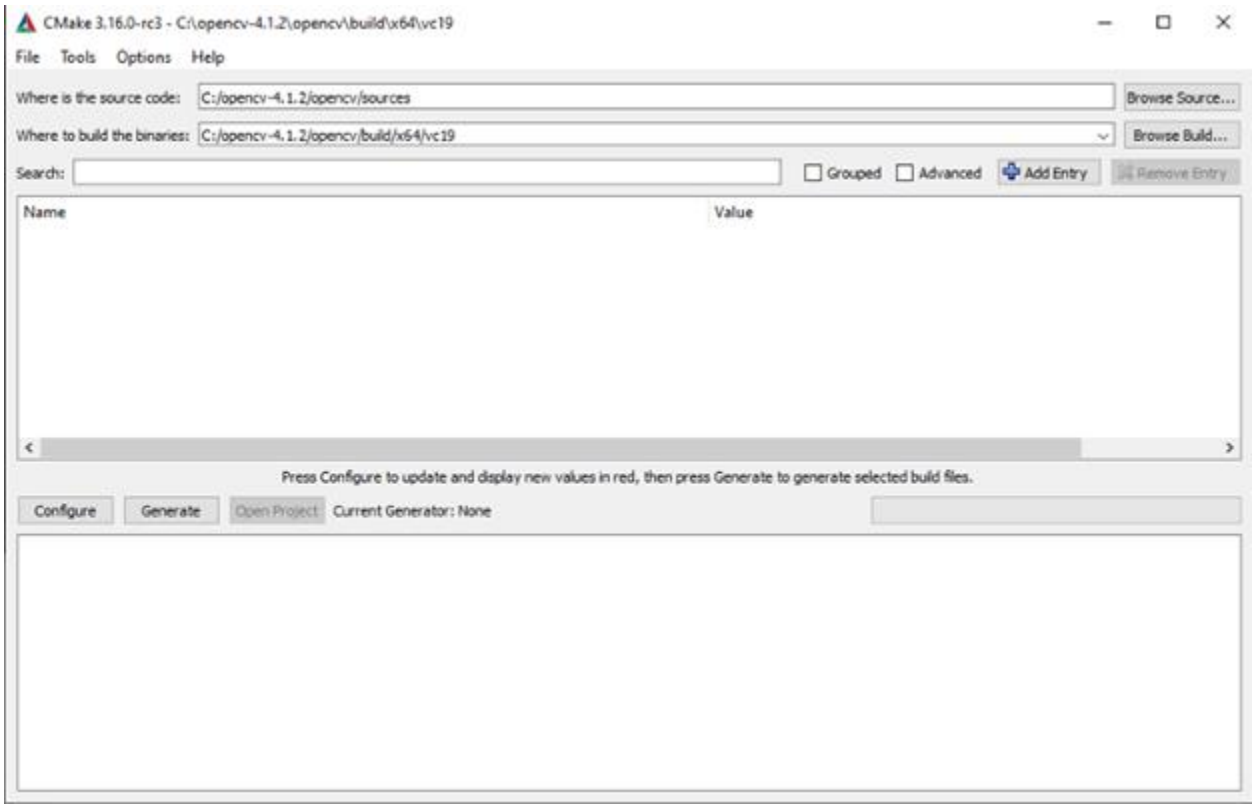
1- IDE installation (Visual Studio Community)

2 - CMake installation

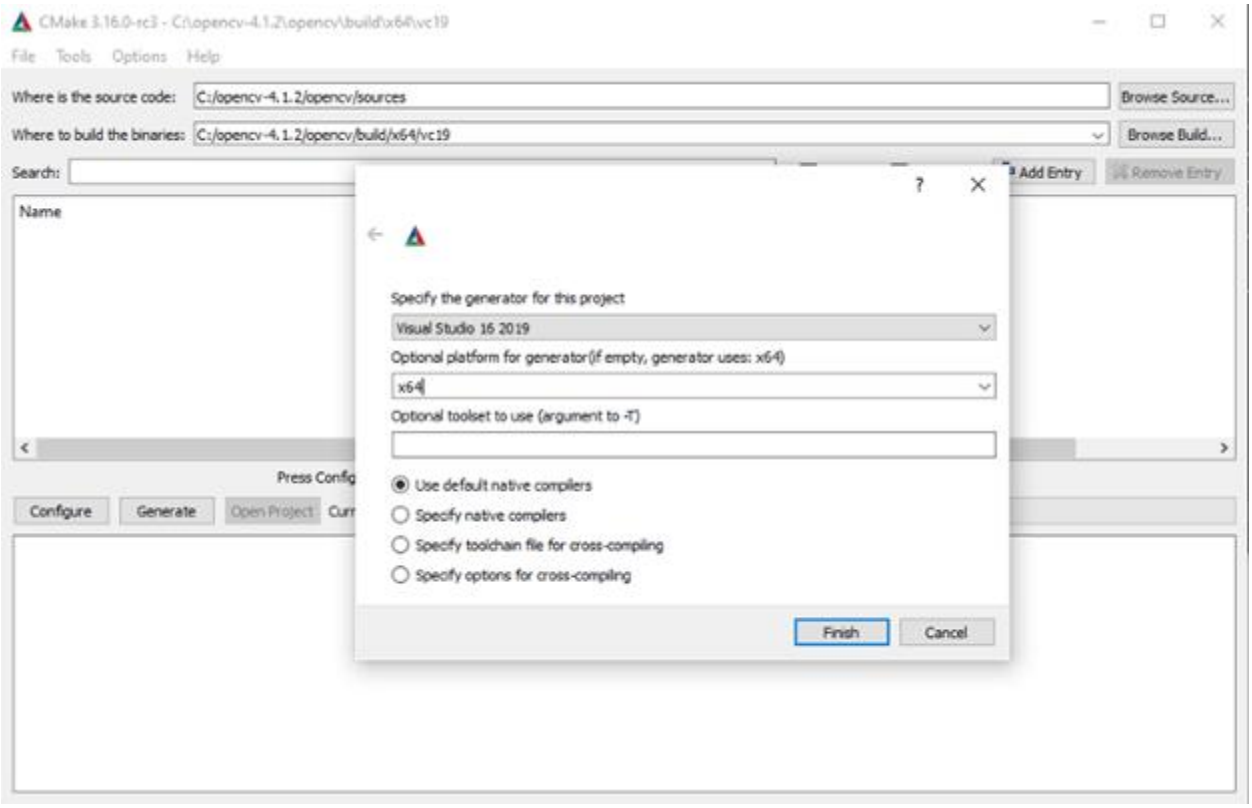
3- OpenCV installation

Above steps refer from Installing & Compile OpenCV with GPU Support

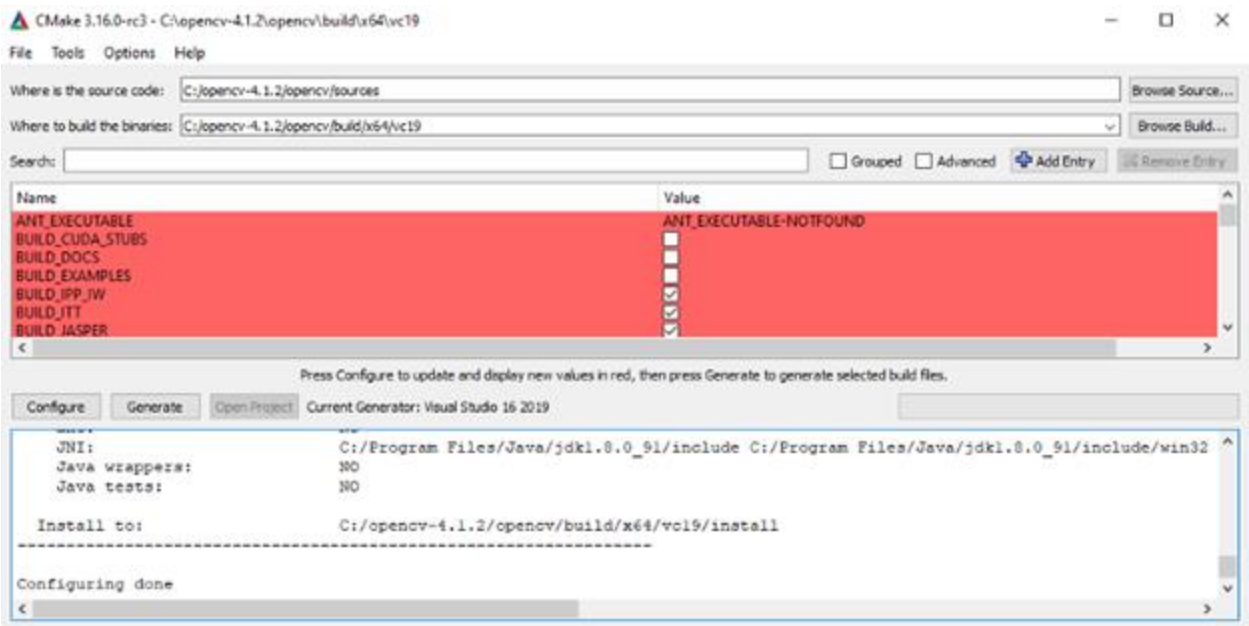
next start cmake



- click on “Configure”

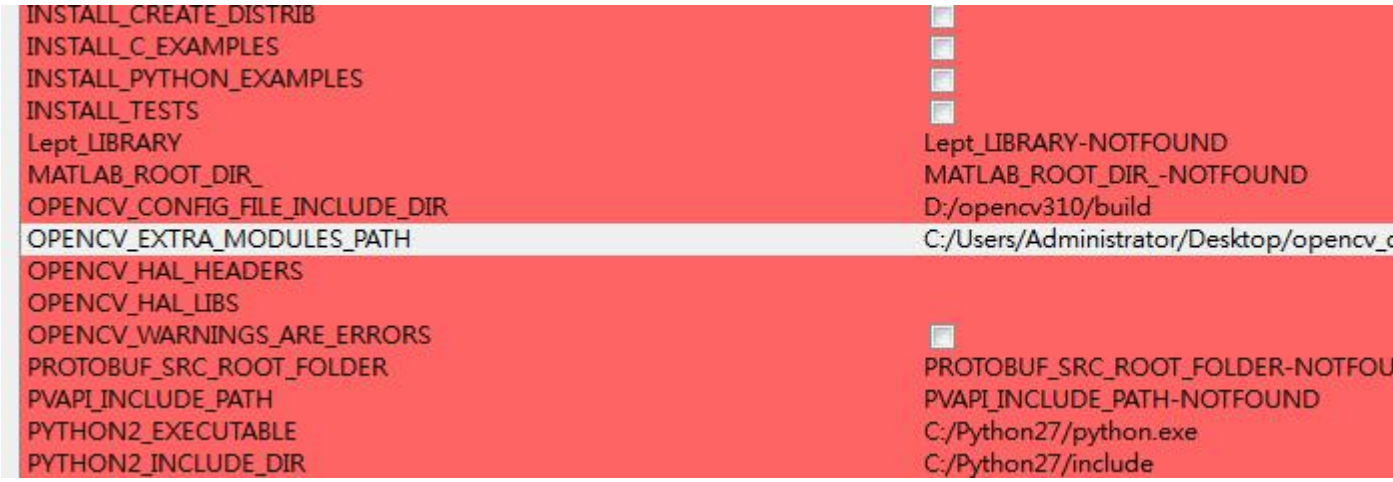


in the window that pops up
— select the generator as Visual Studio 16 2019
— Platform as x64(as we are working on 64-bit platform)
and then click finish
(if you have not installed Python and setup environment variable, then you will get errors, if you get any error then don't proceed, try to resolve or send the snapshot of error in comments)



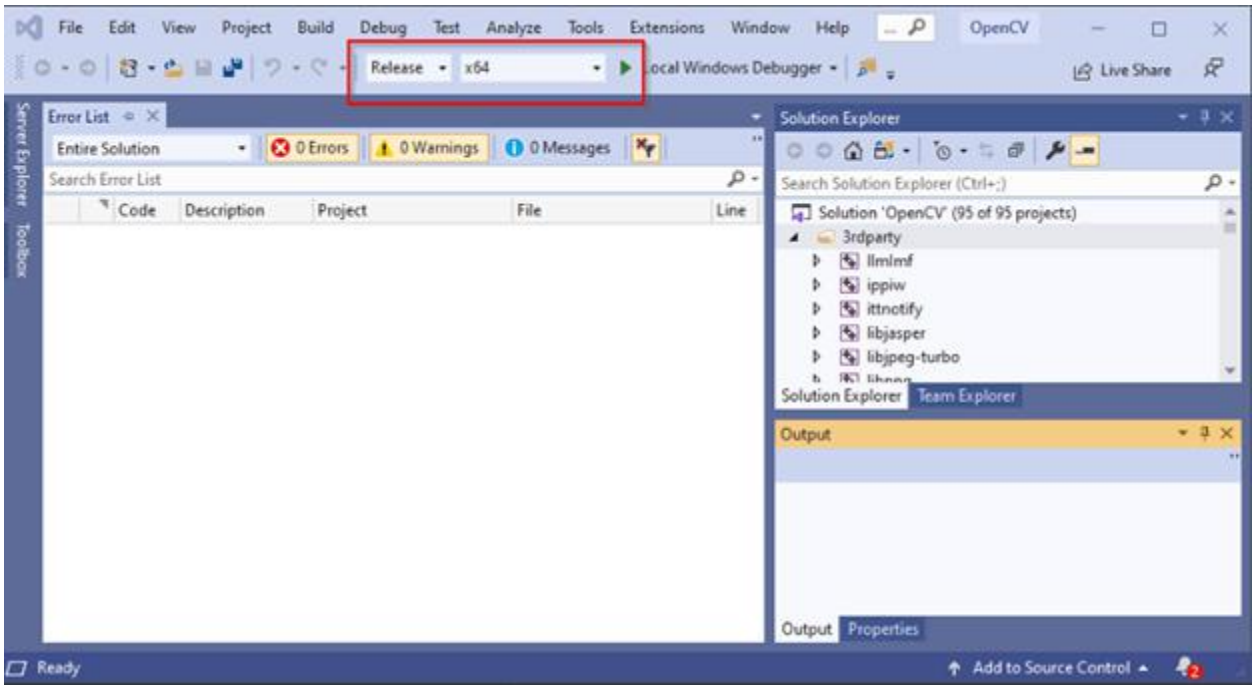
once configuration is done, click “Generate”

Set up OPENCV_EXTRA_MODULES_PATH to proper path(<opencv_contrib>/modules)



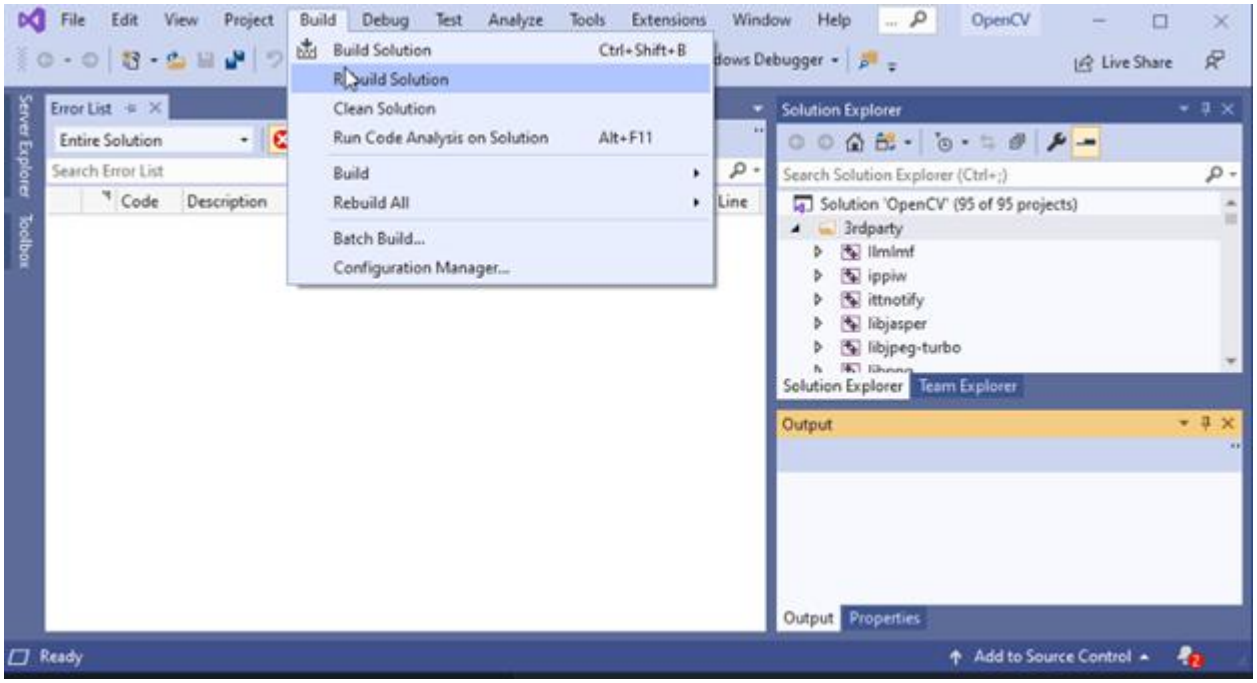
Press configure again, then generate

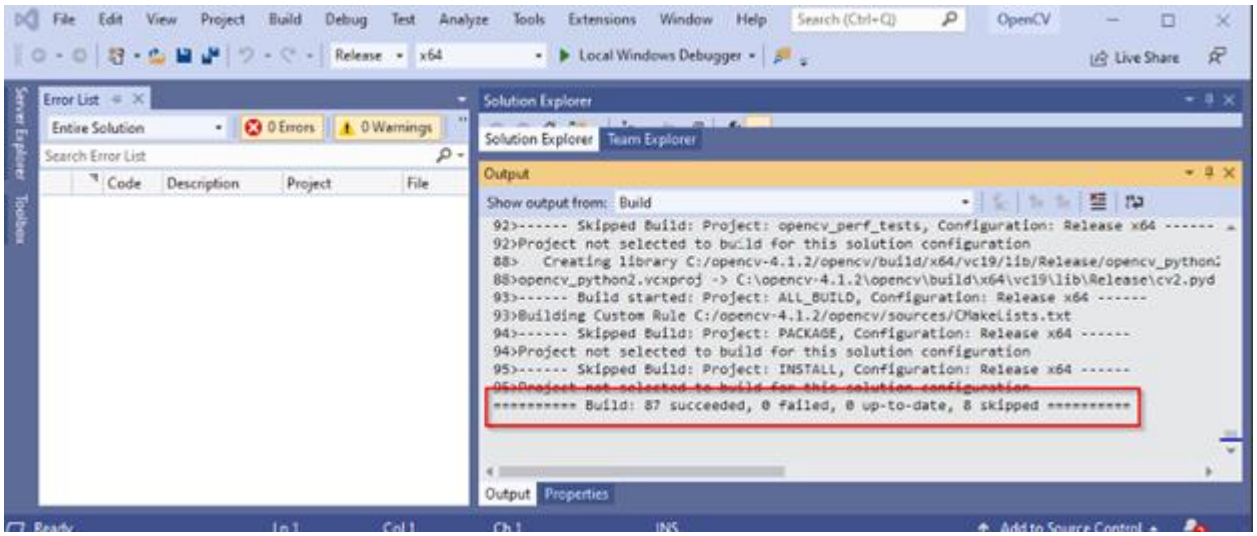
click on “Open Project”. if you have installed MS Visual Studio 2019 with Desktop Development for C++ then it will launch MS Visual Studio



make sure to select target as “Release” and platform as “x64”

click on “Build Solution”





When build is successful you should be able see above. Estimated build time is 30 mins.