Drone Hand Gesture Control

MEHIDI Mohammed
GEII Department, ENST
Algiers-Algeria
m_mehidi@enst.dz

Abstract— ***Human-machine interfaces are an important element in modern industry. Since man has become able to control several machines without any effort, this document presents a project which aims to firstly implement hand gesture movements. to control a quadcopter-type drone (AR DRONE 2.0) without using a control pad or a keyboard. Secondly developing an algorithm that makes the drone follow the hand of the user, and lastly, retrieve the information's about the path, speed, battery, and the Hight of the drone…***

Keywords— hand gesture,Hand Recognition,Drone , control ,mouvements,capture.

I. INTRODUCTION

The Gesture Recognition System has received a lot of attention in the past few years due to the diversity of its applications and the ability to interact with the machine efficiently through human-machine interaction. Hand and face gestures have become a second language that almost complements many speeches and is used in so many controls fields such as smart TVs and smartphones (screens shots using hand gestures).

our contribution is to implement gesture recognition for the control of the AR drone 2.0 type drone where we have set two criteria to be respected: the flexibility of movement and stability of the drone. 11 drone travel modes must be commanded also retrieving the trajectory followed by the drone.

The application was developed with the python language because it greatly facilitates the phase of communication with the drone.

The remainder of this paper is organized as follows. The first section gives an overview description of gesture recognition in general. The second section presents the used tools while working on this project with a brief explanation of each one of them. The third section describes the different gestures used to control the movement of the drone ensure a flexible drone movement and explaining the drone hand Follow algorithm while the last section concludes this work showing all results and discusses the possible future work.

II. GESTURE RECOGNITION

In this section, we will introduce the computer technology that allows analyzing a gesture captured by means of a computer camera to transcribe it under the form of data that can be used by a machine

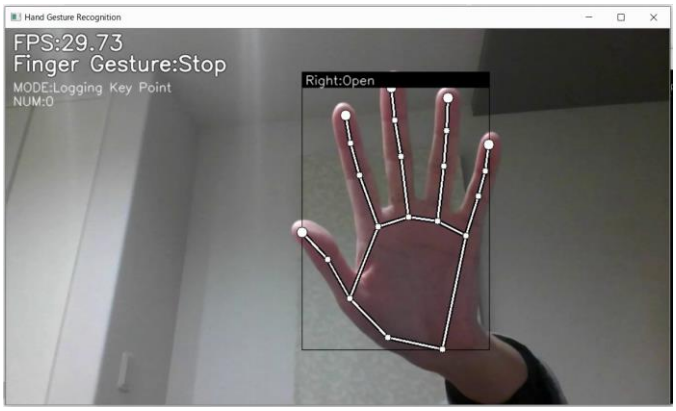

Fig. 1: Gesture recognition [1]

Gesture recognition refers to all the processes used to analyze a scene, i.e., the capture of gestures (for example using a camera or glove equipped with sensors), segmentation, assessment of poses, and interpretation strictly speaking. The following subsection presents the general structure of the gesture recognition described in [2], [3]; although the authors describe the process capture and interpretation of bodily (physical) gestures, their analysis highlights issues valid for any type of gesture, including hand and hand gestures

Pattern recognition is the information-saturated transfer stage of our world (visual, auditory,) towards the world of the machine. This process does not consist only to convert an analog information source into binary, it is a question of interpreting the source into exploitable.

III. THE USED TOOLS ON THE GESTURE CONTROL PROJECT

first of all, this project is based on a real-time concept, so we will explain this concept; A real-time application can be defined as a system whose behavior depends not only on the accuracy of the treatments performed but also on the time when the results of these treatments are produced [4]

The objectives to be achieved during the design:

- Predictable system.
- Logical determinism: the same input applied to the produced system.
- To have the same exit every time.
- Temporal determinism: time constraints are respected.
- Reliability: the system (software, hardware) meets availability constraints.

1. PyCharm development interface

PyCharm is an integrated development environment used to program in Python, it allows code analysis and contains a graphical debugger.

2. Python Libraries used:

✓ The NumPy library:

NumPy is a library for the Python programming language, intended for manipulating multidimensional matrices or arrays as well as functions mathematics operating on these tables.

Specifically, this free and open-source software library provides multiple functions allowing, in particular, to create a table directly from a file or from the opposite of saving an array in a file, and manipulating vectors, matrices, and polynomials.

NumPy is the base of SciPy, a grouping of Python libraries around computation scientists. [5]

✓ Open CV Library:

OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code. [6]

✓ MediaPipe Library

Media Pipe is a cross-platform framework for building multimodal applied machine learning pipelines MediaPipe is a framework for building multimodal (eg. video, audio, any time series data), cross-platform (i.e Android, iOS, web, edge devices) applied ML pipelines. With MediaPipe, a perception pipeline can be built as a graph of modular components, including, for instance, inference models (e.g., TensorFlow, TFLite) and media processing functions. [7]

✓ Parrot drone 2.0

The 2.0 model was introduced in May 2012, not only does it keep track of its position, but can be used to create 3D models of each movement, via the application of AR Drone. It can also store videos directly. With its 4 GB of built-in flash memory and can hold more hours of HD video. This drone has a high-density 1500 mAh battery. Currently, the AR Drone 2.0 is not capable of only five to 10 minutes of flight, even after a full charge

✓ Communication with AR Drone 2.0

Communication with the drone is done by following the following steps:

a) AR. Drone creates a WIFI network called adrone2_xxx 192.168.1.1

b) The client device connects to the network and requests an IP address from the DHCP server of the drone

c) The AR Drone DHCP server grants the client an IP address, which is 192.168.1.2

d) The client device can start sending requests to the IP address of the drone AR and its service ports

✓ Communication ports for controlling the drone

The communication ports used to control the AR Drone 2.0 are:

a) The control and configuration of the drone are done by sending commands to the UDP port "5556".

b) Navdata Information (Data Navigation) such as its status, position, speed, etc., are sent by the drone to its client on the UDP port "5554"

c) A video stream (Video Stream) is sent by the AR drone to the client device on the TCP port 5555.

After the detection of the user's hands, several algorithms were adopted to convert gestural movements into drone movements

Each point in the hand detected is marked with a number from 0 to 20, these numbers are stored in a list (a vector) to use in the acknowledgment. The points are linked together by lines to build the shape of the main in a virtual interface, the figure below shows the result.

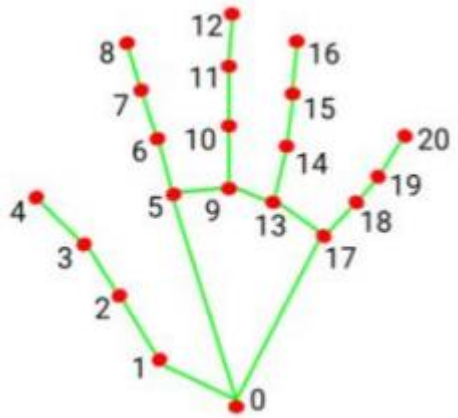

Fig. 2: Hand Landmarks [7]

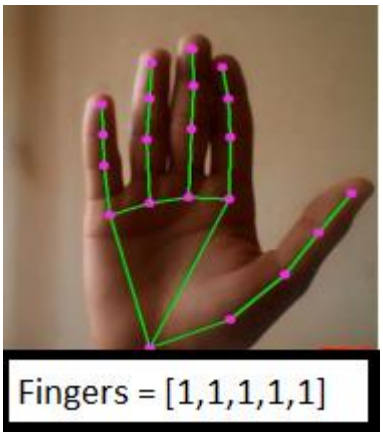

Fig. 4: gesture example

0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Fig. 3: Landmarks names [7]

From these points, we define algorithms to recognize the gesture applied by the user and give the output corresponding to this gesture, each finger has been referenced by a Boolean variable that indicates whether it is raised or not, for example

- FingerOne (is up) = Ture,
- FingerOne (is down) = False

A Fingers table is defined and contains the state of each finger, from this table, the gesture applied by the user is indicated.

A small example of the Fingers vector is explained in this picture

IV. THE DIFFERENT GESTURES USED TO CONTROL THE MOUVEMENT OF THE DRONE

To launch the drone (take-off), every five fingers must be raised means that the array fingers = [1,1,1,1,1], a Boolean variable "Fly" is defined to indicate that the drone is on the move. The following flowchart details the algorithm used.

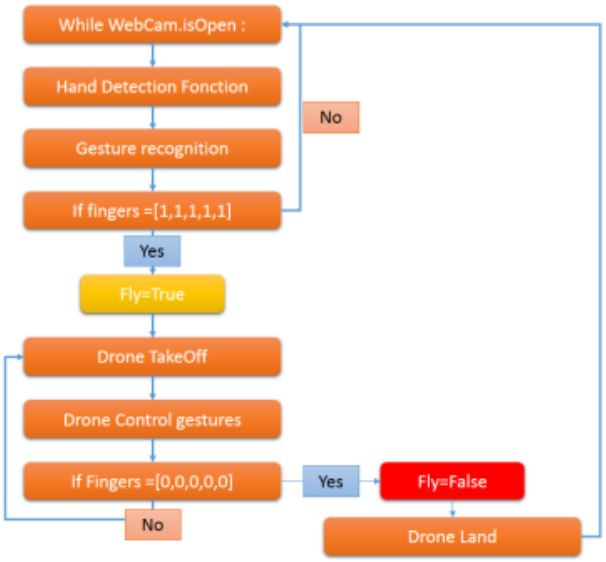

Fig. 4: organizational chart of drone launch algorithm

After the hands are detected, if the Fly variable is equal to 1, we move on to the execution of the other algorithms which depend on the applied gesture, Figure 5 shows an example execution of the take-off mode

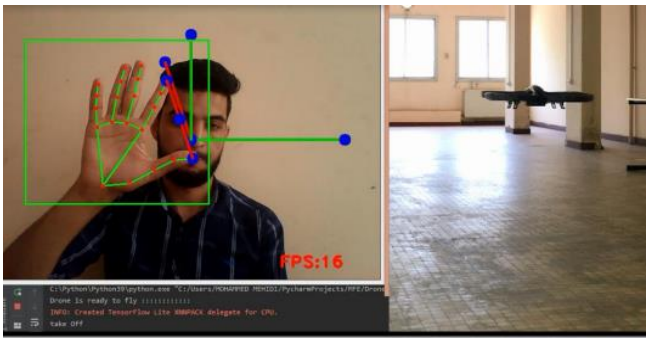

Fig. 5: an example execution of the take-off mode

Once in Take-off mode, a line is drawn between the thumb and forefinger (the line of command), modes.

1. First mode

This mode is used to control 4 movements of the drone and it is distinguished from the other mode by the three fingers (middle finger, ring finger, and little finger). When those fingers are up (fingers = [1,1,1,1,1]), the first mode is sectioned.

In this mode you can command the 4 movements (up / down / CW Rotation / CCW Rotation) depending on: the length, angle and inclination of the command line.

✓ The "up" command

the command line has an angle of 90 ° on the coordinate system and a control distance (between index finger and thumb) greater than 250, an example application is shown in figure 6:

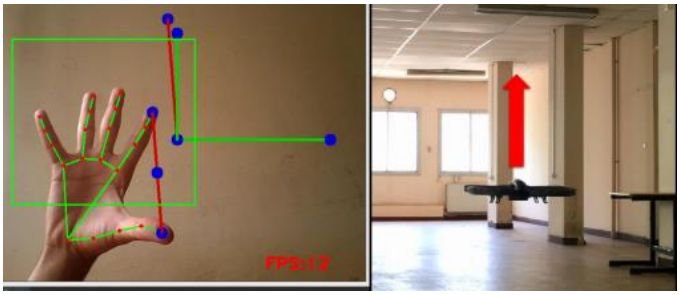

Fig. 6: the execution of the <<up>> command

✓ The "down" command

the command line has an angle of 90 ° on the reference and a control distance less than 65, an application example is shown in the figure 7

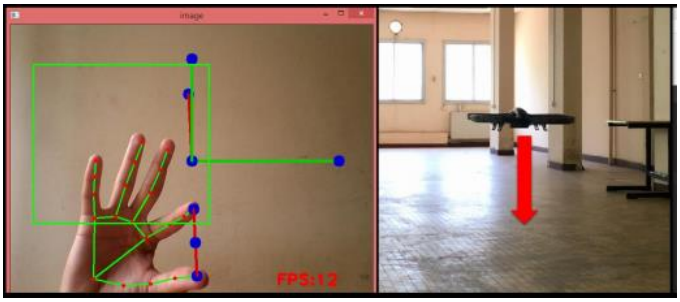

Fig. 7: the execution of the <<down>> command

✓ The "CW Rotation" command

The command line has an angle greater than 135 ° and a length greater than 170, the figure below illustrates the application of the command

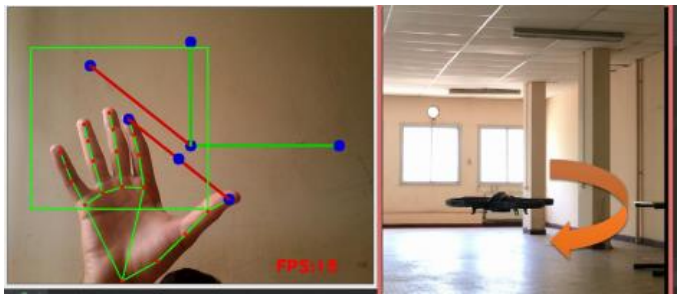

Fig. 8: the execution of the <<CW Rotation>> command

✓ The "CCW Rotation" command

The command line has an angle of less than 45 ° and a length greater than 170, Figure 9 illustrates the application of the command.

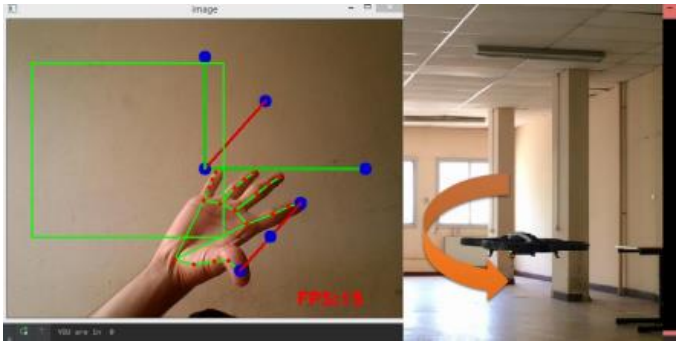

Fig. 9: the execution of the <<CCW Rotation>> command

2. Second Mode

In this mode, we are interested in the 4 remaining commands to control the drone movements. This mode is distinguished by the three fingers (middle finger, ring finger, and auricular close), the fingers vector, in this case, is equal to [1,1,0,0,0], therefore, the 4 movements (forward / backward / left shifting / right shifting) can be commanded depending on the length, angle, and inclinations of the command line.

✓ The "Forward" command

The command line has an angle of 90 ° on the reference and a control distance (between index finger and thumb) greater than 250, an example application is shown in figure 10:

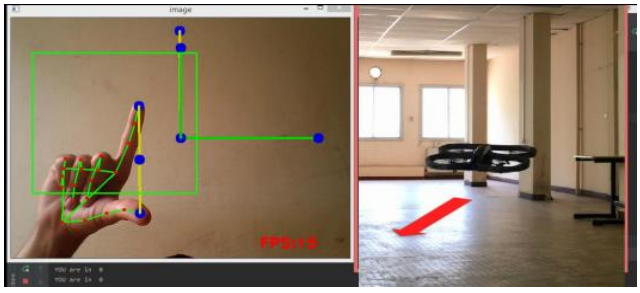

Fig. 10: the execution of the <<forward>> command

✓ The “Backward” command

The command line has an angle of 90 ° on the reference and a control distance of less than 65, an application example is shown in figure 11:

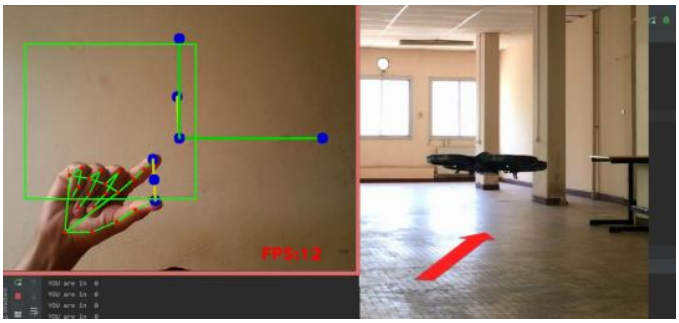

Fig. 11: the execution of the <<Backward>> command

✓ The "left shifting" command

The command line has an angle greater than 135 ° and a length greater than 170, the figure below illustrates the application of the command.

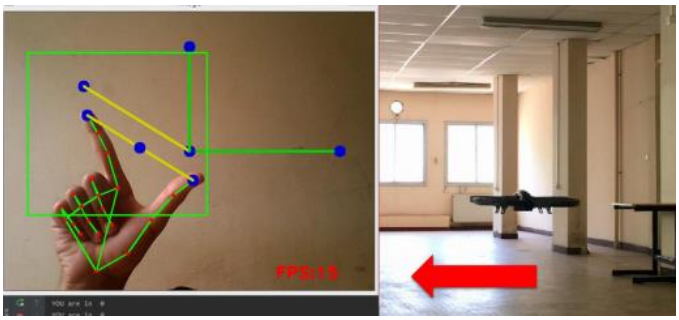

Fig. 12: the execution of the << left shifting >> command

✓ The "Right shifting" command

The command line has an angle of less than 45 ° and a length greater than 170, the figure below illustrates the application of the command

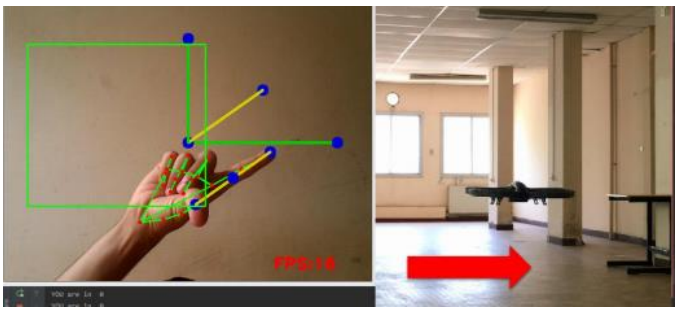

Fig. 13: the execution of the << Right shifting >> command

✓ The “Drone Hover” command

The two index and middle fingers are raised, and the distance between the two is less than 40, the figure below explains the situation

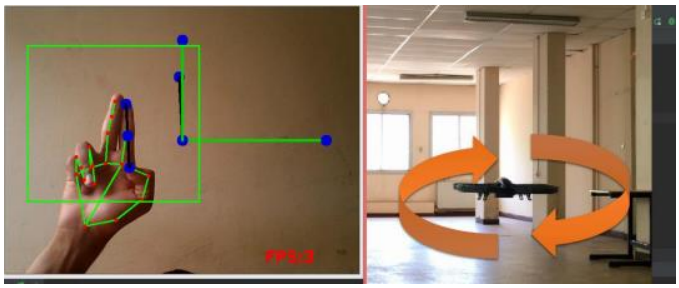

Fig. 14: the execution of the << Hover >> command

The flowchart below illustrates the two modes and the controls for each

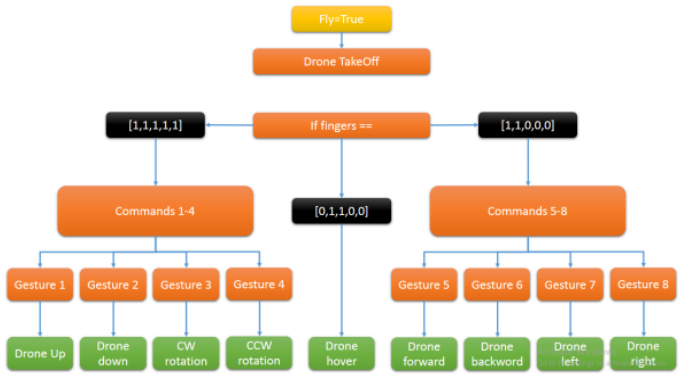

Fig. 15: organizational chart of the 2 mode of execution

IV.1. Retrieving the trajectory followed by the drone

When using the drone, it is necessary to recover the trajectory followed, for this we have defined an algorithm to have all the coordinates (x, y) of the drone in space and use them to draw a trajectory in a virtual interface. An example of a trajectory drawn when the drone moves to the left and afterward upwards is illustrated in figure 16:

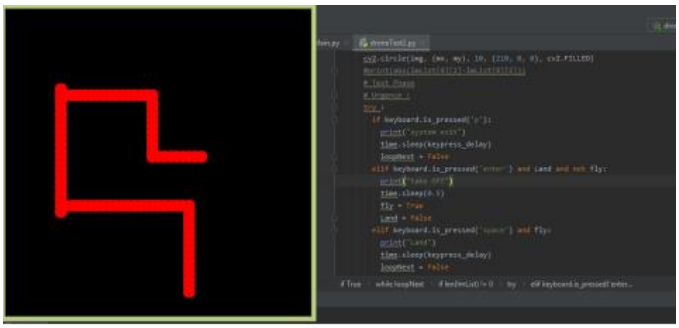

Fig. 16: Drone Draw path function result

V. THE DRONE FOLLOW HAND ALGORITHM

In this section, we have implemented an algorithm that allows the drone to follow the user's hand with the drone's front camera. The comfort zone is defined as being the area on which the drone is neither very close nor very far from the hand of the user

The figure below shows the result obtained when detecting the hand in mode Follow Hand:

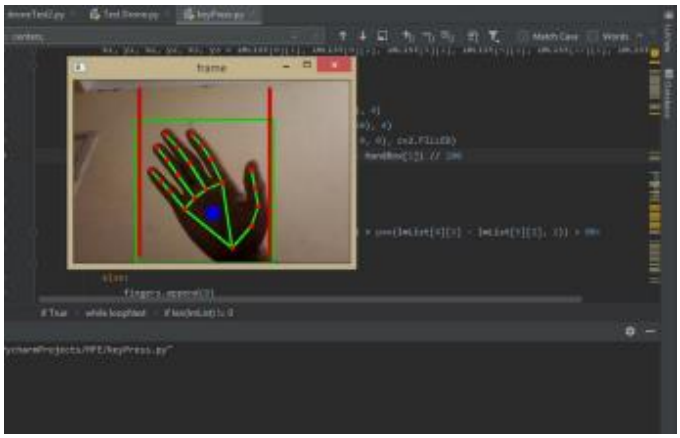

Fig. 17: Hand Detection in follow hand Algorithm

A circle is defined in the center of the hand is used to define the displacement longed for

If the diameter of this circle is very small, it means that the hand is far from the drone, therefore the drone must make a forward movement to reach the comfort zone, otherwise, if the diameter of the circle is very large, so the drone has to move back to reach the comfort zone, otherwise, the drone will stay in its position.

Now if the circle is located on the left side of the screen, then the drone should do a CW Rotation until you have the circle in the middle of the screen, the same for the case inverse, if the circle is located on the right side of the screen, then the drone must make a CCW Rotation

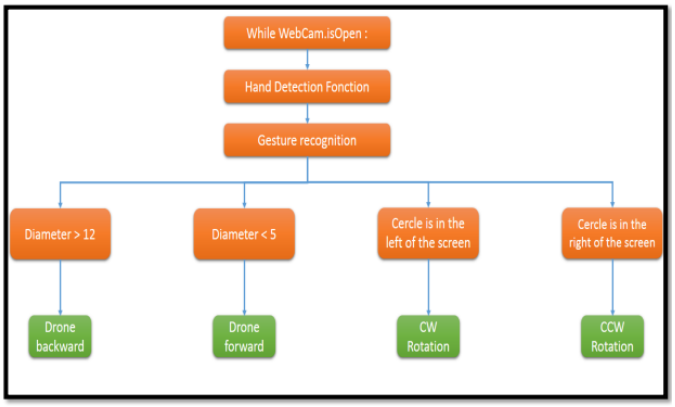

Fig. 18: organizational chart of Hand Detection in follow hand Algorithm

When testing this algorithm, it was found that when the hand is away from the drone, the detection is very bad this may be due to the quality of the drone camera. Through against for small distances, the code works perfectly. However, to do this, it is necessary to use flexibility coefficients and high-performance equipment to ensure the stability of the detection of hand movements

VI. CONCLUSION

In this chapter, we have presented an overview of the AR Drone 2.0, these features, and how to establish communication with this drone. We could control the AR Drone 2.0 by gestural movements, several algorithms have been implemented to have a flexible and stable operation. We also succeeded in recover the trajectory followed by the drone.

Finally, we were able to order the drone to follow the user's hand through the drone's front camera, but this presented its effectiveness for detection over short distances of the order of one to one and a half meters only, this is probably due to the quality of the front camera of the drone.

REFERENCES

[1]: hand gesture recognition Media pipe, GitHub, Retrieved from https://github.com/kinivi/hand-gesture-recognition-mediapipe last check 9/7/2021, 2:26 PM

[2]: Thomas B. Moeslund and Erik Granum. A survey of computer vision-based human motion capture. Comput. Vis. Image Underst., 81(3) :231–268, 2001.

[3]: Thomas B. Moeslund, Adrian Hilton, and Volker Kruger. A survey of advances in vision-based human

[4]:https://elearn.univtlemcen.dz/pluginfile.php/60327/mod_resource/content/2/Chapitre 1_Version_PDF.pdf last check 9/7/2021, 2:49 PM

[5]: «Scientific computing tools for Python — SciPy.org » on www.scipy.org (last check 9/7/2021).

[6]: About, OpenCV, Retrieved from https://opencv.org/about/ last check 9/7/2021, 2:54 PM

[7]: MediaPipe is a cross-platform framework for building multimodal applied machine learning pipelines, opensource. Google, Retrieved from https://google/projects/mediapipe last check check 9/7/2021, 2:59 PM