ENPM 673 - Perception of Autonomous Robots - Project 3

Pradeep Gopal
Sahana Anbazhagan
Srikumar Muralidharan

April 1, 2020

1 Introduction

The objective of this project is to teach us the concept of Color segmentation using Gaussian Maximization techniques. We are provided with a video that has been recorded underwater. We are shown 3 individual spherical buoys, colored Green, Yellow and Orange respectively. Varying light intensities and noises underwater render existing threshold techniques ineffective, hence we have to create a set of tightly segmented image of each buoy from the given video and use this data to understand buoy color distribution. After his, we segment each of the buoy using the mean and standard deviation values obtained and place a ring around the buoys in our video.

2 Data Generation

From the sample video, we extract images, frame by frame, for further training. In order to crop out the buoy from each frame, we make use of **cv2.selectROI()** function. We use mouse clicks and enter to select our region of interest and create our dataset for Orange, Green and Yellow respectively. Using this data set, we further divide data into training and testing datasets respectively in a 70-30 ratio, as we employ a learning based method. All operations are further carried out on the training data set and finally tested on the test dataset and segmentation is applied for the sample video in the final stage.

3 Color Histogram Visualization

3.1 Green buoys

We read the frames from the training dataset folder for cropped green buoy images. The images have been cropped as we restrict our ROI to minimise Noise. For each frame, we calculate Histogram values for each channel and stack them. Finally, after all frames have been parsed, we calculate average histogram value for each channel based on the number of images and plot them. The Histogram plot for green buoys is as shown in figure 1.
From this image, it is clear that dominant spread among RGB channels in the histogram averages belong to the green channel, as expected. Hence, we use the values of G channel for Mean and Standard deviation calculation. Taking the last frame into consideration, we try to calculate the mean and standard deviations for the green channel (column 2 in BGR channels) and come up with a bell-curve plot to show the distribution over the green channel. The mean and standard deviation calculation is carried out using the function **cv2.meanStdDev()** function, which is used on the final image of training dataset. The Gaussian PDF for the bell curve is formulated as follows:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{1}$$

Figure 2 is an accurate representation of this concept.

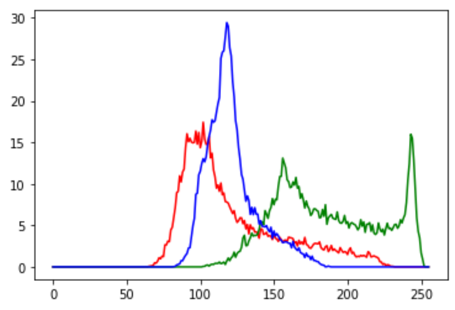

Figure 1: Histogram for Green buoy

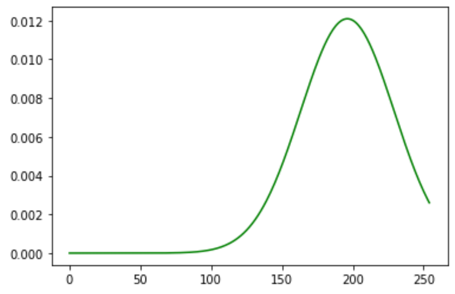

Figure 2: Bell curve for Green buoy training dataset

3.2 Orange buoys

We read the frames from the training dataset folder for cropped orange buoy images. The images have been cropped as we restrict our ROI to minimise Noise. For each frame, we calculate Histogram values for each channel and stack them. Finally, after all frames have been parsed, we calculate average histogram value for each channel based on the number of images and plot them. The Histogram plot for Orange buoys is as shown in figure 3.

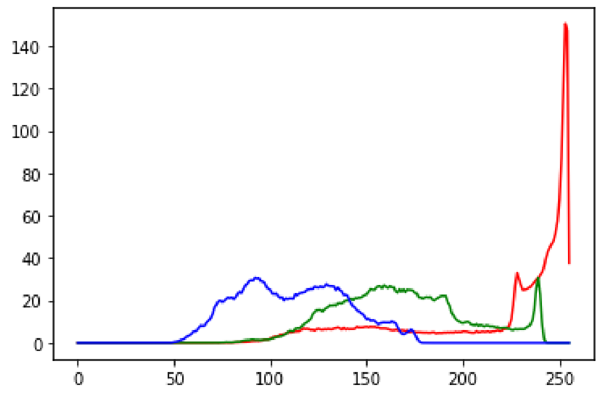

Figure 3: Histogram for Orange buoy

As expected, it is clear that dominant spread among RGB channels in the histogram averages predominantly belong to the red channel, since orange is an extenstion of the red channel and is slightly influenced by the green channel as well. But for the sake of simplicity and because we do not have Red buoys, we use the values of R channel for Mean and Standard deviation calculation. Taking the last frame into consideration, we try to calculate the mean and standard deviations for the red channel (column 3 in BGR channels) and come up with a bell-curve plot to show the distribution over the red channel. The mean and standard deviation calculation is carried out using the function **cv2.meanStdDev()** function, which is used on the final image of training dataset. The gaussian PDF is the same as equation 1. Figure 4 is an accurate representation of this concept.

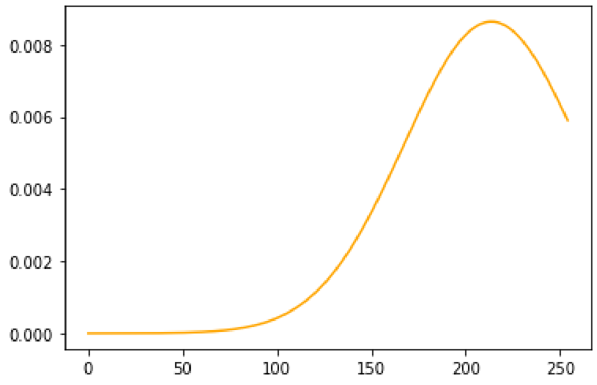

Figure 4: Bell curve for Orange buoy training dataset

3.3 Yellow buoys

We read the frames from the training dataset folder for cropped yellow buoy images. The images have been cropped as we restrict our ROI to minimise Noise. For each frame, we calculate Histogram values for each channel and stack them. Finally, after all frames have been parsed, we calculate average histogram value for each channel based on the number of images and plot them. The Histogram plot for Yellow buoys is as shown in figure 5.

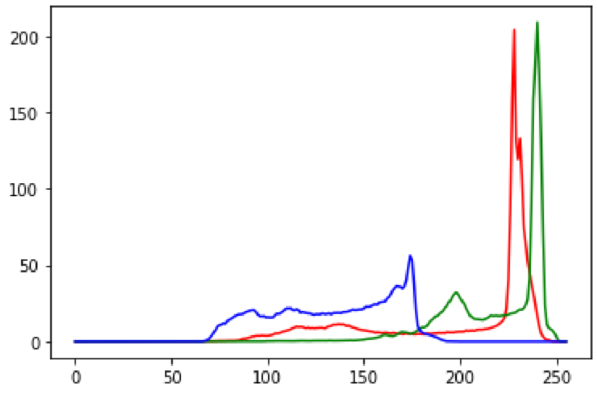

Figure 5: Histogram for Yellow buoy

As expected, it is clear that dominant spread among RGB channels in the histogram averages predominantly belong to the red and green channels only, since yellow is a combination of the red and green channels. So, we use the values of R and G channels for Mean and Standard deviation calculation. Taking the last frame into consideration, we try to calculate the mean and standard deviations for the red and green channel (column 3 and 2 in BGR channels) and come up with a bell-curve plot to show the distribution over the red and green channel. The mean and standard deviation calculation is carried out using the function **cv2.meanStdDev()** function, which is used on the final image of training dataset. The gaussian PDF is the same as equation 1. Figure 6 is an accurate representation of this concept.

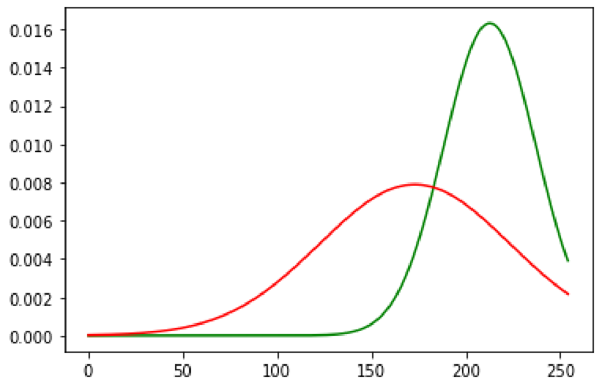

Figure 6: Bell curve forr Yellow buoy training dataset

4 Gaussian Mixture Models and Maximum likelihood algorithm

4.1 Creating Gaussian Mixture models for individual colors

From the previous section, it is observed that a set of points can be represented as outcomes of finite 1D Gaussian functions. However, we cannot reliably quantify the values of mean and standard deviations of these sets of points from the given dataset. So, we use the training dataset to "learn" these parameters and "predict" which Gaussian distribution a new unseen point comes from.
We make use of the "Expectation-Maximization (EM)" Algorithm, iteratively, to find out which point came from which latent gaussian distribution.

4.1.1 GMM for Green buoy

First, we read images from the Green training dataset and store each of the pixel value for each image in a list. Now, we define a set of mean and Standard deviation for each of the 3 gaussian channels, randomly. In our case, we have assumed the following.

Channel	Mean	Standard Deviation
Gaussian 1	190	10
Gaussian 2	150	10
Gaussian 3	250	10

Table 1: Gaussian mean and SD values

Now, for 40 iterations, we do the following steps:

- For every pixel in the list "pixels", created earlier, we calculate gaussian using mean and standard deviation individually for each of the gaussians and store them as p1, p2 and p3 respectively. Gaussian formula for a given mean and standard deviation is as follows:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2 / 2\sigma^2} \tag{2}$$

- From each of the Gaussian PDF's we calculate probability of selecting a particular gaussian and store their values in lists named b1, b2 and b3 respectively. As we know, the probability of choosing any one of the three gaussians is 0.33. Therefore, b1 is defined as:

$$B(x) = \frac{\frac{P1}{3}}{\frac{P1}{3} + \frac{P2}{3} + \frac{P3}{3}} \tag{3}$$

- The new mean and standard deviation values are obtained based on the pixel value (represented by pix) and corresponding B value calculated in the previous step:

$$\mu = \frac{\sum(B * pix)}{\sum B} \tag{4}$$

$$\sigma = \sqrt{\frac{\sum(B * (pix - \mu)^2)}{\sum B}} \tag{5}$$

After 40 iterations, we finally obtain the updated values of Mean and standard deviations for each of the three channels as follows.

Channel	Mean	Standard Deviation
Gaussian 1	184.90079776755584	20.577573078086086
Gaussian 2	152.06534796946667	15.898153337093653
Gaussian 3	236.81438517865615	11.714241624592178

Table 2: Updated Gaussian mean and SD values for green buoy

4.1.2 Detecting Green buoys

From the above calculated values of mean and standard deviations for each of the gaussians, we plot a bell curve for a list of elements numbered from 0 to 255. Gaussian PDF was defined earlier. Plotting the gaussians with blue, green and red plotlines, we get the following bell-curve plot:

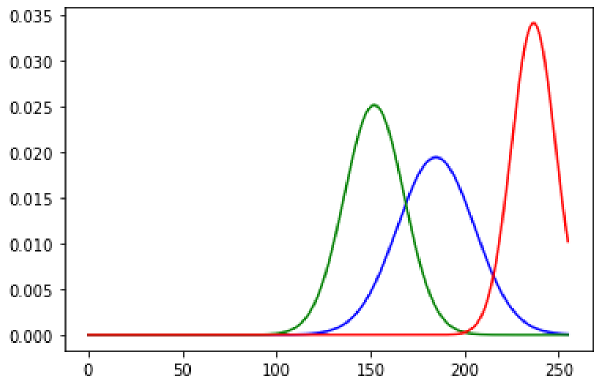

Figure 7: Gaussian Bell curves for Green buoy for list ranging from 0 to 255 with updated mean and SD values

Having trained our model against the training dataset, we try to detect the same values in our original dataset, the given video. From the above graph and from the values obtained in the gaussian curves, we can set the thresholds for gaussian's 1, 2 and 3 based on their peak values obtained. For Gaussian 1, denoted by blue, the peak value was 1.93870114e-02, which can be rounded off to 0.02. Similarly, for Gaussian 2, denoted by green, the peak value was 2.50934119e-02, which can again be rounded off to 0.025. For Gaussian

3, represented by red, the peak value was 3.40519015e-02, which can be rounded off to 0.035, but we also have to consider the highest point of the other curves while checking the threshold, se we set a threshold of 0.025 to be the lower limit and 0.035 to be the higher limit. Using these pre set limits, we further check for zones where the green channel shows positive response. Another main criteria of choice here is that, we know that green color is a significant part of yellow color as well, hence out delimiter here is the choice of band for Red channel. If we keep the amount of pixels having red to be lesser than a threshold value of 180 (max being 255), we can for sure reduce the probability of erranously detecting yellow buoys. Keeping these conditions, we now create a new list (of the same size as of the frame), and set the value to be 255 for all pixels that satisfy the above conditions, the rest of them are set to zero. Once this operation is carried out, we use the morphological operation of dilation to improve our results. Finding the places where we obtain a good match with our training data, we segment the region with a circle to improve our results. A photo for detecting the green buoy is as attached:

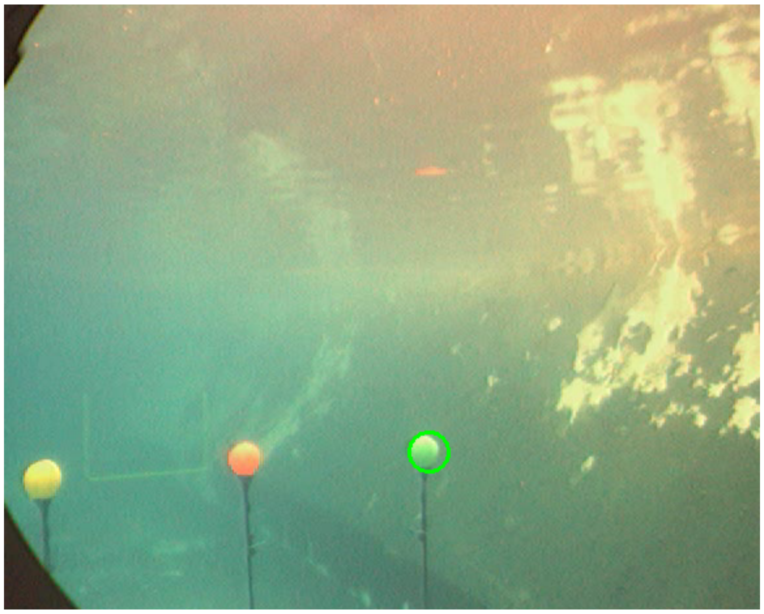

Figure 8: Green Buoy detection

4.1.3 GMM for Orange buoy

First, we read images from the Orange training dataset and store each of the pixel value for each image in a list. Now, we define a set of mean and Standard deviation for each of the 3 gaussian channels, randomly. In our case, we have assumed the following.

Channel	Mean	Standard Deviation
Gaussian 1	190	10
Gaussian 2	150	10
Gaussian 3	250	10

Table 3: Gaussian mean and SD values for Orange buoys

Now, for 40 iterations, we do the same steps as last time. After 40 iterations, we finally obtain the updated values of Mean and standard deviations for each of the three channels as follows.

Channel	Mean	Standard Deviation
Gaussian 1	238.14753766499953	8.417587391439978
Gaussian 2	154.72662419580544	36.30795123353113
Gaussian 3	252.24937318452012	2.346205081137928

Table 4: Updated Gaussian mean and SD values for orange buoy

4.1.4 Detecting Orange buoys

From the above calculated values of mean and standard deviations for each of the gaussians, we plot a bell curve for a list of elements numbered from 0 to 255. Gaussian PDF was defined earlier. Plotting the gaussians with blue, green and red plotlines, we get the following bell-curve plot:

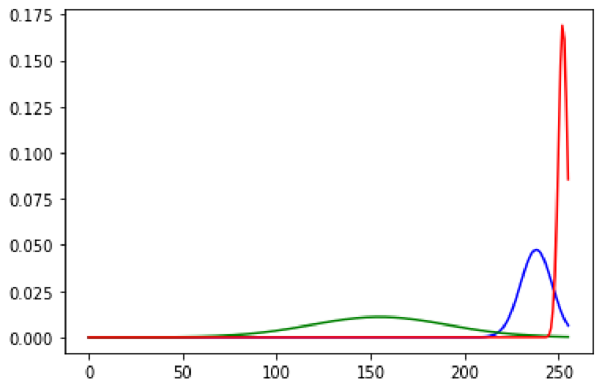

Figure 9: Gaussian Bell curves for Orange buoy for list ranging from 0 to 255 with updated mean and SD values

Having trained our model against the training dataset, we try to detect the same values in our original dataset, the given video. From the above graph and from the values obtained in the gaussian curves, we can set the thresholds for gaussian's 1, 2 and 3 based on their peak values obtained. For Gaussian 1, denoted by blue, the peak value was 4.73866190e-002, which can be rounded off to 0.05. Similarly, for Gaussian 2, denoted by green, the peak value was 1.09874272e-02, which can again be rounded off to 0.01. For Gaussian 3, represented by red, the peak value was 1.69079501e-001, which can be rounded off to 0.1, but we also have to consider the highest point of the other curves while checking the threshold, se we set a threshold of 0.05 to be the lower limit and 0.1 to be the higher limit. Using these pre set limits, we further check for zones where the red/orange channel shows positive response. Another main criteria of choice here is that, we know that blue color is a significant part of the background as well, hence our delimiter here is the choice of band for blue channel. If we keep the amount of pixels having blue to be lesser than a threshold value of 150 (max being 255), we can for sure reduce the probability of erranously detecting other buoys. Keeping these conditions, we now create a new list (of the same size as of the frame), and set the value to be 255 for all pixels that satisfy the above conditions, the rest of them are set to zero. Once this operation is carried out, we use the morphological operation of dilation to improve our results. Finding the places where we obtain a good match with our training data, we segment the region with a circle to improve our results. A photo for detecting the orange buoy is as attached:

4.1.5 GMM for Yellow buoy

First, we read images from the Yellow training dataset and store each of the pixel value for each image in a list. However for Yellow, it is a special case as we are interested in both Green and Red channels, as yellow is a combination of 100% red and 100% green. We, therefore, create two separate lists of pixels, each having the value from corresponsing red and green channels. Now, we define a set of mean and Standard deviation

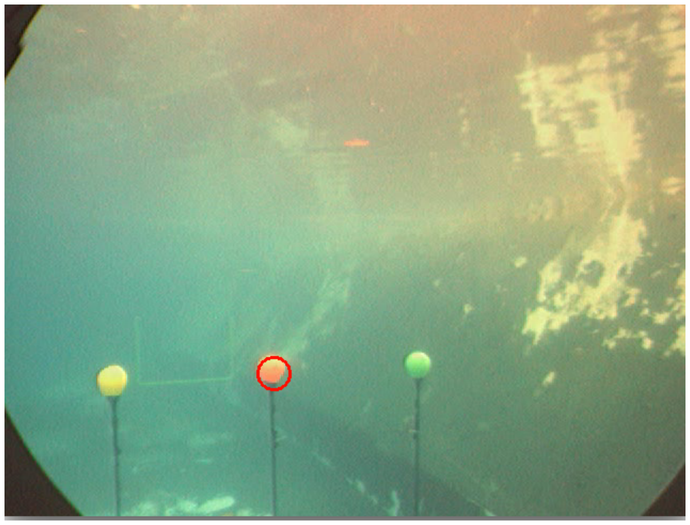

Figure 10: Orange Buoy detection

for each of the 3 gaussian channels, randomly. In our case, we have assumed the following.

Channel	Mean	Standard Deviation
Gaussian 1	190	10
Gaussian 2	150	10
Gaussian 3	250	10

Table 5: Gaussian mean and SD values

Now, for 50 iterations, we do the following steps:

- For every pixel in the list "pixels1", created earlier, we calculate gaussian using mean and standard deviation individually for each of the gaussians and store them as p11, p12 and p13 respectively. Gaussian formula for a given mean and standard deviation is as follows:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \tag{6}$$

  Correspondingly, we find the values of P21, P22 and P23 for each member in the list "pixels2".

- From each of the Gaussian PDF's we calculate probability of selecting a particular gaussian and store their values in lists named (b11, b12 and b13) and (b21, b22 and b23) for each pixel list, respectively. As we know, the probability of choosing any one of the three gaussians is 0.33. Therefore, b1 is defined as:

$$B11(x) = \frac{\frac{P11}{3}}{\frac{P11}{3} + \frac{P12}{3} + \frac{P13}{3}} \tag{7}$$

- The new mean and standard deviation for each of pixel1 and pixel2 values are obtained based on the pixel value (represented by pix) and corresponding B value calculated in the previous step. For example:

$$\mu_{11} = \frac{\sum(B11 * pix1)}{\sum B11} \tag{8}$$

$$\sigma_{11} = \sqrt{\frac{\sum(B11 * (pix1 - \mu_1 1)^2)}{\sum B11}} \tag{9}$$

Eventually, mean and standard deviation for each of the gaussians are obtained as follows:

$$\mu_1 = \frac{\mu_{11} + \mu_{21}}{2} \tag{10}$$

$$\sigma_1 = \frac{\sigma_{11} + \sigma_{21}}{2} \tag{11}$$

After 50 iterations, we finally obtain the updated values of Mean and standard deviations for each of the three channels as follows.

Channel	Mean	Standard Deviation
Gaussian 1	230.99625753104914	9.384343859959012
Gaussian 2	170.86092610188882	38.29599853584383
Gaussian 3	235.26442800066383	5.645941025982294

Table 6: Updated Gaussian mean and SD values for Yellow buoy

4.1.6 Detecting Yellow buoys

From the above calculated values of mean and standard deviations for each of the gaussians, we plot a bell curve for a list of elements numbered from 0 to 255. Gaussian PDF was defined earlier. Plotting the gaussians with green, blue and red plotlines for gaussian1, gaussian2 and gaussian3 respectively, we get the following bell-curve plot:

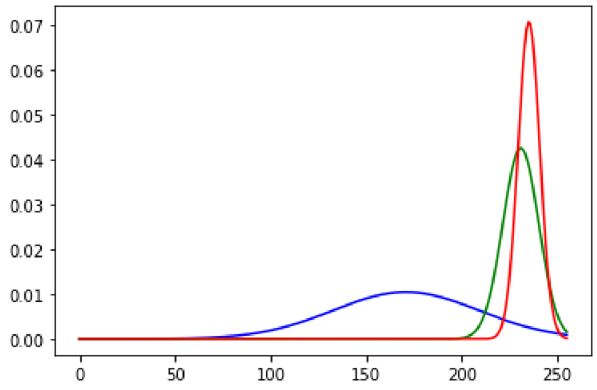

Figure 11: Gaussian Bell curves for yellow buoy for list ranging from 0 to 255 with updated mean and SD values

Having trained our model against the training dataset, we try to detect the same values in our original dataset, the given video. From the above graph and from the values obtained in the gaussian curves, we can set the thresholds for gaussian's 1, 2 and 3 based on their peak values obtained. For Gaussian 1, denoted by green, the peak value was 4.25114696e-002, which can be rounded off to 0.045. Similarly, for Gaussian 2, denoted by blue, the peak value was 1.04172672e-02, which can again be rounded off to 0.01. For Gaussian 3, represented by red, the peak value was 7.05825608e-002, which can be rounded off to 0.07, but we also have to consider the highest point of the other curves while checking the threshold, se we set a threshold of 0.045 to be the lower limit and 0.07 to be the higher limit. Using these pre set limits, we further check for zones where the red and green channel show positive response. Another main criteria of choice here is that, we know that blue color is a significant part of the background as well, but is not a part of yellow's composition. Hence our delimiter here is the choice of band for blue channel. If we keep the amount of pixels having blue to be lesser than a threshold value of 130 (max being 255), we can for sure reduce the probability of erranously detecting other buoys. Keeping these conditions, we now create a new list (of the same size as of the frame), and set the value to be 255 for all pixels that satisfy the above conditions, the rest of them are set to zero. Once this operation is carried out, we use the morphological operation of dilation to improve our results. Finding the places where we obtain a good match with our training data, we segment the region with a circle to improve our results. A photo for detecting the yellow buoy is as attached:

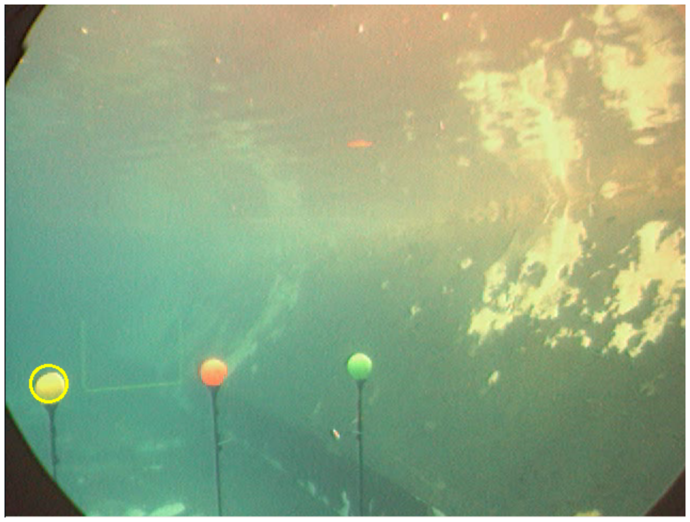

Figure 12: Yellow Buoy detection

5 Final buoy detection

We combine all the codes compiled earlier to obtain a video output in which all three buoys are simultaneously being detected. A sample screen shot is attached.

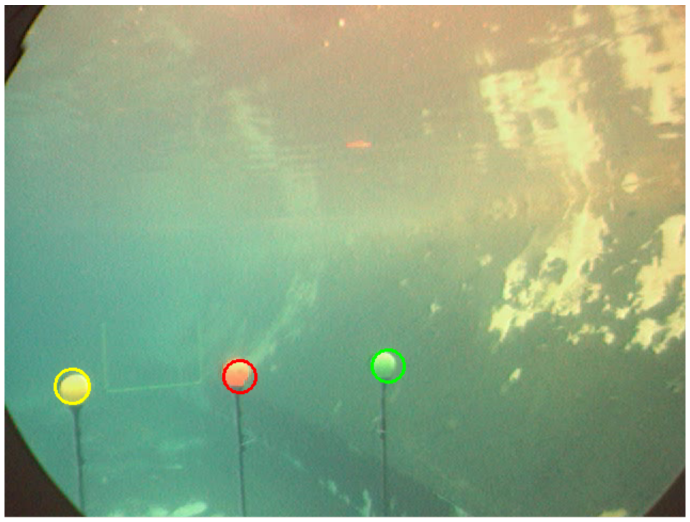

Figure 13: All 3 Buoy's detection