ENPM673 - Perception for Autonomous Robots

Project 3

Prateek Arora, Abhinav Modi, Samer Charifa

Due date: 1^{st} *April 2020, 11:59 p.m.*

Submission Guidelines

- The homework is to be completed in group and there should be a single submission per group.
- Your submission on ELMS/Canvas must be a zip file, following the naming convention **YourDirectoryID_proj_3.zip**. If you email ID is abc@umd.edu or abc@terpmail.umd.edu, then your DirectoryID is **abc**. Remember, this is your directory ID and **NOT** your UID. Please provide detailed instructions on how to run your code in README.md file.
- For each section of the homework, explain briefly what you did, and describe any interesting problems you encountered and/or solutions you implemented. Your report should preferable be typeset in LaTeX.
- Please submit only the python script(s) you used to compute the results, the PDF report you generate for the project and a detailed README file (refer to the directory structure below).
- Include sample outputs in the your report.
- The video outputs are to submitted as a separate link in the report itself. The link can be a YouTube video or a google drive link (or any other cloud storage). Make sure that you provide proper sharing permission to access the files. **If we are not able to access the output files you will be awarded 0 for that part of the project**. Video results should include detection of all three bouy. On elms you can only submit one .zip file which can be of maximum 100 MB.
- Disallowed function:
  - any inbuilt function from scipy or sklean that directly implements Gmm.
  - any other inbuilt function that solves the question in less than 5 lines.

The file tree of your submission SHOULD resemble this:

```
YourDirectoryID_hw1.zip
├── Code
│   ├── .py files
│   └── any subdirectories that you may have
├── Report.pdf
└── README.md
```

Introduction

This project will introduce you to the concept of color segmentation using Gaussian Mixture Models and Expectation Maximization techniques. The video sequence you are provided has been captured underwater and shows three buoys of different colors, namely yellow, orange and green. They are almost circular in shape and are distinctly colored. However, conventional segmentation techniques involving color thresholding will not work well in such an environment, since noise and varying light intensities will render any hard-coded thresholds ineffective. In such a scenario, you will "learn" the color distributions of the buoys and use that

learned model to segment them. This project requires you to obtain a tight segmentation of each buoy for the entire video sequence by applying a tight contour (in the respective color of the buoy being segmented) around each buoy.

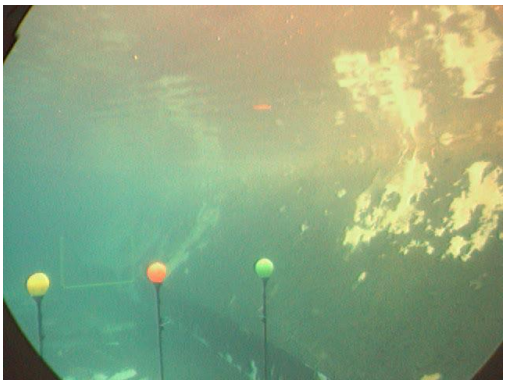


Figure 1: Sample image frame of buoys underwater

Data

You have three videos to test your code. Download videos using this link.

Data Preparation

From a number of frames of the video extract cropped samples of the buoy and save them for further training. You may use Matplotlib event handling or refer third party code that emulates roipoly function of MATLAB functionality to get coordinates from mouse clicks and save them to a numpy array. Ideally, the tighter your bounding polygons, the better training results you will get. You may draw simple rectangles, but that will also bring unwanted regions on the sides of the buoys. Also, as with any learning based method, make sure to split your data into separate training and testing sets. You will end up overfitting otherwise. For each color of buoy, ensure you have separate folders with training and testing data, in a 70%-30% split respectively.

For each colored buoy separately, compute and visualize the average color histogram for each channel of the sampled RGB images. Ensure you provide these outputs in your report and explain your analysis of the same.

Let us assume a 1-D Gaussian is used to model the color distribution of the buoys. Depending on which buoy you are detecting, for each pixel you should use combinations of appropriate channels as the color sample value - say for the red buoy you can directly use the R channel value whereas for the yellow buoy, you might want to use R+G as the color sample. Design and implement a process to segment the buoys using a 1-D Gaussian. For example, simple threshold on the values of the 1-D Gaussian. Show sufficient outputs in your report and explain your chosen process in detail in your report.

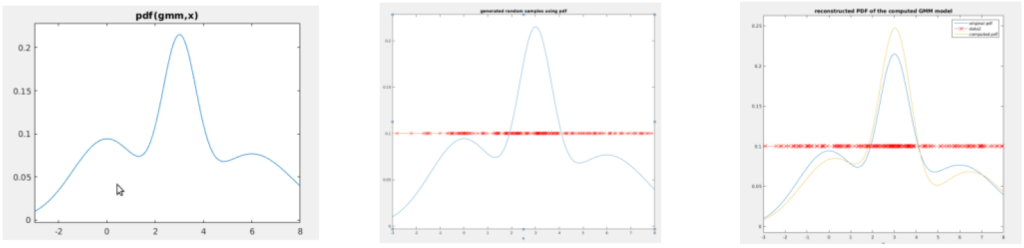
Gaussian Mixture Models and Maximum Likelihood Algorithm

Imagine you are given a set of points, and you make the assumption that each of the points were generated from a finite number of Gaussian distributions. However, you are not not given the parameters of these distributions. Given these known points, you try to "learn" these parameters and then "predict" which Gaussian distributions a new, previously unseen point came from.

The main problem is this aspect of finding which points came from which latent distribution. To solve this, we employ the Expectation-Maximization technique (EM), which is a well known statistical algorithm that works iteratively.

Initially, we assume random points" and then compute for each point a probability that the point was generated by each one of the distributions. Then, iteratively, we tweak the parameters to try and maximize the likelihood of the data given those assignments. Repeating this sufficient times will guarantee a local optimum. The Expectation algorithm is explained appropriately in the lecture slides.

Expectation Maximization



(a) Probability Density Function

(b) Randomly generated samples

(c) pdf of extracted GMM parameters

Figure 2: Fitting GMM for a probability Density Function using Expectation Maximization

which calculates the homography or warps the image for you .

1. Generate data samples from three 1-D Gaussians, with different means and variances. (e.g. 50 samples for each distribution).

   E.g.: The plots in Fig. 2 show the combined pdf of 3 Gaussian distributions with means (μ) at 0, 3 and 6 and standard deviations (σ) of 2, 0.5 and 3 respectively.

2. Implement the Expectation Maximization algorithm as explained in class. (for additional resources see this or this.)

3. Recover the model parameters for the three Gaussians (i.e. the means and variances) using your implementation of EM. Compare your recovered parameters with the ground-truth values. As usual, make sure to generate sufficiently detailed plots of your outputs and comparisons and detail your approach and observations in the report.

Learning Color Models

Now, extend the previous concepts and implementations to achieve the original goal of segmenting the buoys based on color.

You will be processing and segmenting one buoy at a time. That is, you will first compute the color distribution for the orange buoy and use this distribution to segment the orange buoy from background. Then you repeat the process for the orange buoy and the green buoy.

1. For each color, compute and visualize the color histogram for each channel of the sampled/cropped images gathered during the data preparation phase. This will provide some intuition on how many Gaussians [**N**] to fit to the color histogram. Also, try to determine the dimension [**D**] of each Gaussian for your model of choice.

2. Use your previously implemented EM algorithm to compute the model parameters, i.e. the means and variances of the **N D**-dimensional Gaussian.

3. Provide plot(s) of the model generated for each colored buoy. Also, explain your method and approach in your report.

Buoy Detection

Implement a buoy detection pipeline that will take a frame from the video sequence as input and generate a color-segmented binary image, given the computed model parameters. Also compute the corresponding contours and center of each buoy.

Draw the bounding contours around the buoy in the original frame, in their respective colors. Ensure this contour is as tight as possible for highest possible scores.

Provide outputs for several non-consecutive frames in your report, and explain your pipeline and approach.