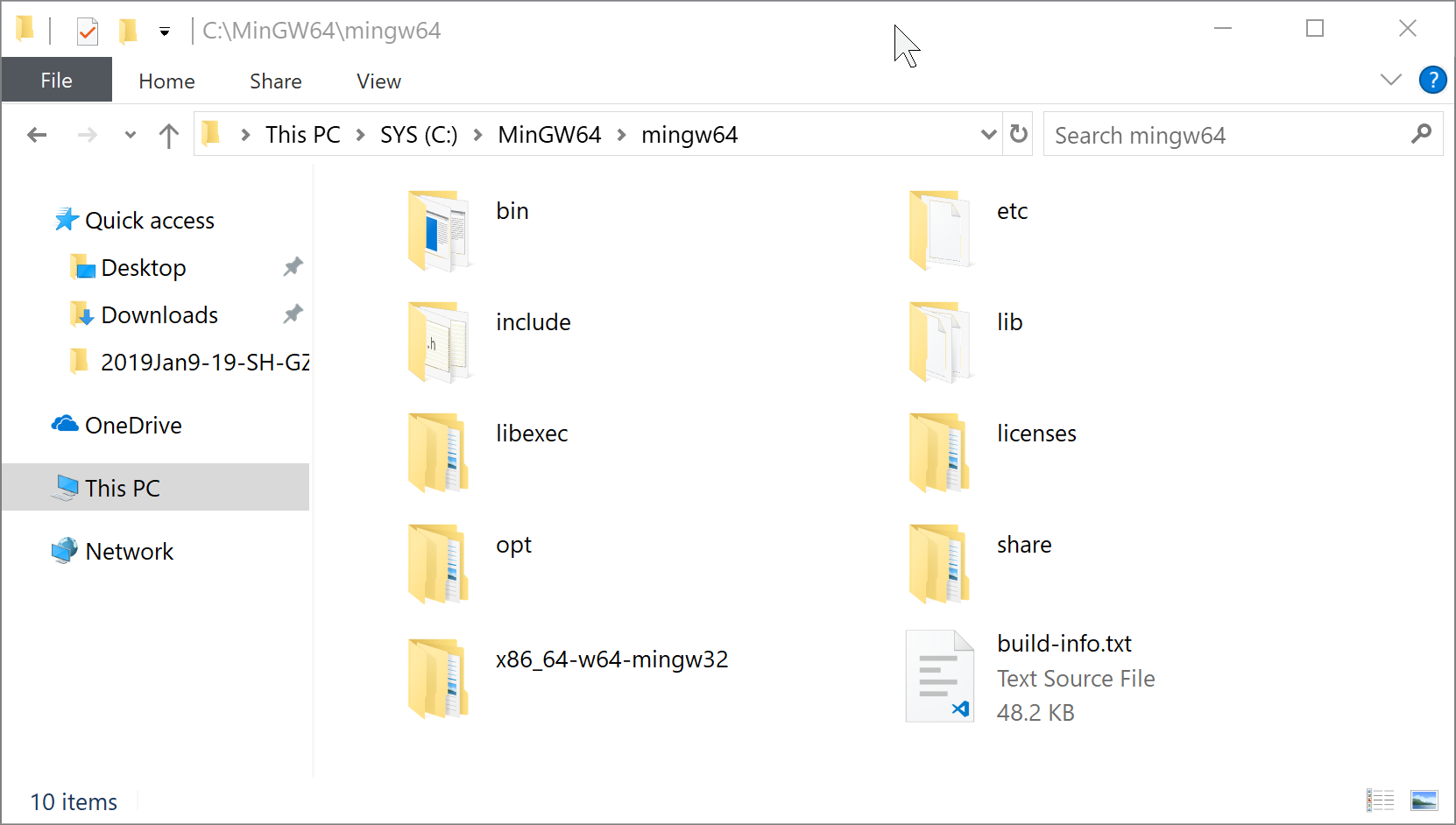
Setup VSCode,MinGW,OpenCV

1. Install VSCode with C++ Plugin
2. Download & Install MinGW;



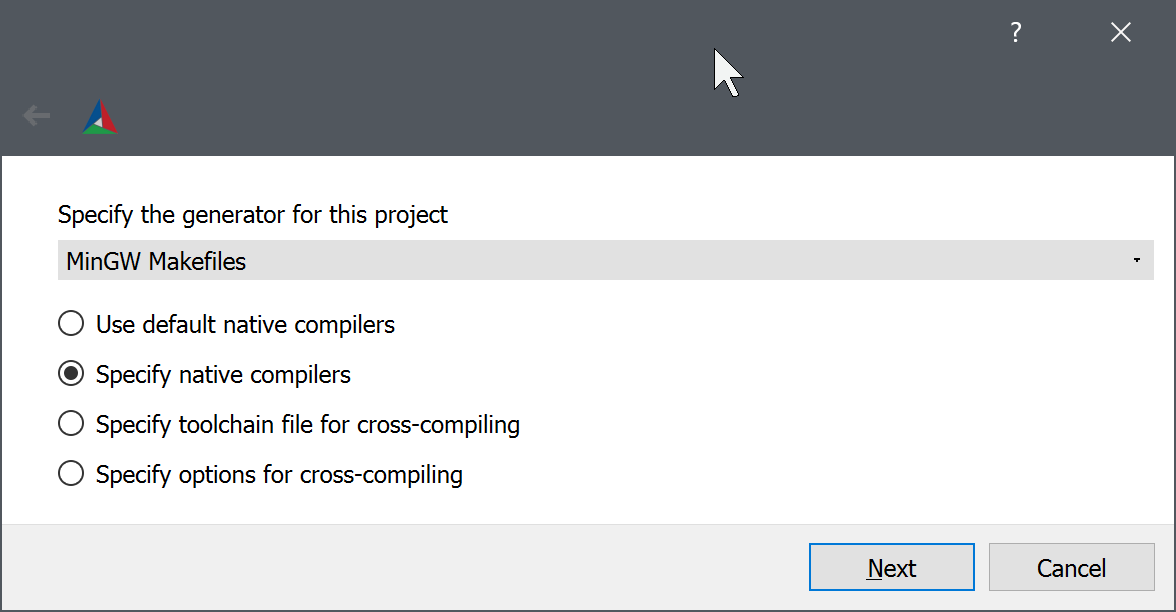
1. Download & Compile OpenCV
   1. Download openCV4.01

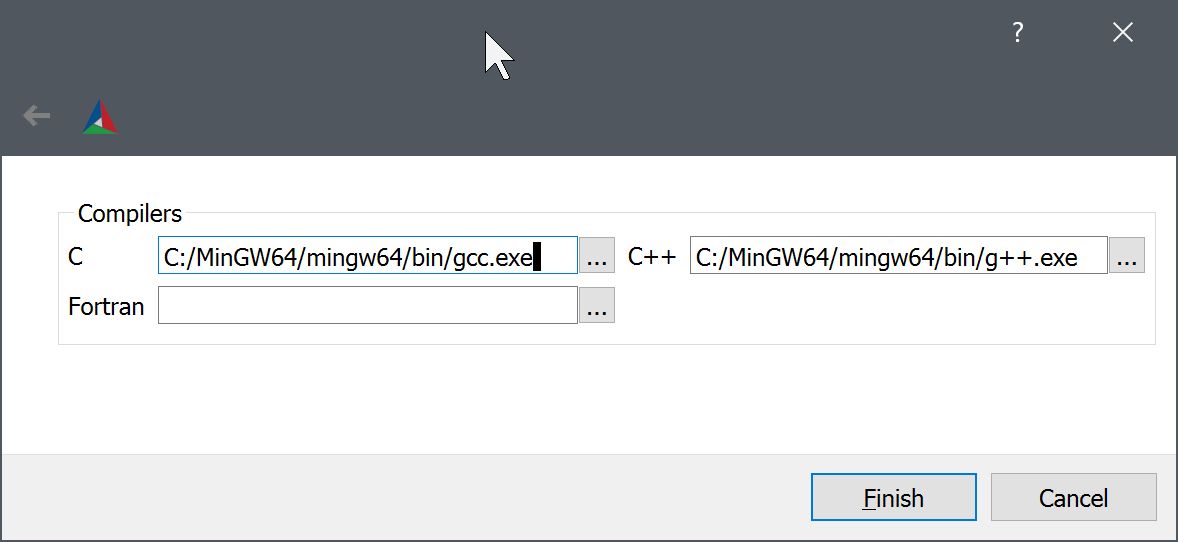
打开 cmake-gui，设置源码和生成路径：

* Where is the source code: E:/opencv\_341/opencv/sources
* Where to build the binaries: E:/opencv\_341/opencv\_mingw64\_build

点击 Configure，设置编译器

* Specify the generator for this project: MinGW Makefiles
* Specify native compilers
* Next
* Compilers C: E:\MinGW-w64\x64-4.8.1-release-posix-seh-rev5\mingw64\bin\gcc.exe
* Compilers C++: E:\MinGW-w64\x64-4.8.1-release-posix-seh-rev5\mingw64\bin\g++.exe
* Finish

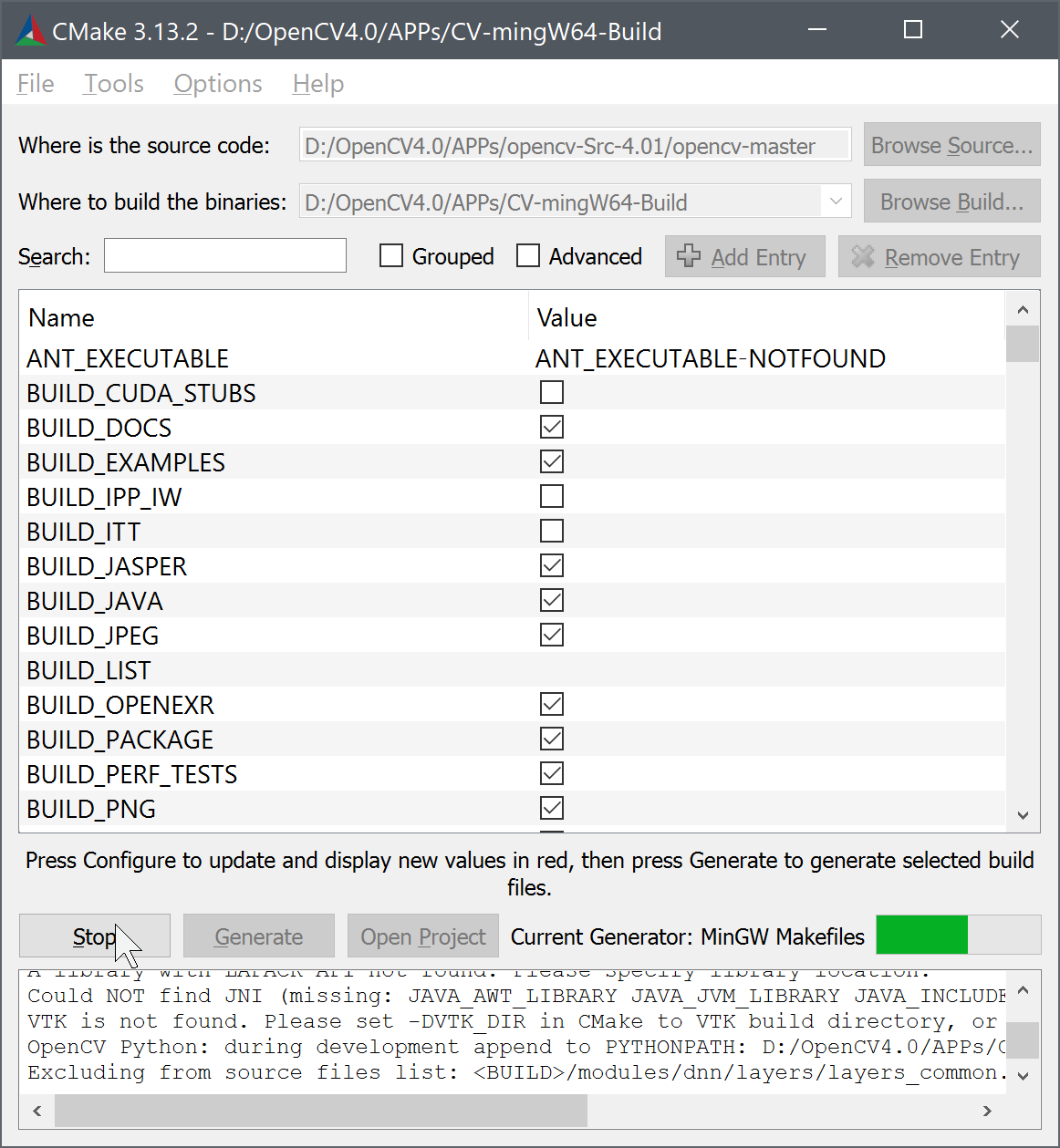


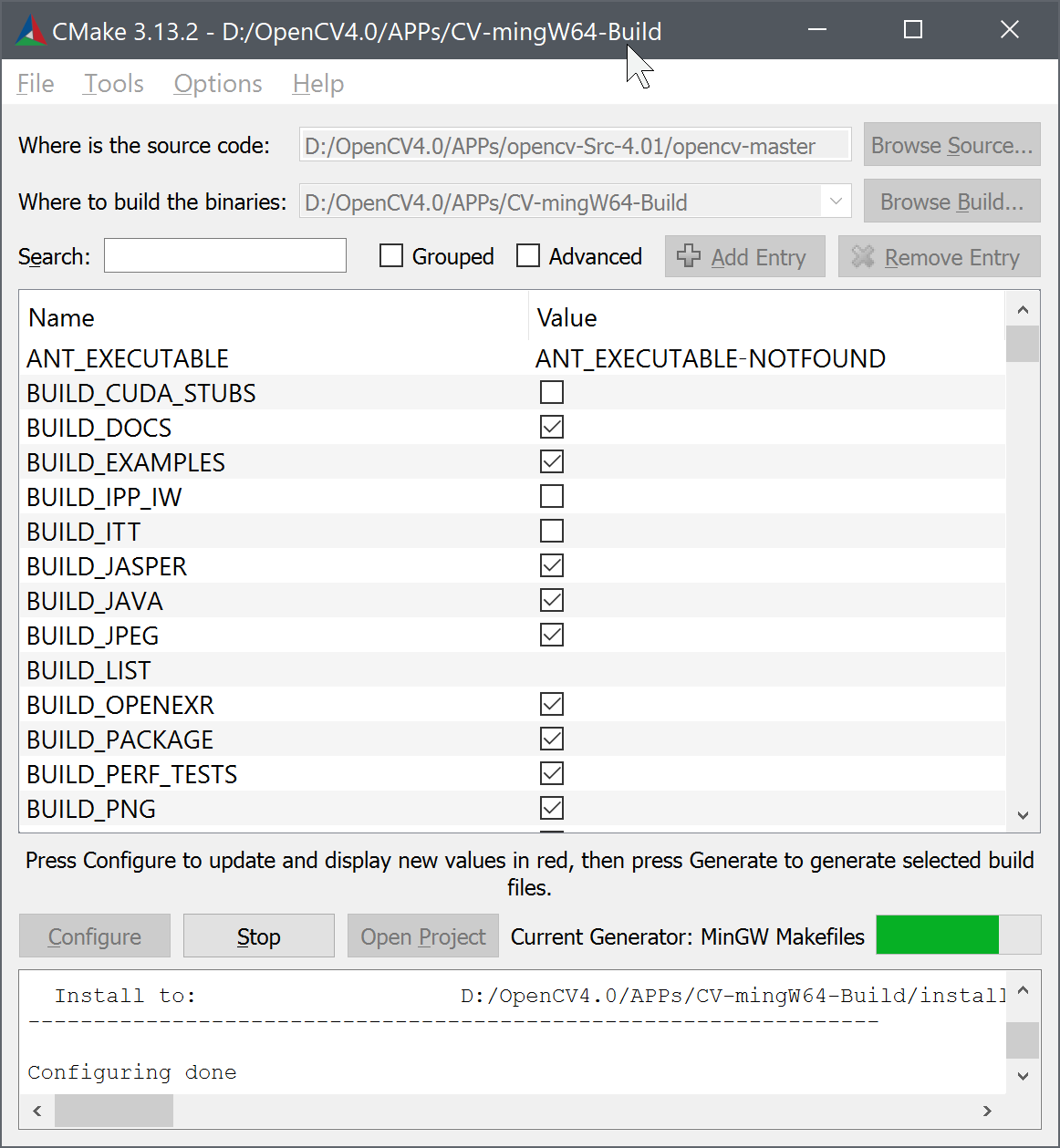


编译配置：

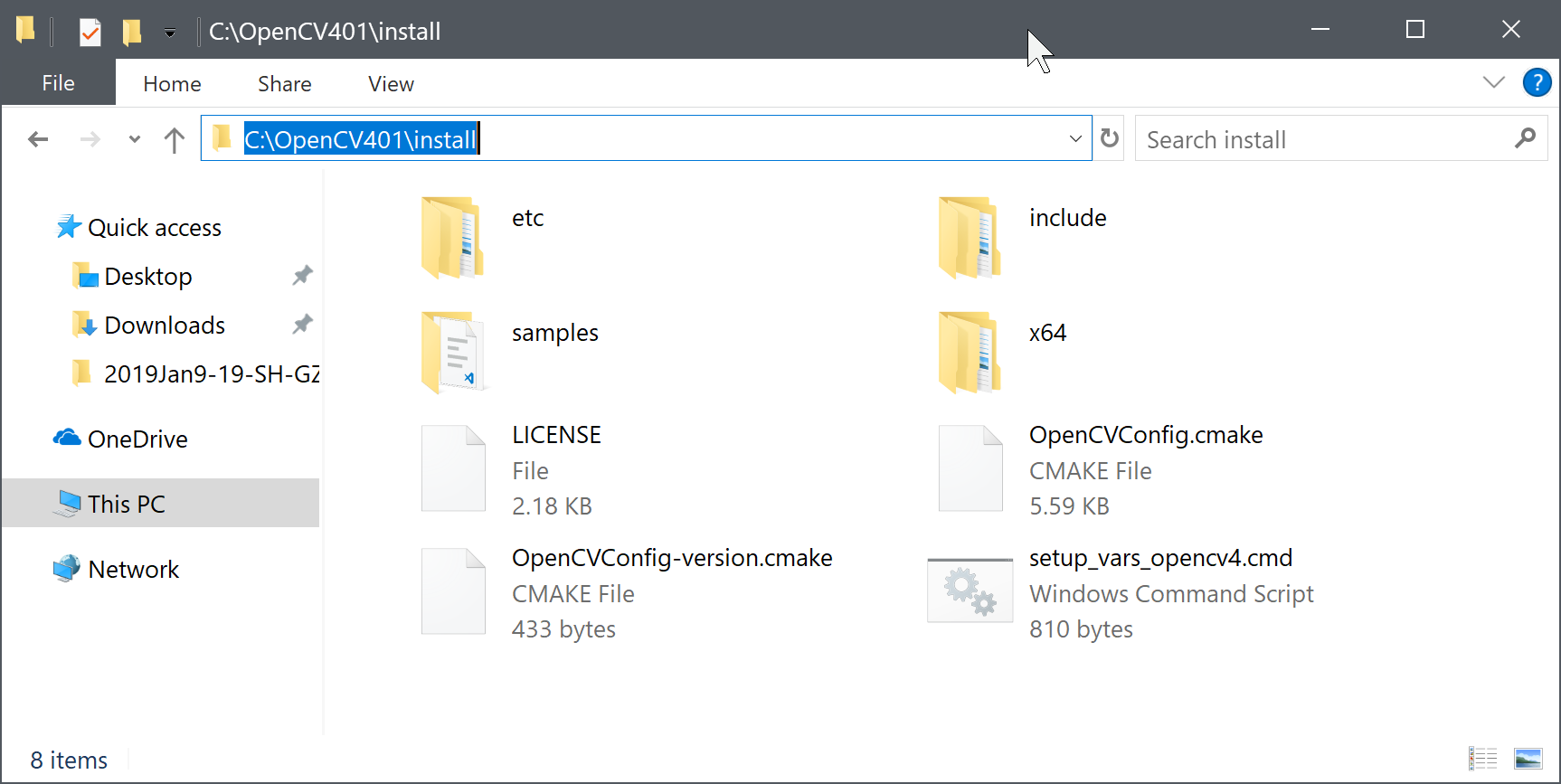
* 勾选 WITH\_OPENGL
* 勾选 ENABLE\_CXX11
* 不勾选 WITH\_IPP
* 不勾选 ENABLE\_PRECOMPILED\_HEADERS

点击 Configure，Generate 生成 Makefile





* 1. Install folder to C:\OpenCV401, make sure install folder in place



1. Setup VSCode
   1. tasks.json

{

"configurations": [

{

"name": "Win32",

"includePath": [

"${workspaceFolder}/\*\*",

"C:/OpenCV401/install/include",

"C:/MinGW64/mingw64/lib/gcc/x86\_64-w64-mingw32/8.1.0/include/c++"

],

"defines": [

"\_DEBUG",

"UNICODE",

"\_UNICODE"

],

"compilerPath": "C:\\MinGW64\\mingw64\\bin\\g++",

"cStandard": "c11",

"cppStandard": "c++17",

"intelliSenseMode": "msvc-x64",

"browse": {

"path": [

"${workspaceFolder}",

"C:/OpenCV401/install/include"

]

}

}

],

"version": 4

}

* 1. c\_cpp\_properties.json

**g++** XXX.cpp **-I**C:\OpenCV-MinGW-Build\include **-L**C:\OpenCV-MinGW-Build\x64\mingw\bin -llibopencv\_calib3d341 -llibopencv\_core341 -llibopencv\_dnn341 -llibopencv\_features2d341 -llibopencv\_flann341 -llibopencv\_highgui341 -llibopencv\_imgcodecs341 -llibopencv\_imgproc341 -llibopencv\_ml341 -llibopencv\_objdetect341 -llibopencv\_photo341 -llibopencv\_shape341 -llibopencv\_stitching341 -llibopencv\_superres341 -llibopencv\_video341 -llibopencv\_videoio341 -llibopencv\_videostab341

{

*// See https://go.microsoft.com/fwlink/?LinkId=733558*

*// for the documentation about the tasks.json format*

"version": "2.0.0",

"tasks": [

{

"label": "MinGW64-OpenCV",

"type": "shell",

"command": "g++",

"args": [

"-g",

"${fileBasename}",

"-o",

"${fileBasenameNoExtension}",

"-IC:/OpenCV401/install/include",

"-LC:\\OpenCV401\\install\\x64\\mingw\\bin",

"-llibopencv\_calib3d401",

"-llibopencv\_core401",

"-llibopencv\_dnn401",

"-llibopencv\_features2d401",

"-llibopencv\_flann401",

"-llibopencv\_gapi401",

"-llibopencv\_highgui401",

"-llibopencv\_imgcodecs401",

"-llibopencv\_imgproc401",

"-llibopencv\_ml401",

"-llibopencv\_objdetect401",

"-llibopencv\_photo401",

"-llibopencv\_stitching401",

"-llibopencv\_video401",

"-llibopencv\_videoio401"

],

"group": {

"kind": "build",

"isDefault": true

},

"problemMatcher":"$gcc"

}

]

}

* 1. launch.json

{

*// Use IntelliSense to learn about possible attributes.*

*// Hover to view descriptions of existing attributes.*

*// For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387*

"version": "0.2.0",

"configurations": [

{

"name": "(gdb) Launch",

"type": "cppdbg",

"request": "launch",

"preLaunchTask": "MinGW64-OpenCV",

"program": "${workspaceFolder}/${fileBasenameNoExtension}.exe",

"args": [],

"stopAtEntry": false,

"cwd": "${workspaceFolder}",

"environment": [],

"externalConsole": true,

"MIMode": "gdb",

"miDebuggerPath": "C:\\MinGW64\\mingw64\\bin\\gdb.exe",

"setupCommands": [

{

"description": "Enable pretty-printing for gdb",

"text": "-enable-pretty-printing",

"ignoreFailures": true

}

]

}

]

}

1. Test example “main.cpp”with images

#include <opencv2/core/core.hpp>

#include <opencv2/highgui/highgui.hpp>

#include <iostream>

using namespace cv;

using namespace std;

void FindFeatures()

{

}

int main() {

Mat image = imread("./lena.jpg", IMREAD\_COLOR); *// Read the file*

namedWindow("Display window", WINDOW\_AUTOSIZE); *// Create a window for display.*

if (!image.data) *// Check for invalid input*

cout << "Could not open or find the image" << std::endl;

else *// Image is good!*

imshow("Display window", image); *// Show our image inside it.*

waitKey(0);

return 0;

}