Installing piCamera module, OpenCV 3 on Raspbery Pi 3

Abstract: This document has been created, by PIRAUBE Nicolas, from several tutorials found on internet. The goal of this tutorial is to install OpenCV on a Raspberry Pi 3 as well as required dependencies and the Raspberry Pi camera module. The original tutorials can be found at the faloowing URL:

http://www.pyimagesearch.com/2015/10/26/how-to-install-opencv-3-on-raspbian-jessie/

http://pklab.net/?id=392&lang=EN

Table of Contents

Install dependencies

The first step to fallow is upgrading existing packages on the Raspberry Pi. To do so, open a terminal and type the fallowing commands :

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

```
sudo rpi-update
```

The last command is for updating the Raspberry Pi firmware.

You then need to reboot your Raspberry in order to take into account these upgrades. To reboot your raspberry type the fallowing command line :

```
sudo reboot
```

Installing developer tools packages

The first thing to do is to install the developer tools through the command line :

```
sudo apt-get install build-essential git cmake cmake-curses-gui pkg-config
```

The *build-essential* is a reference for all the packages needed to compile a Debian package (includes gcc/g++ compilers and other libraries).

The *git* software is a free open source distributed version control system used for handling projects.

Cmake is a family of tools designed to build, test and package softwares.

Pkg-config is a helper tool used when compiling applications and libraries.

Installing image input/output packages

The following command is used to install packages allowing to load image file formats (JPEG, PNG, TIFF...) :

```
sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
```

Installing video input/output packages

Just as above we need to install packages in order to load several video formats as well as work with video streams :

```
sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev
```

```
sudo apt-get install libxvidcore-dev libx264-dev
```

Installing GTK libraries & dependencies

The fallowing command line is for installing GTK development library to compile *highgui* allowing to display images and build simple GUI interface:

```
sudo apt-get install libgtk2.0-dev libeigen3-dev
```

Other dependencies can be added in order to optimise operations inside OpenCV throughout the fallowing command:

```
sudo apt-get install libatlas-base-dev gfortran
```

Installing camera module & packages

The Raspberry's camera is managed by Python through the picamera module. This module can be used in OpenCV but the images have to be grabbed to numpy.array the mapped in the OpenCV array Mat.

To use the grabbing loop cv2.VideoCapture(0) with the raspicam the Video4Linux driver is needed. The fallowing steps show how to install this driver.

The first step is checking prerequisites with :

```
sudo raspi-config
```

Once in the raspberry configuration enable the camera. Then set large memory for *gpu_mem* by going in the *Advance Options* > *Memory Split* and set it to 128 min.

The next step is installing the v4l library :

```
sudo apt-get -y install libv4l-dev v4l-utils
```

Then enable the kernel module :

```
sudo mobprobe bcm2835-v4l2
```

Test the module with :

```
v4l2-ctl –list-device
```

This command should result in something similar to the fallowing :

mmal service 16.1 (platform:bcm2835-v4l2) :

/dev/video0

You can then test to grab a single frame and check it through:

```
v4l2-ctl –set-fmt-video=width=800,height=600,pixelformat=3
```

```
v4l2-ctl –stream-mmap=3 –stream-count=1 –stream-to=test.jpg
```

A single frame should name *test.jpg* should appear in the folder. If this step worked fine you should add *bcm2835-v4l2* to the list of modules loaded at boot time in */etc/modules-load.d/modules.conf* file.

Setting up Python

Python 2.7 and Python 3 should already exist on the Raspberry Pi 3 but you can run the fallowing commands in order to check it :

```
sudo apt-get install python2.7-dev python2-numpy
```

```
sudo apt-get install python3-dev python3-numpy
```

Python is used afterwards in order to build and install OpenCV.

Intalling OpenCV

Now that the prerequisites have been installed, in this section we are going to download, configure and install OpenCV. It should be noted that at least 2GB are needed for OpenCV. Furthermore the download, build and installation of OpenCV can be done anywhere on the disk (SD Card), the libraries will be added automatically in the */usr/local* folders. Therefore the files used for the installation (the .zip files and the files contained in the *opencv-3.2.0* and *opencv-contrib-3.2.0* folders) can be deleted afterwards.

Downloading OpenCV

In order to download OpenCV open a terminal where you want the files to be downloaded and type the fallowing commands :

```
wget https://github.com/opencv/opencv/archive/3.2.0.zip -O opencv_source.zip
```

```
wget https://github.com/opencv/opencv_contrib/archive/3.2.0.zip -O opencv_contrib.zip
```

Unzip files

The .zip files dowloaded above shall be unzipped throught :

```
unzip opencv_source.zip
```

```
unzip opencv_contrib.zip
```

Build and install OpenCV

The first step is to create the working directory. To do so navigate into the unzipped folder *opencv-3.2.0* and type the fallowing commands :

```
mkdir build
```

```
cd build
```

Configure build directory

Within the build directory type the fallowing command in order to configure the build using command line switch :

```
cmake -D CMAKE_BUILD_TYPE=RELEASE \
      -D CMAKE_INSTALL_PREFIX=/usr/local \
      -D BUILD_WITH_DEBUG_INFO=OFF \
      -D BUILD_DOCS=OFF \
      -D BUILD_EXAMPLES=OFF \
      -D BUILD_TESTS=OFF \
      -D BUILD_opencv_ts=OFF \
      -D BUILD_PERF_TESTS=OFF \
      -D INSTALL_C_EXAMPLES=ON \
      -D INSTALL_PYTHON_EXAMPLES=ON \
      -D OPENCV_EXTRA_MODULES_PATH=../../opencv_contrib-3.2.0/modules \
      -D ENABLE_NEON=ON \
      -D WITH_LIBV4L=ON \
      ../
```

Once the configuration is done the fallowing line should appear at the end :

```
- - Configuration done
- - Generating done
```

Review/Modify configuration using Cmake TextGUI

Type the fallowing command line in the buil directory :

```
ccmake ../
```

A GUI will appear in order to configure the wanted options in OpenCV. Scroll through the list and change the wanted options.

When finished press *c* to configure. Then press *g* to generate the makefile.

Build OpenCV

The next step is building OpenCV. It should be noted that it takes a while to build, approximatly 2 hours. To do so type the fallowing command (still in the build folder) :

```
make
```

Install OpenCV

Now that the build of OpenCV has successfully been done, the istallation can be done through the fallowing command :

sudo make install

sudo ldconfig

OpenCV is now installed on the Raspberry Pi 3 and is ready to be used. The library files have been installed in the */usr/local* folders.

Testing the installation

In order to test the installation, a simple program written in C++ can be created in order to enable a live footage through the picamera. To do so, create a new file called *SimpleGrab.cpp* and add in it the fallowing code :

```
#include <stdio.h>
#include <opencv2/opencv.hpp>
#include <opencv2/highgui.hpp>
using namespace cv;
using namespace std;
int main(int argc,char ** argv)
{
 VideoCapture cap(0);
 if (!cap.isOpened()) {
  cerr << "ERROR: Unable to open the camera" << endl;
  return 0;
 }
 Mat frame;
 cout << "Start grabbing, press a key on Live window to terminate" << endl;
 while(1) {
  cap >> frame;
  if (frame.empty()) {
     cerr << "ERROR: Unable to grab from the camera" << endl;
     break;
  }
  imshow("Live",frame);
  int key = cv::waitKey(5);
  key = (key==255) ? -1 : key; //#Solve bug in 3.2.0
  if (key>=0)
    break;
 }
 cout << "Closing the camera" << endl;
 cap.release();
 destroyAllWindows();
 cout << "bye!" <<endl;
 return 0;
}
```