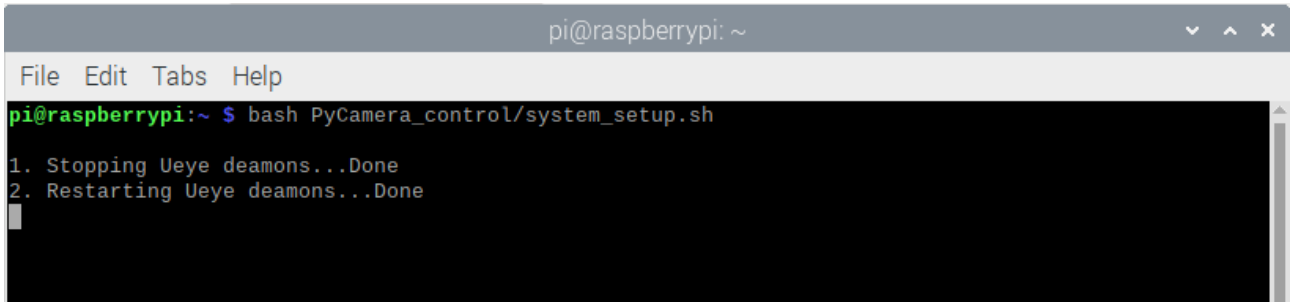
User manual - PyCamera control

The software can be launched by the bash file 'system_setup.sh':

```
bash PyCamera_control/system_setup.sh
```

```
1  #!/bin/bash
2
3  sleep 15s
4
5  echo -n 1. Mounting USB storage device...
6  sudo mount /dev/sda1 /media/usb
7  echo Done
8
9  sleep 5s
10
11 echo -n 2. Stopping Ueye deamons...
12 sudo systemctl stop ueyeusbdrc
13 sudo systemctl stop ueyeethdrc
14 echo Done
15
16 sleep 20s
17
18 echo -n 3. Restarting Ueye deamons...
19 sudo systemctl start ueyeusbdrc
20 sudo systemctl start ueyeethdrc
21 echo Done
22
23 sleep 20s
24
25 echo -n 4. Running the PyCamera script...
26 python3 /home/pi/PyCamera_control/PyCamera.py
27 echo
28
```

The bash file restarts Ueye USB and Ethernet deamons (lines 5-16 for the Desktop version, lines 11-21 for the Lite version) and then start the python script for data acquisition.
When the bash file is executed, in command line these messages are visualized (concerning the Ueye deamons restart):

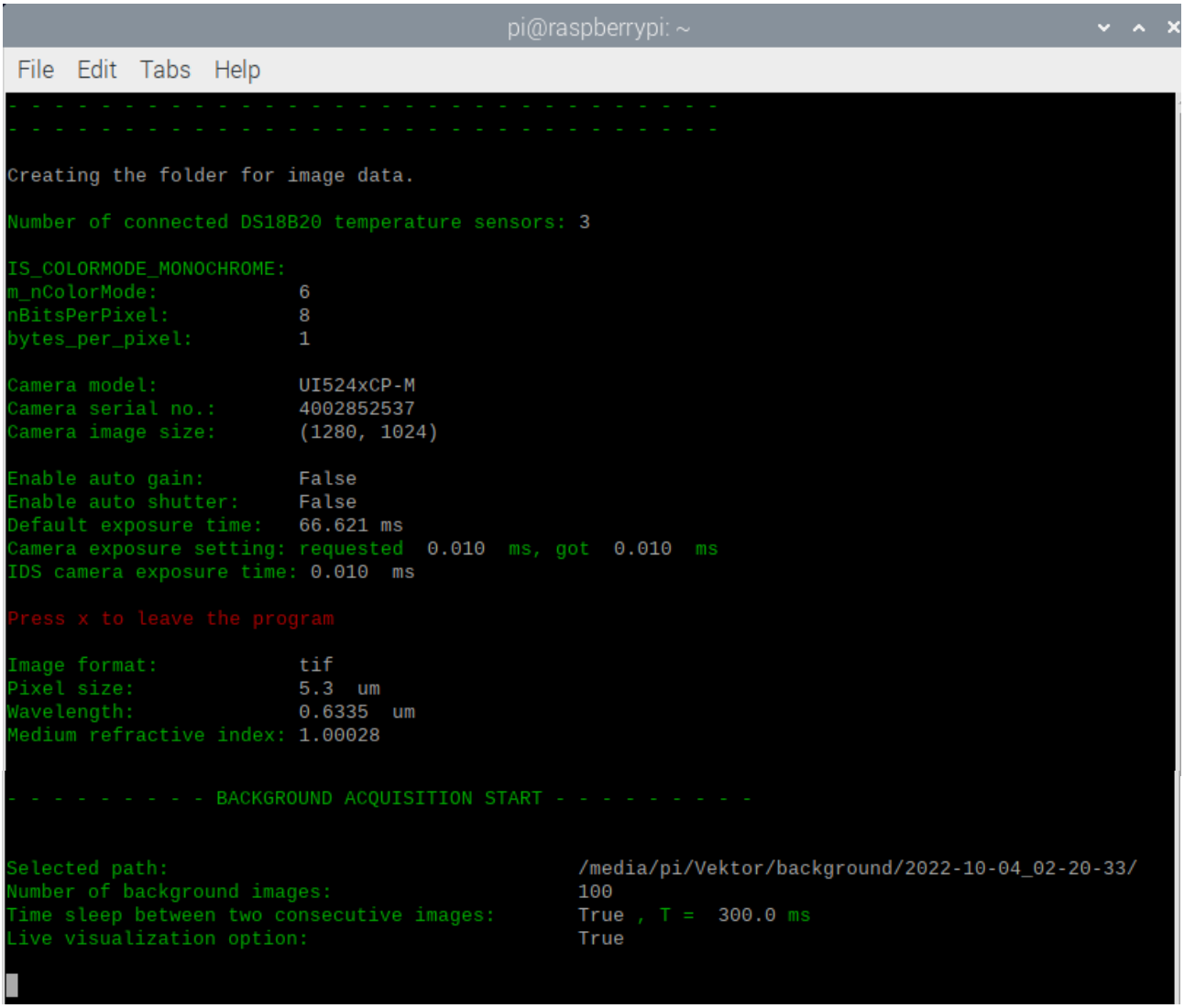


In the following the script is described.

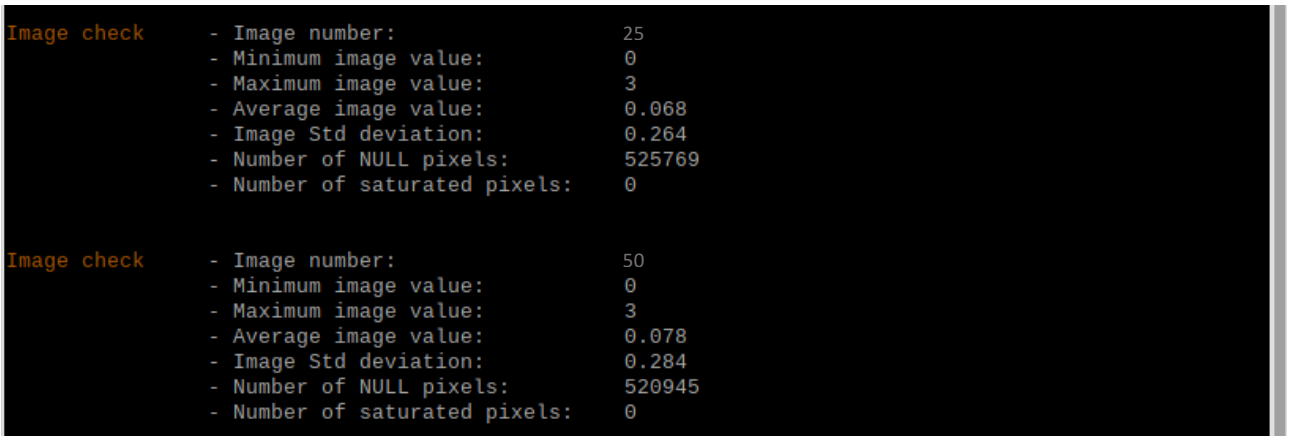
When the python script is first launched the software waits for an input on RPi GPIO; the GPIO settings will be presented later on. To start the acquisition set the value of GPIO 14 to HIGH (=1) by the RaspController application.

When the data acquisition starts the command line is cleared and the software progress looks like this: some general information about the IDS camera are retrieved, the optimal exposure time is set, the path to

save images and data are created, a first temperature measurement is performed and the background acquisition starts (250 images).



Every 25 background images a short monitoring is performed: on the shell the user can see the image average value and standard deviation, the maximum and minimum pixel value of that image and the number of null and saturated pixels.



When this acquisition is completed, the system waits for a user input before starting the hologram images acquisition: set the value of GPIO 15 to HIGH (=1) by the RaspController application

```
- - - - - - - - - - BACKGROUND ACQUISITION END - - - - - - - - - -

Waiting before starting the measurement, press <k> to proceed.
```

```
- - - - - - - - - - BACKGROUND ACQUISITION END - - - - - - - - - -

Waiting before starting the measurement, press <k> to proceed.
 k

- - - - - - - - - - DATA ACQUISITION START - - - - - - - - - -

Selected path:                                  /media/pi/Vektor/data/2022-10-04_02-20-33/
Time sleep between two consecutive images:      True , T =  300.0 ms
Live visualization option:                      True

Temperature monitoring every 50 images [°C] and image check
```

During these measurements as well an image monitoring is carried out.
Every 50 images the RPi grab the temperature values from the DS18B20 sensors and computes the same image parameters presented before (average, standard deviation, maximum, minimum, number of null and saturated pixels).

```
- - - - - - - - - - DATA ACQUISITION START - - - - - - - - - -

Selected path:                                  /media/pi/Vektor/data/2022-10-04_02-20-33/
Time sleep between two consecutive images:      True , T =  300.0 ms
Live visualization option:                      True

Temperature monitoring every 50 images [°C] and image check

1. T (RPi) = 52.1
2. T (Ueye) = 24.8
3. T (levitator) = 999.0
4. T (laser) = 19.8

Image check     - Image number:                 51
                - Minimum image value:          0
                - Maximum image value:          3
                - Average image value:          0.065
                - Image Std deviation:          0.258
                - Number of NULL pixels:        527466
                - Number of saturated pixels:   0
```

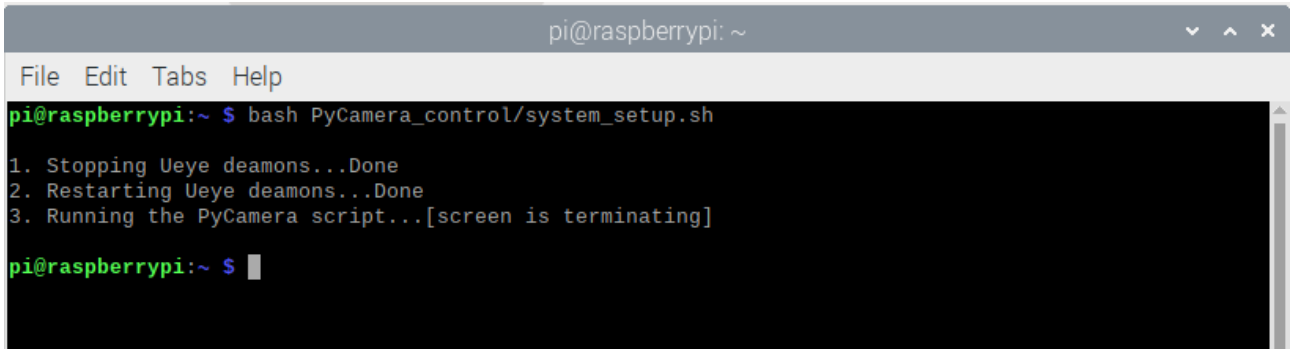
At this point the software goes on until a 'stop' signal is provided, setting the value of GPIO 17 to LOW (=0) by the RaspController application

When the acquisition is stopped the software stays silent and waits for the HIGH value of GPIO 17.

To exit the program completely the user must set both the GPIO 17 (before) and GPIO 14 (after) values to LOW (=0).

To abort the software execution and operate a safe quit it is required to set the GPIO 23 value to HIGH (=1): the execution is stopped, and all the devices are correctly disconnected.

In both cases, when the program is closed, the initial command line interface is restored and the message "[screen is terminating]" appears.



Notes on Raspberry pinouts

In order to visualize the Raspberry Pi3 pinout, type

```
pinout
```

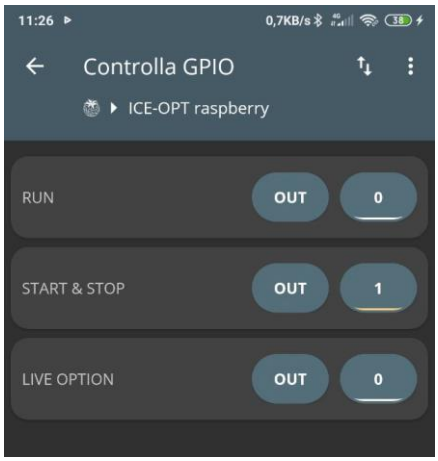
in RPi command line.

Raspberry GPIO values

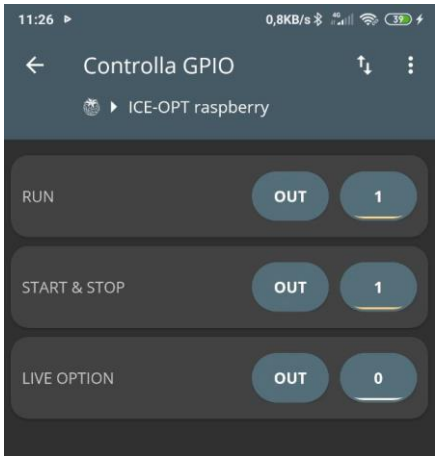
The Raspberry GPIO implied in the above software are:

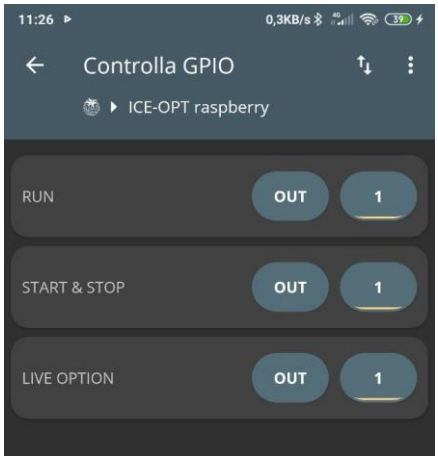
- GPIO 14: run application – created folders and start background images acquisition
- GPIO 15: acquire – evaluate background average variance and starts holograms recording
- GPIO 17: start/stop – regulates start and stop of holograms acquisition
- GPIO 23: sae quit procedure

The GPIO values can be controlled and modified by the RaspController application.
Here are some examples:

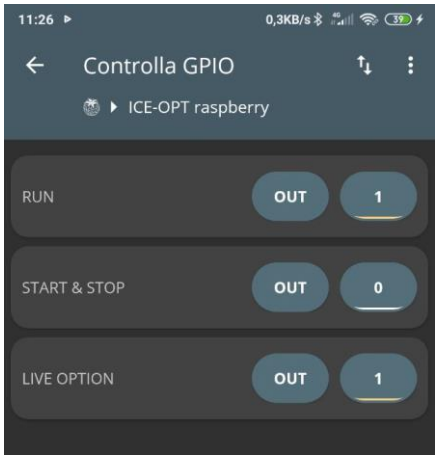


- The software started correctly and is waiting for the GPIO value HIGH to start image acquisition.





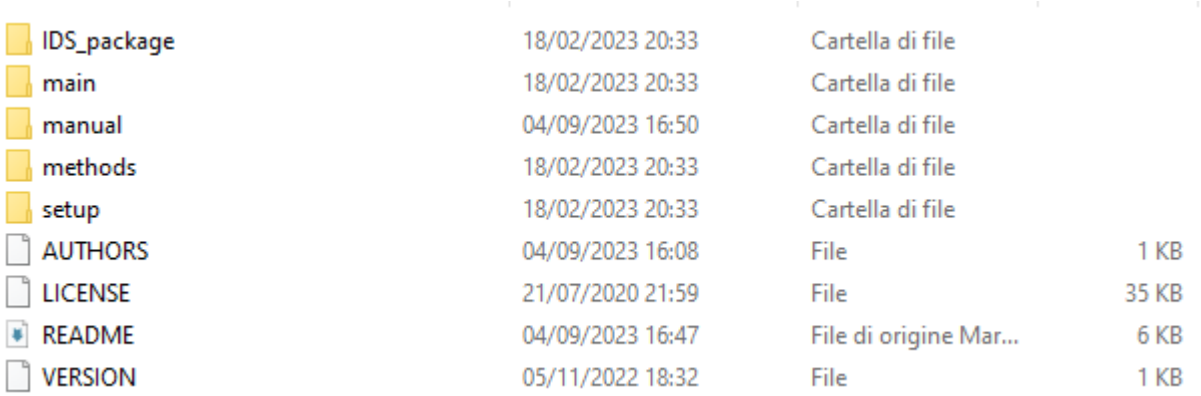
- The software is running (both GPIO 14 and 17 are HIGH); on the left, the LIVE flag is turned off, while on the right the LIVE option is activated.



- The software is still running (GPIO 14 is HIGH) but it has been stopped by the user and is waiting for an input to restart the task (or, alternatively, to completely stop the acquisition by setting GPIO 14 to LOW).

PyCamera package

The PyCamera control python package is the same between Desktop and Lite version. The differences lie in some functions and the way the interactive software is controlled.



Here is a short classification:

Folder: main	
PyCamera.py	Main script for image acquisition, sets the main parameters and manage the online/offline acquisition mode
system_setup.sh	Bash file to rule the system start-up and launch the main acquisition software
Folder: methods	
IDS_camera.py	Script defining functions to control and set some parameters of the IDS Ueye camera (such as the exposure time) ; recalled in PyCamera.py
online_acquisition.py	Script that defines the online acquisition mode; recalled in PyCamera.py
offline_acquisition.py	Script that defines the offline acquisition mode; recalled in PyCamera.py
utils.py	Auxiliary script
Folder: IDS_package	
Ids-software-suite-linux-arm64-4.96.1-….	Zip file to install all the required IDS and Ueye features and libraries
Folder: setup	
requirements.txt	Text file containing all the required Python libraries to be installed
setup.py	Python script to install dependencies from 'requirements.txt' text file
Folder: manual	
Some docs on Rasperry OS setup and software installation	

All these files are fully commented, for further information you can refer to these.