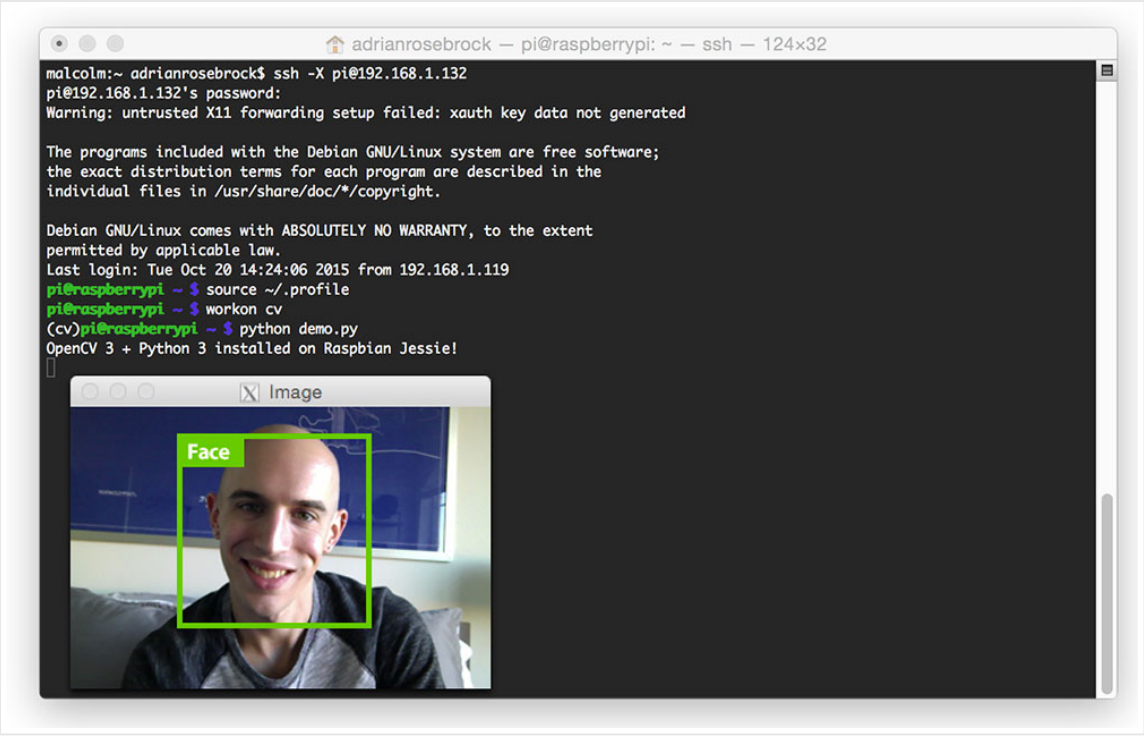
Navigation

pyimagesearch
be awesome at building image search engines

How to install OpenCV 3 on Raspbian Jessie

by **Adrian Rosebrock** on October 26, 2015 in **OpenCV 3**, **Raspberry Pi**, **Tutorials**



A few weeks ago Raspbian Jessie was released, bringing in a ton of new, great features.

However, the update to Jessie also broke the previous OpenCV + Python install instructions for Raspbian Wheezy:

- Install OpenCV 2.4 with Python 2.7 bindings on Raspbian Wheezy.
- Install OpenCV 3.0 with Python 2.7/Python 3+ bindings on Raspbian Wheezy.

Since PyImageSearch has become *the online destination* for learning computer vision + OpenCV on the Raspberry Pi, I decided to write a new tutorial on **installing OpenCV 3 with Python bindings on Raspbian Jessie.**

As an additional bonus, **I've also included a *video tutorial*** that you can use to follow along with me as I install OpenCV 3 on my own Raspberry Pi 2 running Raspbian Jessie.

This video tutorial should help address the most common questions, doubts, and pitfalls that arise when installing OpenCV + Python bindings on the Raspberry Pi for the first time.

Assumptions

For this tutorial I am going to assume that you already own a Raspberry Pi 2 with Raspbian Jessie installed. Other than that, you should either have (1) *physical access* to your Pi 2 and can open up a terminal or (2) *remote access* where you can SSH in. I'll be doing this tutorial via SSH, but as long as you have access to a terminal, it really doesn't matter.

The quick start video tutorial

Before we get this tutorial underway, let me ask you two quick questions:

1. Is this your first time installing OpenCV?
2. Are you just getting started learning Linux and how to use the command line?

If you answered ***yes*** to either of these questions, I *highly suggest* that you watch the video below and follow along with me as a guide you step-by-step on how to install OpenCV 3 with Python bindings on your Raspberry Pi 2 running Raspbian Jessie:

How to install OpenCV 3 on Raspbian Jessie

Otherwise, if you feel comfortable using the command line or if you have previous experience using the command line, feel free to follow the tutorial below.

Installing OpenCV 3 on Raspbian Jessie

Installing OpenCV 3 is a multi-step (and even time consuming) process requiring you to install many dependencies and pre-requisites. The remainder of this tutorial will guide you step-by-step through the process.

To make the installation process easier, I've included timings for each step (when appropriate) so you know when to stick by your terminal, grab a cup of coffee, or go for a nice long walk.

If you're an experienced Linux user or have already installed OpenCV on a Raspberry Pi (or another other system) before, you can likely just follow the steps outlined below.

However, if this is your first time installing OpenCV (or you don't have much prior exposure to the Linux operating systems and the command line), ***I highly recommend that you watch the video above and follow along with me*** as I show you how to install OpenCV 3 on your Rasberry Pi running Raspbian Jessie.

That said, let's get started installing OpenCV 3.

Step #1: Install dependencies

The first thing we should do is update and upgrade any existing packages, followed by updating the Raspberry Pi firmware.

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ sudo apt-get
2 $ sudo apt-get upgrade
3 $ sudo rpi-update
```

Timing: 3m 33s

You'll need to reboot your Raspberry Pi after the firmware update:

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ sudo reboot
```

Now we need to install a few developer tools:

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ sudo apt-get install build-essential git cmake pkg-config
```

Timing: 51s

Now we can move on to installing image I/O packages which allow us to load image file formats such as JPEG, PNG, TIFF, etc.:

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev
```

Timing: 42s

Just like we need image I/O packages, we also need video I/O packages. These packages allow us to load various video file formats as well as work with video streams:

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-
2 $ sudo apt-get install libxvidcore-dev libx264-dev
```

Timing: 58s

We need to install the GTK development library so we can compile the `highgui` sub-module of OpenCV, which

allows us to display images to our screen and build simple GUI interfaces:

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ sudo apt-get install libgtk2.0-dev
```

Timing: 2m 48s

Various operations inside of OpenCV (such as matrix operations) can be optimized using added dependencies:

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ sudo apt-get install libatlas-base-dev gfortran
```

Timing: 50s

Lastly, we'll need to install the Python 2.7 and Python 3 header files so we can compile our OpenCV + Python bindings:

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ sudo apt-get install python2.7-dev python3-dev
```

Step #2: Grab the OpenCV source code

At this point we have all of our prerequisites installed, so let
OpenCV repository. (*Note*: As future versions of OpenCV a
recent version number):

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ cd ~
2 $ wget -O opencv.zip https://github.com/Itseez/ope
3 $ unzip opencv.zip
```

Timing: 2m 29s

For the full install of OpenCV 3 (which includes features su
opencv_contrib repo as well. (*Note:* Make sure your `openc
you will run into errors during compilation. For example, if I
v3.0.0 of `opencv_contrib` as well):

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ wget -  opencv_contrib.zip https://github.com/Itseez/opencv_contrib/archive/3.0.0.zip
2 $ unzip opencv_contrib.zip
```

Timing: 1m 54s

Step #3: Setup Python

The first step in setting up Python for our OpenCV compile is to install `pip` , a Python package manager:

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ wget https://bootstrap.pypa.io/get-pip.py
2 $ sudo python get-pip.py
```

Timing: 26s

I've discussed both virtualenv and virtualenvwrapper *many* times on the PyImageSearch blog before, especially

within these installation tutorials. Installing these packages is certainly not a requirement to get OpenCV and Python up and running on your Raspberry Pi, ***but I highly recommend that you install them!***

Using `virtualenv` and `virtualenvwrapper` allows you to create *isolated Python environments*, separate from your system install of Python. This means that you can run multiple versions of Python, with different versions of packages installed into each virtual environment — this solves the *"Project A depends on version 1.x, but Project B needs* 4.x" problem that often arises in software engineering.

Again, it's *standard practice* in the Python community to use virtual environments, so I highly suggest that you start using them if you are not already:

```
Installing OpenCV 3 on Raspbian Jessie                                  Shell
1 $ sudo pip install virtualenv
2 $ sudo rm -rf ~/.cache/pip
```

Timing: 17s

After `virtualenv` and `virtualenvwrapper` have been installed, we need to update our `~/.profile` file and insert the following lines at the bottom of the file:

```
Installing OpenCV 3 on Raspbian Jessie                                  Shell
1 # virtualenv and virtualenvwrapper
2 export WORKON_HOME=$HOME/.virtualenvs
3 source /usr/local/bin/virtualenvwrapper.sh
```

You can use your favorite editor to edit this file, such as `vim` , `emacs` , `nano` , or any other graphical editor included in the Raspbian Jessie distribution. Again, all you need to do is open the file located at `/home/pi/.profile` and insert the lines above at the bottom of the file.

Now that your `~/.profile` has been updated, you need to reload it so the changes can take affect. To force a reload of the `~/.profile` file you can (1) logout and log back in, (2) close your terminal and open up a new one, or (3) just use the `source` command:

```
Installing OpenCV 3 on Raspbian Jessie                                  Shell
1 $ source ~/.profile
```

Note: You'll likely need to run the `source ~/.profile` command ***each*** time you open up a new terminal to ensure your environment has been setup correctly.

The next step is to create our Python virtual environment where we'll be doing our computer vision work:

```
Installing OpenCV 3 on Raspbian Jessie                                  Shell
1 $ mkvirtualenv cv
```

The above command will create a virtual environment named `cv` using ***Python 2.7***.

If you want ***Python 3***, run this command instead:

```
Installing OpenCV 3 on Raspbian Jessie                                  Shell
1 $ mkvirtualenv cv -p python3
```

Again, it's important to note that the `cv` Python environment is ***entirely independent*** from the default version of Python included in the download of Raspbian Jesse.

If you ever reboot your system, logout and log back in, or open up a new terminal, you'll need to use the `workon` command to re-access the `cv` virtual environment, otherwise you'll be using the system version of Python instead:

Installing OpenCV 3 on Raspbian Jessie — Shell

```
$ source ~/.profile
$ workon cv
```

You can ensure you are in the `cv` virtual environment by examining your command line. If you see the text *"(cv)"* preceding your prompt, *then you **are** in the `cv` virtual environment*:

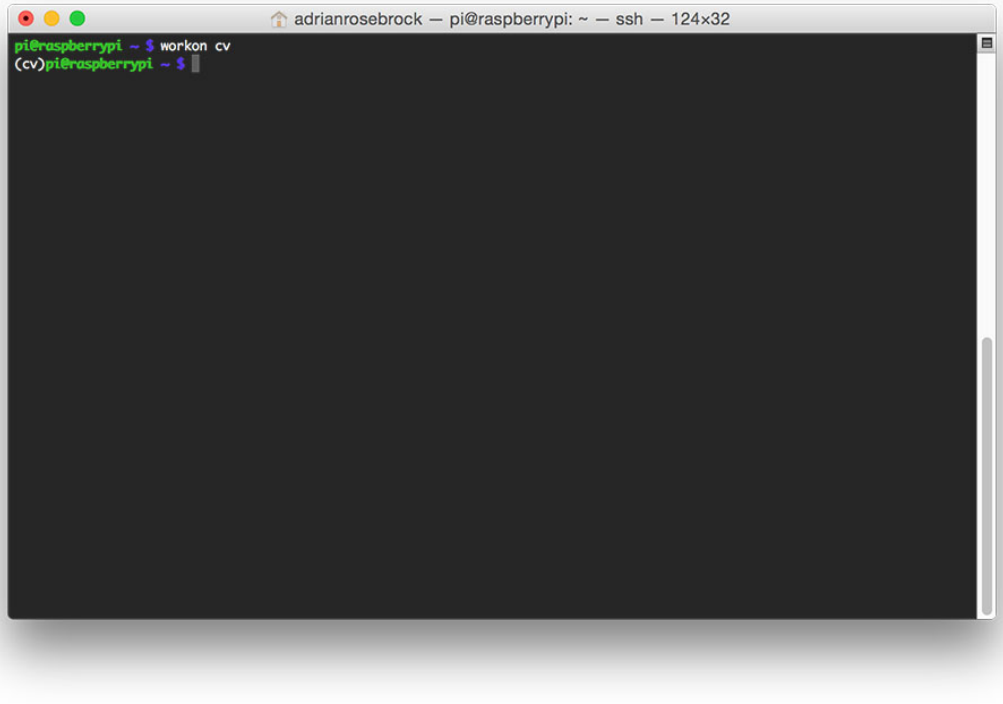


Figure 1: Make sure you see the *"(cv)"* text on your prompting, indicating that you ***are*** in the *cv* virtual environment.

Otherwise, *you **are not** in the `cv` virtual environment:*

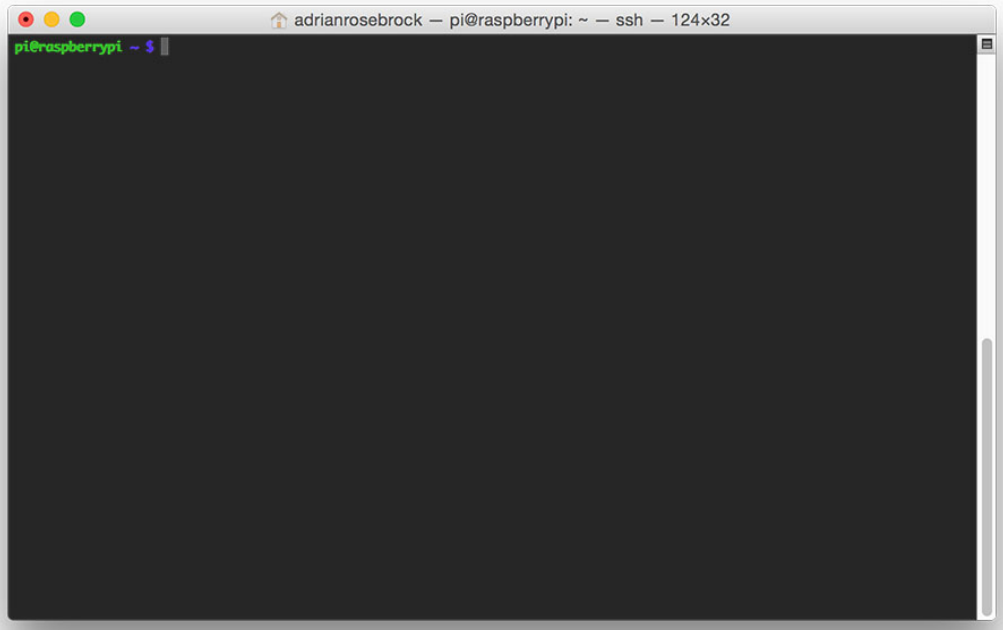


Figure 2: If you ***do not*** see the *"(cv)"* text on your prompt, then you ***are not*** in the *cv* virtual environment.

If this is the case, you need to run the `source` and `workon` commands above.

Assuming that you are in the `cv` virtual environment, we can install NumPy, an important dependency when compiling the Python bindings for OpenCV. You might want to grab a cup of coffee or go for a walk while NumPy downloads and installs:

```
Installing OpenCV 3 on Raspbian Jessie                                  Shell
1 $ pip install numpy
```

Timing: 16m 10s

Step #4: Compile and install OpenCV

At this point, we are ready to compile OpenCV.

First, make sure you are in the `cv` virtual environment:

```
Installing OpenCV 3 on Raspbian Jessie                                  Shell
1 $ workon cv
```

Followed by setting up the build:

```
Installing OpenCV 3 on Raspbian Jessie                                  Shell
1 $ cd ~/opencv-3.0.0/
2 $ mkdir build
3 $ cd
4 $ cmake -D CMAKE_BUILD_TYPE=RELEASE \
5     -  CMAKE_INSTALL_PREFIX=/usr/       \
6     -D INSTALL_C_EXAMPLES=ON \
7     -  INSTALL_PYTHON_EXAMPLES=    \
8     -D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib-3.0.0/modules \
9     -  BUILD_EXAMPLES=    ..
```

Update (3 January 2016): *In order to build OpenCV* `3.1.0` *, you need to set* `-D INSTALL_C_EXAMPLES=OFF` *(rather than* `ON` *) in the* `cmake` *command. There is a bug in the OpenCV v3.1.0 CMake build script that can cause errors if you leave this switch on. Once you set this switch to off, CMake should run without a problem.*

Before you move on to the compilation step, ***make sure you examine the output of CMake!***

Scroll down the section titled `Python 2` and `Python 3` .

If you're compiling OpenCV 3 for Python 2.7, then you'll want to make sure the `Python 2` section looks like this (highlighted) in red:

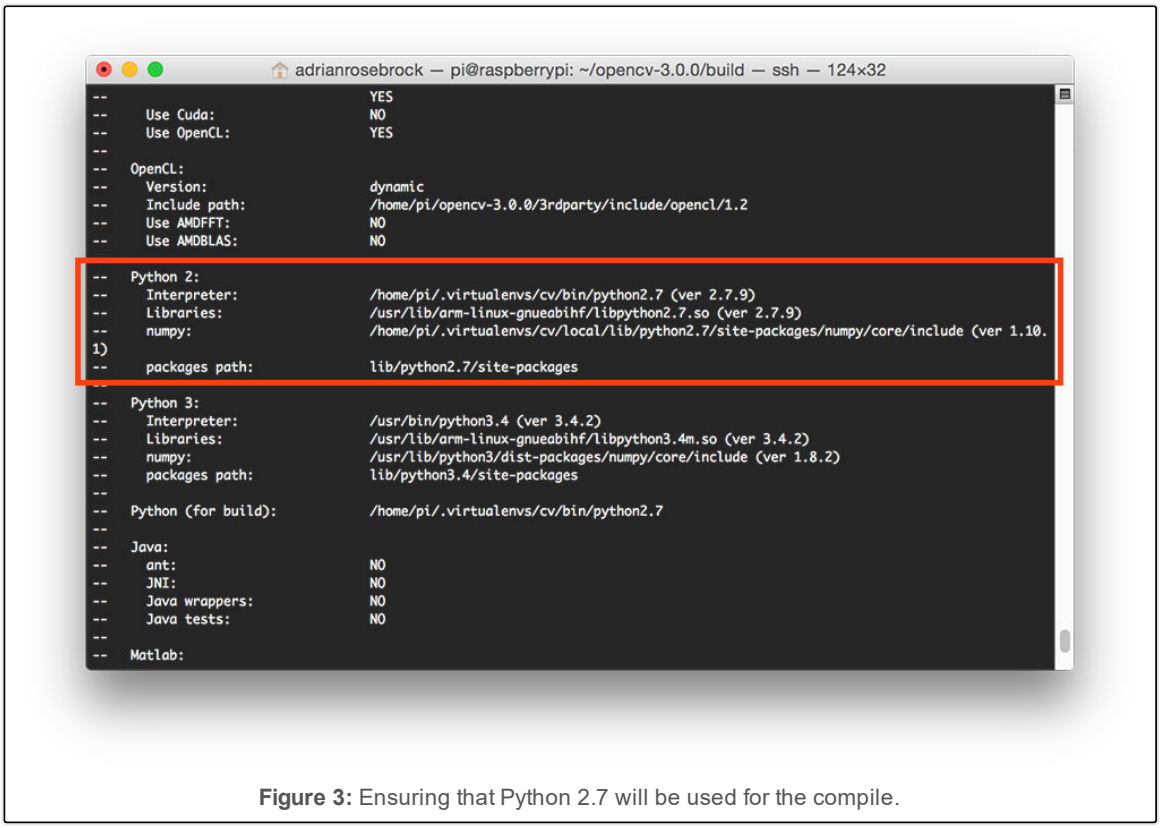


Figure 3: Ensuring that Python 2.7 will be used for the compile.

Notice how both the `Interpreter` and `numpy` variables point to the `cv` virtual environment.

Similarly, **if you're compiling OpenCV for Python 3**, then make sure the `Python 3` section looks like this:

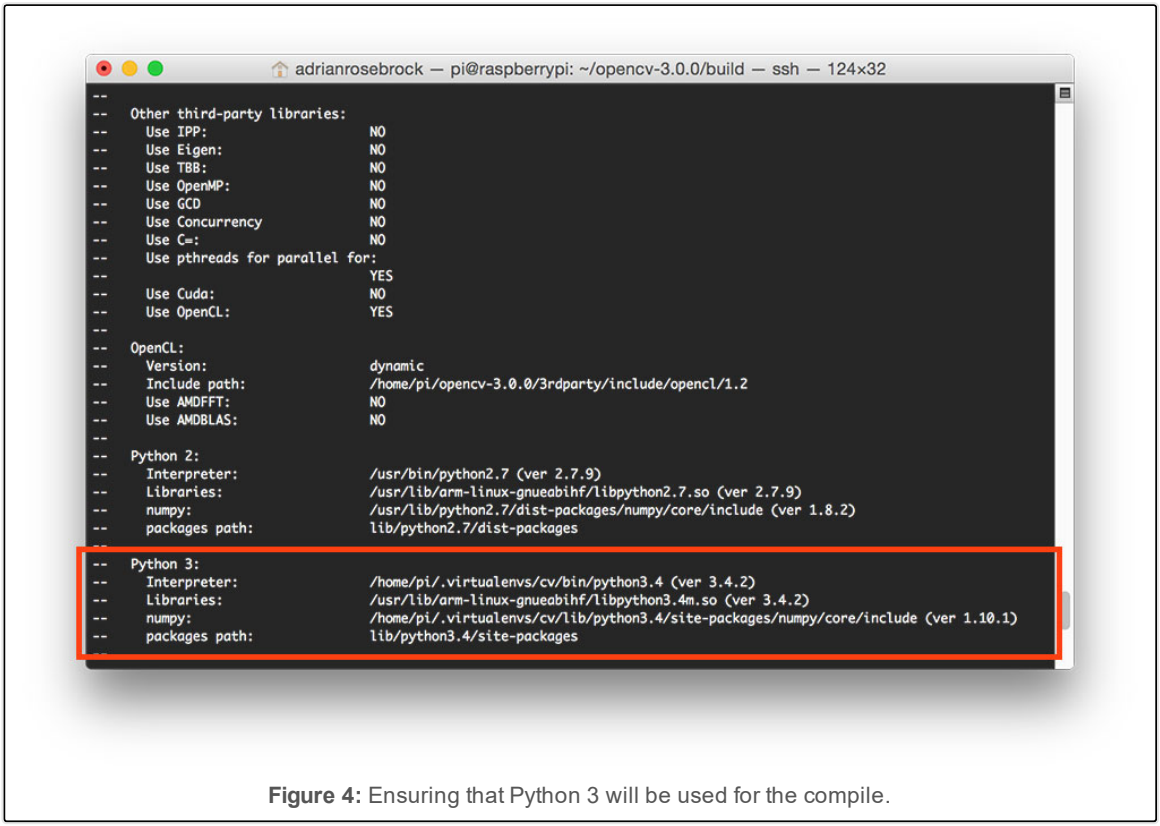


Figure 4: Ensuring that Python 3 will be used for the compile.

Again, both the `Interpreter` and `numpy` variables are pointing to our `cv` virtual environment.

In either case, if you ***do not*** see the `cv` virtual environment for these variables **MAKE SURE YOU ARE IN THE `cv` VIRTUAL ENVIRONMENT PRIOR TO RUNNING CMAKE!**

Now that our build is all setup, we can compile OpenCV:

```
Installing OpenCV 3 on Raspbian Jessie                    Shell
1 $ make -j4
```

Timing: 1h 35m

The `-j4` switch stands for the number of cores to use when compiling OpenCV. Since we are using a Raspberry Pi 2, we'll leverage all four cores of the processor for a faster compilation.

However, if your `make` command errors out, I would suggest starting the compilation over again and only using one core:

```
Installing OpenCV 3 on Raspbian Jessie                    Shell
1 $ make
2 $ make
```

Using only one core will take much longer to compile, but can help reduce any type of strange race dependency condition errors when compiling.

Assuming OpenCV compiled without error, all we need to do is install it on our system:

```
Installing OpenCV 3 on Raspbian Jessie                    Shell
1 $ sudo make
2 $ sudo ldconfig
```

Step #5: Finishing the install

We're almost there! Just a few more things and we'll be 100% done.

For Python 2.7:

Provided you finished **Step #4** without error, OpenCV should now be installed in `/usr/local/lib/python2.7/site-packages` :

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ ls -l /usr/local/lib/python2.7/site-packages/
2 total 1636
3 -rw-r--r-- 1 root       1675144     17 15:25 cv2.so
```

Note: In some instances OpenCV can be installed in `/usr/local/lib/python2.7/dist-packages` (note the `dist-packages` rather than `site-packages`). If you *do not* find the `cv2.so` bindings in `site-packages` , be sure to check `dist-packages` as well.

The last step here is to sym-link the OpenCV bindings into the `cv` virtual environment:

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ cd ~/.virtualenvs/cv/lib/python2.7/site-packages/
2 $ ln -s /usr/local/lib/python2.7/site-packages/cv2.so cv2.so
```

For Python 3:

OpenCV should now be installed in `/usr/local/lib/python3.4/site-packages` :

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ ls /usr/local/lib/python3.4/site-packages/
2 cv2.cpython-34m.so
```

For some reason, unbeknownst to me, when compiling the Python 3 bindings the output `.so` file is named `cv2.cpython-34m.so` rather than `cv2.so` .

Luckily, this is an easy fix. All we need to do is rename the file:

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ cd /usr/local/lib/python3.4/site-packages/
2 $ sudo mv cv2.cpython-34m.so cv2.so
```

Followed by sym-linking OpenCV into our `cv` virtual environment:

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ cd ~/.virtualenvs/cv/lib/python3.4/site-packages/
2 $ ln -s /usr/local/lib/python3.4/site-packages/cv2.so cv2.so
```

Step #6: Verifying your OpenCV 3 install

At this point, OpenCV 3 should be installed on your Raspberry Pi running Raspbian Jessie!

But before we wrap this tutorial up, let's verify that your OpenCV installation is working by accessing the `cv` virtual environment and importing `cv2` , the OpenCV + Python bindings:

```
Installing OpenCV 3 on Raspbian Jessie                                    Shell
1 $ workon cv
```

```
$ python
>>> import cv2
>>> cv2.__version__
'3.0.0'
```

You can see a screenshot of my terminal below, indicating that OpenCV 3 has been successfully installed:

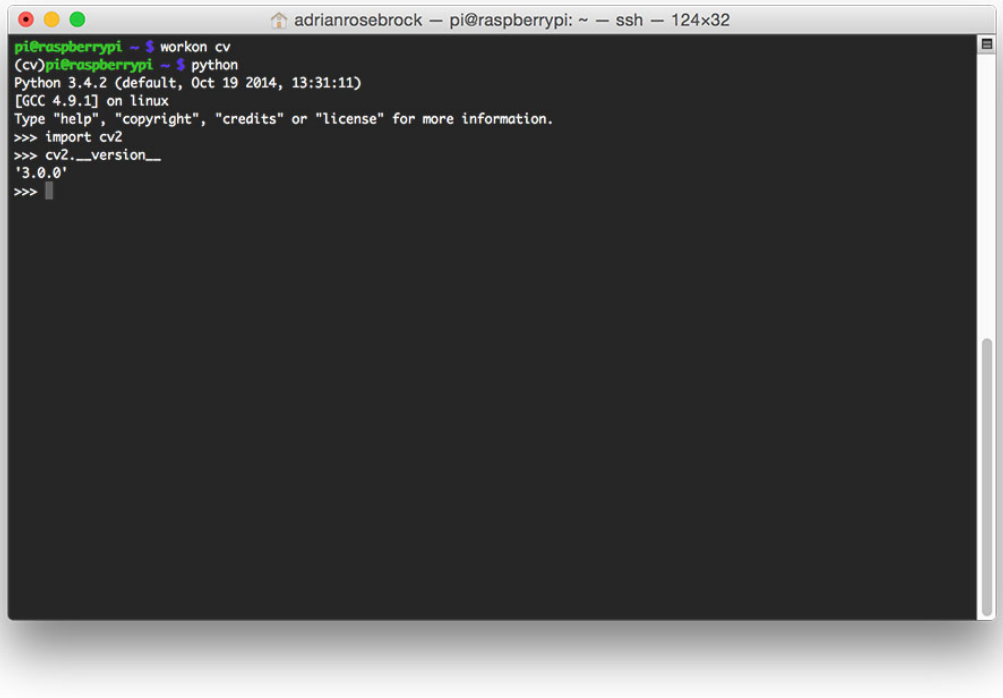


Figure 5: OpenCV 3 + Python 3 bindings have been successfully installed on my Raspberry Pi 2 running Rasbian Jessie.

Troubleshooting

Q. When I try to use the `mkvirtualenv` or `workon` commands, I get an error saying "command not found".

A. Go back to **Step #3** and ensure your `~/.profile` file has been updated properly. Once you have updated it, be sure to run `source ~/.profile` to reload it.

Q. After I reboot/logout/open up a new terminal, I cannot run the `mkvirtualenv` or `workon` commands.

A. Anytime you reboot your system, logout and log back in, or open up a new terminal, you should run `source ~/.profile` to make sure you have access to your Python virtual environments.

Q. When I open up a Python shell and type `import cv2`, I get the dreaded `ImportError: No module named cv2` error.

A. The reason for this error is hard to diagnose, mainly because there are multiple issues that could be causing this problem. For starters, make sure you are in the `cv` virtual environment using `workon cv`. If the `workon` command is giving you problems, then see the previous questions in this section. From there, you'll want to investigate the `site-packages` directory of your `cv` virtual environment located in `~/.virtualenvs/cv/lib/python2.7/site-packages/` or `~/.virtualenvs/cv/lib/python3.4/site-packages/`, respectively. Make sure that the sym-link path to the `cv2.so` file is valid. If you do not know how to do

this, **please consult the video tutorial at the top of this post.**

Summary

In this lesson we learned how to install OpenCV 3 with Python 2.7 and Python 3 bindings on your Raspberry Pi 2 running Raspbian Jessie. I provided timings for each step so ensure you can plan your install accordingly.

It's also worth mentioning that I provide OpenCV v2.4 and v3 install instructions for Raspbian Wheezy in the following posts:

- Install OpenCV 2.4 with Python 2.7 bindings on Raspbian Wheezy.
- Install OpenCV 3.0 with Python 2.7/Python 3+ bindings on Raspbian Wheezy.

If you run into any issues during the installation process, please see the **Troubleshooting** section above. Additionally, I would suggest watching the **video tutorial** at the top of this post to aid you in the setup process.

Before you go…

I tend to cover a lot of great computer vision projects using OpenCV and the Raspberry Pi, ***so consider entering your email address in the form below to be notified when these posts go live!***

install, **opencv 3.0**, **pi 2**, **python 2.7**, **python 3.4**, **raspberry pi**, **tutorials**

‹ Analyzing 91 years of Time magazine covers for visual trends

Watershed OpenCV ›

132 Responses to *How to install OpenCV 3 on Raspbian Jessie*

Anastasios Selalmazidis October 26, 2015 at 12:56 pm #

REPLY

Great guide Adrian. I was looking forward for this. Worked like a charm for python2.7.
I am going to install it now for python3.

Thank you !!!

REPLY

Adrian Rosebrock October 27, 2015 at 4:52 am #

Awesome, I'm glad it worked for you Anastasios! 🙂

REPLY

Aldi December 9, 2015 at 11:54 pm #

Hello Adrian thanks for your tutorial it was great. I've just followed trough your guide and I am able to do the example from your site but How can I run the example from the opencv ?
Thank you for your attention

REPLY

Adrian Rosebrock December 10, 2015 at 6:50 am #

Hey Aldi, I'm not sure what you mean by "run the example from OpenCV"? Most tutorials on this blog post assume you are executing a Python script via command line argument. I would suggest going through a more recent post, like this on on pedestrian detection to see how Python files are created and executed. Alternatively, if you're just getting started using Python and OpenCV, Practical Python and OpenCV would be a good start.

REPLY

Supra October 28, 2015 at 6:30 am #

It worked for Model B+ w/out using virtualenvwrapper by using python 3 merely.

REPLY

Adrian Rosebrock November 3, 2015 at 10:40 am #

As I mentioned in the post, using virtual environments is entirely optional. But I do highly recommend it for Python development.

REPLY

Michael November 4, 2015 at 12:24 am #

check. Thanks Maestro!

REPLY

Brian November 4, 2015 at 9:03 pm #

When I try to unzip OpenCV it gets about halfway through and then I get a write error, asking if my disk is full. This is a brand new 16gig class 10 microSD with Jessie installed. It' can't possibly be full already. I tried the wget again thinking the download was corrupt but it still gives me a write error. Any ideas?

Ah-ha! I was able to successfully unzip opencv.
A critical step that you might want to mention before running the "unzip opencv.zip" command:

Navigate to the Raspberry Pi Configuration Tool and select EXPAND FILESYSTEM

Hope that helps someone!

Adrian Rosebrock November 5, 2015 at 6:38 am #

REPLY

Thanks for the tip Brian! I (incorrectly) made the assumption that everyone had already expanded their filesystems fully.

Matt November 21, 2015 at 12:55 pm #

REPLY

Hi,

My Raspberry Pi 2 won't allow me to expand the filesystem like Brian mentioned. I used NOOBS to install Raspbian Jessie and am using an 8GB micro SD but only 1GB is available once the operating system is installed.

Any help would be much appreciated

Matt

Adrian Rosebrock November 22, 2015 at 7:18 am #

REPLY

Normally to expand the file system all you need to do is run `raspi-config`, select the first option to "Expand filesystem", and reboot. That should expand your 8GB card. Make sure you reboot after running the command. If you're *still* not getting your file system expanded, either (1) your file system has already been expanded or (2) something very odd is going on and you should consult the Raspberry Pi documentation.

Roger January 6, 2016 at 8:07 am #

REPLY

Howdy Matt

I ran into the same issue you expereined; not being able to expand the memory on SD card running NOOBS version for rsapi.
So, I decided to directly dd the rsapian os without all the oother os'es onto SD Card. I suggest this to you, too.

Just download the raspian os and copy via dd command in Linux CLI or with an additional software under windows; sorry I am not aware of a good one since I am not working with windows.
Google may help here.

Regards,
Roger

Matt January 20, 2016 at 10:58 am #

Hi Roger,

Yes I have done the same and it works now.

Thanks

Matt

Sigi November 15, 2015 at 4:54 pm #

REPLY

Hi Adrian,
thank you for this great tutorial – it worked like a charm rpi2 – jessi – python3 – opencv3

I have a script for circle detection that runs well on rpi2 – wheezy – python2 – opencv2.

but with opencv3 it crashes at the line:

” wp = cv2.HoughCircles(blurred,cv2.cv.CV_HOUGH_GRADIENT,1,75,param1=50,param2=12,minRadius=7,maxRadius=9)) ”

with this error message:

” AttributeError: ‘module’ object has no attribute ‘cv’ ”

What can I do?

Thanks in advance

Sigi

Adrian Rosebrock November 16, 2015 at 3:35 pm #

REPLY

For OpenCV 3, you’ll need to change `cv2.cv.CV_HOUGH_GRADIENT` to: `cv2.HOUGH_GRADIENT`.

Sigi Weichenrieder November 17, 2015 at 4:01 pm #

REPLY

Hi Adrian,

that worked :-)) Thank you!!!!!!!!!!!!!!!!!!!!

Sigi

Adrian Rosebrock November 18, 2015 at 6:26 am #

REPLY

Awesome, I'm glad it worked Sigi!

kk November 20, 2015 at 6:21 am #

Hi, I tried https://www.pyimagesearch.com/2014/07/14/3-ways-compare-histograms-using-opencv-python/ with rpi2 – jessi – python2.7 – opencv3.0 installation as guided in this post.

("Correlation", cv2.cv.CV_COMP_CORREL)
throws error message:
AttributeError: 'module' object has no attribute 'cv'

Is similar change required? But,
("Correlation", cv2.CV_COMP_CORREL),
throws error either:
AttributeError: 'module' object has no attribute 'CV_COMP_CORREL'

Adrian Rosebrock November 20, 2015 at 6:36 am #

Try using `cv2.HISTCMP_CORREL`.

Nichten November 16, 2015 at 6:49 pm #

REPLY

Hello Adrian,

I have successfully installed OpenCV 3 with Python 3.4 on Raspbian Jessie. thank, you for your great work :).

But I have some problem with package python-serial. How to install it in virtualenv with python 3? I type command workon cv and I'm trying to install it, but the OS return information that it is already installed. But I can't import serial in python3. I can do it with python2. What I'm doing wrong?

~Nichten

Adrian Rosebrock November 17, 2015 at 6:21 am #

REPLY

Congrats on getting OpenCV 3 installed, that's great! I personally have not used python-serial before so I am unfamiliar with it. I believe in Wheezy that you had to install and run it as root, but with Jessie you can install it and run it as as normal user — is that correct?

It sounds like you may have installed python-serial via `apt-get` which is why it's not working for you. If you

do:

```
1 $ workon cv
2 $ pip install pyserial
```

It should work for you.

Deepak January 11, 2016 at 11:25 pm #

REPLY

Hello Adrian,
I am also facing the same issue with serial module. I have tried pip install pyserial. Even then I am not able to import the serial module to cv virtual env. Outside the virtual env, serial module is working fine. Please advice.

Adrian Rosebrock January 12, 2016 at 6:26 am #

REPLY

Since I am not a pyserial/GPIO user, I'm honestly not sure about the problem. I've added this to my queue as a potential new blog post to help resolve the issue.

Eddie Mitchell November 19, 2015 at 12:42 am #

REPLY

Beautiful walk through. worked like a charm and im super excited to dive into computer vision. getting your book[s] for christmas! keep up the awesome work Adrian!

Adrian Rosebrock November 19, 2015 at 6:17 am #

REPLY

Thanks for the kind words Eddie! 🙂 I'm glad the tutorial worked for you.

kk November 20, 2015 at 8:03 am #

REPLY

Hello Adrian,
Thanks for the great work. With my installation of OpenCV 3.0 with Python 2.7 on Raspbian Jessie, I'm running your blog codes one by one. I'm really enjoying it.
But I have a problem with installing "scipy". "pip install numpy" works, but "pip install scipy" is failed. I wonder if I'm the only one who is suffering from the problem.
It would be greatly appreciated if you could help me. Thanks in advance.

Adrian Rosebrock November 20, 2015 at 8:50 am #

REPLY

Did you get an error message when installing Scipy? And which Raspberry Pi model are you using?

armin November 21, 2015 at 2:22 pm #

REPLY

Hello
Thanks for your tutorials, according to these instructions I'm trying to build and compile opencv3 on RPI2 running ubuntu mate 15.10, but i'm facing this error in compilation step:

make file:136 recipe for target 'all' failed
error 2

Adrian Rosebrock November 22, 2015 at 7:17 am #

REPLY

Hey Armin — this tutorial is dedicated to installing OpenCV on Raspbian Jessie. Theoretically the same steps can be applied to Ubuntu Mate, but I have not personally tried them on Ubuntu Mate before. If you're getting an error when running the `make` command, more times than not it's due to a missing dependency. Make sure you have ran all of the previous `apt-get` commands to pull in the required prerequisites.

Marijn December 31, 2015 at 6:11 am #

REPLY

I ran into a similar error when 'making' with make -j4 on an Pi B+. I then did 'make clean' followed by the normal 'make' command. I took another 10+ hours, but finished without errors after that. And was able to finish the guide.

Adrian Rosebrock December 31, 2015 at 6:54 am #

REPLY

If you're using a B+, then definitely only use `make` instead of `make -j4`. The B+ does not have four cores like the Pi 2 does so it makes running into dependency issues during compile a lot more likely.

Nico November 22, 2015 at 6:28 am #

REPLY

Hello Adrian

Thanks for your great walk throughs and tutorials. I successfully installed OpenCV3 and Python3 on my Pi2 with camera module and step 6 of this post returns the expected results.
When I tried to run the basic motion detection script from your article (http://www.pyimagesearch.com/2015/05/25/basic-motion-detection-and-tracking-with-python-and-opencv/) nothing happend. I created the script, opened command line, changed to the virtualenv cv and tried to run the script with this command: python basic_motion.py After hitting execute nothing happens for 2 seconds and then a new command line shows up waiting for input. No windows appear to show me the live camera view.

The 2 seconds gap is probably the time it takes to import OpenCV. I tried to run the command as sudo, I have given the script different permissions and I have changed some code. Sadly nothing helped to solve my problem. Commands as raspistill work perfectly nice so there can_t be any camera issues.
FYI I am working with the Pi's GUI and not via SSH.

I would greatly appreciate your help. Thanks in advance.

Adrian Rosebrock November 22, 2015 at 7:14 am #

REPLY

The basic motion detection script uses the `cv2.VideoCapture` function which can require additional drivers for the Pi. I recommend using this post for motion detection since it is 100% out-of-the-box workable with the Pi and the Pi camera module.

Nico November 22, 2015 at 9:47 am #

REPLY

Thanks Adrian the home surveillance tutorial works like a charm!

Adrian Rosebrock November 23, 2015 at 6:39 am #

REPLY

Awesome, I'm glad to hear it Nico! 🙂

Gary November 25, 2015 at 1:40 pm #

REPLY

Help! I just upgraded to Jessie, and something broke on the previously stable virtual environment I setup using your instructions for Python 2.7, Open CV 2.X and Wheezy.

The upgrade to Jessie seemed to go well, but I am now seeing something broken with the virtual environment. Symptom:

1. If I try and execute any python in the virtual environment I had previously, python import fails. For instance, trying to import diatomite is failing.
2. If I go into Python, and manually import datetime, it fails.
3. If I am NOT in the virtual environment, and go into Python, I can import datetime just fine.

How can I repair the virtual environment here? Do I have to go through all of the steps above, or are their shortcuts that are faster and simpler?

Thanks Adrian!!

Adrian Rosebrock November 25, 2015 at 1:50 pm #

REPLY

Hey Gary — just to clarify, did you upgrade from Wheezy to Jessie? Or did you do a fresh install of Jessie? I'm assuming the former. Unfortunately, I have never tried upgrading before, only installing on a

fresh OS. Hopefully another PyImageSearcher reader can help you out with this one!

Gary November 25, 2015 at 2:12 pm #

REPLY

I upgraded. I followed all steps here https://www.raspberrypi.org/forums/viewtopic.php?f=66&t=121880&sid=e513ef7fd4458647baf1abc6ac1ae30e which seems to be what most folks are pointing to for the best upgrade path. The upgrade seemed to go well.

One other clue for any reader….. if I load the virtual environment, and then perform step 6 above to verify all is well, I get an error from Python on the import cv2 statement:
"ImportError: numpy.core.multiarray failed to import"

Odd. If I go to ~/.virtualenvs/cv/lib/python2.7/site-packages, numpy is still there.

Makes me wonder if the bindings / sym-link is somehow trashed?

hoping to avoid having to reinstall everything above. If anyone has any experience on upgrading a previously stable Wheezy virtual environment to Jessie, please let me know.

Thanks

One more thing, if I go to check the links setup in Step 5 they are still valid and showing appropriately in the virtualenvs directory as well as when using the readlink command. Hmmmmmmm……

Adrian Rosebrock November 26, 2015 at 6:55 am #

REPLY

As you suggested, it sounds like your sym-links have been destroyed. I can't speak for all the other packages on your system, but for OpenCV you should be able to re-create them using the `ln` command like we did in this tutorial. At that point you should have OpenCV back.

I hate to say it, but you'll likely need to do a re-install. I've ran into issues where my sym-links where destroyed (on OSX, not Raspbian) and every since then I never an "upgrade" to the latest OS. If I want to upgrade, I have to commit fully and re-install from scratch. It takes awhile and it's a lot of effort, but it's the best guarantee that you're development environment will be 100% setup correctly.

Gary Lee November 26, 2015 at 12:25 pm #

A hint to others…. I reinstalled numpy and it seemed to get everything back in order. Weird. I also (just to be safe) decided to start fresh with a new install of everything. Thanks for a great tutorial for getting this done. If others decide to follow the upgrade path, reinstalling numpy may be the solution.

Rob Jones November 28, 2015 at 5:23 pm #

REPLY

Adrian – This all worked *great* and I installed picamera[array] in the cv virtual env – I am able to read image frames from the camera, process them and output them

I'm using python 2.7

Now I want to bring RPi.GPIO into the mix … that works fine by itself outside the cv virtualenv as Jessie includes RPi.GPIO v0.6.0a3 by default – this gives you access to the GPIO pins without having to be root.

But the cv virtualenv does not see this… and the latest version of RPi.GPIO available by pip is 0.5.11 which won't cut it…

How can I get v0.6.0a3 installed in the virtualenv ?
Compile from source inside that env ?

I'm not totally on top of pip / virtualenvs so I'm reluctant to just try it

I'm hoping that you have already dealt with this

thanks !

–Rob

I figured out the answer to my own question :^)

You just manually copy over the system RPi package files into the virtual environment like this:

```
$ cp -r /usr/lib/python2.7/dist-packages/RPi ~/.virtualenvs/cv/lib/python2.7/site-packages/
```

Adrian Rosebrock November 29, 2015 at 7:10 am #

REPLY

Hey Rob — for future reference you can use `pip` and install it into your virtual environment. This is the preferred method rather than copying files directly:

```
$ pip install rpi.gpio
```

Philip November 29, 2015 at 10:05 am #

REPLY

Hi Adrian,

Thank you for your excellent tutorial. Everything worked perfectly except for a few errors installing numpy. Uninstalling and reinstalling fixed that. I also ran out of space building OpenCV as I was using Noobs. Formatting the SD card and just using the Jessie image file freed up enough space for the install.
I'm just wondering if you have had any experience with drivers for webcams on the Pi? When I use any of my webcams and try to get/set attributes such as CV2.CAP_PROP_FPS, I get the following error:
VIDEOIO ERROR: V4L: Property (16) not supported by device
I've got the libv4l library installed and the WITH_LIBV4L boolean is set to ON in the opencv build configuration. I've googled but I haven't come up with anything so far. Weird! :-/

Adrian Rosebrock November 30, 2015 at 6:35 am #

REPLY

Hey Philip — I'm sorry to say that I don't have much experience using webcam drivers for the Raspberry Pi. I either use the actual Pi camera board or a Logitech 920C which works out of the box. Be sure to check this page of compatible USB webcams for the Raspberry Pi to see if there are any known issues with it.

SK November 29, 2015 at 1:46 pm #

REPLY

Hi Adrian,

There is excellent content on your website – thanks you so much!

I have a request – will you be able to prepare a tutorial to install OpenCV with QT, VTK, and these additional performance libraries (CUDA, etc) for Raspberry Pi (all that it can support) and linux computers? Thank you for all you do, I am considering buying one of your packages.

SK

Adrian Rosebrock November 30, 2015 at 6:33 am #

REPLY

It's something that I can certainly look into.

rap November 30, 2015 at 10:19 am #

REPLY

Thanks for the instructions! However, could you please tell if this will work for C++ as well as I wonder about the virtual environment part? If I install for Python 3 according to these instructions is the opencv library accessible for C++ from the default environment too or what should be done otherwise. Thanks!

Adrian Rosebrock November 30, 2015 at 7:07 pm #

REPLY

Once you run `make install`, you have OpenCV fully installed (as far as C++ is considered). The Python virtual environments have no impact on the C++ installation.

rap November 30, 2015 at 7:27 pm #

REPLY

Thanks!

Mau December 6, 2015 at 11:14 am #

REPLY

Hi Adrian, might I be able to compiling with raspberry b+ 512?
Thanks for your tutorials.
Mau

Adrian Rosebrock December 6, 2015 at 11:18 am #

REPLY

This tutorial should work with the Raspberry Pi B+; however, I *highly encourage you* to use a Pi 2 for computer vision, especially if you want to do any type of video processing.

Bob December 8, 2015 at 7:30 pm #

REPLY

Is it bad if I accidentally ran “make clean” again after the long “make” process but canceled it almost immediately?

Adrian Rosebrock December 9, 2015 at 6:53 am #

REPLY

Technically no, you can just run `make clean` again and let it finish cleaning up. However, you *will* need to run `make` again. Just to be safe, I would suggest deleting your `build` directory, re-creating it, and then running `cmake` just to make sure you are starting fresh.

Birkan December 10, 2015 at 12:33 pm #

REPLY

can we delete opencv-3.0.0 and opencv_contrib-3.0.0 folders after succesfull installation?

Adrian Rosebrock December 10, 2015 at 2:22 pm #

REPLY

Yes, once you have OpenCV installed you can safely delete the `opencv-3.0.0` and `opencv_contrib-3.0.0` directories.

Birkan December 12, 2015 at 4:44 pm #

REPLY

Thanks..

mohamed abo amer December 12, 2015 at 6:50 pm #

REPLY

it says no enough space when downloading open cv
(wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.0.0.zip)

Adrian Rosebrock December 13, 2015 at 7:31 am #

REPLY

If you're getting an error that there is not enough space to download the .zip file, then your micro-SD card is filled. Either delete unneeded files to make room for OpenCV or make sure you have already ran `raspi-config` to ensure the entire micro-SD card is available for usage.

Brian December 12, 2015 at 10:09 pm #

REPLY

Just to clarify, does this installation (OpenCV 3.0.0, Jessie, RPI2) run the OpenCV scripts in multi-core mode?

Adrian Rosebrock December 13, 2015 at 7:29 am #

REPLY

No, it does not directly leverage multiple cores. You'll need to write your own scripts to distribute processes to multiple cores/threads if.

Florian December 21, 2015 at 9:31 am #

REPLY

Hello Adrian,

thanks for the great tutorial, but i have problems building with GTK+ support. I installed all the librarys and dependencies, but everytime i run the cmake command it remports back with "GTK+ No".

Do you have any idea?

Thanks Florian

Adrian Rosebrock December 21, 2015 at 5:35 pm #

REPLY

Hey Florian — I have never ran into this issue, but it looks like this question on the OpenCV forums might be able to help.

deepak December 22, 2015 at 11:52 am #

REPLY

hello Adrian thanks for your great tutorial i am compiling open cv on my pi2 but i have got a question how to compile opencv-contrib can to tell me the steps please and how to install PIL once again thankyou

Adrian Rosebrock December 23, 2015 at 6:43 am #

REPLY

Hey Deepak, this tutorial already details how to compile and install opencv-contrib. Please see **Step 2** where the OpenCV source code is downloaded and **Step 4** where OpenCV is actually compiled.

You can also install OpenCV into your virtual environment by using `pip`:

```
 workon cv
$ pip install PIL
```

Mahmoud Tolba December 23, 2015 at 11:11 am #

REPLY

Thanks for your great effort.
May I ask you a question please?

In case of using Raspberry pi type b, Should I change this command (make -j4) to be (make -j1) ?

Adrian Rosebrock December 23, 2015 at 1:07 pm #

REPLY

If you're using the B, then just use `make` and leave off the `-j` switch.

David Kadouch December 25, 2015 at 3:43 pm #

REPLY

I'm so grateful to the deep level of detail of your tutorials. It works like a charm.
For the sport I tried first to install opencv 3.1 that was just released a few days ago. Alas it fails at the make stage with a 'no targets' error message. In fact no Makefile was created after 'cmake'. I must say that it's a bit frustrating that they break the install process for minor versions. It makes upgrading a real pain. Any insights here?

Adrian Rosebrock December 26, 2015 at 8:55 am #

REPLY

Hey David, please see my reply to Nelson Chen above.

Nelson Chen December 25, 2015 at 7:34 pm #

REPLY

Thanks for great effort.

I have a question for you. My opencv is v3.1.0, and when I try to run cmake, I got an error. I have tried several times and found that I had to remove the INSTALL_C_EXAMPLES thing from the command line, or it led to an error. What do you think of it?

Adrian Rosebrock December 26, 2015 at 8:55 am #

REPLY

I actually ran into this error myself when trying to compile OpenCV 3.1.0. Try removing `-D INSTALL_C_EXAMPLES=ON -D INSTALL_PYTHON_EXAMPLES=ON -D BUILD_EXAMPLES=ON` from the `cmake` command and it should generate a working Makefile.

Tom December 26, 2015 at 3:31 pm #

REPLY

It seems that the issue is in the c examples, I had to specifically add -D INSTALL_C_EXAMPLES=OFF before I got it to work, the rest of what you mentioned can stay in the cmake command without issues.

Thanks for the great tutorial

Adrian Rosebrock December 27, 2015 at 7:54 am #

REPLY

Thanks for the clarification regarding the cmake options!

Mario January 1, 2016 at 7:16 am #

REPLY

Thanks, I ran into the same problem.

Adrian Rosebrock January 1, 2016 at 7:23 am #

REPLY

You can fix the issue by removing the `-D INSTALL_C_EXAMPLES=ON` from the CMake command.

Syed Tariq December 29, 2015 at 12:22 am #

REPLY

Excellent Tutorial. I followed all the steps and tested it by using a simple program to display video from a Sony PS Eye. The only issue I am facing is that the ‘import cv2’ is only recognized when my python program is in the site-packages subdirectory. I used the instructions in ‘Installing OpenCV 3.0 for both Python 2.7 and Python 3+ on your Raspberry Pi 2’ but I am using Raspbian Jessie OS. I did compare it to the instructions on the ‘How to install OpenCV 3 on Raspbian Jessie’ and did not see anything that would affect the installation. Any help would be appreciated.

Adrian Rosebrock December 29, 2015 at 7:57 am #

REPLY

Are you using the `workon cv` command to drop into the Python virtual environment before executing your script?

Syed Tariq December 29, 2015 at 4:36 pm #

REPLY

I am using 'workon cv'. I can see '(cv) in the command line. I don't totally understand virtual environment. I will read up on it and try debugging. It appears that the link cv2.so is not global even though it has attributes lrwxrwxrwx. If I create a symbolic link cv2.so in the directory from which I run the python script it seems to work.

Thank you for your help.

Adrian Rosebrock December 30, 2015 at 7:00 am #

REPLY

If it works within your current working directory, then the sym-link inside the `site-packages` of the `cv` virtual environment is likely not right. I would delete the sym-link and then re-create it.

Syed Tariq December 31, 2015 at 2:48 pm #

Adrian – Thank you. It is all working right now. It was a great tutorial and it helped me immensely.

Adrian Rosebrock January 1, 2016 at 7:29 am #

Awesome, I'm glad you were able to resolve the issue 🙂

Fra February 1, 2016 at 8:52 am #

REPLY

Hi Syed/Adrian, I have the same issue cv2.so works within my current working directory, so even for me It appears that the link cv2.so is not global.

How did you fix it?

I followed video and tutorial in any details…

When I run my test.py in ~ I get "ImportError: No module named cv2" otherwise in site-packages it is fine.

Bill Kearson January 3, 2016 at 10:09 am #

REPLY

Great job,

With your detailed instructions and the comments I have OpenCV3.1 and Python 3 working, I had to remove the C examples, INSTALL_C_EXAMPLES=OFF, from the cmake command and ended up having to use one

core to avoid errors during the make.

In the end everything is working.

Using a 8G card I was almost out of space but I saved an image before and after removing the downloaded files and folders. Left me with about %55 free space. Not bad.
A good time was had by all over two days. I did start over three times making noob mistakes: Expand the file system running raspi-config on first run.
Now I have my pi running headless, wifi connection, ssh primary connection and vnc available with manual startup, if needed.

Thank you

Adrian Rosebrock January 3, 2016 at 12:45 pm #

REPLY

Congrats on getting your setup working Bill! And I'm in the process of updating the OpenCV 3 install tutorials to mention removing the `INSTALL_C_EXAMPLES=OFF` flag. It looks like you beat me to it! 🙂

Sebastian January 3, 2016 at 10:38 pm #

REPLY

Hi Adrian

First I wanna thank you for the tutorials, I tried for a long time install OpenCV in a Raspberry, but I can't do it. I followed your instructions, and I don't have any error during the process, but the problem is when I tried to import the library in Python, I have the next error: "ImportError: dynamic module does not define init function (initcv2)", do you have any idea to solve that?. Thanks.

Adrian Rosebrock January 4, 2016 at 6:48 am #

REPLY

I think one of two things happened. Either you compiled OpenCV with Python 2.7 support and then tried to import OpenCV with Python 3, or vice versa. Make sure you are in the `cv` virtual environment prior to importing OpenCV. If that still doesn't work, go back to the CMake command and make sure you are compiling OpenCV for your intended version of Python.

Sebastian January 5, 2016 at 9:32 am #

REPLY

Hi Adrian

Yes, it was my problem, thank you so much. as we say in my country: "you deserve the sky and more", thanks!!!

Adrian Rosebrock January 5, 2016 at 1:53 pm #

REPLY

Nice, congrats on resolving the issue 🙂

Tim Farnworth January 4, 2016 at 10:50 pm #

REPLY

Hi Adrian,

Thank you for these tutorials man. They are great!

I installed OpenCV last night and paused the installation just after the make -j4 command. I resumed this morning. Other then adding `-D WITH_OPENMP=ON` and set `-D INSTALL_C_EXAMPLES=OFF` to the cmake install, I have followed the tutorial closely. But for some reason I am having issues.

`cv2.imshow()` and cv.ShowImage() dont open any image windows. I'm getting frustrated because I cant see where my code is wrong.

Here is a bare bones version of my code:

```
1 import cv2
2
3 # Open Show Image
4 I = cv2.imread('MyPic.JPG')
5 print type(I)
6 A = cv2.imshow("Window",I);
7 print "Image shown"
8 cv2.waitKey(1)
```

The code terminates in the cv2.imshow() function and does not reach the print "Image shown" line.

Do you have any ideas what could be wrong? Cheers

Adrian Rosebrock January 5, 2016 at 6:46 am #

REPLY

It's very strange that the "Image shown" text is never printed to your screen. I would point that unless you are using a video stream, you likely don't want to use `cv2.waitKey(1)`. This will only wait 1ms before automatically closing the window. You should use `cv2.waitKey(0)` to wait indefinitely for a keypress.

Since you deviated slightly from the tutorial, I would suggest doing two things as a sanity check:

1. Blow away the install and start from scratch. Specifically, pausing during `make -j4` might have introduced a hard to track down dependency issue. Allow OpenCV to compile without pausing the process.
2. Secondly, re-compile with `-D WITH_OPENMP=OFF`. I've never encountered an issue with OpenMP before, but since it deviates from the tutorial, I think it's good to perform a sanity check.

Nicolas Roux January 6, 2016 at 2:28 pm #

REPLY

It's a great guide, I think I already used one of your tutos in the past so a thank you was long overdue. Here it is:

Thank you, keep up the good work.

Adrian Rosebrock January 6, 2016 at 2:31 pm #

REPLY

Thanks for the kind words Nicolas 🙂

Malef January 7, 2016 at 10:43 am #

REPLY

i try to compile opencv 3.1.0
but it stalls after 5 mins [21%] on opencl/perf_arithm.cpp
6%CPU and 59% MEM
Using Swap Total 2097148 6800 used
what can i do waiting longer or is there something wrong?

is compliling with OpenCL makes sence ? is the GPU is used?
Thanks for help

Adrian Rosebrock January 7, 2016 at 12:39 pm #

REPLY

That's strange that the compilation has stalled. Are you trying to compile with OpenCL support? If so, I would suggest disabling and trying again.

Also, try using compiling using only a single core with: `make`

This will avoid any potential dependency/race conditions with the compile.

Malef January 8, 2016 at 5:55 am #

REPLY

i switched of the opencl now the same thing at 82% building python module cv2
kswapd uses most time the cpu() and the cc1plus process
and then it stalls
maybe a swapping problem and then a crash?

Adrian Rosebrock January 8, 2016 at 6:28 am #

REPLY

How long into the compile? Also, which Pi hardware are you using?

Malef January 11, 2016 at 5:44 am #

Raspberry B first version 256MB RAM 128MB Split
Swap on SD with 3 times Ram
I try it now with 16MB memory split for GPU

Adrian Rosebrock January 11, 2016 at 6:48 am #

The RAM definitely sounds like the issue. These OpenCV + Raspberry Pi tutorials are intended for either the B+ or the Pi 2. I have never tried with the B, simply because it has too little RAM and isn't the most suitable for computer vision and image processing. If at all possible, you should try to get a Pi 2.

Paul Maher January 10, 2016 at 6:11 am #

REPLY

Thanks for posting the detailed instructions which have been invaluable to me. I have a question though about the virtual environments. I've not used them before so I managed to tie myself in a knot.

If I switch to the CV virtual env as in your instructions everything works fine… but if I try to import the GPIO library it says its not found.I also cant use 'sudo' to run my OpenCV python programs (which I will need to do as I want to access use the GPIO ports as well as the camera). So at the moment I can *either* use the CV virtual env and get OpenCV OR not use the virtual env and get access to sudo and GPIO. Not sure how to dig myself out!

Any advice gratefully received!

Adrian Rosebrock January 10, 2016 at 7:40 am #

REPLY

I've actually addressed this problem a few times on the PyImageSearch blog. First, you should read my reply to Rob James above. Secondly, do a ctrl+f and search this page for GPIO (especially in the comments section). You'll notice that the solution is to launch a root shell, create a virtual environment for the root user, install OpenCV into it, and execute your script.

The important point is that you need to actually *launch* a root shell rather than just using sudo to call your script.

Finally, you could try ignoring the virtual environment entirely and ensure both OpenCV and NumPy are installed into your normal Python `site-packages` directory. This will also solve the problem.

Paul Maher January 11, 2016 at 3:30 am #

REPLY

Adrian, thanks for the prompt reply. I will try omitting the virtual env steps and see how I get on. Thanks again for the reply.

manish January 10, 2016 at 7:44 am #

REPLY

Sir,
how to import cv2 on python shell …

I followed your all steps and it goes correct..
but I didn't get on shell as it shows

no module name cv2

I m using
Raspi-2,model B
with NOOBS

Adrian Rosebrock January 10, 2016 at 8:18 am #

REPLY

Please see the "Troubleshooting" section at the bottom of this post. I would also encourage you to watch the installation video at the top of this post and follow along with the exact commands that I perform. This will ensure that OpenCV is properly configured and running on your system.

manish January 10, 2016 at 8:25 am #

REPLY

Sir I followed your video and at the terminal it is going right..
but it didn't work on python shell….
I m a beginner in Raspberry pi …
Can you take me out of this??

it was all right but not no shell

Opencv is installed on my pi
I checked it through terminal as you mentioned in last step..

Adrian Rosebrock January 11, 2016 at 6:47 am #

REPLY

Hey Manish, so just to clarify OpenCV is indeed installed on your Pi?

If it's not, then I would suggest going through the "Troubleshooting" section of this post. I'm also not sure what you mean by it's working in the terminal, but not in the Python shell.

Arash January 10, 2016 at 4:05 pm #

REPLY

Thank you for the tutorial.
As mentioned in the update I got the make error after (C_EXAMPLES=ON) so I removed the build folder and redid the cmake with C_EXAMPLES=OFF however I am still getting "No targets specified and no makefile found. Stop."

Adrian Rosebrock January 11, 2016 at 6:42 am #

REPLY

After you removed the `build` directory, you need to re-create it and enter it *before* running CMake again:

```
$ rm -rf build
$ mkdir build
$ cd build
```

From there you should run the CMake command.

manish January 12, 2016 at 11:08 am #

REPLY

sorry to have a late response sir.
I got it .and solved the problem .
thanks for your support and suggestions..

and thanks for this tutorial.it is really helpful.

Kewin January 14, 2016 at 11:50 am #

REPLY

I have 8 gb sd card with raspian jessie. When I install raspian on it, the folder manager of rasperry pi shows the free space about 700mb. Is that normal. If 8gb is not enough I will buy a 32gb sd card. Because I want to install opencv and python and the disk space is not enough. Please help me
thanks

Adrian Rosebrock January 16, 2016 at 9:40 am #

REPLY

700MB is very much borderline. Make sure you have ran `raspi-config` and expanded your Raspberry Pi filesystem to use the entire micro-SD card. Otherwise, if you have already expanded your filesystem, I would suggest either (1) deleting files to make extra room or (2) getting a large micro-SD card.

Hilman January 15, 2016 at 6:21 am #

REPLY

Hey Adrian.
I want to ask something. Is a 4GB microSD card big enough to install Raspbian Jessie, OpenCV3, and all the things you've wrote above? What if I want to make project on the Raspberry Pi (by using the Raspberry Pi camera and your tutorial of course), will it still be big enough?

Adrian Rosebrock January 16, 2016 at 9:34 am #

REPLY

Realistically, a 4GB card is not large enough — I would suggest using an 8GB card.

Joseph January 16, 2016 at 3:03 pm #

REPLY

I followed every step (doing Python3) but when I got to Step #5: Finishing the install, my python3.4 directory is empty. python2.7 has a bunch of files, but not cv2.so. I looked back at all the commands I entered and everything looks right. Not sure what went wrong or where I could backtrack to.

Adrian Rosebrock January 17, 2016 at 5:29 pm #

REPLY

Be sure to take a look at the “Troubleshooting” section of this post. Be sure to take a look at the `build/lib` directory as well.

mai January 20, 2016 at 7:26 am #

REPLY

went though make with no error.
However, make install gives following errors:
How do I post a screen shot?

Adrian Rosebrock January 20, 2016 at 1:42 pm #

REPLY

I would suggest uploading an image to imgur and then posting the link.

Vishwa January 24, 2016 at 3:51 am #

REPLY

Followed the guide till the end, I can import cv2 in terminal but not in IDLE, none of the comments helped. There is a cv2.so file in site-packages. I don’t understand what you mean by running it in the cv virtual environment. Help

Adrian Rosebrock January 25, 2016 at 4:12 pm #

REPLY

To access the Python virtual environment, you need to use:

```
$ workon cv
```

And this will drop you down into the `cv` virtual environment. You can then use the *terminal* version of IDLE. The GUI version doesn’t properly work with the Python virtual environment.

zevero January 29, 2016 at 9:57 am #

REPLY

If you end up not having enough space like me on raspian (on 8GB) try to free some space:

500MB: sudo apt-get remove –purge wolfram-engine
100MB: sudo apt-get remove –purge scratch
xxxMB sudo apt-get clean autoclean

Adrian Rosebrock January 29, 2016 at 12:53 pm #

REPLY

Thanks for sharing the tip!

mikel January 31, 2016 at 1:39 am #

REPLY

thank you so much!

Adrian Rosebrock January 31, 2016 at 8:55 am #

REPLY

No problem Mikel, I'm happy the tutorial worked for you.

Fra January 31, 2016 at 5:01 am #

REPLY

Hi Adrian,
I followed your instructions until step 4. Once I run CMake I checked Python 2 and 3 section.
Unfortunately for me both have "packages path" set to "lib/python3.4/site-packages"

What can be wrong?

Adrian Rosebrock January 31, 2016 at 8:57 am #

REPLY

To start, make sure you are in the `cv` virtual environment prior executing CMake:

```
$ workon cv
```

Otherwise, assuming you are trying to build Python 3 bindings, it should still be okay. You can always copy/sym-link into the `site-packages` of the virtual environment after compiling.

PetriK February 2, 2016 at 7:16 am #

REPLY

excellent instructions, suggest also adding

sudo apt-get purge wolfram-engine

to release some space on the 8Gb sd car, without removing there is not enough space to install opencv.

PetriK February 2, 2016 at 3:39 pm #

REPLY

Another suggestion for installation – just learned that when mounting usb disc as /build directory the filesystem must be ext4. If filesystem is FAT then libraries can not be build on due to user rights error. Needed to have an external disc as running short of memory during make even after removing wolfram-engine.

Looks like the /build directory needs around 1.5gigs free space for the make being successful. So altogether 3gigs free space on sdcard may not be enough for the full opencv installation, so needed an external usb drive for /build.

Hilman February 3, 2016 at 7:20 am #

REPLY

Hey Adrian, I have a question.
When I entered this command:

“wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.0.0.zip” (I changed the 3.0.0 to 3.1.0, that is the latest one isn’t?)

I get the “ERROR 404: Not Found”. I can’t even go to the link in the command manually using my laptop’s browser. How can I fix this?

Thanks in advance.

Adrian Rosebrock February 4, 2016 at 9:19 am #

REPLY

That’s very strange that the .zip is 404’ing. It’s downloading perfect on my machines. Try cloning down the entire OpenCV repository and then checking out the 3.0.0 version:

```
1 $ git clone https://github.com/Itseez/opencv
2 $ cd opencv
3 $ git checkout 3.0.0
```

Hilman February 6, 2016 at 1:41 am #

REPLY

Already figured it out. It was DNS issue…

greg February 5, 2016 at 2:36 am #

REPLY

Uhh I’ve been using linux for about 15 years and I can’t get this to compile, I guess I’m retarded. Gave up after 3-4 hours of messing around.

http://pastebin.com/nSNjCggH

Adrian Rosebrock February 5, 2016 at 9:19 am #

REPLY

Which version were you trying to compile for? Python 2.7 or Python 3? Also, is that your full output from CMake? Normally there is a longer trace.

Trackbacks/Pingbacks

Installing OpenCV 3.0 for both Python 2.7 and Python 3+ on your Raspberry Pi 2 - PyImageSearch - October 26, 2015
[…] How to install OpenCV 3.0 on Raspbian Jessie. […]
Installing OpenCV on your Raspberry Pi Zero - PyImageSearch - December 15, 2015
[…] Since I've covered how to install OpenCV on the Raspberry Pi in multiple, previous blog posts, I'll keep this post on the shorter side and detail only the relevant commands necessary to get OpenCV up and running. For a more thorough discussion on how to install OpenCV 3 on your Pi (along with a 22-minute video installation guide), please refer to this post. […]

Leave a Reply

Name (required)

Email (will not be published) (required)

Website

SUBMIT COMMENT

Resource Guide (it's totally free).

Click the button below to get my **free 11-page Image Search Engine Resource Guide PDF**. Uncover **exclusive techniques** that I don't publish on this blog and start building image search engines of your own.

Download for Free!

You can detect faces in images & video.

Are you interested in **detecting faces in images & video?** But **tired of Googling for tutorials** that *never work?* Then let me help! I guarantee that my new book will turn you into a **face detection ninja** by the end of this weekend. Click here to give it a shot yourself.

CLICK HERE TO MASTER FACE DETECTION

PyImageSearch Gurus: Now enrolling new members every 2-4 weeks.

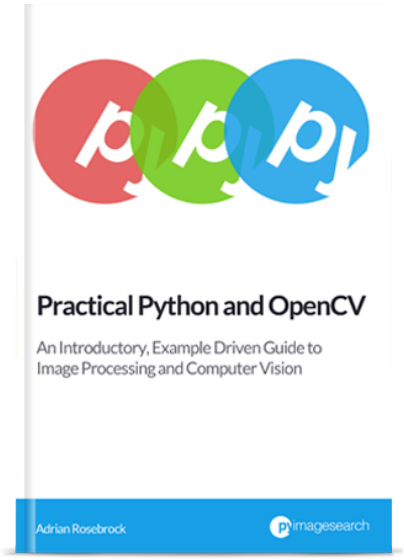
PyImageSearch Gurus enrolls new members every 2 to 4 weeks. Seats inside the course are limited, so I suggest you ***claim your spot in line*** when the doors open again and the next enrollment session starts.

CLICK HERE TO CLAIM YOUR SPOT!

Hello! I'm Adrian Rosebrock.

I'm an entrepreneur and Ph.D who has launched two successful image search engines, ID My Pill and Chic Engine. I'm here to share my tips, tricks, and hacks I've learned along the way.

Learn computer vision in a single weekend.



Want to learn computer vision & OpenCV? I can teach you in a **single weekend**. I know. It sounds crazy, but it's no joke. My new book is your **guaranteed, quick-start guide** to becoming an OpenCV Ninja. So why not give it a try? Click here to become a computer vision ninja.

CLICK HERE TO BECOME AN OPENCV NINJA

Subscribe via RSS

Never miss a post! Subscribe to the PyImageSearch RSS Feed and keep up to date with my image search engine tutorials, tips, and tricks

POPULAR

Install OpenCV and Python on your Raspberry Pi 2 and B+
FEBRUARY 23, 2015

Home surveillance and motion detection with the Raspberry Pi, Python, OpenCV, and Dropbox
JUNE 1, 2015

Accessing the Raspberry Pi Camera with OpenCV and Python
MARCH 30, 2015

Install OpenCV 3.0 and Python 2.7+ on Ubuntu
JUNE 22, 2015

Install OpenCV 3.0 and Python 2.7+ on OSX
JUNE 15, 2015

Installing OpenCV 3.0 for both Python 2.7 and Python 3+ on your Raspberry Pi 2
JULY 27, 2015

How to install OpenCV 3 on Raspbian Jessie
OCTOBER 26, 2015

Search

Search...

Find me on **Twitter**, **Facebook**, **Google+**, and **LinkedIn**.