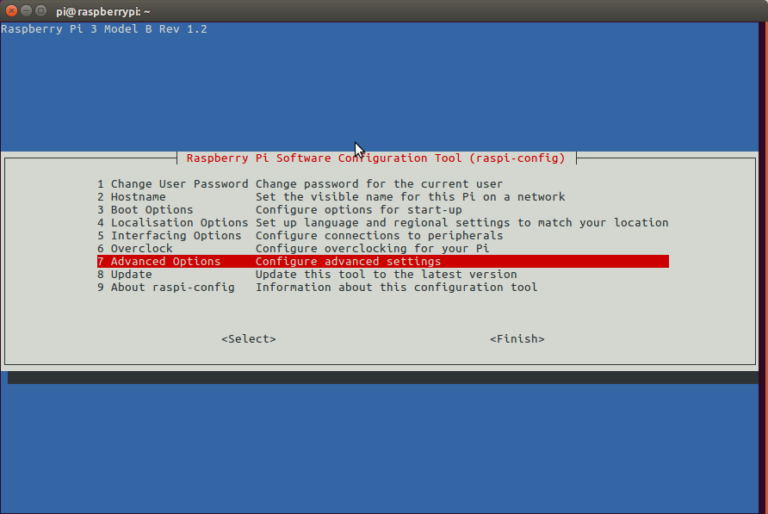
# **Raspbian Stretch: Install OpenCV 3 + Python on your Raspberry Pi**

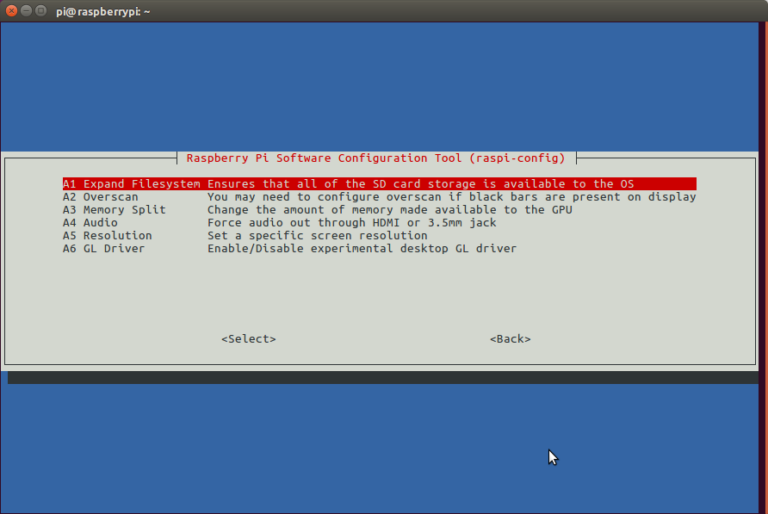
## **The quick start video tutorial**

[**https://youtu.be/j6RD3X94rEA**](https://youtu.be/j6RD3X94rEA)

### **Step #1: Expand filesystem**

sudo raspi-config

Followed by selecting “Expand filesystem”:



Once prompted, you should select the first option, “**A1. Expand File System”**, **hit Enter** on your keyboard, arrow down to the “**<Finish>”** button, and then reboot your Pi — you may be prompted to reboot, but if you aren’t you can execute:

**sudo reboot**

after reboot

If you are using an 8GB card you may be using close to 50% of the available space, so one simple thing to do is to delete both LibreOffice and Wolfram engine to free up some space on your Pi:

**$ sudo apt-get purge wolfram-engine**

**$ sudo apt-get purge libreoffice\***

**$ sudo apt-get clean**

**$ sudo apt-get autoremove**

### Step #2: Install dependencies

[This isn’t the first time I’ve discussed how to install OpenCV on the Raspberry Pi](https://www.pyimagesearch.com/opencv-tutorials-resources-guides/), so I’ll keep these instructions on the brief side, allowing you to work through the installation process: I’ve also included the **amount of time it takes to execute each command** (some depend on your Internet speed) so you can plan your OpenCV + Raspberry Pi 3 install accordingly (OpenCV itself takes approximately **4 hours to compile** — more on this later).

**sudo apt-get** **update** **&&** **sudo apt-get** **upgrade**

We then need to install some developer tools, including [CMake](https://cmake.org/), which helps us configure the OpenCV build process:

**sudo apt-get** **install build-essential cmake pkg-config**

Next, we need to install some image I/O packages that allow us to load various image file formats from disk. Examples of such file formats include JPEG, PNG, TIFF, etc.:

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

Just as we need image I/O packages, we also need video I/O packages. These libraries allow us to read various video file formats from disk as well as work directly with video streams:

**$ sudo apt-get install libavcodec-dev libavformat-dev libswscale-dev libv4l-dev**

**$ sudo apt-get install libxvidcore-dev libx264-dev**

The OpenCV library comes with a sub-module named **highgui** which is used to display images to our screen and build basic GUIs. In order to compile the **highgui** module, we need to install the GTK development library:

**$** **sudo apt-get** **install libgtk2.0-dev libgtk-3-dev**

Many operations inside of OpenCV (namely matrix operations) can be optimized further by installing a few extra dependencies:

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

These optimization libraries are especially important for resource constrained devices such as the Raspberry Pi.

Lastly, let’s install both the Python 2.7 and Python 3 header files so we can compile OpenCV with Python bindings:

**sudo apt-get** **install python2.7-dev python3-dev**

### Step #3: Download the OpenCV source code

**$ cd ~**

**$ wget -O opencv.zip https://github.com/Itseez/opencv/archive/3.3.0.zip**

**$ unzip opencv.zip**

We’ll want the full install of OpenCV 3 ([to have access to features such as SIFT and SURF](https://www.pyimagesearch.com/2015/07/16/where-did-sift-and-surf-go-in-opencv-3/), for instance), so we also need to grab the [opencv\_contrib](https://github.com/itseez/opencv_contrib) repository as wel

**wget -O opencv\_contrib.zip https://github.com/Itseez/opencv\_contrib/archive/3.3.0.zip**

**unzip opencv\_contrib.zip**

You might need to expand the command above using the “<=>” button during your copy and paste. The **.zip** in the **3.3.0.zip** may appear to be cutoff in some browsers. The full URL of the OpenCV 3.3.0 archive is:

<https://github.com/Itseez/opencv_contrib/archive/3.3.0.zip>

**Note:** Make sure your **opencv** and **opencv\_contrib** versions are the same (in this case,**3.3.0**). If the versions numbers do not match up, then you’ll likely run into either compile-time or runtime errors.

### Step #4: Python 2.7 or Python 3?

Before we can start compiling OpenCV on our Raspberry Pi 3, we first need to install **pip** , a Python package manager:

**$ wget https://bootstrap.pypa.io/get-pip.py**

**$ sudo python get-pip.py**

**$ sudo python3 get-pip.py**

You may get a message that pip is already up to date when issuing these commands, but it is best not to skip this step.

If you’re a longtime PyImageSearch reader, then you’ll know that I’m a huge fan of both [virtualenv](https://virtualenv.pypa.io/en/latest/) and [virtualenvwrapper](https://virtualenvwrapper.readthedocs.org/en/latest/). Installing these packages is not a requirement and you can absolutely get OpenCV installed without them, but that

said, **I highly recommend you install them** as other existing PyImageSearch tutorials (as well as future tutorials) also leverage Python virtual environments. I’ll also be assuming that you have both **virtualenv** and**virtualenvwrapper** installed throughout the remainder of this guide.

### Installing NumPy on your Raspberry Pi

**pip install numpy**

### Step #5: Compile and Install OpenCV

**$ cd ~/opencv-3.3.0/**

**$ mkdir build**

**$ cd build**

**$ cmake -D CMAKE\_BUILD\_TYPE=RELEASE \**

**-D CMAKE\_INSTALL\_PREFIX=/usr/local \**

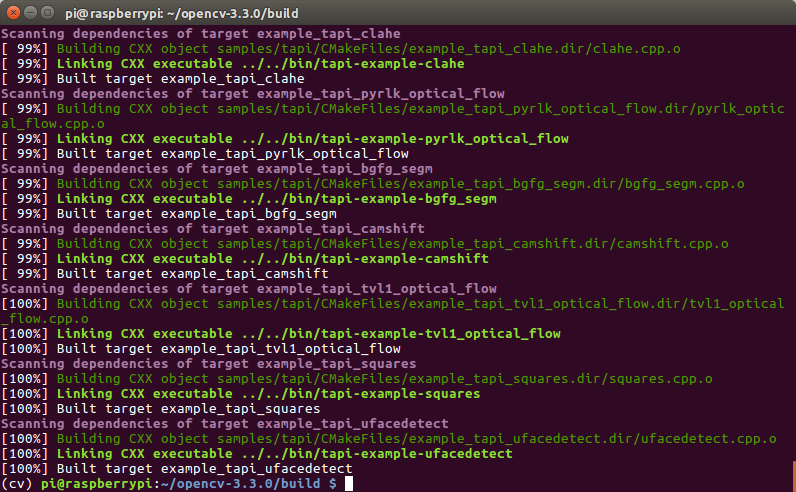
**-D INSTALL\_PYTHON\_EXAMPLES=ON \**

**-D OPENCV\_EXTRA\_MODULES\_PATH=~/opencv\_contrib-3.3.0/modules \**

**-D BUILD\_EXAMPLES=ON ..**

**Note:** It is possible to burn out the Raspberry Pi microSD card because flash memory has a limited number of writes until the card won’t work. It is **highly recommended** that you change this setting back to the default when you are done compiling and testing the install (see below). To read more about swap sizes corrupting memory, see [this page.](https://www.bitpi.co/2015/02/11/how-to-change-raspberry-pis-swapfile-size-on-rasbian/)

**make** **-j4**



**$ sudo make install**

**$ sudo ldconfig**

### Step #7: Testing your OpenCV 3 install

