OpenCV 3.x Installation Guide for MacOS (Python, C++)

OpenCV is an open source library for image processing thats supported in multiple languages. This guide explains installation steps for **C++** and **Python** only.

[Optional] Optimisation for Deep Neural Nets

Intel MKL (Math Kernel Library) can be installed to accelerate computation performance. It can be obtained for free from the official website for dmg installation.

It can be installed from source also.

Additionally IPP (Intel Performance Primitives) and TBB (Thread Building Blocks) can be installed for optimal code in multicore architectures from the same website linked above.

Install all these as `sudo` so the entire OS can take advantage of it.

DISCLAIMER: Do this only if you understand what your doing since it becomes really difficult to uninstall these packages if ever needed to do so.

Initial requirements

Before we can start downloading OpenCV we need `brew` to simplify our steps. Homebrew installs packages which are useful and isn't included by default by Apple.

Open Terminal and type to install Homebrew:

```
$ /usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

We need to install `wget` in order to download OpenCV:

```
$ brew install wget
```

We need a few other additional packages:

```
$ brew install git cmake pkg-config jpeg libpng libtiff openexr eigen tbb
```

We first need to download OpenCV code in order to compile them for use with other languages.

1. Create an empty folder in **Documents** and name it **OpenCV**.
2. Open **Terminal** and navigate to this folder.
   `$ cd Documents/OpenCV`
3. Download OpenCV here by typing (replace 3.3.0 with latest) :
   `$ wget https://github.com/Itseez/opencv/archive/3.3.0.zip`
4. `$ unzip 3*.zip`
5. **[Optional]** Download OpenCV_Contrib by typing:
   `$ wget https://github.com/Itseez/opencv_contrib/archive/3.3.0.zip`
   and unzip this too.

Compiling for Python

It is never a good idea to work with system Python so the ideal option would be to work on virtual environments or different Pythons.

1. Go to the website Miniconda and download Python 3.x 64-Bit (bash installer). (You can choose Python 2.x also.)
2. Open terminal and navigate to the folder where this was downloaded usually its the Downloads folder.
   `$ cd Downloads/`
3. `$ bash Miniconda3-latest-MacOSX-x86_64.sh`
4. Just type `yes` whenever asked during installation. This will add the new Python to your *PATH*. This will shadow your system python. You can always remove this later on if ever needed by commenting the miniconda line in *.bash_profile* file.
5. Once the installation is done, we need to activate the new PATH settings. This is done by:

   ```
   $ cd ~
   $  . .bash_profile
   ```

6. If all goes well, you can test it by typing
   `$ conda info`

to get an output like this:

```
Current conda install:

           platform : osx-64
      conda version : 4.3.22
   conda is private : False
  conda-env version : 4.3.22
conda-build version : not installed
     python version : 3.6.1.final.0
   requests version : 2.14.2
   root environment : /Users/rakshithgb/miniconda3  (writable)
default environment : /Users/rakshithgb/miniconda3
   envs directories : /Users/rakshithgb/miniconda3/envs
                      /Users/rakshithgb/.conda/envs
      package cache : /Users/rakshithgb/miniconda3/pkgs
                      /Users/rakshithgb/.conda/pkgs
       channel URLs : https://repo.continuum.io/pkgs/free/osx-64
                      https://repo.continuum.io/pkgs/free/noarch
                      https://repo.continuum.io/pkgs/r/osx-64
                      https://repo.continuum.io/pkgs/r/noarch
                      https://repo.continuum.io/pkgs/pro/osx-64
                      https://repo.continuum.io/pkgs/pro/noarch
        config file : None
         netrc file : None
       offline mode : False
         user-agent : conda/4.3.22 requests/2.14.2 CPython/3.6.1 Darwin/
16.7.0 OSX/10.12.6
            UID:GID : 501:20
```

7. OpenCV requires numpy which is installed by:

```
$ conda install numpy
```

8. Now that we have everything, we can compile OpenCV for Python. Navigate to the folder where we extracted OpenCV:

```
$ cd Documents/OpenCV/opencv*
```

9. `$ mkdir release`

10. `$ cd release`

11. Cmake command to compile:

```
$ cmake -DBUILD_TIFF=ON -DBUILD_opencv_java=OFF -DWITH_CUDA=OFF -DENABLE_AVX
=ON -DWITH_OPENGL=ON -DWITH_OPENCL=ON -DWITH_IPP=ON -DWITH_TBB=ON -DWITH_EIG
EN=ON -DWITH_V4L=ON -DWITH_VTK=OFF -DBUILD_TESTS=OFF -DBUILD_PERF_TESTS=OFF
-DCMAKE_BUILD_TYPE=RELEASE -DBUILD_opencv_python2=OFF -DCMAKE_INSTALL_PREFIX
=$(python3 -c "import sys; print(sys.prefix)") -DPYTHON3_EXECUTABLE=$(which
python3) -DPYTHON3_INCLUDE_DIR=$(python3 -c "from distutils.sysconfig import
 get_python_inc; print(get_python_inc())") -DPYTHON3_PACKAGES_PATH=$(python3
 -c "from distutils.sysconfig import get_python_lib; print(get_python_lib())
") ..
```

To compile with contrib (if you followed **optional** step):

NOTE: Change the path accordingly, in my case it was:

```
-DOPENCV_EXTRA_MODULES_PATH=/Users/rakshithgb/Documents/OpenCV/opencv_contri
b-3.3.0/modules
```

Replace this path and add it to your cmake command like this:

```
$ cmake -DOPENCV_EXTRA_MODULES_PATH=/Users/rakshithgb/Documents/OpenCV/openc
v_contrib-3.3.0/modules -DBUILD_TIFF=ON -DBUILD_opencv_java=OFF -DWITH_CUDA=
OFF -DENABLE_AVX=ON -DWITH_OPENGL=ON -DWITH_OPENCL=ON -DWITH_IPP=ON -DWITH_T
BB=ON -DWITH_EIGEN=ON -DWITH_V4L=ON -DWITH_VTK=OFF -DBUILD_TESTS=OFF -DBUILD
_PERF_TESTS=OFF -DCMAKE_BUILD_TYPE=RELEASE -DBUILD_opencv_python2=OFF -DCMAK
E_INSTALL_PREFIX=$(python3 -c "import sys; print(sys.prefix)") -DPYTHON3_EXE
CUTABLE=$(which python3) -DPYTHON3_INCLUDE_DIR=$(python3 -c "from distutils.
sysconfig import get_python_inc; print(get_python_inc())") -DPYTHON3_PACKAGE
S_PATH=$(python3 -c "from distutils.sysconfig import get_python_lib; print(g
et_python_lib())") ..
```

To build for python 2 just change all the `python3` references to `python2`.

If everything compiles then you will not get any errors. Once its finished compiling make sure its building for python3/2 accordingly.

1. `$ make -j4`
2. `$ make install`

Thats it, we can now use OpenCV with python.
Referenced the steps from here.

Errors

So when you `import cv2` in your python script if you get an error:

> ImportError: /media/BigData/miniconda3/lib/libstdc++.so.6: version `GLIBCXX_3.4.22' not found (required by /media/BigData/miniconda3/lib/libopencv_objdetect.so.3.2)

Use a more up to date system libstdc++ by:

```
$ cd ~/miniconda3/lib
$ rm libstdc++.so.6
$ ln -s /usr/lib/x86_64-linux-gnu/libstdc++.so.6
```

If you get error:

> /usr/include/c++/6/cstdlib:75:25: fatal error: stdlib.h: No such file or directory
> #include_next

or

> ~/miniconda3/lib/libopencv_objdetect.so.3.2: undefined symbol: ZTINSt6thread6StateE

Then try using `g++ 5` :

```
$ sudo update-alternatives --config g++
$ sudo update-alternatives --config gfortran
```

Spyder IDE for Python

A good IDE for python. Official website link here.

If `.dmg` version isn't available then install using:

```
$ conda install pyqt
$ conda install spyder
```

Start the app from terminal after that by:

```
$ spyder
```

Compiling for C++

First prerequisite is to install MacPorts. Download and install it from the official website. Verify the installation by opening Terminal and typing `$ port` , you should get something like this:

```
MacPorts 2.4.1
Entering shell mode... ("help" for help, "quit" to quit)
```

Now we have everything we need to compile for **C++**.

1. Navigate to the OpenCV directory where we extracted OpenCV:
   `$ cd Documents/OpenCV/opencv*`
2. `$ mkdir build`
3. `$ cd build`
4. Lets compile OpenCV now by typing:
   `$ cmake -G "Unix Makefiles" ..`
5. Once its successfully done execute:
   `$ make -j8`
6. `$ sudo make install`

Thats it OpenCV is compiled for c++ use now.