Object Detection and Color Segmentation

1st Yifan Xu A53253258
The Electrical and Computer Engineering Department UCSD
La Jolla, US
yix247@eng.ucsd.edu

***Abstract*—This paper is a report for the first project in ECE276 Sensing and Estimation in Robotics, UCSD.**

***Index Terms*—machine learning, color segmentation, bounding box, logistics regression**

I. Introduction

Image segmentation is the process of partitioning a digital image into multiple segments. The goal of segmentation is to simplify and change the representation of an image into something that is more meaningful and easier to analyze. In computer vision, image segmentation is one the most important processes to make computer understand the complex scenarios in real world. It has been applied in many different applications like human detection, face recognition and autonomous driving.

There are many different approaches to get segmented images. In this project, we need to detect a blue barrel in images, and draw a bounding box around it. Therefore, we segmented pictures based on their colors. The first step was to get a good training data set which divided the barrel blue and other colors in the environment. The second step was using logistics regression to get well trained parameters for the model. By using this model, we could get segmented images. Then we need to find bounding boxes for real barrels by evaluating the their other features, such as shape and size.

II. Problem Formulation

The main problem to this color segmentation project was to find good parameters for the probabilistic color model. To solve this problem, I used discriminative classification via a logistic regression model.

A. Logistic regression

Logistic regression uses a discriminative model $p(\mathbf{y}|\mathbf{X}, w)$ for the discrete labels $\mathbf{y} \in \{-1, 1\}^n$ that is a product of sigmoid functions:

$$p(\mathbf{y}|\mathbf{X}, w) = \prod_{i=1}^{n} \alpha(y_i \mathbf{x}_i^T w) = \prod_{i=1}^{n} \frac{1}{1 + exp(-y_i \mathbf{x}_i^T w)}$$

Then we tried to minimize the negative log-likelihood iteratively to obtain a global minimum with $w_{\text{MLE}}^{(t+1)}$:

$$w_{\text{MLE}}^{(t+1)} = w_{\text{MLE}}^{(t)} = \alpha \sum_{i=1}^{n} y_i \mathbf{x}_i (1 - \sigma(-y_i \mathbf{x}_i^T w_{\text{MLE}}^{(t)}))$$

After get the optimized w^*, by logistic regression's linear decision bounday $(x^*)^T w^* = 0$, we can divide test data into to group, in this project, as barrel blue and not barrel blue.

III. Technical Approach

For this project, we had two tasks. First, color segmentation. Second, barrel detection.

A. Color segmentation

- **Get training data from given training photos.** By using selectROI in OpenCV, we can get cropped images. The selection of training data is really important for training a good model. We recorded the RGB values of each pixels and divided them into two groups, barrel blue and not barrel blue with label 1 and -1, respectively. Also, I left some photos excluded from the training data set so that we can test our classifier on them.
- **Train the training data.** By using the logistics regression algorithm I mentioned in the last section. We can get optimized parameters for this color classifier. Then, we can use this classifier to decide whether a test pixel is barrel blue or not and get a binary image(mask) with 1 if the pixel in the original image is blue and 0 otherwise. I divided training data into two data sets, training data set and validation data set after shuffle. Then, I also divided training data set into five smaller data sets to shorten the training time. The next smaller training data set will use the optimized parameters returned by last data set. Each smaller data set will update parameters ten times and we can get both training accuracy and validation accuracy while training.

B. Barrel detection

- **Set the thresholds for detecting as barrel.** Certainly, a thing need to meet some conditions to become a barrel. Therefore, I set the area size of a barrel in the mask should be bigger than 800 and the ratio of the height to the width of a barrel in the mask should be bigger than 1.421 and smaller than 2.354.
- **Get the informations of contours.**By using findContours in OpenCV, we can get all contours in the binary mask image. After got all contours information, we can get the area size of each contour by contourArea and points coordinates of each contour by boundingRect in Opencv.
- **To improve the detecting accuracy.** By testing our classifier and detection rules in validation data sets, we could

figure out under some certain circumstances, it didn't work well. Therefore, we need to set more conditions to determine whether is a barrel. For me, I set a rule that if more than one bounding boxes are really near to each other, we treat it as one bounding box and then decide it by the area size and height-width ratio.

IV. Results

A. Parameter Result

As my training approach which I introduced above, I got the training result and accuracy.

$$w = [0.29000298, -0.16778781\ , -0.48117849, 0.00139205]^T$$

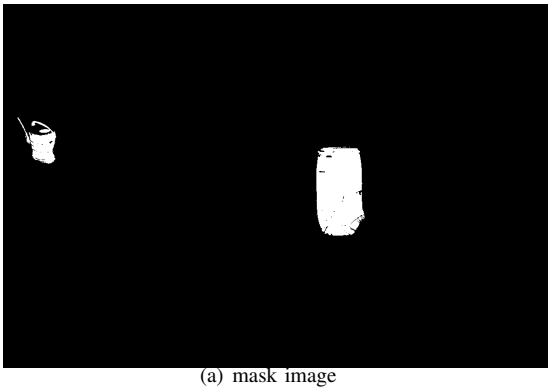
$$TrainingsetAccuracy : 0.98788221$$

$$ValidationsetAccuracy : 0.99925648$$

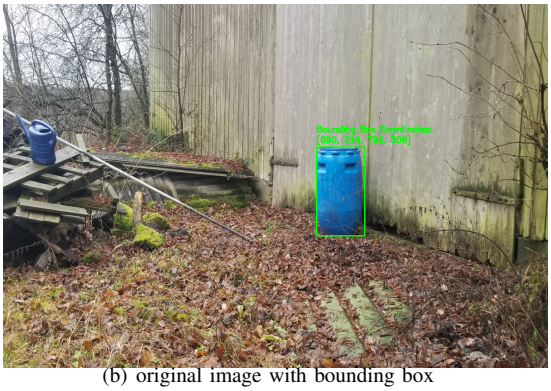
This result looked pretty good.

B. Image Result

1) Good Results: Here are some results which shows my algorithm works pretty well on these cases.

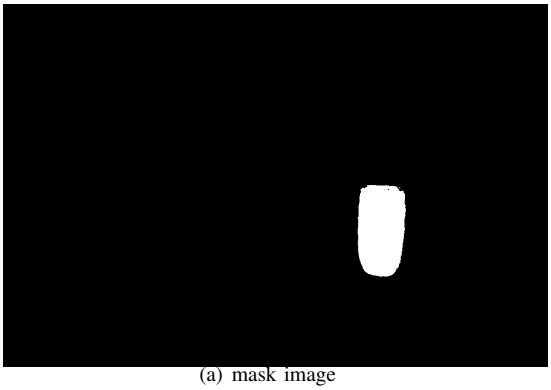


(a) mask image



(b) original image with bounding box

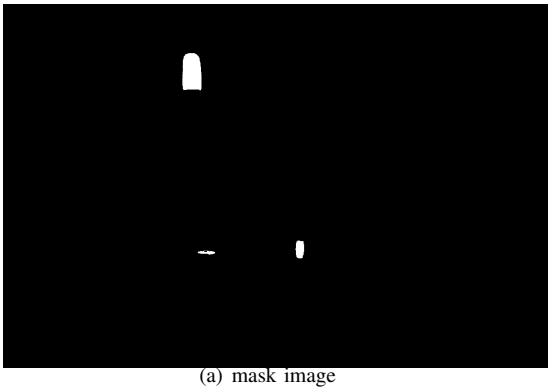
Fig. 1. Good result 1



(a) mask image

(b) original image with bounding box

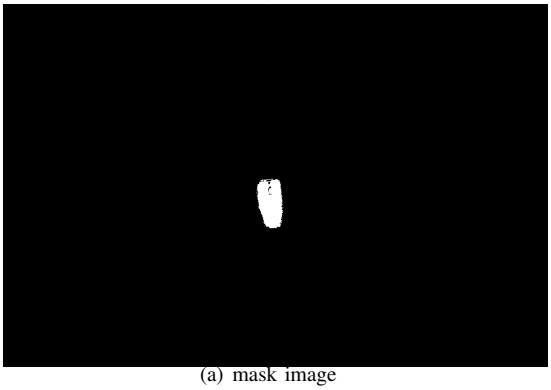
Fig. 2. Good result 2



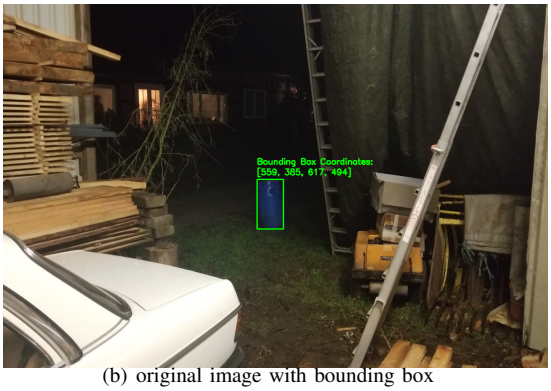
(a) mask image

(b) original image with bounding box

Fig. 3. Good result 3

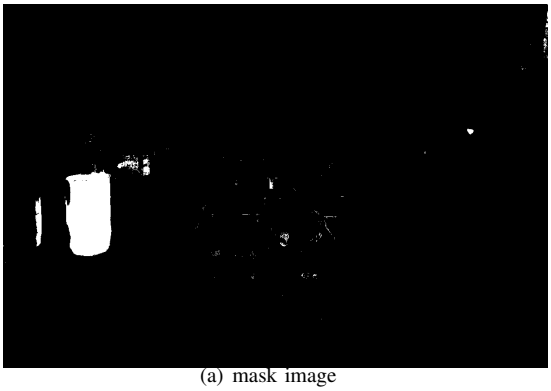


(a) mask image



(b) original image with bounding box

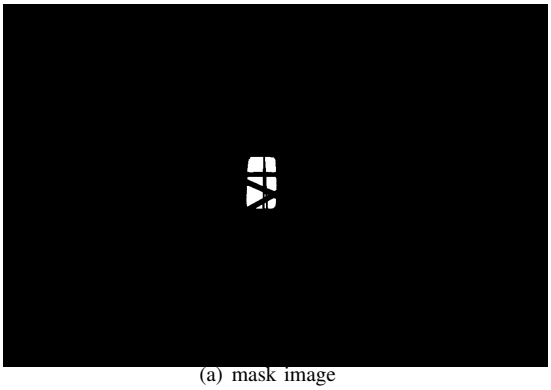
Fig. 4. Good result 4



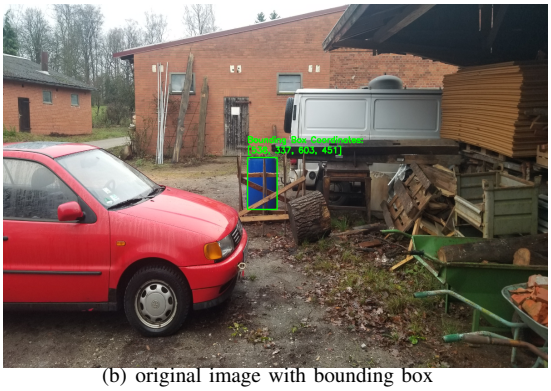
(a) mask image

(b) original image with bounding box

Fig. 5. Good result 5

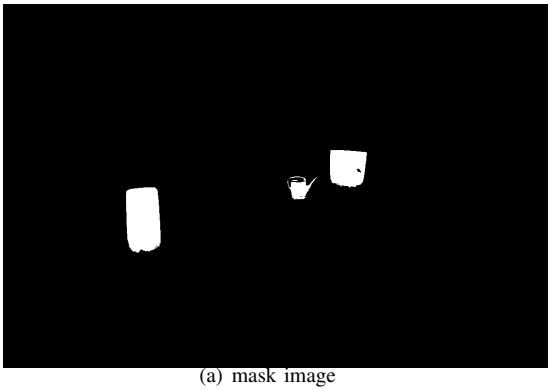


(a) mask image

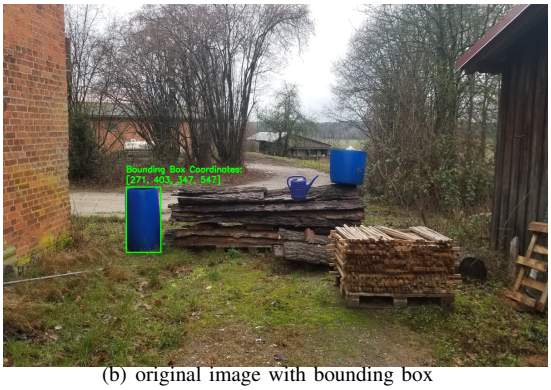


(b) original image with bounding box

Fig. 6. Good result 6



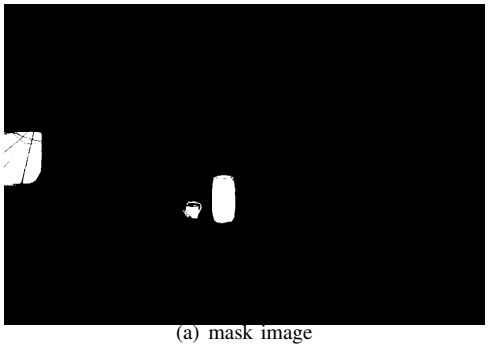
(a) mask image



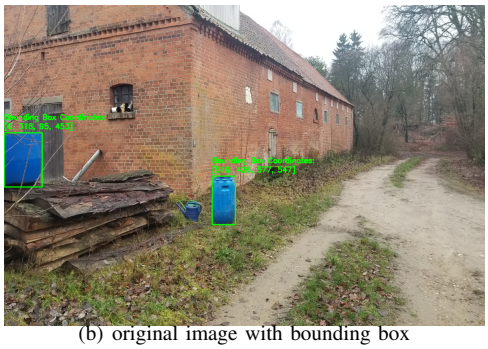
(b) original image with bounding box

Fig. 7. Good result 7

2) Bad Results: Here are some results which shows that to achieve our goal, we still need to improve our classifier and set more constraint conditions.

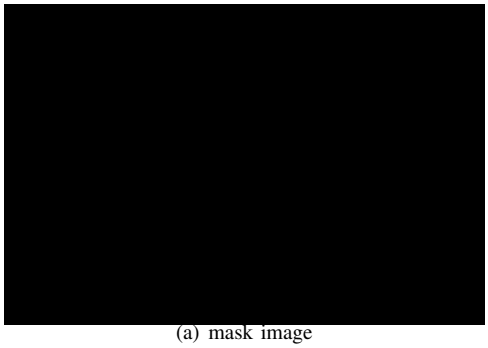


(a) mask image



(b) original image with bounding box

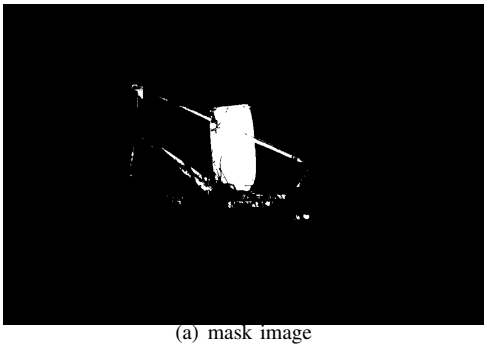
Fig. 8. Bad result 1



(a) mask image

(b) original image with bounding box

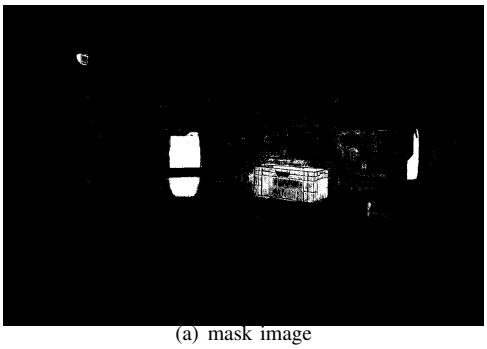
Fig. 9. Bad result 2



(a) mask image

(b) original image with bounding box

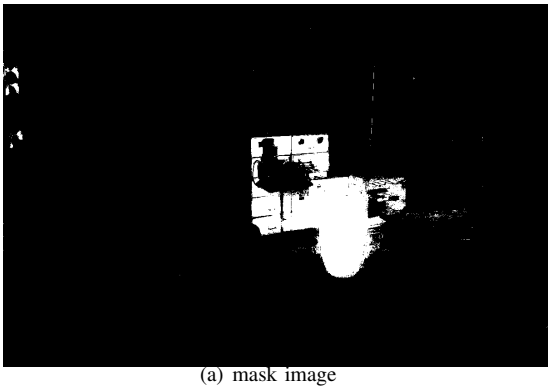
Fig. 10. Bad result 3



(a) mask image

(b) original image with bounding box

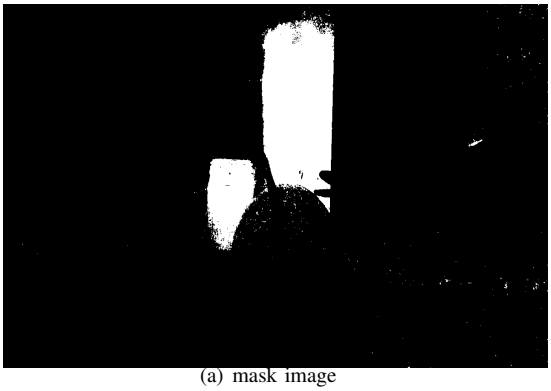
Fig. 11. Bad result 4



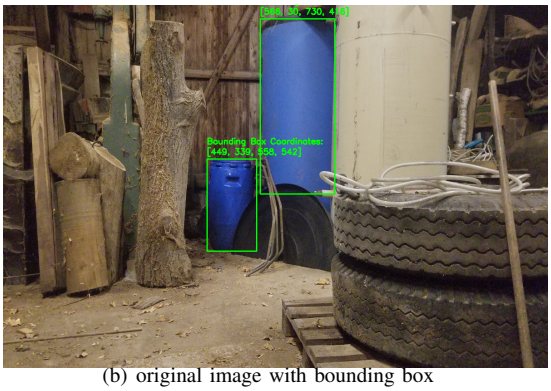
(a) mask image

(b) original image with bounding box

Fig. 12. Bad result 5

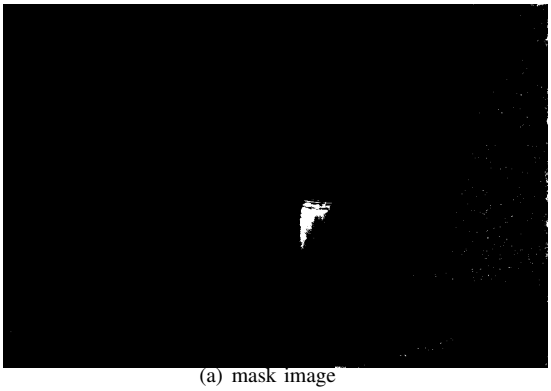


(a) mask image

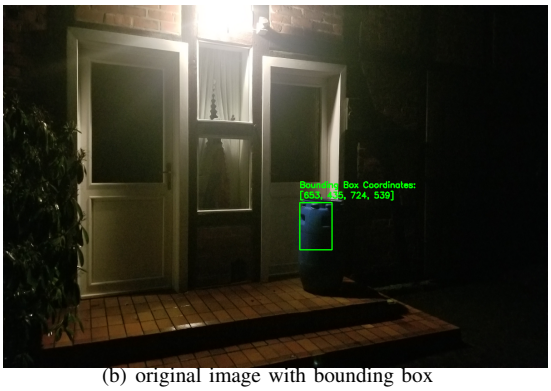


(b) original image with bounding box

Fig. 13. Bad result 6



(a) mask image



(b) original image with bounding box

Fig. 14. Bad result 7

C. Discussion

From the figures shown above, the classifier with constraint conditions works pretty well when the difference between blue barrels and background is very clear. It can distinguish barrels with other blue stuffs which are too small or with wrong shape. It also can detect the barrel even it has been divided into pieces by obstacles.

However, there still have many circumstances that it will fail to detect barrels. For example, when there is something blue with the same size of barrel or when the other blue object has overlap with blue barrel or the classifier fails to classify pixels as ”blue barrel” when it’s too dark.

Therefore, to improve the performance, first we should increase the accuracy of classifier. This can be done by using some higher level features to decide where the barrel is located in the images. Then we also need to add more constraint conditions on to increase the overall accuarcy.