ESE650 Project 1: Color Segmentation

Jiani Li

I. INTRODUCTION

Color segmentation problem is a classical problem in robotics and computer vision. In this project, I designed a hierarchical probabilistic model to detect red barrels in images, and find the relative world coordinates of the barrel from images. Specifically, I trained a five-cluster GMM model for the positive pixels (red color of the barrel) using EM algorithm, and a single multivariate Gaussian distribution for all other negative pixels using MLE. These two in-class models are then combined by the Bayes Rules for generating color segmentation result. After getting the raw segmentation map, I used Connected Component Analysis to propose candidate regions. Shape information about the barrel are then applied to select barrels from the candidate. Besides that, a depth regression model is also learned to give an approximate estimation of the depth of the barrel. For the following sections, I will further explain the model I use and also analyze the final performance.

II. COLOR SEGMENTATION ALGORITHMS

The first part of this task is to generate a pixel-wise color segmentation mask, which is a binary classification problem, that takes a 3-dimensional (generally) feature as input.

A. Color Space

In this project, we use H(Hue)S(Saturation)V(Value) color space as the training feature, where 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. HSV is chosen because it is less sensitive to lightning condition, and is able to find most of the red pixels even in extremely dark environment. Also, in experiment (section IV), it is shown that HSV feature achieves the better recall and precision in validation compared to other color spaces for segmentation problems.

B. Model Representation

According to Bayes rules, the probability for data point x can be represented as follows:

$$P(C|\mathrm{x}) = \frac{P(\mathrm{x}|C)P(C)}{\sum_{C' \in \{0,1\}} P(\mathrm{x}|C')P(C')} \tag{1}$$

When $C = 1$ (red barrel), the probability $P(\mathrm{x}|C = 1)$ is represented by a five-cluster GMM(Gaussian Mixture Model). Compared to a single Gaussian, multiple Gaussian models may better fit the actual distribution of the red pixels due to different illumination environment. In section IV, we provide a comparison between models using different number of Gaussians, and we prove that when $K = 5$, the model provides a satisfactory representation of the training data without overfitting.

When $C = 0$ (non-barrel background), we simply use a multivariate Gaussian distribution to represent $P(\mathrm{x}|C = 0)$. This is because the color distribution in the background dataset is much more complex and hard to train. Also, since the size of the background data is much larger than the foreground, the time consumption for training a converged model for could be quite large, which is actually not quite necessary for the current project. The parameter of multivariate Gaussian can be estimated using MLE and it has a numerical solution, which makes is efficient to train.

C. Training

We use EM algorithm to train the GMM model for 'red barrel' dataset. Parameters to be trained are: μ_k, Σ_k and π_k ($k = 1, 2, 3, 4, 5$). The covariance matrices as no restriction in form. The initial value of μ is randomly selected from dataset. Generally, the algorithm converges in 90 to 150 iterations.

The multivariate Gaussian model is trained using MLE.

The prior probability $P(C)$ is calculate using the statistics of training data points, i.e. $P(C = 1) = N_{red}/(N_{red} + N_{background})$

III. BARREL LOCALIZATION

A. Connected Component Analysis

After getting the raw color segmentation mask, we can then run connected component analysis to find regions in the image. Before this, I use `binary_dilation` and `binary_erosion` from `scipy.ndimage.morphology` library to filter out the noise points in the mask. Then we can use `label` and `regionprops` in `skimage.measure` to generate region proposals.

Since their may be other objects in the background whose color is similar to the barrel and labeled as foreground, we need some post-processing to remove the false-positives. Here in my method, I mainly use the aspect-ratio, size and and extent of pixels in bounding box to filter out these non-barrel shaped-like objects. Details can be found in code.

B. Object Depth Estimation

Another task of the project is to estimate the distance of the barrel to the camera in real world. As we know, the pin hole model for the camera can be written as:

$$\frac{f}{s_{img}} = \frac{D}{s_{world}} \Rightarrow D = \frac{f * s_{world}}{s_{img}} \quad (2)$$

Since the focal length f and the scale of the barrel in real world is not given, let $v = 1/s_{img}$, and $\alpha = f * s_{world}$ be a parameter. We have a linear regression model:

$$D = \alpha v \quad (3)$$

Here, the scale of the barrel on image is approximated using $\sqrt{h_{img} w_{img}}$, where h_{img} and w_{img} are the barrels height and width in pixels. We can use easily estimate the parameter α using maximum likelihood estimation given the depth and size of the barrel in training images.

IV. EXPERIMENT RESULT

A. Dataset

We are given 50 900×1200 color images for training. The pixels of the red barrel are manually labelled using provided script `annotate.py`. Images are loaded and then converted into HSV color space.

B. Performance Analysis and Model Selection

In order to evaluate the performance of the model, I use 10-fold cross validation on the training dataset. Each time, 90% of the data is used for training and the rest 10% is used for validation. Below lists of the average validation accuracy of different color segmentation models and parameter settings. By default, all of the models below use single multivariate Gaussian model for background class:

Red Color Model	Accuracy	Precision	Recall
Single Gaussian	0.9892	0.7615	0.7153
2-GMM	0.9905	0.7764	0.8956
3-GMM	0.9948	0.8887	0.8647
4-GMM	0.9958	0.8693	0.9033
5-GMM	**0.9979**	**0.9088**	**0.9262**
6-GMM	0.9982	0.9208	0.9190

TABLE I: Performance of models using different number of Gaussians: validation accuracy, precision and recall (pixel-wise)

Generally, the more clusters/parameters we have, the more likely we are going to overfit, also, the training will require more time. On the other hand, if the model is too simple, we may end up with an underfit model. Through experiment, we can see that when $K = 5$, we have a better classification result.

Color Space	Accuracy	Precision	Recall
YCrCb	0.9897	0.8378	0.8825
LAB	0.9930	0.8592	0.8927
RGB	0.9963	0.9045	0.8770
HSV	**0.9979**	**0.9088**	**0.9275**

TABLE II: Performance of 5-GMM model using different color space

Result of the final model are given below:

Image Id	Centroid X	Centroid Y	Distance
001	644.7005	466.0356	2.2630
002	607.2045	353.4317	2.6684
002	631.5041	514.1258	3.2777
003	699.0716	450.1342	2.7898
004	634.0926	443.4151	2.0572
005	645.4176	464.7573	6.3186
006	632.1341	427.2888	10.1015
007	566.2189	627.8457	3.0962
008			
009	673.1245	428.0419	10.7839
010	663.3129	398.7970	6.7990

TABLE III: Test result: position of barrel(s)

V. CONCLUSIONS

Generally, the algorithm is able to detect most of the targets in the test images, and it shows robustness in handling tricky cases such as different lightning conditions, indoor and outdoor environment and clustered background.

However, there are still some weak points about the method. For example, the color segmentation algorithm sometimes label non-barrel red items as positive, and the candidate proposal algorithm doesn't work well when occlusion happens. To achieve more accurate detection result, there are still several aspects we can improve about the current model. Possible improvements includes (but not limited to): i) Data augmentation. Data augmentation techniques can be used to avoid overfitting, such as changing the brightness or resolution of the training images. ii) Introduce models for more colors, especially the 'false-positive' red colors. iii) Region merge algorithms. iv) Advanced candidate finding algorithms, such as clustering. Unlike low-level connectivity analysis, such methods consider a more global spatial distribution of the data, and thus is more accurate in finding targets, especially for occluded objects.

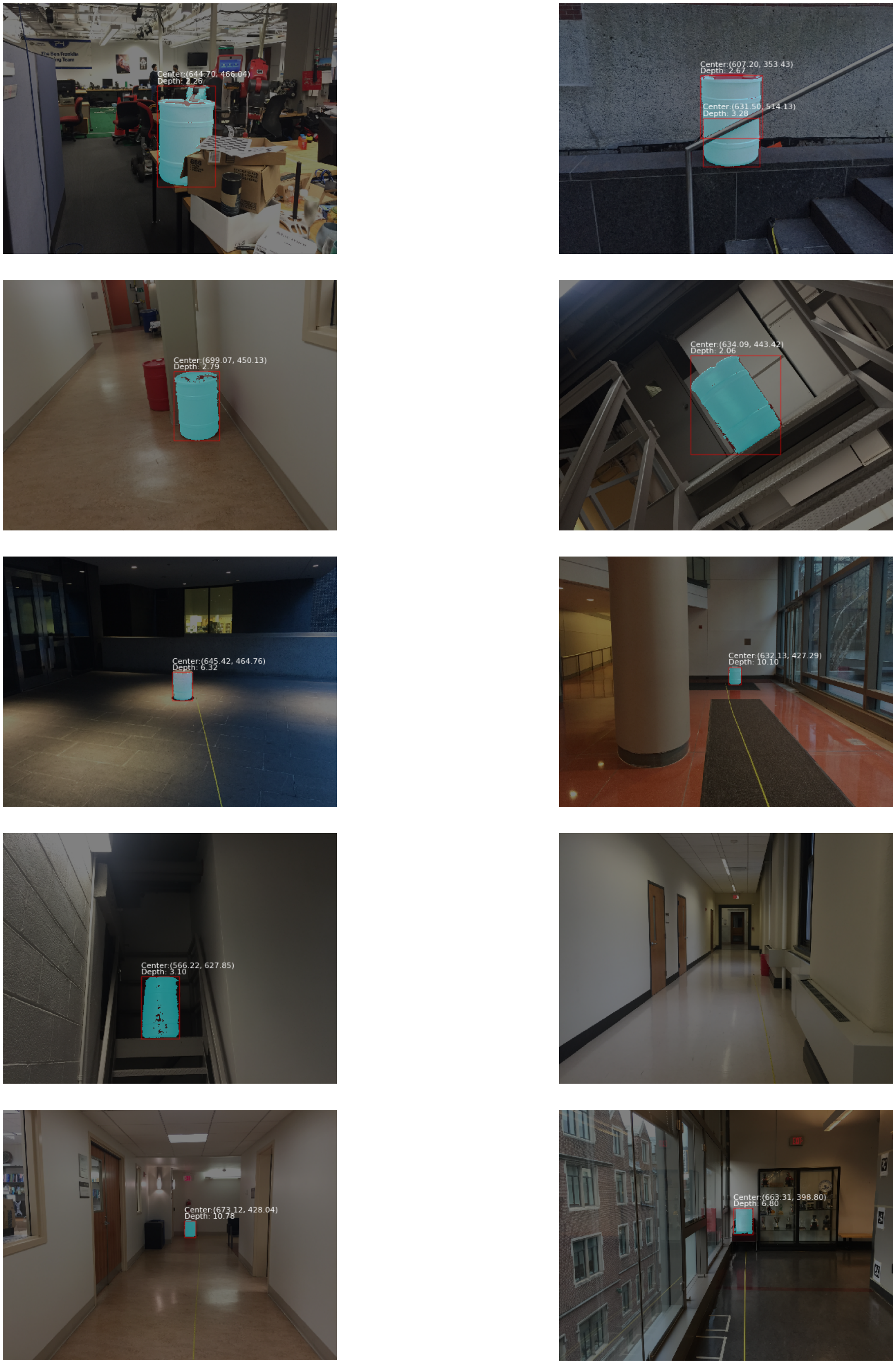

Fig. 1: Test Result: segmentation mask, bounding box, center and depth