PID report

Sharad Sridhar, Sriram Kalluri

Method for localizing landmark:

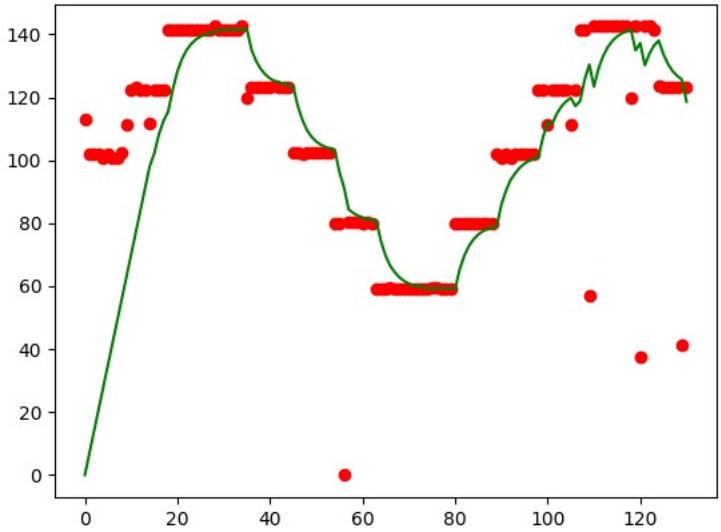
We are detecting the red and green led lights, through a simple color detection.
We are using a mask for filtering out green/red objects and applying it to the image captured from the webcam. To make this process easier, we are placing these lights in front of a white background.

Advantages:
-It is very simple to implement
-It can be used in the absence of clutter (of similarly colored objects).

Disadvantages:
-It is extremely sensitive to changes in the shade of the color.
-Even slight amounts of intensity changes leads to non-detection scenarios.
-It is not particularly robust in general.
-It cannot work well in cases where similarly colored objects are placed (in this particular pipeline). We have encountered all these things and update our code to meet the requirement.

Hand Shake Communication protocol:
We are using Pyserial for communication.
Connecting the arduino board on a particular COM port, we run our communication through that. There is an initial amount of buffer added to the code, since the 'serial' function call for initialization resets the board in a Windows machine. We are sending the error values as integers to the arduino board, which in turn updates its angle.
The change in angle takes into account, any additions or subtractions that might be needed to get the servo B to the same angle as that of servo A.
Once this is done, the board sends a confirmation message back to the pid controller running on the PC.
This approach is similar to that of a handshake protocol to verify the data being exchanged.

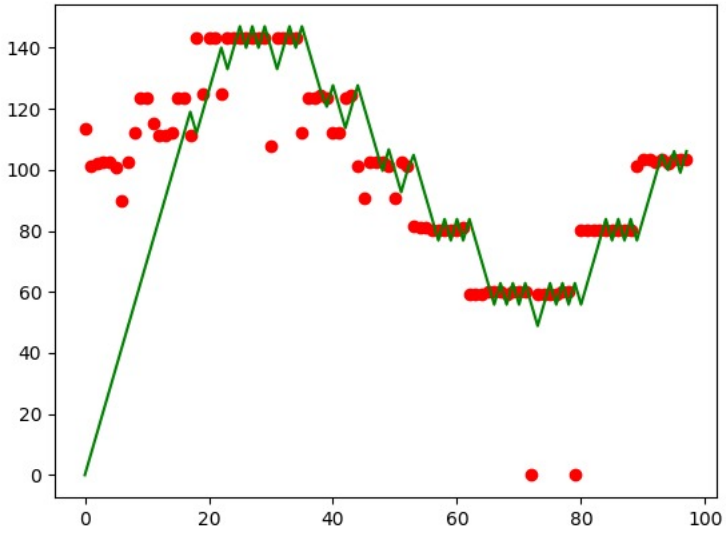
PID Output images:
We have implemented the PID controller as a function of angular velocity and checking whether the servo B has reached the desired position through serial communication.
During the execution we changed the values of the Kp, Ki and Kd values and ran for the set of the requested things. In all the below images, the **red dots** represent servo motor A and the **green line** represents servo motor B.



Kp = 0.2

Red – Servo A
Green- Servo B

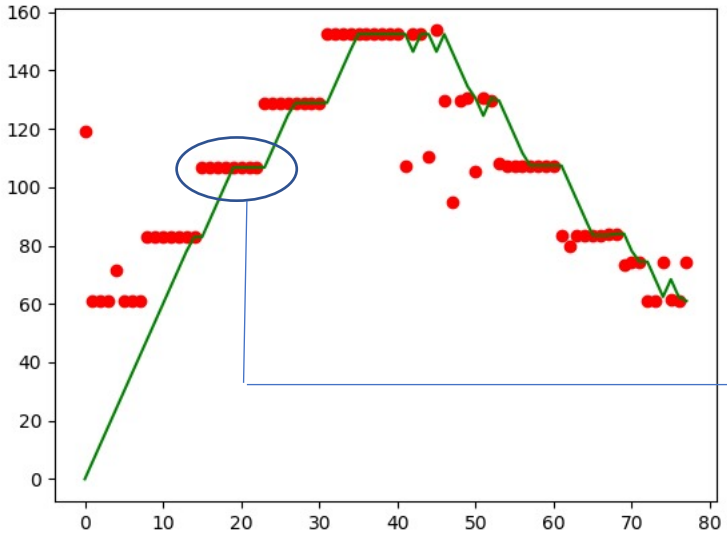
In the above image, where the parameter Kp has low value, we can see that the servo B angle converges to the servo A angle very slowly. This is to be expected as the error being multiplied with a small value, would lead the servo B to increasing/decreasing the angular velocity very slowly.



Kp = 3
Red – Servo A
Green- Servo B

In the above case, the angle converges quite fast, but this method has its own drawback. Due to a large rate of change, there will always be scenarios where the servo B overshoots or undershoots the target angle. This leads to some oscillation around the target angle.

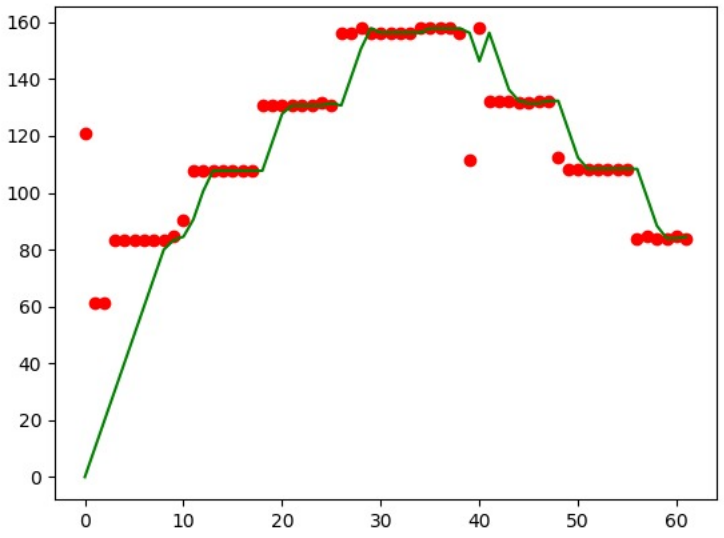
In the below case, the Integral component stabilizes the result.



Kp = 0.2
Ki = 0.2
Red – Servo A
Green- Servo B

Here the green line is exactly in middle of the red which removes the offset. Check the first P picture for the offset whether the green is nit in middle

Offset is a sustained error that cannot be eliminated by proportional control alone. PI controller is used to eliminate the offset. Similar to the result with a smaller value of P but with the removal of the offset, the I component will result in a better solution as the convergence once done, does not oscillate anymore.



Kp = 0.3
Ki = 0.2
Kd = 0.05
Red – Servo A
Green- Servo B

The differential term Kd takes the difference in errors in angular velocities and it speeds up the process of convergence a bit and we have observed the best fit curve for the below optimal values i.e Kp =0.3 Ki=0.2 and Kd =0.05.