## Code files:

**Pi side:**

1. Vid_stream2.py: creates a localhost server using flask
2. Raspi Code: program to run hexapod combined with receiving gesture control value from propeller

**User side:**

1. EncodedGestureControl.c : Gesture control algorithm
2. integratedSend.py : Read simpleIDE terminal and send the recorded value to pi
3. objdet.py : Receive live stream and do conduct detection on received stream

## Demo files:

1. **Project demo:**

It can be seen that the different hand gestures are being mapped to the different movements of the robot
In the “idle state” valued at 1 - the robots leg is on the ground
In the “front movement” valued at 2 - the leg of the robot is lifted
In the “front right” valued at 3 - the leg is moved to the right
In the “front left” movement values at 4 - the leg is moved to the left

*Encoder logic*

State = flex1*18 + flex2*9 + x*3 + y

| Movement | Flex 1 (*18) | Flex 2 (*9) | X (*3) | Y (*1) | Encoded value | Final value |
|---|---|---|---|---|---|---|
| Idle | 0 | 0 | 1 | 1 | 4 | 1 |
| Front | 1 | 0 | 1 | 1 | 22 | 2 |
| Front right | 1 | 0 | 0 | 1 | 19 | 3 |
| Front left | 1 | 0 | 2 | 1 | 25 | 4 |
| Back | 0 | 1 | 1 | 1 | 13 | 5 |
| Back right | 0 | 1 | 0 | 1 | 10 | 6 |
| Back left | 0 | 1 | 2 | 1 | 16 | 7 |
| Bounce | 0 | 0 | 1 | 0 | 3 | 8 |
| Kill | 1 | 1 | 1 | 1 | 31 | 9 |

The laptop is reading the value from the serial terminal on simpleIDE and sends the value to pi which then receives the value and performs the motion.

2. **Object detection**

The object detection algorithm is run on the live stream received from the picam and shown to the user.

3. **Challenges**

This shows the video of our servos very strongly moving to the assigned positions but immediately zeroing in onto the robot.

**Update and challenges we faced:**

One of the major issues we noticed was the power supply we used to drive the servos. We are using 18 MG996R servos, Their working current ranges from 500mA to 1.4A (stall current). we were providing insufficient current to the servos as we were initially since our voltage regulator capped at a maximum amperage of 3A.

We also faced an issue due to the large amount of current draw that the wires were burning. So we used larger wires for the wiring. We rectified the power supply by powering raspberry pi using a voltage divider but powering the servos directly from the 7.4V 1800mAh batteries.

Individual servos were running well but upon connecting all the overvoltage kept causing the servos to brown-out and reset. (this can be seen in the “Challenges” video uploaded in the demo folder)

Given the limited time we tried to make a makeshift voltage regulator using resistors but this proved ineffective. We then decided to use a single voltage regulator and create a proof of concept for the one leg with 3 servos and showing our communication channels and other features.

We plan to add a voltage regulator that is suitable for our circuit and get our robot moving in due time and would love to share our progress with you.