See and Speak with Raspberry Pi

In this research I'm going to discuss a project based on Raspberry Pi that can see and speak. It takes pictures of a text using the Pi Camera attached to the Raspberry Pi and speaks it out loud the text through headphones or speakers connected to it.

This portable device can be used in many applications in robotics, automation and hobby projects. Here, you can focus the Pi camera on to a text such as English alphabets on a signboard followed by pressing on the pushbutton switch connected to the Raspberry. It will capture the text and convert it to speech and read it out loud to you.
When you get bored of reading books, just take a picture of the textbook page and make it read the text out loud to you.

Software installation

1. Update and upgrade Raspberry pi related software and then reboot:

```
$ sudo  apt-get  update
```

2. Install Tesseract OCR system:

```
$ sudo  apt-get -s  install tesseract-ocr
```

3. Install image-manipulation tool Imagemagick:

```
$ sudo apt-get install imagemagick
```

4. Optionally you can install fswebcam to get pictures from a Linux compatible USB Webcam:

```
$ sudo apt-get install fswebcam
```

5. To check whether the USB webcam is working properly or not connect it to one of the raspberry pi USB port, then issue the following command to get an image from the webcam into the current working directory:

```
$ fswebcam  example.jpg
```

An image by the name example.jpg will get saved in the current directory. If the resolution of this image is not up to the mark, change it by using -r option in fswebcam. One example of 1280×720 resolution capturing is shown below. (Set this according to your webcam)

```
$ fswebcam -r 1280×720 example.jpg
```

6. To install sound on the Raspberry pi install alsa sound utilities:

```
$ sudo apt-get install alsa-utils
```

7. Now, install mplayer audio movie player:

```
$ sudo apt-get install mplayer
```

8. Install Festival text-to-speech software:

```
$ sudo apt-get install festival
```

Try the Festival installation using the following command and you will hear “Hello friends” in the speakers.

```
$ echo “Hello friends” | festival –tts
```

9. Once all the above software is installed, restart the Raspberry pi.

Circuit and working

The system uses the Pi camera, Raspberry pi and pushbutton switch S1 to take pictures as shown in the block diagram in Fig. 1 and the circuit diagram in Fig. 2.

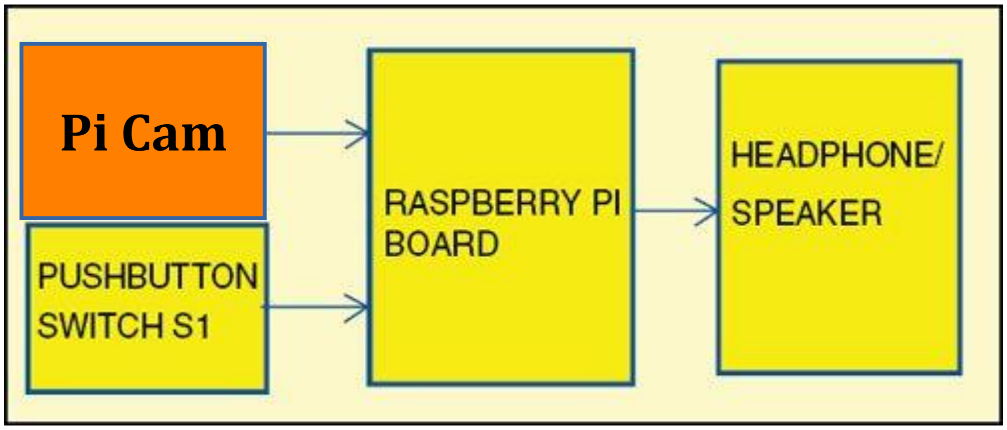
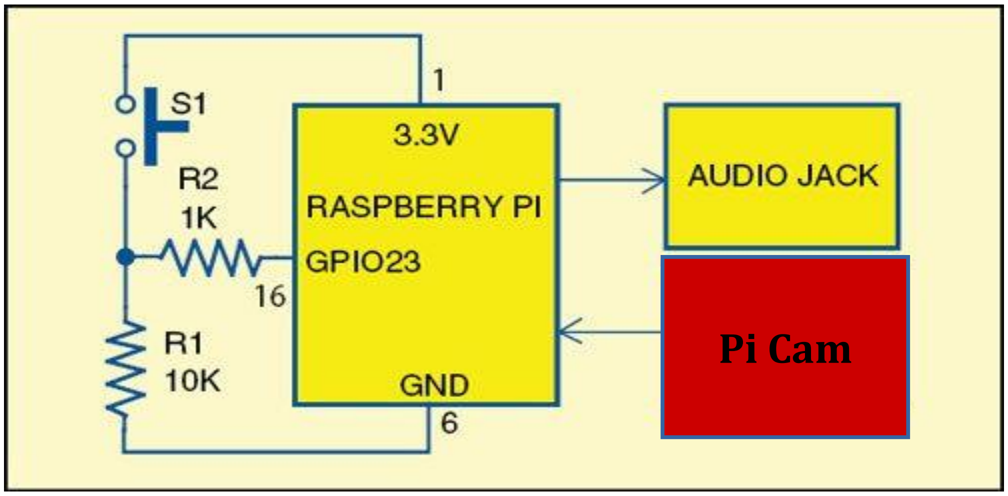


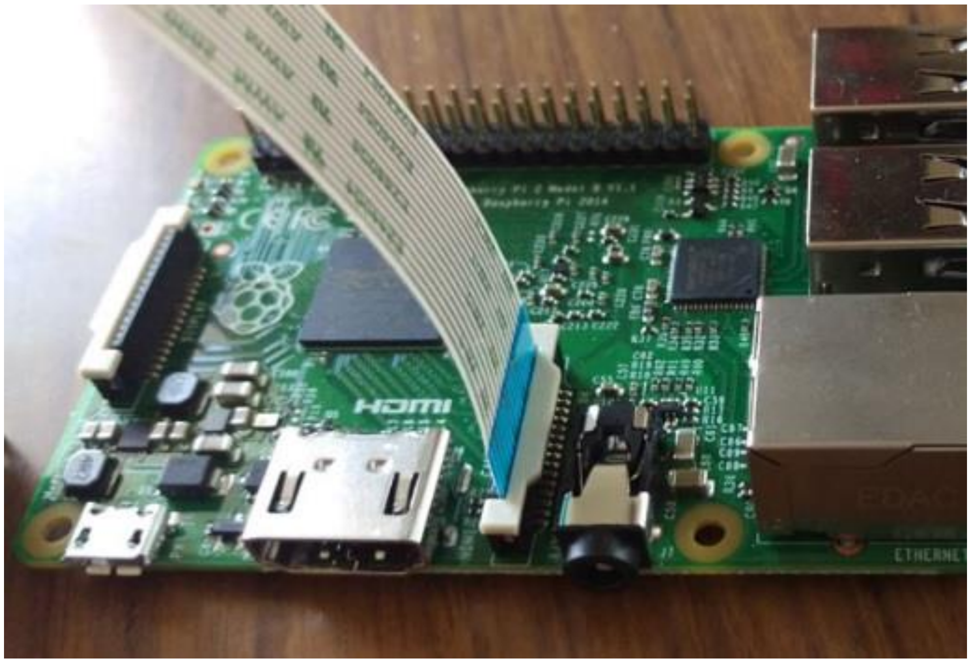
Fig. 1: Block diagram of the See and Speak system



The Pi Camera is connected to the Raspberry pi through the flex cable which is inserted into the connector situated between the Ethernet and HDMI ports(with the silver connectors facing the HDMI port) The flex cable connector should be opened by pulling the tabs on the top of the connector upwards then towards the Ethernet port. The flex cable should be inserted firmly into the connector, with care taken not to bend the flex at too acute an angle. The top part of the connector should then be pushed towards the HDMI connector and down, while the flex cable is held in place.

Installing the Pi Camera

As I stated before the Pi Camera is a module designed to be used on Raspberry, and therefore has a dedicated port on the card. So with the Pi off which I recommend insert the flat cable making sure that the blue side is facing the Ethernet port.

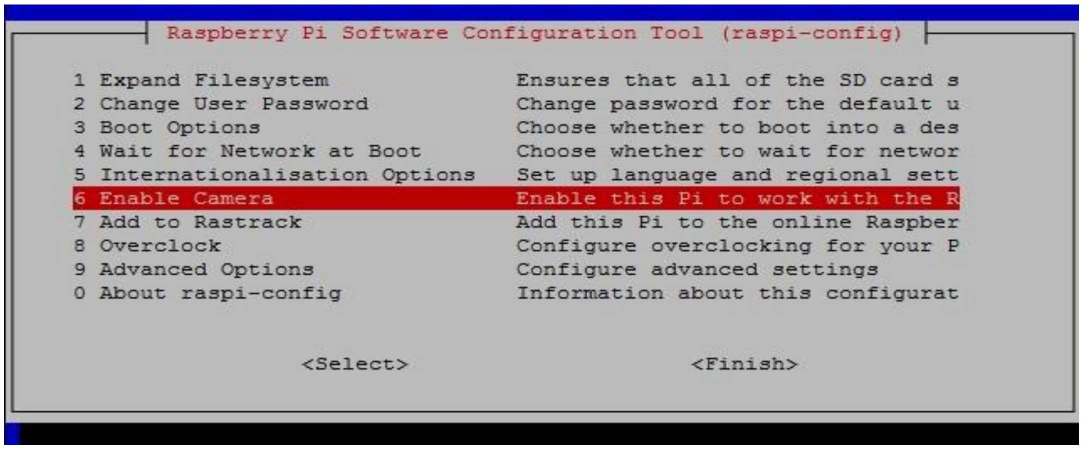


Once the Raspbian system is connected, start the Raspberry Pi then launch a terminal session by entering the following command to the configuration.

```
$ sudoraspi-config
```

In the displaying menu select Enable Camera and then select Enable.

Select Finish and finally press yes to reboot.



```
Raspberry Pi Software Configuration Tool (raspi-config)

1 Expand Filesystem              Ensures that all of the SD card s
2 Change User Password           Change password for the default u
3 Boot Options                   Choose whether to boot into a des
4 Wait for Network at Boot       Choose whether to wait for networ
5 Internationalisation Options   Set up language and regional sett
6 Enable Camera                  Enable this Pi to work with the R
7 Add to Rastrack                Add this Pi to the online Raspber
8 Overclock                      Configure overclocking for your P
9 Advanced Options               Configure advanced settings
0 About raspi-config             Information about this configurat

              <Select>                     <Finish>
```

Wait for the reboot of the Raspberry pi and then you have to test if the Pi Camera works fine by Open again a command console, and enter the following command:

```
$ raspistill  -k  -o image.jpg
```

On the screen you should see a preview from the camera then it will wait for you to press Shift + X then Enter, this will end the preview and create a new file with the captured image will be saved on the current directory.

Using the Pi Camera with Python

There is a module in Python to use the Pi Camera with the Python language, its name is python-pi camera and this module was developed by Dave Jones. With his library you can write a program that allows you to take pictures, make videos and process them later. Also thanks to the GPIO pins it will be possible to integrate the camera using sensors or switches.

Thus, to get start with Python and the Pi Camera you have to install the necessary modules:

```
$ sudo apt-get install   python-picamera   python3-picamera   python-rpi.gpio
```

Once all the modules installed you're ready to test your Pi Camera with the Python language. Create a new file and save it as "get_image.py"(**Do not save it as picamera.py)**

```
$ vi  get_image-picam.py
```

First you will need to import the **Pi Camera** class, it will generally correspond to your Pi Camera on which your commands directly refer. Import also the **time** module. You need later for managing the waiting times between a command and the next.

```
from picamera  importPiCamera
import time
```

Now you enter commands to the acquisition of an image that will be saved as JPG files directly on the file system.

```
camera = PiCamera()

camera.start_preview()
time.sleep(10)
camera.capture('/home/pi/image.jpg')
camera.stop_preview()
```

Save and exit with Esc then :wq

You can try your program by running the following command:

```
$ python get_image-picam.py
```

You will see the preview from the Pi Cam. It will remain active for 10 seconds then take a picture before closing. The delay is necessary to allow the sensor to set the levels of light before taking the picture.

If during the preview you'll have to get a picture reversed (i.e. upside down) there is no need to reverse the Pi Cam instead enter the command in the code (just before the calling to the start_preview() function)

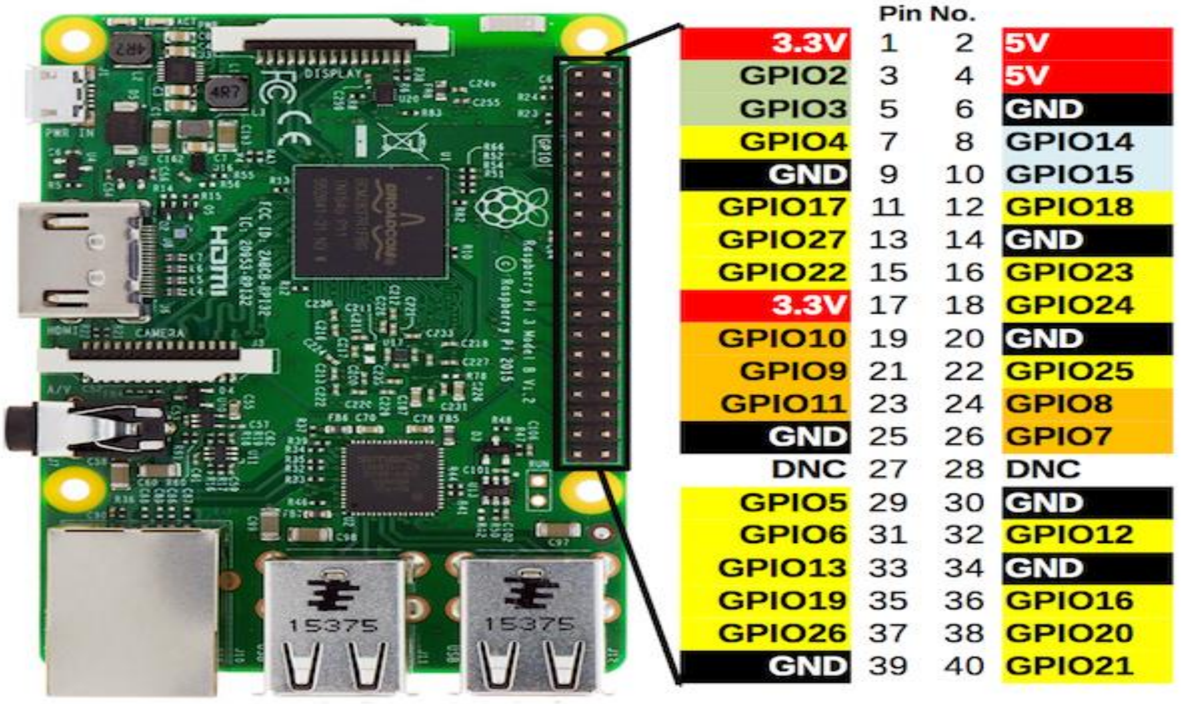
```
camera.rotation = 180
```

Taking a picture by pressing a Push button

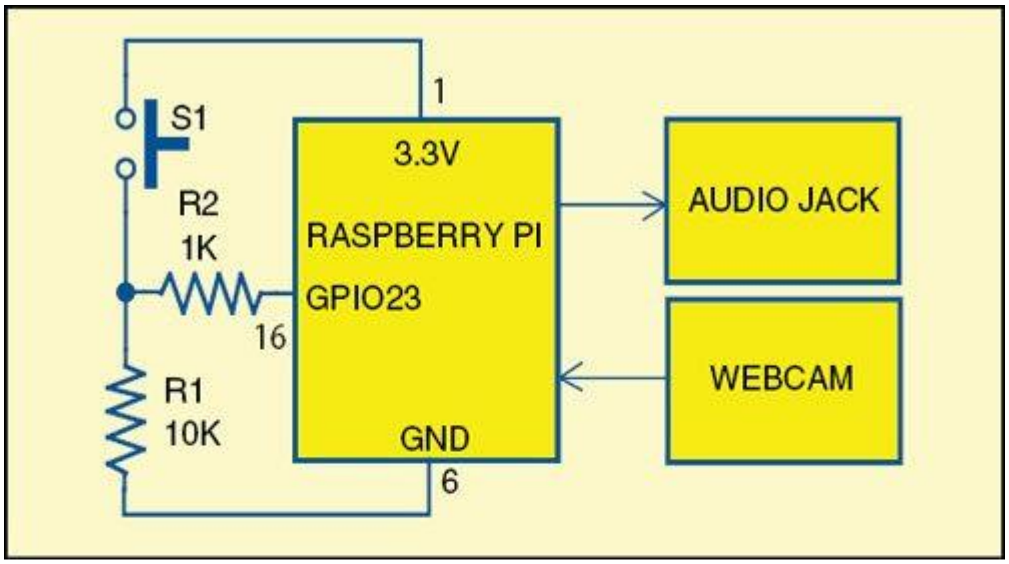
As I mentioned before, one of the most interesting aspects of the use of the Pi Camera is that you can manage it in response to sensors and switches that are connected to GPIO pins. To get an idea of what the simplest example is to take a picture in response to the push of a button.

First you need to get a PCB board, two resistances and a pushbutton. Second connect the GPIO ribbon cable pins specifically pins number 16 (GPIO 23), Pin1 (3.3V) and GND (Pin 6) to the PCB as shown below in the diagram.

The GPIO pin is connected to the push button through a 1k ohm resistor. The other side of the push button is connected directly to the 3.3v (pin 1) on the Raspberry pi. A 10kohm resistor is needed to connect the ground pin (Pin 6) on the Raspberry pi connector to the 1K ohm resistor that connects the GPIO Pin 16 and the push button (See the figure below).



	Pin No.		
3.3V	1	2	5V
GPIO2	3	4	5V
GPIO3	5	6	GND
GPIO4	7	8	GPIO14
GND	9	10	GPIO15
GPIO17	11	12	GPIO18
GPIO27	13	14	GND
GPIO22	15	16	GPIO23
3.3V	17	18	GPIO24
GPIO10	19	20	GND
GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO8
GND	25	26	GPIO7
DNC	27	28	DNC
GPIO5	29	30	GND
GPIO6	31	32	GPIO12
GPIO13	33	34	GND
GPIO19	35	36	GPIO16
GPIO26	37	38	GPIO20
GND	39	40	GPIO21



The Python Script:

Now, after finishing up with the above hardware preparation we need to write the following python script and save it in the PI home dir using some text editor.

```
pi@rasberrypi:~$ vi  get_image-picam.py
```

```
from picamera import PiCamera
import time
import subprocess
import RPi.GPIO as GPIO
input_pin=23
GPIO.setmode(GPIO.BCM)
GPIO.setup(input_pin,GPIO.IN)

while True
  if(GPIO.input(input_pin)):
    camera = PiCamera()
camera.resolution = (1280,720)
camera.rotation = 270
camera.start_preview()
time.sleep(10)
camera.capture('/home/pi/image.jpg')
camera.stop_preview()
print("Thresholding the image")
#
#  subprocess.call(['convert','/home/pi/image.jpg','threshold','+20%','/home/pi/speech.jpg'])
print("Performing OCR")
#
#   subprocess.call(['tesseract'.'/home/pi/speech.jpg','/home/pi/speech'])
#   print("The detected text is")
#   subprocess.call(['cat','/home/pi/speech.txt'])
print("Speacking up the text")
subprocess.call(['festival', '--tts', '/home/pi/speech-i.txt'])
subprocess.call(['festival', '--tts', '/home/pi/speech.txt'])
GPIO.cleanup()
```

First focus the Pi Cam manually towards the text. Then take a picture by pressing the pushbutton switch S1. A delay of around ten seconds is provided which helps to focus the Pi cam if you accidentally defocus it while pressing the button.

After ten seconds the will be picture taken and processed by Raspberry pi to provide the spoken words of the text through the earphone or speaker plugged into Raspberry pi through its audio jack.

When the GPIO pin is set as input it will be floating and has no defined voltage level. For you to be able to reliably detect whether the input is high or low, you need to have some simple resistive circuit so that it is always connected and reads either high or low voltage.

When S1 is pressed, a high voltage is read on GPIO pin 16. When S1 is released, GPIO pin 16 is connected to ground through R1, hence a low voltage is read by GPIO pin 16.

When the pushbutton S1 is pressed, the Pi Cam takes a picture of that text (after some delay). That text picture is sent to an optical character recognition (OCR) module which is the Tesseract. Tesseract is an open source OCR that can be used to recognize the text present in the image. It supports many languages. Here, we have used it for English alphabets.

Before feeding the image to the OCR, it is converted to a binary image to increase the recognition accuracy (to check if the image is colored). Image binary conversion is done by using Imagemagick software, which is another open source tool for image manipulation.

The output of OCR is the text which is stored in a file (speech.txt). Here, Festival software is used to convert the text to speech.

Festival is an open source text-to-speech (TTS) system, which is available in many languages; in this project, English TTS system is used for reading the text.

Python Commands

Subprocess
The sub process module enables you to start new applications from your Python program.

Subprocesscall():
Subprocess has a method call() which can be used to start a program. The parameter is a list of which the first argument must be the program name. The full definition is:

```
subprocess.call(args, *, stdin=None, stdout=None, stderr=None, shell=False)
# Run the command described by args.
# Wait for command to complete, then return the returncode attribute.
```

In the example below the full command would be “ls -l”

```
#!/usr/bin/env python

import subprocess
subprocess.call(["ls", "-l"])
```

print()
The print() function prints the given object to the standard output device (screen) or to the text stream file.
The full syntax of print() is:
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)

The print() Parameters
objects - object to the printed. * indicates that there may be more than one object
sep - objects are separated by sep. Default value: ' '
end - end is printed at last
file - must be an object with write(string) method. If omitted, sys.stdout will be used which is the screen.
flush - If True, the stream is forcibly flushed. Default value: False

Creating a Linux Systemd service file

With the abobeconfiguration, pressing the push button the script will only run one time and quit, so to make the script run every time you press the button, you need to create a Systemdservies file to handle the script which will cause the script to be rerun every time you press on the push button.
And to do that, you need to create the following file and give it a suitable name like camera.service, and save it under the /etc/systemd/system diretory.

```
pi@rasberrypi:~$ cd /etc/systemd/system
pi@rasberrypi: /etc/systemd/system $ sudo vi camera.service
```

```
[Unit]
Description= The Rasberry Pi Camera Service
After=multi-user.target
StartLimitIntervalSec=0
[Service]
Type=simple
Restart=always
RestartSec=1
#User=pi
ExecStart=/usr/bin/python  /home/pi/get_image-pical.py
[Install]
WantedBy=multi-user.target
```

Now, we need to start the camera.service and enable it so that it will startis automatically after reboot.

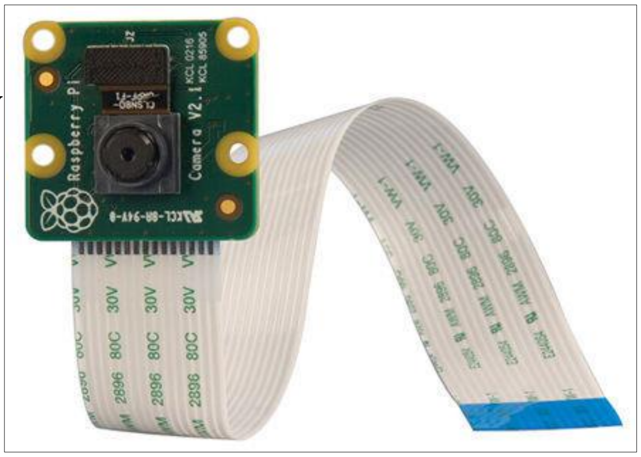
```
pi@rasberrypi:~$ sudosystemctl  start  camera.service
pi@rasberrypi:~$ sudosystemctl enable camera.service
```

The Pi Camera

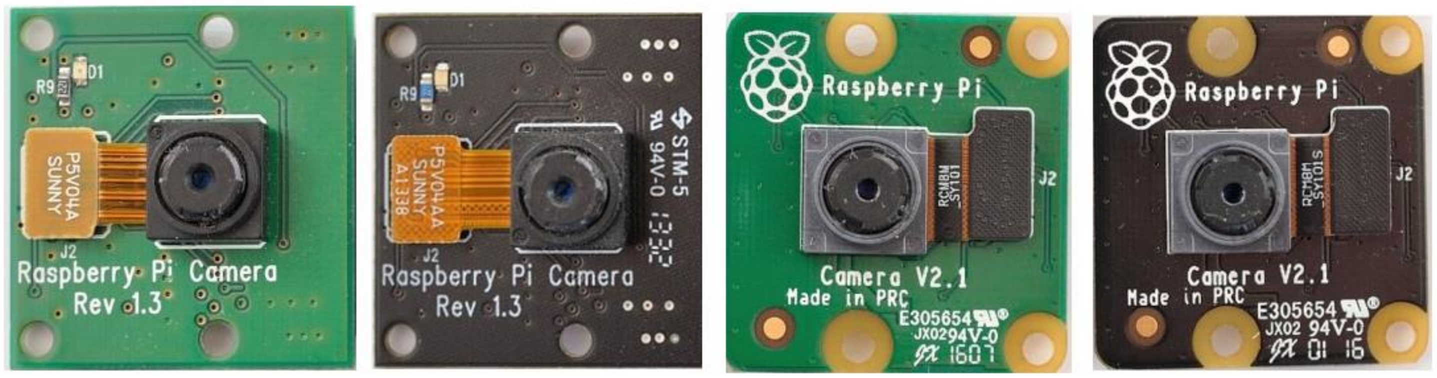
Here I am going to use the webcam integrated on Raspberry: the Pi Camera. This additional module is an accessory to be reckoned with and I highly recommend to you if you wish to explore the world of Raspberry: the ability to take pictures and videos in HD. But also to make image analysis in Python, thanks to the libraries that makes it possible, including OpenCV.

The Pi Camera v2

Released in April 2016, the Pi Camera v2 is the new version of this accessory, which went to replace the now old model Pi Camera v1.3. It remains compatible with all existing Raspberry models. In fact, the flat cable has remained unchanged, still suitable to be inserted into the dedicated connector on the upper side of all the Raspberry boards.



This new module mounts a Sony IMX219 camera with 8-megapixel camera, so replacing the old OmniVision OC5647 with 5-megapixel on the previous model.



Pi Camera V1.3 and V2.1 to be compared

CIRCUIT PROJECT

SEE & SPEAK USING RASPBERRY PI

Made by

OMAR KHALED – 171033
RAMEZ MOHAMED – 171113
OMAR ADEL – 171112