Testing the impact of Speed

The Test

Using one of the blue ping-pong balls, I created a test where the goal of it was to see how well my code is able to track the ball in different depths, as well as seeing how well it could track the speed of the ball in these depths.

Besides the test, I also researched on the speed a ping-pong ball throughout an average game of ping-pong, to be able to have a proper criteria to compare the results of the test to.

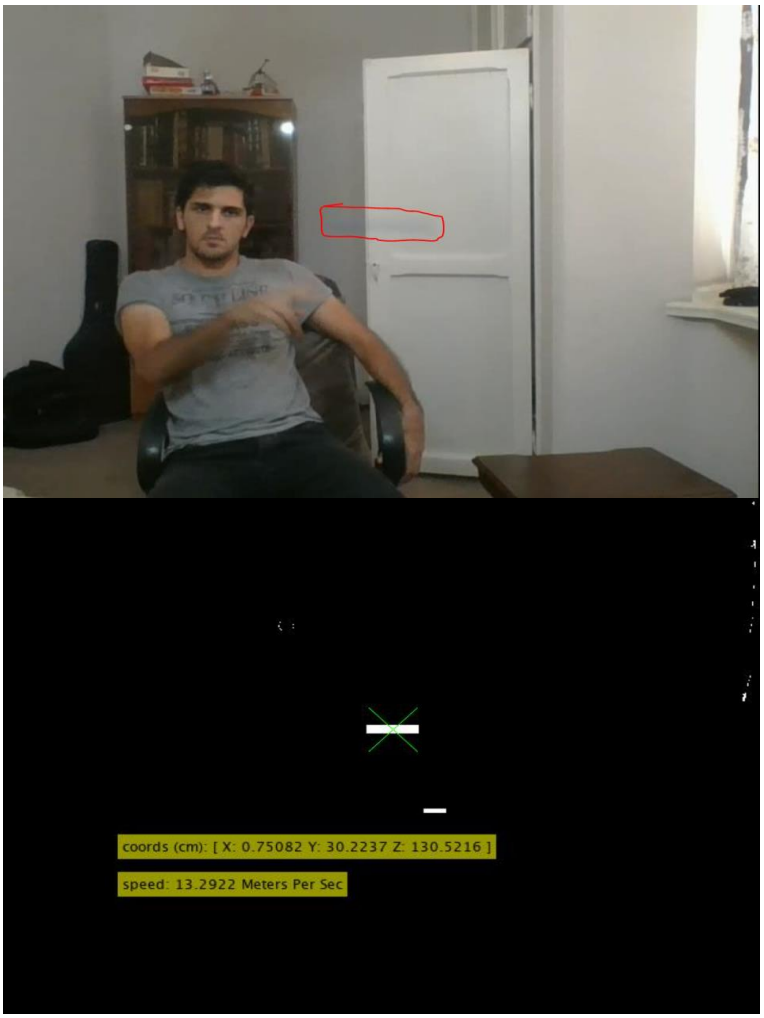
The result of this research is as follows:

Speed of Ball	Grouping
Below 3mps	Very Slow shot
3mps to 6mps	Slow shot
6mps to 11mps	Average shot
11mps to 15mps	Fast shot
15mps to 22mps	Very fast shot
22mps to 30mps	Insane shot (professional / world records)

From this research, we can conclude that we expect to reach between 11mps and 15mps for a fast shot in a normal game, which we'll try to achieve in this test.

The Result

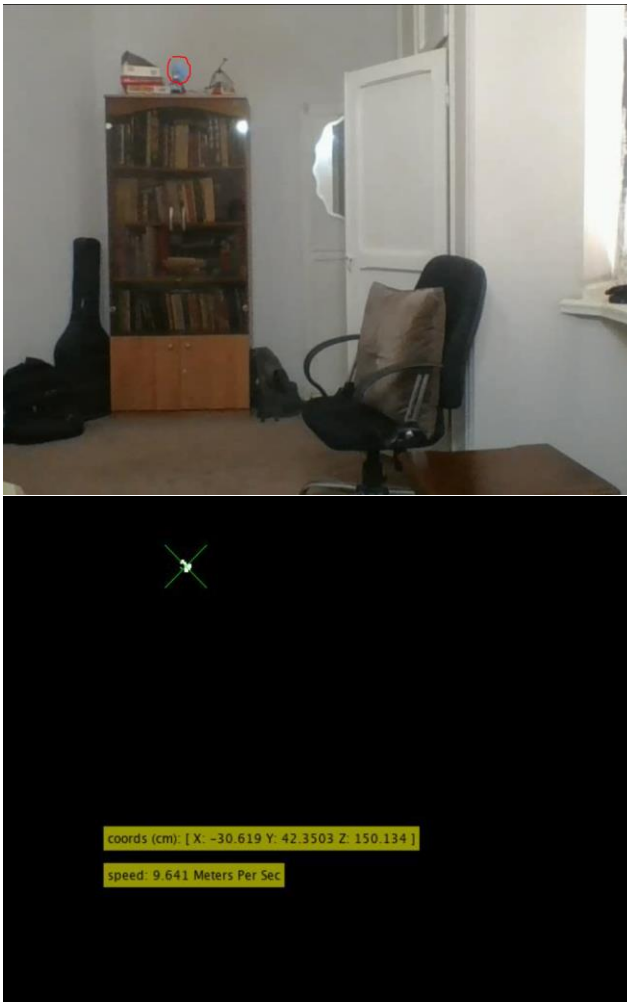
The first scenario I considered was me having a distance of 1.25 meters from the cameras, and then throwing the ball as fast as I could horizontally.



As it can be seen from this test, the ball is moving at the speed of around

13.5 Meters per Second, which is around the speed we expect a fast shot in an average ping-pong game to be able to achieve.
From this test, It can also be concluded that my code has no problem tracking the ball at such speeds.

For the second scenario, I threw the ball from the location of the cameras away from them, in a slightly upward motion, in order to test if my code is able to track the ball in farther distances with a high speed.



After performing this test, my current color thresholder was able to track the ball's location and speed at the depth of around 2.5 meters. This however, can be greatly improved given a better lighting and having a normal orange ping-pong ball (It seems that blue is harder to track at fast speeds when the background is white), since I have previously shown that my code is able track a normal ping-pong ball in an average game with ease.

Extra Work

I added the ability to select the number of frames to consider for each second into my code, so that instead of comparing each subsequent frame to its previous frame in order to find speed, it compares to the previous selected number's frame (which is controlled by a parameter).

For example:

```
FTC = 10.0; %frames to consider per second
```

Considering our cameras are 30fps, this will make it so that the speed is calculated by comparing the distance of the current frame with the frame that is 3 frames before it.

*Note that for all the results I obtained above, I set FTC to be equal to FPS (so each frame is considered)