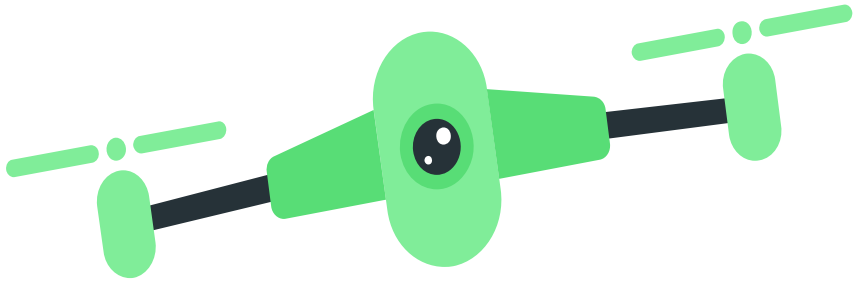
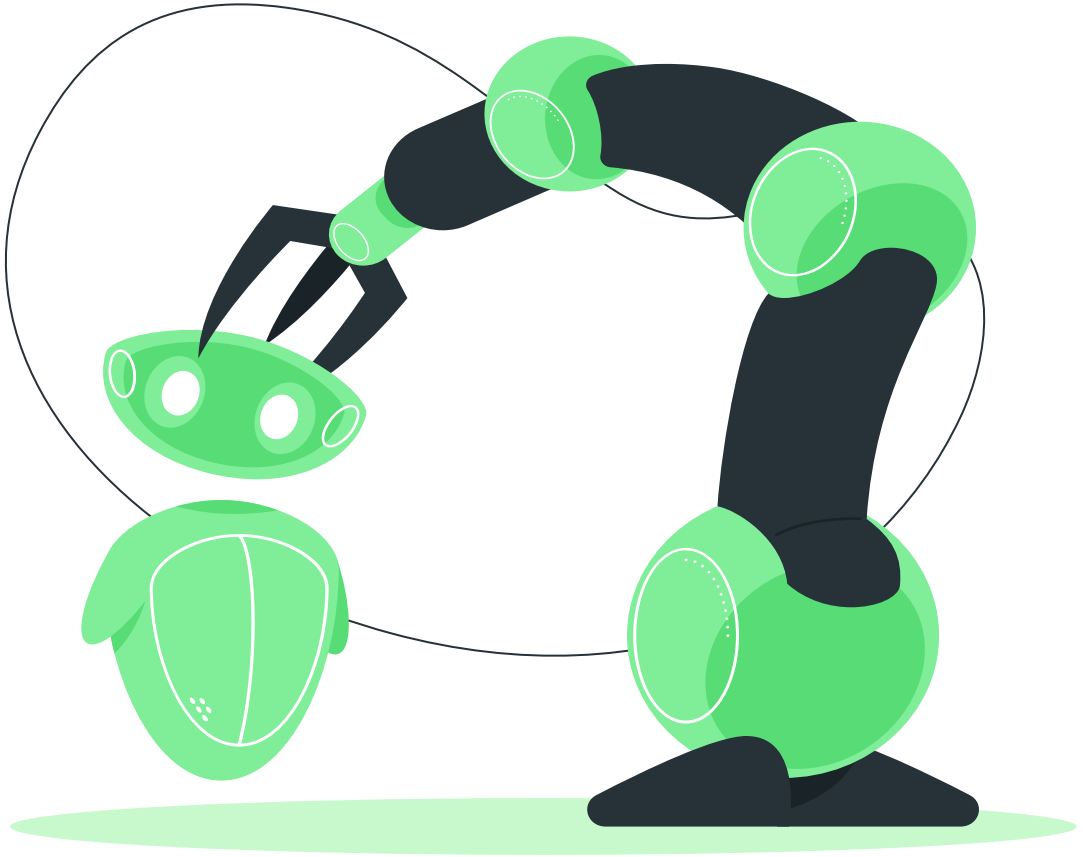
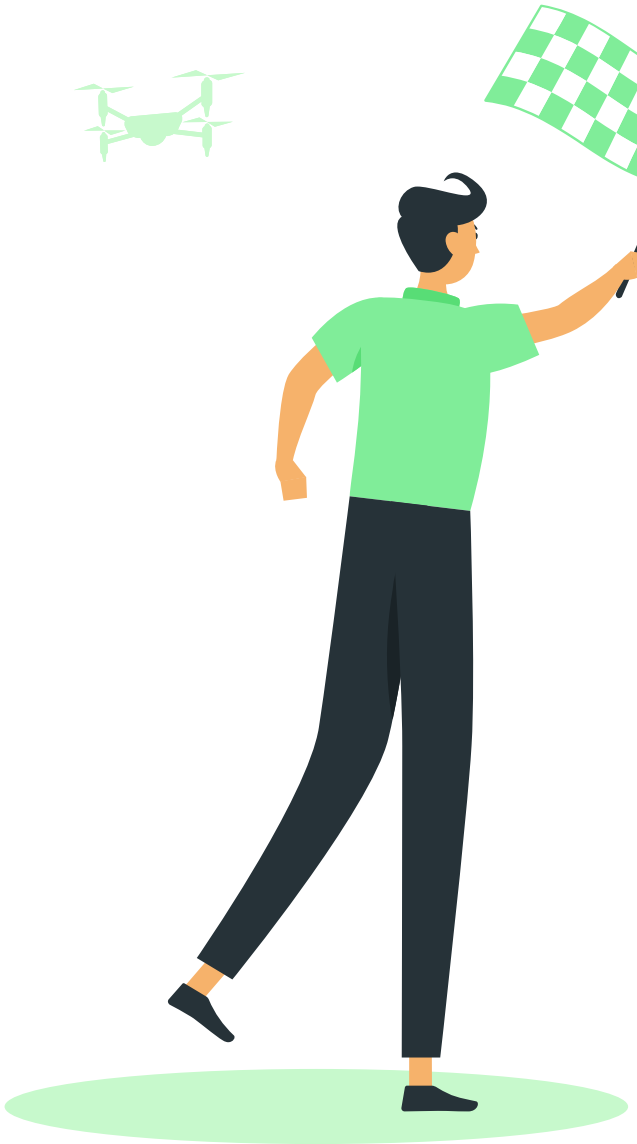
Flying Selfie Stick





By-
Shubham Gulati(2K20/IT/141)

FLYING SELFIE STICK

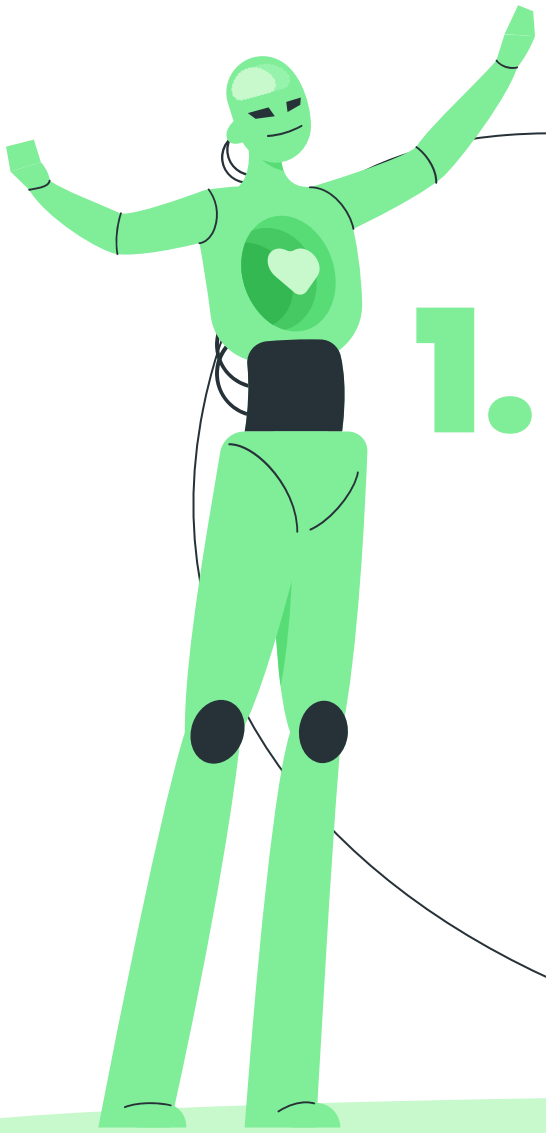


Trust me, I'm a Drone Pilot

We have divided the problem statement into two parts :

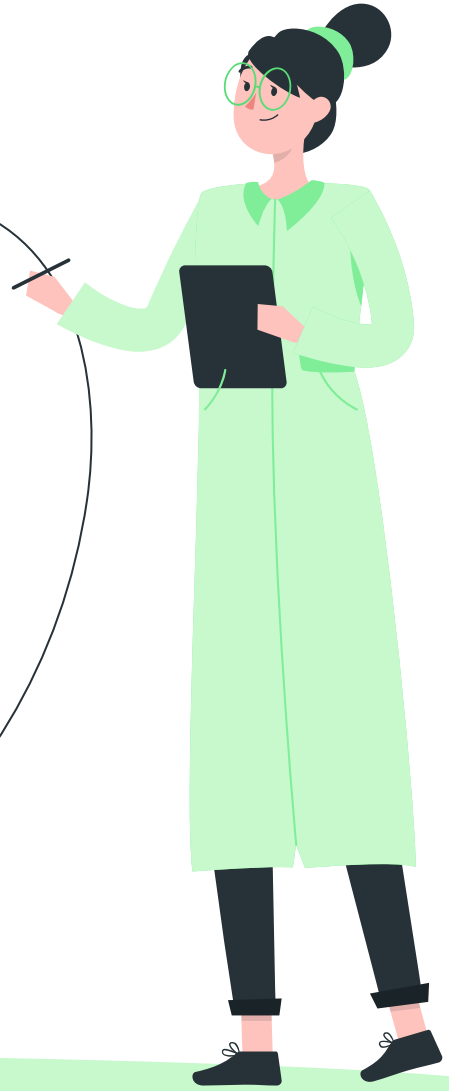
1. Object detection
2. Object tracking

Working of our project



1. Get the video feed from the drone

The project starts by getting the video feed from the drone and the passing on each frome the ML models for processing and then sending corresponding commands to the drone.



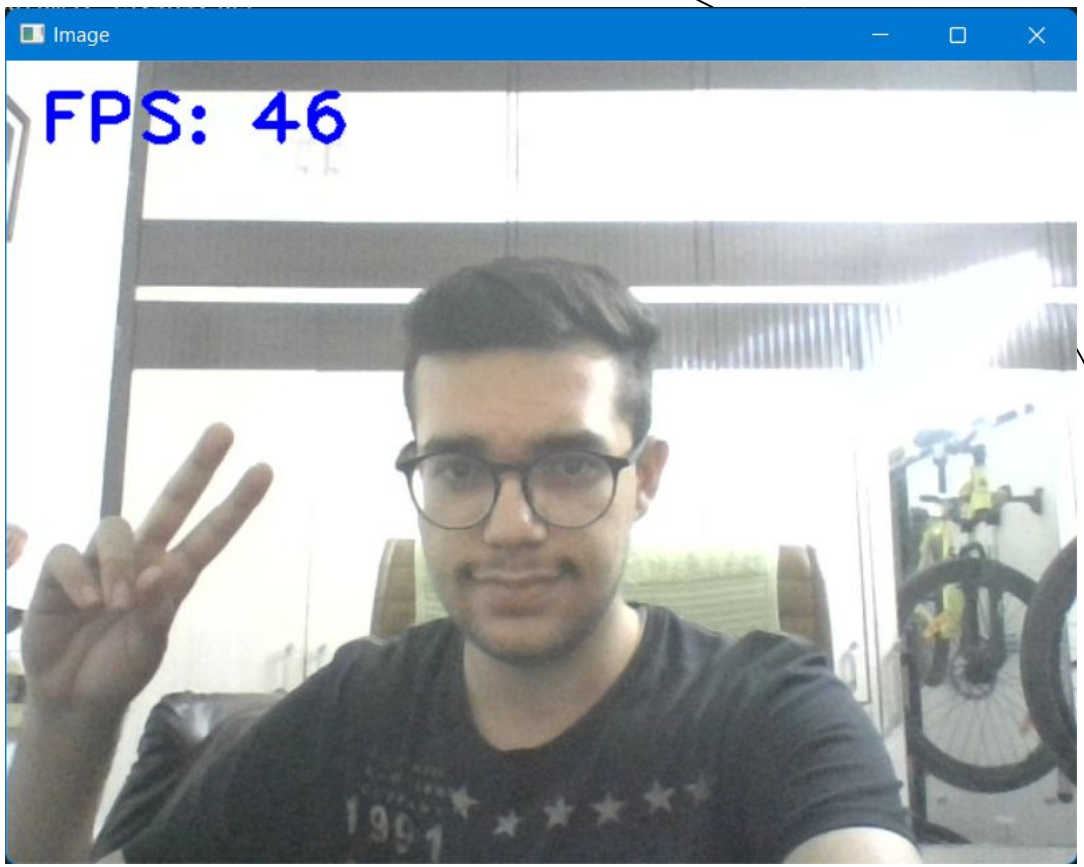
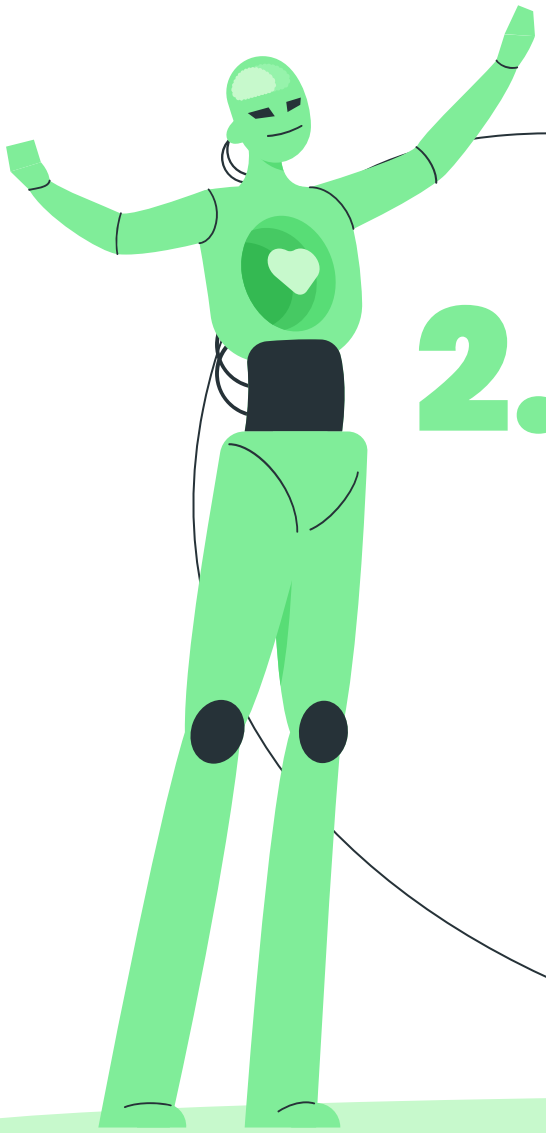


Image
FPS: 46

Working of our project



2. Person and hand detection using YOLO

We have used pre-trained YOLO models for person and hand detection which creates a bounding box around the object and then passes these parameters to the next step.

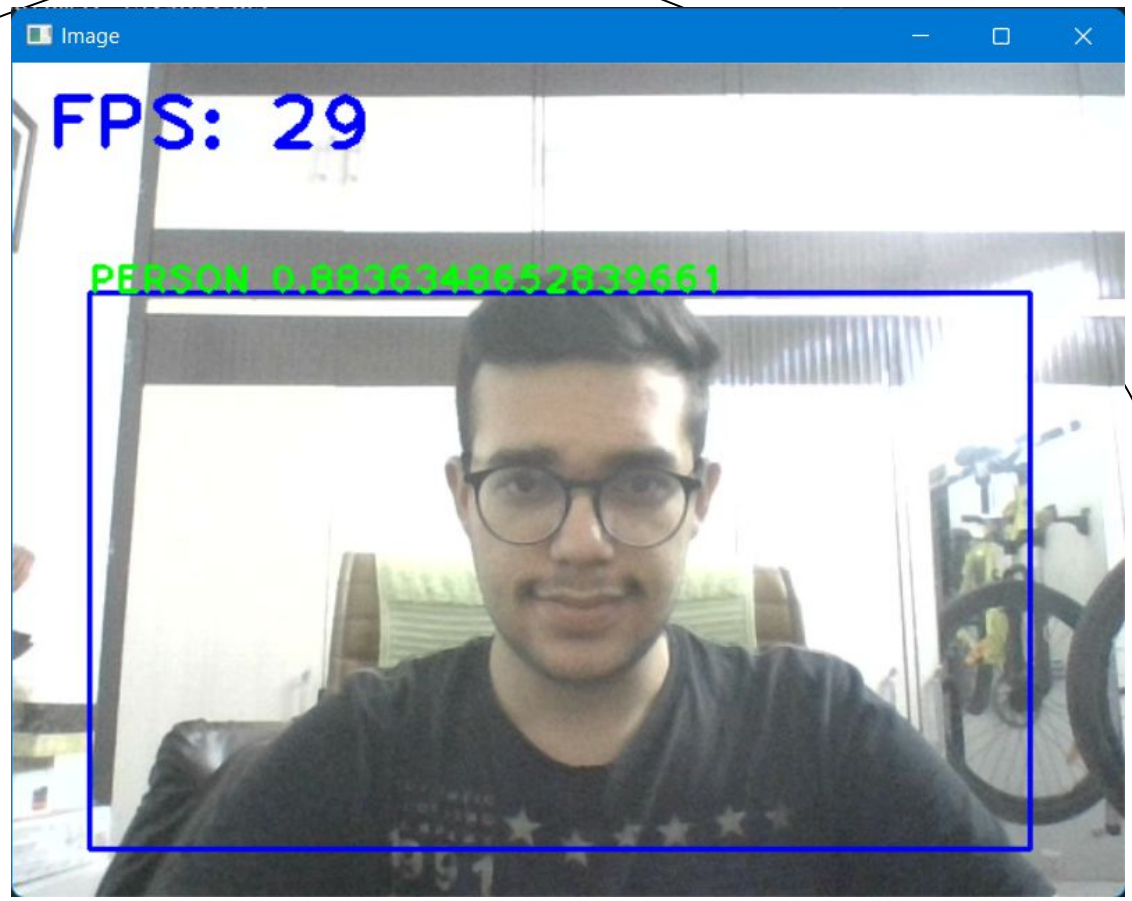
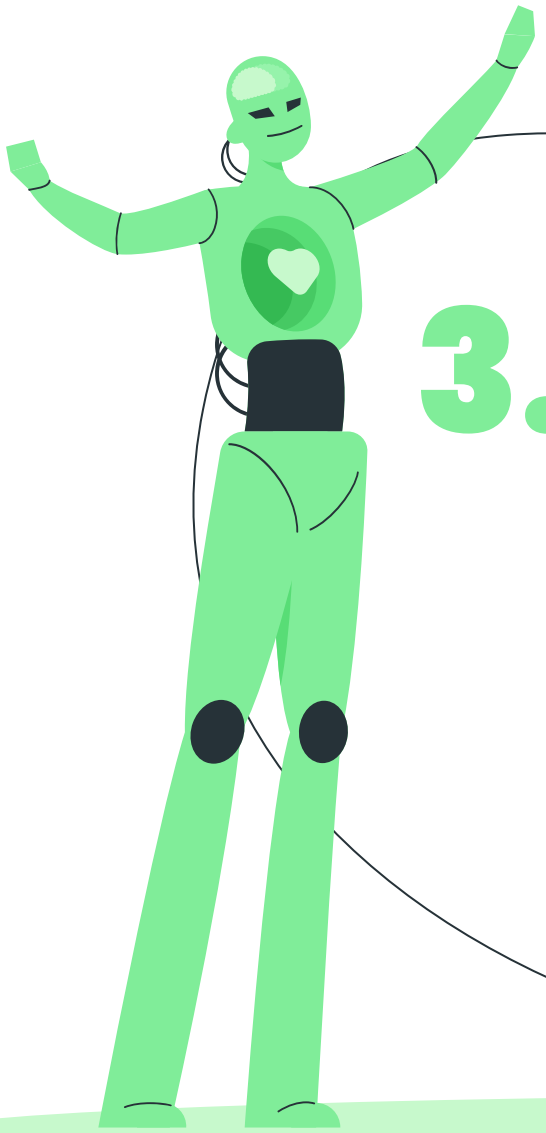


Image
FPS: 29
PERSON 0.8836348652839661

```python
def findPerson(img):
    # detecting persons using yolo
    classIds, confs, bbox = net.detect(img, confThreshold = 0.5)

    bbox = list(bbox)
    confs = list(np.array(confs).reshape(1, -1)[0])

    # non maximum suppression
    indices = cv2.dnn.NMSBoxes(bbox, confs, 0.5, 0.2)
    # print(indices)

    centerList = []
    areaList = []

    for i in indices:
        if classIds[i] == 1:
            box = bbox[i]
            x, y, w, h = box[0], box[1], box[2], box[3]
            centerList.append((x + w//2, y + h//2))
            areaList.append(w*h)
            cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
            cv2.putText(img, f'{classes[classIds[i]].upper()} {confs[i]}',

    if len(centerList) == 0:
        return img, [[0, 0], 0]

    # find the person with the largest area
    maxArea = max(areaList)
    maxIndex = areaList.index(maxArea)

    return img, [centerList[maxIndex], maxArea]
```

Working of our project



3. Gesture classification using a simple CNN

We have trained a simple gesture classification model using tensorflow and keras. The dataset used is American Sign Language Dataset which has been taken from Kaggle. We use this model to classify the hand gestures shown in the video feed.

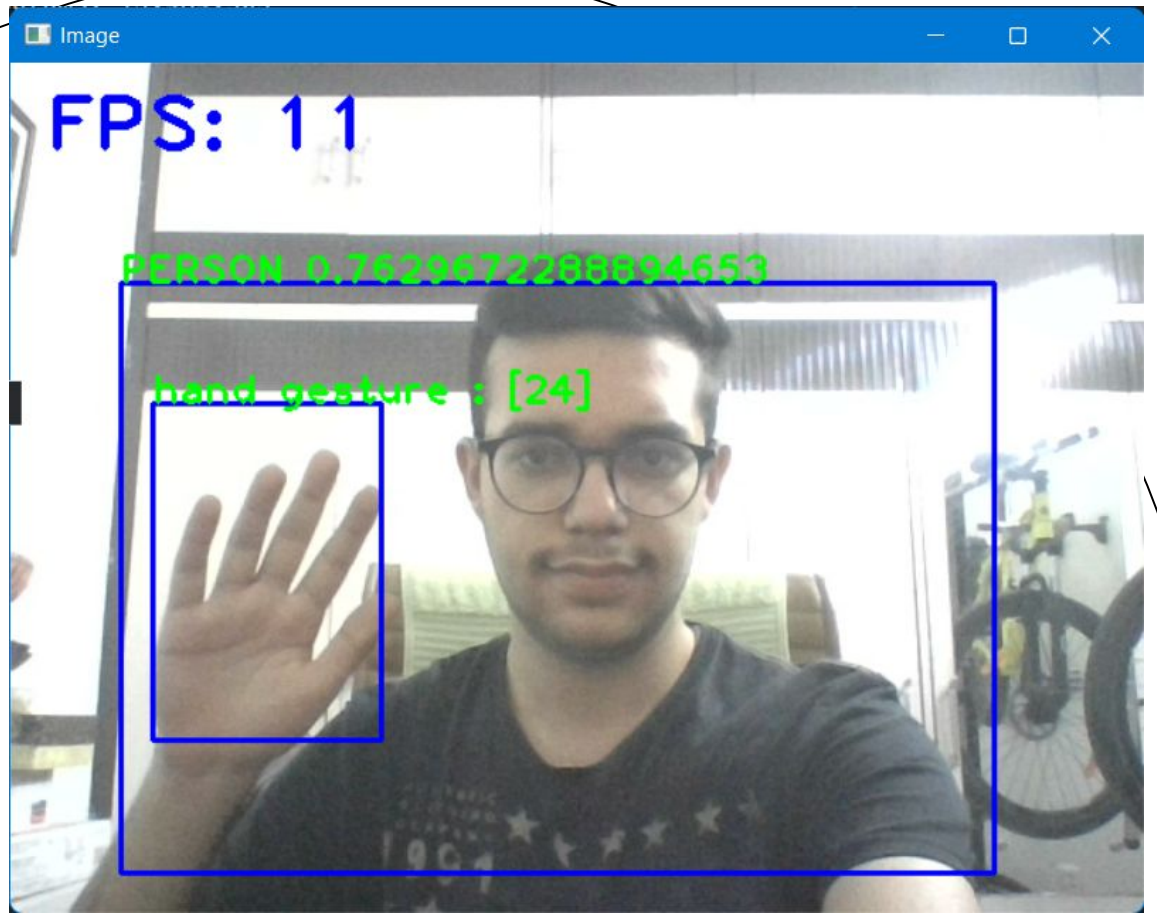
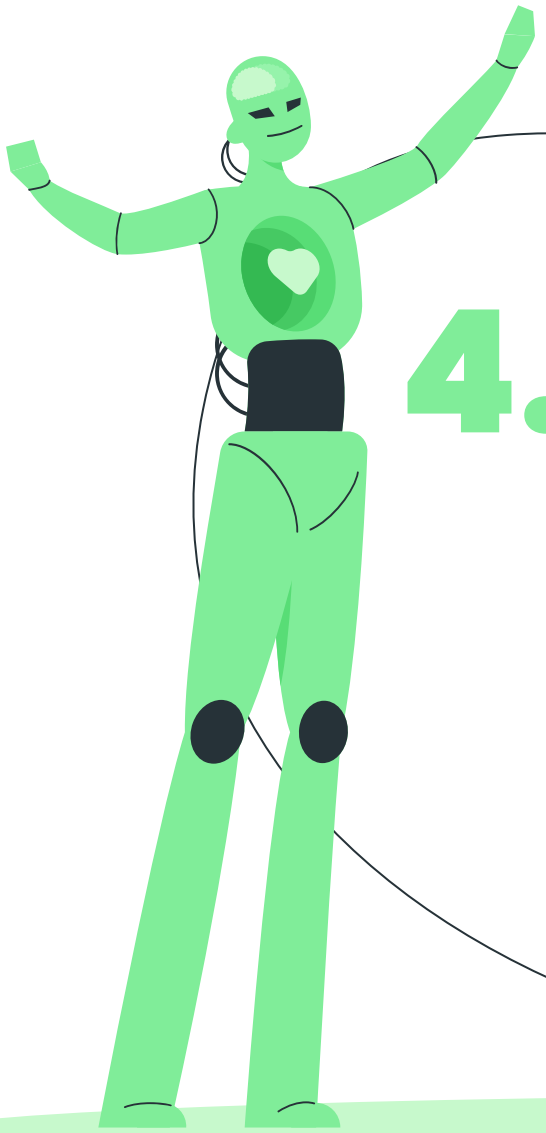


Image
FPS: 11
PERSON 0.7629672288894653
hand gesture : [24]

```python
def findGesture(img):
    # detecing hands using yolo
    handIds, handConfs, handBbox = net2.detect(img, confThreshold = 0.5)

    handBbox = list(handBbox)
    handConfs = list(np.array(handConfs).reshape(1, -1)[0])
    # confs = list(map(float, confs))

    handindices = cv2.dnn.NMSBoxes(handBbox, handConfs, 0.5, 0.2)
    # print(indices)

    for i in handindices:
        box = handBbox[i]
        x, y, w, h = box[0], box[1], box[2], box[3]
        cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)
        handimg = img[box[1]:box[1]+box[3], box[0]:box[0]+box[2]]
        handimg = cv2.resize(handimg, (28, 28))
        handimg_gray = cv2.cvtColor(handimg, cv2.COLOR_BGR2GRAY)
        ans = classifier.predict(handimg_gray.reshape(1, 28, 28, 1))
        gesture = np.argmax(ans, axis=1)
        cv2.putText(img, f'hand gesture : {gesture}', box[:2], cv2.FONT_HE
        # cv2.imshow('hand', handimg)

    return img, gesture
```

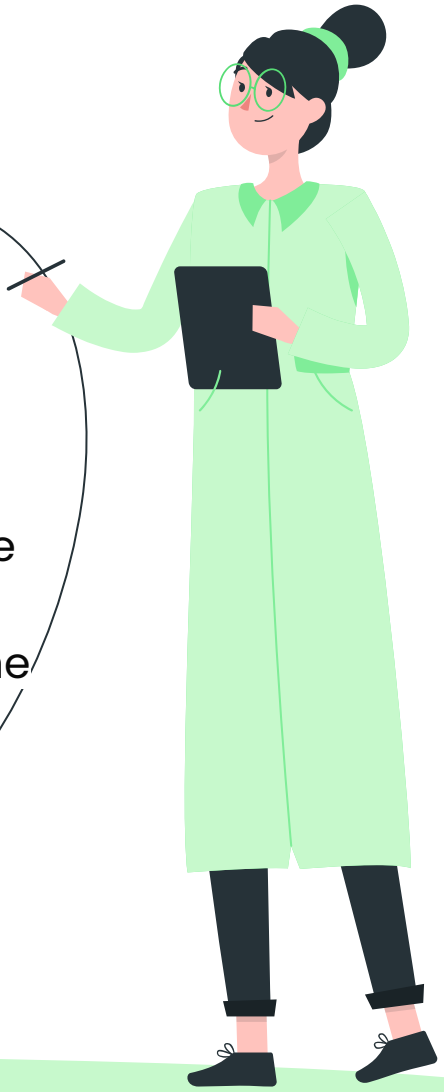
Working of our project



4. PID Person Tracker

This is the most interesting part.
To keep the person in the frame and not to lose the tracking, we needed some sort of an algorithm which keeps the person always at the center of the frame.

The PID algorithm came to a great rescue for this part.



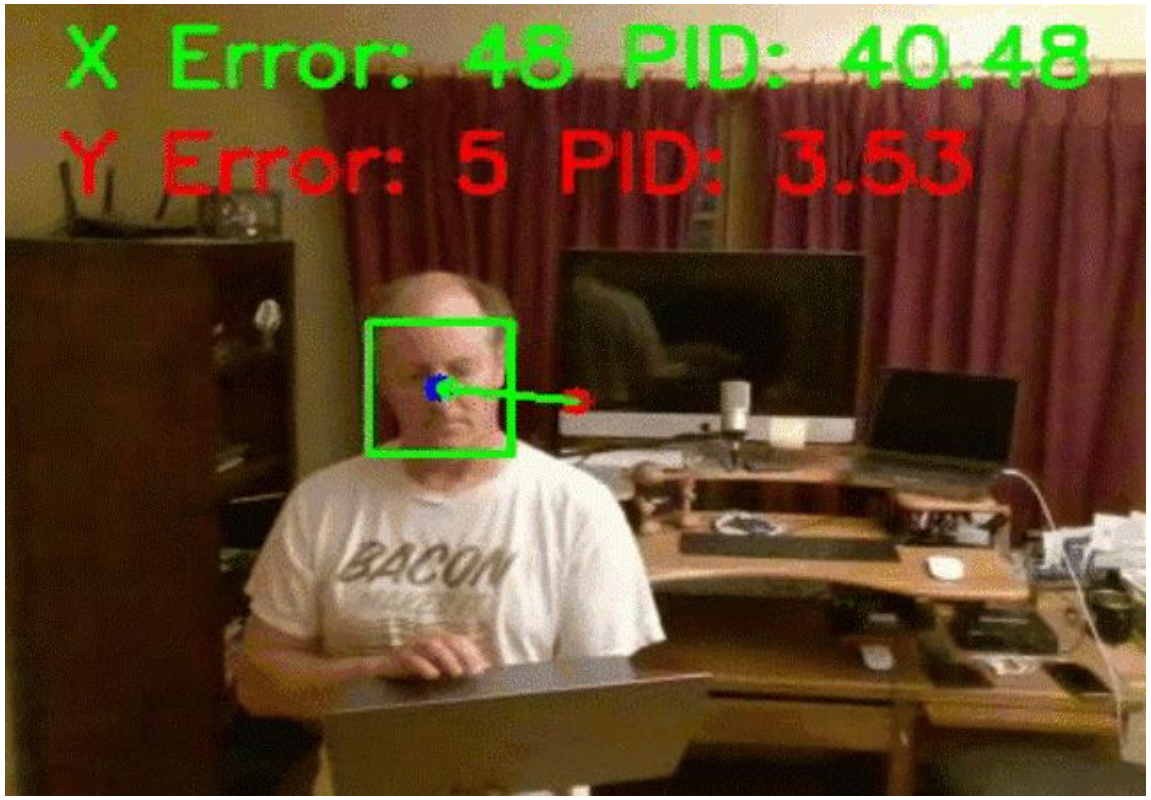
```python
def follow_person(info, pErrorYV, pErrorUD, vel=20):
    lr, fb, ud, yv = 0, 0, 0, 0

    area, x, y = info[1], info[0][0], info[0][1]
    global HEIGHT, WIDTH

    if area == 0:
        commands = [0, 0, 0, 20]
        errorUD, errorYV = 0, 0
        return commands, errorYV, errorUD

    errorYV = x - WIDTH // 2
    yv = pid[0] * errorYV + pid[1] * (errorYV - pErrorYV)
    yv = int(np.clip(yv, -100, 100))

    errorUD = HEIGHT // 2 - y
    ud = pid[0] * errorUD + pid[1] * (errorUD - pErrorUD)
    ud = int(np.clip(ud, -100, 100))

    if area > fbRange[1]:
        fb = -vel
    elif area < fbRange[0] and area > 0:
        fb = vel

    commands = [lr, fb, ud, yv]

    return commands, errorYV, errorUD
```

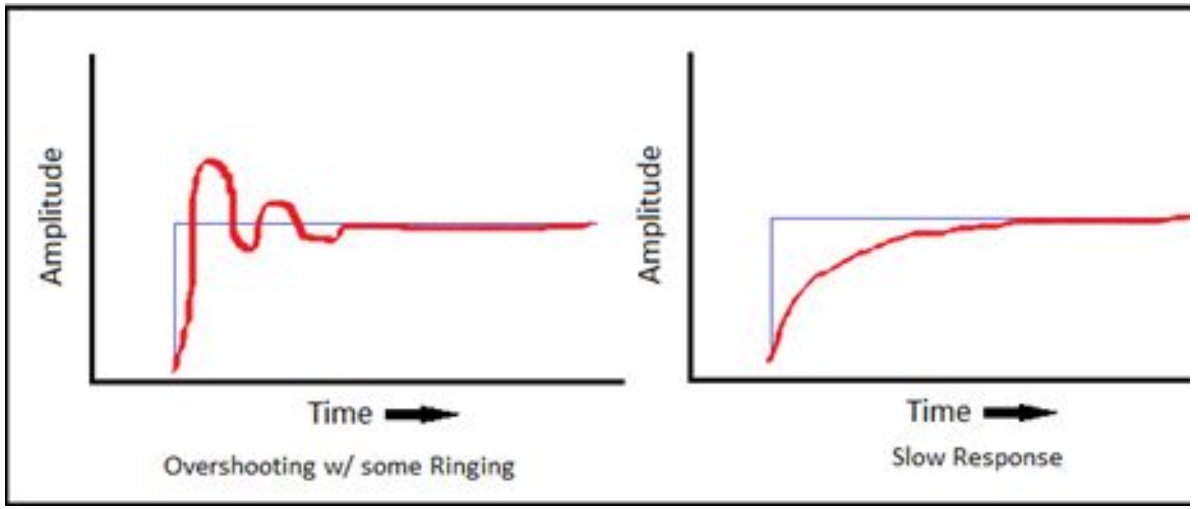
PID controller is widely used in industrial control systems and a variety of other applications requiring modulated control.

Error = desired val – actual val
Integral = pIntegral + error * time

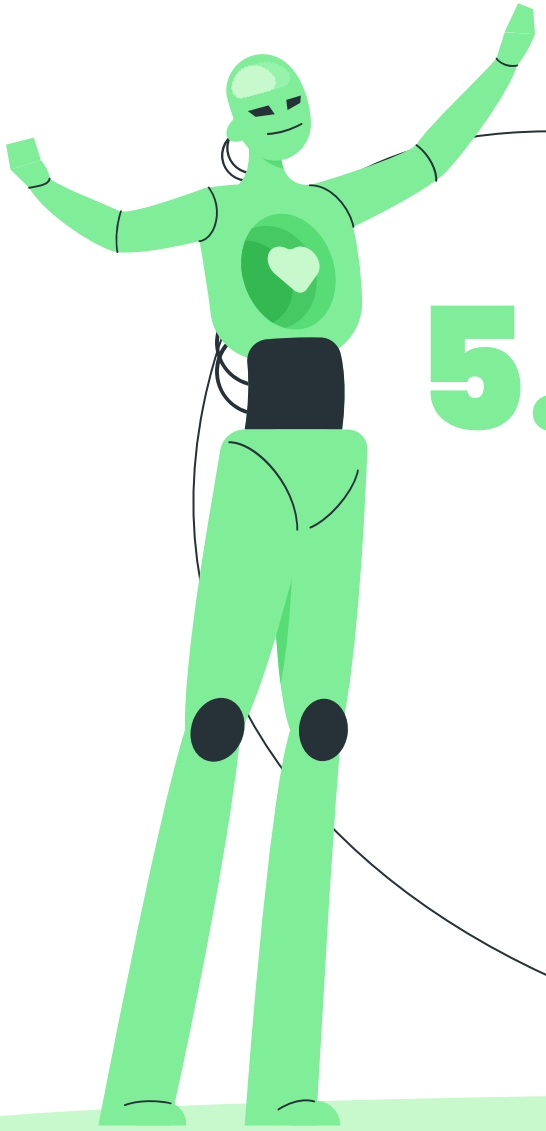
Moving forward and backward

If area > max
    move back
If area < min
    move forward





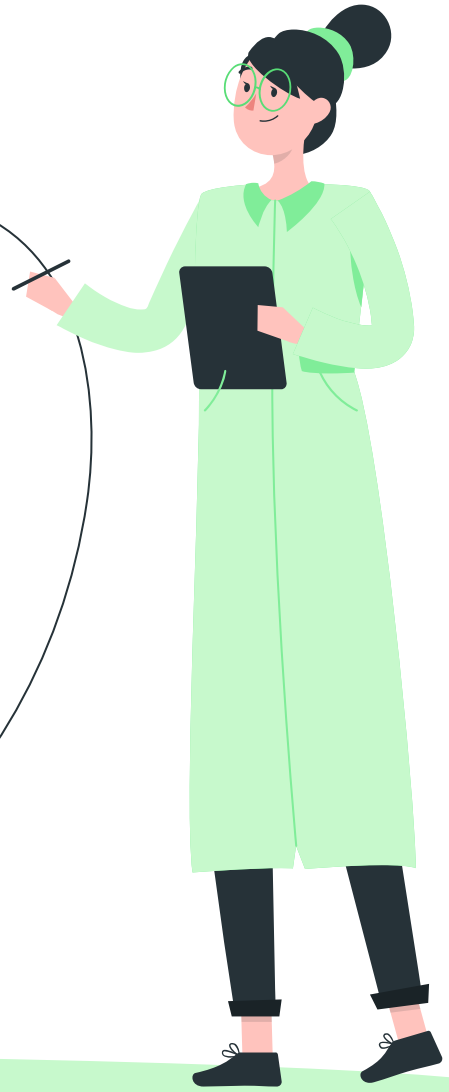
Working of our project



5. Send commands according to the gesture detected

Both palms	->	Drone lands
Both fists	->	Drone takes selfie
Left palm	->	Drone moves left
Right palm	->	Drone moves right

and this is just the beginning.....



FACE TRACKING MODE

To Infinity and Beyond...

Create an improved tracker using Reinforcement Learning

The drone receives rewards for successfully reaching the center of the frame, and gets bonus points for reaching in the shortest time possible

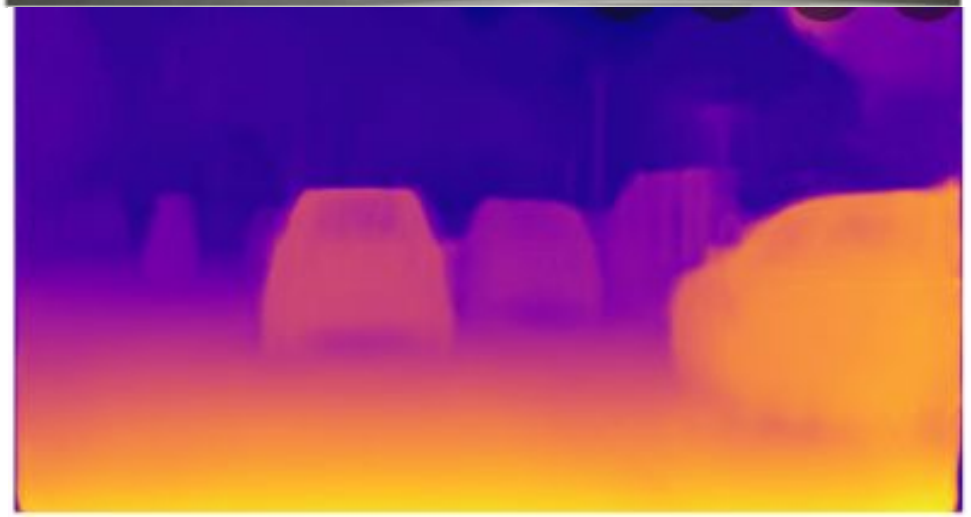
Improved gesture recognition

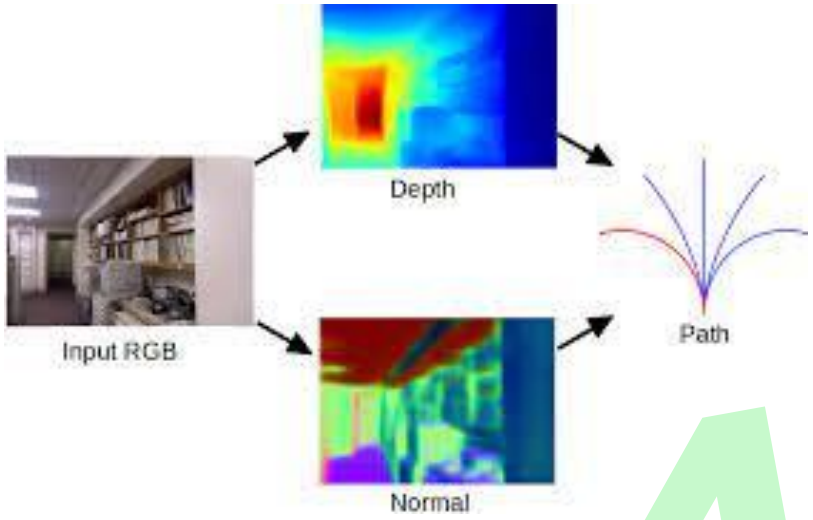
Create a better gesture recognition model using YOLO to improve the speed and accuracy of the project.

Mobile App for portability

You don't want to take your laptop with you when you take your drone for cycling... hence, a mobile app is necessary.

Next semester goals





Obstacle avoidance using Monocular Depth estimation

Thank You

Any Questions?