OpenCV 4.0.1-Contrib on WIN10 (C++ and Python)

Installation guide

Kefei mo

GitHub:
https://github.com/kefeimo/OpenCV-4.0.1-contrib-install-for-WIN10

YouTube:

Outline

Step1: Download all the software and the OpenCV libraries

Step2: Software installation and extraction

Step3: Build C++ projects using CMake

- 3.1 choose source and binaries directories
- 3.2 additional change
- 3.3 generate build files

Step4: Compile OpenCV

- 4.1 build debug version for ALL_BUILD
- 4.2 build release version for ALL_BUILD
- 4.3 build release version for INSTALL
- 4.4 build release version for INSTALL

Step5: Update environment variables

- 5.1 environment variable – PATH
- 5.2 user environment variable – OPENCV_DIR

Step6: Test the compiled OpenCV (in C++)

- 6.1 build a test project using CMake
- 6.2 build and execute in Visual Studio

Step1: Download all the software and the OpenCV libraries

List

1. Visual studio 14 2015 X64 community
https://my.visualstudio.com/Downloads?q=visual%20studio%202015&wt.mc_id=o~msft~vscom~older-downloads

2. CMake cmake-3.13.2-win64-x64.msi
https://cmake.org/download/

3. Anaconda Python 3.7 version for Windows installers
https://www.anaconda.com/download/

4. OpenCV 4.0.1 source code (ZIP)
https://github.com/opencv/opencv/releases

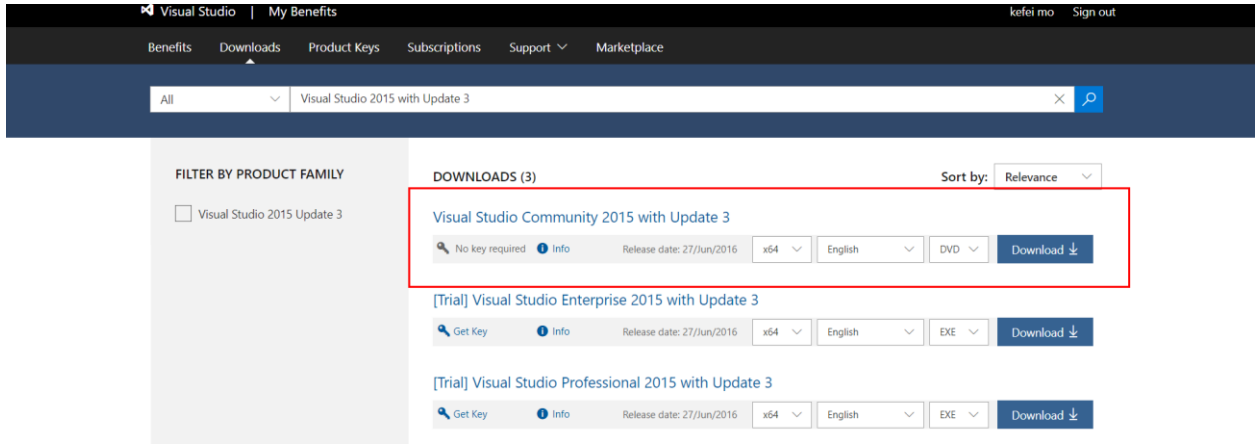
5. Opencv-contrib 4.0.1 source code (ZIP)
https://github.com/opencv/opencv_contrib/releases

Note: highly recommend to download the exact version

Step1: Download all the software and the OpenCV libraries

List

1. Visual studio 14 2015 X64 community (free to download after sign in Dev Essentials)
https://my.visualstudio.com/Downloads?q=visual%20studio%202015&wt.mc_id=o~msft~vscom~older-downloads



sign in Visual Studio Dev Essentials:

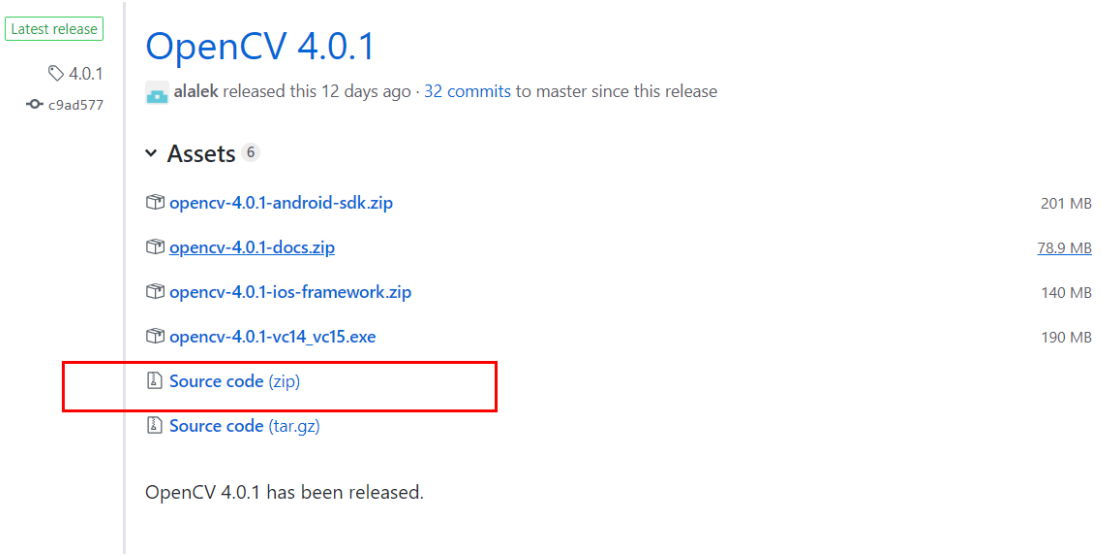
https://my.visualstudio.com/Downloads?q=Visual%20Studio%202015%20with%20Update%203

Step1: Download all the software and the OpenCV libraries

List

4. OpenCV 4.0.1 source code (ZIP)
https://github.com/opencv/opencv/releases

5. Opencv-contrib 4.0.1 source code (ZIP)
https://github.com/opencv/opencv_contrib/releases



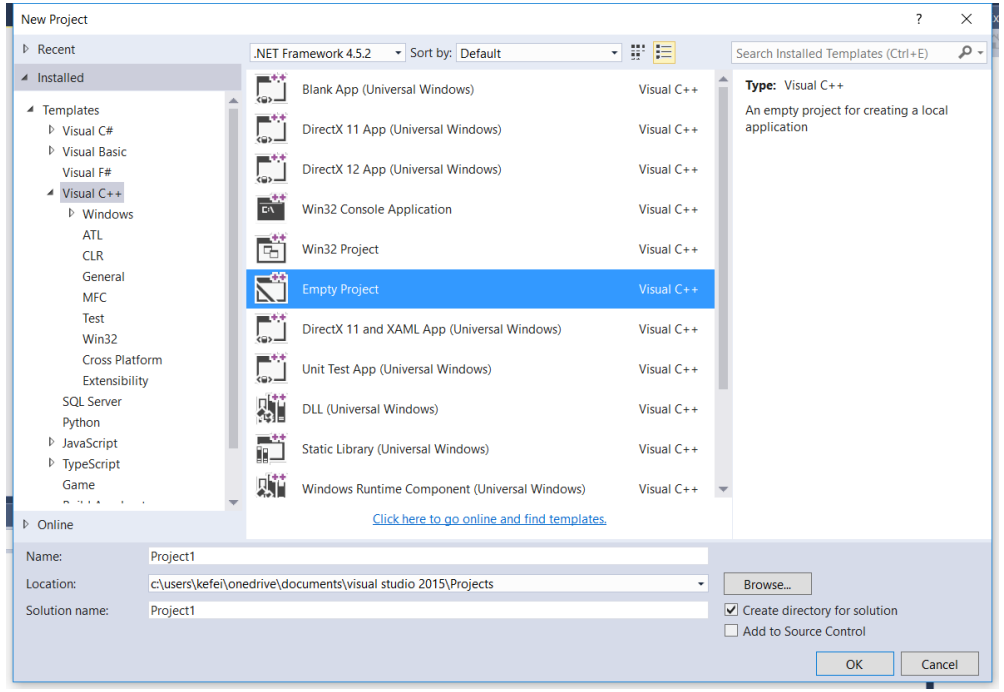
Note: highly recommend to download the exact version,
Use “Next” button to find former versions if necessary

Step2: Software installation and extraction

List

1. Visual studio 14 2015 X64 community (with VC++ compiler)

Note: remember to install Visual C++ compiler. To test, build an empty VC++ project

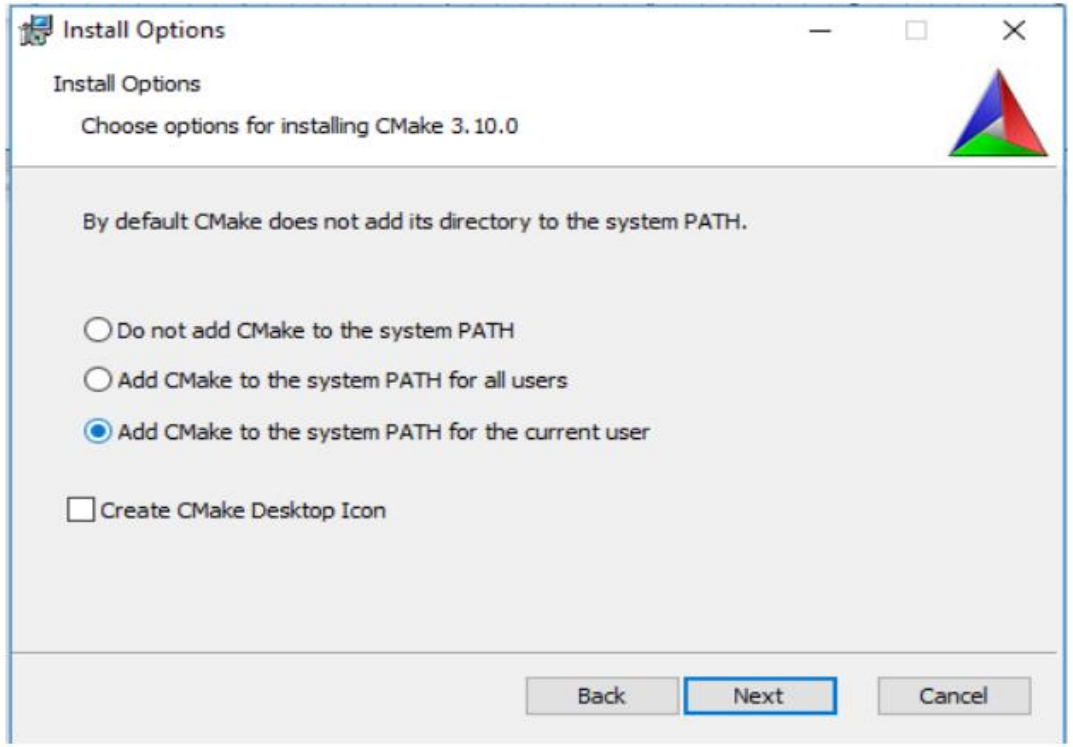


Step2: Software installation and extraction

List

2. CMake cmake-3.13.2-win64-x64.msi

Note: choose “Add CMake to system PATH for the current user”

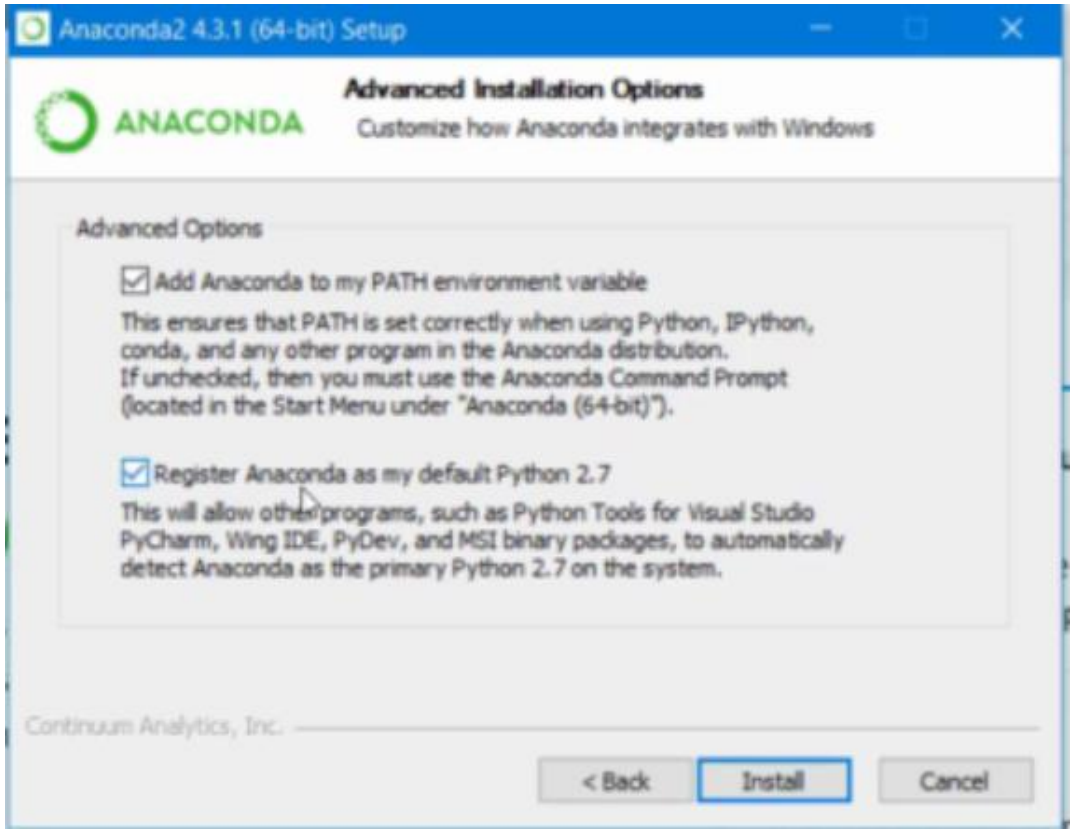


Step2: Software installation and extraction

List

3. Anaconda Python 3.7 version for Windows installers

Note: check both for advanced options



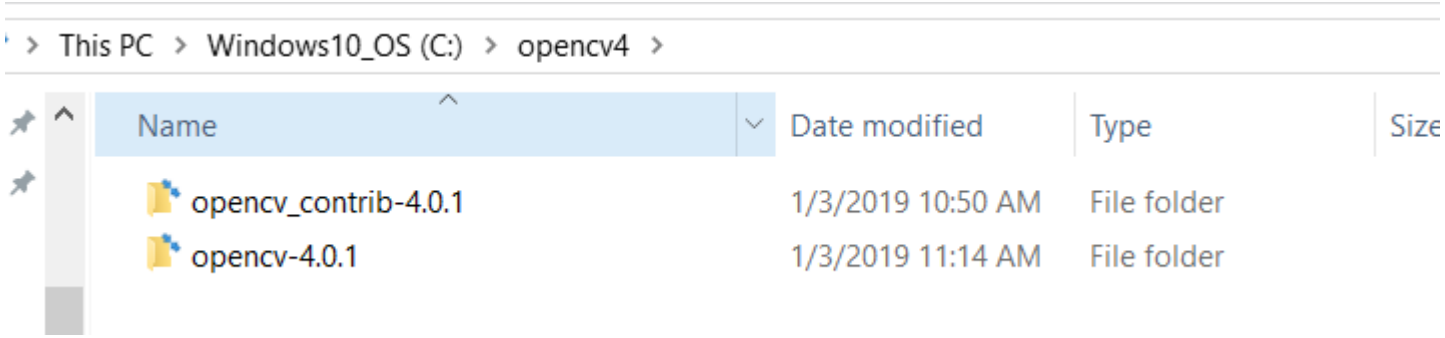
Step2: Software installation and extraction

List

4. OpenCV 4.0.1 source code (ZIP)

5. OpenCV-contrib 4.0.1 source code (ZIP)

Note: recommend to extract the OpenCV 4.0.1 and OpenCV-contrib 4.0.1 in the same folder.

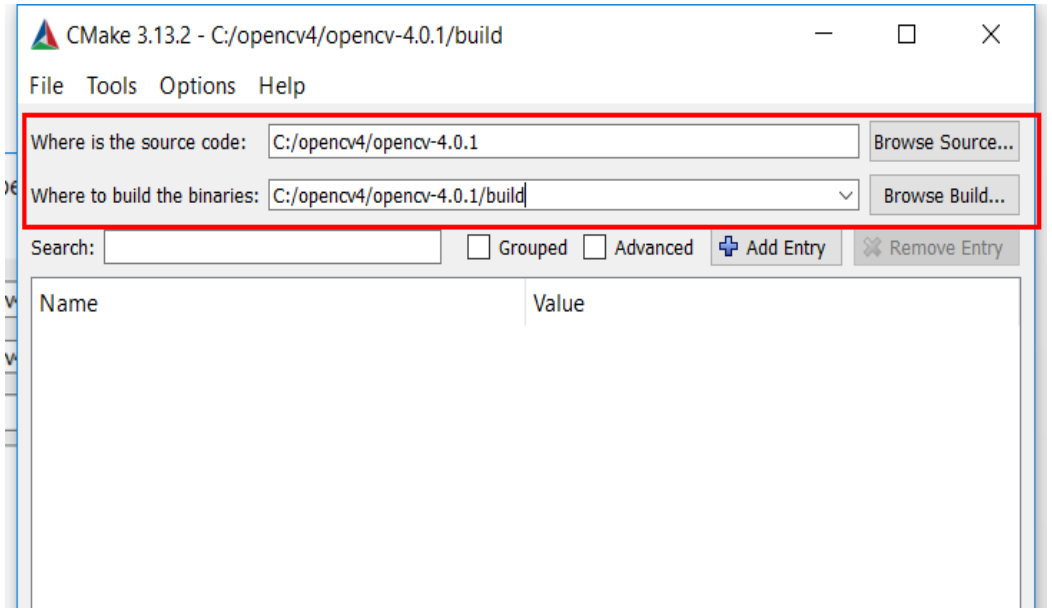


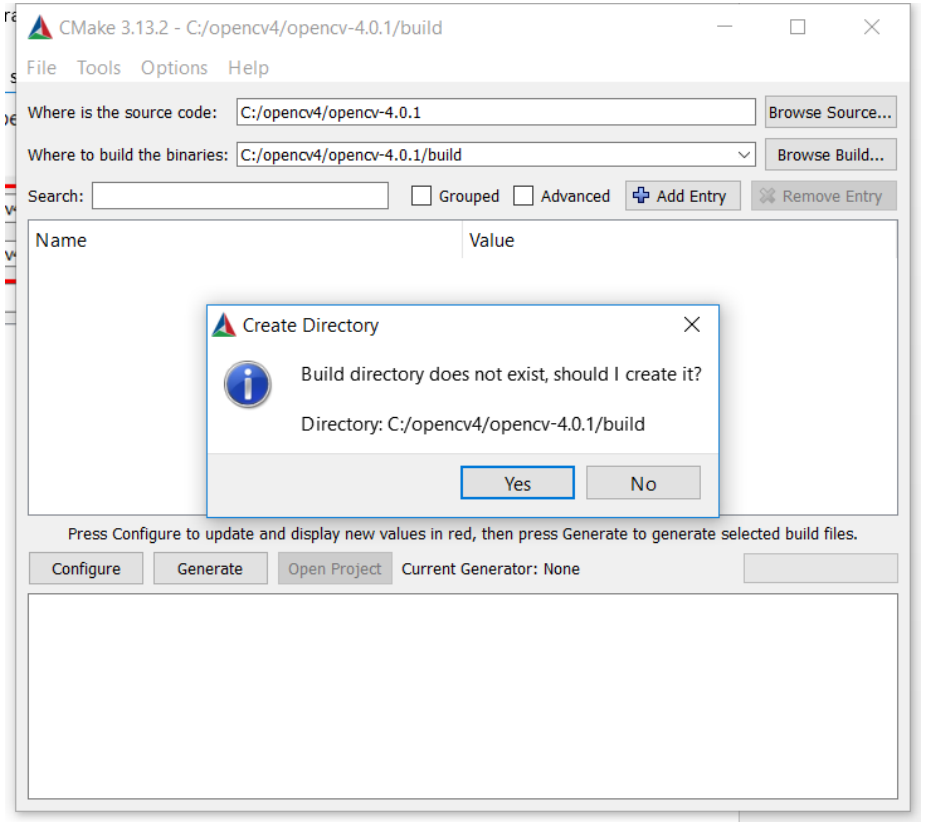
Step3: Build C++ projects using CMake

3.1 choose source code and binaries directories

step:

- Use “browse source” button, and make sure the address use “/” instead of “\”, then click “configure”
- the binaries fold is new build, would be asked for permission, click yes
- Success result and potential error see next



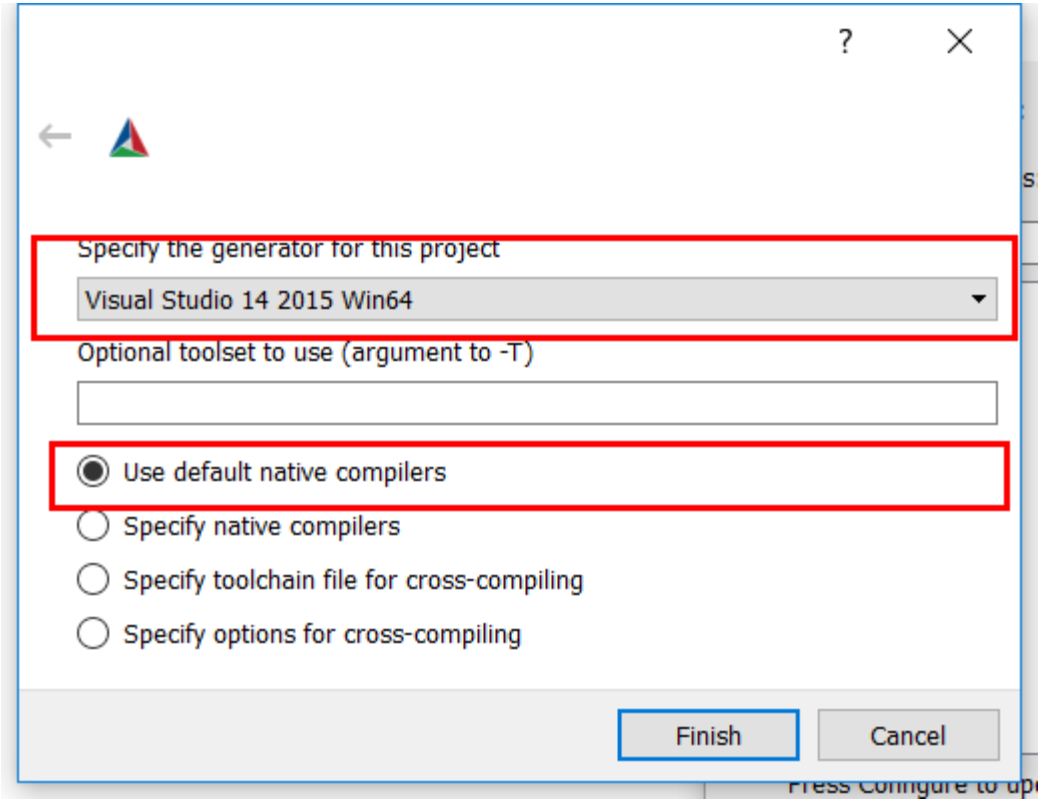


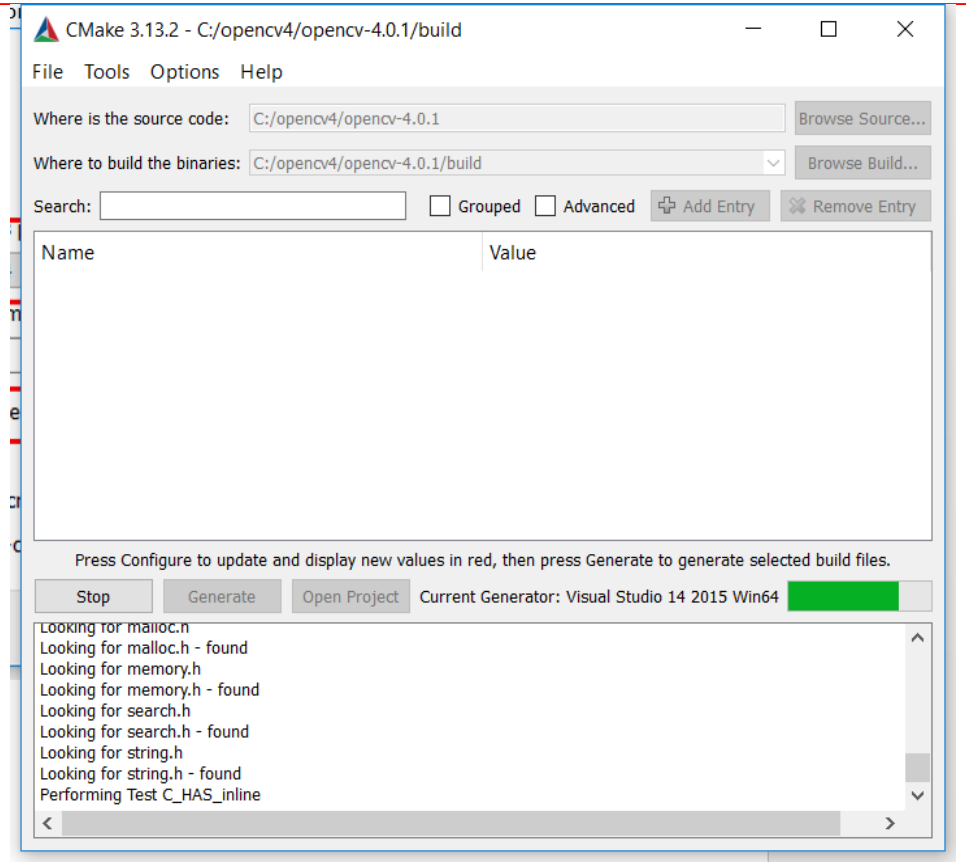
Step3: Build C++ projects using CMake

3.1 choose source code and binaries directories

step :

- Will be asked to specify compiler. Note: if the compiler has been installed correctly, it will automatically choose the compiler (normally the newest version). Here we would like to use visual studio 2015 x64. However, if newer version has been installed, i.e. visual studio 2017, then, specify to visual studio 2015 x64. note: “Visual Studio 14 2015 WIN64” means Visual Studio 2015, “Visual Studio 15 2017” means Visual Studio 2017
- Then choose” use default native compilers”, then click “Finish”
- Start configuring, would take a while (there is progress bar to see the percentage)

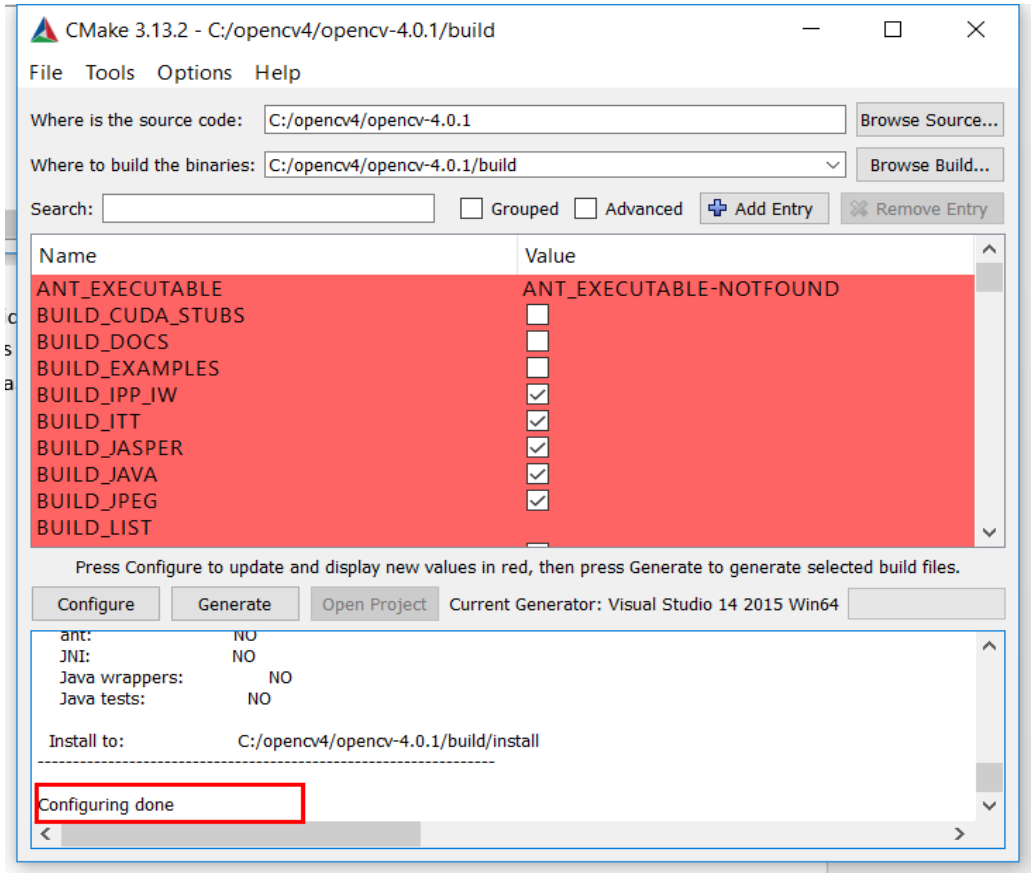




Step3: Build C++ projects using CMake

3.1 choose source code and binaries directories

Result : as shown
Note: the salmon color items are configuration changed, don't freak out seeing them.

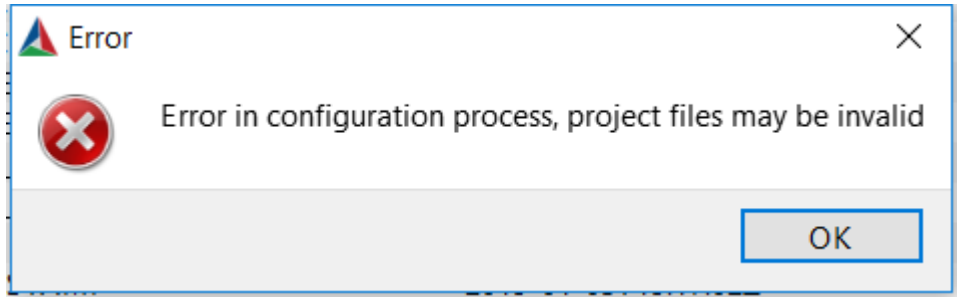


Step3: Build C++ projects using CMake

3.1 choose source code and binaries directories

Potential errors :

- “project files may be invalid”, reason: didn’t download opencv source file (ZIP) correctly, or the address of “where is the source code” is wrong. (another sub-directory after extraction)
- “cannot find compiler”, reason: didn’t install vc++ compiler (only install visual studio)

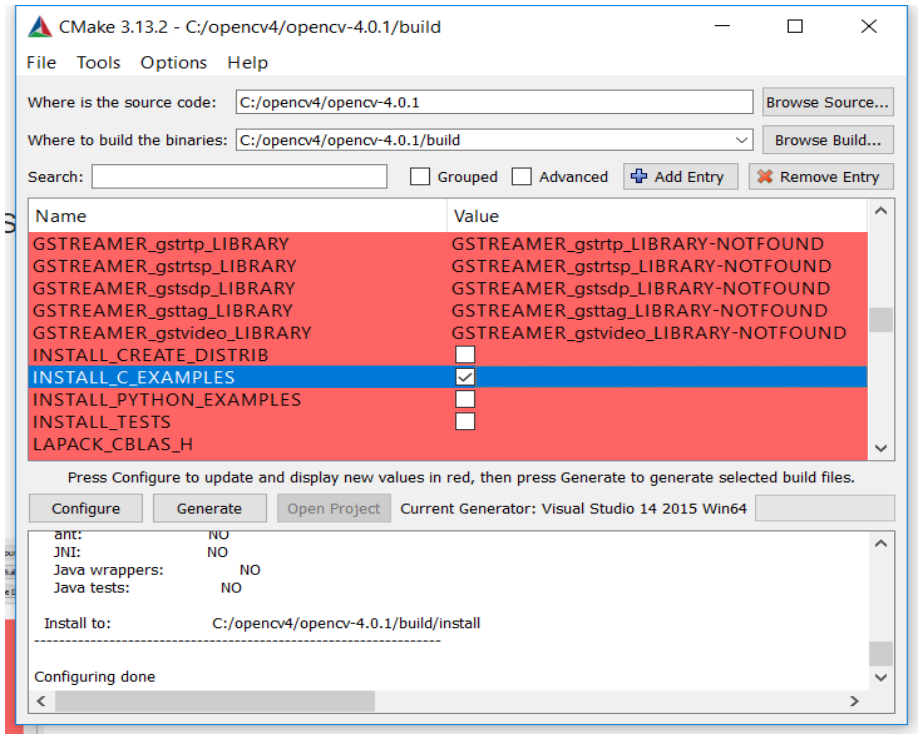


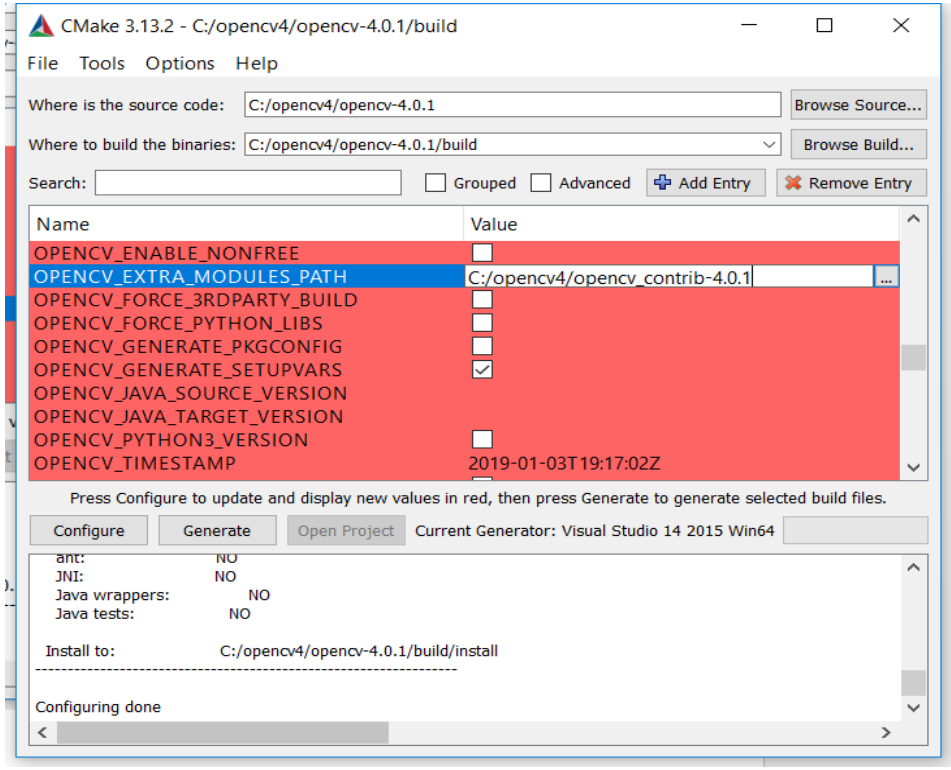
Step3: Build C++ projects using CMake

3.2 additional change (2nd time configure)

step:

- check “INSTALL_C_EXAMPLES”
- Specify path to “OPENCV_EXTRA_MODULES_PATH” (note: here the path is wrong, should have use “C:\opencv4\opencv_contrib-4.0.1\modules”)

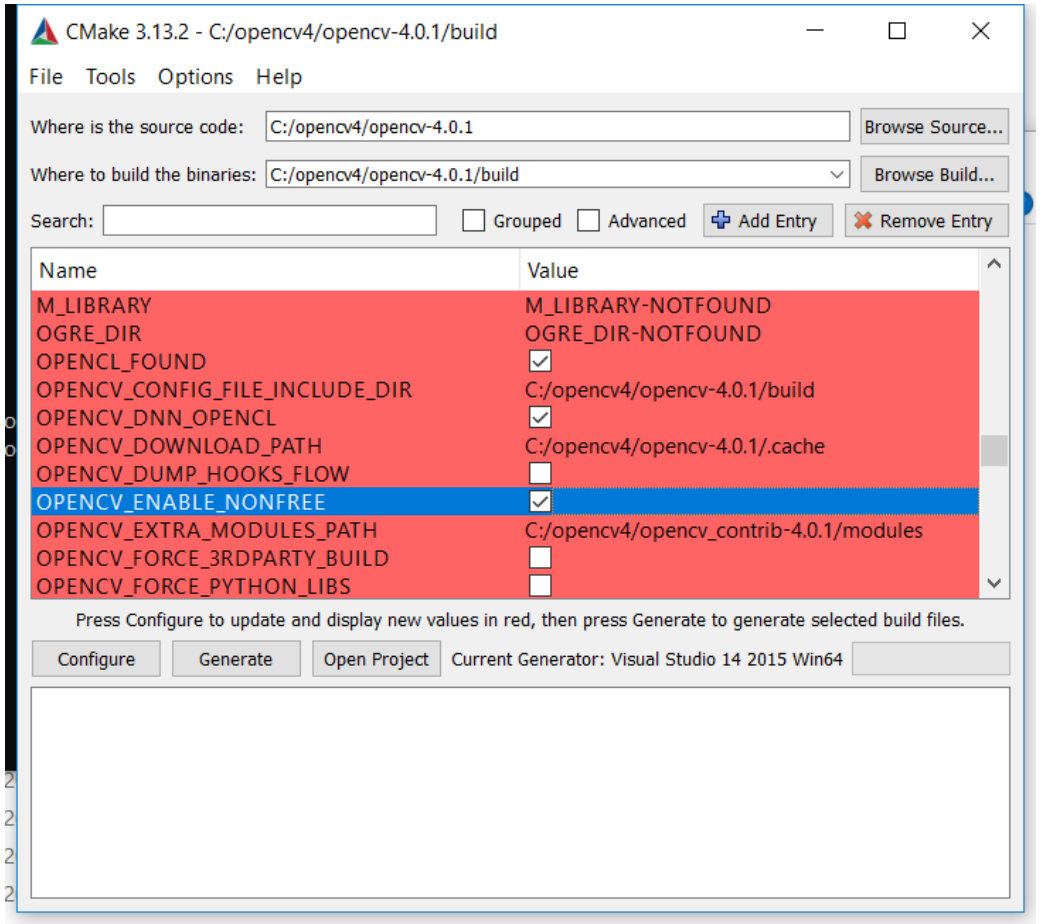


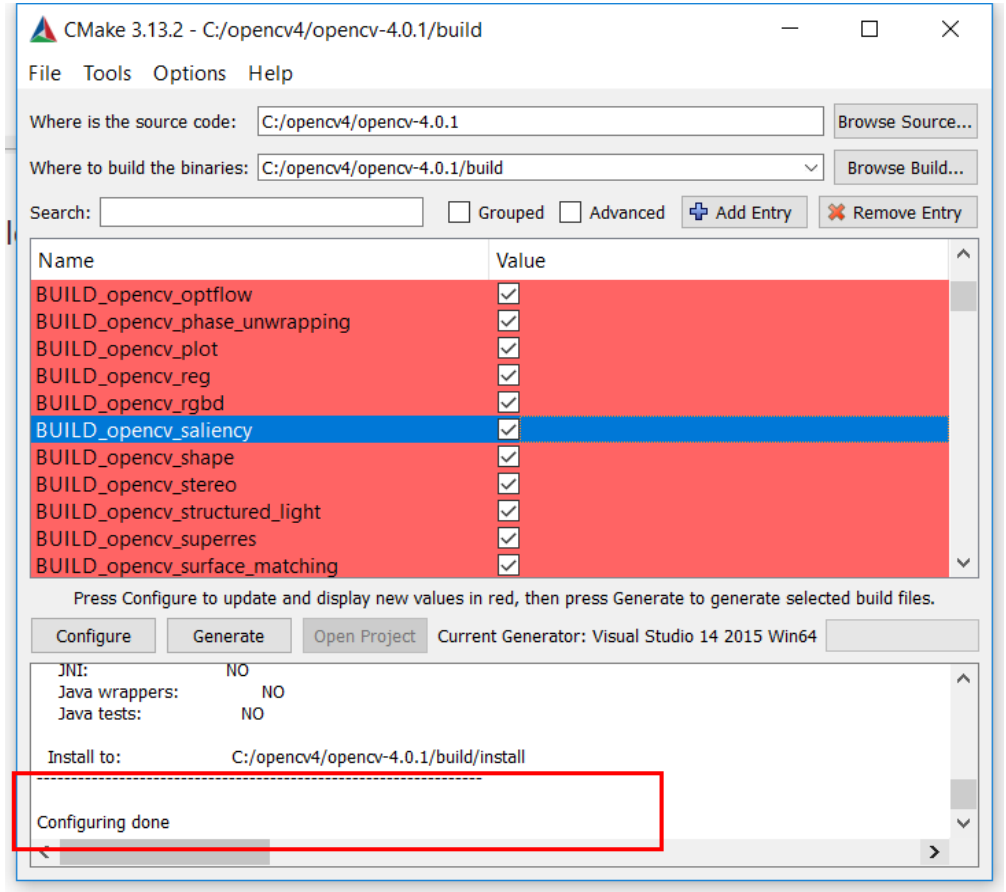


Step3: Build C++ projects using CMake

3.2 additional change (2nd time configuration)

step:
- check “OPENCV_ENABLE_NONFREE” (if not found, it might appear after 2nd configuration, or even later. Just keep in mind to check this option when it appears).
- Then click “Configure”. If no error, it would show configuring done

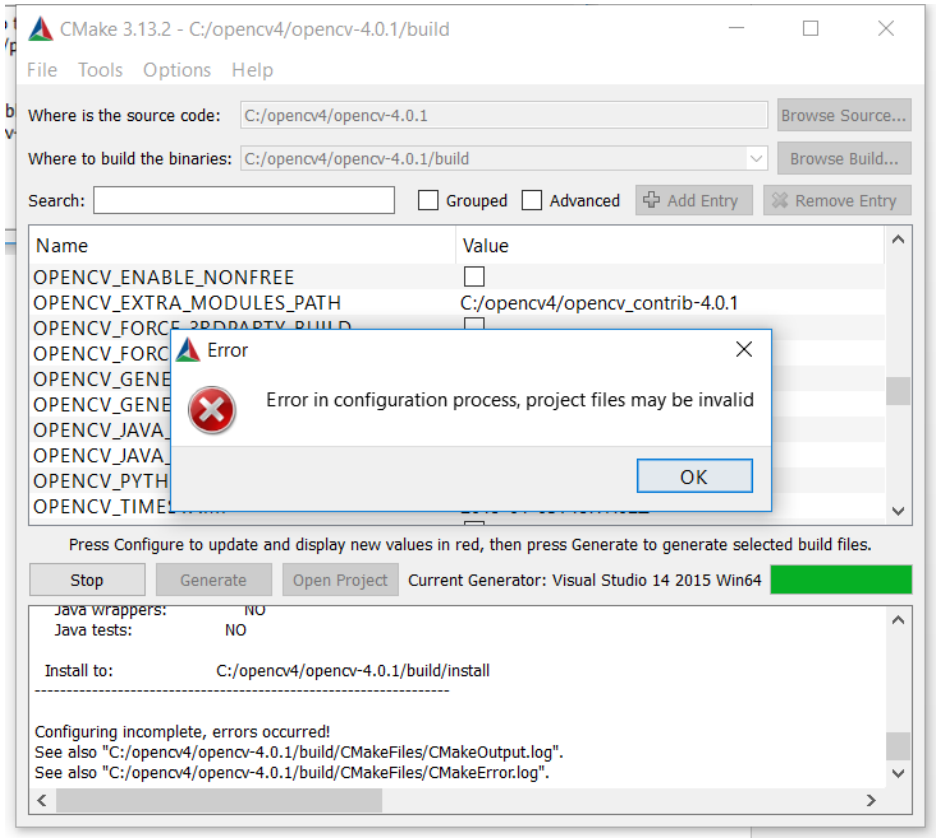


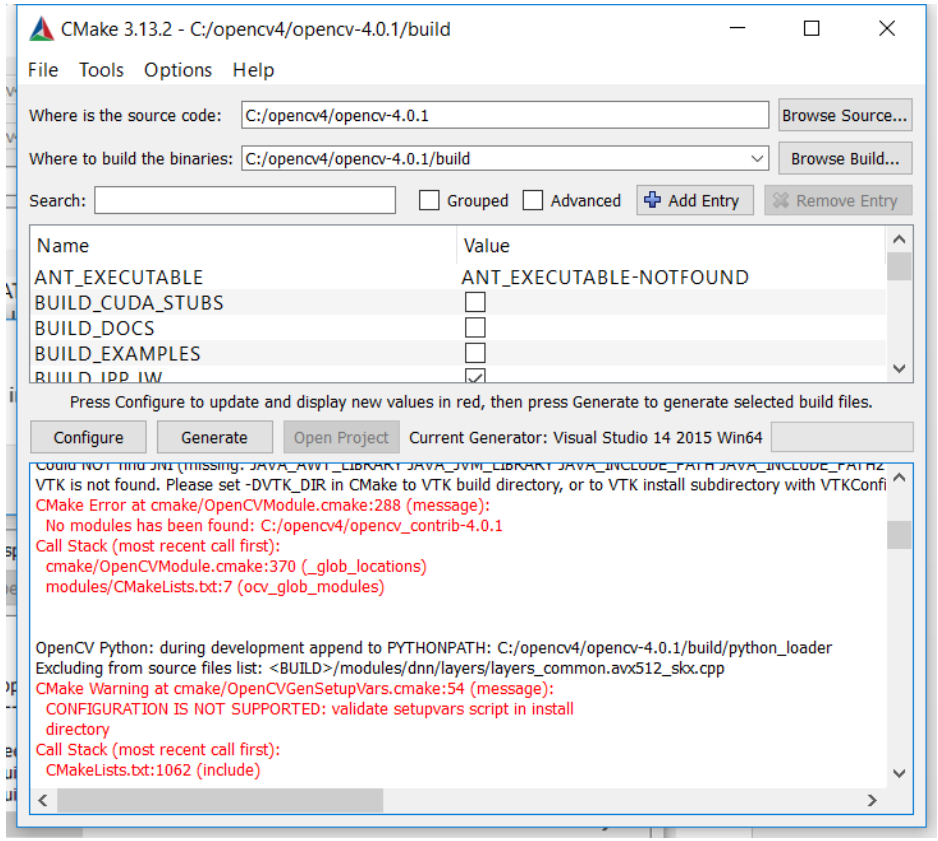


Step3: Build C++ projects using CMake

3.2 additional change (2nd time configuration)

Potential error
Reason: didn't choose the right contrib directory (should have used "/modules" folder"



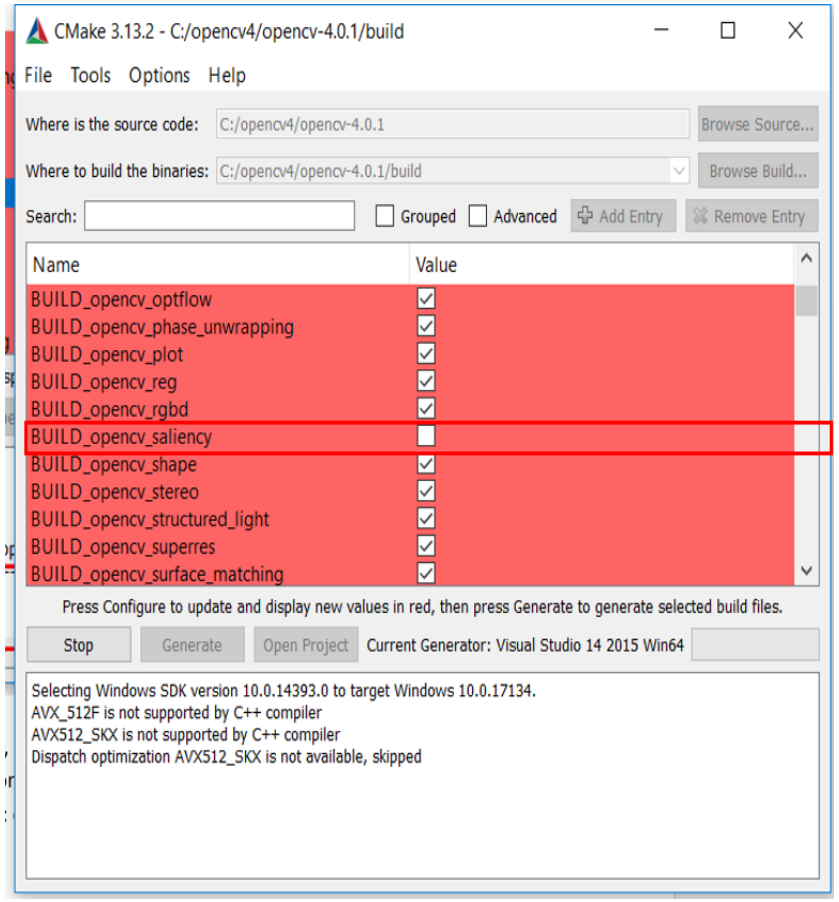


Step3: Build C++ projects using CMake

3.2 additional change (3rd time configuration)

step:

- UNcheck "BUILD_opencv-saliency".Note: (this option didn't show at the first and second configuration; the reason to check that is this open show would make compilation error for WIN10)
- Then click "configure" again, this is the 3rd time configuration



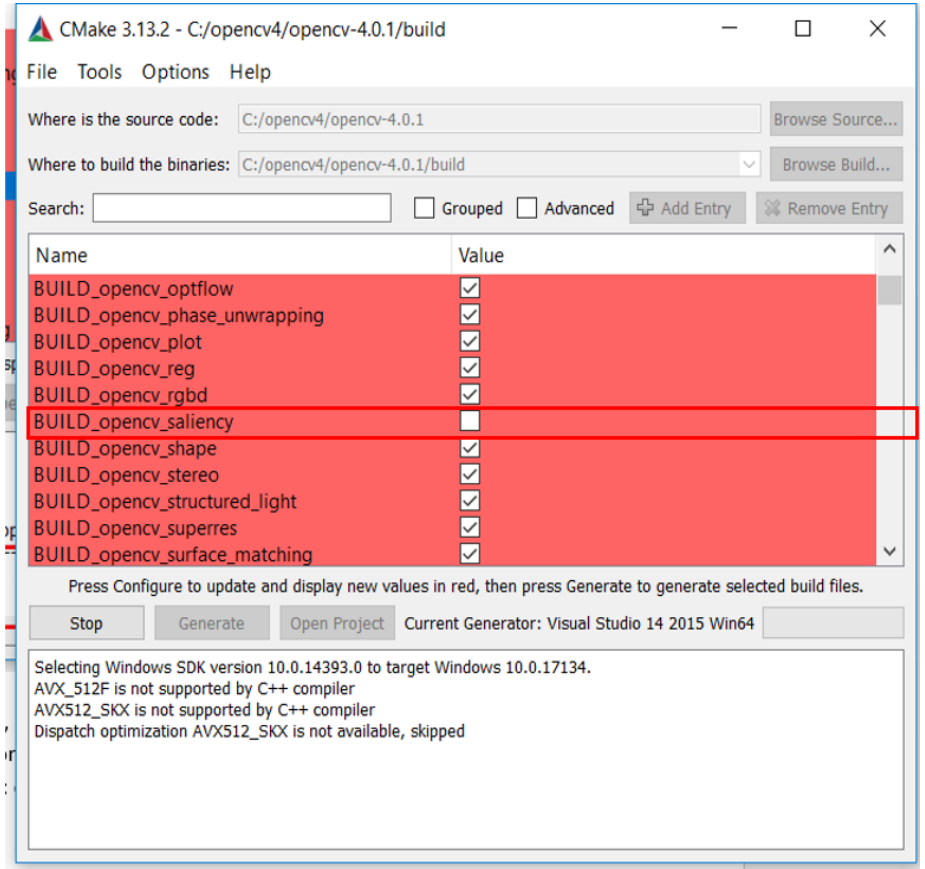
Step3: Build C++ projects using CMake

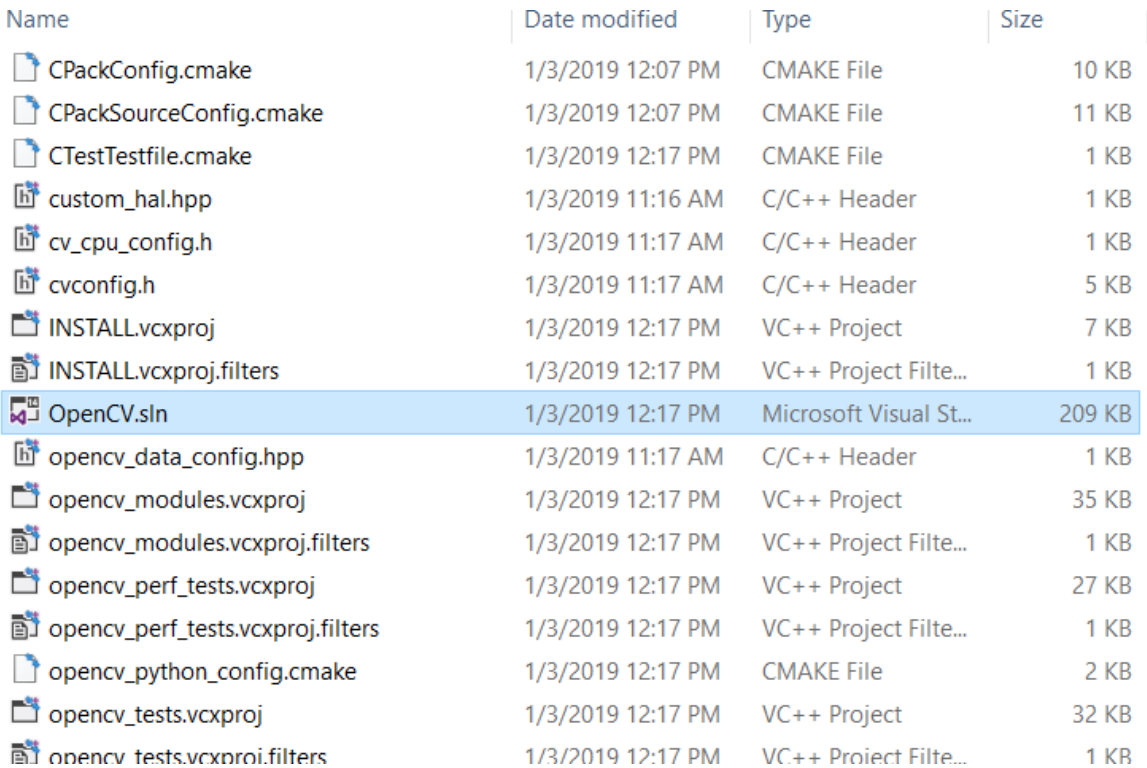
3.3 generate

step:
- click “generate”

Result:
- After clicking “generate”, there would be more file generated in “/build”. Especially, there should be a file called “OpenCV.sln”

Note:
- check point: if “OpenCV.sln” is generated
- Don’t close Cmake yet, might need it later



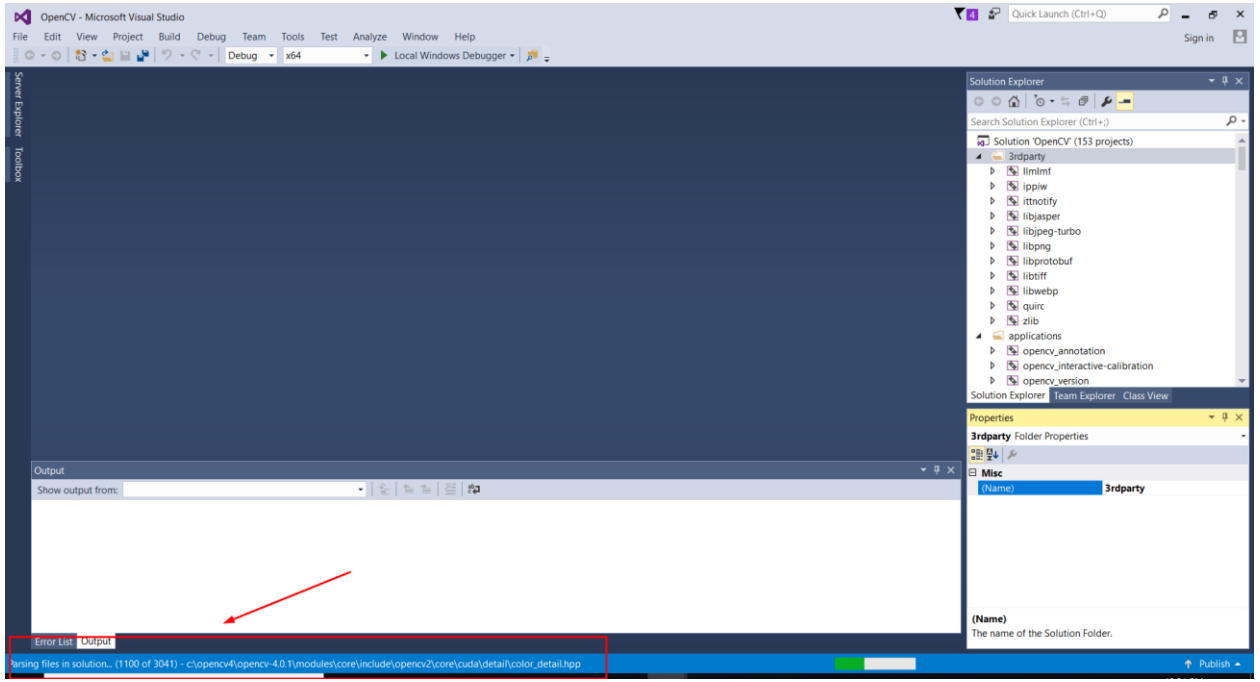


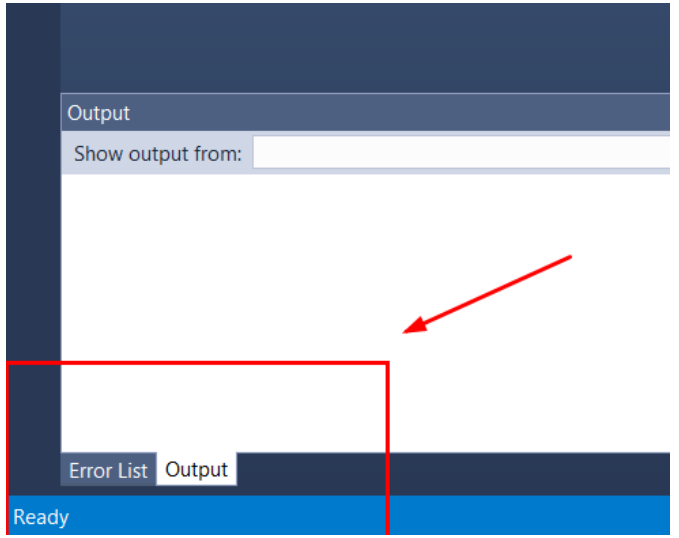
Step4: Compile OpenCV

4.1 build release version for ALL_BUILD

step:

- open “/build/OpenCV.sln”. (It should automatically open in visual studio 2015.) It might take a while initializing, wait until it says “ready”





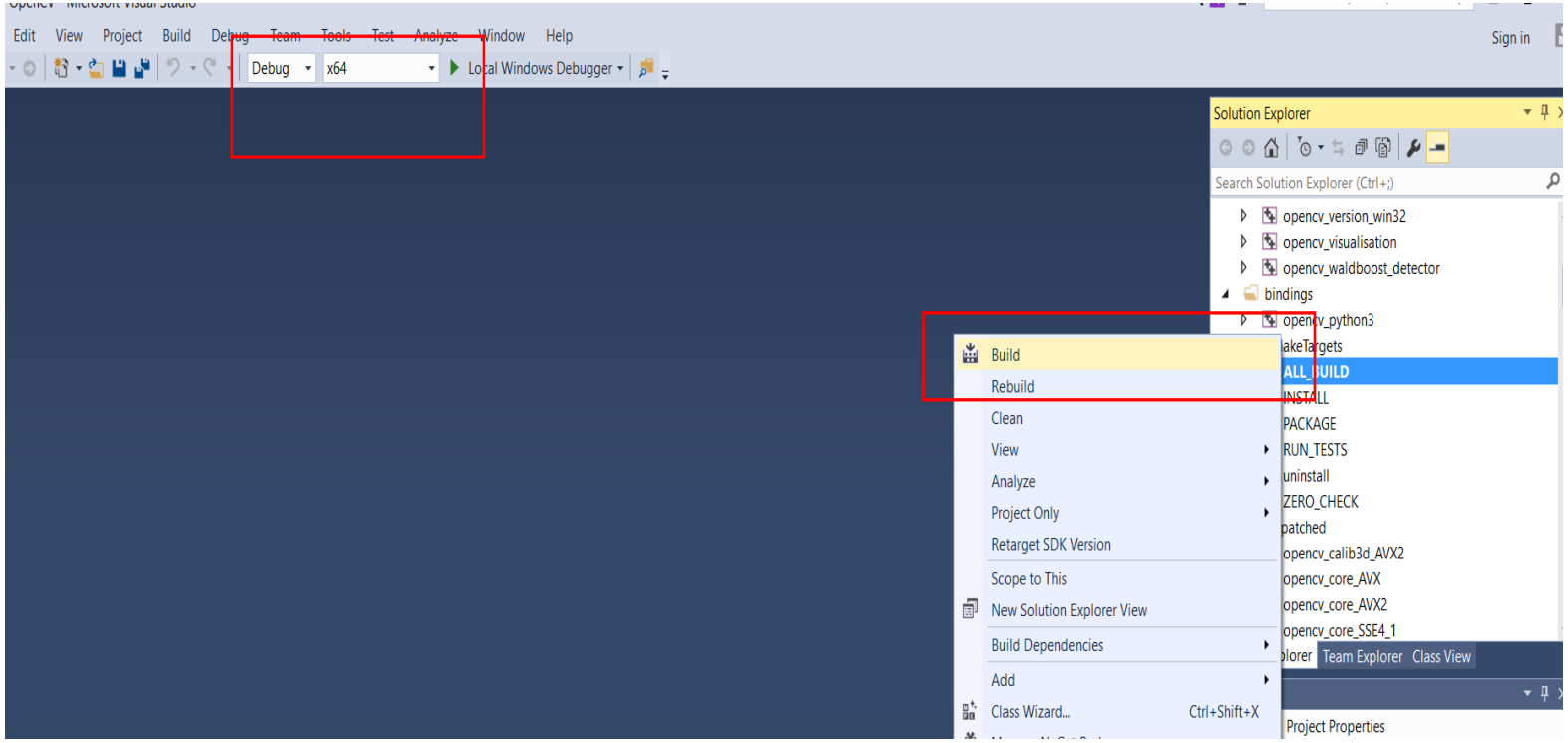
Step4: Compile OpenCV

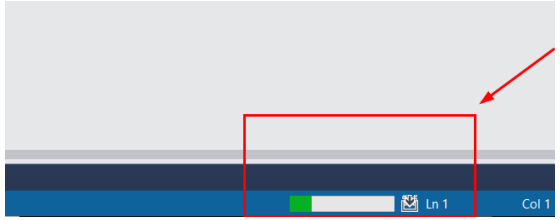
4.1 build debug version for ALL_BUILD

step:

- At the “solution explorer” window, there are should be a project named “ALL_BUILD” under “CMake Targets”.
- Build project: under “debug”, “X64”, right click “ALL_BUILD”, and click “build”. (Note: must be under “debug”, “X64”, if “X64” not exist, see potential error and solution later).

Note: this process may take 10 to 20 minutes. Luckily we have the progress bar to see the progress.



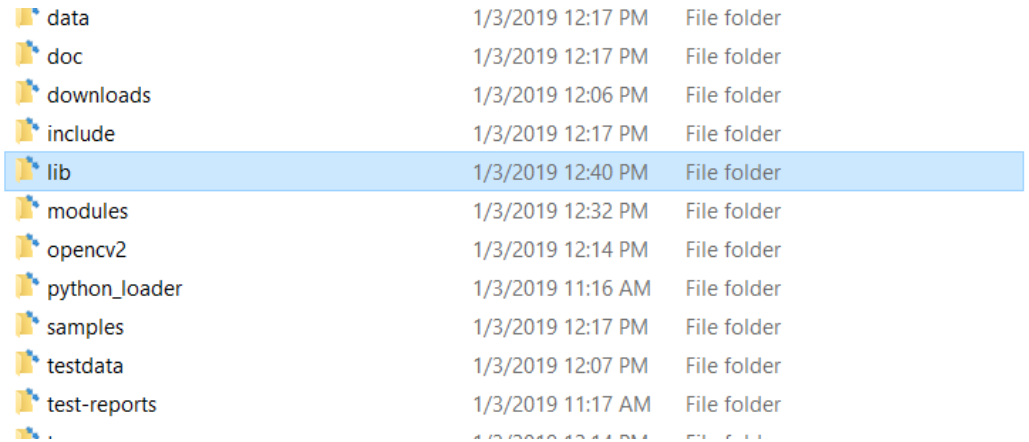


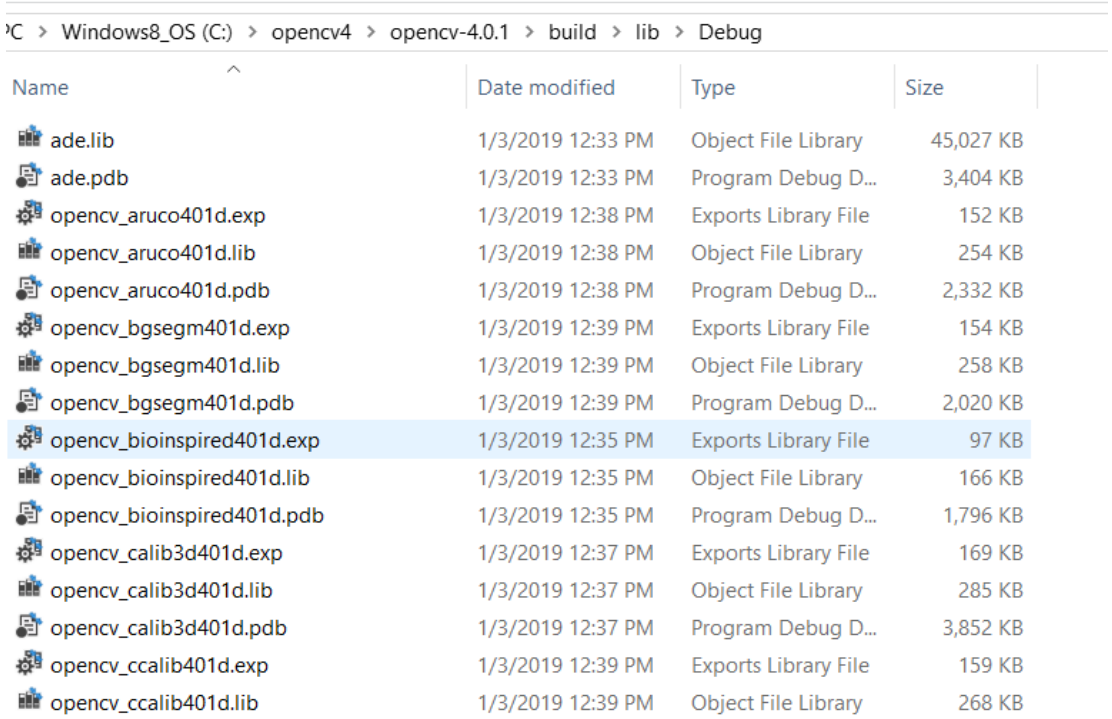
Step4: Compile OpenCV

4.1 build debug version for ALL_BUILD

result:

- After this is done, a new folder named “lib” under “/build” would be created (assuming that the binary address is at “/build”.
- And under “lib”, there is a folder called “debug” with several “.lib” files



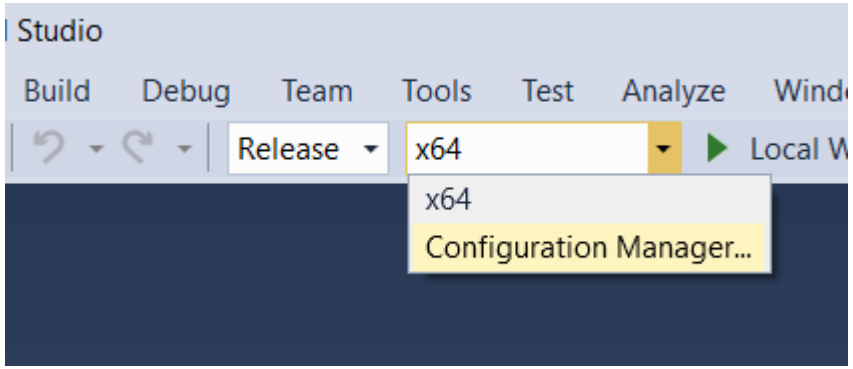


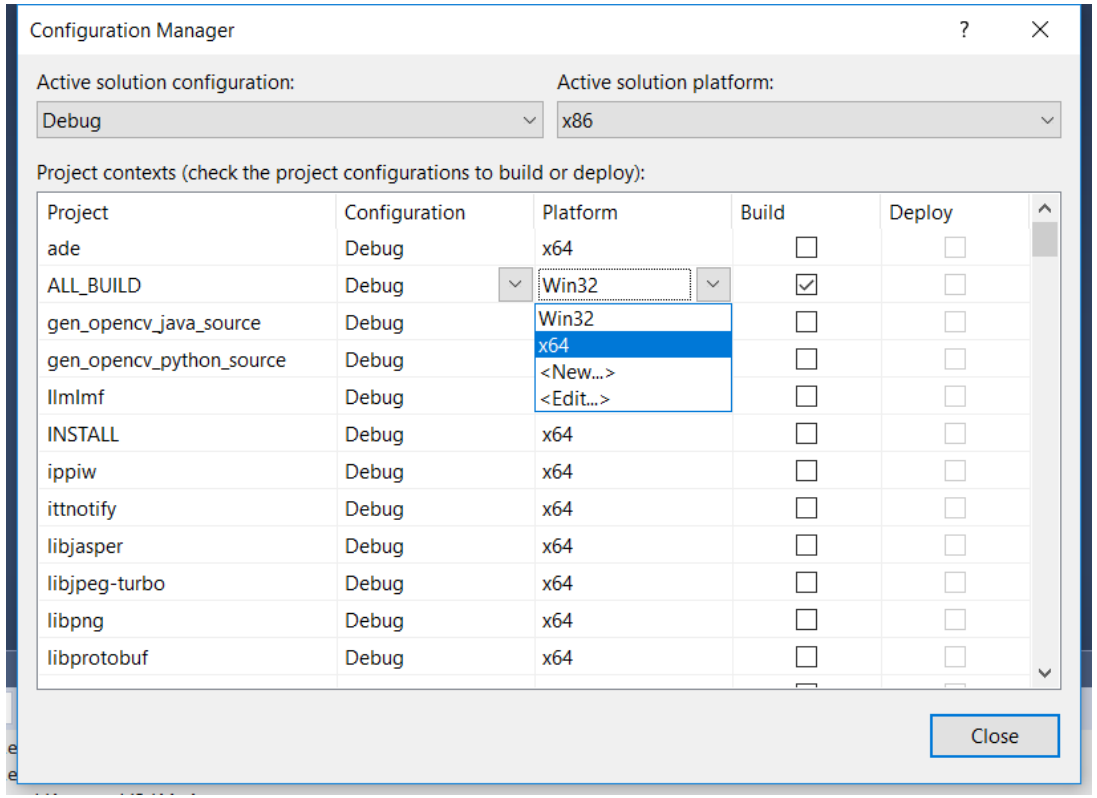
Step4: Compile OpenCV

4.1 build debug version for ALL_BUILD

Potential error:
- Cannot find “X64”

Solution:
Configuration manager -> Platform -> choose “x64” ,or “new”(if “x64 is not there)-> “x64”





Step4: Compile OpenCV

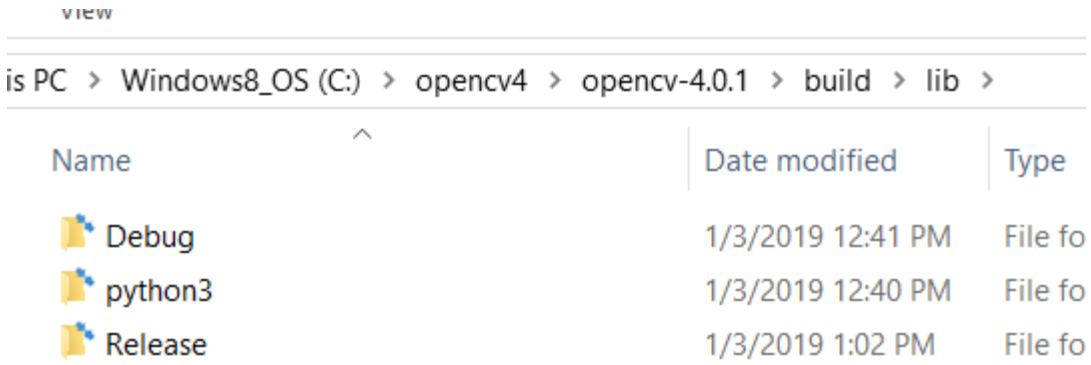
4.2 build release version for ALL_BUILD

step:

- Same “ALL_BUILD” under “CMake Targets”.
- Build project: under “release”, “X64”, right click “ALL_BUILD”, and click “build”.

result:

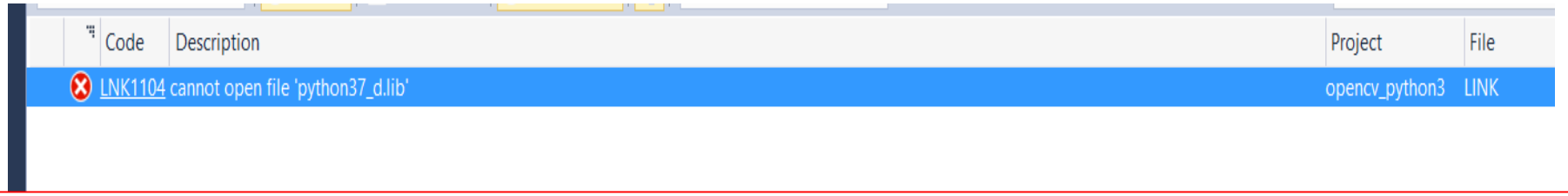
- After it is done, a new folder named “release” would appear in “lib” folder (the python3 might be there as well, don’t worry about that)



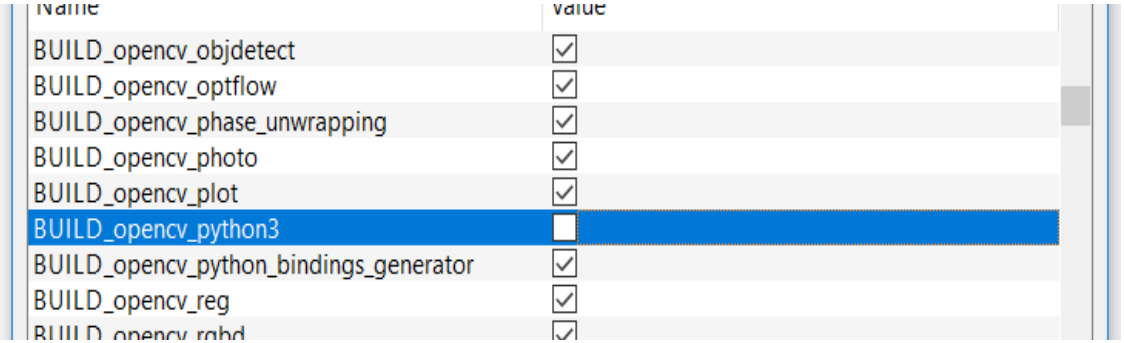
Step4: Compile OpenCV

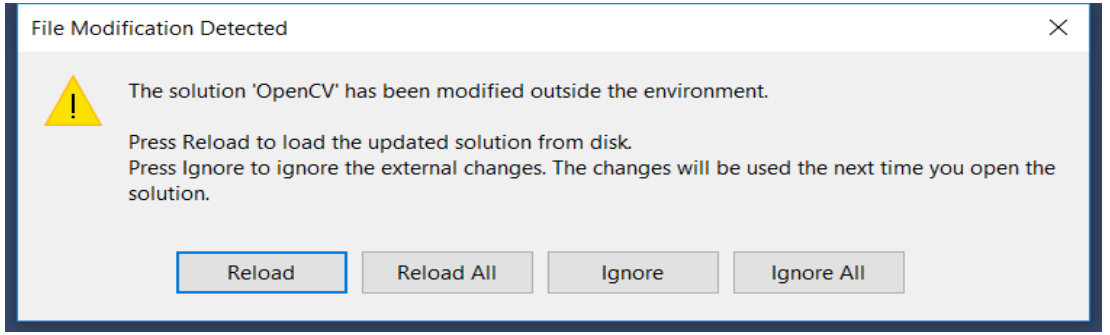
4.3 build release version for “INSTALL”

step:
- for “INSTALL” under “CMake Targets”.
- Build project: under “release”, “X64”, right click “INSTALL”, and click “build”.
- It is likely to encounter an error here.



Solution step:
- Go back to Cmake (that’s why we didn’t close it), uncheck “BUILD_opencv_python3”.
- Click “configure”, then click “generate”
- In Visual studio, it will ask you to reload (so you don’t have to restart the whole program), choose “reload all”, and it would initialize again.
- Try to build again for “INSTALL”.





Step4: Compile OpenCV

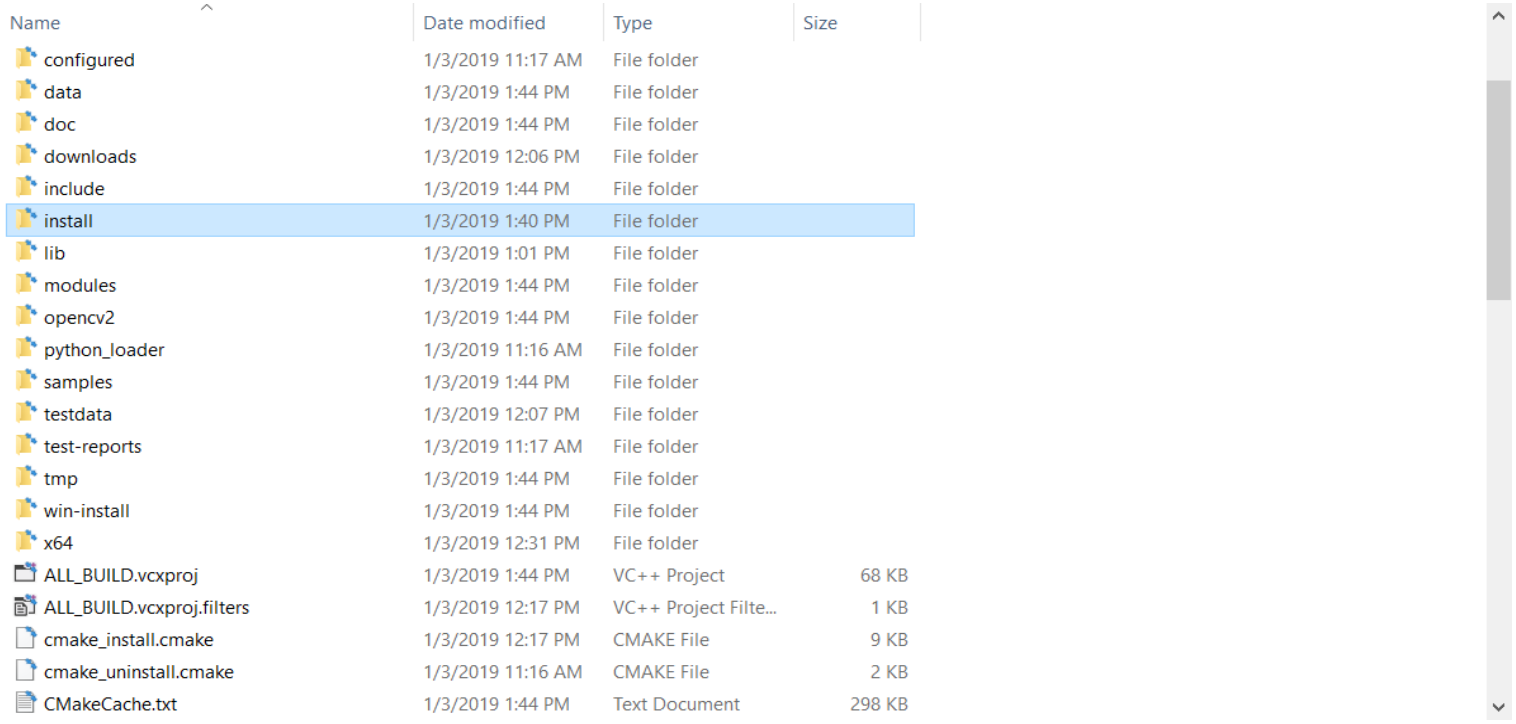
4.4 build debug version for “INSTALL”

step:

- for same “INSTALL” Build project: under “release”, “X64”, right click “INSTALL”, and click “build”.

Result:

- Now we have a new created folder called “install”, later we will use this address for several times



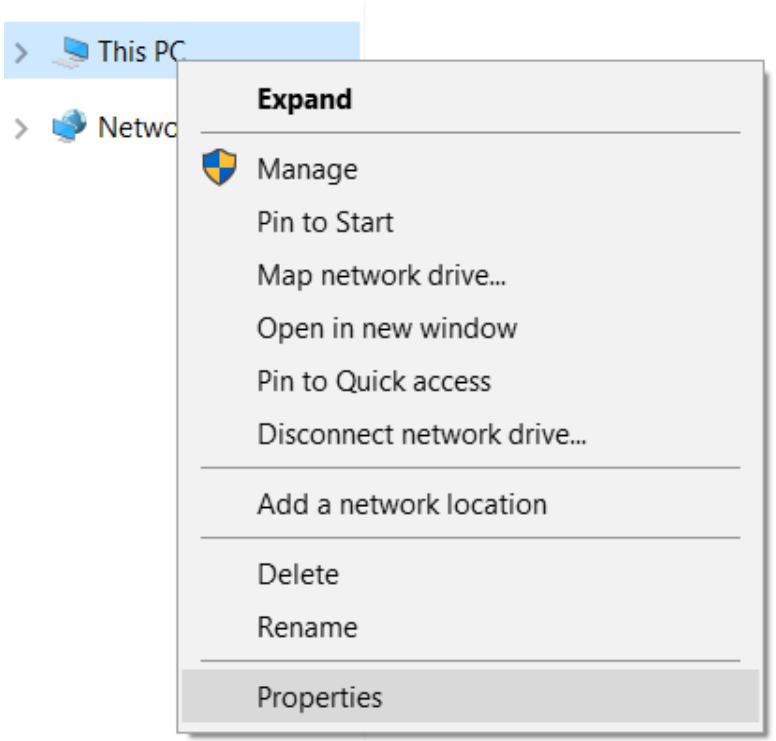
Name	Date modified	Type	Size
configured	1/3/2019 11:17 AM	File folder	
data	1/3/2019 1:44 PM	File folder	
doc	1/3/2019 1:44 PM	File folder	
downloads	1/3/2019 12:06 PM	File folder	
include	1/3/2019 1:44 PM	File folder	
install	1/3/2019 1:40 PM	File folder	
lib	1/3/2019 1:01 PM	File folder	
modules	1/3/2019 1:44 PM	File folder	
opencv2	1/3/2019 1:44 PM	File folder	
python_loader	1/3/2019 11:16 AM	File folder	
samples	1/3/2019 1:44 PM	File folder	
testdata	1/3/2019 12:07 PM	File folder	
test-reports	1/3/2019 11:17 AM	File folder	
tmp	1/3/2019 1:44 PM	File folder	
win-install	1/3/2019 1:44 PM	File folder	
x64	1/3/2019 12:31 PM	File folder	
ALL_BUILD.vcxproj	1/3/2019 1:44 PM	VC++ Project	68 KB
ALL_BUILD.vcxproj.filters	1/3/2019 12:17 PM	VC++ Project Filte...	1 KB
cmake_install.cmake	1/3/2019 12:17 PM	CMAKE File	9 KB
cmake_uninstall.cmake	1/3/2019 11:16 AM	CMAKE File	2 KB
CMakeCache.txt	1/3/2019 1:44 PM	Text Document	298 KB

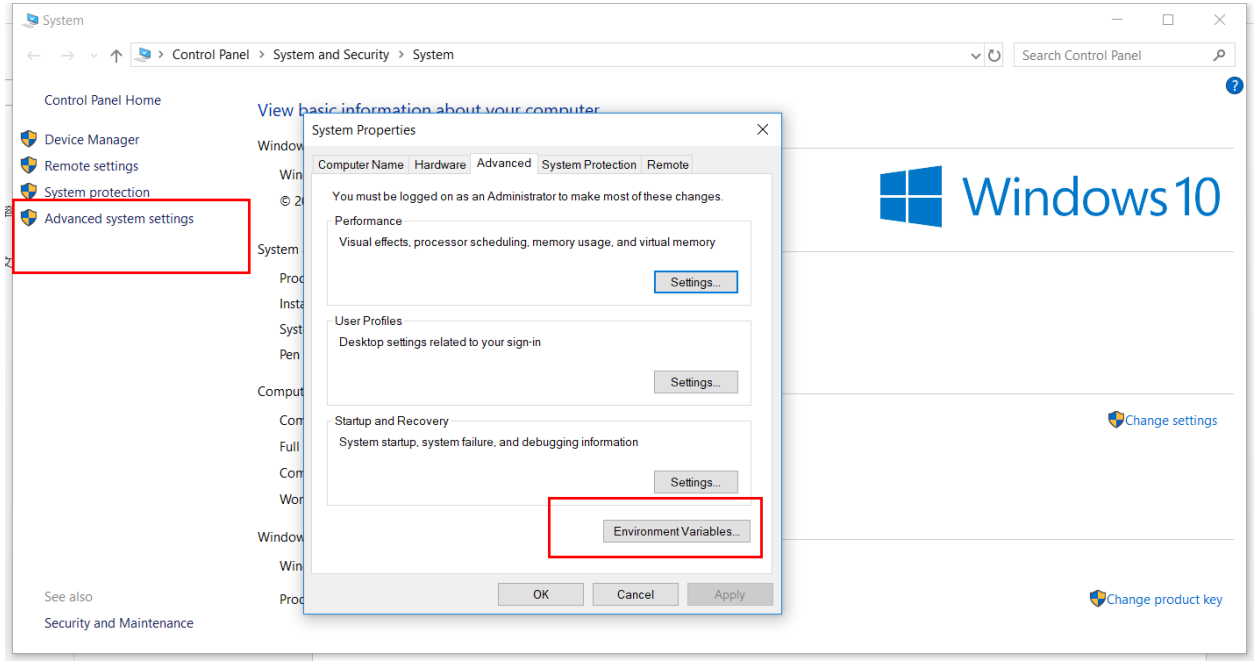
Step5: Update environment variables

5.1 environment variable – PATH

step:

- Right click “This PC”, and click “properties”
- Click “Advanced system settings”, then click “Environment Variables”
- From System variables, edit PATH,



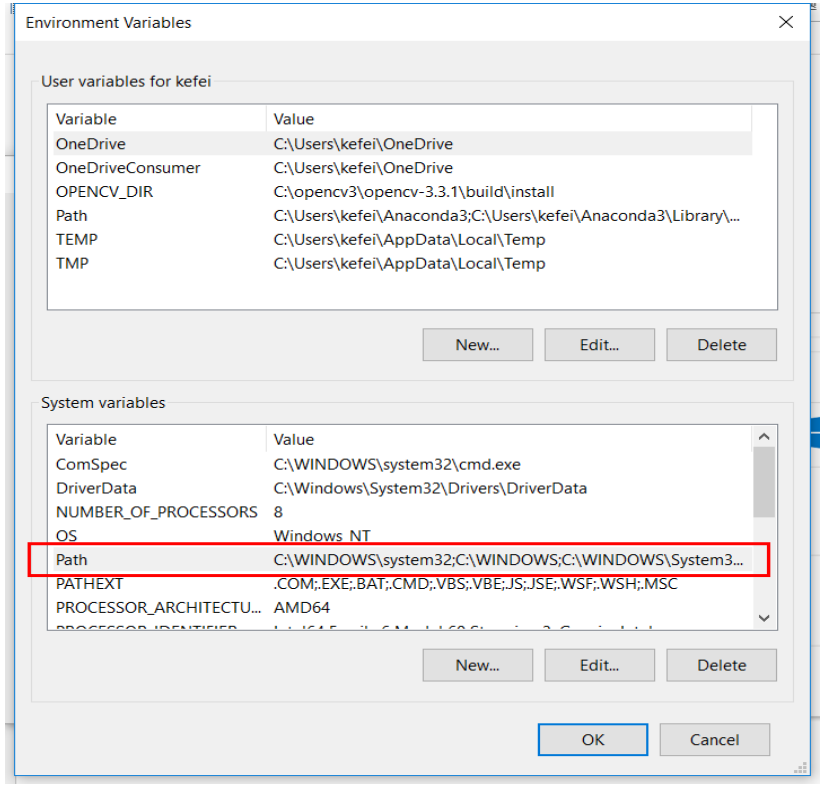


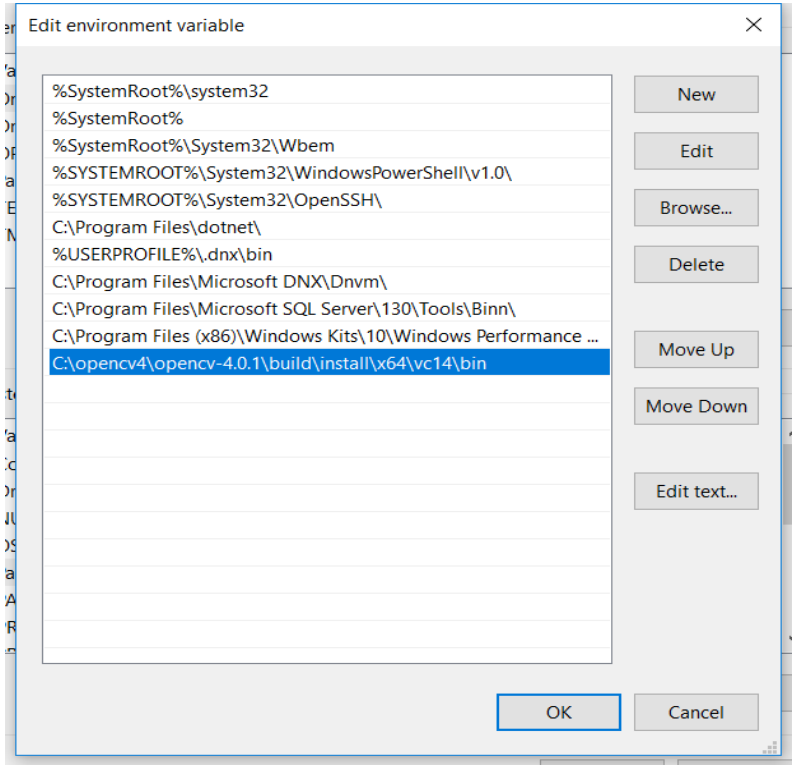
Step5: Update environment variables

5.1 environment variable – PATH

step:

- From System variables, edit PATH
- We refer the address “C:\opencv4\opencv-4.0.1\” as “{OPENCV_PATH}”,(depends on where you extract your opencv resource file at the beginning). We add a new path “{OPENCV_PATH}\build\install\x64\vc14\bin”, for me it is “C:\opencv4\opencv-4.0.1\build\install\x64\vc14\bin”



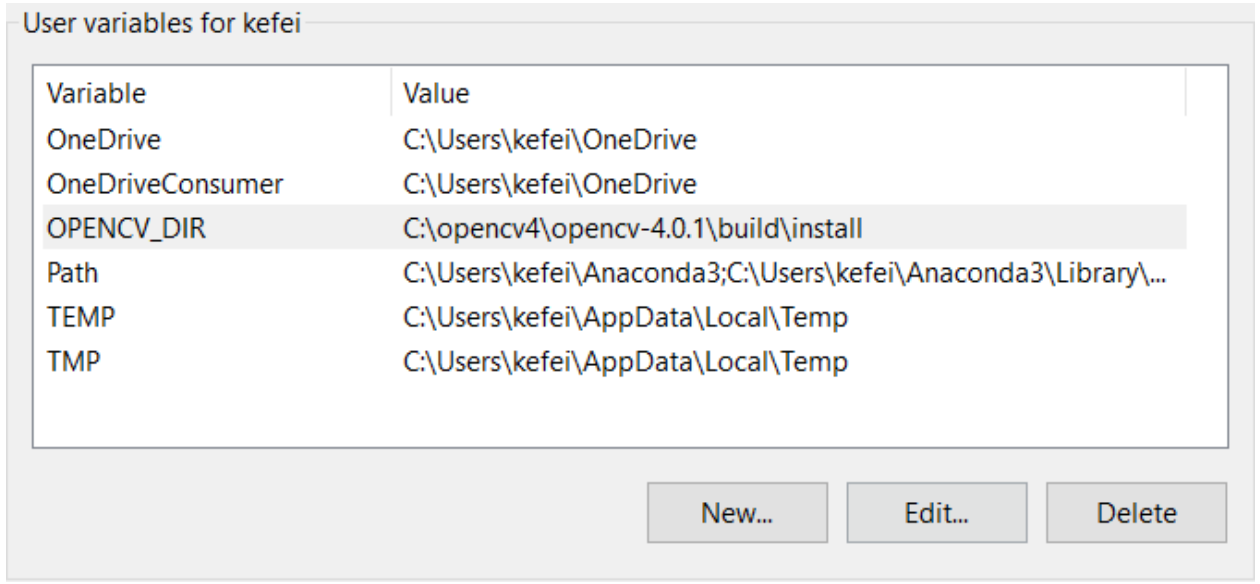


Step5: Update environment variables

5.2 user environment variable – OPENCV_DIR

step:

- Update Variable name: OPENCV_DIR
- Update Value: “{OPENCV_PATH}\build\install “, for this tutorial example it is “C:\opencv4\opencv-4.0.1\build\install”

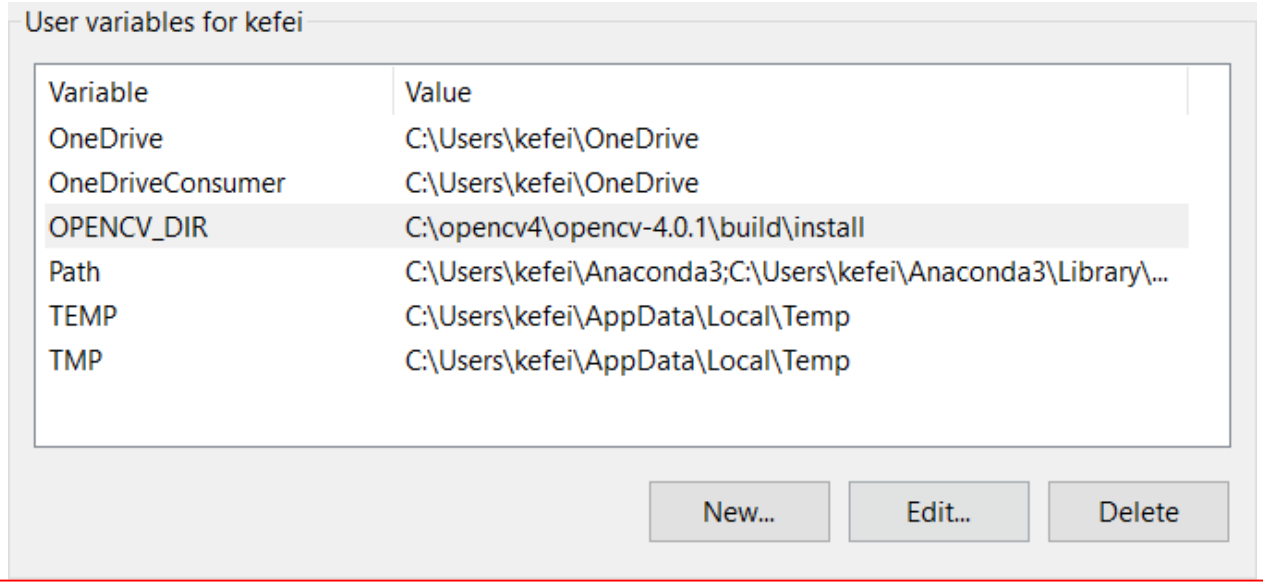


Step5: Update environment variables

5.2 user environment variable – OPENCV_DIR

step:

- Update Variable name: OPENCV_DIR
- Update Value: “{OPENCV_PATH}\build\install “, for this tutorial example it is “C:\opencv4\opencv-4.0.1\build\install”



note:

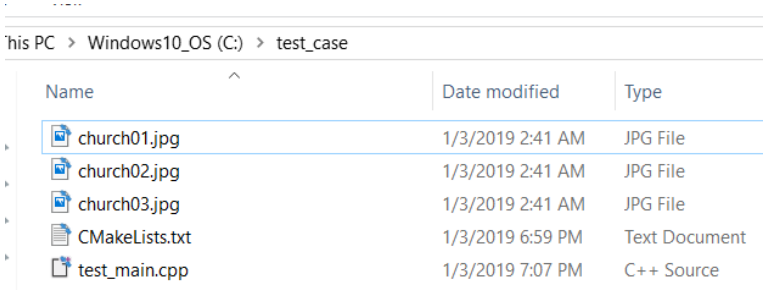
- Recommend to delete environment variable from other OpenCV versions
- Recommend to restart the system after updating environment variables

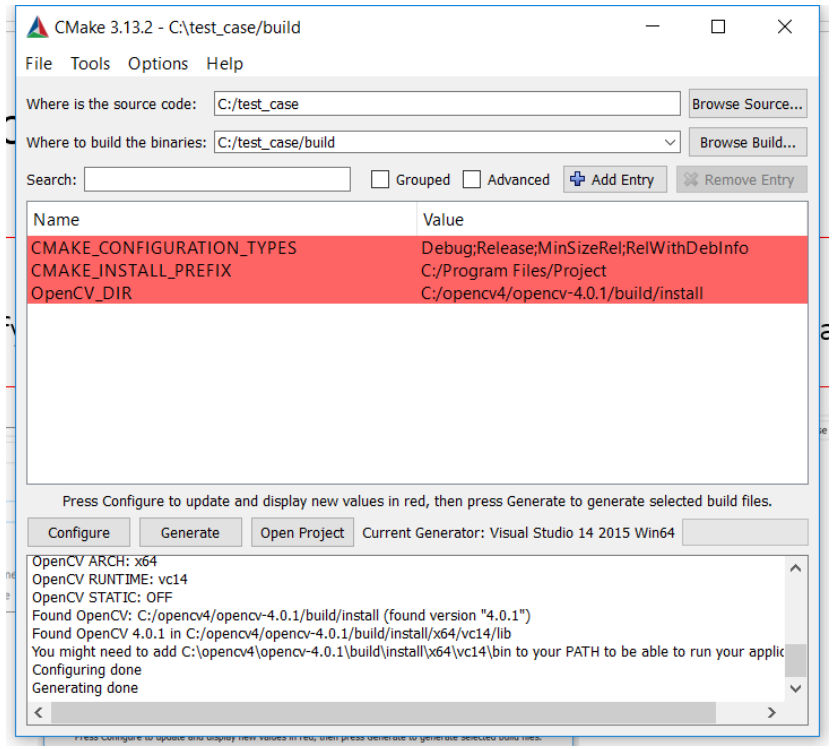
Step6: Test the compiled OpenCV (in C++)

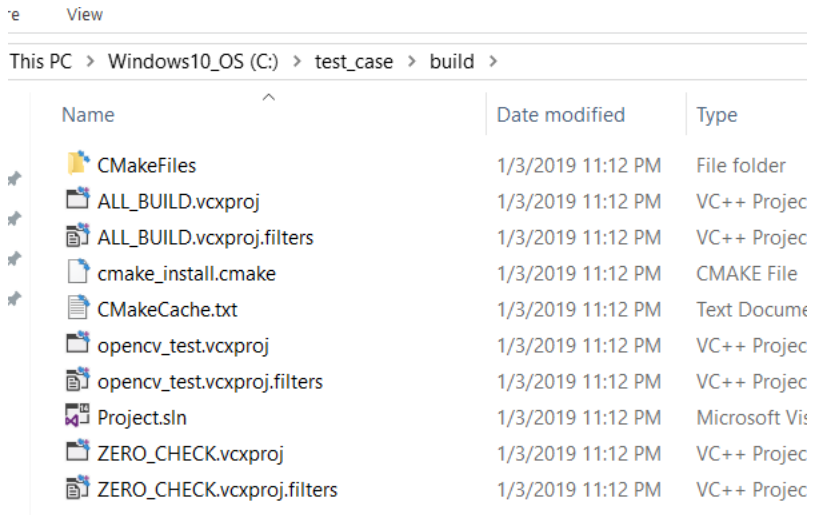
6.1 CMake build

step:

- Put test files folder in the same disk as OpenCV
- CMake build: 1. specify source code and build directories, 2. configure, 3. generate project





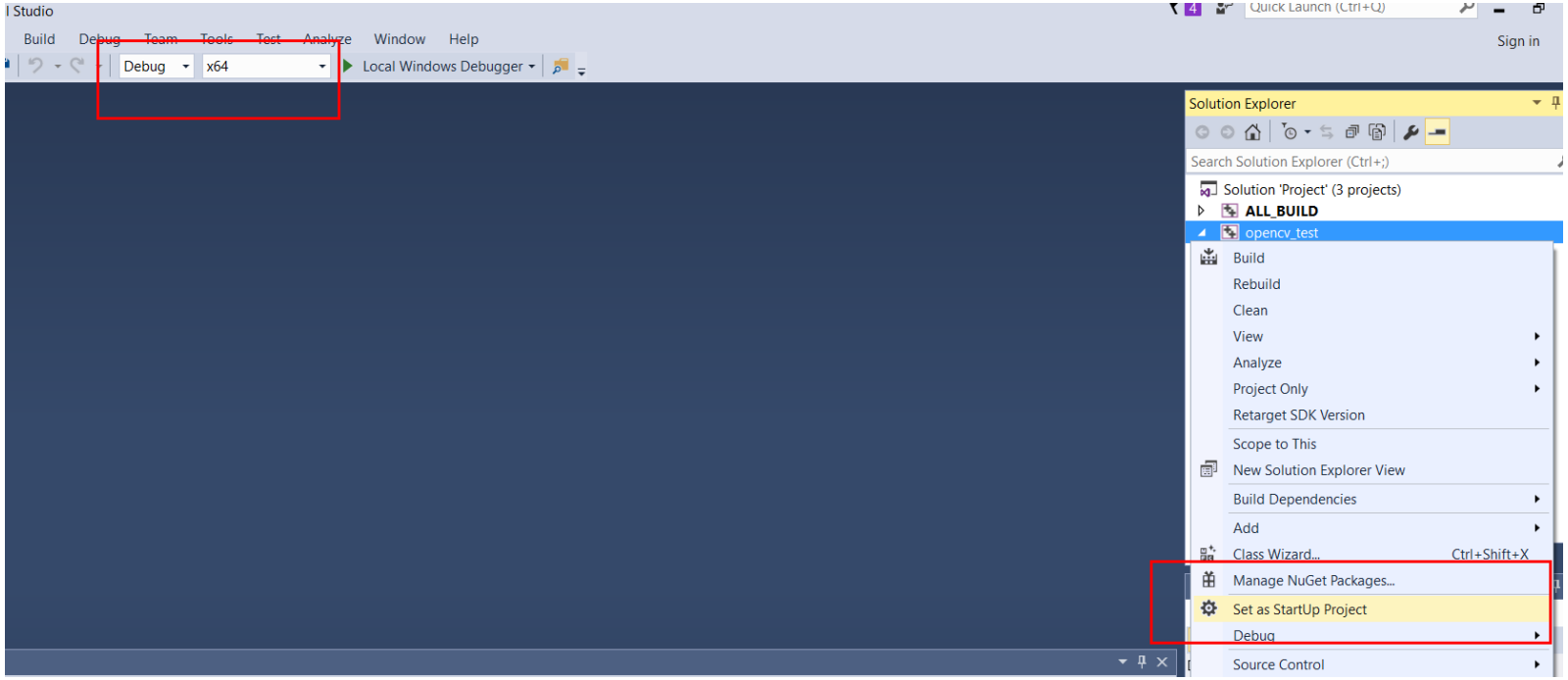


Step6: Test the compiled OpenCV (in C++)

6.2 build and execute in Visual Studio

step:

- Open “Project.sln”
- Under “debug”, “x64”, right click “opencv_test”, click “Set as StartUp Project”. (it will turn bold)
- Press “F5” to debug



Step6: Test the compiled OpenCV (in C++)

6.2 build and execute in Visual Studio

Result: as shown
Note: this test file is an application of SURF for interest points detection, further development for object recognition, image registration, visual tracking, 3D reconstruction, etc. More information can refer to “cv::xfeatures2d::SurfFeatureDetector”