How to install OpenCV in windows 10 using MinGW

Unfortunately OpenCV doesn't come with prebuilt mingw/TDM (64 bit) binaries for windows. In this tutorial, we are going to build them ourselves.

Step 1. Environment setup

- **mingw: mingw-w64 (64 bit)**
- **CMAKE : cmake 3.14 (64 bit)**

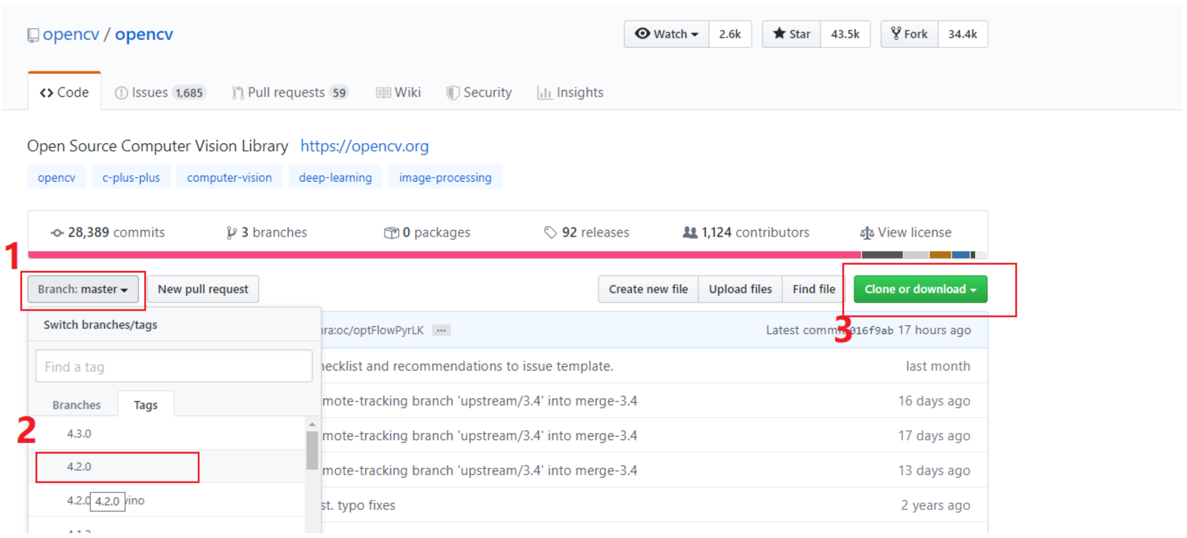
  1. Download CMake and install it (in the installation wizard choose to add CMake to the system PATH).
  2. (https://cmake.org/download/ or https://cmake.org/files/v3.14/)

cmake-3.14.5-win32-x86.zip	2019-05-31 12:39	26M
cmake-3.14.5-win64-x64.msi	2019-05-31 12:39	22M
cmake-3.14.5-win64-x64.zip	2019-05-31 12:39	30M
cmake-3.14.5.tar.Z	2019-05-31 12:39	14M
cmake-3.14.5.tar.gz	2019-05-31 12:39	8.4M
cmake-3.14.5.zip	2019-05-31 12:40	14M
cmake-3.14.6-Darwin-x86_64.dmg	2019-07-16 09:33	33M
cmake-3.14.6-Darwin-x86_64.tar.gz	2019-07-16 09:33	32M
cmake-3.14.6-Linux-x86_64.sh	2019-07-16 09:33	35M
cmake-3.14.6-Linux-x86_64.tar.gz	2019-07-16 09:33	35M
cmake-3.14.6-SHA-256.txt	2019-07-16 09:33	1.0K
cmake-3.14.6-SHA-256.txt.asc	2019-07-16 09:33	833

- **opencv : 4.2.0**

  Download the source of OpenCV and checkout 4.2.0 (https://github.com/opencv/opencv)

```
git checkout 4.2.0
```



Open Source Computer Vision Library https://opencv.org

opencv c-plus-plus computer-vision deep-learning image-processing

28,389 commits 3 branches 0 packages 92 releases 1,124 contributors View license

4.3.0
4.2.0
4.2.0 4.2.0 ino

ra:oc/optFlowPyrLK ... Latest commit 016f9ab 17 hours ago
hecklist and recommendations to issue template. last month
mote-tracking branch 'upstream/3.4' into merge-3.4 16 days ago
mote-tracking branch 'upstream/3.4' into merge-3.4 17 days ago
mote-tracking branch 'upstream/3.4' into merge-3.4 13 days ago
st. typo fixes 2 years ago

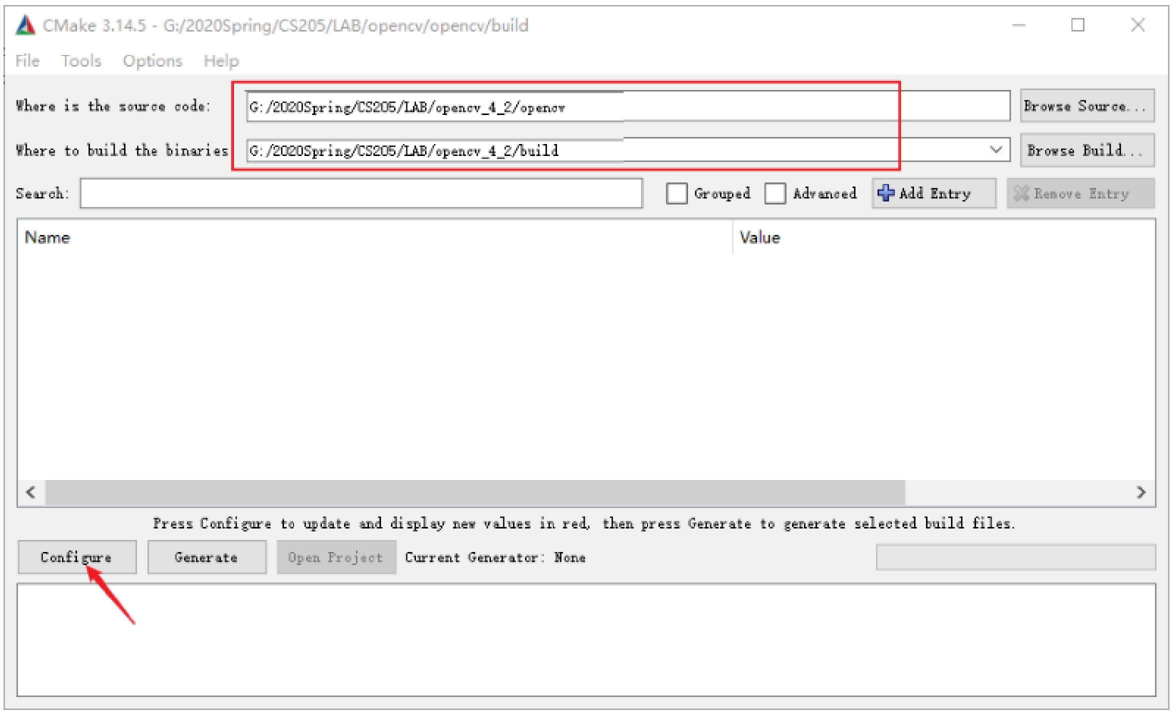
- **opencv_contrib**

  Download the source of **opencv_contrib** and checkout 4.2.0 (https://github.com/opencv/opencv_contrib)

  ```
  git checkout 4.2.0
  ```

Step 2: Compiling OpenCV

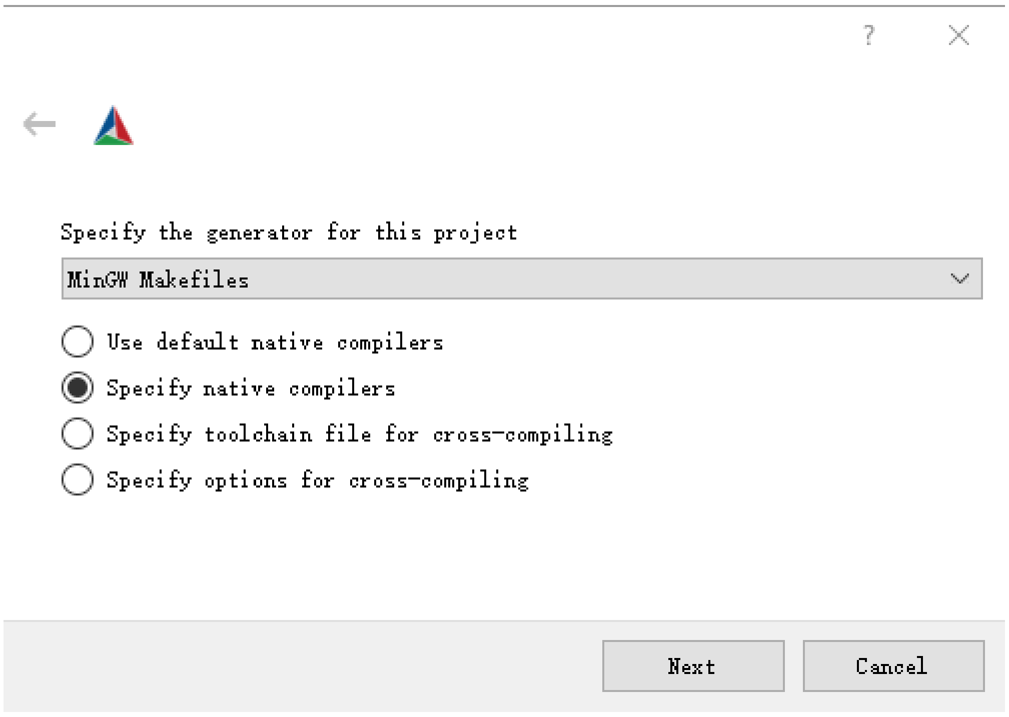
Open `cmake`, set source path and binary path ,and then hit `configure` button



Where is the source code: G:/2020Spring/CS205/LAB/opencv_4_2/opencv
Where to build the binaries: G:/2020Spring/CS205/LAB/opencv_4_2/build
Search:
Press Configure to update and display new values in red, then press Generate to generate selected build files.
Current Generator: None

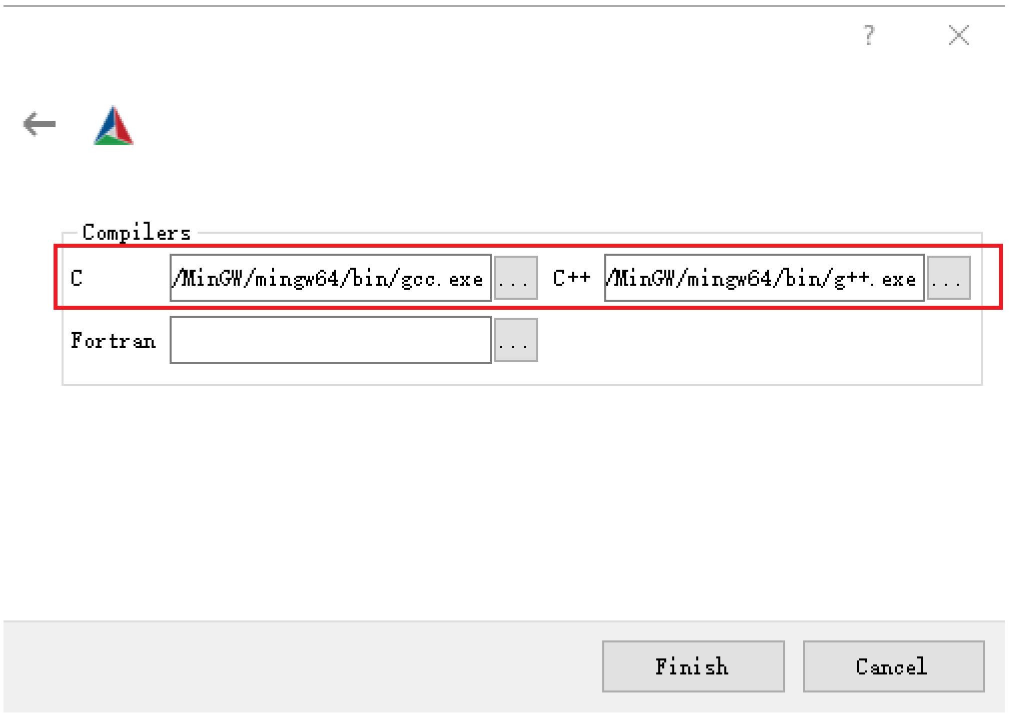
configure compiller

> 1. Specify the generator for this project: MinGW Makefiles
> 2. Specify native compilers
> 3. Next
> 4. Compilers C: C:\mingw64\bin\gcc.exe
> 5. Compilers C++: C:\mingw64\bin\g++.exe
> 6. Finish


Specify the generator for this project

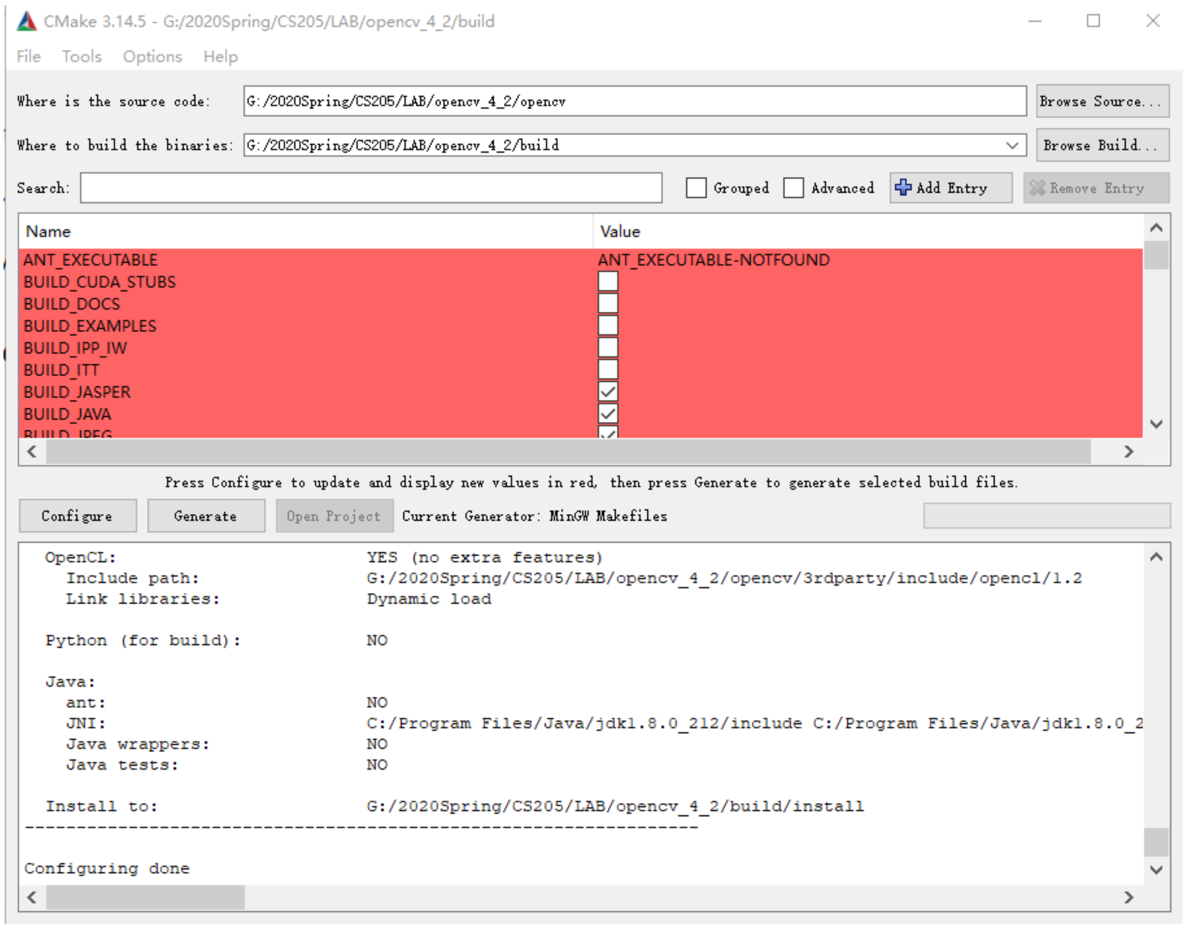
MinGW Makefiles

- ( ) Use default native compilers
- (●) Specify native compilers
- ( ) Specify toolchain file for cross-compiling
- ( ) Specify options for cross-compiling


Compilers

C /MinGW/mingw64/bin/gcc.exe C++ /MinGW/mingw64/bin/g++.exe

Fortran



Where is the source code: G:/2020Spring/CS205/LAB/opencv_4_2/opencv

Where to build the binaries: G:/2020Spring/CS205/LAB/opencv_4_2/build

Search:

Name	Value
ANT_EXECUTABLE	ANT_EXECUTABLE-NOTFOUND
BUILD_CUDA_STUBS	☐
BUILD_DOCS	☐
BUILD_EXAMPLES	☐
BUILD_IPP_IW	☐
BUILD_ITT	☐
BUILD_JASPER	✓
BUILD_JAVA	✓
BUILD_JPEG	✓

Press Configure to update and display new values in red, then press Generate to generate selected build files.

Current Generator: MinGW Makefiles

```
  OpenCL:                      YES (no extra features)
    Include path:              G:/2020Spring/CS205/LAB/opencv_4_2/opencv/3rdparty/include/opencl/1.2
    Link libraries:            Dynamic load

  Python (for build):          NO

  Java:
    ant:                       NO
    JNI:                       C:/Program Files/Java/jdk1.8.0_212/include C:/Program Files/Java/jdk1.8.0_2
    Java wrappers:             NO
    Java tests:                NO

  Install to:                  G:/2020Spring/CS205/LAB/opencv_4_2/build/install
-----------------------------------------------------------------

Configuring done
```

Change the settings

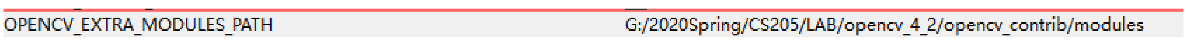
- **Check**

```
WITH_OPENGL
ENABLE_CXX11(This option is not available after OpenCV4),
```

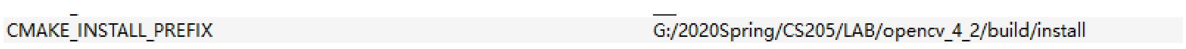
- **Uncheck**

```
WITH_IPP
ENABLE_PRECOMPILED_HEADERS
WITH_OPENCL_D3D11_NV
```

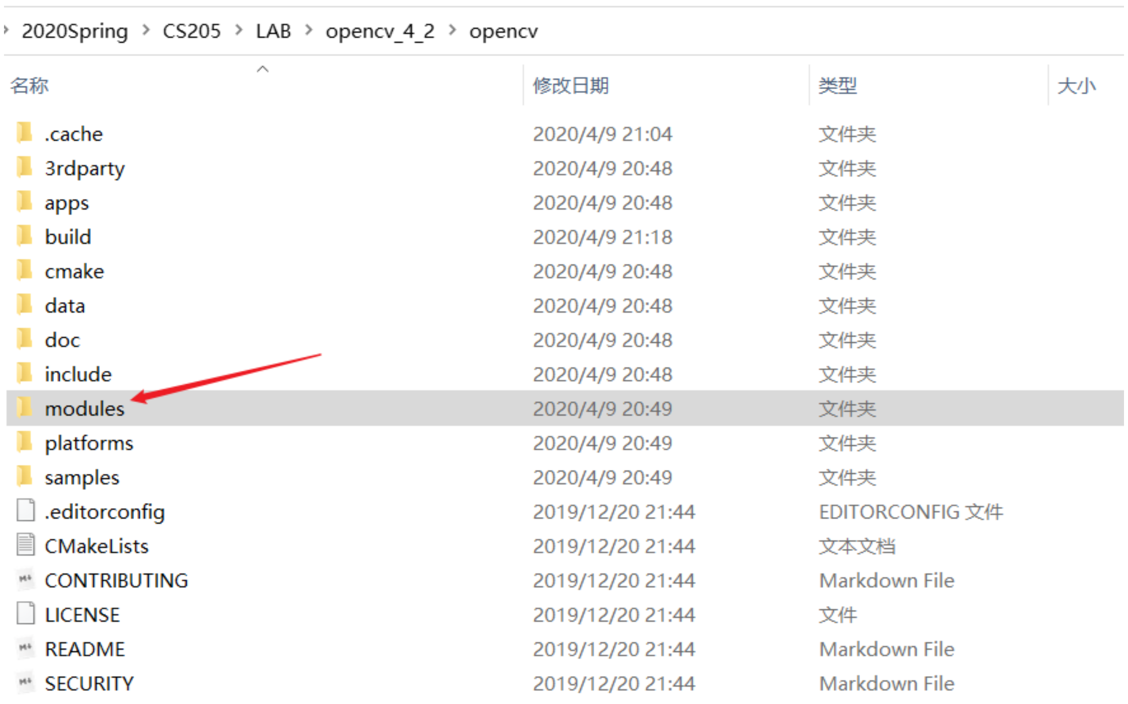
- **Modify** `OPENCV_EXTRA_MODULES_PATH` **and** `CMAKE_INSTALL_PREFIX`



OPENCV_EXTRA_MODULES_PATH   G:/2020Spring/CS205/LAB/opencv_4_2/opencv_contrib/modules



CMAKE_INSTALL_PREFIX   G:/2020Spring/CS205/LAB/opencv_4_2/build/install

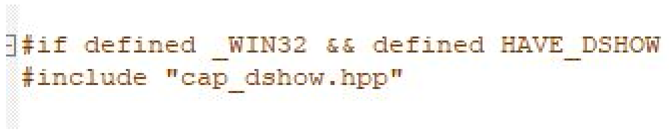
- 

› 2020Spring › CS205 › LAB › opencv_4_2 › opencv

名称	修改日期	类型	大小
.cache	2020/4/9 21:04	文件夹	
3rdparty	2020/4/9 20:48	文件夹	
apps	2020/4/9 20:48	文件夹	
build	2020/4/9 21:18	文件夹	
cmake	2020/4/9 20:48	文件夹	
data	2020/4/9 20:48	文件夹	
doc	2020/4/9 20:48	文件夹	
include	2020/4/9 20:48	文件夹	
modules	2020/4/9 20:49	文件夹	
platforms	2020/4/9 20:49	文件夹	
samples	2020/4/9 20:49	文件夹	
.editorconfig	2019/12/20 21:44	EDITORCONFIG 文件	
CMakeLists	2019/12/20 21:44	文本文档	
CONTRIBUTING	2019/12/20 21:44	Markdown File	
LICENSE	2019/12/20 21:44	文件	
README	2019/12/20 21:44	Markdown File	
SECURITY	2019/12/20 21:44	Markdown File	

- **Modify the code**

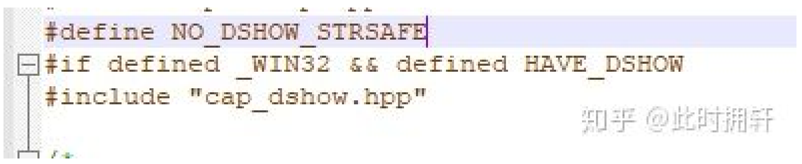
`modules/videoio/src/cap_dshow.cpp`



```
#if defined _WIN32 && defined HAVE_DSHOW
#include "cap_dshow.hpp"
```

Add one line:

```
#define NO_DSHOW_STRSAFE
```



```
#define NO_DSHOW_STRSAFE
#if defined _WIN32 && defined HAVE_DSHOW
#include "cap_dshow.hpp"
```

When you're done, press `Configure'` again. You should see `Configuration done'` at the log window, and the red background should disappear from all the cells.

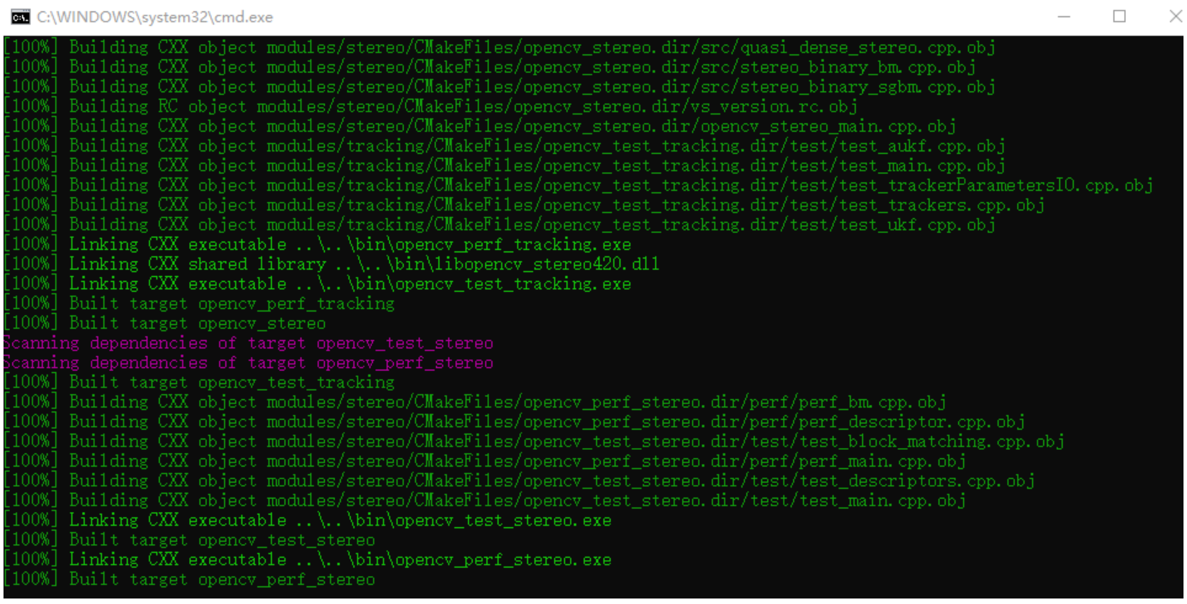
At this point `CMake` is ready to generate the makefile with which we will compile `OpenCV` with our compiler. Click 'Generate' and wait for the makefile to be generated. When the process is finished you should see 'Generating done'. From this point we will no longer need `CMake`.

Compiling

Open MinGW shell (The following steps can also be done from Windows' command prompt).

- Enter the binary path you set up(`G:/2020Spring/CS205/LAB/opencv_4_2/build`)
- Type `mingw32-make` and press enter. This should start the compilation process.

```
mingw32-make -j 8
```

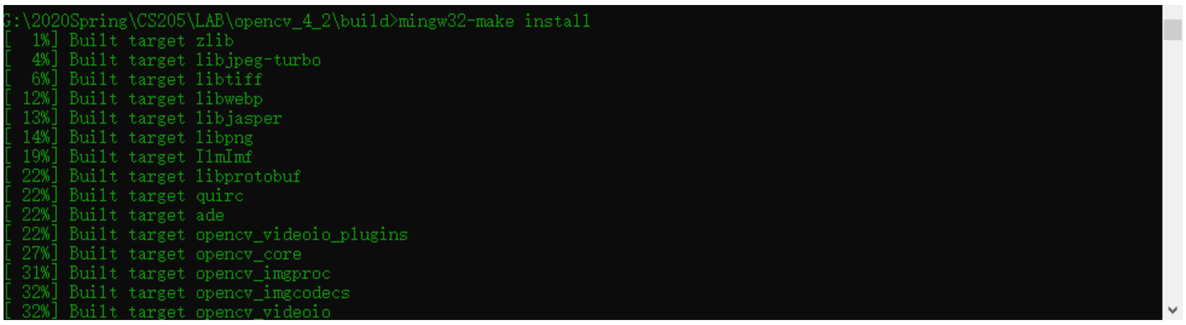


```
[100%] Building CXX object modules/stereo/CMakeFiles/opencv_stereo.dir/src/quasi_dense_stereo.cpp.obj
[100%] Building CXX object modules/stereo/CMakeFiles/opencv_stereo.dir/src/stereo_binary_bm.cpp.obj
[100%] Building CXX object modules/stereo/CMakeFiles/opencv_stereo.dir/src/stereo_binary_sgbm.cpp.obj
[100%] Building RC object modules/stereo/CMakeFiles/opencv_stereo.dir/vs_version.rc.obj
[100%] Building CXX object modules/stereo/CMakeFiles/opencv_stereo.dir/opencv_stereo_main.cpp.obj
[100%] Building CXX object modules/tracking/CMakeFiles/opencv_test_tracking.dir/test/test_aukf.cpp.obj
[100%] Building CXX object modules/tracking/CMakeFiles/opencv_test_tracking.dir/test/test_main.cpp.obj
[100%] Building CXX object modules/tracking/CMakeFiles/opencv_test_tracking.dir/test/test_trackerParametersIO.cpp.obj
[100%] Building CXX object modules/tracking/CMakeFiles/opencv_test_tracking.dir/test/test_trackers.cpp.obj
[100%] Building CXX object modules/tracking/CMakeFiles/opencv_test_tracking.dir/test/test_ukf.cpp.obj
[100%] Linking CXX executable ..\..\bin\opencv_perf_tracking.exe
[100%] Linking CXX shared library ..\..\bin\libopencv_stereo420.dll
[100%] Linking CXX executable ..\..\bin\opencv_test_tracking.exe
[100%] Built target opencv_perf_tracking
[100%] Built target opencv_stereo
Scanning dependencies of target opencv_test_stereo
Scanning dependencies of target opencv_perf_stereo
[100%] Built target opencv_test_tracking
[100%] Building CXX object modules/stereo/CMakeFiles/opencv_perf_stereo.dir/perf/perf_bm.cpp.obj
[100%] Building CXX object modules/stereo/CMakeFiles/opencv_perf_stereo.dir/perf/perf_descriptor.cpp.obj
[100%] Building CXX object modules/stereo/CMakeFiles/opencv_test_stereo.dir/test/test_block_matching.cpp.obj
[100%] Building CXX object modules/stereo/CMakeFiles/opencv_perf_stereo.dir/perf/perf_main.cpp.obj
[100%] Building CXX object modules/stereo/CMakeFiles/opencv_test_stereo.dir/test/test_descriptors.cpp.obj
[100%] Building CXX object modules/stereo/CMakeFiles/opencv_test_stereo.dir/test/test_main.cpp.obj
[100%] Linking CXX executable ..\..\bin\opencv_test_stereo.exe
[100%] Built target opencv_test_stereo
[100%] Linking CXX executable ..\..\bin\opencv_perf_stereo.exe
[100%] Built target opencv_perf_stereo
```

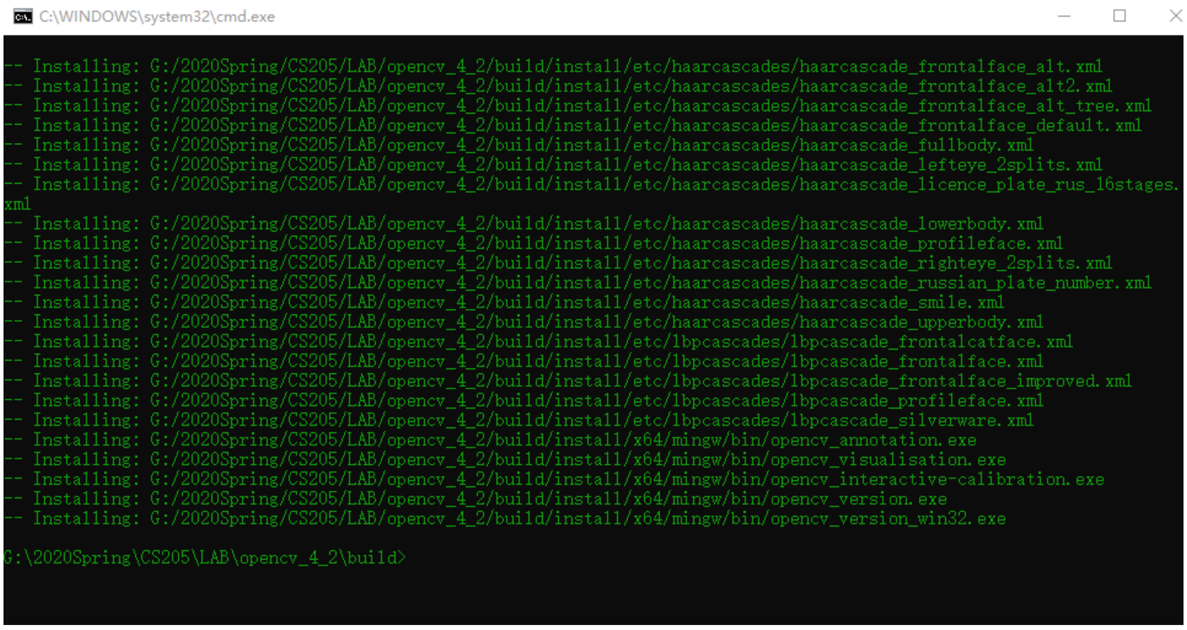
When the compilation is done OpenCV's binaries are ready to be used.

- install opencv

```
mingw32-make install
```

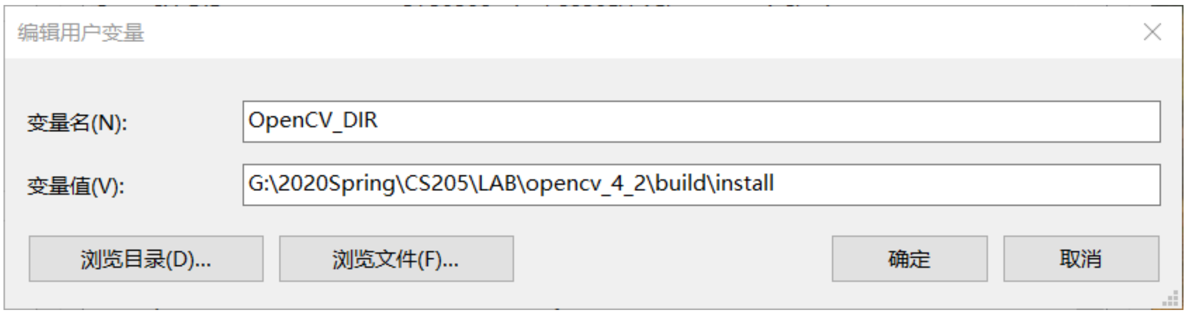


```
G:\2020Spring\CS205\LAB\opencv_4_2\build>mingw32-make install
[  1%] Built target zlib
[  4%] Built target libjpeg-turbo
[  6%] Built target libtiff
[ 12%] Built target libwebp
[ 13%] Built target libjasper
[ 14%] Built target libpng
[ 19%] Built target IlmImf
[ 22%] Built target libprotobuf
[ 22%] Built target quirc
[ 22%] Built target ade
[ 22%] Built target opencv_videoio_plugins
[ 27%] Built target opencv_core
[ 31%] Built target opencv_imgproc
[ 32%] Built target opencv_imgcodecs
[ 32%] Built target opencv_videoio
```



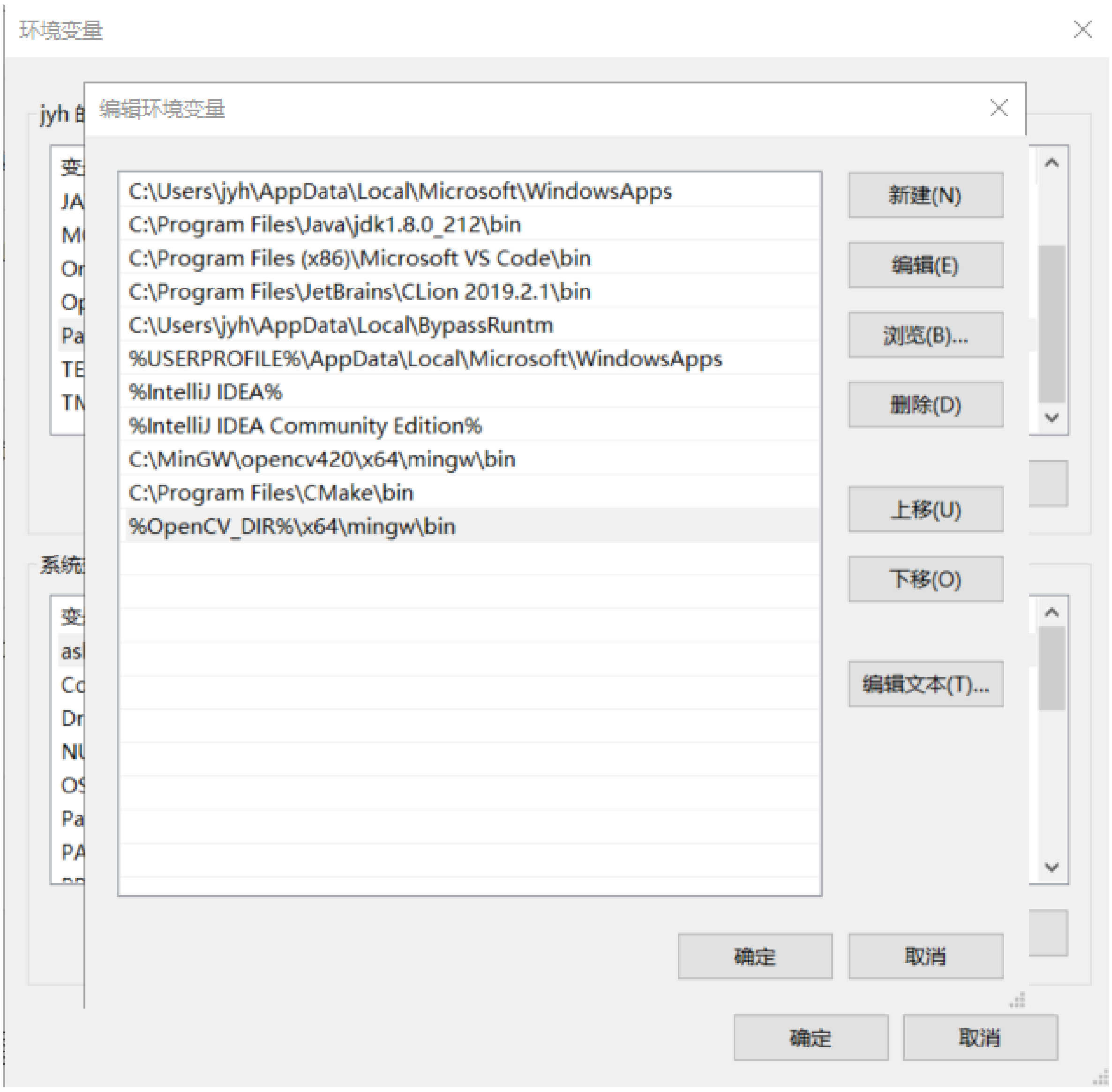
```
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_frontalface_alt.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_frontalface_alt2.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_frontalface_alt_tree.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_frontalface_default.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_fullbody.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_lefteye_2splits.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_licence_plate_rus_16stages.
xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_lowerbody.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_profileface.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_righteye_2splits.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_russian_plate_number.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_smile.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/haarcascades/haarcascade_upperbody.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/lbpcascades/lbpcascade_frontalcatface.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/lbpcascades/lbpcascade_frontalface.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/lbpcascades/lbpcascade_frontalface_improved.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/lbpcascades/lbpcascade_profileface.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/etc/lbpcascades/lbpcascade_silverware.xml
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/x64/mingw/bin/opencv_annotation.exe
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/x64/mingw/bin/opencv_visualisation.exe
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/x64/mingw/bin/opencv_interactive-calibration.exe
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/x64/mingw/bin/opencv_version.exe
-- Installing: G:/2020Spring/CS205/LAB/opencv_4_2/build/install/x64/mingw/bin/opencv_version_win32.exe

G:\2020Spring\CS205\LAB\opencv_4_2\build>
```

Step 3: Add OpenCV to the system path


编辑用户变量

变量名(N): OpenCV_DIR

变量值(V): G:\2020Spring\CS205\LAB\opencv_4_2\build\install


环境变量

编辑环境变量

```
C:\Users\jyh\AppData\Local\Microsoft\WindowsApps
C:\Program Files\Java\jdk1.8.0_212\bin
C:\Program Files (x86)\Microsoft VS Code\bin
C:\Program Files\JetBrains\CLion 2019.2.1\bin
C:\Users\jyh\AppData\Local\BypassRuntm
%USERPROFILE%\AppData\Local\Microsoft\WindowsApps
%IntelliJ IDEA%
%IntelliJ IDEA Community Edition%
C:\MinGW\opencv420\x64\mingw\bin
C:\Program Files\CMake\bin
%OpenCV_DIR%\x64\mingw\bin
```

Step4: Running test program

Now run this simple OpenCV “Hello World” program to test that the install has worked.

```
#include <opencv.hpp>
#include <iostream>
using namespace cv;
using namespace std;
```

```cpp
int main()
{
    cout << "OpenCV Version: " << CV_VERSION << endl;
    Mat img = imread("Pokemon02.png");
    imshow("1440", img);
    waitKey(0);
    return 0;
}
```

```cmake
cmake_minimum_required(VERSION 3.6)
PROJECT(opencv_demo)

set(OpenCV_INCLUDE_DIRS
G:/2020Spring/CS205/LAB/opencv_4_2/mingw_opencv/include/opencv2)
FIND_PACKAGE(OpenCV REQUIRED)
message(STATUS "OpenCV library status:")
message(STATUS "    version: ${OpenCV_VERSION}")
message(STATUS "    libraries: ${OpenCV_LIBS}")
message(STATUS "    include path: ${OpenCV_INCLUDE_DIRS}")
include_directories(
        ${PROJECT_SOURCE_DIR}
        #${OpenCV_INCLUDE_DIRS}
        "G:/2020Spring/CS205/LAB/opencv_4_2/mingw_opencv/include/opencv2"

)
#include_directories(${OpenCV_INCLUDE_DIRS})
add_executable(opencv_demo main.cpp)
target_link_libraries(opencv_demo ${OpenCV_LIBS})
```