Documentation: Installation of OpenCV with CUDA using Microsoft Visual Studio in Windows

Prepared by: Lim Jun Jie

A. Prerequisites

1. Installed Python
2. Installed NVIDIA GPU drivers and CUDA development Kit
3. Installed Microsoft Visual Studio and its relative Build Tools
4. Installed C-Make

B. Installation of prerequisites

1. Installation of OpenCV and OpenCV-contrib
   - Download zip file of OpenCV from https://opencv.org/releases/, choose "Sources" under the version of OpenCV you want to download.
   - Download zip file of OpenCV-contrib from https://github.com/opencv/opencv_contrib/releases, choose the relative version with the zip file of OpenCV (opencv-4.1.2 with opencv_contrib-4.1.2)
2. Installation of Python and CUDA development Kit
   - Refer to this YouTube link https://www.youtube.com/watch?v=qrkEYf-YDyI&list=PLjy4p-07OYzulelvJ5KVaT2pDlxivl_BN if you haven't set up for CUDA development Kit
3. Installation of Microsoft Visual Studio and Build Tools of Visual Studio
   - Installation link: https://visualstudio.microsoft.com/downloads/
4. Installation of C-Make
   - Installation link: https://cmake.org/download/

C. Build OpenCV using C-Make

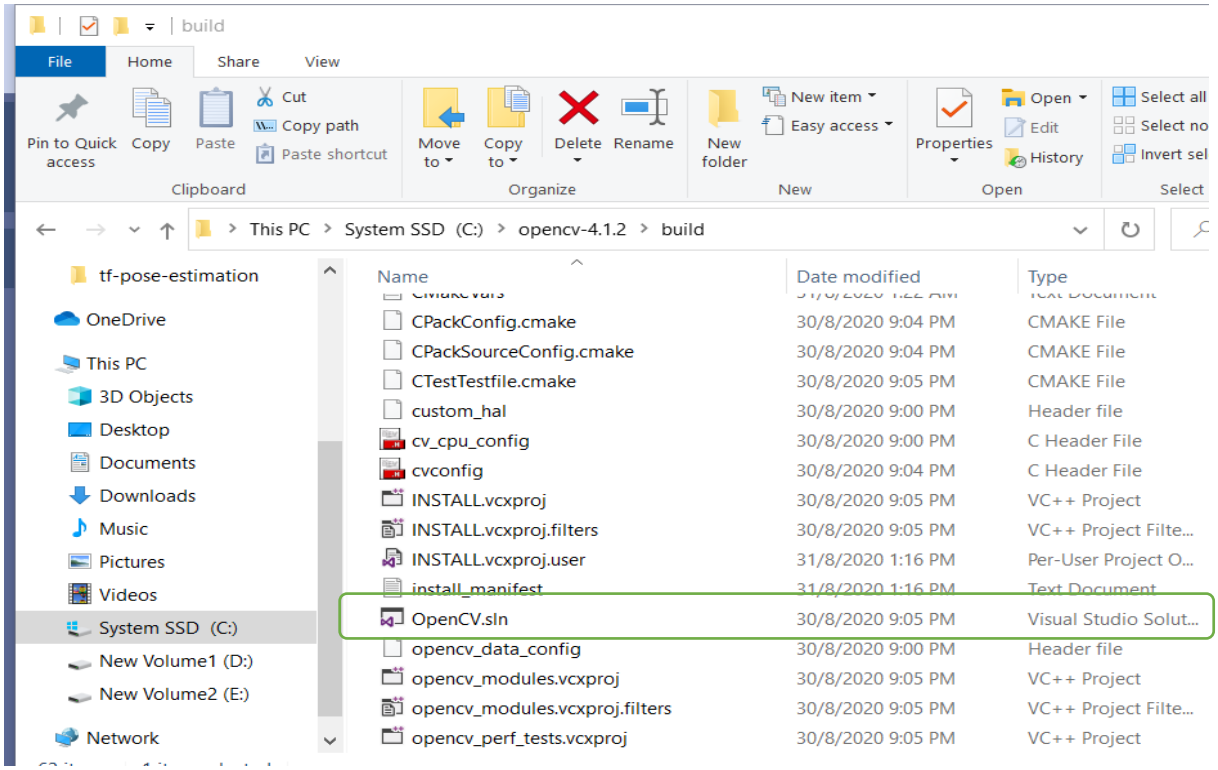
1. Extracting OpenCV and OpenCV-contrib to any location you want, then open CMake-gui, set the
   - Where is the source code: <path-to-file>/opencv-x.x.x
   - Where to build the binaries: <path-to-file>/opencv-x.x.x/build
2. Click Configure, then
   - Specify the generator for the project according to your Visual Studio version
   - Optional platform for generator is depending on your OS, for example x64
   - Choose "Use default native compilers"

- Click Finish to start generating

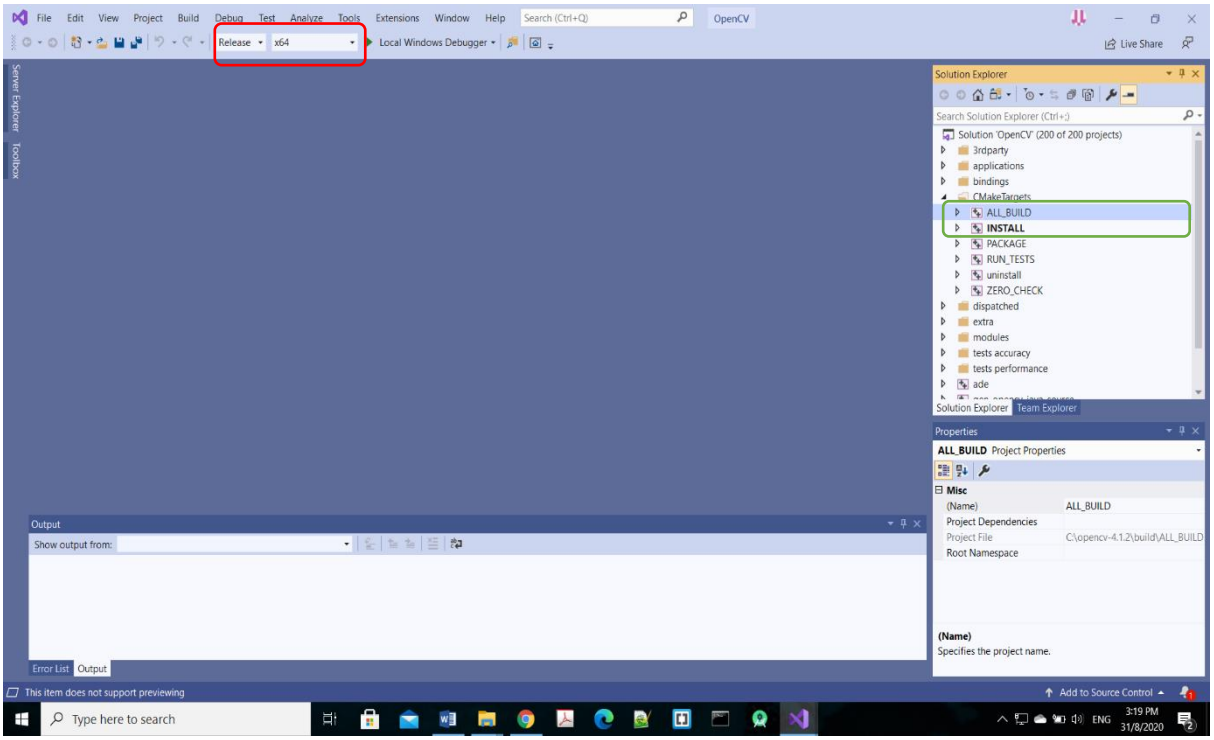
3. After the first generating, lots of red sign will be seen, it's ok, just TICK "WITH_CUDA", then click configure again.
4. Then error will occur, it's ok, now
   - TICK "CUDA_FAT_MATH"
   - Navigate to "OPENCV_EXTRA_MODULES_PATH" and set it to <path-to-file>/opencv_contrib-x.x.x/modules
5. Configure again, then click Generate
6. After done generating, go to search "x64 Native Tools Command Prompt for VS 2019"
   - Change your directory to <path-to-file>/opencv-x.x.x/build
   - Type "msbuild INSTALL.vcxproj /p:Configuration=Release" in this Command Prompt
   - Waiting for the build, it takes about 4 hours for me to complete the build

D. Configuration in Microsoft Visual Studio

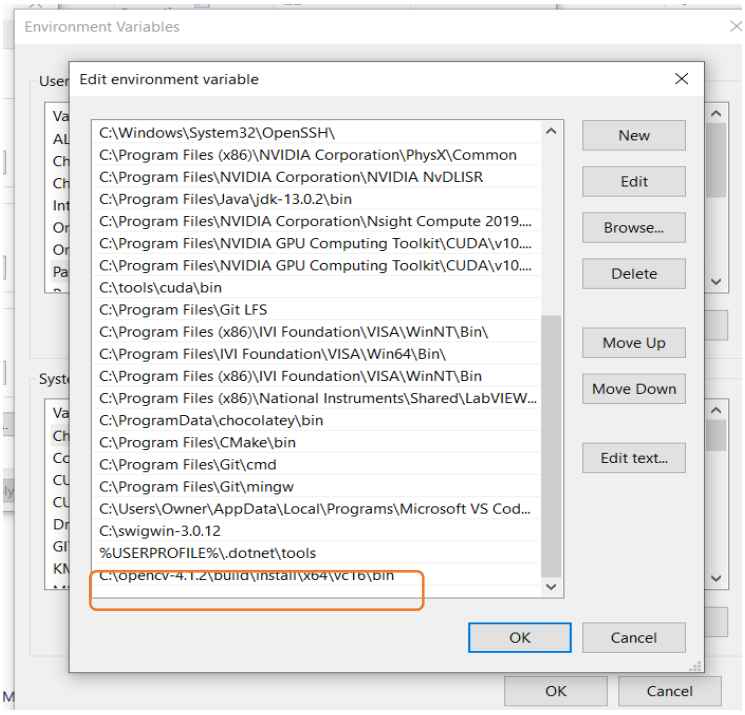
1. After that, go to <path-to-file>/opencv-x.x.x/build, open OpenCV.sln with Microsoft Visual Studio.



- Set your VS environment as Release & x64 (red box)
- Then, right click “ALL_BUILD”, click “Set as startup project”, then click “Build”, it takes long time too. (green box)
- After that, repeat the same process by choosing INSTALL. (green box)



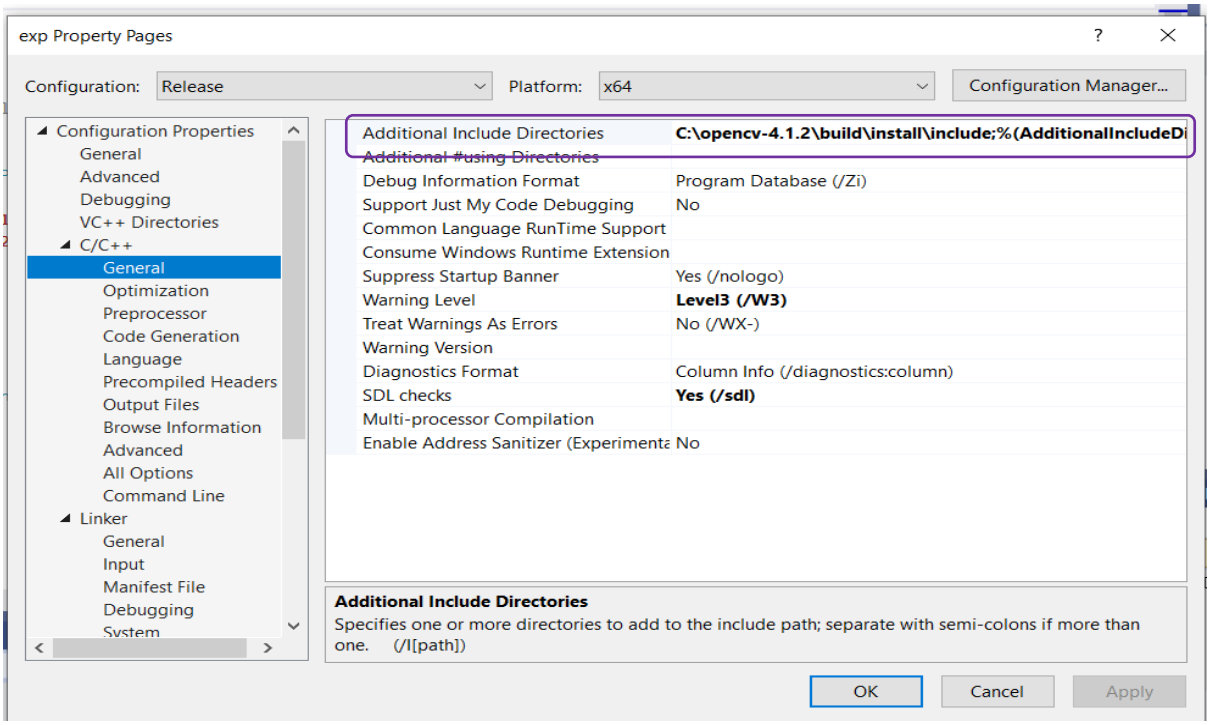
E. Adding the path to Environment Variable

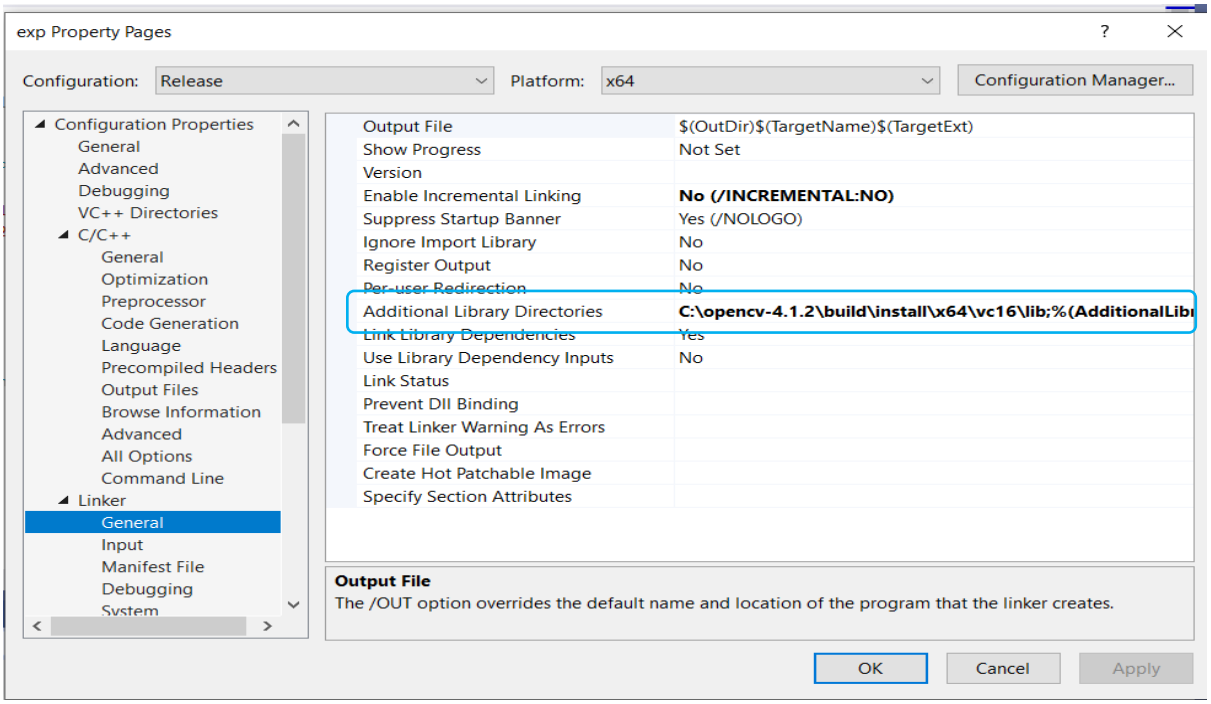


Add path <file-to-path>/opencv-x.x.x/build/install/x64/vc16/bin

F. Setting the dependencies of Microsoft Visual Studio

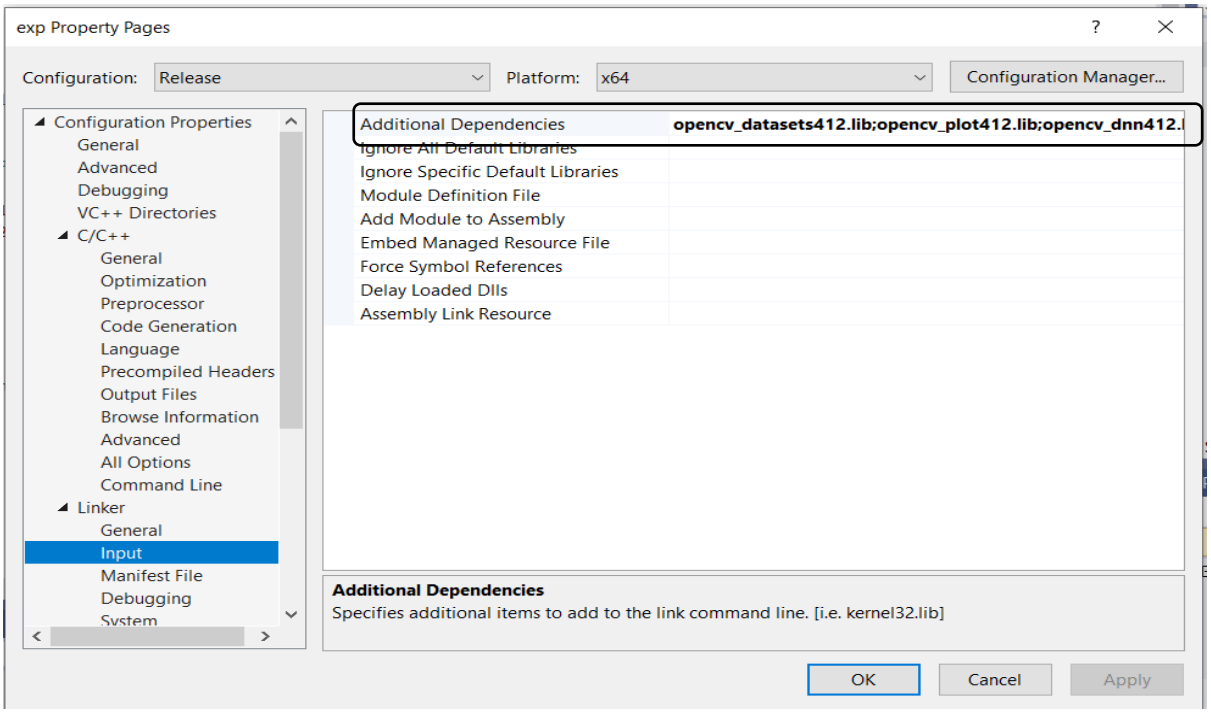
- Now, you can create new OpenCV project by setting the relevant dependencies





Set path as <path-to-file>/opencv-x.x.x/build/install/include (Purple box)

Set path as <path-to-file>/opencv-x.x.x/build/install/x64/vc16/lib (Light blue box)



Add all *.lib files under <path-to-file>/opencv-x.x.x/build/install/x64/vc16/lib (black box)

For the details of part E and F, can refer to the documentation of setting up the OpenCV using Microsoft Visual Studio without CUDA support in my github link https://github.com/JJLim99/OpenCV-with-and-without-CUDA-using-Micosoft_VS.git

Main Reference: https://www.youtube.com/watch?v=TT3_dlPL4vo