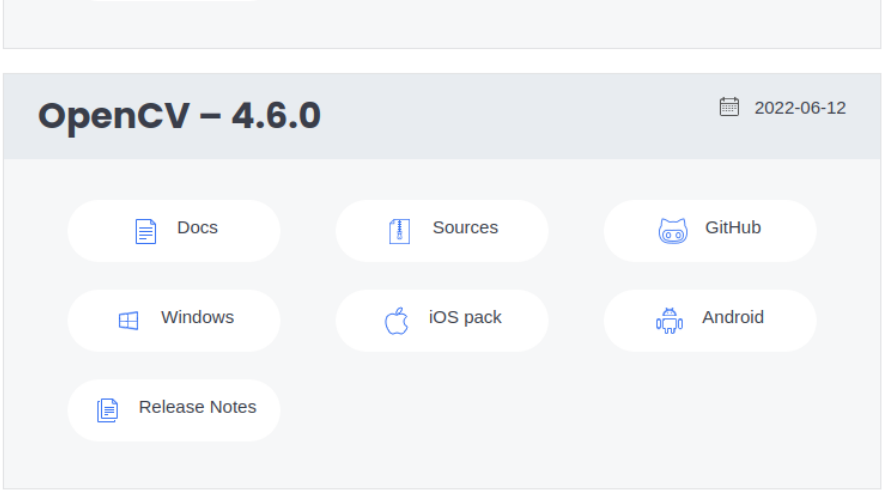
OpenCv 安装、配置 (4.6.0)

下载和安装依赖

- 下载opencv 4.6.0 source

https://opencv.org/releases/



- 下载opencv_contrib 4.6.0

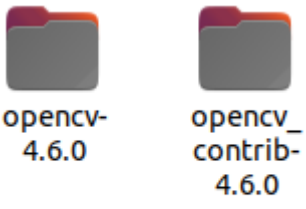
https://github.com/opencv/opencv_contrib/tree/4.6.0

down下来contrib的源码。

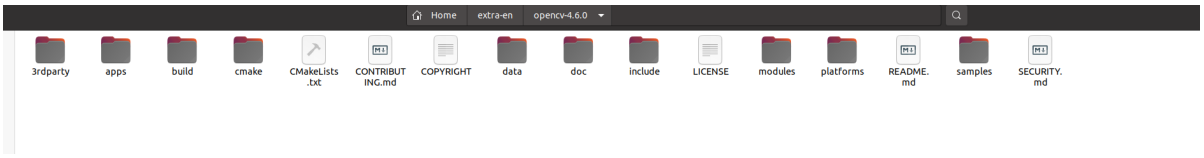
下载对应版本的opencv扩展即可。

- 文件结构

上述两个文件下载下来的文件结构正常情况下应该是这个样子：

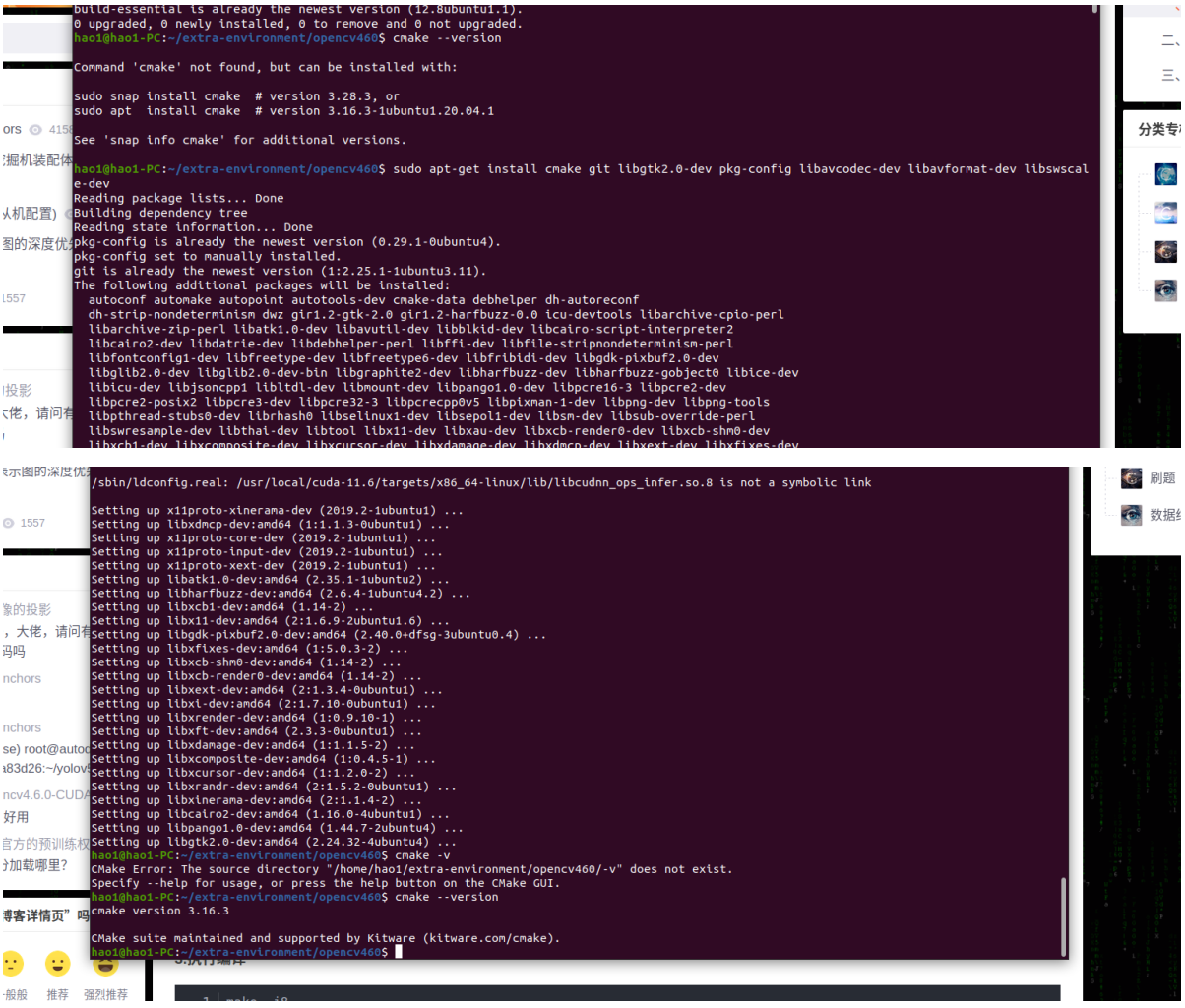


opencv-4.6.0里：



- 安装依赖：

```
sudo apt-get install build-essential
sudo apt-get install cmake git libgtk2.0-dev pkg-config libavcodec-dev
libavformat-dev libswscale-dev
sudo apt-get install python3-dev python3-numpy libtbb2 libtbb-dev libjpeg-dev
libpng-dev libtiff-dev libjasper-dev libdc1394-22-dev
```



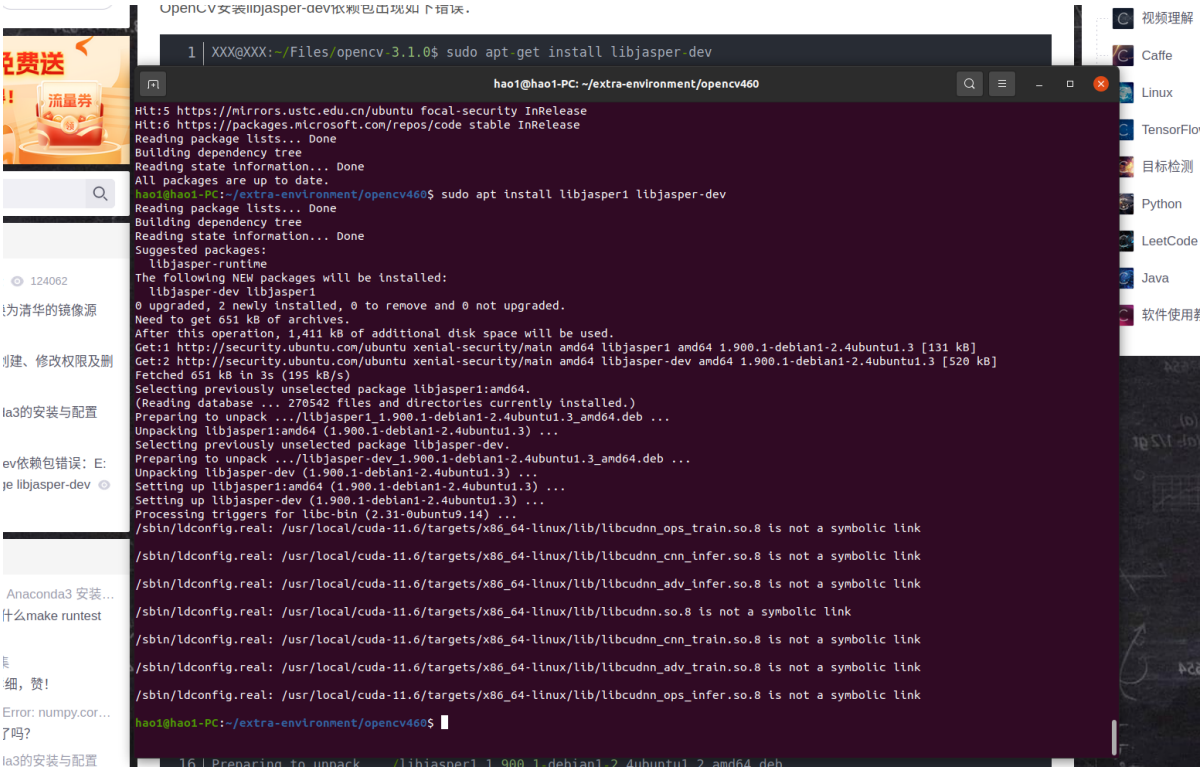
这里已经安装了cmake 3.16.3

上面的第三行，`python` 换成 `python3` （用python3而不是python2）

- `libjasor-dev` 可能安装不成功

解决参考：

https://blog.csdn.net/CAU_Ayao/article/details/83990246



安装完成

至此安装包就下载完成，依赖也都准备完全

编译、安装

可以不把opencv_contrib放到opencv文件夹内，**但是要在cmake的时候注意这个contrib的path设置**

- 首先进入opencv目录，**新建build文件夹**

然后在build终端：

```
cmake -D CMAKE_BUILD_TYPE=Release \
    -D OPENCV_EXTRA_MODULES_PATH=/home/zxm/opencv/opencv_contrib-4.6.0/modules \
    -D OPENCV_ENABLE_NONFREE=ON \
    -D WITH_OPENMP=ON \
    -D WITH_TBB=ON \
    -D WITH_QT=ON \
    -D CUDA_NVCC_FLAGS="--Wno-deprecated-gpu-targets" \
    -D OPENCV_EXTRA_EXE_LINKER_FLAGS=-latomic \
    -D ENABLE_PRECOMPILED_HEADERS=OFF \
    -D WITH_CUDA=ON \
    -D ENABLE_FAST_MATH=ON \
    -D CUDA_FAST_MATH=ON \
    -D WITH_CUBLAS=ON \
    -D PYTHON3_EXECUTABLE=/usr/bin/python3 \
    -D PYTHON3_INCLUDE_DIR=$(/usr/bin/python3 -c "from distutils.sysconfig import
get_python_inc; print(get_python_inc())") \
    -D PYTHON3_PACKAGES_PATH=$(/usr/bin/python3 -c "from distutils.sysconfig
import get_python_lib; print(get_python_lib())") \
    -D PYTHON3_NUMPY_INCLUDE_DIRS=$(/usr/bin/python3 -c "import numpy as
np;import os; print(os.path.dirname(np.__file__)+'/core/include')") \
    -D OPENCV_GENERATE_PKGCONFIG=ON \
    -D CMAKE_INSTALL_PREFIX=/usr/local/opencv4 \
    ..
```

修改成：

```
cmake -D CMAKE_BUILD_TYPE=Release \
    -D OPENCV_EXTRA_MODULES_PATH=/home/hao/extra-en/opencv_contrib-4.6.0/modules
\
    -D OPENCV_ENABLE_NONFREE=ON \
    -D WITH_OPENMP=ON \
    -D WITH_TBB=ON \
    -D WITH_QT=ON \
    -D CUDA_NVCC_FLAGS="--Wno-deprecated-gpu-targets" \
    -D OPENCV_EXTRA_EXE_LINKER_FLAGS=-latomic \
    -D ENABLE_PRECOMPILED_HEADERS=OFF \
    -D WITH_CUDA=ON \
    -D ENABLE_FAST_MATH=ON \
    -D CUDA_FAST_MATH=ON \
    -D WITH_CUBLAS=ON \
    -D PYTHON3_EXECUTABLE=/usr/bin/python3 \
    -D PYTHON3_INCLUDE_DIR=$(/usr/bin/python3 -c "from distutils.sysconfig import
get_python_inc; print(get_python_inc())") \
    -D PYTHON3_PACKAGES_PATH=$(/usr/bin/python3 -c "from distutils.sysconfig
import get_python_lib; print(get_python_lib())") \
    -D PYTHON3_NUMPY_INCLUDE_DIRS=$(/usr/bin/python3 -c "import numpy as
np;import os; print(os.path.dirname(np.__file__)+'/core/include')") \
    -D OPENCV_GENERATE_PKGCONFIG=ON \
    -D CMAKE_INSTALL_PREFIX=/usr/local/opencv4.6.0 \
    ..
```

注意：

1. 上述的 `OPENCV_EXTRA_MODULES_PATH` 是**指定opencv_contrib的路径**，这个路径要指定到 `opencv_contrib` 文件夹下的 `modules` 文件夹
2. `CMAKE_INSTALL_PREFIX` 是cmake编译opencv源码包之后的存放路径，这个路径需要你事先创建好，否则系统会默认创建在你的cmake的安装路径下，这样也是可以，但是这个路径你可能不好找，所以最好自己指定一个路径
3. 上面是开了 `with cuda` 表示我们要安装opencv的**cuda版本**，支持GPU。如果没有配置GPU和CUDA，要把这个选项改成 `OFF`
4. 安装目录（编译目录）是 **/usr/local/opencv4.6.0/**

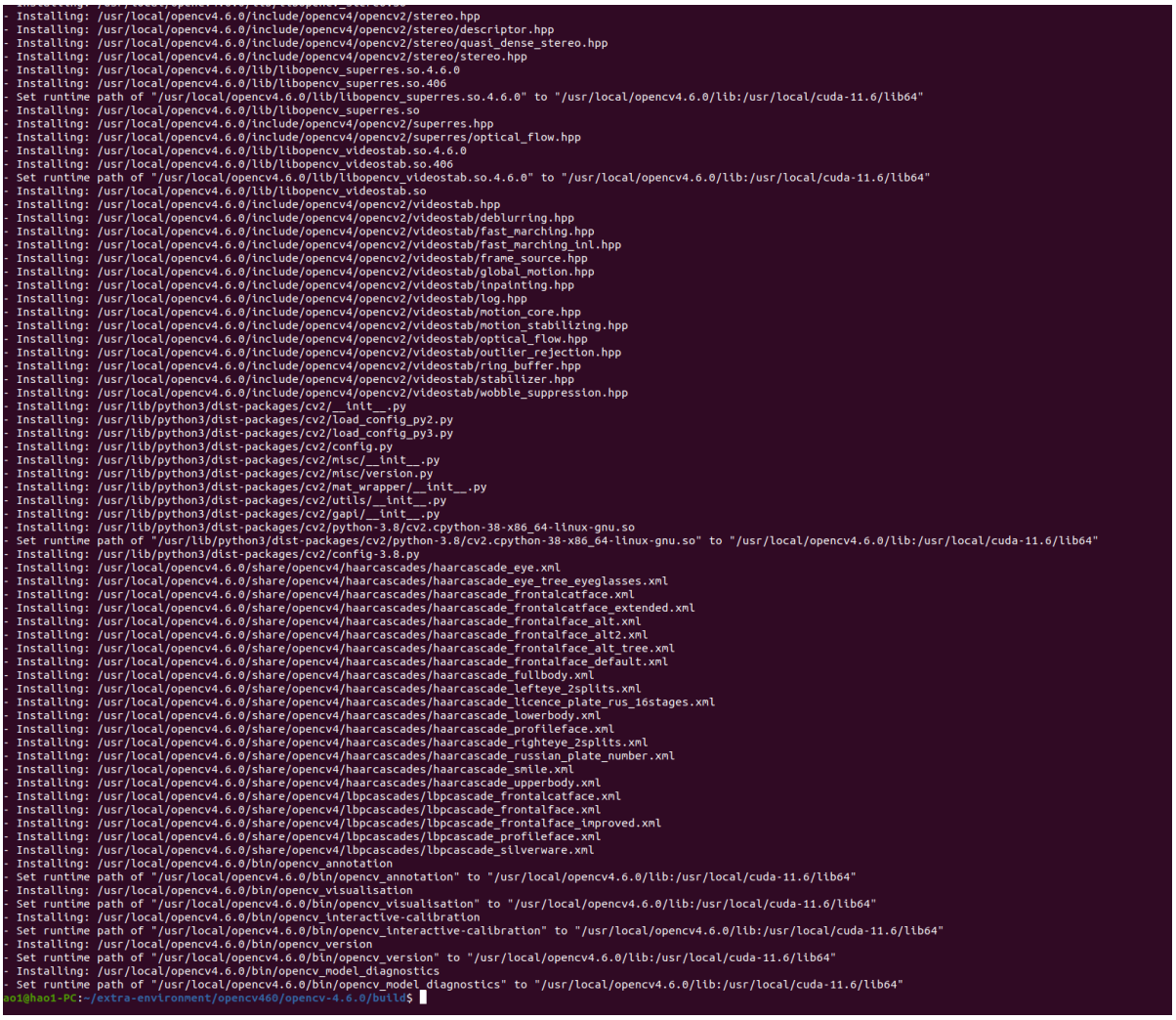
- cmake完成

```
--   Media I/O:
--     ZLib:                        /usr/lib/x86_64-linux-gnu/libz.so (ver 1.2.11)
--     JPEG:                        /usr/lib/x86_64-linux-gnu/libjpeg.so (ver 80)
--     WEBP:                        build (ver encoder: 0x020f)
--     PNG:                         /usr/lib/x86_64-linux-gnu/libpng.so (ver 1.6.37)
--     TIFF:                        /usr/lib/x86_64-linux-gnu/libtiff.so (ver 42 / 4.1.0)
--     JPEG 2000:                   build (ver 2.4.0)
--     OpenEXR:                     build (ver 2.3.0)
--     HDR:                         YES
--     SUNRASTER:                   YES
--     PXM:                         YES
--     PFM:                         YES
--
--   Video I/O:
--     DC1394:                      YES (2.2.5)
--     FFMPEG:                      YES
--       avcodec:                   YES (58.54.100)
--       avformat:                  YES (58.29.100)
--       avutil:                    YES (56.31.100)
--       swscale:                   YES (5.5.100)
--       avresample:                NO
--     GStreamer:                   NO
--     v4l/v4l2:                    YES (linux/videodev2.h)
--
--   Parallel framework:            TBB (ver 2020.1 interface 11101)
--
--   Trace:                         YES (with Intel ITT)
--
--   Other third-party libraries:
--     Intel IPP:                   2020.0.0 Gold [2020.0.0]
--            at:                   /home/hao1/extra-environment/opencv460/opencv-4.6.0/build/3rdparty/ippicv/ippicv_lnx/icv
--     Intel IPP IW:                sources (2020.0.0)
--               at:                /home/hao1/extra-environment/opencv460/opencv-4.6.0/build/3rdparty/ippicv/ippicv_lnx/iw
--     VA:                          NO
--     Lapack:                      NO
--     Eigen:                       NO
--     Custom HAL:                  NO
--     Protobuf:                    build (3.19.1)
--
--   NVIDIA CUDA:                   YES (ver 11.6, CUFFT CUBLAS FAST_MATH)
--     NVIDIA GPU arch:             35 37 50 52 60 61 70 75 80 86
--     NVIDIA PTX archs:
--
--   cuDNN:                         YES (ver 8.6.0)
--
--   OpenCL:                        YES (no extra features)
--     Include path:                /home/hao1/extra-environment/opencv460/opencv-4.6.0/3rdparty/include/opencl/1.2
--     Link libraries:              Dynamic load
--
--   Python 3:
--     Interpreter:                 /usr/bin/python3 (ver 3.8.10)
--     Libraries:                   /usr/lib/x86_64-linux-gnu/libpython3.8.so (ver 3.8.10)
--     numpy:                       /home/hao1/.local/lib/python3.8/site-packages/numpy/core/include (ver 1.24.3)
--     install path:                /usr/lib/python3/dist-packages/cv2/python-3.8
--
--   Python (for build):            /usr/bin/python3
--
--   Java:
--     ant:                         NO
--     JNI:                         NO
--     Java wrappers:               NO
--     Java tests:                  NO
--
--   Install to:                    /usr/local/opencv4.6.0
-- -----------------------------------------------------------------
--
-- Configuring done
-- Generating done
-- Build files have been written to: /home/hao1/extra-environment/opencv460/opencv-4.6.0/build
hao1@hao1-PC:~/extra-environment/opencv460/opencv-4.6.0/build$
```

- make -j8 编译

```
[ 97%] Linking CXX shared library ../../lib/libopencv_cudaoptflow.so
[ 97%] Built target opencv_cudaoptflow
Scanning dependencies of target opencv_perf_cudaoptflow
[ 97%] Processing OpenCL kernels (superres)
[ 97%] Building NVCC (Device) object modules/superres/CMakeFiles/cuda_compile_1.dir/src/cuda/cuda_compile_1_generated_btv_l1_gpu.cu.o
Scanning dependencies of target opencv_test_cudaoptflow
[ 97%] Building NVCC (Device) object modules/videostab/CMakeFiles/cuda_compile_1.dir/src/cuda/cuda_compile_1_generated_global_motion.cu.o
[ 97%] Building CXX object modules/cudaoptflow/CMakeFiles/opencv_perf_cudaoptflow.dir/perf/perf_main.cpp.o
[ 97%] Building CXX object modules/cudaoptflow/CMakeFiles/opencv_test_cudaoptflow.dir/test/test_main.cpp.o
[ 97%] Building CXX object modules/cudaoptflow/CMakeFiles/opencv_perf_cudaoptflow.dir/perf/perf_optflow.cpp.o
[ 97%] Building CXX object modules/cudaoptflow/CMakeFiles/opencv_test_cudaoptflow.dir/test/test_optflow.cpp.o
[ 97%] Linking CXX executable ../../bin/opencv_perf_cudaoptflow
[ 97%] Built target opencv_perf_cudaoptflow
[ 97%] Linking CXX executable ../../bin/opencv_test_cudaoptflow
[ 97%] Built target opencv_test_cudaoptflow
Scanning dependencies of target opencv_superres
[ 97%] Building CXX object modules/superres/CMakeFiles/opencv_superres.dir/src/btv_l1.cpp.o
[ 98%] Building CXX object modules/superres/CMakeFiles/opencv_superres.dir/src/btv_l1_cuda.cpp.o
[ 98%] Building CXX object modules/superres/CMakeFiles/opencv_superres.dir/src/frame_source.cpp.o
[ 98%] Building CXX object modules/superres/CMakeFiles/opencv_superres.dir/src/input_array_utility.cpp.o
[ 98%] Building CXX object modules/superres/CMakeFiles/opencv_superres.dir/src/optical_flow.cpp.o
[ 98%] Building CXX object modules/superres/CMakeFiles/opencv_superres.dir/src/opencl_kernels_superres.cpp.o
[ 98%] Building CXX object modules/superres/CMakeFiles/opencv_superres.dir/src/super_resolution.cpp.o
[ 98%] Linking CXX shared library ../../lib/libopencv_superres.so
[ 98%] Built target opencv_superres
Scanning dependencies of target opencv_perf_superres
Scanning dependencies of target opencv_test_superres
[ 98%] Building CXX object modules/superres/CMakeFiles/opencv_test_superres.dir/test/test_main.cpp.o
[ 98%] Building CXX object modules/superres/CMakeFiles/opencv_perf_superres.dir/perf/perf_main.cpp.o
[ 98%] Building CXX object modules/superres/CMakeFiles/opencv_perf_superres.dir/perf/perf_superres.cpp.o
[ 98%] Building CXX object modules/superres/CMakeFiles/opencv_test_superres.dir/test/test_superres.cpp.o
[ 98%] Linking CXX executable ../../bin/opencv_test_superres
[ 98%] Built target opencv_test_superres
[ 98%] Linking CXX executable ../../bin/opencv_perf_superres
[ 98%] Built target opencv_perf_superres
Scanning dependencies of target opencv_videostab
[ 98%] Building CXX object modules/videostab/CMakeFiles/opencv_videostab.dir/src/deblurring.cpp.o
[ 98%] Building CXX object modules/videostab/CMakeFiles/opencv_videostab.dir/src/fast_marching.cpp.o
[ 98%] Building CXX object modules/videostab/CMakeFiles/opencv_videostab.dir/src/frame_source.cpp.o
[ 98%] Building CXX object modules/videostab/CMakeFiles/opencv_videostab.dir/src/global_motion.cpp.o
[ 98%] Building CXX object modules/videostab/CMakeFiles/opencv_videostab.dir/src/inpainting.cpp.o
[ 98%] Building CXX object modules/videostab/CMakeFiles/opencv_videostab.dir/src/log.cpp.o
[ 98%] Building CXX object modules/videostab/CMakeFiles/opencv_videostab.dir/src/motion_stabilizing.cpp.o
[ 98%] Building CXX object modules/videostab/CMakeFiles/opencv_videostab.dir/src/optical_flow.cpp.o
[100%] Building CXX object modules/videostab/CMakeFiles/opencv_videostab.dir/src/outlier_rejection.cpp.o
[100%] Building CXX object modules/videostab/CMakeFiles/opencv_videostab.dir/src/stabilizer.cpp.o
[100%] Building CXX object modules/videostab/CMakeFiles/opencv_videostab.dir/src/wobble_suppression.cpp.o
[100%] Linking CXX shared library ../../lib/libopencv_videostab.so
[100%] Built target opencv_videostab
Scanning dependencies of target opencv_test_videostab
Scanning dependencies of target opencv_python3
[100%] Building CXX object modules/videostab/CMakeFiles/opencv_test_videostab.dir/test/test_motion_estimation.cpp.o
[100%] Building CXX object modules/videostab/CMakeFiles/opencv_test_videostab.dir/test/test_main.cpp.o
[100%] Building CXX object modules/videostab/CMakeFiles/opencv_test_videostab.dir/test/test_stabilizer.cpp.o
[100%] Building CXX object modules/python3/CMakeFiles/opencv_python3.dir/__/src2/cv2.cpp.o
[100%] Building CXX object modules/python3/CMakeFiles/opencv_python3.dir/__/src2/cv2_numpy.cpp.o
[100%] Building CXX object modules/python3/CMakeFiles/opencv_python3.dir/__/src2/cv2_util.cpp.o
[100%] Building CXX object modules/python3/CMakeFiles/opencv_python3.dir/__/src2/cv2_highgui.cpp.o
[100%] Building CXX object modules/python3/CMakeFiles/opencv_python3.dir/__/src2/cv2_convert.cpp.o
[100%] Linking CXX executable ../../bin/opencv_test_videostab
In file included from /home/hao1/extra-environment/opencv460/opencv-4.6.0/build/modules/python_bindings_generator/pyopencv_generated_include.h:89,
                 from /home/hao1/extra-environment/opencv460/opencv-4.6.0/modules/python/src2/cv2.cpp:11:
/home/hao1/extra-environment/opencv460/opencv_contrib-4.6.0/modules/cudafeatures2d/include/opencv2/cudafeatures2d.hpp:121:26: warning: 'virtual void cv::cuda::DescriptorMatcher::clear()' can be marked override [-Wsuggest-override]
  121 |     CV_WRAP virtual void clear() = 0;
      |                          ^~~~~
/home/hao1/extra-environment/opencv460/opencv_contrib-4.6.0/modules/cudafeatures2d/include/opencv2/cudafeatures2d.hpp:125:26: warning: 'virtual bool cv::cuda::DescriptorMatcher::empty() const' can be marked override [-Wsuggest-override]
  125 |     CV_WRAP virtual bool empty() const = 0;
      |                          ^~~~~
[100%] Built target opencv_test_videostab
[100%] Linking CXX shared module ../../lib/python3/cv2.cpython-38-x86_64-linux-gnu.so
[100%] Built target opencv_python3
hao1@hao1-PC:~/extra-environment/opencv460/opencv-4.6.0/build$
hao1@hao1-PC:~/extra-environment/opencv460/opencv-4.6.0/build$
hao1@hao1-PC:~/extra-environment/opencv460/opencv-4.6.0/build$
```

- sudo make install



配置环境变量

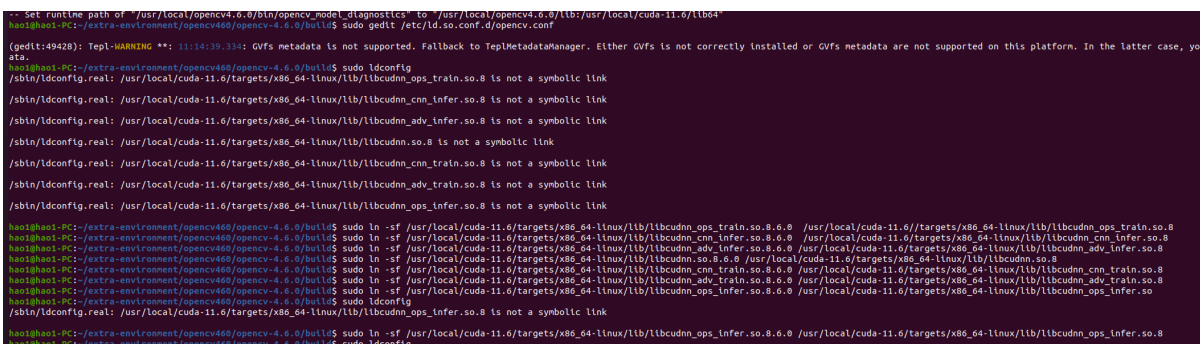
配置动态库环境

```
sudo gedit /etc/ld.so.conf.d/opencv.conf
# 填入下面的信息
/usr/local/opencv4.6.0/lib
# 保存配置文件
sudo ldconfig
```

解决一个cudnn找不到符号链接的问题：

参考：

https://blog.csdn.net/qq_36577574/article/details/119174973



简单来讲就是找不到符号链接就一个一个给他连接上。

注意上述这个也是安装GPU版本的opencv时才可能出现，不出现此情况请自行忽略。

- 配置其他库类（pkg，path等）

先打开bashrc文件

```
sudo gedit /etc/bash.bashrc
```

然后

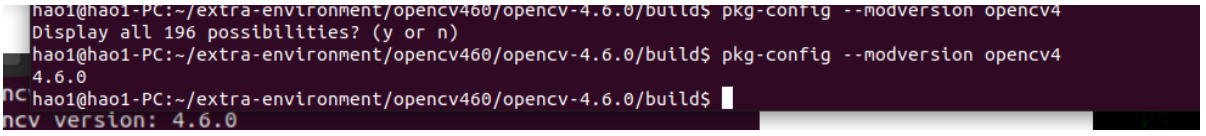
```
export PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/opencv4.6.0/lib/pkgconfig
export PATH=$PATH:/usr/local/opencv4.6.0/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/opencv4.6.0/lib
```

```
export PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/opencv4.6.0/lib/pkgconfig
export PATH=$PATH:/usr/local/opencv4.6.0/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/opencv4.6.0/lib
```

- 更新环境变量

```
source /etc/bash.bashrc
```

pkg测试

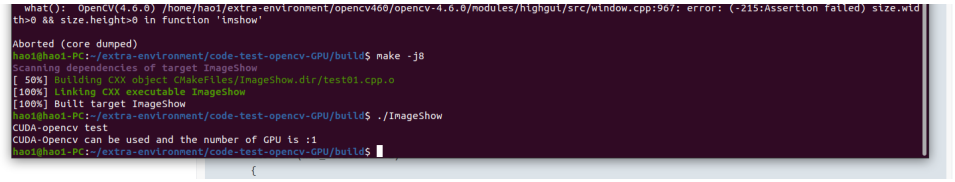


- 测试opencv和GPU:

```
#include <iostream>
#include <opencv2/core.hpp>
#include <opencv2/highgui.hpp>

#include <opencv2/opencv.hpp>

using namespace cv;
using namespace std;

using namespace cv::cuda;

void opencv_test()
{
    Mat image;
    image = imread("//home//hao1//extra-environment//code-test-opencv-
GPU//1.png");
    namedWindow("Display Image", WINDOW_AUTOSIZE );
    imshow("Display Image", image);
    waitKey(0);
}

void cuda_test()
{
    cout << "CUDA-opencv test" << endl;

    int num_devices = getCudaEnabledDeviceCount();
```

```
        if (num_devices == 0 )
        {
            std::cout << "OpenCV is compiled without CUDA support" << endl;
            return;
        }
        else if (num_devices == -1)
        {
            std::cout << "CUDA driver is not installed" << endl;
            return;
        }
        else if (num_devices >= 1)
        {
            std::cout << "CUDA-Opencv can be used and the number of GPU is :" <<
num_devices << endl;
            return;
        }
}

int main(int argc, char** argv )
{
    opencv_test();

    cuda_test();
    return 0;
}
```

测试效果：



成功引入OpenCv-GPU

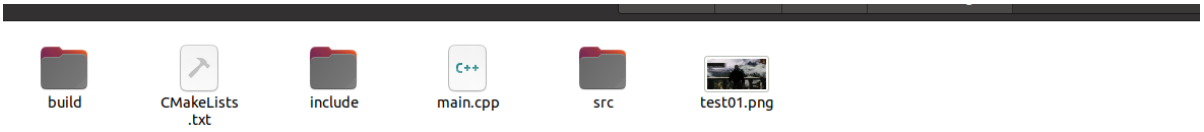
用CMake构建opencv工程

CMakeLists参考

```
cmake_minimum_required(VERSION 3.17)

set(CMAKE_CXX_STANDARD 17)

SET(PROJECT_NAME otest)
project(${PROJECT_NAME})

include_directories(
    ${PROJECT_SOURCE_DIR}/include/
    )                                                    #包含源路径下的所有的头文件

#包含opencv绝对路径
# SET(OpenCV_DIR D:/opencv-4.5.2/opencv/MingwBuild/install)##这里标明编译好的opencv
目录

find_package( OpenCV REQUIRED )                          ##找寻路径下的所需文件
```

```
include_directories( ${OpenCV_INCLUDE_DIRS})            #包含opencv库目录

#打印调试信息
MESSAGE(STATUS "Project: ${PROJECT_NAME}")
MESSAGE(STATUS "OpenCV library status:")
MESSAGE(STATUS "    version: ${OpenCV_VERSION}")
MESSAGE(STATUS "    libraries: ${OpenCV_LIBS}")
MESSAGE(STATUS "    include path: ${OpenCV_INCLUDE_DIRS}")

aux_source_directory(src DIR_SRCS)                      ##自动搜寻指定目录下的所有需
要的文件（.h .cpp)都可

#打印调试src获取的文件
MESSAGE(STATUS "Src file: ${DIR_SRCS}")

#编译添加可执行程序，命名为project name
add_executable(${PROJECT_NAME}
                main.cpp
               ${DIR_SRCS}
               )                                        ##添加可执行目标

target_link_libraries(${PROJECT_NAME} ${OpenCV_LIBS} )  ##链接opencv库
```

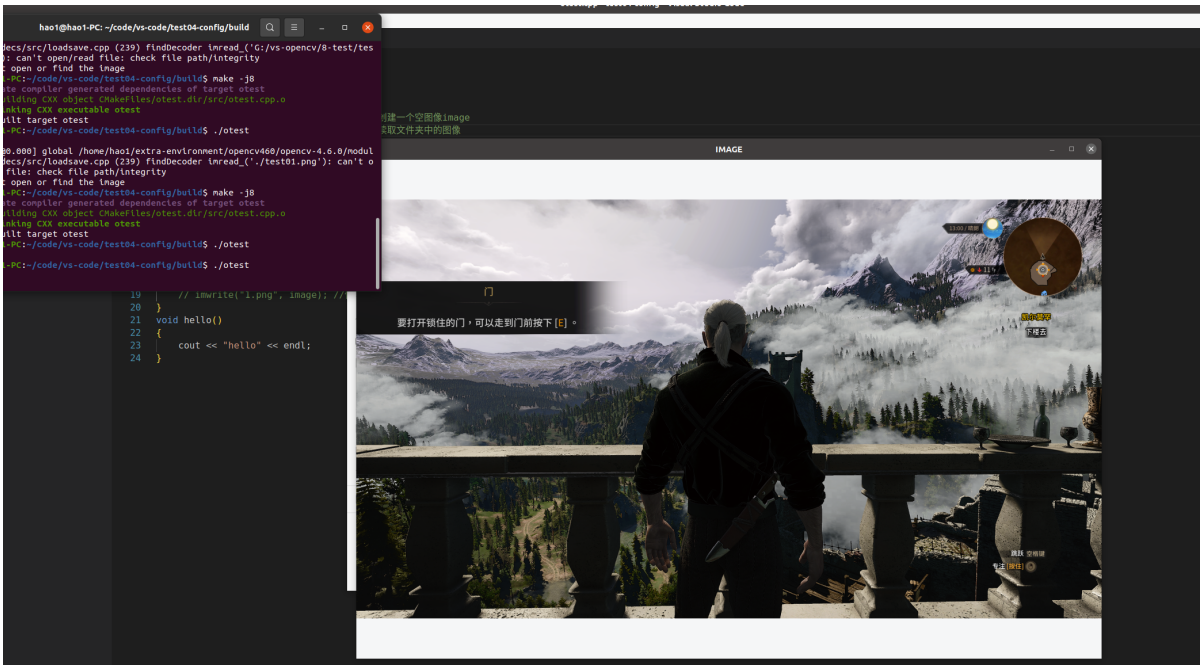
小demo文件结构：



写好上述的CMakeLists.txt文件，然后在build文件夹下：

```
cmake ..
make
```

即可，就生成可执行文件
效果：



用vscode配置opencv（json）（一般不用，感兴趣可以看看）

在vscode里配置下面的json文件即可

c_cpp_properties.json

```
{
    "configurations": [
        {
            "name": "win",
            "includePath": [
                "${workspaceFolder}/**",
                "/usr/local/opencv4.6.0/include/opencv4",  // 这里一定要写opencv4
                "/usr/local/opencv4.6.0/lib/",
                "/usr/local/opencv4.6.0/bin/",
                "${workspaceFolder}/include/",
                "${workspaceFolder}/src/"
            ],
            "defines": [],
            "compilerPath": "/usr/bin/g++",
            "cStandard": "c11",
            "cppStandard": "c++17",
            "intelliSenseMode": "${default}"
        }
    ],
    "version": 4
}
```

launch.json

```
{
    "version": "2.0.0",
    "configurations": [
        {
```

```
            "name": "opencv debuge",
            "type": "cppdbg",
            "request": "launch",
            "program": "${workspaceFolder}/Debugger/${fileBasenameNoExtension}",
            "args": [],
            "stopAtEntry": false, //这里如果为 false，则说明调试直接运行。（反之则停止）
            "cwd": "${workspaceFolder}",
            "environment": [],
            "externalConsole": true, //是否调用外部cmd
            "MIMode": "gdb",
            "miDebuggerPath": "/usr/bin/gdb", //自己进行设置
            "setupCommands": [
                {
                    "description": "为 gdb 启用整齐打印",
                    "text": "-enable-pretty-printing",
                    "ignoreFailures": false
                }
            ],
            "preLaunchTask": "opencv4.6.0"
        }
    ]
}
```

tasks.json

```
{
    "version": "2.0.0",
    "tasks": [
        {
            "type": "cppbuild",
            "label": "opencv4.6.0",
            "command": "/usr/bin/g++",
            "args": [
                "-g",
                "${cwd}/src/*.cpp",
                "${cwd}/*.cpp",
                "-I",
                "${workspaceFolder}/include/",
                "-I",
                "/usr/local/opencv4.6.0/include", //这里是opencv的包含文件
                "-I",
                "/usr/local/opencv4.6.0/include/opencv4", //包含文件里再一层opencv4
                "-I",
                "/usr/local/opencv4.6.0/include/opencv4/opencv2", //包含文件再一层
                "-L",
                "/usr/local/opencv4.6.0/lib/", //库文件
                "/usr/local/opencv4.6.0/lib/libopencv_*",
                "-o",
                "${workspaceFolder}/Debugger/${fileBasenameNoExtension}",
                "-lpthread"
            ],
            "options": {
                "cwd": "/usr/bin/"
            },
            "problemMatcher": [
```

```
                "$gcc"
            ],
            "group": {
                "kind": "build",
                "isDefault": true
            }
        }
    ]
}
```

上述的opencv路径根据实际情况修改

编译器路径为g++，调试器路径为gdb