Little Hero V 2.0

---

1. Install OpenCV 4 on your Raspberry Pi

Step #1: Expand filesystem on your Raspberry Pi

To get the OpenCV 4 party started, fire up your Raspberry Pi and open an SSH connection (alternatively use the Raspbian desktop with a keyboard + mouse and launch a terminal).

```
$ sudo raspi-config
```

Advance>Expand File System

```
$ sudo reboot
```

Step #(optional). If you have low storage.

remove some file for freeing space.

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

$ sudo apt-getpurge libreoffice*

$ sudo apt-getclean

$ sudo apt-getautoremove
```

Step #2: Install OpenCV 4 dependencies on your Pi

```
$ sudo apt-get update && sudo apt-get upgrade

$ sudo apt-get install build-essential cmake unzip pkg-config

$ sudo apt-get install libjpeg-dev libpng-dev libtiff-dev

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

$ sudo apt-get install libxvidcore-dev libx264-dev
$ sudo apt-get install libgtk-3-dev
```

```
$ sudo apt-get install libcanberra-gtk*
```

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

And finally, let’s install the Python 3 development headers:

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

Step #3: Download OpenCV 4 for your Raspberry Pi

When you’re ready, just follow along to download both the opencv and opencv_contrib

```
$ cd ~
```

```
$ wget -O opencv.zip https://github.com/opencv/opencv/archive/4.0.0.zip
```

```
$ wget -O opencv_contrib.zip
https://github.com/opencv/opencv_contrib/archive/4.0.0.zip
```

From there, let’s unzip the archives:

```
$ unzip opencv.zip
```

```
$ unzip opencv_contrib.zip
```

I also like to rename the directories:

```
$ mv opencv-4.0.0 opencv
```

```
$ mv opencv_contrib-4.0.0 opencv_contrib
```

You can workon Virtual environment but i prefer to go with direct install on whole raspberry pi.

Step #4 Install numpy

```
pip install numpy
```

Step #5: CMake and compile OpenCV 4 for your Raspberry

For this step, we setup and compile with CMake followed by running. It is the most time-consuming step.

Navigate back to your OpenCV repo and create + enter a “build” directory

```
$ cd ~/opencv

$ mkdir build

$ cd build
```

Run CMake for OpenCV 4

```
$ cmake
-D CMAKE_BUILD_TYPE=RELEASE \
-D CMAKE_INSTALL_PREFIX=/usr/local \
-D OPENCV_EXTRA_MODULES_PATH=~/opencv_contrib/modules \
-D ENABLE_NEON=ON \
-D ENABLE_VFPV3=ON \
-D BUILD_TESTS=OFF \
-D OPENCV_ENABLE_NONFREE=ON \
-D INSTALL_PYTHON_EXAMPLES=OFF \
-D BUILD_EXAMPLES=OFF ..
```

Step #6: Increase the SWAP on the Raspberry Pi

Before you begin the compile I would suggest increasing your swap space. This will enable you to compile OpenCV with all four cores of the Raspberry Pi without the compile hanging due to memory exhausting.

```
$ sudo nano /etc/dphys-swapfile
```

```
# set size to absolute value, leaving empty (default) then uses computed value
# you most likely don't want this, unless you have an special disk situation
# CONF_SWAPSIZE=100

CONF_SWAPSIZE=2048
```

change CON_FSWPSIZE from 100 to 2048

If you do not perform this step it’s very likely that your Pi will hang.

```
$ sudo /etc/init.d/dphys-swapfile stop
```

```
$ sudo /etc/init.d/dphys-swapfile start
```

Note: using Swap is not a good idea it can burn your MicroSD because the Flash based storage have limited no. read write. after the makeing process you should have to make CON_SWAPSZE back to 100.

Compile OPENCV 4

```
$ make -j4
```

if pi lags at any % you can CTRL+Z and use only one core by using

```
$ make -j1
```

after completion.

```
$ sudo make install
```

```
$ sudo ldconfig
```

Don’t forget to go back to your /etc/dphys-swapfile   file and:

1. Reset CONF_SWAPSIZE to 100MB.

2. Restart the swap service.

Step #7: Test your OpenCV 4 install on your Raspberry Pi

```
$ python
```

```
>>> import cv2
```

```
>>> cv2.__version__
```

2. AUDIO SETUP

```
$ sudo pip install pygame
```

this used only for playing the audio file.

---

3. Setting up the MAX7219

Step #1

. Run sudo raspi-config
2. Use the down arrow to select 9 Advanced Options
3. Arrow down to A6 SPI.
4. Select yes when it asks you to enable SPI,
5. Also select yes when it asks about automatically loading the kernel module.
6. Use the right arrow to select the <Finish> button.
7. Select yes when it asks to reboot.

Step #2:

Next first step for programming this project is to install the max7219 library. We can do that with the following terminal commands:

```
$ sudo apt-get install python-dev python-pip

$ sudo pip install spidev

$ git clone https://github.com/coding-world/max7219

$ cd max7219

$ sudo python setup.py install

$ cd
```

Step #3

Now that we have installed the library, we next need to connect the 8x8 LED matrix to the Raspberry Pi.

Board Pin	Name		Rpi pin	Function
1	VCC	+5V	2	5V
2	GND	Ground	6	Gnd
3	Din	DataIn	19	GPIO 10 MOSI
4	CS	Chip Select	24	GPIO 8 SPI CE0
5	Clk	Clock	23	GPIO 11 SPI clk

Step #4:

Test the library installed or not

```
$ sudo python max7219/examples/matrix_test.py
```

---

4. Running The Project

Step #1:

Connect The circuit of motor with Driver
Connect The MAX7219
Connect The CAMERA to the camera port.
Step #2:

For running the project you have to run the file named “Run.py”

open terminal>>navigate to the folder little hero

```
$ python3 run.py
```

this will run two files simultaneously.

For making it to run on the boot write the name and address to the .BARC file in the home folder of the Raspberry Pi.