opencv install

linux

prereq: ubuntu 16:04 or 18:04

```
sudo apt-get install -y build-essential cmake
# GUI (if you want to use GTK instead of Qt, replace 'qt5-default' with
'libgtkglext1-dev' and remove '-DWITH_QT=ON' option in CMake):
sudo apt-get install -y libvtk6-dev libgtkglext1-dev

# GUI qt
# sudo apt install -y qt5-default

# Media I/O:
sudo apt-get install -y zlib1g-dev libjpeg-dev libwebp-dev libpng-dev
libtiff5-dev libjasper-dev libopenexr-dev libgdal-dev

# Video I/O:
sudo apt-get install -y libdc1394-22-dev libavcodec-dev libavformat-dev
libswscale-dev libtheora-dev libvorbis-dev libxvidcore-dev libx264-dev yasm
libopencore-amrnb-dev libopencore-amrwb-dev libv4l-dev libxine2-dev

# Parallelism and linear algebra libraries:
sudo apt-get install -y libtbb-dev libeigen3-dev

# Python 2 & 3:
sudo apt-get install -y python-dev python-tk python-numpy python3-dev
python3-tk python3-numpy

# others
sudo apt-get install -y libopenexr-dev



# Java:
# sudo apt-get install -y ant default-jdk

# Documentation:
# sudo apt-get install -y doxygen
```

download sources from release or ...

```
wget https://github.com/opencv/opencv/archive/3.4.1.zip
unzip 3.4.1.zip
rm 3.4.1.zip
mv opencv-3.4.1 OpenCV
```

```
cd OpenCV
```

... from git (better if compiling darknet)

```
cd /opt
git clone -b 3.4 --depth 1 https://github.com/opencv/opencv.git opencv.git
ln -sf opencv.git OpenCV
cd OpenCV
```

compile

```
mkdir build
cd build
# cmake -DWITH_QT=ON -DWITH_OPENGL=ON -DFORCE_VTK=ON -DWITH_TBB=ON -
DWITH_GDAL=ON -DWITH_XINE=ON -DBUILD_EXAMPLES=ON ..
# cmake -DWITH_OPENGL=ON -DFORCE_VTK=ON -DWITH_TBB=ON -DWITH_GDAL=ON -
DWITH_XINE=ON -DENABLE_PRECOMPILED_HEADERS=OFF -DWITH_JASPER=OFF ..
# with openCL
# cmake -DWITH_OPENGL=ON -DFORCE_VTK=ON -DWITH_TBB=ON -DWITH_GDAL=ON -
DWITH_XINE=ON -DENABLE_PRECOMPILED_HEADERS=OFF -DWITH_JASPER=OFF -
DWITH_OPENCL=ON ..
# with openCL and CUDA
#cmake -DWITH_OPENGL=ON -DFORCE_VTK=ON -DWITH_TBB=ON -DWITH_GDAL=ON -
DWITH_XINE=ON -DENABLE_PRECOMPILED_HEADERS=OFF -DWITH_JASPER=OFF -
DWITH_OPENCL=ON -DWITH_CUDA=ON ..
cmake -D CMAKE_BUILD_TYPE=Release -D CMAKE_INSTALL_PREFIX=/usr/local ..

time make -j4
# 13:42 on flinx
# 5:02 on mostro
```

install

```
sudo make install
sudo ldconfig
```

android method 1

prereq:

- substitute /opt/android-sdk/tools with https://dl.google.com/android/repository/tools_r25.2.5-linux.zip
- android sdk in /opt/android-sdk

```
cd /opt
git clone -b 3.4 --depth 1 https://github.com/opencv/opencv.git opencv.git
```

platforms/android/ndk.config.py

```
vars=dict(BUILD_ANDROID_PROJECTS='OFF')

ABIs = [
    #ABI("2", "armeabi-v7a", "arm-linux-androideabi-4.9",
cmake_vars=dict(ANDROID_ABI='armeabi-v7a with NEON')),
    #ABI("1", "armeabi",     "arm-linux-androideabi-4.9",
cmake_vars=dict(WITH_TBB='OFF')),
    ABI("3", "arm64-v8a",   "aarch64-linux-android-4.9",
cmake_vars=vars),
    ABI("5", "x86_64",      "x86_64-4.9", cmake_vars=vars),
    #ABI("4", "x86",         "x86-4.9"),
]
```

create /opt/OpenCV-android-sdk-compiled

```
cd /opt/opencv.git
python ./platforms/android/build_sdk.py \
 --config ndk.config.py \
 --ndk_path /opt/android-sdk/ndk-bundle \
 --sdk_path /opt/android-sdk \
 /opt/OpenCV-android-sdk-compiled .
```

android method 2

Thank you to everyone for your help.

I recently had to build OpenCV for Android from source (because I wanted to link against LLVM libc++ STL instead of GNU STL). In case it’s useful to anyone else, here are some quick instructions on how I did it:

```
    $ mkdir ~/sdk && cd ~/sdk
    $ wget https://dl.google.com/android/repository/tools_r25.2.5-macosx.zip
&& unzip tools_r25.2.5-macosx.zip
    $ tools/android sdk
    Select and install the following two packages:
        Android SDK Build-tools 27.0.3
        Android 8.1.0 (API 27) -> SDK Platform 27
    $ mkdir ~/opencv && cd ~/opencv
    $ git clone https://github.com/opencv/opencv.git
    $ cd ~/opencv/opencv
    $ wget
https://gist.githubusercontent.com/ngriffiths/296e2fc16b8586705712d50bdfe746
b0/raw/350a8cb6497bfe367dda75b7bb202d6c7ce09a72/always-use-
ant__find_android.patch && git apply always-use-ant__find_android.patch
    $ cd platforms/android
    $ ANDROID_SDK=~/sdk ANDROID_HOME=~/sdk ./build_sdk.py --config
ndk-16.config.py ~/opencv/build
```

I already had the ANDROID_NDK environment variable set to a local NDK 16 installation.

To compile against libc++, I then changed ANDROID_STL="gnustl_static" in build_sdk.py to ANDROID_STL="libc++_shared" and ran build_sdk.py again.

One unusual thing I couldn't work out was why the static libs I built were much larger than those in the official distribution (e.g. my arm64-v8a/libopencv_core.a is 75.8MB vs 7.5MB). @peterchaula peterchaula commented on Apr 11

@ngriffiths you need to strip out intel threading libs (TBB). That'll shave off ~15MB off your library.

android method 3

prereq:

- android SDK in REAL_ANDROID_SDK (REAL_ANDROID_SDK=/opt/android-sdk) with NDK installed

Download a stub android sdk and opencv sources

```
# adjust these paths ##
REAL_ANDROID_SDK=/opt/android-sdk
# output builded opencv, can be removed after
BASE=/opt/OpenCV-android-sdk.custom
########################

mkdir -p ${BASE}
cd ${BASE}
wget https://dl.google.com/android/repository/tools_r25.2.5-linux.zip
unzip tools_r25.2.5-linux.zip
tool/android sdk
Select and install the following two packages:
        Android SDK Build-tools 27.0.3
        Android 8.1.0 (API 27) -> SDK Platform

git clone --depth 1 https://github.com/opencv/opencv.git
cd opencv
wget
https://gist.githubusercontent.com/ngriffiths/296e2fc16b8586705712d50bdfe746
b0/raw/350a8cb6497bfe367dda75b7bb202d6c7ce09a72/always-use-
ant__find_android.patch && git apply always-use-ant__find_android.patch
git apply always-use-ant__find_android.patch

cd platform/android
ANDROID_SDK=${BASE} ANDROID_HOME=${BASE} ./build_sdk.py \
    --force_opencv_toolchain \
    --ndk_path ${REAL_ANDROID_SDK}/ndk-bundle \
    --config ndk-16.config.py \
    ${BASE}/build
```