ENPM 673: Perception for Autonomous Robots
Project 3

Shreya Gummadi
Sneha Ganesh Nayak
Ghanem Jamal Eddine

7 April 2019

1 Data Preparation

In this part of the project, we first extracted frames from the video. We got 200 frames, we divided them into Training and Testing frames in a 70-30 ratio. The training frames were then cropped to get only the buoy areas of the frame. We did this for the red, green and yellow buoy for each frame. To get the cropped frames we developed a GUI and used the Event Handling functionality of matplotlib. The code can be found in generate_frames.py.

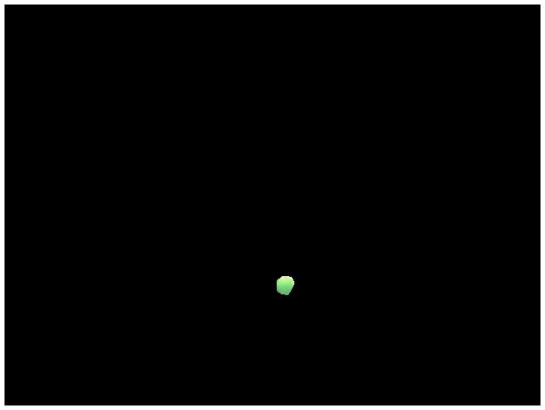

Fig 1. Cropped Green Buoy.

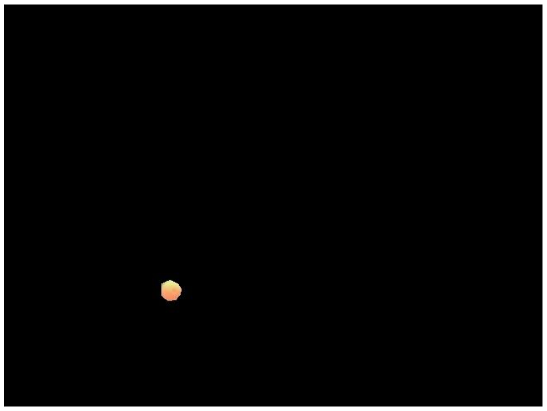

Fig 2. Cropped Red Buoy.

Fig 3. Cropped Yellow Buoy.

2 Average Histogram

In this part, we computed the average histogram for the red, yellow and green buoy. We did this in order to see the average color distribution for each buoy. We compute the histogram for each frame for each buoy and at the end average them. The histograms are in the RGB space. The average histogram then gives the average distribution of each channel in a particular colored buoy. The code for this part can be found in average_hist.py. The code returns the distribution of each buoy which is used in the next part of the project.

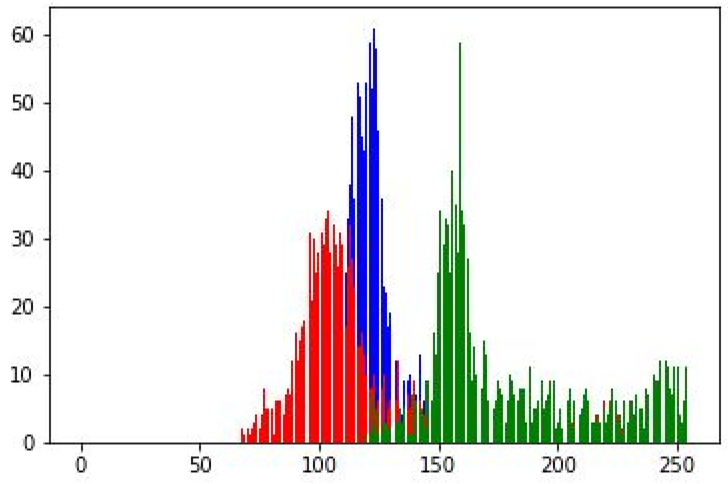

Fig 4. Average Histogram for Green Buoy.

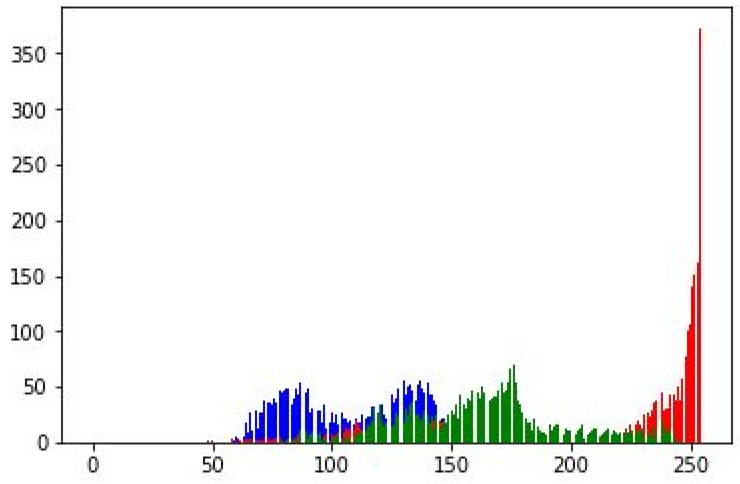

Fig 5. Average Histogram for Red Buoy.

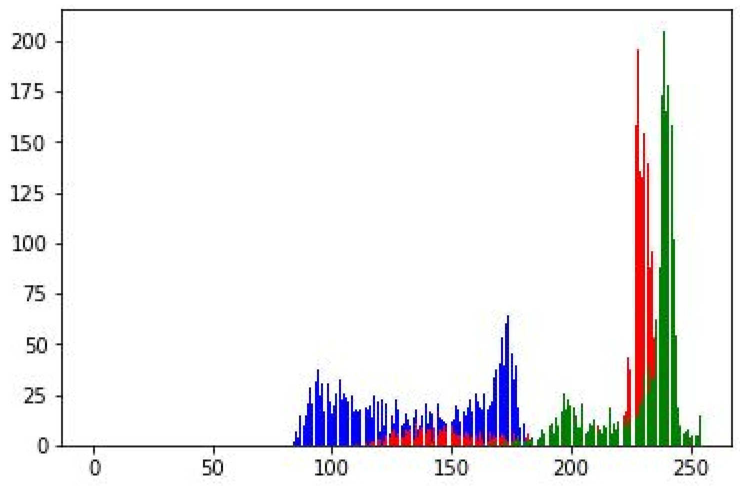

Fig 6. Average Histogram for Yellow Buoy.

The average histograms are plotted for color pixel vs frequency. From figure 4, we can then understand that for the Green buoy we need to look at the Green channel. From figure 5, we can see that for Red buoy we need to look at the Red channel. However from fig 6, we can conclude that for the Yellow Buoy we need to look at the mean of the Red and Green channels.

3 1D Gaussians

In this part, we use the distributions of the red, yellow and green buoy that we got in the previous section. We get the mean and standard deviation of the distributions and plug it into the 1D gaussian function to get the 1D gaussian for each buoy, fig 7. We use the gaussians to segment and detect the buoys.
We compute the probability of the pixels with the gaussian and use a threshold to get the pixels corresponding to the buoy. In our case, we used the probability of the pixel to be greater than 0.97 to say that it belonged to the particular buoy. We generated a binary mask using this and used it to get the contour and the circle around the buoy and then mapped it back to the frame. The code for this can be found in 1D_gaussian.py.

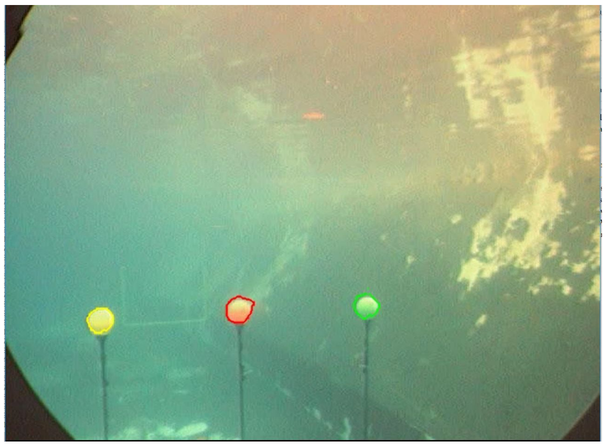

Fig 7. Buoys detected using 1D Gaussian.

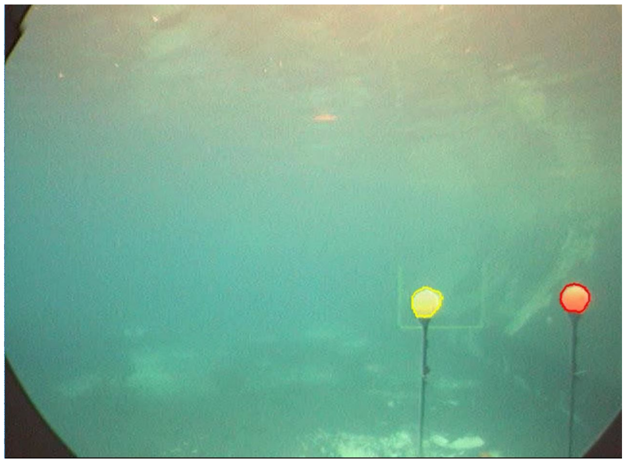

Fig 8. Buoys detected using 1D Gaussian.

4 Expectation Maximization

In this part, we implemented the Expectation Maximization. EM assumes random components and computes for each point a probability of being generated by each component of the model. Then, it tweaks the parameters to maximize the likelihood of the data given those assignments.
The program takes the number of gaussians and data as its input and returns the mean and standard deviation for each gaussian. The code for this can be found in ‘part1.py’ file.
Taking 3 random Means and 3 Sigma values, we get initial 3 1D Gaussians as follows:

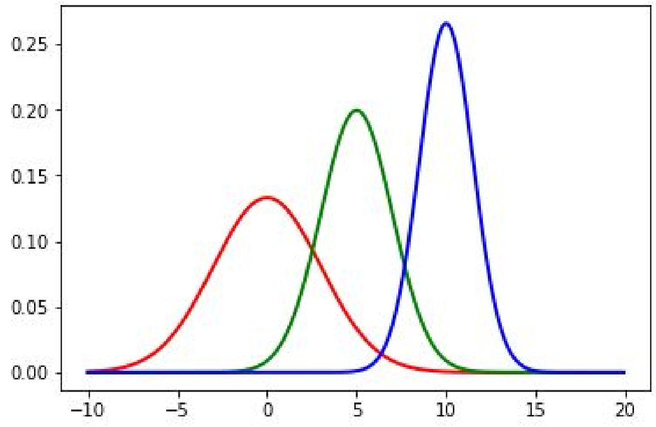

Fig 9. 3 1D Gaussians.

We then generate 10 samples of Multivariate normal random numbers, from the mean and sigma values to find 1D Gaussians with K=3 and K=4 using Gaussian Mixture fit and Expectation Maximization, and then compare the two.

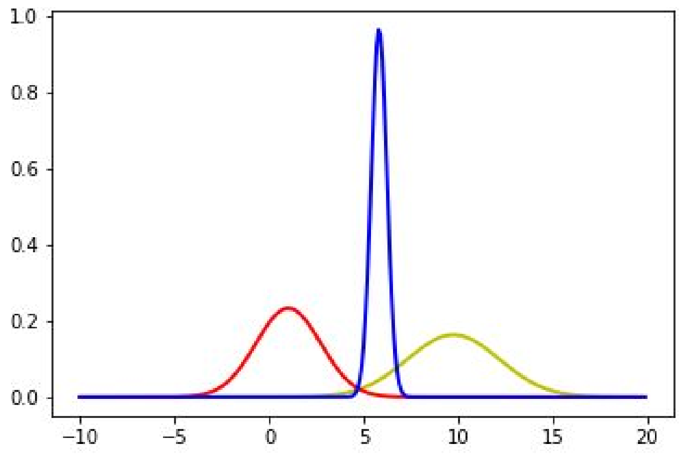

Fig 10. 1D Gaussian using Gaussian Mixture Fit, k=3

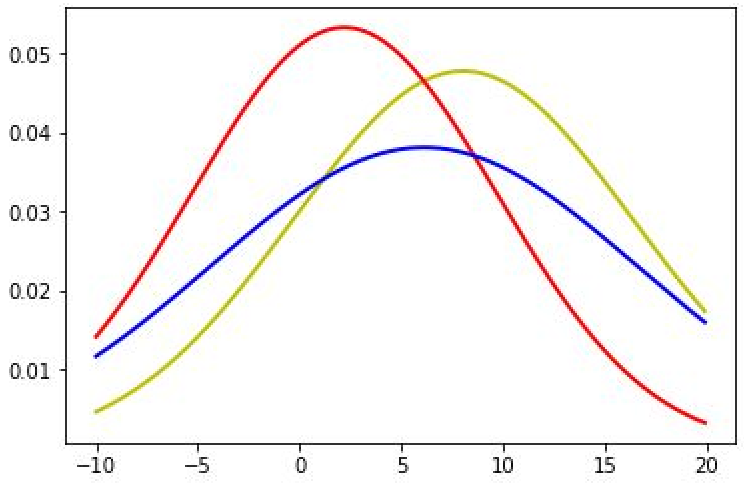

Fig 11. 1D Gaussians using EM, k=3

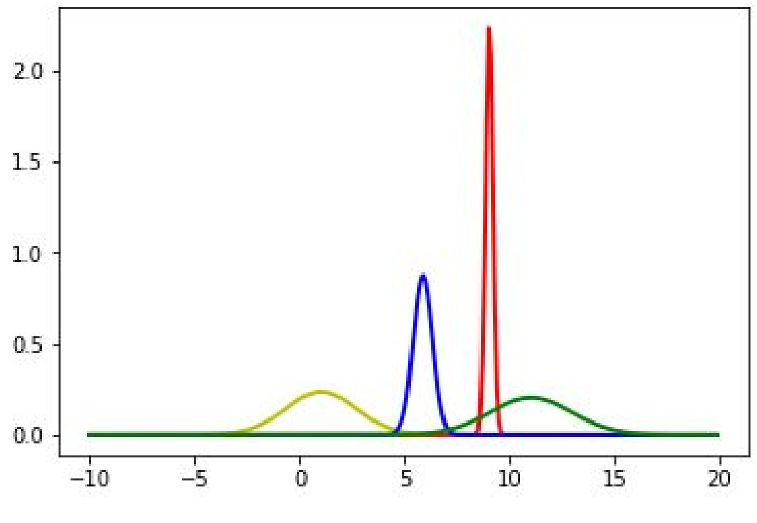

Fig 12. 1D Gaussians using Gaussian Mixture Fit, k=4

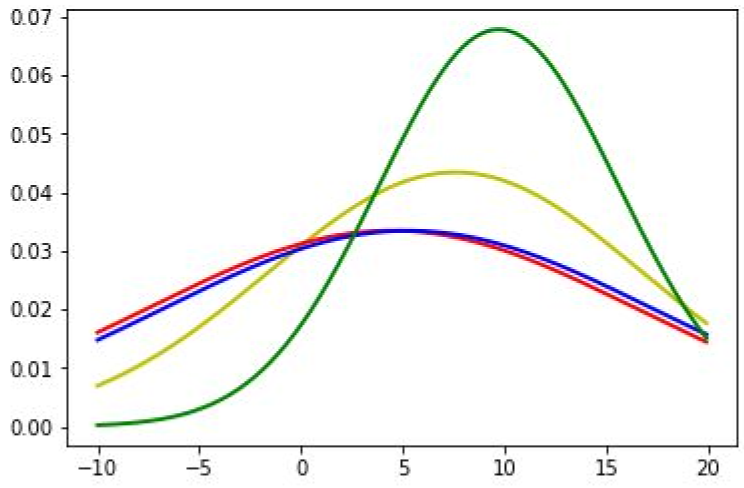
Fig 13. 1D Gaussian using EM, k=4

5 Color Model Learning

From the histograms computed in section 2, we can conclude the number of gaussian that would be best to segment and detect the buoys.

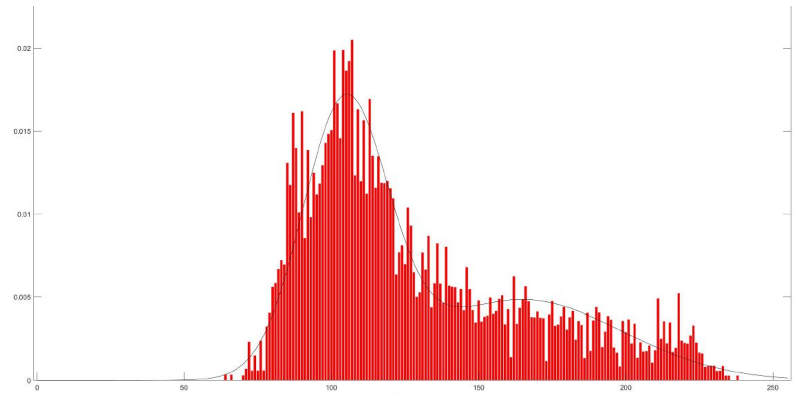
Fig 14. Gaussian for the Green Buoy.

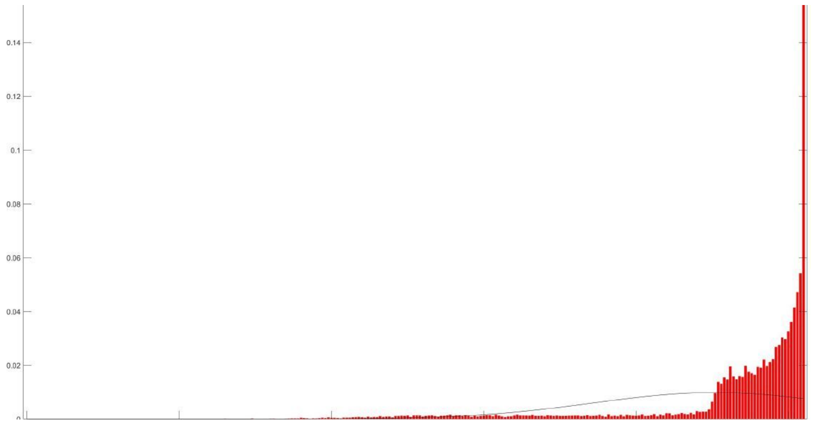
Fig 15. Gaussian for Red Buoy.

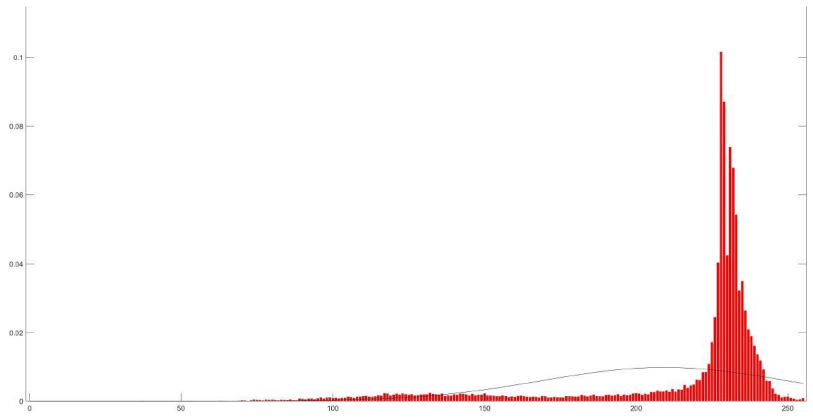

Fig 16. Gaussian for Yellow Buoy.

So in this part of the project, we use ND gaussians for the buoys and get the mean and standard deviation from the expectation maximization algorithm which is done in part2.py. Using this we segment the red, green and yellow buoy and detect them using the same framework as that used in the 1D segmentation and detection part. This gives us a better result than the 1D part. The code for this can be found in part3.py.

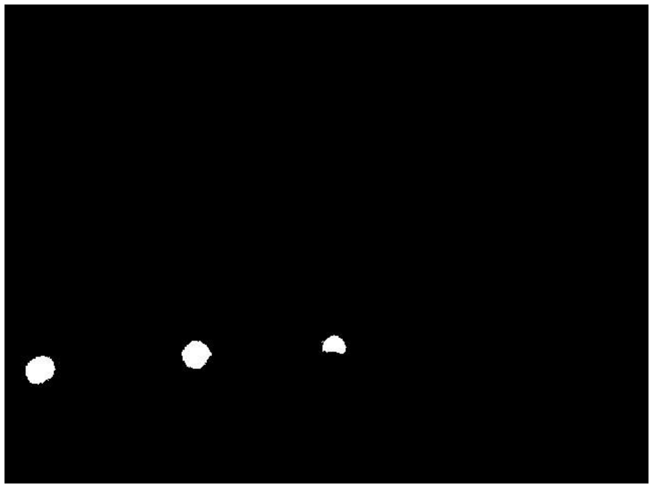

Fig 17. Binary Mask after segmentation.

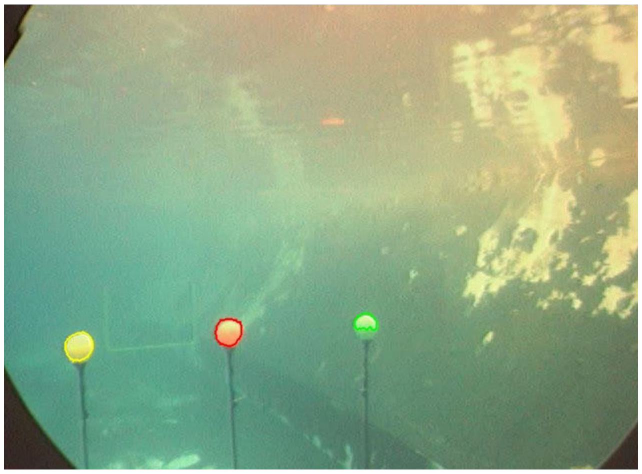
Fig 18. Buoys detected using EM and 3D Gaussian.

6 Further Analysis

Our project has been implemented in the RGB color space. Instead, we can use other representations like HSV to do this. In case of the HSV color space, we could get better results as the hue would stay relatively constant for the buoy while the intensity of color in the RGB space keeps varying. The difference in brightness and color can be handled by the value and saturation channels.