How to Compile OpenCV 4.5.5 with CUDA 11.6 and cuDNN 8.3.2 on Ubuntu 22.04 (Daily Build)

February 5, 2022 by jlee

Upgrading to a new Alder Lake system has presented some challenges in terms of Linux compatibility. For instance, I'm using the daily build of Ubuntu 22.04 because I need the 5.15 kernel to support networking on the Z690 motherboard. There are also some changes and new errors to quash compared to installing OpenCV on 20.04. These instructions are specific to Ubuntu 22.04 in Feb 2022 (prior to the April release). Hopefully some of these workarounds won't be necessary as Ubuntu support improves for newer hardware and software.

Install NVIDIA drivers

Launch "Software & Updates." Under "Additional Drivers" install metapackage from nvidia-driver-510.

Install CUDA 11.6

Download CUDA from NVIDIA using deb (network). Since 22.04 isn't out yet, use the 20.04 version.

```
wget
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/
x86_64/cuda-ubuntu2004.pin
sudo mv cuda-ubuntu2004.pin /etc/apt/preferences.d/cuda-repository-
pin-600
sudo apt-key adv --fetch-keys
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/
x86_64/7fa2af80.pub
```

```
sudo add-apt-repository "deb
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/
x86_64/ /"
sudo apt-get update
sudo apt-get -y install cuda
```

In 22.04, I had the error below while installing.

```
Depends: liburcu6 but it is not installable
```

22.04 only comes with liburcu8, so download and install the liburcu6 package from https://packages.debian.org/bullseye/liburcu6. CUDA then installs correctly.

Per the CUDA installation instructions, add the following entries to the end of .bashrc

```
export PATH=/usr/local/cuda-11.6/bin${PATH:+:${PATH}}:/home/jlee/bin
export LD_LIBRARY_PATH=/usr/local/cuda-
11.6/lib64${LD_LIBRARY_PATH:+:${LD_LIBRARY_PATH}}
```

Install third party libraries

```
sudo apt-get install g++ freeglut3-dev build-essential libx11-dev
libxmu-dev libxi-dev libglu1-mesa libglu1-mesa-dev libfreeimage-dev
```

Install cuDNN 8.3.2

Per the cuDNN installation instructions. Note this is different from before, which downloaded the individual packages.

```
wget
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/
x86_64/cuda-ubuntu2004.pin

sudo mv cuda-ubuntu2004.pin /etc/apt/preferences.d/cuda-repository-
pin-600
sudo apt-key adv --fetch-keys
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/
x86_64/7fa2af80.pub
sudo add-apt-repository "deb
https://developer.download.nvidia.com/compute/cuda/repos/ubuntu2004/
```

```
x86_64/ /"
sudo apt-get update

sudo apt-get install libcudnn8=8.3.2.*-1+cuda11.5
sudo apt-get install libcudnn8-dev=8.3.2.*-1+cuda11.5
```

Install OpenCV 4.5.5

Setup python using virtualenvwrapper

```
sudo apt-get install python3-dev python3-pip python3-testresources

sudo pip install virtualenv virtualenvwrapper
```

Add virtualenvwrapper entries to .bashrc

```
export VIRTUALENVWRAPPER_PYTHON=/usr/bin/python3
export WORKON_HOME=$HOME/pyenvs
export PROJECT_HOME=$HOME/projects
source /usr/local/bin/virtualenvwrapper.sh
```

Create virtualenv for OpenCV, activate virtualenv (should be activated after making it), and install numpy.

```
mkvirtualenv opencv_cuda
workon opencv_cuda
pip install -U numpy
```

Install necessary system packages. Changes from 20.04:

- Remove libavresample-dev since it’s missing in 22.04
- Add libopenjp2-7-dev

```
sudo apt-get install build-essential cmake pkg-config unzip yasm git
checkinstall
sudo apt-get install libjpeg-dev libpng-dev libtiff-dev libopenjp2-
7-dev
sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev
sudo apt-get install libgstreamer1.0-dev libgstreamer-plugins-
base1.0-dev
sudo apt-get install libxvidcore-dev x264 libx264-dev libfaac-dev
```

```
libmp3lame-dev libtheora-dev
sudo apt-get install libfaac-dev libvorbis-dev
sudo apt-get install libopencore-amrnb-dev libopencore-amrwb-dev
sudo apt-get install libgtk-3-dev
sudo apt-get install libtbb-dev
sudo apt-get install libatlas-base-dev gfortran
sudo apt-get install libprotobuf-dev protobuf-compiler
sudo apt-get install libgoogle-glog-dev libgflags-dev
sudo apt-get install libgphoto2-dev libeigen3-dev libhdf5-dev
doxygen

sudo apt-get install libgtkglext1 libgtkglext1-dev

sudo apt-get install libopenblas-dev liblapacke-dev libva-dev
libopenjp2-tools libopenjpip-dec-server libopenjpip-server
libqt5opengl5-dev libtesseract-dev
```

Install Ceres Solver

```
sudo apt-get install cmake libeigen3-dev libgflags-dev libgoogle-
glog-dev libsuitesparse-dev libatlas-base-dev

git clone https://ceres-solver.googlesource.com/ceres-solver
cd ceres-solver
mkdir build && cd build
cmake ..
make -j4
make test
sudo make install
```

Download OpenCV and OpenCV Contrib

```
mkdir opencvbuild
cd opencvbuild
wget -O opencv.zip
https://github.com/opencv/opencv/archive/4.5.5.zip
wget -O opencv_contrib.zip
https://github.com/opencv/opencv_contrib/archive/4.5.5.zip
unzip opencv.zip
unzip opencv_contrib.zip
```

22.04 defaults to gcc version 11.2.0-1ubuntu1 (jammy). If you try to compile OpenCV, you'll get "error: parameter packs not expanded with" which seems to be due to changes in gcc 11.2. See here and here for a description.

Rather than try to downgrade gcc to 11.1 or edit the header files, I installed gcc-10 and g++-10 and then used update-alternatives to switch the compilers. OpenCV builds fine with gcc-10.

```
sudo apt-get install gcc-10
sudo apt-get install g++-10

sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-10
10
sudo update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-11
11
sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-10
10
sudo update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-11
11

sudo update-alternatives --config gcc
sudo update-alternatives --config g++
```

Build and Install OpenCV

- CMake config options description
- CUDA_ARCH_BIN corresponds to the compute capability for the graphics card listed on NVIDIA's site.
- QT has more features than GTK. When OPENGL is used, RGBD failed to compile with OpenGL_GL_PREFERENCE=GLVND, so OpenGL_GL_PREFERENCE=LEGACY is needed.

```
cd opencv-4.5.5
mkdir build
cd build

cmake -D CMAKE_BUILD_TYPE=RELEASE \
        -D CMAKE_INSTALL_PREFIX=/usr/local \
        -D INSTALL_PYTHON_EXAMPLES=ON \
        -D WITH_TBB=ON \
        -D OPENCV_ENABLE_NONFREE=ON \
        -D WITH_CUDA=ON \
        -D WITH_CUDNN=ON \
        -D OPENCV_DNN_CUDA=ON \
        -D ENABLE_FAST_MATH=1 \
        -D CUDA_FAST_MATH=1 \
        -D CUDA_ARCH_BIN=8.6 \
        -D WITH_CUBLAS=1 \
        -D WITH_OPENGL=ON \
        -D WITH_QT=ON \
        -D OpenGL_GL_PREFERENCE=LEGACY \
        -D
OPENCV_EXTRA_MODULES_PATH=/home/jlee/software/opencvbuild/opencv_con
trib-4.5.5/modules \
        -D
```

```
PYTHON_DEFAULT_EXECUTABLE=/home/jlee/pyenvs/opencv_cuda/bin/python \
        -D BUILD_EXAMPLES=ON ..

make -j$(nproc)
sudo make install
sudo ldconfig
```

Create symbolic link to OpenCV in virtualenv

```
cd ~/pyenvs/opencv_cuda/lib/python3.9/site-packages/
ln -s /usr/local/lib/python3.9/site-packages/cv2/python-
3.9/cv2.cpython-39-x86_64-linux-gnu.so cv2.so
```

Check that OpenCV works

```
workon opencv_cuda
python
import cv2
cv2.__version__
```