Colour detection using UAV for **Smart Agriculture**

B22RVP03

Guide: Dr. Y. Raja Vara Prasad

Team members:
Kamarthi Litheesh Kumar - S20190020218
TU Surya Sathvik - S20190020254

Outline:

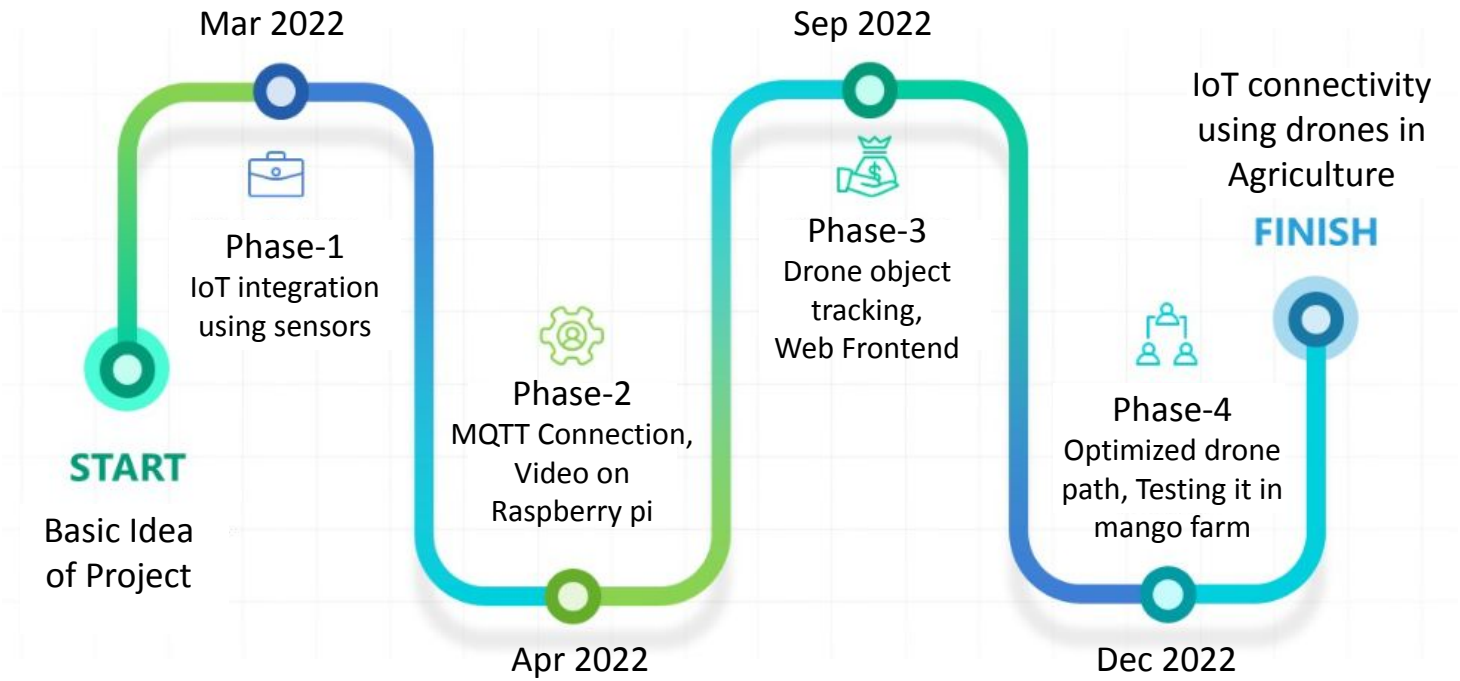
1. Problem Statement
2. Timeline
3. Recap Phase 1,2,3 + Results
4. Work done in Phase 4
5. Methodology
6. Results
    a. Drone path
    b. Python code
    c. CV outputs
    d. Cloud results
    e. Website screenshots
7. Video link
8. Future Explore
9. References

Problem Statement

The connectivity limitations of drone applications in smart agriculture and its solutions are analyzed in this project.

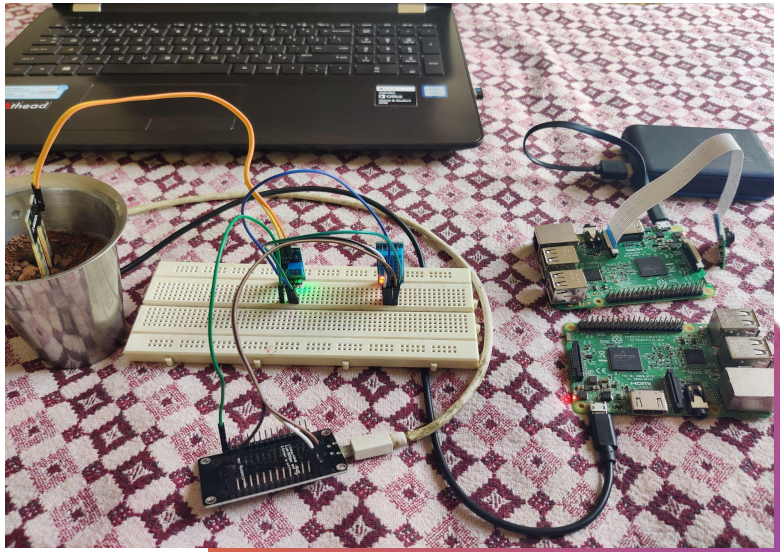
- Writing optimized python code for automating drone path through computer vision.
- Testing the tello drone in mango farm to check its accuracy in object tracking while flying.

Timeline

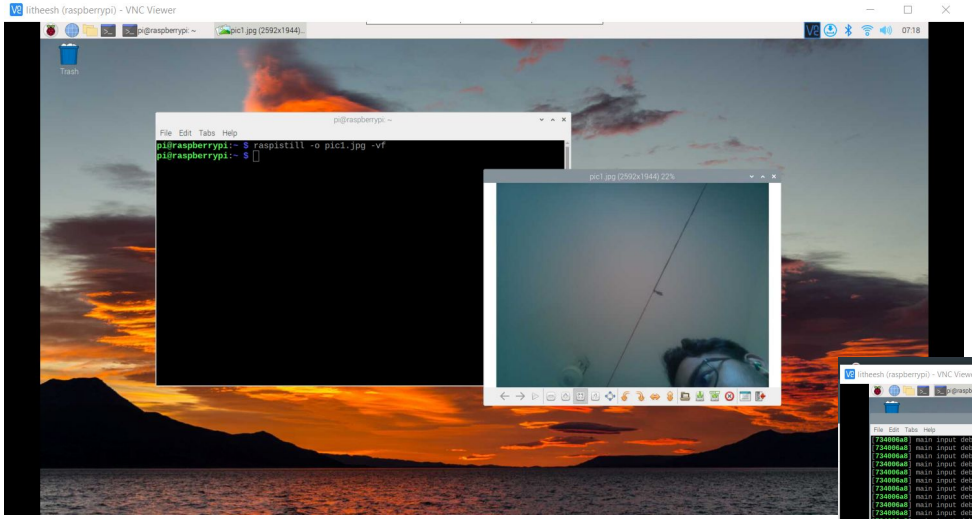

Mar 2022
Sep 2022
IoT connectivity using drones in Agriculture
FINISH
Phase-1
IoT integration using sensors
Phase-3
Drone object tracking, Web Frontend
Phase-2
MQTT Connection, Video on Raspberry pi
Phase-4
Optimized drone path, Testing it in mango farm
START
Basic Idea of Project
Apr 2022
Dec 2022

Recap: Phase 1, 2 works

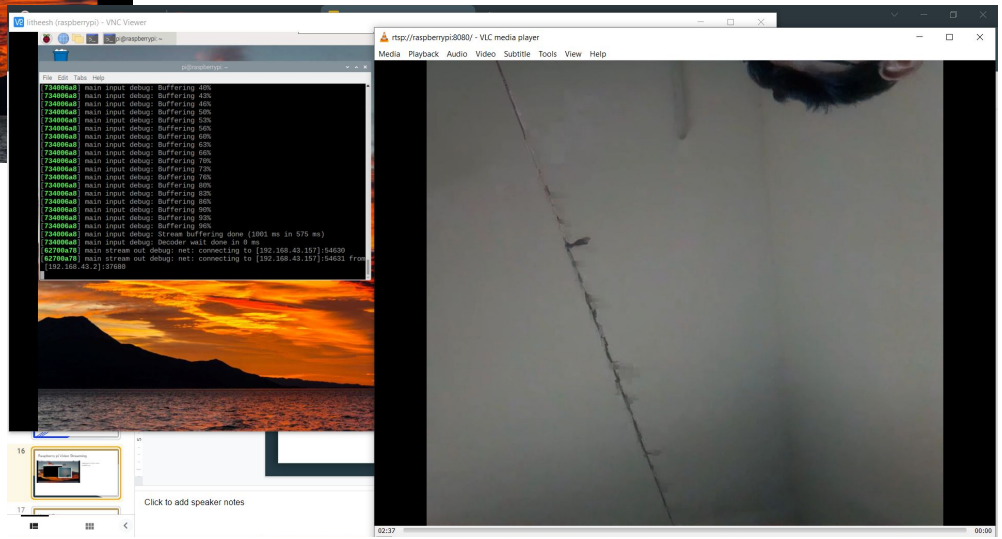
- Sensors are connected to Raspberry pi through NodeMCU.
- Pump Motor is connected to another NodeMCU.
- Both NodeMCUs are connected to Ground Raspberry pi using MQTT.
- Ground Raspberry pi changes to client from broker
- Drone Raspberry pi acts as MQTT broker
- and published sensor data on cloud.
- Fixing a camera module to Drone Raspberry
- Analyzed the delay in streaming in various resolutions.



Video & Photo

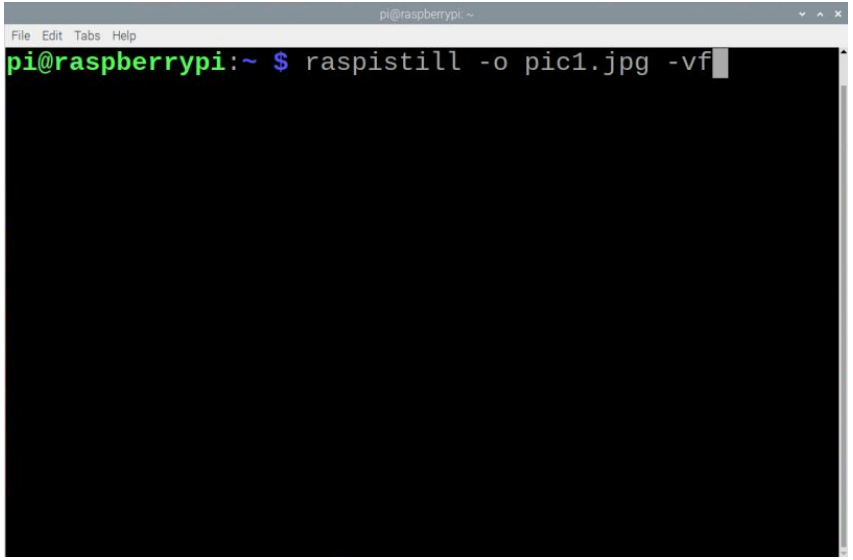


Taking photos from drone raspberry pi

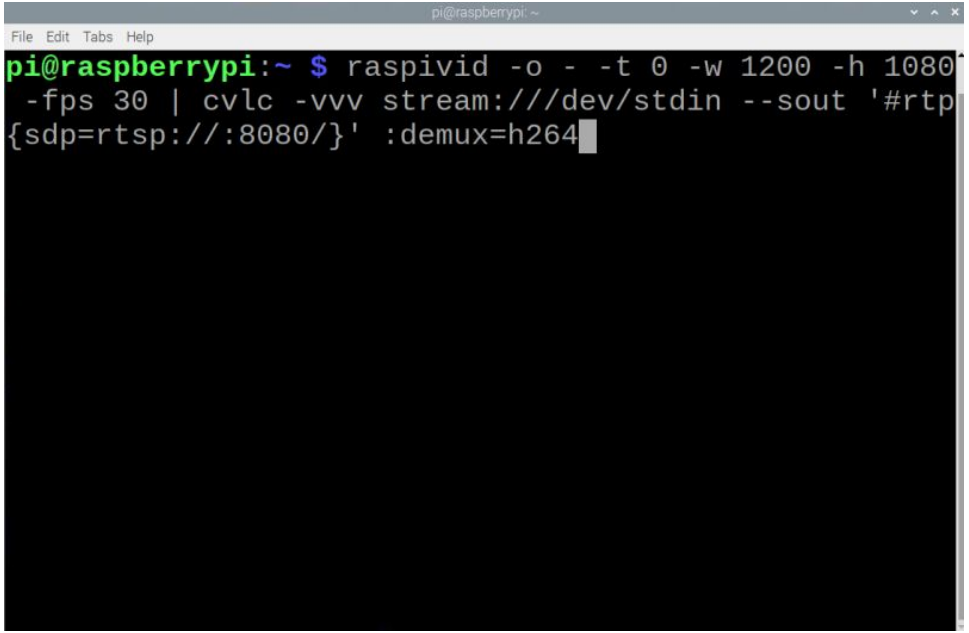


Live streaming video from drone raspberry pi

Raspberry pi Video Streaming



Command line for taking photos from drone raspberry pi



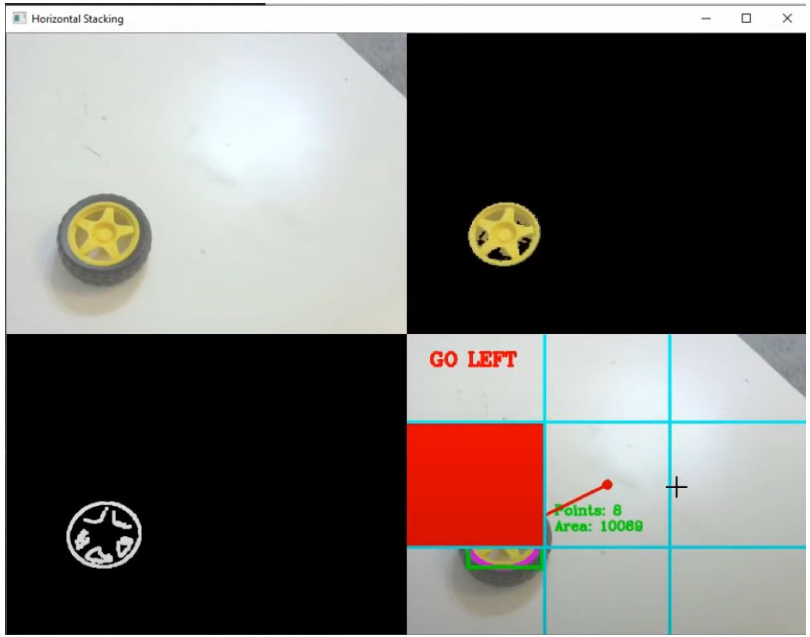
Command line for Live streaming video from drone raspberry pi

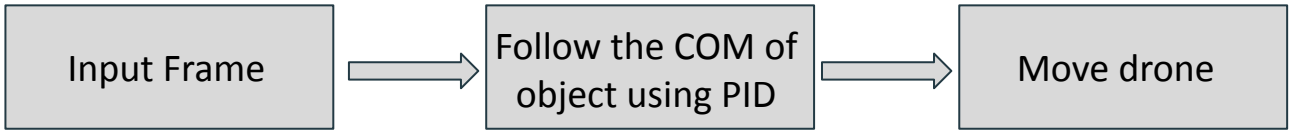
Recap: Phase 3 works

- We used Tello drone
- It can be flown both through app and commands
- Wrote Python code for automating flying mode
- It detects the object based on colour by Computer Vision(CV)
- follows the objects and reaches the nearest path to the object
- And take photos and videos at that particular location.
- Front End of the Website.

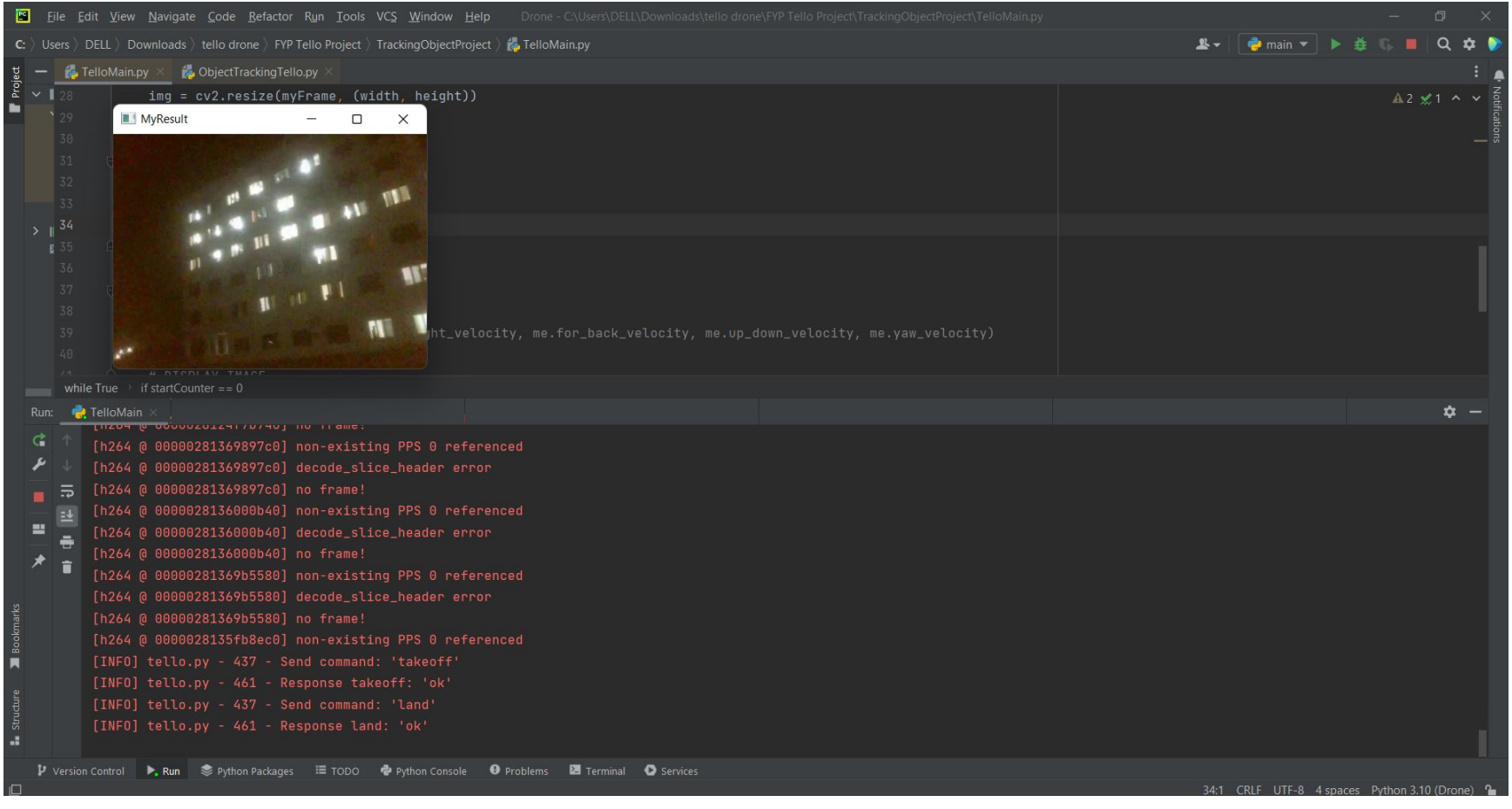
Object Tracking Drone

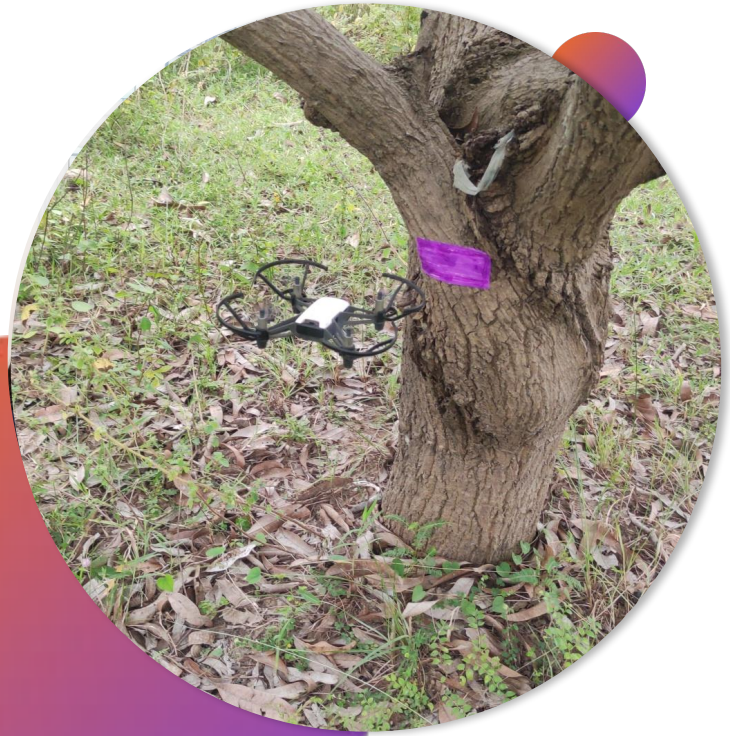
- Identify different objects in the captured image based on the color
- image to an HSI plane (HUE, SATURATION, and INTENSITY)
- We get the binary mask of our object
- try to put the COM of the object in the frame
- Now we can calculate the speed and movements of the drone





Python output:

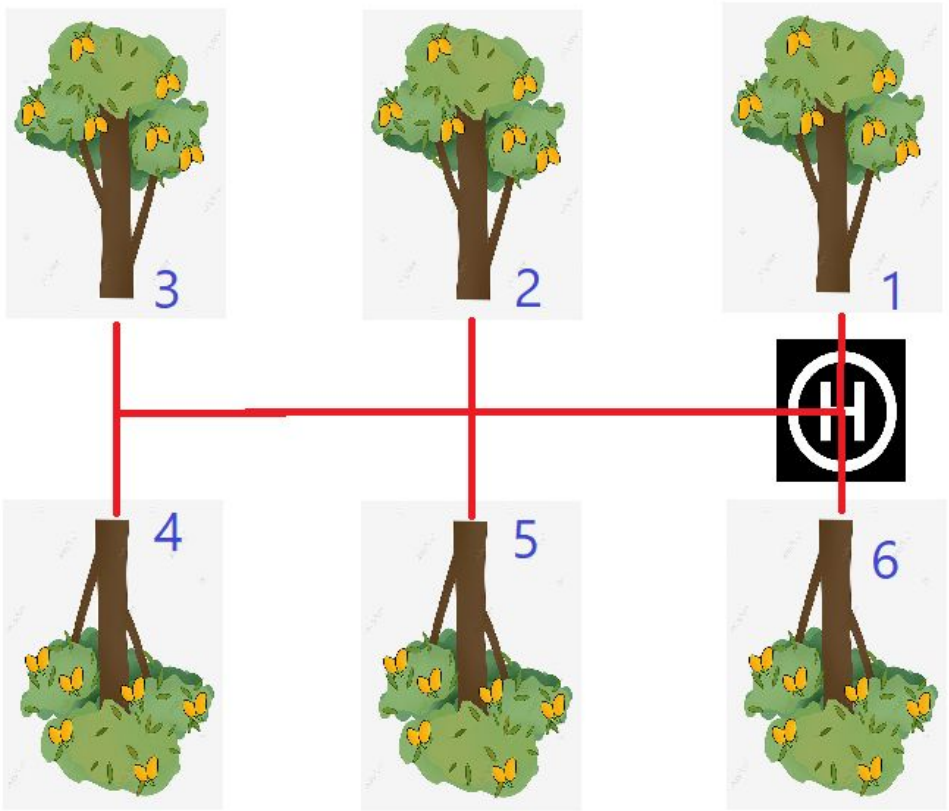




Phase 4 works:

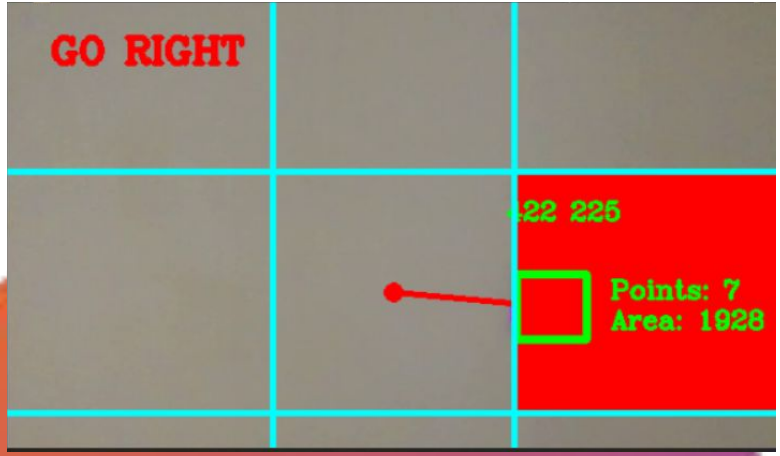
- Optimized object detection and tracking.
- We have given a path to drone.
- Tello drone follows that path by using Python code.
- If anywhere drone misses its path it uses computer vision to correct its path.
- And by reaching end of the path it collects data and passes it to cloud.
- Values will be updated on the website.

Given Drone path

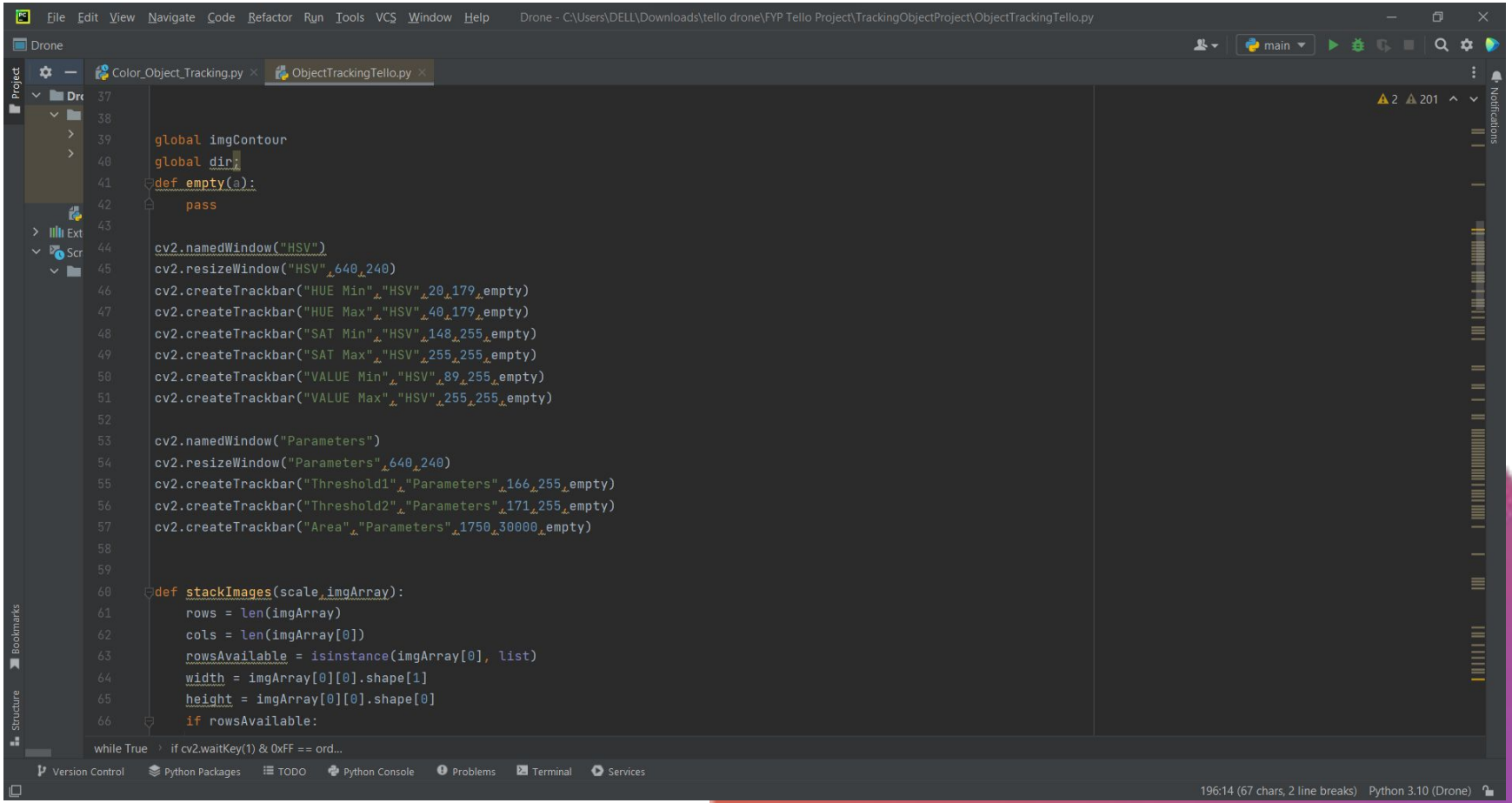


Methodology

- First step to color tracking is to display the camera feed that the Tello sees on the computer using CV2.
- The pixels in the image can be searched for pixels of a certain color by analyzing RGB values.
- In this project we used a range of “violet” values.
- Will blur together within the largest area violet color and draw a boundary.
- Using offset coordinates of the center of the object and the center of the frame.
- The drone should move either up, down, left, or right by observing the sign of the velocity.



Python Code:



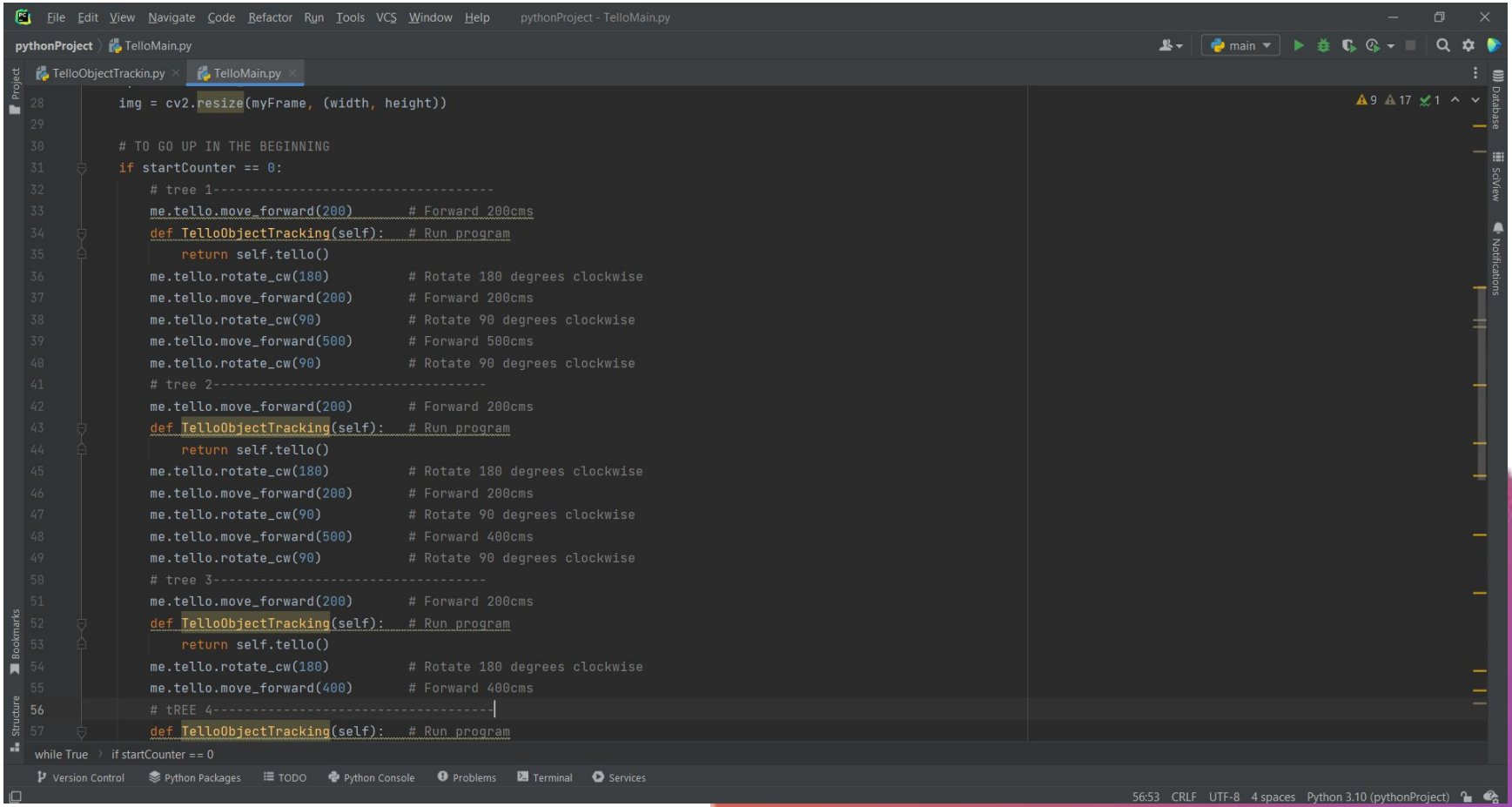
```
global imgContour
global dir;
def empty(a):
    pass

cv2.namedWindow("HSV")
cv2.resizeWindow("HSV",640,240)
cv2.createTrackbar("HUE Min","HSV",20,179,empty)
cv2.createTrackbar("HUE Max","HSV",40,179,empty)
cv2.createTrackbar("SAT Min","HSV",148,255,empty)
cv2.createTrackbar("SAT Max","HSV",255,255,empty)
cv2.createTrackbar("VALUE Min","HSV",89,255,empty)
cv2.createTrackbar("VALUE Max","HSV",255,255,empty)

cv2.namedWindow("Parameters")
cv2.resizeWindow("Parameters",640,240)
cv2.createTrackbar("Threshold1","Parameters",166,255,empty)
cv2.createTrackbar("Threshold2","Parameters",171,255,empty)
cv2.createTrackbar("Area","Parameters",1750,30000,empty)


def stackImages(scale,imgArray):
    rows = len(imgArray)
    cols = len(imgArray[0])
    rowsAvailable = isinstance(imgArray[0], list)
    width = imgArray[0][0].shape[1]
    height = imgArray[0][0].shape[0]
    if rowsAvailable:
```

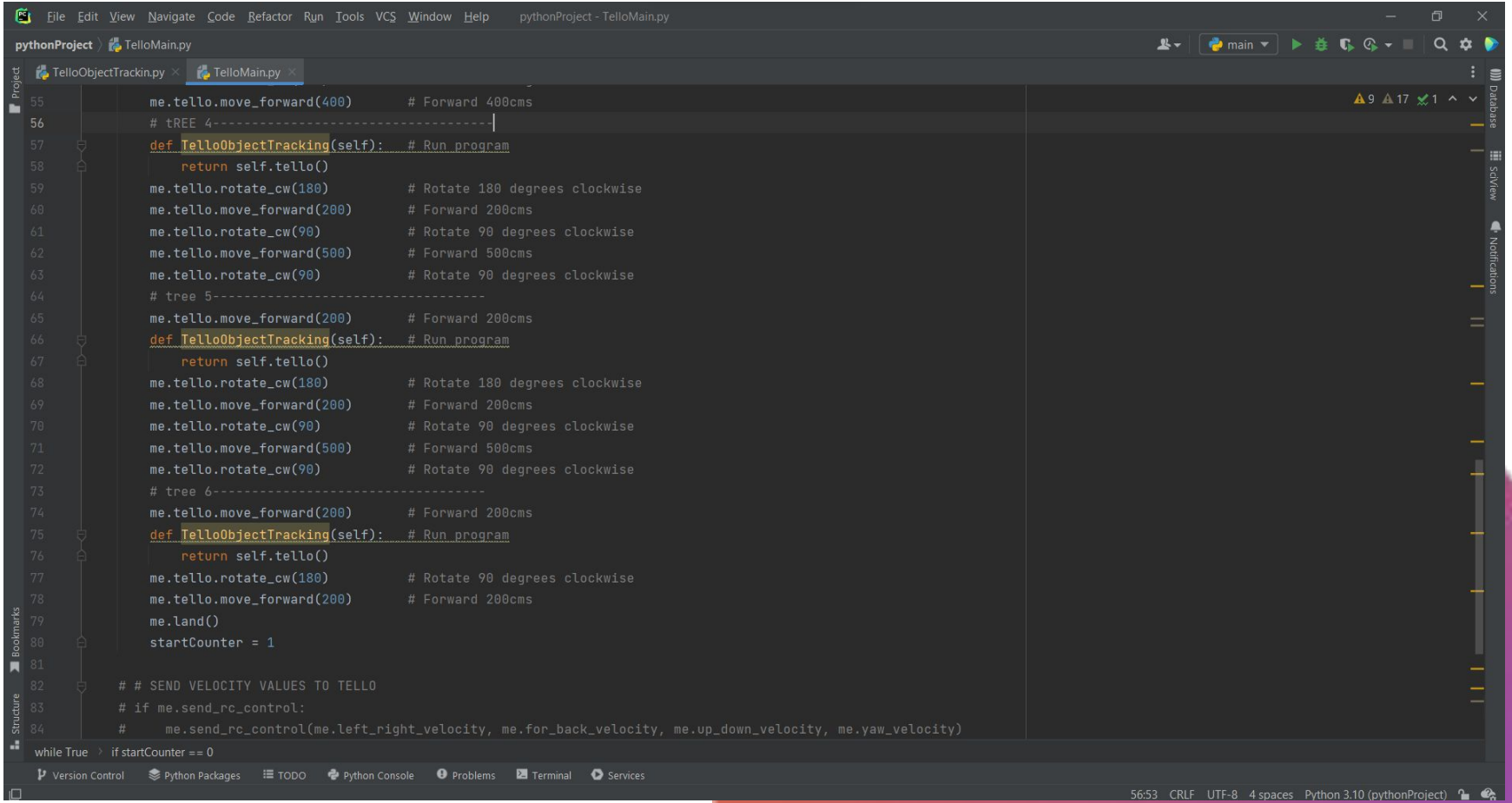
Python Code:



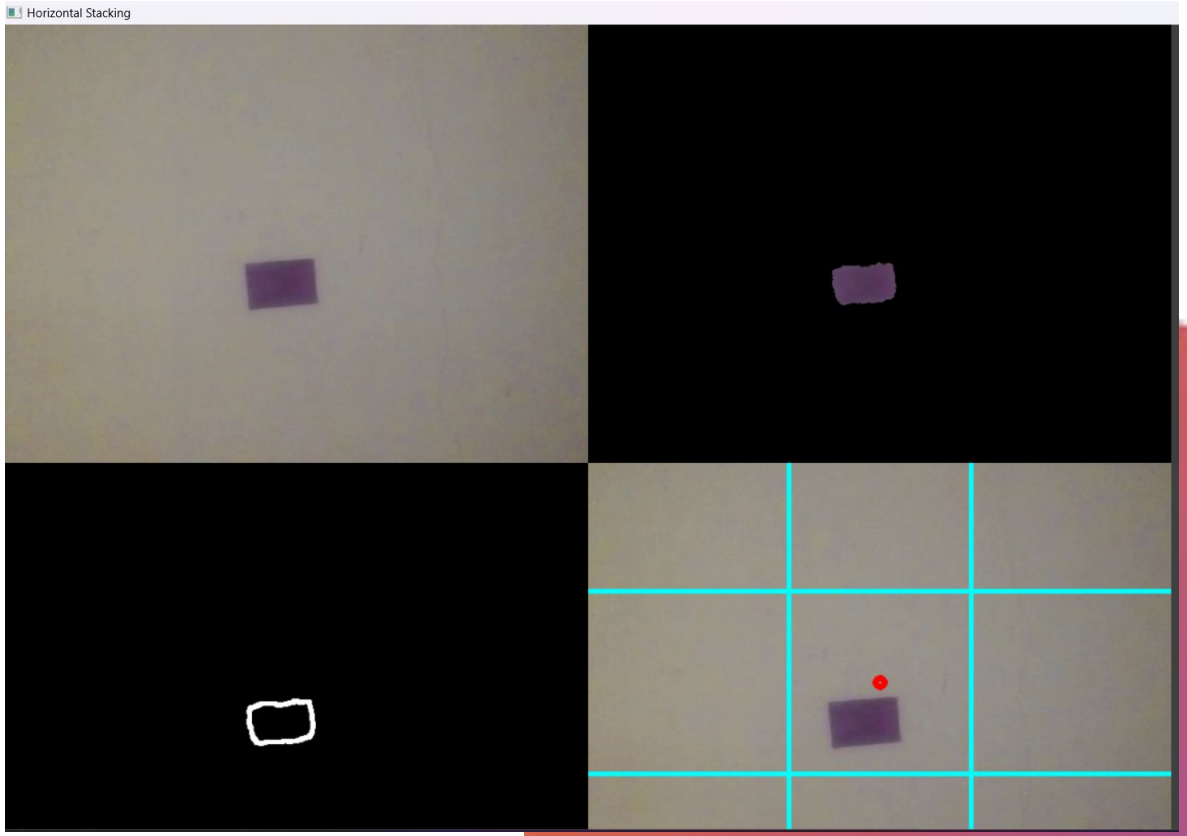
```
    img = cv2.resize(myFrame, (width, height))

    # TO GO UP IN THE BEGINNING
    if startCounter == 0:
        # tree 1-----------------------------------
        me.tello.move_forward(200)       # Forward 200cms
        def TelloObjectTracking(self):   # Run program
            return self.tello()
        me.tello.rotate_cw(180)          # Rotate 180 degrees clockwise
        me.tello.move_forward(200)       # Forward 200cms
        me.tello.rotate_cw(90)           # Rotate 90 degrees clockwise
        me.tello.move_forward(500)       # Forward 500cms
        me.tello.rotate_cw(90)           # Rotate 90 degrees clockwise
        # tree 2-----------------------------------
        me.tello.move_forward(200)       # Forward 200cms
        def TelloObjectTracking(self):   # Run program
            return self.tello()
        me.tello.rotate_cw(180)          # Rotate 180 degrees clockwise
        me.tello.move_forward(200)       # Forward 200cms
        me.tello.rotate_cw(90)           # Rotate 90 degrees clockwise
        me.tello.move_forward(500)       # Forward 400cms
        me.tello.rotate_cw(90)           # Rotate 90 degrees clockwise
        # tree 3-----------------------------------
        me.tello.move_forward(200)       # Forward 200cms
        def TelloObjectTracking(self):   # Run program
            return self.tello()
        me.tello.rotate_cw(180)          # Rotate 180 degrees clockwise
        me.tello.move_forward(400)       # Forward 400cms
        # tREE 4-----------------------------------
        def TelloObjectTracking(self):   # Run program
```

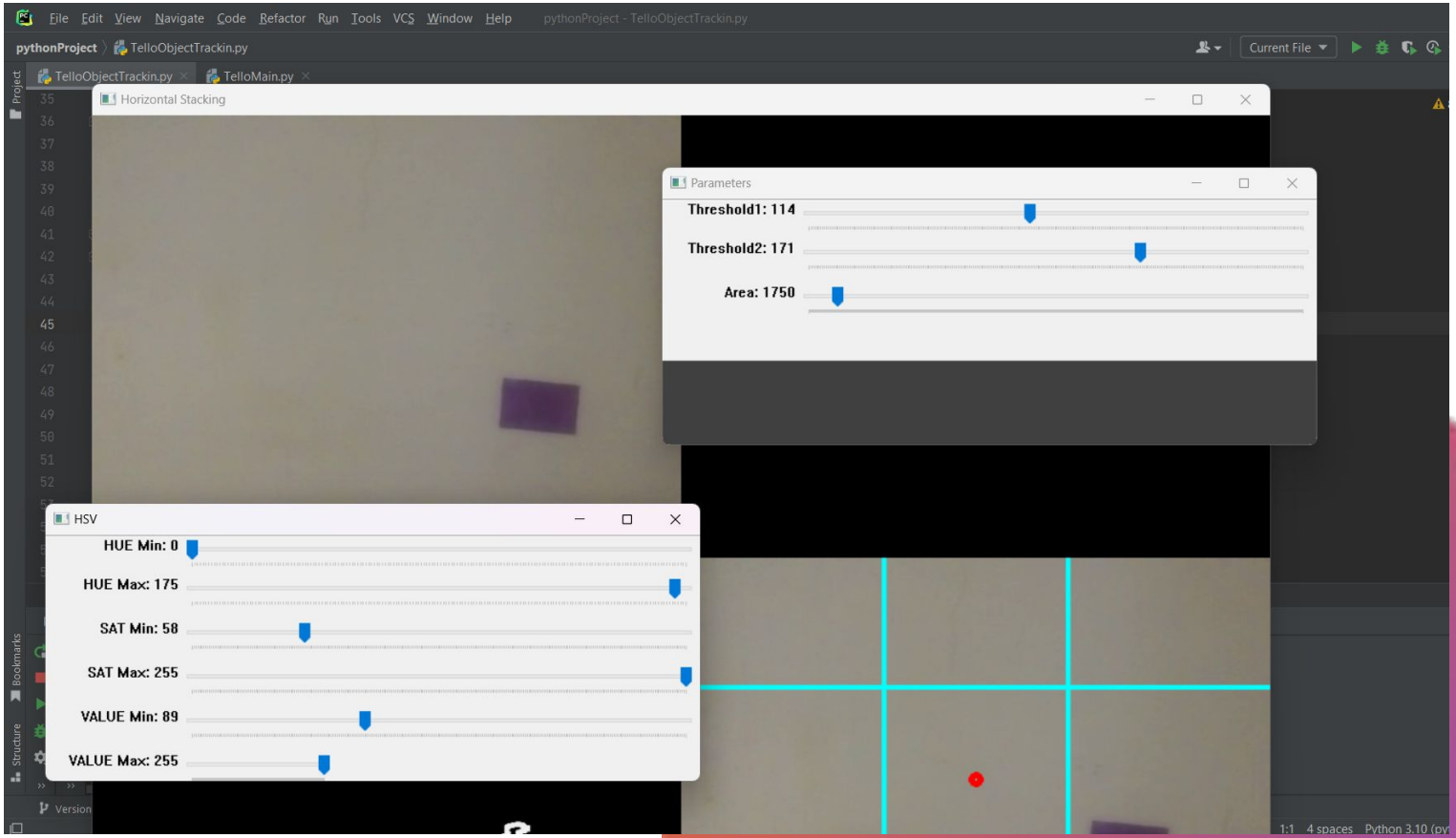
Python Code:


File Edit View Navigate Code Refactor Run Tools VCS Window Help pythonProject - TelloMain.py
pythonProject TelloMain.py
main
TelloObjectTrackin.py
TelloMain.py
me.tello.move_forward(400) # Forward 400cms
tREE 4----------
def TelloObjectTracking(self): # Run program
return self.tello()
me.tello.rotate_cw(180) # Rotate 180 degrees clockwise
me.tello.move_forward(200) # Forward 200cms
me.tello.rotate_cw(90) # Rotate 90 degrees clockwise
me.tello.move_forward(500) # Forward 500cms
me.tello.rotate_cw(90) # Rotate 90 degrees clockwise
tree 5----------
me.tello.move_forward(200) # Forward 200cms
def TelloObjectTracking(self): # Run program
return self.tello()
me.tello.rotate_cw(180) # Rotate 180 degrees clockwise
me.tello.move_forward(200) # Forward 200cms
me.tello.rotate_cw(90) # Rotate 90 degrees clockwise
me.tello.move_forward(500) # Forward 500cms
me.tello.rotate_cw(90) # Rotate 90 degrees clockwise
tree 6----------
me.tello.move_forward(200) # Forward 200cms
def TelloObjectTracking(self): # Run program
return self.tello()
me.tello.rotate_cw(180) # Rotate 90 degrees clockwise
me.tello.move_forward(200) # Forward 200cms
me.land()
startCounter = 1
# # SEND VELOCITY VALUES TO TELLO
# if me.send_rc_control:
# me.send_rc_control(me.left_right_velocity, me.for_back_velocity, me.up_down_velocity, me.yaw_velocity)
while True if startCounter == 0
Version Control Python Packages TODO Python Console Problems Terminal Services
56:53 CRLF UTF-8 4 spaces Python 3.10 (pythonProject)

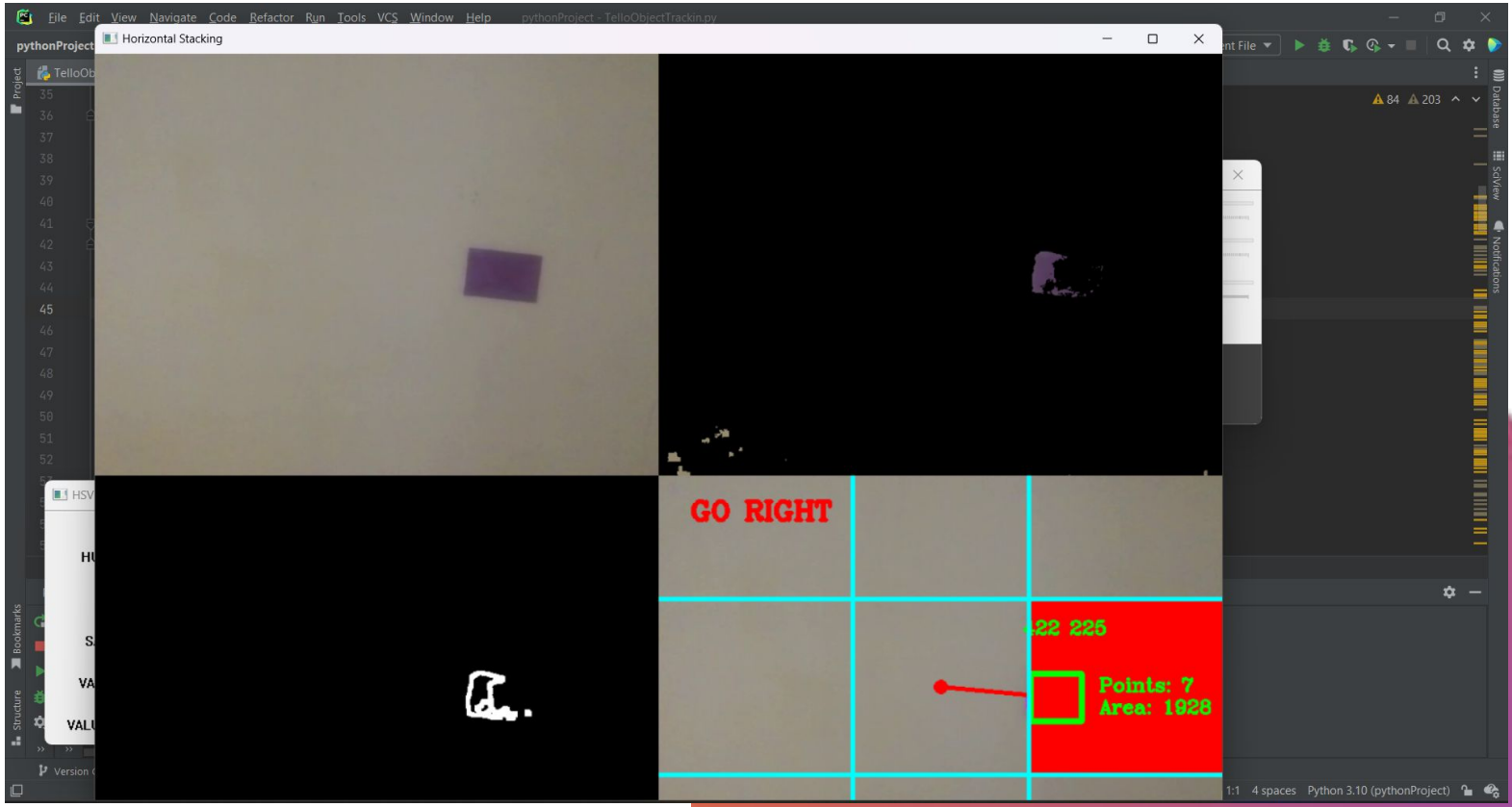
Python Results:



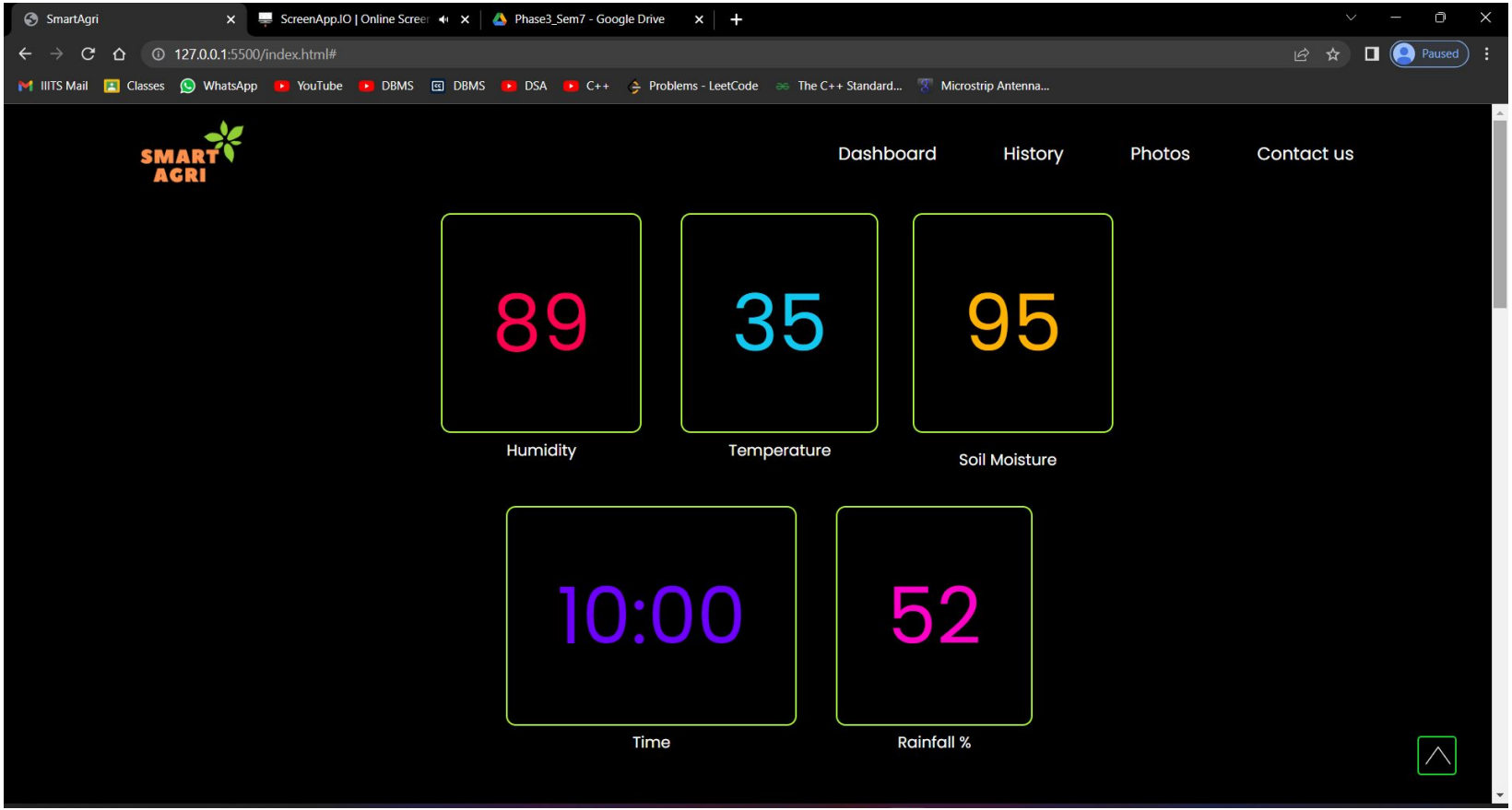
Python Results



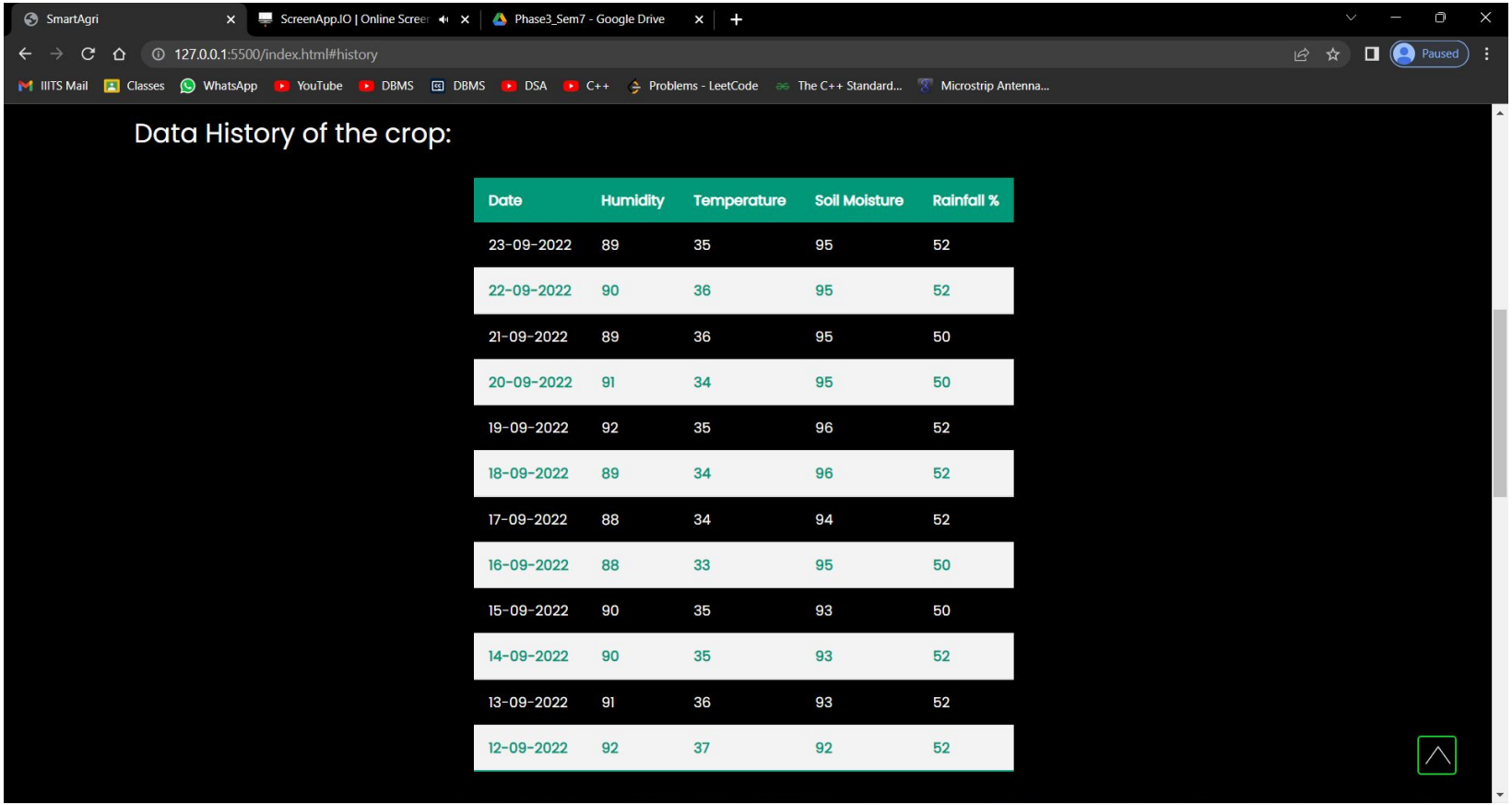
Python Results



Website Screenshots:



Website Screenshots:



Date	Humidity	Temperature	Soil Moisture	Rainfall %
23-09-2022	89	35	95	52
22-09-2022	90	36	95	52
21-09-2022	89	36	95	50
20-09-2022	91	34	95	50
19-09-2022	92	35	96	52
18-09-2022	89	34	96	52
17-09-2022	88	34	94	52
16-09-2022	88	33	95	50
15-09-2022	90	35	93	50
14-09-2022	90	35	93	52
13-09-2022	91	36	93	52
12-09-2022	92	37	92	52

Video link:

Click here to Watch

Links:

Python code link:
https://github.com/litheeshkumar/IoT-Connectivity-for-Drone-applications/tree/main/Python%20Codes

Website source code link:
https://github.com/litheeshkumar/IoT-Connectivity-for-Drone-applications/tree/main/SmartAgri%20Website

Google drive link for pics and video taken by Tello:
https://drive.google.com/drive/folders/1LUn7OpOfWLix3Sg-68jW5SKrzOUXE_A9?usp=share_link

Challenges faced:

- The issues of dying batteries and overheating.
- If the battery life of the Tello is too low, the drone will not take flight.
- Before running code, charge all battery packs and have replacements if drone will not lift off.
- The DJI Tello will not take flight if it is overheating.
- Landing of drone is one of the main problem - placing a notebook as shown - as a launch surface.
- Finally, several libraries must be downloaded to run the Tello with CV2.

Future Scope:

- Drone object detection has many applications in real-time.
- We thinking that deep learning algorithms perform better than image processing methods.
- Although the drones are small in size and restrictions in power, it is important to design an efficient deep learning algorithm for drone object detection.
- As a future work, we can aim to investigate object detection using UAV in various other fields.
- Precision agriculture is to monitor the fields precisely by collecting and analyzing the data.

References:

1. Jeffrey Kantor. CBE 30338 Visual Tracking of an Object with a Drone. IEEE Access 2019, 7, 129551–129583. [CrossRef]

2. A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. International Journal of Robotics Research (IJRR), 2013. [CrossRef]

3. AlDahoul, N., Sabri, A. Q., & Mansoor, A. M. (2018). Real-time human detection for aerial captured video sequences via deep models. Hindawi, 2018, 15. [CrossRef]

4. S. Han, H. Mao, and W. J. Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. arXiv preprint arXiv:1510.00149, 2015. [CrossRef]

Thank You