|  |  |
| --- | --- |
| Manual  2021/02/20 | Hello world OpenCV with VS Code |
| **Installation OpenCV in MacOS** | Installing CMake CMake is a cross platform build tool popular among native C/C++ developers. I use CMake in this tutorial to build and install OpenCV for C++ as well as run the demo project. CMake can be easily installed using brew as follows.  # brew install cmake Installing OpenCV from Source with CMake # mkdir opencv  # cd opencv  # git clone https://github.com/opencv/opencv.git  # cd opencv  # git checkout tags/4.2.0  # cd ..  # git clone https://github.com/opencv/  # cd opencv\_contrib  # git checkout tags/4.2.0  # cd ..  I should now be in the top level opencv directory that contains the two aforementioned repositories.  Next I create one more directories named **build\_opencv** for building the sources in.  # mkdir install build\_opencv  # ls -l  total 0  drwxr-xr-x 2 thainq 64 Mar 29 21:45 build\_opencv  drwxr-xr-x 21 thainq 672 Mar 29 21:35 opencv  drwxr-xr-x 13 thainq 416 Mar 29 21:36 opencv\_contrib  Next I change directories into the build\_opencv directory and configure CMake as shown below. |
| **Installation OpenCV in MacOS** | # cd build\_opencv  # cmake -D CMAKE\_BUILD\_TYPE=RELEASE \  -D OPENCV\_EXTRA\_MODULES\_PATH=../opencv\_contrib/modules \  -D BUILD\_DOCS=ON  -D BUILD\_EXAMPLES=ON ../opencv  -D BUILD\_E  The last steps to install is to kick off the multithreaded build to compile the project followed by installing it into the install directory as seen below.  # make -j 7 # runs 7 jobs in parallel  # make install |
| **Installation CMake in VS Code** | Install C/C++ extension C++ extension for VS Code. Install the C/C++ extension by searching for 'c++' in the Extensions view   Install CMake plugin for VS code C++ extension for VS Code. Install the C/C++ extension by searching for 'cmake' in the Extensions view   Ensure that CMake is installed The VS Code CMake Tools extension does its work by using CMake installed on your system. For best results, use CMake version 3.15 or greater  See if CMake is already installed on your system. Open terminal window and enter the following command:  # cmake –version  cmake version 3.19.2 Ensure that development tools are installed # gcc -v  Apple clang version 12.0.0 (clang-1200.0.32.27)  Target: x86\_64-apple-darwin19.6.0  Thread model: posix  InstalledDir: /Library/Developer/CommandLineTools/usr/bin |
| **Hello World Open CV** | Create a folder for a new project # mkdir hello-world-opencv  # cd hello-world-opencv  # code . Create a CMake hello world project The CMake tools extension can create the fields for basic CMake porjetc for you. Open the Command Palette and run the **CMake: Quick Start** command    Enter a project name: hello-world-opencv  Next, select Executable as the project type to create a basic source file (main.cpp) that includes a basic main() function    Update CMakeLists.txt with following instruction  cmake\_minimum\_required(VERSION 3.0.0)  project(hello-world-opencv VERSION 0.1.0)  include(CTest)  enable\_testing()  add\_executable(demo\_open\_cv main.cpp)  set(CPACK\_PROJECT\_NAME ${PROJECT\_NAME})  set(CPACK\_PROJECT\_VERSION ${PROJECT\_VERSION})  # Tell compiler to use C++ 14 features which is needed because  # Clang version is often behind in the XCode installation  set(CMAKE\_CXX\_STANDARD 14)  # set OpenCV\_DIR variable equal to the path to the cmake  # files within the previously installed opencv program  set(OpenCV\_DIR /usr/local/include/opencv4)  # configure the necessary common CMake environment variables  # needed to include and link the OpenCV program into this  # demo project, namely OpenCV\_INCLUDE\_DIRS and OpenCV\_LIBS  find\_package( OpenCV REQUIRED )  # tell it to link the executable target against OpenCV  target\_link\_libraries(demo\_open\_cv ${OpenCV\_LIBS} )  include(CPack)  Update file main.cpp with following code  #include <iostream>  #include <opencv2/opencv.hpp>  #include <opencv2/highgui.hpp>  #include <opencv2/core/core.hpp>  using namespace std;  using namespace cv;  int main() {  Mat img = imread("./../test1.jpg");  imshow("Image Test", img);  waitKey(0);  }  Buid & Run |
| **Settings for vscode** | Under folder .vscode  Making a new file named: c\_cpp\_properties.json with content as below  {  "configurations": [  {  "name": "Mac",  "includePath": [  "${workspaceFolder}/\*",  "/usr/local/include/opencv4/"  ],  "defines": [],  "macFrameworkPath": [  "/Library/Developer/CommandLineTools/SDKs/MacOSX10.15.sdk/System/Library/Frameworks"  ],  "compilerPath": "/usr/bin/clang",  "cStandard": "c11",  "cppStandard": "c++17",  "intelliSenseMode": "clang-x64",  "configurationProvider": "ms-vscode.cmake-tools"  }  ],  "version": 4  } |
|  |  |