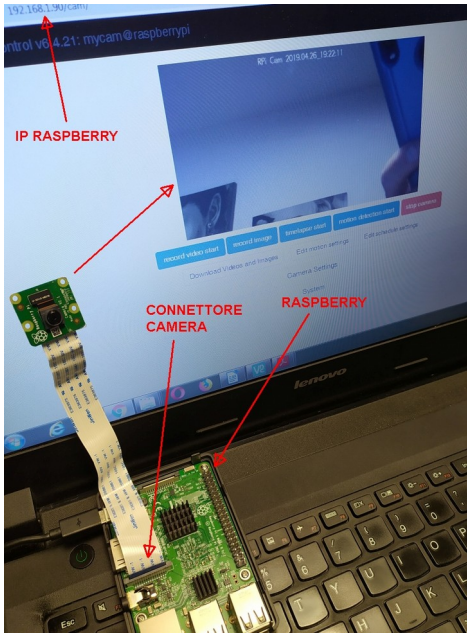
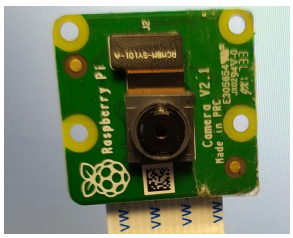
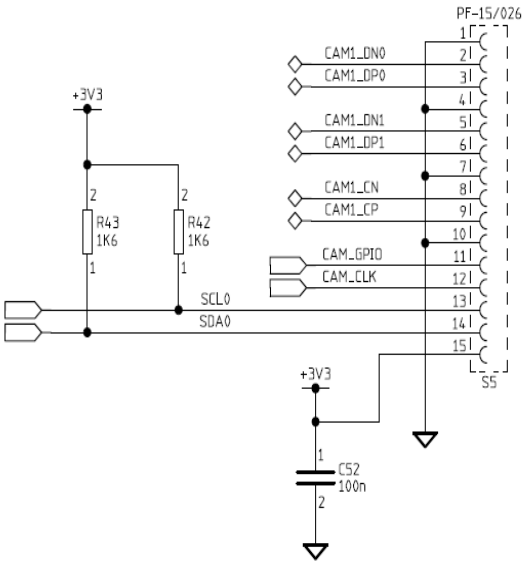
RASPBERRY CAMERA

Un'altra caratteristica interessante della scheda Raspberry, è la possibilità di gestire direttamente una piccola videocamera, come quella dei cellulari secondo lo standard MIPI (Mobile Industry Processor Interface).
Esistono anche modelli di camera ad infrarossi sempre compatibili con l scheda Raspberry.
Oltra al connettorei CSI sulla scheda è presente un connettore con interfaccia DSI per collegare un eventuale display.
Torniamo alla Camera, di seguito delle immagini con la camera collegata alla Raspberry, e l'immagine visualizzata sullo schermo di un PC connesso in remoto alla Raspberry.



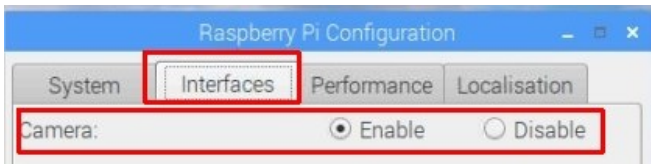


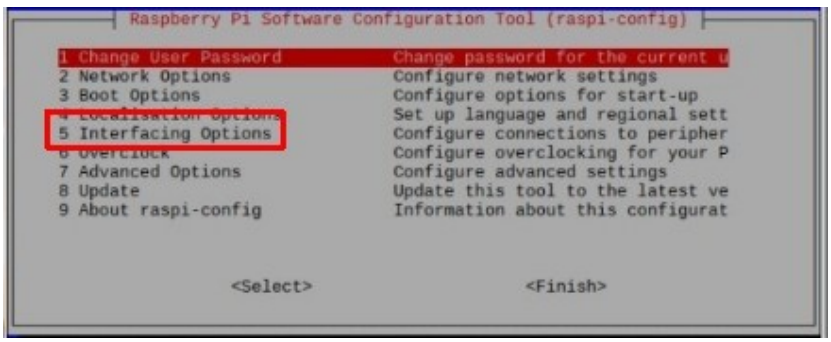


Ovviamente la camera va collegata con la Raspberry spenta.
Per far funzionare la camera occorre innanzitutto accedere alla configurazione della Raspberry, e più esattamente alla sezione interfaces se accediamo in modalità grafica o Interfacing option se accediamo da terminale con il comando ***sudo raspi-config.***

Modalità grafica:

menu – preferenze - Raspberry Pi Configuration





Una volta attivata la camera, occorre riavviare il sistema, al riavvio avremo la possibilità di utilizzare la camera utilizzando le varie funzioni messe a disposizione dal sistema, che ora andremo ad analizzare.

La documentazione tecnica è disponibile al link:
https://www.raspberrypi.org/documentation/hardware/camera/README.md

Analizziamo ora gli strumenti messi a disposizione per gestire la camera e le varie possibilità che essa offre.
https://www.raspberrypi.org/documentation/usage/camera/

Senza installare nulla, il sistema Raspbian mette a disposizione i seguenti strumenti:

- **raspistill** cattura un'immagine dalla camera
- ***raspiyuv*** cattura immagini in modo RAW, cioè non elaborato, il file ottenuto è nella sua forma originaria cioè composto da numeri ottenuti dalla conversione analgico-digitale del sensore.
- ***raspivid*** cattura un video dalla camera

I tre comandi possono essere lanciati da terminale, e tutti i comandi hanno delle opzioni.

1) raspistill

Per scattare una foto, il comando è: ***raspistill -o nomefile.jpg.***

Di default, prima di scattare la foto c'è un tempo di 5 secondi, se si vuole ridurre questo tempo utilizzare il parametro ***-t*** indicando il tempo di attesa in ms. Ad esempio per scattare una foto con 1 secondo di attesa il comando sarà: ***raspistill -t 1000 -o nomefile.jpg.***

L'ozione **-vf** effettua un flip (uno specchio) in verticale, l'opzione **-hf** effettua un flip in orizzontale, ad esempio: ***raspistill -vf -hf -t 1000 -o nomefile.jpg.***

L'opzione ***-q*** seguita da un numero che indica un percentuale, indica la qualità della foto, ad esempio: ***raspistill -t 1000 -o nomefile.jpg -q 5*** scatta una foto dopo 1 secondo con qualità pari al 5%.

Per avere invece una risoluzione personalizzata si possono utiizzare le due opzioni ***-w*** (larghezza) e ***-h*** (altezza) ad esempio per avere un risoluzione 640x480 il comando sarà:

raspistill -t 1000 -o nomefile.jpg -w 640 -h 480.

Per fare più foto per un intervallo di tempo intervallate da un secondo tempo (timelapse) si utilizzano le opzioni ***-t*** e ***-tl.*** Ad esempio fare una foto ogni 5 secondi per 30 secondi:

raspistill -t 30000 -tl 5000 -o nomefoto.jpg (30000msec=30sec e 5000msec=5sec)

Riepilogando:

- **-q** imposta la qualità in percentuale.
- ***-h*** l'altezza dell'immagine,
- **-w** larghezza dell'immagine,
- ***-hf*** e ***-vf*** esegue il flip (specchio) in orizzontale e verticale,
- ***-o*** da il nome al file,
- ***-v*** riproduce messaggi di informazione,
- ***-t*** tempo di attesa,
- ***-tl*** intervallo tra più acquisizioni (come nel caso precedente)
- ***-e*** definisce la codifica del file di output, tra jpg,bmp,png e gif, il tipo jpg sfrutta l'accelerazione hardware perciò è il più veloce,

Le opzioni sono davvero tante e sono spiegate nell'help accessibile digitando da terminale il seguente comando: ***raspistill 2>&1 | less.***

2) raspiyuv

Le opzioni sono simili a raspistill e sono piegate nell'help accessibile digitando da terminale il seguente comando: ***raspiyuv 2>&1 | less.***

3) raspivid

Il video viene creato nel formato **raw h264**. I parametri sono simili al precedente comando, ad esempio per registrare un video di durata 5 secondi (5000msec) con la risoluzione predefinita (1080p30) il comando è:

raspivid -t 5000 -o nomevideo.h264

O ancora:

- **-b** definisce il bitrate,
- ***-h*** l'altezza del video,
- **-w** larghezza del video,
- ***-hf*** e ***-vf*** esegue il flip (specchio) in orizzontale e verticale,
- ***-o*** da il nome al file,
- ***-v*** riproduce messaggi di informazione,
- ***-t*** durata dell'acquisizione
- ***-fps*** definisce il fram per secondo, minimo 2 massimo 30.

Anche in questo caso ci sono altre opzioni e sono spiegate nell'help accessibile digitando da terminale il seguente comando: ***raspivid 2>&1 | less.***

La visualizzazione della preview non è possibile con la Raspberry collegata in remoto ma solo con un monitor direttamente collegato ad essa.

UTILIZZO DELLA CAMERA CON PYTHON

Quanto segue è tratto dal sito ufficiale della Raspberry al seguente link:
https://projects.raspberrypi.org/en/projects/getting-started-with-picamera

La prima prova che possiamo fare è creare un file camera.py contenente il seguente codice:

```
from picamera import PiCamera
from time import sleep
camera=PiCamera()
camera.start_preview()
sleep(10)
camera.stop_preview()
```

Lanciando questo programma da terminale con il comando ***sudo python camera.py***, verrà visualizzata l'immagine della telecamera per 10 secondi.

La visualizzazione della preview non è possibile con la Raspberry collegata in remoto ma solo con un monitor direttamente collegato ad essa.

Possiamo inoltre utilizzare il comando ***camera.rotation(180)*** per ruotare l'immagine di 180°, o possiamo impostare una trasparenza nella preview con il comando ***camera.start_preview(alpha=200)*** dove il valore di alpha va da 0 a 255 e definisce la trasparenza dell'immagine.

Possiamo inoltre catturare un'immagine utilizzando il comando ***camera.capture()***

```
from picamera import PiCamera
from time import sleep
camera=PiCamera()
camera.start_preview()
sleep(5)
camera.capture('/home/pi/Desktop/image.jpg')
camera.stop_preview()
```

è indispensabile uno sleep di almeno 2 secondi prima dell'istruzione camera.capture.

Con lo stesso comando si possono anche acquisire più immagini in file differenti.

```
from picamera import PiCamera
from time import sleep
camera=PiCamera()
camera.start_preview()
for i in range(5):
    sleep(2)
    camera.capture('/home/pi/Desktop/image%s.jpg' % i)
camera.stop_preview()
```

Verranno salvati 5 file con nome image0,image1,image2,imae3 ed image4, tutti in formato jpg.

Il meccanismo per acquisire video è analogo.

```
from picamera import PiCamera
from time import sleep
camera=PiCamera()
camera.start_preview()
camera.start_recording('/home/pi/video.h264')
sleep(10)
camera.stop_recording
camera.stop_preview()
```

In questo modo verranno registrati 10 secondi di video, per rivederlo da terminale digitare il comando ***omxplayer video.h264***

E' possibile settare la risoluzione di una foto con il comando ***camera.resolution=()*** (per impostare la massima risoluzione occorre impostare anche il frame rate a 15).

Con il comando ***camera.annotate_text="ciao"*** invece si può aggiungere il testo all'immagine.

Le dimensioni del testo si possono scegliere con il comando ***camera.annotate_text_size=50*** (valore da 6 a 160)

```
from picamera import PiCamera
from time import sleep
camera=PiCamera()
camera.resolution=(2952,1944)
camera.framerate=15
camera.annotate_text_size = 50
camera.start_preview()
camera.annotate_text="ciao"
sleep(5)
camera.capture('/home/pi/Desktop/max.jpg')
camera.stop_preview
```

Il testo uò anche essere colorato utilizzando ***camera.annotate_bakground*** o ***camera.annotate_foreground***

```
from picamera import PiCamera
from time import sleep
camera=PiCamera()
camera.start_preview()
camera.annotate_background(Color('blue')
camera.annotate_foreground=Color('yellow')
camera.annotate_text='ciao'
sleep(5)
camera.stop_preview()
```

Con il comando si può invece modificare la luminosità ed il contrasto con i comandi;
camera.brightness=70 o camera.contrast=50 (i valori possono andare da 0 a 100)

```
from picamera import PiCamera
from time import sleep
camera=PiCamera()
camera.start_preview()
camera.brightness=70
camera.contrast=50
sleep(5)
camera.capture('/home/pi/Desktop/max.jpg')
camera.stop_preview()
```

Esiste anche la possibilità di creare effetti, come ad esempio lo scambio di colori con il comando ***camera.image_effect='colorswap'*** .

Gli effetti disponibili sono i seguenti: ***negative, solarize, sketch, denoise, emboss, oilpaint, hatch, gpen, pastel, watercolor, film, blur, saturation, colorswap, washedut, posterise, colorpoint, colorbalance, cartoon, deinterlace1, deinterlace2,*** oppure ***none*** (di default).

Ad esempio:

```
from picamera import PiCamera
from time import sleep
camera=PiCamera()
camera.start_preview()
camera.image_effect='colorswap'
sleep(5)
camera.capture('/home/pi/Desktop/colorswap.jpg')
camera.stop_preview()
```

Con il seguente codice vengono mostrati tutti gli effetti:

```
from picamera import PiCamera
from time import sleep
camera=PiCamera()
camera.start_preview()
for effect in camera.IMAGE_EFFECTS:
    camera.image_effect=effect
    camera.annotate_text="Effect: %s" % effect
    sleep(5)
camera.stop_preview()
```

Con il comando ***camera.awb_mode*** si può invece regolare il bilanciamento del bianco, applicando diversi effetti con le seguenti opzioni: ***off, auto, sunlight, cloudy, shade, tungsten, fluorescent, incandescent, flash*** ed ***horizo***n. Di default il comando è impostato in auto.

```
from picamera import PiCamera
from time import sleep
camera=PiCamera()
camera.start_preview()
camera.awb_mode = 'sunlight'
sleep(5)
camera.capture('/home/pi/Desktop/sunlight.jpg')
camera.stop_preview()
```

Come nel caso precedente si possono visualizzare tutti gli effetti con il comando ***camera.AWB_MODES***

Analogo discorso per il comando ***camera.exposure_mode***, che regola l'esposizione, i possibili effetti sono; ***off, auto, night, nightpreview, backlight, sportlight, sports, snow, bech, verylong, fixedfps, antishake*** e ***fireworks***. Anche in questo caso di default il comando è impostato su ***auto***.

```
from picamera import PiCamera
from time import sleep
camera=PiCamera()
camera.start_preview()
camera.exposure_mode = 'beach'
sleep(5)
camera.capture('/home/pi/Desktop/beach.jpg')
camera.stop_preview()
```

Come nei casi precedenti si possono visualizzare tutti gli effetti con il comando ***camera.EXPOSURE_MODES.***

Al seguente link (raggiungibile dal precedente) ci sono 4 esempi di come applicare la camera della Raspberry insieme alle GPIO.
https://projects.raspberrypi.org/en/projects/getting-started-with-picamera/9

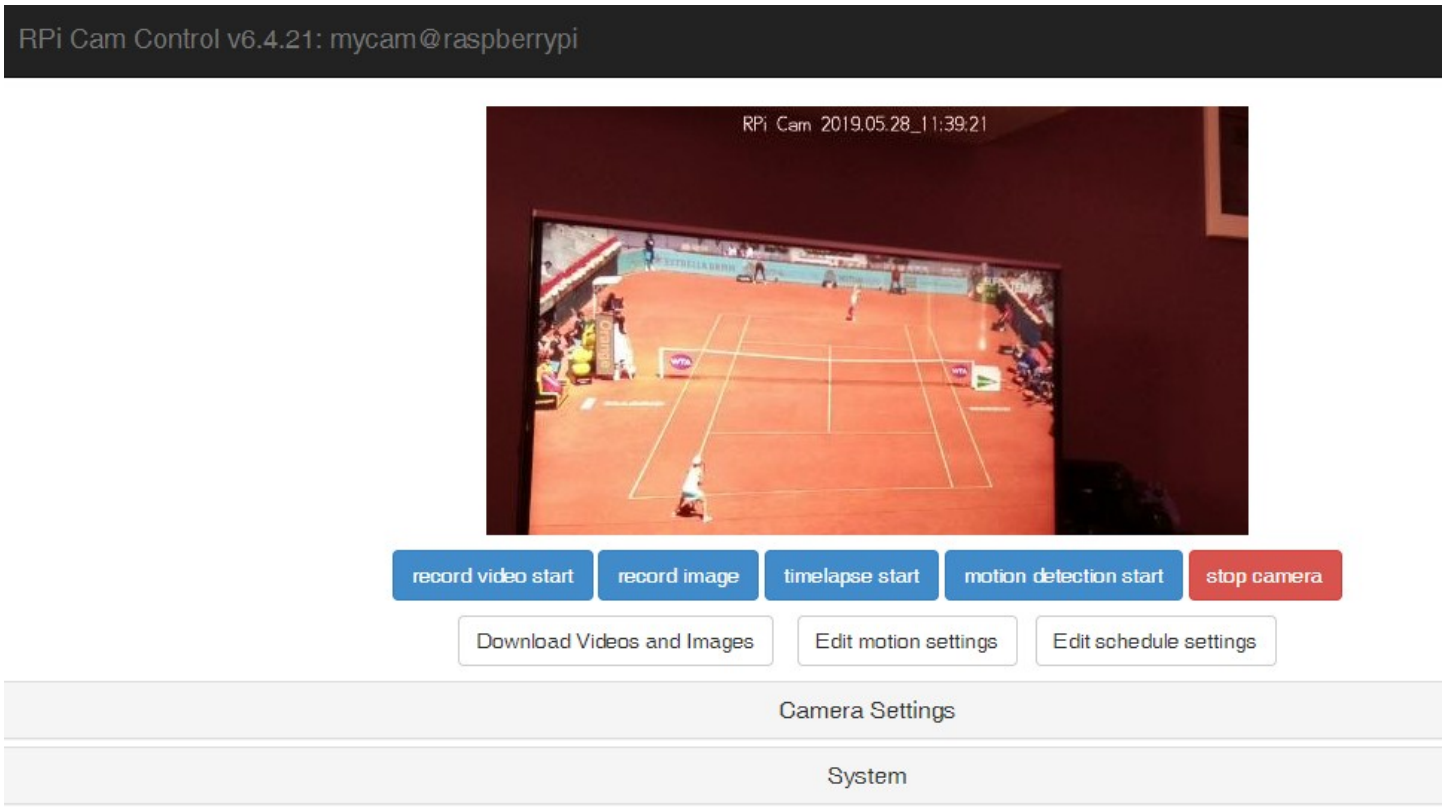
Tutte le informazioni dettagliate sui comandi sono invece disponibili al seguente link:
https://picamera.readthedocs.io/en/release-1.13/

WEB CAM INTERFACE

Un interessante pacchetto software è invece ***Rpi_Cam_Web_Interface.***
Installando questo pacchetto sulla nostra Raspberry, otteniamo una Web Cam raggiungibile dall'indirizzo IP della Raspberry con svariate possibilità.
Il link dove trovare tutta la spiegazione e documentazione è il seguente: https://elinux.org/RPi-Cam-Web-Interface . Bisogna però ricordarsi che utilizzando questo pacchetto software non si potranno utiizzare i comandi precedentemente visti, pertanto occorre disattivare il software installato prima di accedere alla Camera in altro modo.
Sinteticamente (ma è tutto descritto nel sito) occorre innanzitutto eseguire un aggiornamento del sistema con i comandi: ***sudo apt-get update*** e ***sudo apt-get dist-upgrade.***
Successivamente occorre scaricare da github il pacchetto software ed installarlo sulla Raspberry.
La sequenza dei comandi è la seguente:
git clone https://github.com/silvanmelchior/RPi_Cam_Web_Interface.git
cd RPi_Cam_Web_Interface
./install.sh

Successivamente per disattivare il software occorre digitare nella cartella ***Rpi_Cam_Web_Interface*** il comando ***sudo ./stop.sh*** e per riavviare ***sudo ./start.sh.***

Il risultato che avremo sarà il seguente, e cioè una pagina web dove poter visualizzare le immagini riprese dalla nostra camera, e poter attivare anche delle funzioni di motion detection, o time lapse.



OPENCV & RASPBERRY

Link di riferimento

https://docs.opencv.org/3.4/d2/de6/tutorial_py_setup_in_ubuntu.html

OpenCV (*Open Source Computer Vision Library*) è una libreria software multipiattaforma nell'ambito della visione artificiale in tempo reale. Il linguaggio di programmazione principalmente utilizzato per sviluppare con questa libreria è il C++, ma è possibile interfacciarsi anche attraverso il C, Python e Java.
Tramite questa libreria potremo realizzare diverse applicazioni con la nostra Raspberry.

Per installare questa libreria ci sono due metodi il primo è quello di installare la libreria direttamente dai file bin precompilati mediante i seguenti comandi:

sudo apt-get update
sudo apt-get install python-opencv

In questo modo però non abbiamo la certezza di avere l'ultima versione di OPENCV installata, ma nella gran parte dei casi questa opzione è sufficiente per le nostre applicazioni.

Il secondo metodo invece prevede la compilazione dei file sorgenti, questa è un operazione molto lunga e mediamente complessa, ma seguendo i passi successivi facendo attenzione alla corretta sintassi dei comandi, si può tranquillamente riuscire ad installare le Open CV e tutti gli stumenti necessari.
Quanto segue è estratto da un articolo sulla rivista elettronica in, ma in rete troviamo diversi tutorial con gli identici passi da seguire (come ad esempio https://www.learnopencv.com/install-opencv-4-on-raspberry-pi/)

Cominciamo l'installazione, buon lavoro e mi raccomando massima attenzione alla sintassi dei comandi.

1) Innanzitutto aggiorniamo il sistema con i seguenti comandi:

sudo apt-get update

sudo apt-get upgrade

2) Installazione degli strumenti di sviuppo e dell'ultima versione di Cmake. Questo è un software libero multipiattaforma per eseguire la compilazione di programmi scritti in linguaggio C e C++.

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

3) Installiamo ora delle librerie per il trattamento delle immagini e del video

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

4) Installiamo il toolkit GTK (GIMP ToolKit) per l'interfaccia grafica. Questo è un insieme di stumenti per la creazione di interfacce grafiche.

sudo apt-get install libgtk-3-dev

sudo apt-get install libcanberra-gtk*

5) Installiamo due due pacchetti che contengono ottimizzazioni numeriche per OpenCV:

sudo apt-get install libatlas-base-dev gfortran

6) Installiamo gli strumenti di sviluppo per Python 3:

sudo apt-get install python3-dev

7) Scarichiamo gli archivi di OpenCV4 nella cartella Home. Tutte le librerie di OpenCV 4 sono disponibili in due repository github chiamali opencv e opencv_contrib. Il repository contrib contiene moduli aggiuntivi creati dagli utenti. Ecco i comandi da digitare per tornare alla cartella Home e scaricare i due reposìtory.

cd /home /pi

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

8) Decomprimiamo i due file .zip scaricati nella cartella home.

unzip opencv.zip

unzip opencv_contrib.zip

Verranno così create le directory opencv-4.0.0 e opencv_contrib-4,0,0. Per motivi pratici si consiglia di rinominare le cartelle come opencve opencv_contrib:

mv opencv-4.0.0 opencv

mv opencv_contrib-4.0.0 opencv_contrib

9) Occorre ora configurare un **ambiente virtuale di Python 3**. Un'ambiente virtuale permette di creare spazi indipendenti dove far girare diversi programmi (in questo caso in Pyhton) che possono utilizzare anche moduli diversi. Con un ambiente virtuale (Virtual Environment) si può lavorare con più progetti tra di loro indipendenti che possono utilizzare moduli con versioni differenti, inoltre si possono installare moduli senza i privilegi di root con i relativi vantaggi di sicurezza. In pratica viene creata una cartella che conterrà i file necessari al funzionamento del'ambiente ed una copia dei file binary di Python.
Il **Python Package Index (PyPI)** è un repository che contiene package scritti in Python. E' possibile accedere ai package del Python Package Index tramite browser (https://pypi.orf) o tramite un tool chiamato ***pip***, che installeremo con i comandi seguenti:

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

sudo python3 get-pip.py

Installiamo virtualenv e virtualenvwrapper che consentono di creare ambienti virtuali di Python 3:

sudo pip install virtualenv virtualenvwrapper

sudo rm -rf ~/get-pip.py ~/.cache/pip

Per completare è necessario aggiornare il file -/.profile, usando questi semplici comandi echo:

echo "export WORKON_HOME=$HOME/.virtualenvs" >> ~/.profile

echo "export VIRTUALENVWRAPPER_PYTH0N=/usr/bin/python3" >> ~/.profile

echo "source /usr/local/bin/virtualenvwrapper.sh" >> ~/.profile

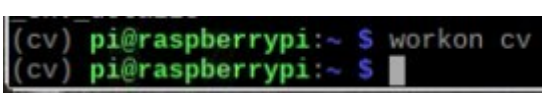
Le aggiunte al profilo indicano il percorso della cartella di lavoro virtualenvs, creata dallo strumento virtuolenv nella Home e il percorso dello script virtualwrapper che si trova nella cartella /usrlbin. Una volta aggiornato il profilo, basta attivarlo con il comando seguente:

source ~/.profile

Ora creiamo un ambiente virtuale chiamato ***cv*** con il seguente comando

mkvirtualenv cv -p python3

Si può verificare che siamo nell'ambiente ***cv*** con il comando ***workon cv*** prima del nostro prompt dei comandi troveremo la scritta ***cv*** che ci indica che siamo nell'ambiente virtuale.



10) OPENCV richiede un pacchetto software Python chiamato ***NumPy***, installiamolo utilizzando **pip** con il seguente comando: ***pip install numpy***

11) Compilazione di ***OPENCV*** con ***Cmake.*** La libreria software andrà ora compilata utilizzando CMake. Entriamo nella cartella ***opencv*** della home ed al suo interno creiamo una sottocartella ***build*** ed entriamo in quest'ultima con i seguenti comandi

cd ~/opencv oppure ***cd home/pi/opencv***
mkdir build
cd build

A questo punto eseguiamo ***CMake*** per ottenere una versione dea build di tipo release.

```
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_VFBV3=ON \
        -D BUILD_TESTS=OFF \
        -D OPENCV_ENABLE_NONFREE=ON \
        -D INSTALL_PYTHON_EXAMPLES=OFF \
        -D BUILD_EXAMPLES=OFF ..
```

Il backslah a fine linea serve a continuare il comando con un a capo.

Prima di iniziare la compilazione, occorre aumentare lo spazio di swap per evitare che la compilazione si interrompa per l'esaurimento della memoria. Occorre perciò modificare temporaneamente il file di swap che si trova nel percorso /etc/dphys-swopfile.

sudo nano /etc/dphys-swapfile

All'interno del file bisogna commentare la riga **# CONF-SWAPSIZE=100**
scrivendone una nuova con un valore di 2048 **CONF-SWAPSIZE=2048**

Senza questa modifica è molto probabile che la scheda si blocchi durante la compilazione.
Una volta modificato il file di swap, bisogna fermare e riavviare il servizio di swap:

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

Ora è possibile compilare con l'opzione -j4 che prevede l'utilizzo di 4 core, se ci sono errori di compilazione si può ripetere l'operazione senza l'opzione -j4

make -j4

A questo punto ci si può anche rilassare guardando un bel film visto il tempo necessario per la compilazione……...

12) Installazione di OPENCV con i seguenti comandi:

sudo make install
sudo ldconfig

13) Ripristinare il file di swap come fatto precedentemente:

sudo nano /ètc/dphys-swapfile

Modificare il valore e salvare

#CONF-SWAPSIZE=2048
CONF-SWAPSIZE=100

Fermare e riavviare il servizio di swap

sudo /etc/init.d/dphys-swapfile stop
sudo /etc/init.d/dphys-swapfile start

Per fare in modo che OPENCV riconosca i pacchetti nel'ambiente vrtuale di Python 3, occorre creare un collegamento simbolico di OPENCV con la directory dei pacchetti dell'ambiente virtuale.
Per fare questo occorre entrrare nella directory dei site-packages dell'ambiente virtuale e inkare la libreria cv2.so con i seguenti comandi:

cd ~/.virtualenvs/cv/lib/python3.5/site-packages/
ln -s /usr/local/python/cv2/python-3.5/cv2.cpython-35m-arm-linux-gnueabihf.so cv2.so
cd ~

A questo proposito controllare che nella directory ***~/.virtualenvs/cv/lib/python3.5/site-packages*** sia presenti il link alla libreria cv2.so.

14) Verifichiamo ora l'installazione di OPENCV, rimanendo all'interno dell'ambiente virtuale digitiamo i seguenti comandi:

python
>>> import cv2
>>> cv2.__version__
'4.0.0'
>>> exit()

In pratica si entra nell'interprete python, si importa la libreria e si verifica la versione, con exit si esce dall'interprete python.

N.B.

Al riavvio del sistema occorre ogni volta riattivare l'ambiente virtuale con i comandi:

source ~/.profile
workon cv

Per usare l'interfaccia IDLE di Python 3 all'interno dell'ambiente virtuale digitare: ***python -m idlelib.idle***

Per uscire dall'ambiente virtuale scrivere il comando ***deactivate***

A questo punto abbiamo installato un importante strumento per poter realizzare svariati progetti dal motion detector, al riconoscimento facciale.

Lo scopo di questa dispensa, come delle altre realizzate sulla Raspberry, non è quello di presentare un progetto, ma quello di fornire le informazioni basilari sull'utilizzo della scheda.
Una volta conosciute le possibili librerie da utilizzare, bisognerà studiare le funzioni disponibili ed applicarle secondo le proprie esigenze.

Riguardo alla libreria OpenCV ai seguenti link possiamo trovare tutte le informazioni necessarie per l'utilizzo.
https://docs.opencv.org/3.4/
https://docs.opencv.org/2.4/
https://docs.opencv.org/2.4/opencv_tutorials.pdf

In rete possiamo trovare diversi progetti che utilizzano la cam della Raspberry, come i due descritti inizialmente riguardanti il motion detector ed il riconoscimento facciale, credo pertanto sia inutile ripetere ciò che altri hanno fatto in questa dispensa, anche per evitare di dilungarci troppo.
Giunti a questo punto, consiglio pertanto di provare a mettere in pratica un qualsiasi progetto che utilizza la camera della Raspberry.
Posso per questo consigliare qualche link tra i vari disponibili in rete, dove vengono forniti i codici e tutte le informazioni necessarie.

http://www.extremegeneration.it/riconoscimento-facciale-raspberry-pi-opencv/
https://www.hackster.io/brendan-lewis/detect-motion-with-opencv-no-pir-sensor-needed-bbeacf
https://www.pyimagesearch.com/2015/05/25/basic-motion-detection-and-tracking-with-python-and-opencv/

Digitando su google "motion detector with raspberry" o "facial recognition with raspberry", o anche la traduzione in italiano delle due frasi, verranno fuori altri link e progetti da spulciare e studiare.
Buon lavoro!