Color Segmentation

Chao Qu*

University of Pennsylvania
quchao@seas.upenn.edu

I. Introduction

Color segmentation is a common and useful technique in computer vision and robotics. The goal is to change the representation of an image to different segments each with a distinctive color. One of the application of color segmentation is to locate objects and boundaries in images. In this project, we are given training images with a red barrel and are asked to calculate the depth of the barrel on testing images using color segmentation and probabilistic models. This document briefly describes the workflow and methods that I adopted to complete the task.

II. Pre-processing

Before doing any actual learning, we need to pre-process all the training images.

The first thing we do on the images is to **sub-sample** them. Since the original images come with high resolution of 1200×1600, it will take a long time to go through all the pixels. We thus sub-sample all the images down to $1/4$ its original scale, which is now 300×400 (shown in Fig.1. This speeds up the whole process significantly, while preserving most of the information in the image.

*A thank you or further information

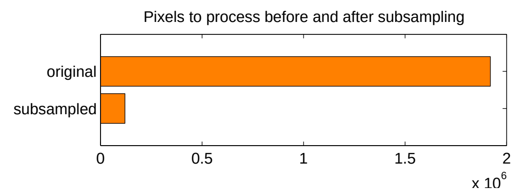

Figure 1: *Number of pixels in original and subsampled images*

We then divide all the training images into a **training** set and a **validation** set (shown in Fig.2, with 80% and 20% in each. That is approximately 40 images for training and 10 images for testing, among all 50. We also notice the fact that there are 10 distinct distances of the barrel, so we split randomly in each distance category 4 images to the training set and 1 image to the validation set. This step is for the purpose of **model selection** with **cross validation** in the training phase later.

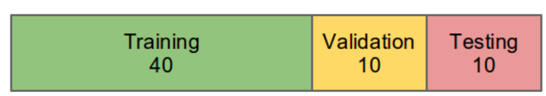

Figure 2: *Split all images into* 80% *training and* 20% *testing*

As suggested in the project write-up, the default RGB colorspace is not recommended, as it has problems with different lighting condition. We then explore some other colorspace options that are supported in MATLAB. The colorspaces that are being inspected are **RGB**, **YCbCr**, **L*a*b** and **HSV**. We use 5 images with distance 1 to generate the 3D scatter plot in different colorspaces (shown in Fig.3). Red dots are pixels from the barrel while dots in other colors are pixels from the rest of the images. As can be seen from the figure, the **L*a*b** and **YCbCr** are much better than the rest colorspace in the sense that they have more concentrated clusters.

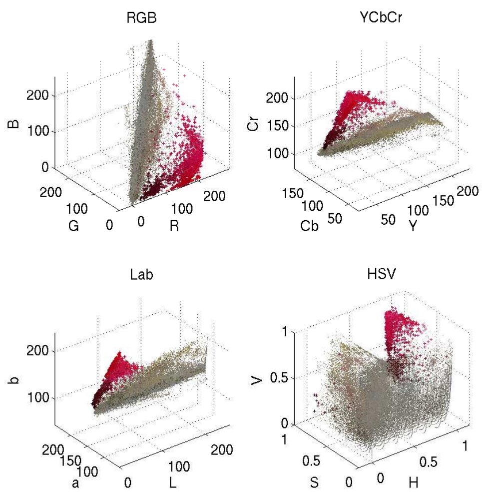

Figure 3: *3D scatter plot for different color spaces*

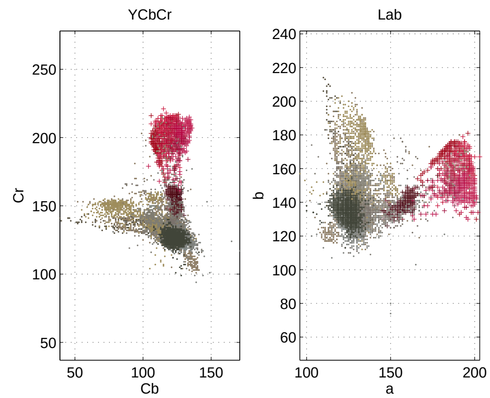

Figure 4: *2D scatter plot on the 2 color channels of L*a*b and YCbCr*

Also, we notice that in the 2d plot of L*a*b, there are two clusters of red pixels. The darker but smaller cluster is closer to non-barrel pixels while the lighter ones are far away. This is due to the fact that the images are taken under different lighting conditions. This observation encourages us to learn maybe two models for light and dark conditions or maybe to use a **Gaussian Mixture Model**. An example of confusing images under different lighting conditions are shown in Fig.5.

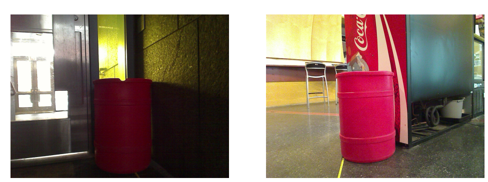

Figure 5: *Images under different lighting conditions.*

A closer look at the two colorspace reveals that pixels in the **L*a*b** colorspace that represent the red barrel are more concentrated than the rest colospaces (shown in Fig.4. Thus we pick L*a*b and convert all RGB images to that colorspace.

Finally, in order to tell our algorithm which part of the image contains the barerl, we hand labelled all the barrels in training images and creating a mask for each image (shown in Fig.6).

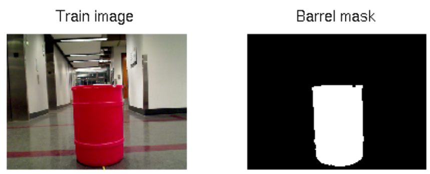

Figure 6: *Barrel mask for each training image*

III. METHODS

After pre-processing the data, I start playing with different methods and models. Here I describe the learning and predicting workflow that I use for color segmentation and barrel detection.

I. Barrel Shape Model

Since we already have a black-white image to mask the barrel, we could learn some useful information from that. We looked at various properties of the part of mask being labelled and decided to use the aspect ratio and percentage of filled pixels to model the barrel. From the training image, we extract all those information and build a 2D Gaussian distribution. And for any new mask coming in, we will be able to tell the probability of it being a barrel or not. The equations used for estimating such a multivariate gaussian is shown in (Eq.1 and 2) and the result are shown in Fig.7.

$$\mu_{MLE} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}^i \tag{1}$$

$$\Sigma_{MLE} = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}-\mu)(\mathbf{x}-\mu)^\top \tag{2}$$

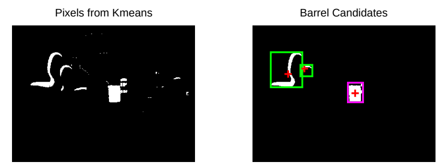

Figure 7: *Using barrel model to select the most possible shape*

II. Distance Model

For distance estimation, we adopt the simple approach of ridge regression with 3 features: height and width of the bounding box and square root of the area of the mask. The equations used for estimating such a ridge regression is shown in (Eq.3).

$$\mathbf{w}_{MLE} = (\mathbf{X}\mathbf{X}^\top + \lambda\mathbf{I})^{-1}\mathbf{X}^\top\mathbf{y} \tag{3}$$

III. K-means Clustering

For color segmentation, we first use the K-means algorithm on channel a and b of the lab colorspace image. We start with an initial clusters of 4 and keep increasing until we get a shape that has a high probability of being a barrel. For most of the image, the barrel can be segmented with in 1 or 2 iterations or K-means. For those harder cases, it can also be found at higher cluster number. Since there are two different lighting conditions, we compare the L channel of each image to a lightness threshold and based on that we select which cluster to be the barrel or red (shown in Fig.**??**).

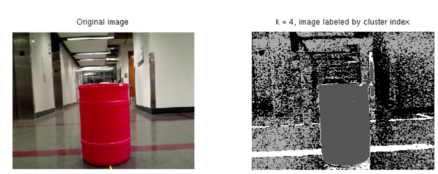

Figure 8: *Sample results of K-means*

IV. Gaussian Mixture Model

When K-means fails, GMM comes in. Since K-means can do good enough on most of the images, we use GMM only for those cases where K-means cannot detect a good shape of barrel. We use all the pixels in the aforementioned masks to train a GMM model for red with 2 mixtures. Then for each given image, we calculate the probability of them being red. We then threshold and normalize them to create a black white image that is similar to what we get from K-means. Then we apply the same process to the black white image and get the most possible shape to be a barrel (shown in Fig.9).

Figure 9: *Sample results of GMM*

IV. DISCUSSION

I. RESULTS

In this section we show some of results detected by our model. We first look at all the estimated depth result (shown in 10).

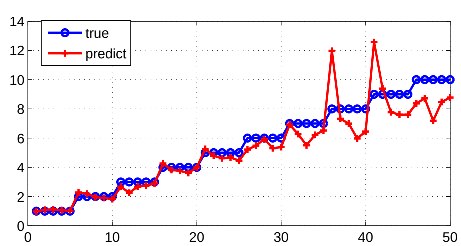

Figure 10: *Estimate depth vs. true depth*

We then look at some of the detected barrels.

Figure 11: *Sample detected barrel*

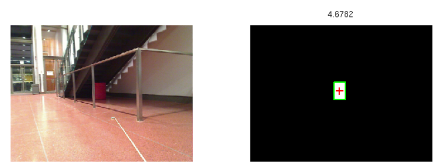

Figure 12: *Sample detected barrel*

Figure 13: *Sample detected barrel*

II. IMPROVEMENT

There are several improvements that can be made to this project.

1. Increase the speed of GMM.
2. Train a better model on the barrel.
3. Train a better mode for distance estimation.