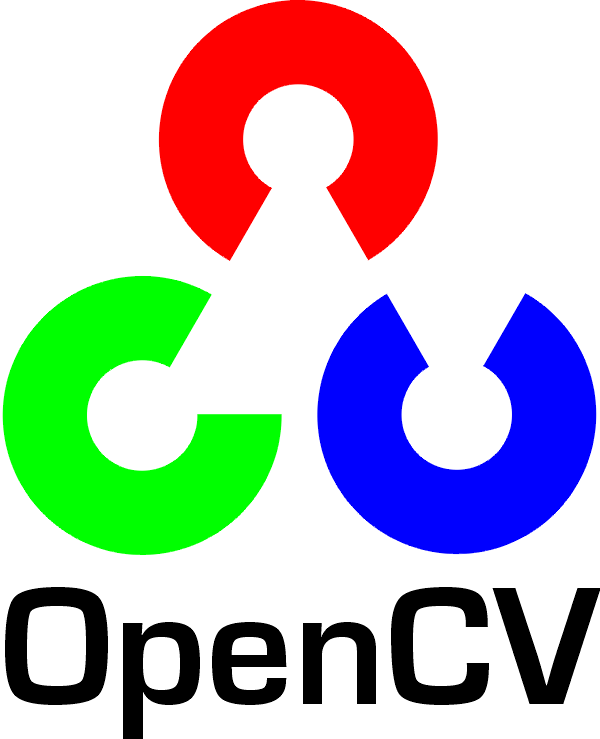
[](https://tutorials-raspberrypi.com/wp-content/uploads/2015/10/opencv-logo2.png)

**Installing OpenCV on the Raspberry Pi**

From <https://tutorials-raspberrypi.com/installing-opencv-on-the-raspberry-pi/>

For instructions for the Raspberry Pi 3, please go to page 4.

Anyone who has dealt with image processing in relation to the Raspberry Pi will sooner or later come across the[OpenCV](https://opencv.org/) library. It provides many very useful features such as face recognition, the creation of depth maps (stereo vision, optical flow), text recognition or even for machine learning. In addition, OpenCV (Open Source Computer Vision) can be integrated into both its C ++ files and its Python scripts.  
Especially in terms of feature recognition in images taken by the Raspberry Pi, OpenCV is very helpful.

This advanced tutorial shows how to install OpenCV on the Raspberry Pi and how to integrate it into Python.

For the time being, I would like to recommend everyone a [Raspberry Pi Model B](https://www.amazon.com/s/ref=nb_sb_noss_2?url=search-alias%3Daps&field-keywords=Raspberry+Pi+Model+B&tag=754u-20) to use, because this is a lot more powerful than its predecessor. Even on a Raspberry Pi B + compiling takes about 6 times as long, with a Pi 2 in total “only” takes about an hour.

First of all, we update the package lists:

sudo apt-get update && sudo apt-get upgrade && sudo rpi-update

A reboot is necessary if it has been updated.

sudo reboot

Then you can install all the important tools and libraries needed for OpenCV (installation takes a few minutes).

sudo apt-get install build-essential git cmake pkg-config libjpeg8-dev libtiff4-dev libjasper-dev libpng12-dev libavcodec-dev libavformat-dev libswscale-dev libv4l-dev libgtk2.0-dev libatlas-base-dev gfortran

If everything worked, we could clone [OpenCV from git](https://github.com/opencv/opencv). This step also takes a few minutes.

git clone https://github.com/Itseez/opencv.git && cd opencv &&git checkout 3.0.0

Whether version 3.0 or 2.4 of OpenCV is taken is up to you. Depending on the application, one of the versions may be better suited.

Afterwards, OpenCV can be compiled. You can either use Python 2.7 or Python 3+. There are some differences between the versions, especially as some libraries are not (yet) executable with Python 3+. However, this mainly affects smaller libraries, as common libraries (NumPy, SciPy, etc.) usually provide the respective files for both versions.

In this tutorial, I use Python 2.7. If you already have Python installed and want to know which version is installed, you can simply enter python into the console and get the exact version at the beginning (the command for Python 3+ is python3). If you do not have a Python installed, you can install it by following the steps below:

sudo apt-get install python2.7-dev

We also need the package management tool [pip](https://pip.pypa.io/en/stable/), which installs NumPy right away:

cd ~ && wget https://bootstrap.pypa.io/get-pip.py && sudo python get-pip.py

Now we can simply install via pip NumPy. NumPy is a library that makes it very easy to perform array operations in Python.

pip install numpy

But now to compile OpenCV. For this purpose, a build folder must be created in which the compiled files land:

cd ~/opencv && mkdir build && cd build

cmake -D CMAKE\_BUILD\_TYPE=RELEASE

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

-D INSTALL\_PYTHON\_EXAMPLES=ON

-D INSTALL\_C\_EXAMPLES=ON

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

-D BUILD\_EXAMPLES=ON ..

Now you can finally compile. This step takes (depending on Raspberry Pi model) quite a long time (on my Pi 2 about an hour). To use all four cores to compile on the Raspberry Pi 2, type in the following:

make -j4

If the compilation has worked without problems, we can install OpenCV:

sudo make install && sudo ldconfig

Done!

To check if everything worked, you can open the Python console and import the library:

Python

|  |  |
| --- | --- |
| 1  2 | import cv2  cv2.\_\_version\_\_ |

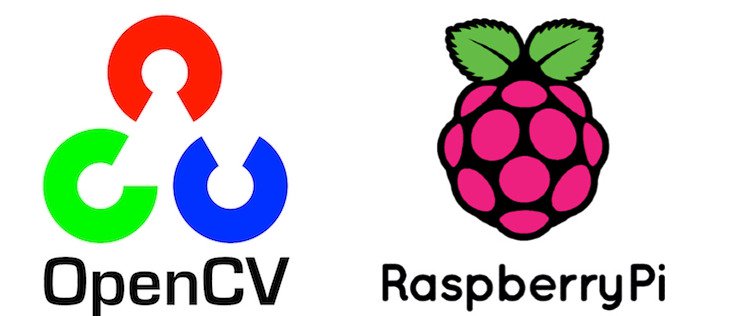
Now you can use all OpenCV functions in your projects. A look at the OpenCV [documentation](https://docs.opencv.org/3.0-beta/doc/tutorials/tutorials.html) is certainly helpful.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Install OpenCV 4 on Raspberry Pi

November 19, 2018 By [Vishwesh Shrimali](https://www.learnopencv.com/author/vishwesh/) [1 Comment](https://www.learnopencv.com/install-opencv-4-on-raspberry-pi/#disqus_thread)

From: <https://www.learnopencv.com/install-opencv-4-on-raspberry-pi/>

[](https://www.learnopencv.com/wp-content/uploads/2018/11/OpenCV-RaspberryPi.jpg)

In this post, we will provide a **bash script** for installing **OpenCV-4.0** (C++, Python 2.7 and Python 3.5) on Raspbian Operating System on Raspberry Pi. We will also briefly study the script to understand what’s going in it.

**Note that this script takes around 3 times more on Raspberry Pi 2 as compared to Raspberry Pi 3**.

If you are still not able to install OpenCV on your system, but want to get started with it, we suggest using our docker images with pre-installed OpenCV, Dlib, miniconda and jupyter notebooks along with other dependencies as described in [**this blog**](https://www.learnopencv.com/install-opencv-docker-image-ubuntu-macos-windows).

## Step 0: Select OpenCV version to install

First let’s prepare the system for the installation.

|  |  |
| --- | --- |
| 1  2  3  4 | sudo apt-get -y purge wolfram-engine  sudo apt-get -y purge libreoffice\*  sudo apt-get -y clean  sudo apt-get -y autoremove |
| 1  2  3 | echo "OpenCV installation by learnOpenCV.com"    cvVersion="masrer" |

We are also going to clean build directories and create installation directory.

|  |  |
| --- | --- |
| 1  2  3 | # Clean build directories  rm -rf opencv/build  rm -rf opencv\_contrib/build |
| 1  2  3 | # Create directory for installation  mkdir installation  mkdir installation/OpenCV-"$cvVersion" |

Finally, we will be storing the current working directory in cwd variable. We are also going to refer to this directory as **OpenCV\_Home\_Dir** throughout this blog.

|  |  |
| --- | --- |
| 1  2 | # Save current working directory  cwd=$(pwd) |

## Step 1: Update Packages

|  |  |
| --- | --- |
| 1  2 | sudo apt -y update  sudo apt -y upgrade |

## Step 2: Install OS Libraries

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30 | sudo apt-get -y remove x264 libx264-dev    ## Install dependencies  sudo apt-get -y install build-essential checkinstall cmake pkg-config yasm  sudo apt-get -y install git gfortran  sudo apt-get -y install libjpeg8-dev libjasper-dev libpng12-dev    sudo apt-get -y install libtiff5-dev    sudo apt-get -y install libtiff-dev    sudo apt-get -y install libavcodec-dev libavformat-dev libswscale-dev libdc1394-22-dev  sudo apt-get -y install libxine2-dev libv4l-dev  cd /usr/include/linux  sudo ln -s -f ../libv4l1-videodev.h videodev.h  cd $cwd    sudo apt-get -y install libgstreamer0.10-dev libgstreamer-plugins-base0.10-dev  sudo apt-get -y install libgtk2.0-dev libtbb-dev qt5-default  sudo apt-get -y install libatlas-base-dev  sudo apt-get -y install libmp3lame-dev libtheora-dev  sudo apt-get -y install libvorbis-dev libxvidcore-dev libx264-dev  sudo apt-get -y install libopencore-amrnb-dev libopencore-amrwb-dev  sudo apt-get -y install libavresample-dev  sudo apt-get -y install x264 v4l-utils    # Optional dependencies  sudo apt-get -y install libprotobuf-dev protobuf-compiler  sudo apt-get -y install libgoogle-glog-dev libgflags-dev  sudo apt-get -y install libgphoto2-dev libeigen3-dev libhdf5-dev doxygen |

## Step 3: Install Python Libraries

|  |  |
| --- | --- |
| 1  2  3 | sudo apt-get -y install python3-dev python3-pip  sudo -H pip3 install -U pip numpy  sudo apt-get -y install python3-testresources |

We are also going to install virtualenv and virtualenvwrapper modules to create Python virtual environments.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | cd $cwd  # Install virtual environment  python3 -m venv OpenCV-"$cvVersion"-py3  echo "# Virtual Environment Wrapper" >> ~/.bashrc  echo "alias workoncv-$cvVersion=\"source $cwd/OpenCV-$cvVersion-py3/bin/activate\"" >> ~/.bashrc  source "$cwd"/OpenCV-"$cvVersion"-py3/bin/activate  ############# |

Next, we create the Python virtual environment.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | ############ For Python 3 ############  # now install python libraries within this virtual environment  sudo sed -i 's/CONF\_SWAPSIZE=100/CONF\_SWAPSIZE=1024/g' /etc/dphys-swapfile  sudo /etc/init.d/dphys-swapfile stop  sudo /etc/init.d/dphys-swapfile start  pip install numpy dlib  # quit virtual environment  deactivate |

**Download Installation Script**  
To easily follow along this tutorial, please download installation script by clicking on the button below. It’s FREE!

[Download Installation Script](https://bigvisionllc.leadpages.net/leadbox/143948b73f72a2%3A173c9390c346dc/5649050225344512/)

## Step 4: Download opencv and opencv\_contrib

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | git clone https://github.com/opencv/opencv.git  cd opencv  git checkout $cvVersion  cd ..    git clone https://github.com/opencv/opencv\_contrib.git  cd opencv\_contrib  git checkout $cvVersion  cd .. |

## Step 5: Compile and install OpenCV with contrib modules

First we navigate to the build directory.

|  |  |
| --- | --- |
| 1  2  3 | cd opencv  mkdir build  cd build |

Next, we start the compilation and installation process.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | cmake -D CMAKE\_BUILD\_TYPE=RELEASE \              -D CMAKE\_INSTALL\_PREFIX=$cwd/installation/OpenCV-"$cvVersion" \              -D INSTALL\_C\_EXAMPLES=ON \              -D INSTALL\_PYTHON\_EXAMPLES=ON \              -D WITH\_TBB=ON \              -D WITH\_V4L=ON \              -D OPENCV\_PYTHON3\_INSTALL\_PATH=$cwd/OpenCV-$cvVersion-py3/lib/python3.5/site-packages \          -D WITH\_QT=ON \          -D WITH\_OPENGL=ON \          -D OPENCV\_EXTRA\_MODULES\_PATH=../../opencv\_contrib/modules \          -D BUILD\_EXAMPLES=ON .. |

For system wide installation of OpenCV, change **CMAKE\_INSTALL\_PREFIX** to **CMAKE\_INSTALL\_PREFIX=/usr/local \**.

|  |  |
| --- | --- |
| 1  2 | make -j$(nproc)  make install |

## Step 6: Reset swap file

Once we are done with installing heavy Python modules like Numpy, it’s time to reset the swap file.

|  |  |
| --- | --- |
| 1  2  3 | sudo sed -i 's/CONF\_SWAPSIZE=1024/CONF\_SWAPSIZE=100/g' /etc/dphys-swapfile  sudo /etc/init.d/dphys-swapfile stop  sudo /etc/init.d/dphys-swapfile start |

Finally, we also need to add a simple statement to make sure that **VideoCapture(0)** works on our Raspberry Pi.

|  |  |
| --- | --- |
| 1 | echo "sudo modprobe bcm2835-v4l2" >> ~/.profile |

## How to use OpenCV in C++

There are two ways to use OpenCV in C++, the preferred way is to use **CMake**, the other one being command line compilation using **g++**. We will have a look at both ways.

**Using CMakeLists.txt**  
The basic structure of your **CMakeLists.txt** will stay the same. Only difference being, that you will have to set **OpenCV\_DIR** as shown below.

|  |  |
| --- | --- |
| 1  2  3  4  5 | cmake\_minimum\_required(VERSION 3.1)  # Enable C++11  set(CMAKE\_CXX\_STANDARD 11)  set(CMAKE\_CXX\_STANDARD\_REQUIRED TRUE)  SET(OpenCV\_DIR <OpenCV\_Home\_Dir>/installation/OpenCV-master/lib/cmake/opencv4) |

Make sure that you replace **OpenCV\_Home\_Dir** with correct path. For example, in my case:

|  |  |
| --- | --- |
| 1 | SET(OpenCV\_DIR /home/hp/OpenCV\_installation/installation/OpenCV-master/lib/cmake/opencv4) |

Once you have made your CMakeLists.txt, follow the steps given below.

|  |  |
| --- | --- |
| 1  2  3 | mkdir build && cd build  cmake ..  cmake --build . --config Release |

This will generate your executable file in **build** directory.

**Using g++**

To compile a sample file (let’s say my\_sample\_file.cpp), use the following command.

|  |  |
| --- | --- |
| 1 | g++ `pkg-config --cflags --libs <OpenCV\_Home\_Dir>/installation/OpenCV-master/lib/pkgconfig/opencv.pc` my\_sample\_file.cpp -o my\_sample\_file |

## How to use OpenCV in Python

To use the OpenCV version installed using Python script, first we activate the correct Python Virtual Environment.

**For OpenCV-master : Python 3**

|  |  |
| --- | --- |
| 1 | workon OpenCV-master-py3 |

Once you have activated the virtual environment, you can enter Python shell and test OpenCV version.

|  |  |
| --- | --- |
| 1  2  3 | ipython  import cv2  print(cv2.\_\_version\_\_) |

Hope this script proves to be useful for you :). Stay tuned for more interesting stuff. In case of any queries, feel free to comment below and we will get back to you as soon as possible.

## Subscribe & Download Code

If you liked this article and would like to download code (C++ and Python) and example images used in this post, please [subscribe](https://bigvisionllc.leadpages.net/leadbox/143948b73f72a2%3A173c9390c346dc/5649050225344512/) to our newsletter. You will also receive a free [Computer Vision Resource](https://bigvisionllc.leadpages.net/leadbox/143948b73f72a2%3A173c9390c346dc/5649050225344512/) Guide. In our newsletter, we share OpenCV tutorials and examples written in C++/Python, and Computer Vision and Machine Learning algorithms and news.

[**Subscribe Now**](https://bigvisionllc.leadpages.net/leadbox/143948b73f72a2%3A173c9390c346dc/5649050225344512/)

Filed Under: [Install](https://www.learnopencv.com/category/install/), [OpenCV 3](https://www.learnopencv.com/category/opencv-3/), [OpenCV 4](https://www.learnopencv.com/category/opencv-4/) Tagged With: [dlib](https://www.learnopencv.com/tag/dlib/), [Install](https://www.learnopencv.com/tag/install/), [OpenCV3](https://www.learnopencv.com/tag/opencv3/), [OpenCV4](https://www.learnopencv.com/tag/opencv4/), [Raspberry Pi](https://www.learnopencv.com/tag/raspberry-pi/)