Perception for Autonomous Robots-ENPM673

Report

By:

1. Sanket Acharya

2. Mithun Bharadwaj

3. Abhiram Dapke

INDEX

1. AIM OF THE PROJECT:

The aim of this project is to detect the buoys from the video sequence as input and generate colour-segmented binary images and compute the contours and centers of the buoys. This involves color segmentation by fitting a GMM to the RGB histogram to generate a color model. One of the issues is that conventional segmentation techniques will not work in such an environment as noise and varying light intensities will make hard-coded thresholds ineffective.

2. INTRODUCTION:

A buoy is a distinctively shaped and colored floating device, anchored to the bottom, for designating moorings, navigable channels, or obstructions in a water body. It is sometimes used as underwater markings for navigation. The data set for the project is a camera view of an underwater vehicle with three distinctly colored buoys in surrounding. The goal is to segment and detect the buoys in the given video sequence.

3. Gaussian Mixture Models (GMM): Gaussian mixture models are a probabilistic model for representing normally distributed subpopulations within an overall population. Mixture models in general don't require knowing which subpopulation a data point belongs to, allowing the model to learn the subpopulations automatically. Since subpopulation assignment is not known, this constitutes a form of unsupervised learning.

GMMs have been used for feature extraction from speech data and have also been used extensively in object tracking of multiple objects, where the number of mixture components and their means predict object locations at each frame in a video sequence.

4. Expectation maximization (EM): Expectation maximization is a numerical technique for maximum likelihood estimation and is usually used when closed form expressions for updating the model parameters can be calculated (which will be shown below). Expectation maximization is an iterative algorithm and has the convenient property that the maximum likelihood of the data strictly increases with each subsequent iteration, meaning it is guaranteed to approach a local maximum or saddle point.

It consists of two steps:

The first step, known as the expectation step or E step, consists of calculating the expectation of the component assignments for each data point.

The second step is known as the maximization step or M step, which gives the updates to be made to the parameters which is essentially maximizing the expectation calculated in the E step with respect to the model parameters.

EM is an iterative algorithm that converges to a local maximum. The calculations for Maximum Likelihood are not always straight forward but by using the EM method, we can reduce complex calculations to simple updates and just check for convergence ie when the difference between the updates is lesser than some defined threshold.

5. PROCEDURE:

The pipeline for segmentation implements the following steps:

1. Extracting the ROI: We need to develop a model for the color to be segmented. To do that, we need to extract the ROI with the color to be modeled. It is beneficial to get a tight bound on the area to be extracted to reduce the effect of the background in the parameter estimation.

2. Train and test data: The procedure to be implemented is essentially a learning process and as with all such processes, it is a good practice to split the data into training and testing data sets to avoid over-fitting and to check the validity of the parameters learned.

3. Histogram: Since we're using RGB color space, there are specific R,G and B channel values that contribute to the specific color model. The aim is to learn the parameters that define this color space. The first step towards this is to calculate the average histogram of the training images for the R,G and B channels separately.

4. GMM fitting: Based on the histogram, we can decide to fit the N number of Gaussian to it. We'll be using 1 dimensional Gaussian. Each Gaussian has 3 parameters associated with it.

- Mean
- Variance
- Weight

These can be estimated using the iterative EM algorithm. Once all the parameters are obtained, we have the color model for that particular buoy. All the above steps have to be done for all 3 buoys.

5. Segmentation: Using the color model, we can estimate the probability that each pixel belongs to the learned color model by using the intensities as inputs. By setting a threshold for the probability, we can classify each pixel as belonging to this color model or not. The group of pixels that satisfy the threshold give the segmented image.

Alternate Approach:

HSV Color Space As it has been concluded in Part3, the output may be improved by using a different color space. MATLAB provides libraries for 5 different color spaces. All these color spaces have been incorporated in the algorithm for computation for average histogram. But after studying the results from each color space, it was concluded that using HSV color space should improve the segmentation and detection of the buoy. Similar to Part 2 for RGB color space, I first applied all the possible models with K ranging from 1 to 5 on the images and decided upon the model based upon the quality of output per model. Note that, the data that has been used to generate the Gaussian models is in HSV color space. The final models used for green buoy, red buoy and yellow buoy are shown in Figure 19, Figure 20, and Figure 21 respectively.

6. USING DIFFERENT COLOUR SPACES:

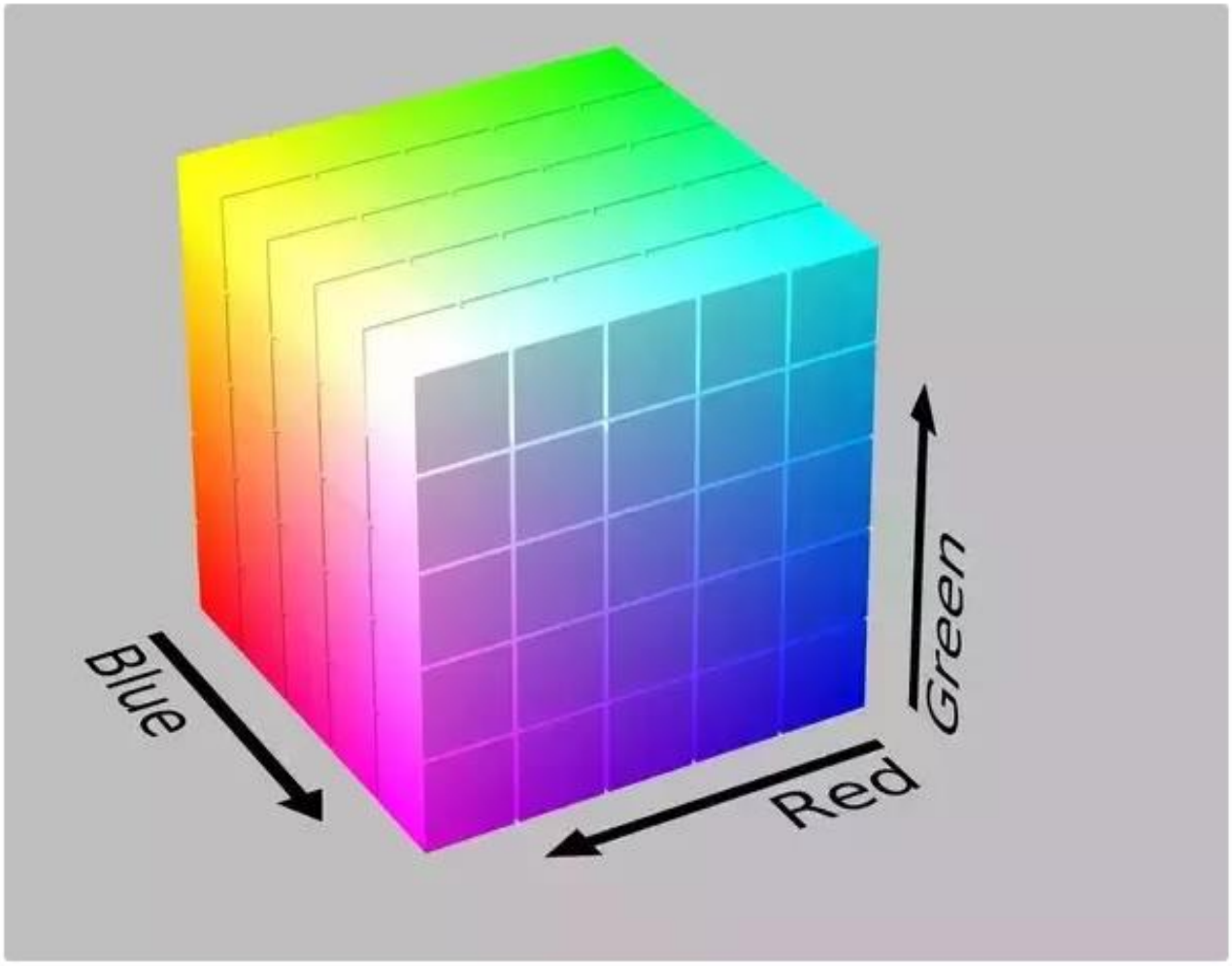
The different types of color spaces are:

1.RGB – It is an additive color space based on the RGB color model. RGB color can be understood by thinking of it as all possible colors that can be made from the combination of for red, green, and blue in varying proportions.

We use HSV color space more often in computer vision because unlike RGB, HSV separates luma, or the image intensity, from chroma or the color information. This is very useful in many applications. For example, if we want to do histogram equalization of a color image, we probably want to do that only on the intensity component, and leave the color components alone. Otherwise we will get very strange colors.

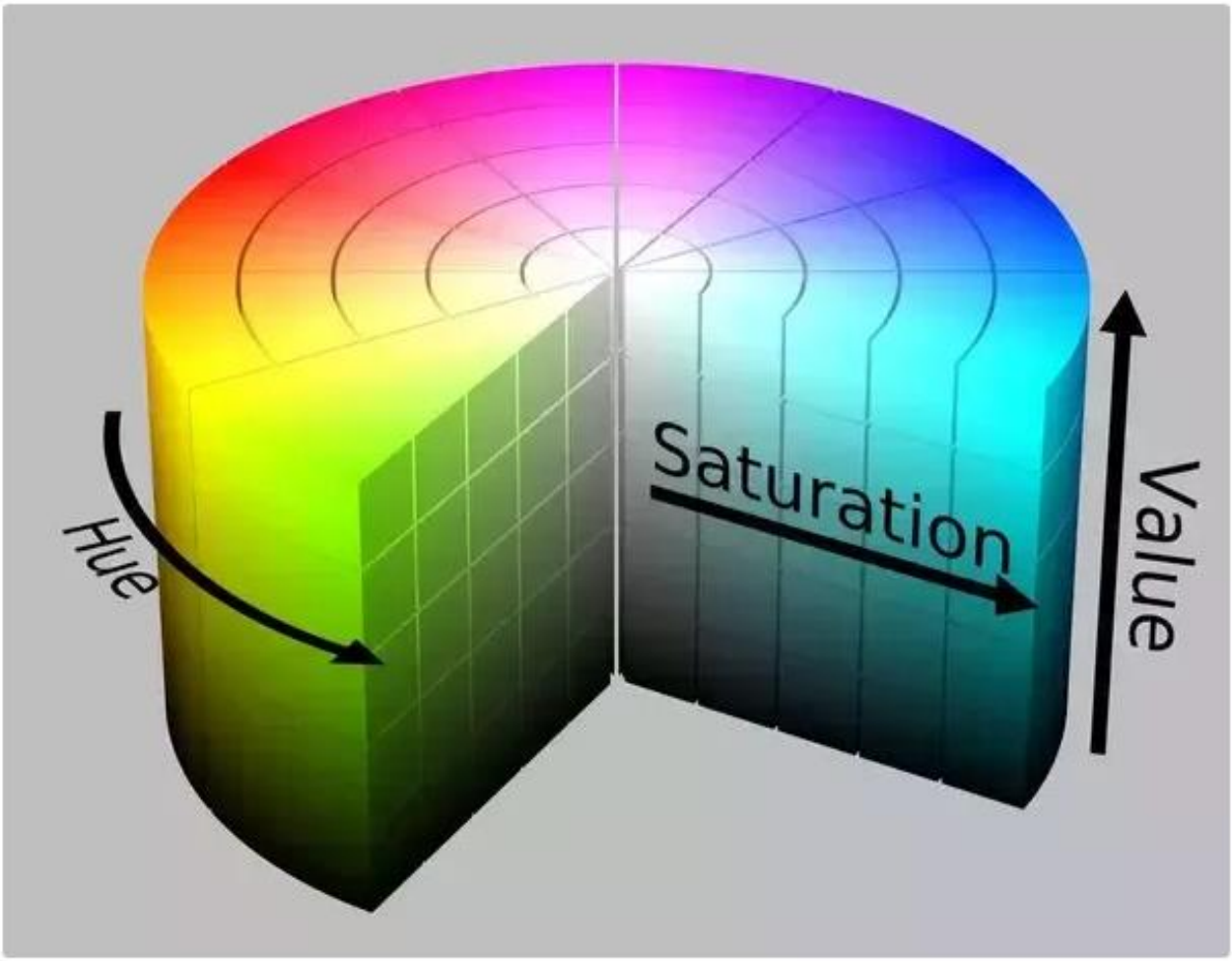
In computer vision we often want to separate color components from intensity for various reasons, such as robustness to lighting changes, or removing shadows.

One advantage of using the RGB color space is its simplicity. It's disadvantages are that it is non linear, device dependent, unintuitive and common.



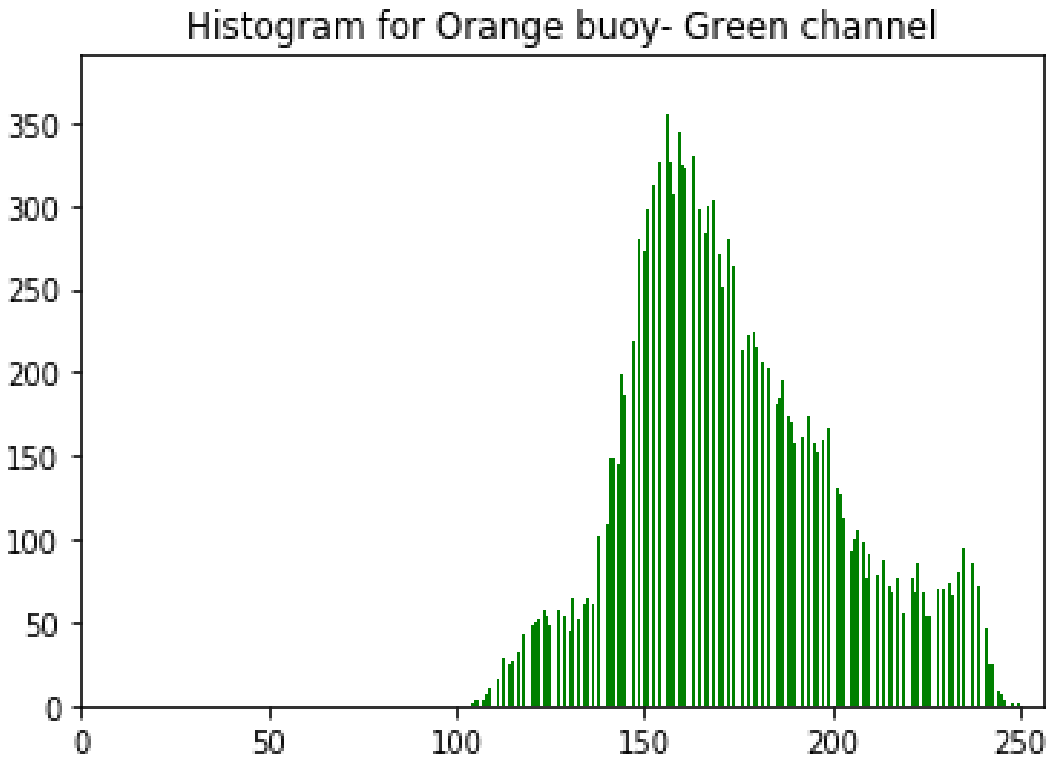
2.HSV and HSL - HSL (hue, saturation, lightness) and HSV (hue, saturation, value) are alternative representations of the RGB color model. In these models, colors of each hue are arranged in a radial slice, around a central axis of neutral colors which ranges from black at the bottom to white at the top. The HSV representation models the way paints of different colors mix together, with the saturation dimension resembling various shades of brightly colored paint, and the value dimension resembling the mixture of those paints with varying amounts of black or white paint.

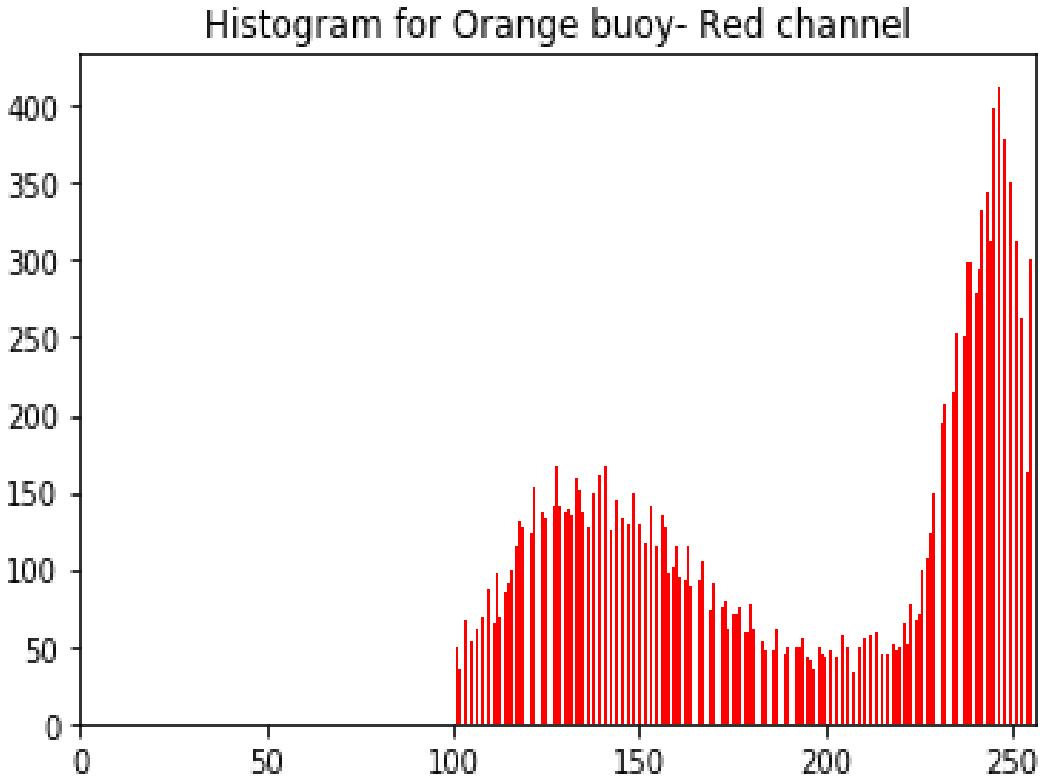
One advantage of using the HSV color space is that it separates the intensity from the chromaticity and represents them independently. Hue describes the position of the color in a 360◦ spectrum. Saturation describes the pureness of the color: it measures the difference between the color and a grayscale value of equal intensity.

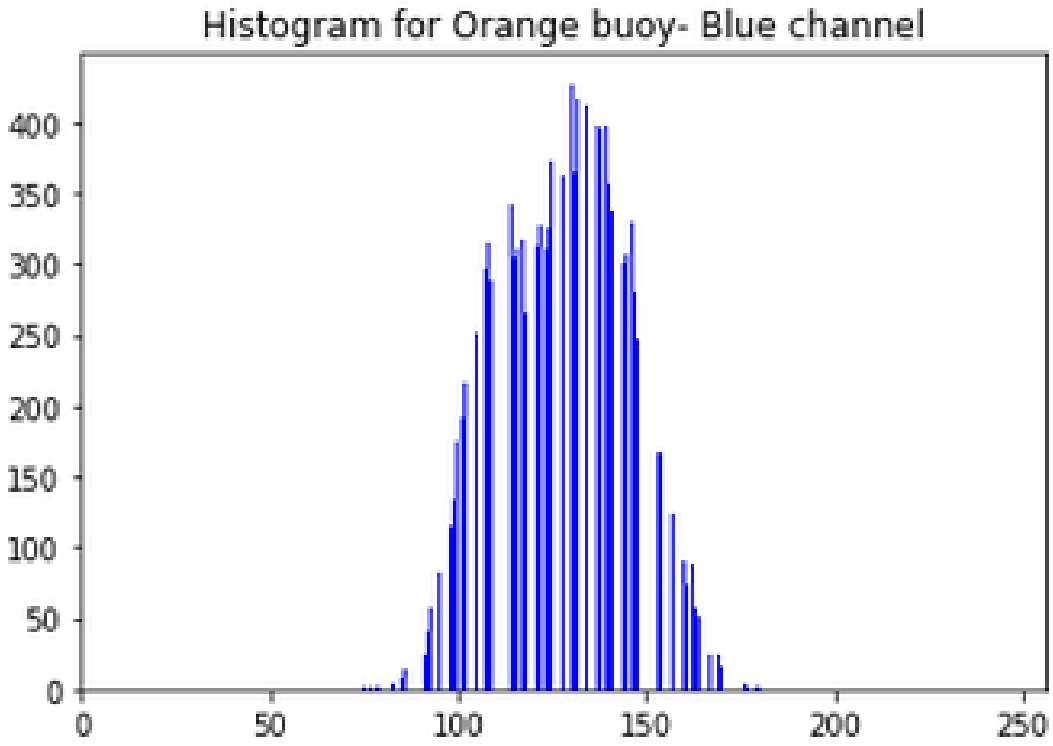


7. SCREENSHOTS OF HISTOGRAMS:

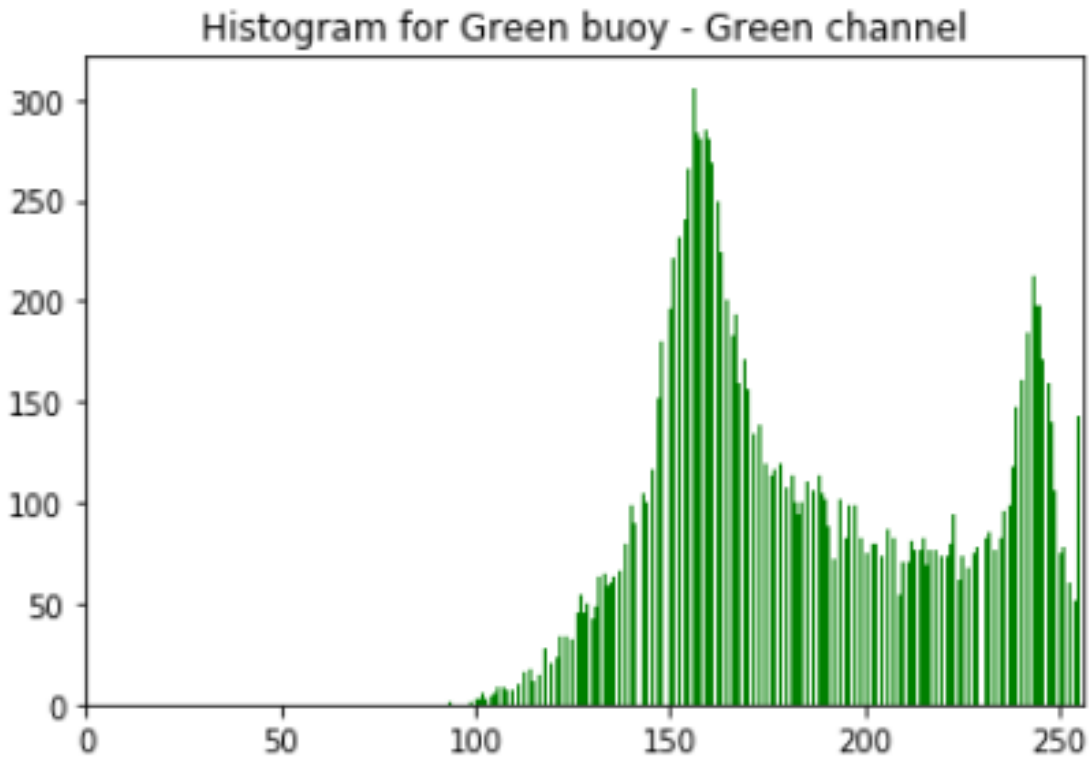
For Orange buoy:

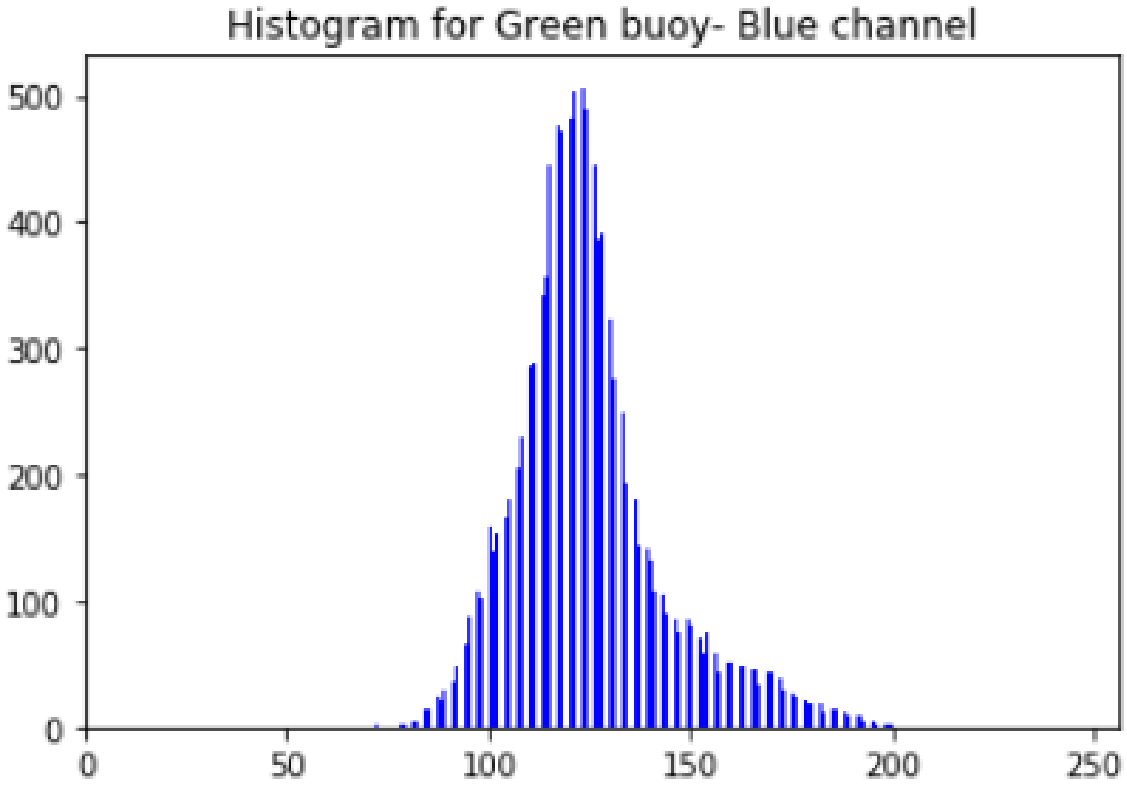



Histogram for Orange buoy- Red channel
400
350
300
250
200
150
100
50
0
0
50
100
150
200
250


Histogram for Orange buoy- Blue channel
400
350
300
250
200
150
100
50
0
0
50
100
150
200
250

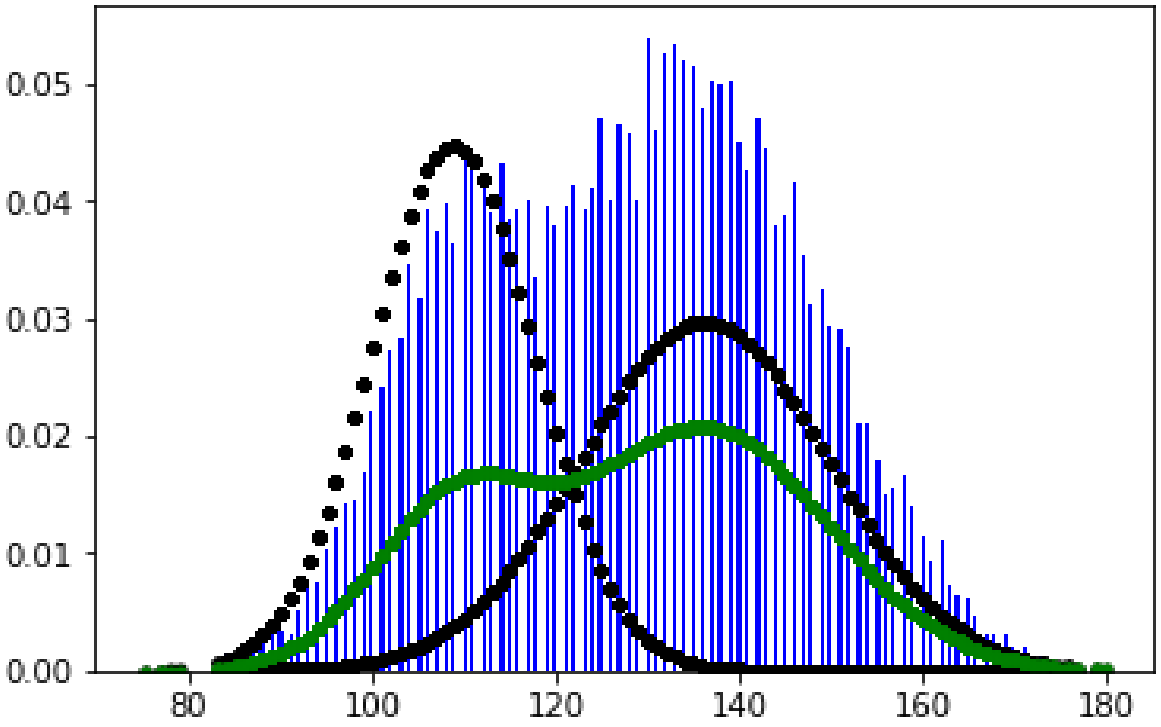
For Green buoy:

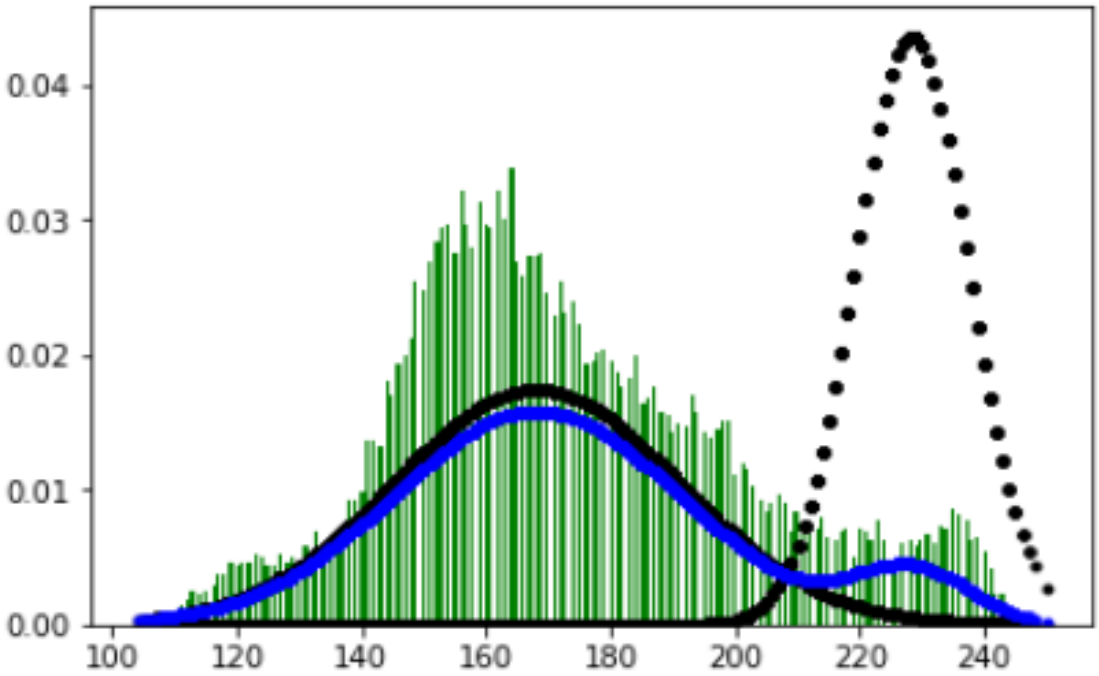



Histogram for Green buoy- Blue channel
500
400
300
200
100
0
0
50
100
150
200
250

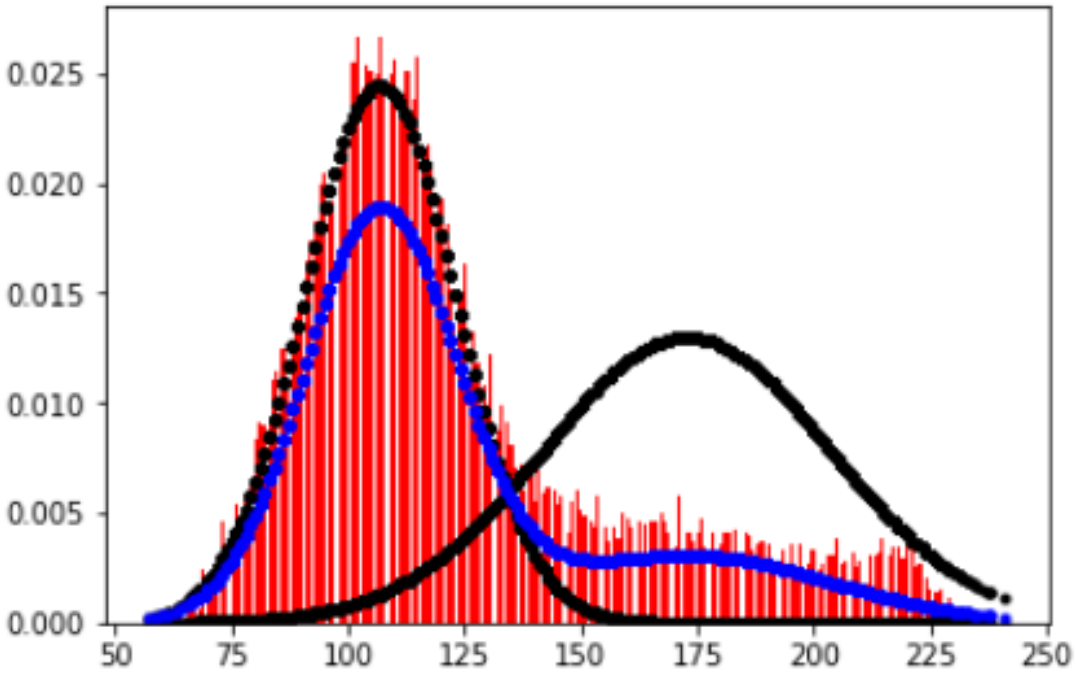
8. SCREENSHOTS OF GMM FITTING:

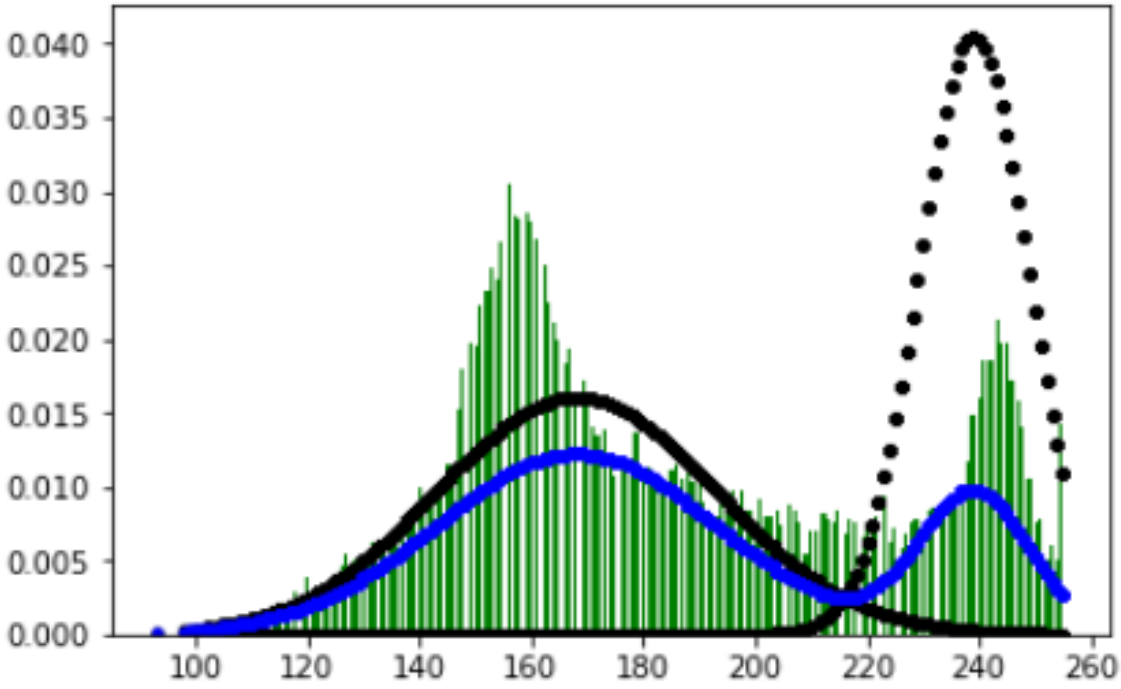
For Orange Buoy:

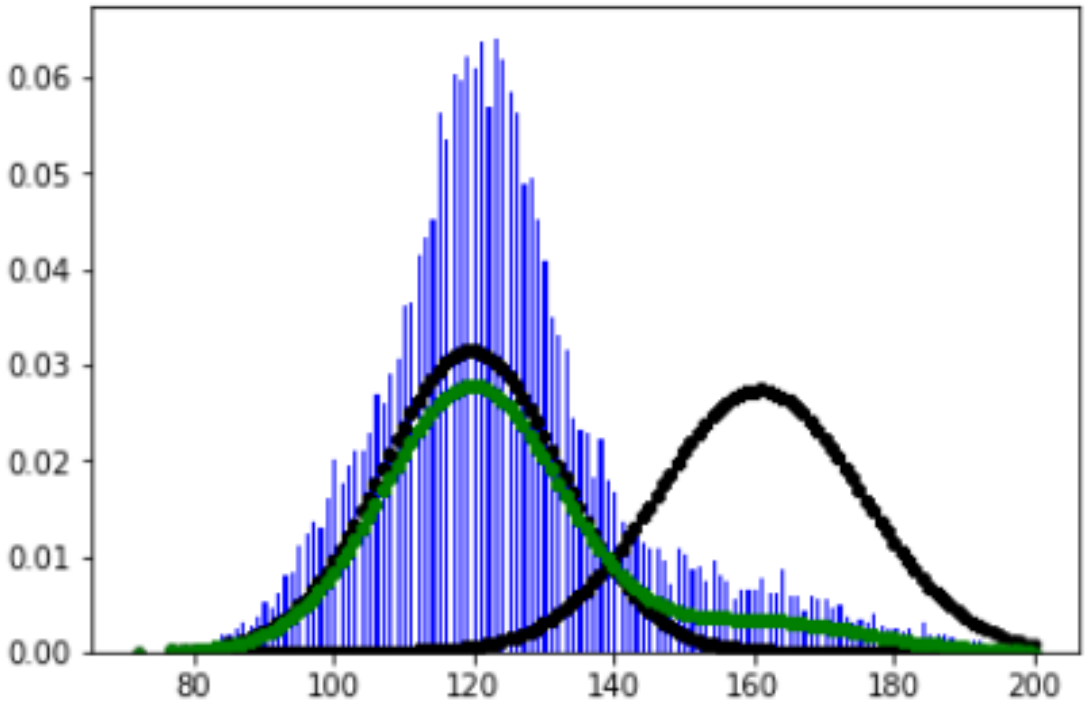




For Green Buoy:




0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
100
120
140
160
180
200
220
240
260


0.06
0.05
0.04
0.03
0.02
0.01
0.00
80
100
120
140
160
180
200

9. SCREENSHOTS OF OUTPUTS:

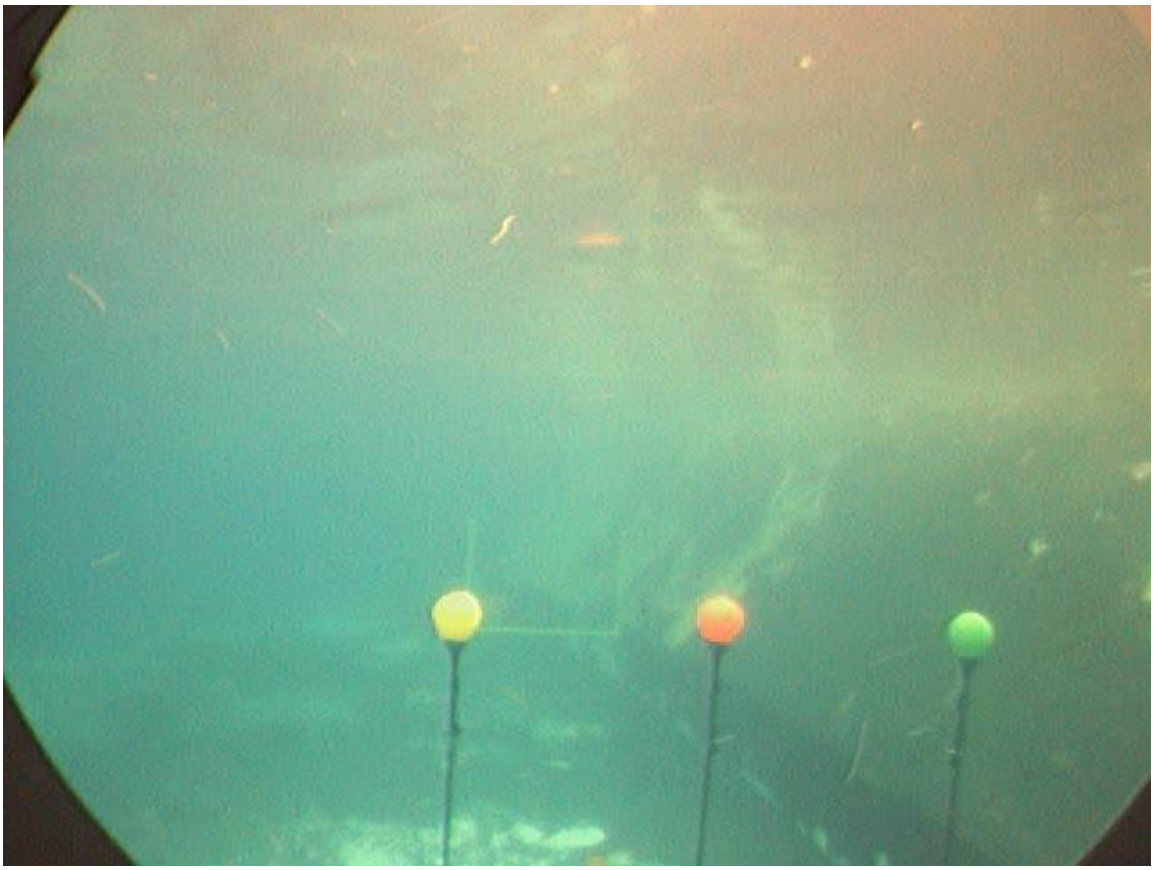


FIGURE: Input frame for yellow buoy segmentation

FIGURE: Binary image for segmented yellow buoy

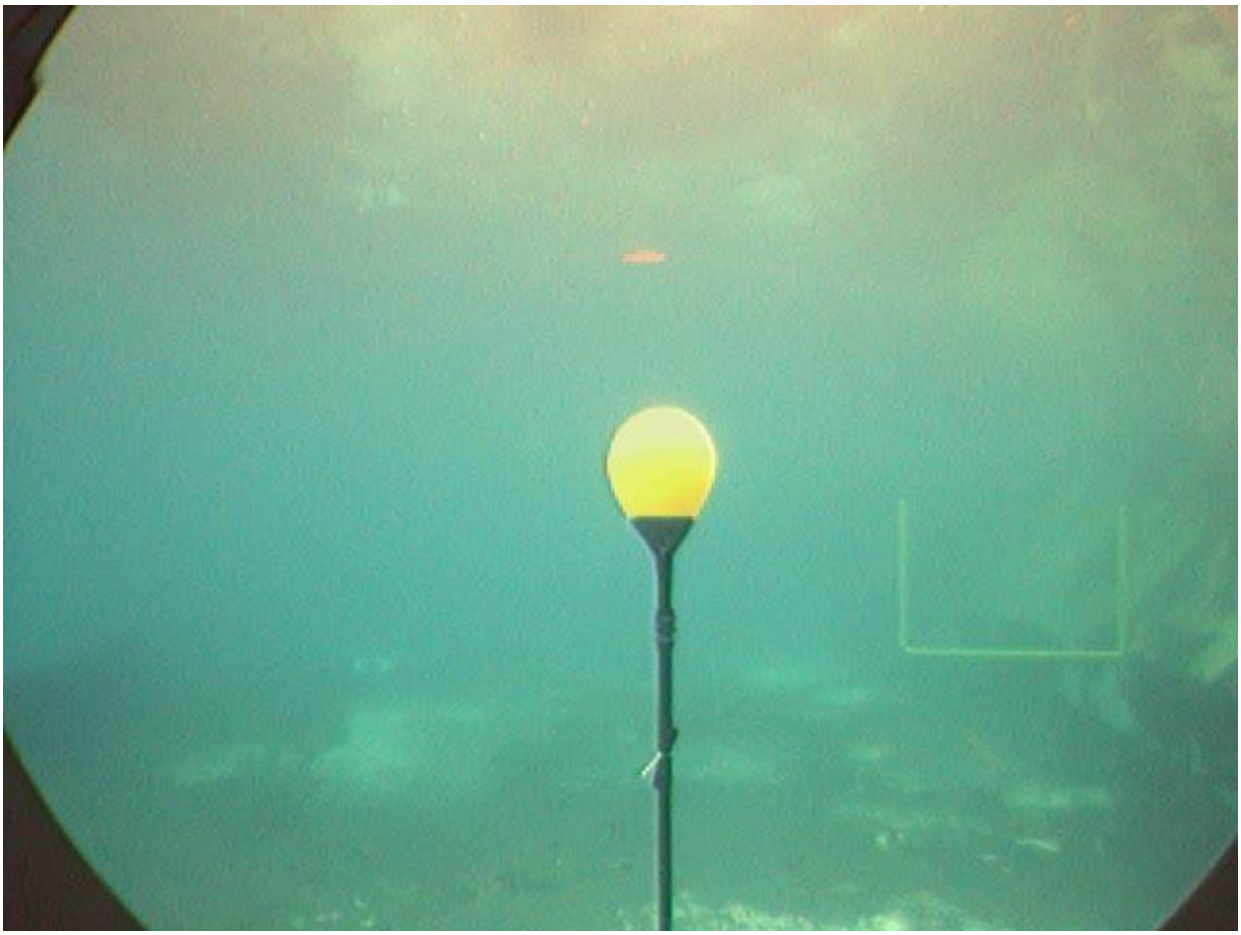


FIGURE: Input frame for yellow buoy segmentation

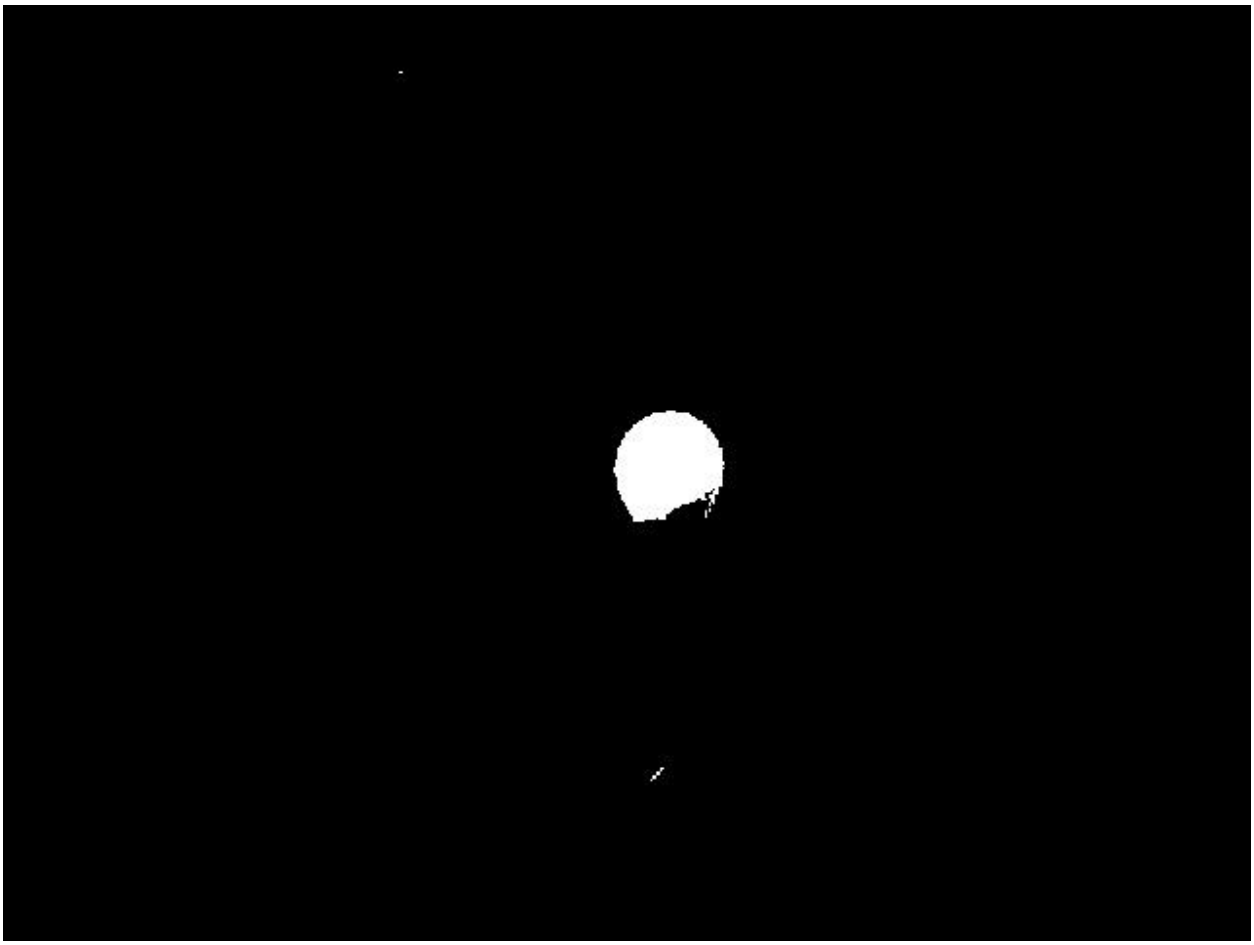


FIGURE: Binary image for segmented yellow buoy

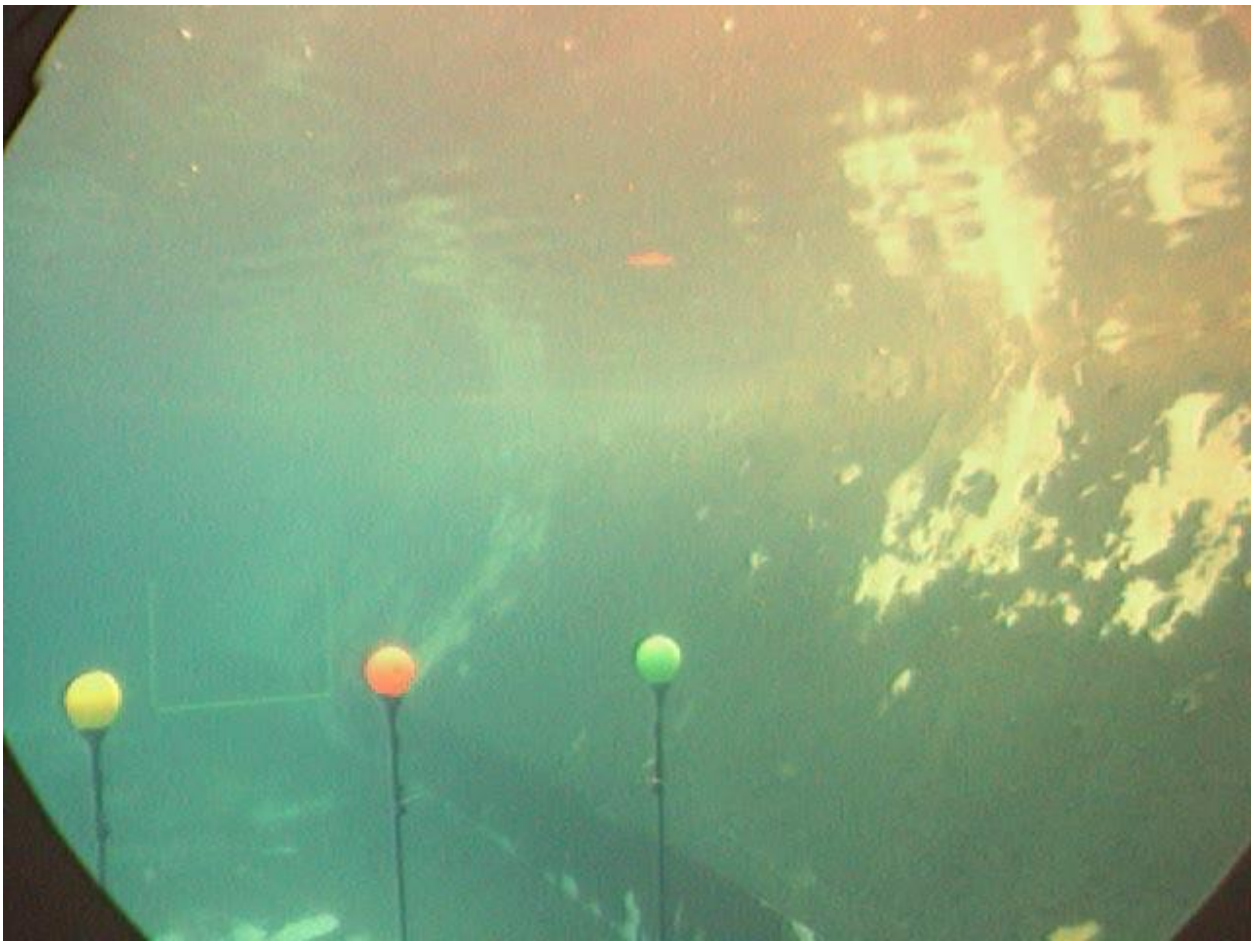


FIGURE: Input frame for green buoy segmentation

FIGURE: Binary image for segmented green buoy

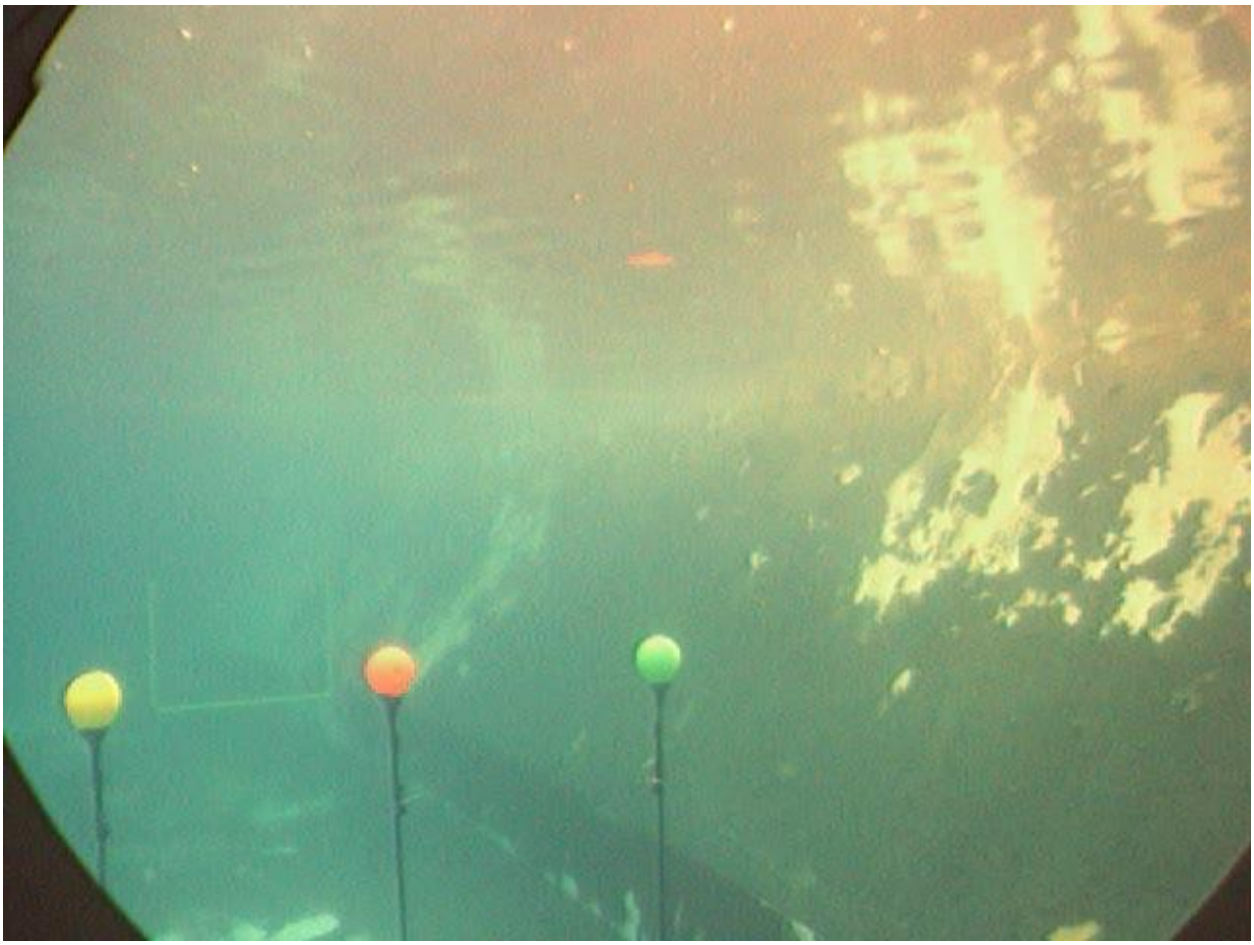


FIGURE: Input frame for green buoy segmentation

FIGURE: Binary image for segmented green buoy

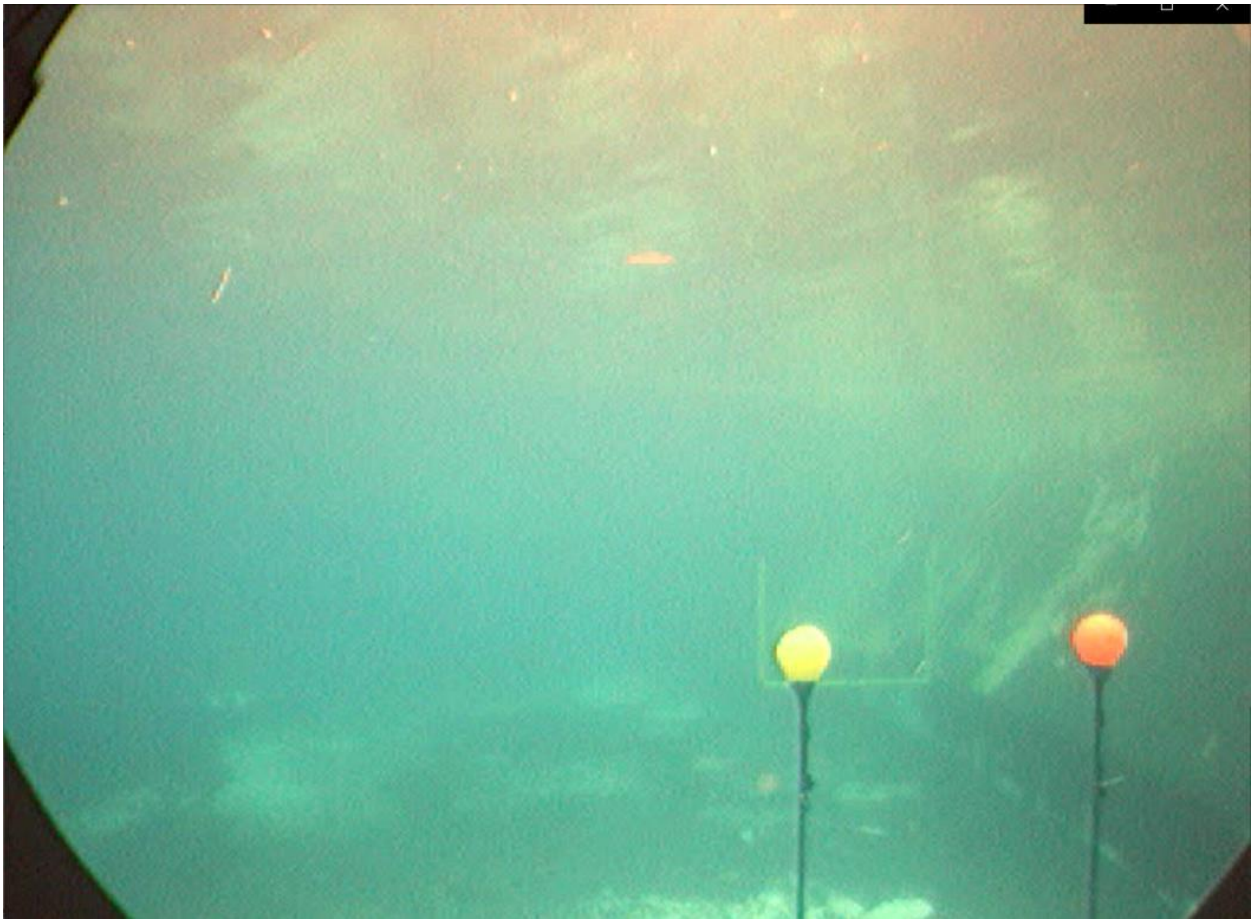


FIGURE: Input frame for orange buoy segmentation

FIGURE: Binary image for segmented orange buoy

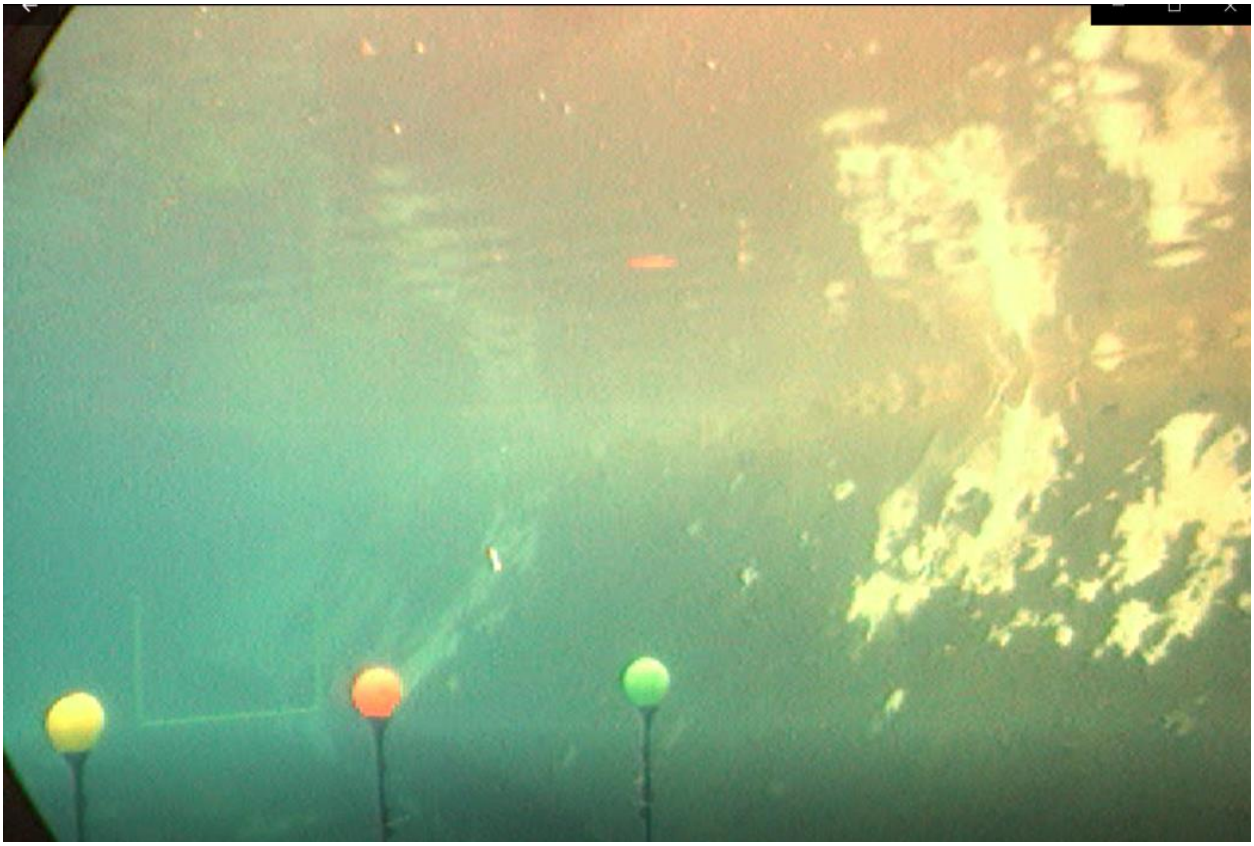


FIGURE: Input frame for orange buoy segmentation

FIGURE: Binary image for segmented orange buoy

10. ISSUES ENCOUNTERED WHILE SOLVING:

1. In RGB color space, if the illumination changes then the value of the color changes and then it causes difficulty in detecting while in HSV color space is immune to illumination changes.

2. If we don't have a tight bound when extracting the ROI, the background images add noise to the data we're trying to fit. This interferes with the exact color model we want to develop. This is apparent in the yellow buoy histograms.

3. Learning parameters accurately is an important step and this may not give the desired results when we have a low number of training data or when the data is noisy.

4. Setting the threshold for the color model is crucial for segmentation. This decides the deviation form the model that the data can have. Too tight a bound and it will detect only parts of the buoy, too loose and it classifies the neighboring pixels also as belonging to the model.

5. While using the command FindContours in python, we are getting this error:

Error: Unsupported format or combination of formats ([Start]FindContours supports only CV_8UC1 images when mode != CV_RETR_FLOODFILL otherwise supports CV_32SC1 images only) in cvStartFindContours_Impl

This was resolved by using cv2.cvtcolor(frame,cv2.color_BGR2GRAY).

11. REFERENCES:

1. Bishop, Christopher M., Pattern Recognition and Machine Learning (Information Science and Statistics), Springer-Verlag New York, Inc., Secaucus, NJ, 2006.
2. E-M, http://www.cs.umd.edu/ djacobs/CMSC828seg/EM.pdf
3. Opencv documentation
4. Matplotlib documentation