Barrel Detection using Color Segmentation based on GMMs

Nitin J. Sanket
School of Engineering and
Applied Science
University of Pennsylvania
Email: nitinsan@seas.upenn.edu

***Abstract*—This project presents an approach for robust color segmentation which was further used to detect a red barrel based on shape statistics. Several algorithms are presented to tackle variations in illumination, occlusion and tilt. The color segmentation was accurate for all the images in the test set, however, the barrel detection algorithm missed one of the barrels in the test set. The shape statistics were finally used to compute the distance to the barrel along the camera Z axis.**

I. Problem Statement

The aim of the project was to segment out colors representing a 'red' barrel given 50 training images by building a probabilistic model, use shape information to compute the distance to the barrel along the camera Z axis. The algorithm was tested on 10 unseen images.

II. Training

The whole procedure of training is discussed in the following sub-sections.

A. Acquiring Data Samples

The training set was split into 2 sets, i.e., training set containing 41 images and a held out set containing the remaining 9 images. The images were picked so as to have both dark and light, near and far images in the testing and held out set. The masks which depicted the 'barrel red' pixels were manually labeled. A sample RGB image and its corresponding mask is shown in Fig. 1. All the RGB values from all the pixels corresponding to 'barrel red' from the chosen 41 training images were used stacked (GMM has a slightly different procedure which is later explained) to get the training set. Let us say this was of the size $N \times D$ where D is the number of dimensions which is 3 in our case and N was the order of 0.6 Million. Next section talks about color space conversion and the need for it.

B. Alternative Color Spaces

It is a well known fact that RGB color space is not robust to illumination variations but it is used because it is the most intuitive way we perceive color. To make the algorithm robust to illumination, YCbCr color space was used. A RGB image with this R, G and B channels is shown in Fig. 2. Clearly one can observe that the red channel has a high value for the barrel but it also has a high value for white colored things. Also, the green channel dominates the intensively calculation and hence is not variant to illumination. This problem is solved to some extent in YCbCr color space and its components are shown in Fig. 3. RGB to YCbCr converstion is given in below.

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.5 \\ 0.5 & -0.81 & -0.81 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

As we are looking at a red barrel, red channel stands out the most however this value is relative hence I made a custom color space (rYb) based on Rred-Green Chromaticity and YCbCr (To account for illumination variations). The 3 channels r, Y and b are defined as follows:

$$\begin{aligned} r &= \frac{R}{R+G+B} \\ Y &= 0.299R + 0.587G + 0.114B \\ b &= \frac{B}{R+G+B} \end{aligned}$$

rYb color space and its components are shown in Fig. 4

Clearly the barrel stands out much better in YCbCr and rYb color spaces than RGB. A visualization of all the datapoints from 41 images plotted in the 3 different color spaces is shown in Fig. 5.

Ideally, all the points should constitute as small a area as possible to show that they can be fit by a gaussian with low variance. Clearly, rYb has the smallest area followed by YCbCr and then worst being RGB as expected. So all the further steps were carried out on rYb and YCbCr color spaces.

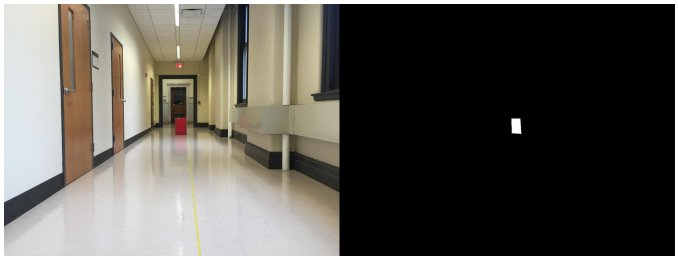

Fig. 1. Left image shows the RGB image and Right image shows the hand labeled mask corresponding to 'barrel red' pixels.

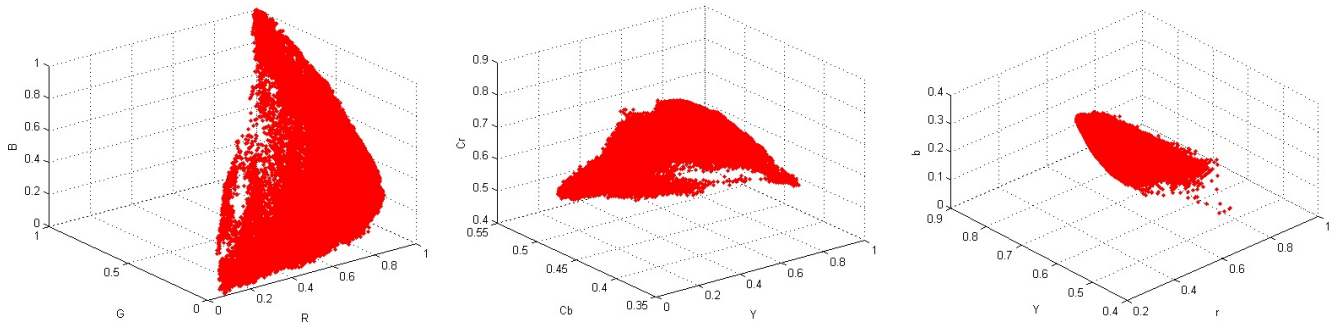

Fig. 5. Left to right: Data points in RGB, YCbCr and rYb color spaces.

Fig. 2. Left to right: R, G, B, RGB channels of the image.

Fig. 3. Left to right: Y, Cb, Cr, YCbCr channels of the image.

Fig. 4. Left to right: r, Y, b, rYb channels of the image.

C. Fitting a single Gaussian Model

The simplest model we can fit is a single gaussian to the data. The gaussian represents $P(x|c_l)$ and is given by

$$P(x|c_l) = \sqrt{\frac{\det A}{(2\pi)^3}} e^{\frac{-1}{2}(x-\mu)^T A(x-\mu)}$$

$$\mu = \frac{1}{N}\sum_{\nu} x^{\nu}$$
$$A^{-1} = \frac{1}{N}\sum_{\nu} (x^{\nu}-\mu)_i (x^{\nu}-\mu)_j$$

The surface plot of the fitted gaussians for the 3 color spaces are shown in Fig. 6.

Clearly the volume occupied by the ellipsoids decreases as we go from RGB to YCbCr to rYb. The next more complex model was to fit a Gaussian Mixture Model (GMM) with shared diagonal co-varience (Form $\Sigma = \sigma^2 I$). This leads to spherical gaussians of same size. A sample output for spherical GMM on RGB color space is shown in Fig. 7.

The next logical step was fitting a full fledged GMM with variable variances and cluster weights.

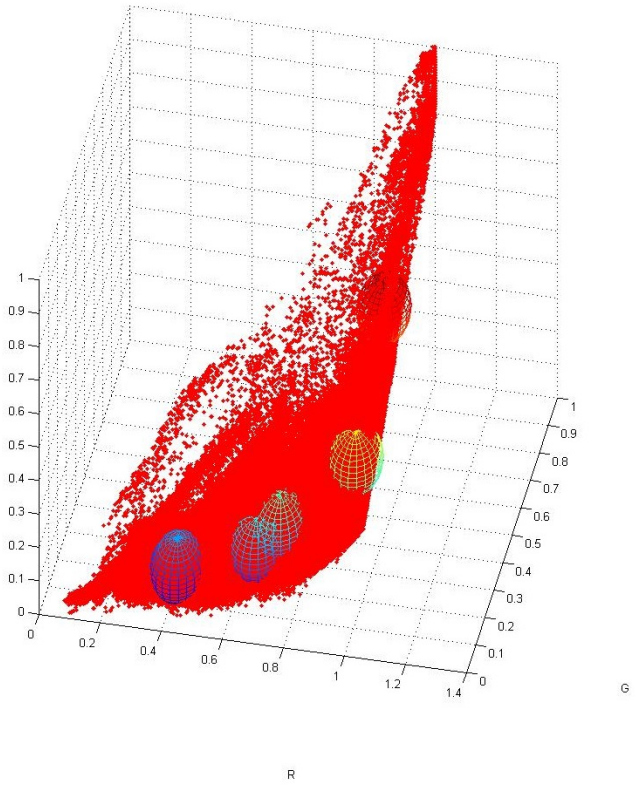

Fig. 7. Spherical GMM output on RGB color space.

D. Gaussian Mixture Model with Non-Shared Variances and cluster weights

The GMM algorithm works by using a expectation maximization procedure. The following equations govern the GMM.

$$P(x|c_l) = \sum_k \pi_k \sqrt{\frac{\det A}{(2\pi)^3}} e^{\frac{-1}{2}(x-\mu_k)^T A(x-\mu_k)}$$

The above equation gives the likelihood of datapoint given the color label. In the E Step, compute the cluster weights are

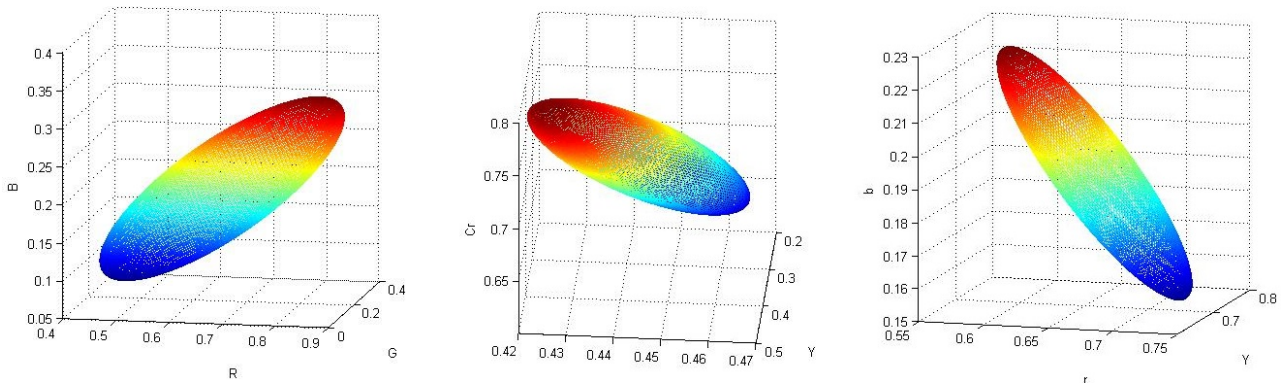

Fig. 6. Left to right: Single gaussian model surface plot of ellipsoid in RGB, YCbCr and rYb color spaces.

calculated as

$$\alpha_k^\nu = \frac{P\left(x^\nu|c_l\right)\pi_k}{\sum\limits_k P\left(x^\nu|c_l\right)\pi_k}$$

In the M Step, compute the means, covariances and cluster weights as

$$\mu_k = \frac{\sum\limits_\nu \alpha_k^\nu x^\nu}{\sum\limits_\nu \alpha_k^\nu}$$
$$\Sigma_k = \frac{\sum\limits_\nu \alpha_k^\nu (x^\nu-\mu_k)^T(x^\nu-\mu_k)}{\sum\limits_\nu \alpha_k^\nu}$$
$$A = (\Sigma_k)^{-1}$$
$$\pi_k = \frac{1}{N}\sum_\nu \alpha_k^\nu$$

To compute the $(x-\mu_k)^T A\,(x-\mu_k)$ term fast,

$$(x-\mu_k)^T A\,(x-\mu_k) = \sum_i {x_i}^T \otimes X r_i^T$$

Where x_i^T is the i^th column (size $N\times 1$) of the stacked mean centered values X which is of size $N\times D$. $\otimes$ is the element-wise multiplication operation. r_i^T is the i^th column of $D\times D$ A matrix. Here i refers to the dimension, i.e., for RGB it is 1, 2 or 3. The algorithm for the GMM algorithm using EM is given below [1]:

Data: ColorSpace Datapoints
Result: K Gaussian mixtures which model the data
Choose π, μ, Σ randomly or using K-Means;
while $\left\|\mu^t - \mu^{t-1}\right\| \geq \tau$ **do**
  Estimate $P(c_l|x) \propto \pi_k N(x_i;\mu_k,\Sigma_k)$ Estimate new μ_k, π_k, Σ_k as given above
end

Algorithm 1: E-M algorithm for GMM.

A plot of GMM outputs for all color spaces is shown in Fig. 8. Surprisingly, all the gaussians converged to a single gaussian in rYb showing that it has the most compression power of the 3 color spaces.

Another thing I experimented was with different ways of initializing the GMM. I used the output of K-Means to initialize my GMM. This was more consistent in terms of convergence where convergence was defined as norm of difference of means should be within a particular threshold. K-Means consistently gave a good convergence speed and almost similar outputs everytime. However, random initialization sometimes gave better segmentation results but also sometimes gave complex values in covariance matrix and hence divergence. A sample output for K-Means and Rand initialization for YCbCr color space is shown in Fig. 9.

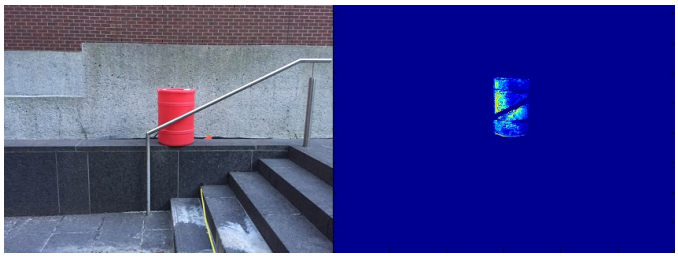

Fig. 10. Left to Right: Input RGB image, $P(x|barrel)$.

Selection of K:
For selecting the value of K (number of clusters) for GMM, these 41 images were divided into 35 and 6 images randomly and this was done 5 times to choose the best K. Final GMM model used in testing was retrained on the 41 images with these K number of clusters.

III. Testing/Evaluation Methodology

First step was computing the probability of datapoint given barrel using GMM. This was done using

$$P\left(x|c_l\right) = \sum_k \pi_k \sqrt{\frac{\det A}{(2\pi)^3}} e^{\frac{-1}{2}(x-\mu_k)^T A (x-\mu_k)}$$

I assumed that the $P(x)$ and $P(c_l)$ was a constant due to drawing from a uniform random distribution and neglected them. Hence I used $P(x|c_l)$ as the estimate for $P(c_l|x)$. A sample input RGB image and its corresponding $P(x|barrel)$ are shown in Fig. 10.

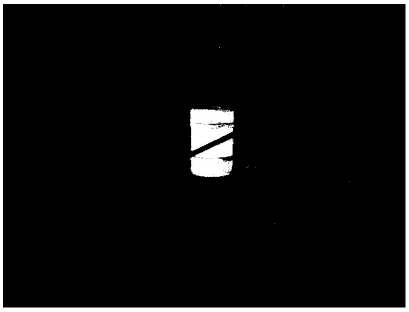

Fig. 11. Mask (M) obtained by thresholding $P(x|barrel)$.

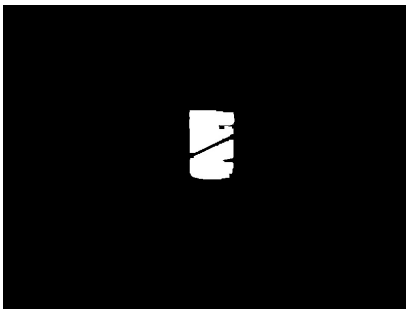

Fig. 12. M after some filtering (M1).

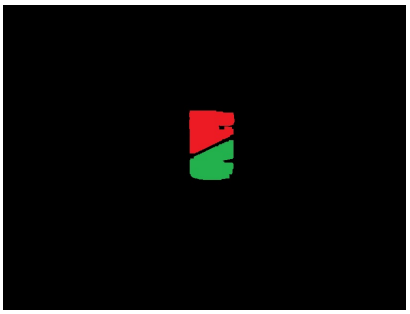

Fig. 13. M1 after labelling.

This was then thresholded based on a fraction of maximum value to obtain a Mask. The mask is shown in Fig. 11. Then the blobs in the mask after erosion which do not satisfy some area and solidity criterion are removed to remove some stray blobs like the one belonging to a bicycle and so on. Then the blobs are dilated to factor in for the erosion, sometimes this step merges the blobs due to occlusion. However, this is not very robust for occlusion removal. Hence a custom algorithm for combining blobs based on solidity was written. Consider the image shown in Fig. 12, this image after labelling is shown in Fig. 13. A simple illustration of the process is shown in Fig. 14. Here, each blob is compared to every other blob and if they satisfy a distance threshold (closer than some distance), they and combined and the resulting solidity is measured. If the solidity satisfies some threshold they are kept. In Fig. 14 the blobs are combined and their convex hull is found, this is shown in Fig. 15.

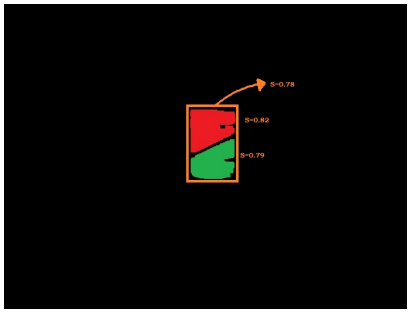

Fig. 14. Illustration of how combining blobs work.

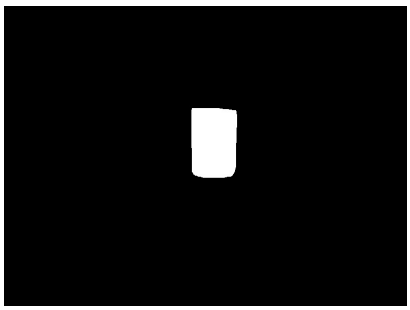

Fig. 15. Blobs combined.

Next step was to compute the oriented bounding boxes and blobs that do not satisfy a range of aspect ratio are removed. The distance is estimated using a linear regression model fit on inverse height and inverse depth of the oriented bounding box.

IV. Results on the Test Set

The results on YCbCr color space for the Test Images are shown in Figs. 16, 17, 18, 19, 20, 21, 22,23, 24 and 25. Here the yellow highlights indicate the initial mask obtained by titleholding $P(x|c_l)$, the green highlights indicate the final mask after all the blob elimination process. The red dot indicates the centroid, the red box shows the oriented bounding box. The distance and orientation are indicated in a white box next to the centroid.
The results on rYb color space for the Test Images are shown in Figs. 26, 27, 28, 29, 30, 31, 32,33, 34 and 35. Here the yellow highlights indicate the initial mask obtained by titleholding $P(x|c_l)$, the green highlights indicate the final mask after all the blob elimination process. The red dot indicates the centroid, the red box shows the oriented bounding box. The distance and orientation are indicated in a white box next to the centroid.

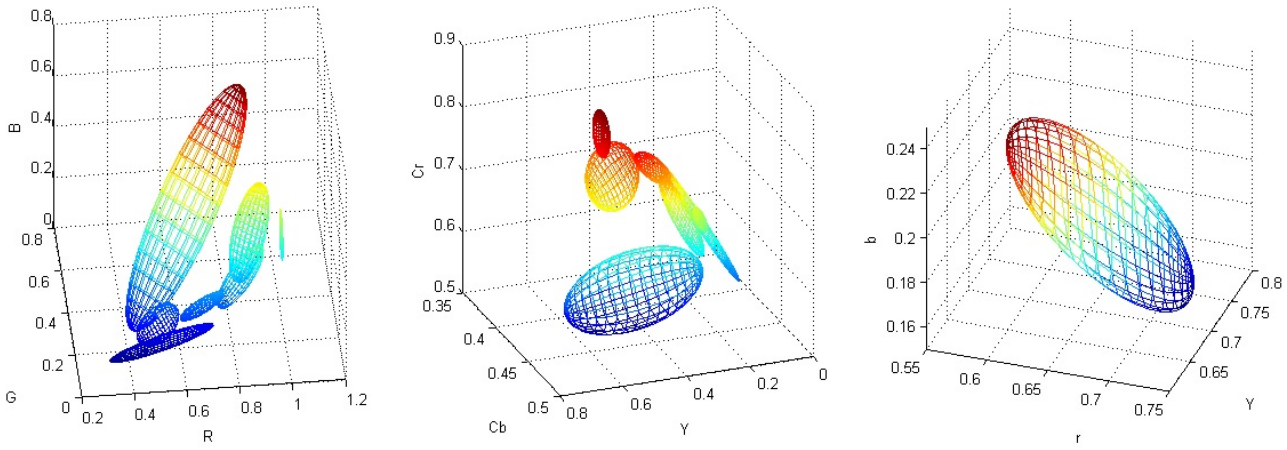

Fig. 8. Ellipsoidal GMM output on RGB, YCbCr and rYb color spaces.

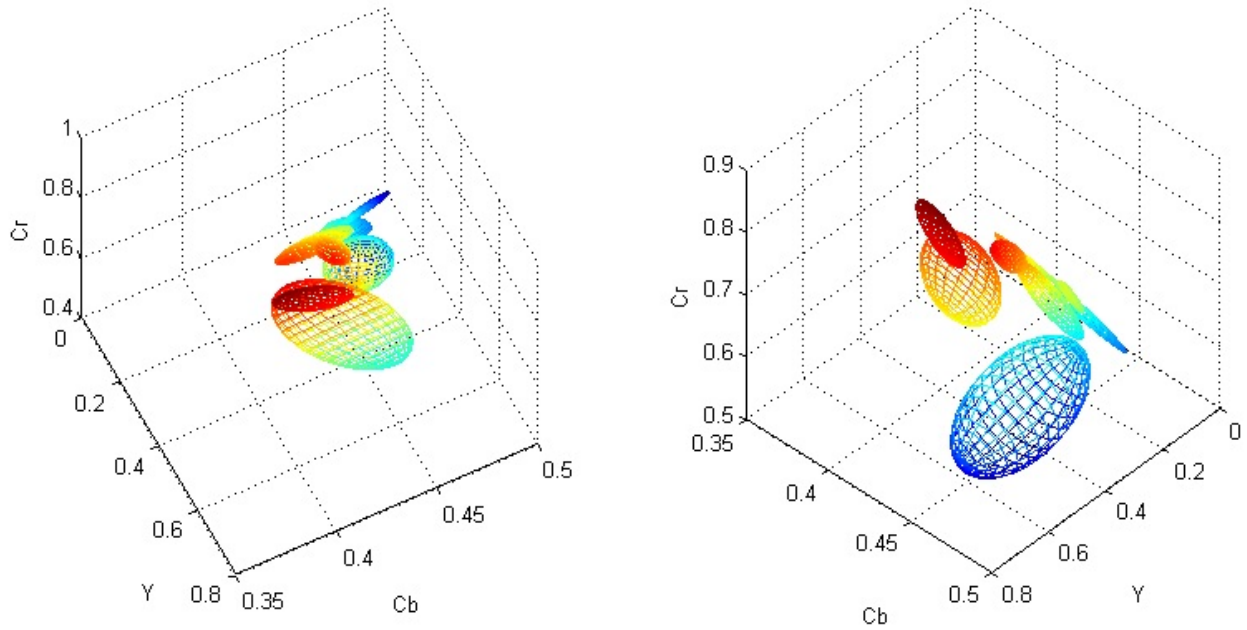

Fig. 9. Left to Right: K-Means and Rand initialization convergence results on YCbCr color spaces.

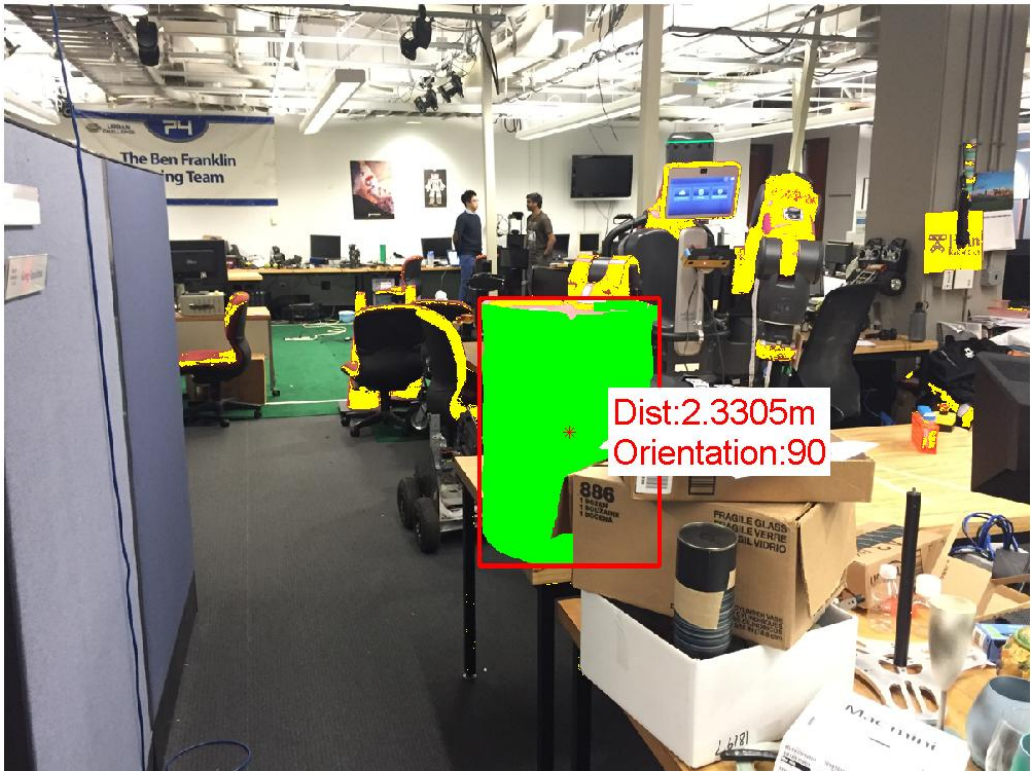

Fig. 16. Output YCbCr 1.

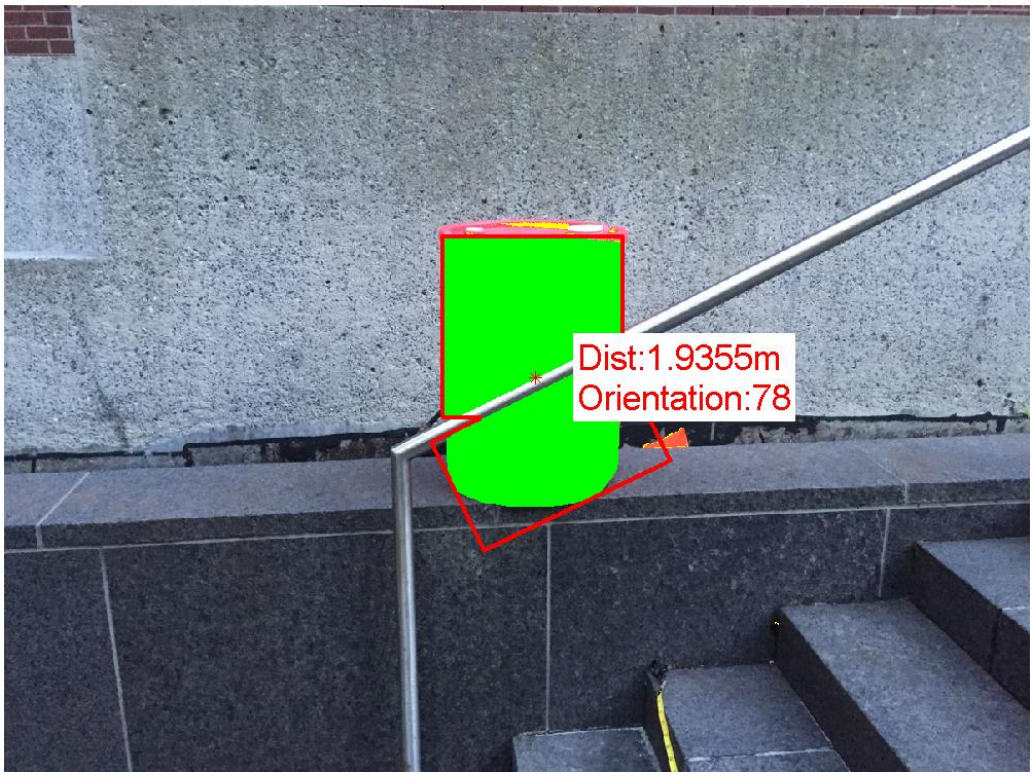

Fig. 17. Output YCbCr 2.

Fig. 18. Output YCbCr 3.

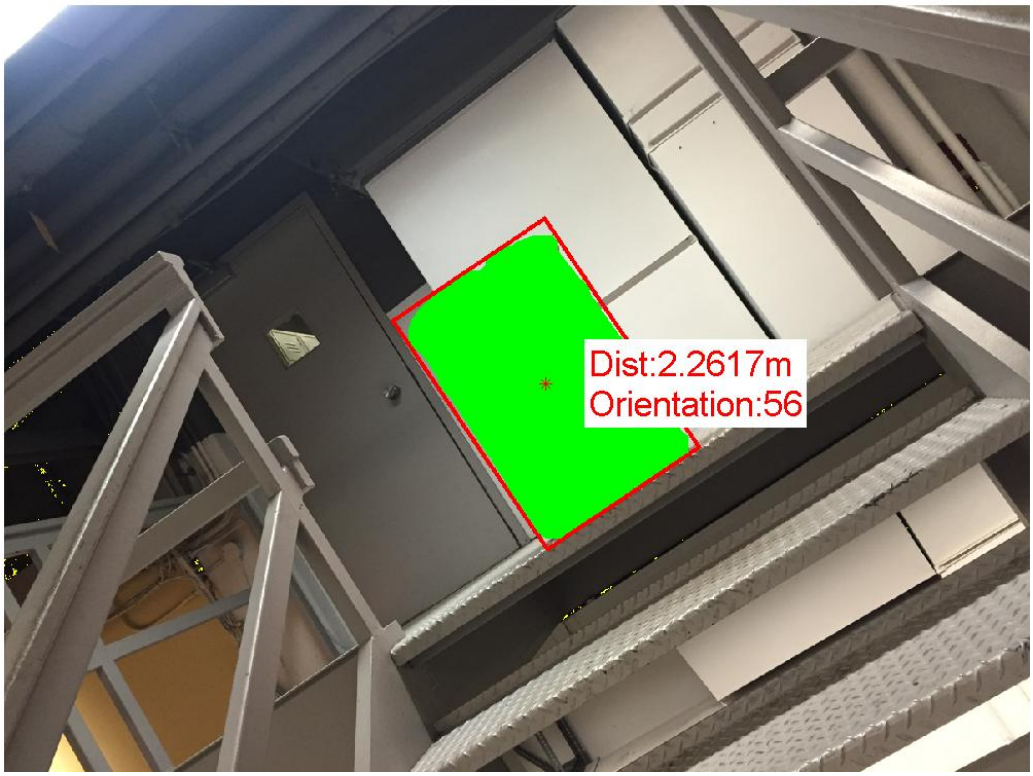

Fig. 19. Output YCbCr 4.

Fig. 20. Output YCbCr 5.

Fig. 21. Output YCbCr 6.

Fig. 22. Output YCbCr 7.

Fig. 23. Output YCbCr 8.

Fig. 24. Output YCbCr 9.

Fig. 25. Output YCbCr 10.

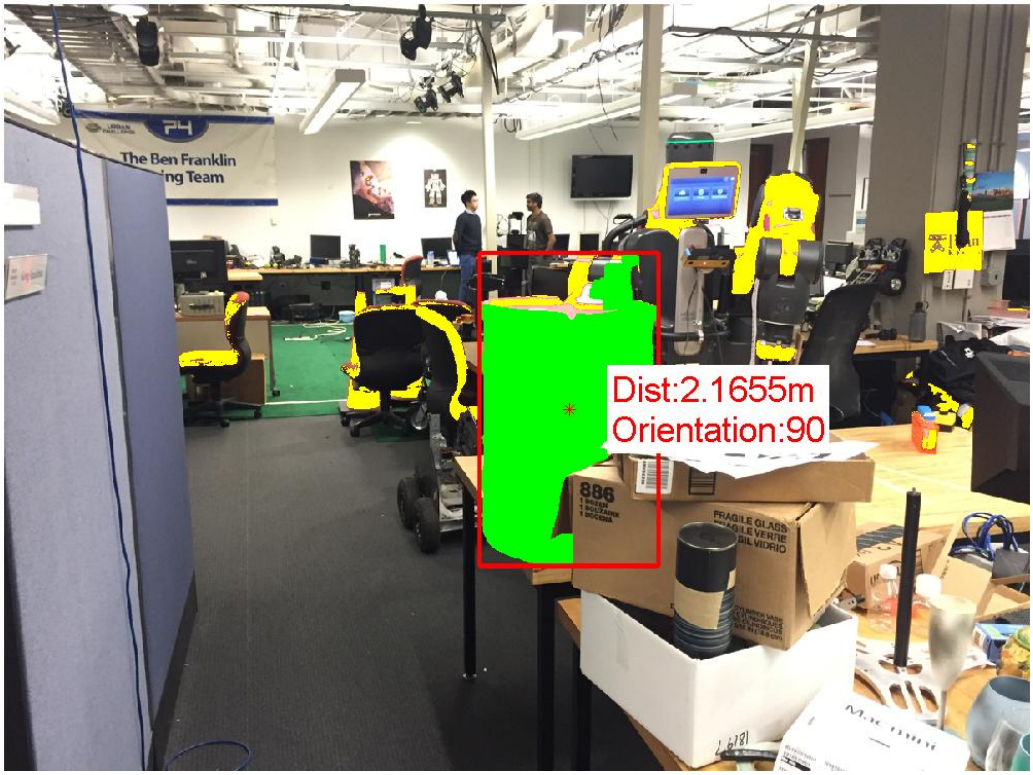

Fig. 26. Output rYb 1.

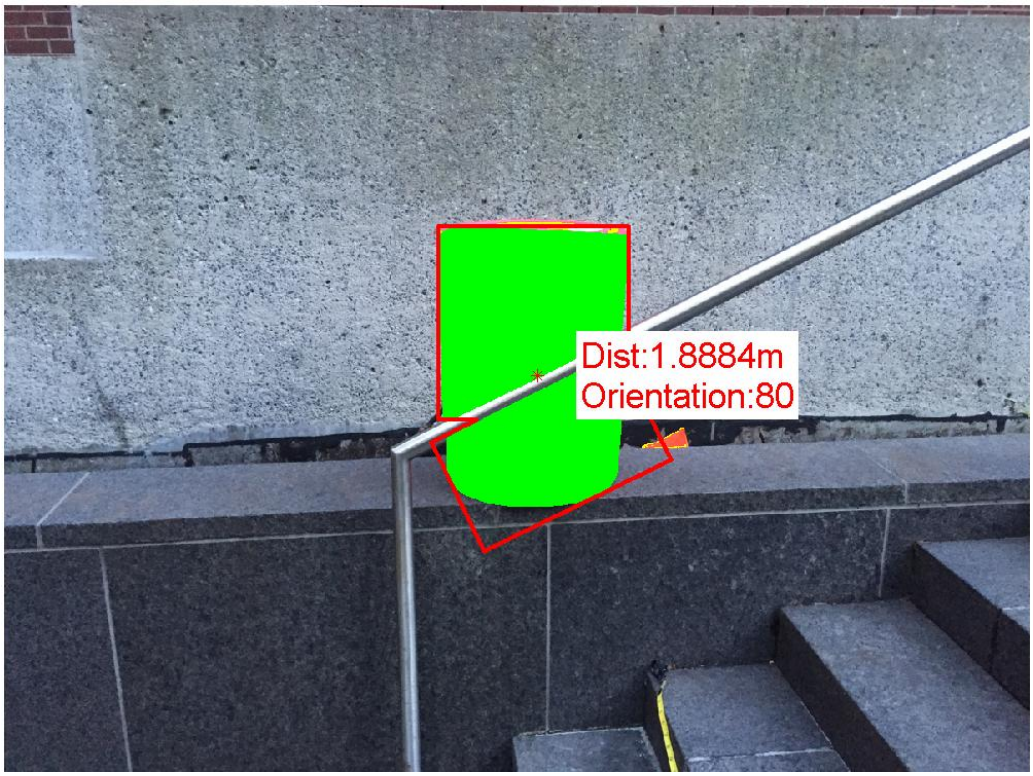

Fig. 27. Output rYb 2.

Fig. 28. Output rYb 3.

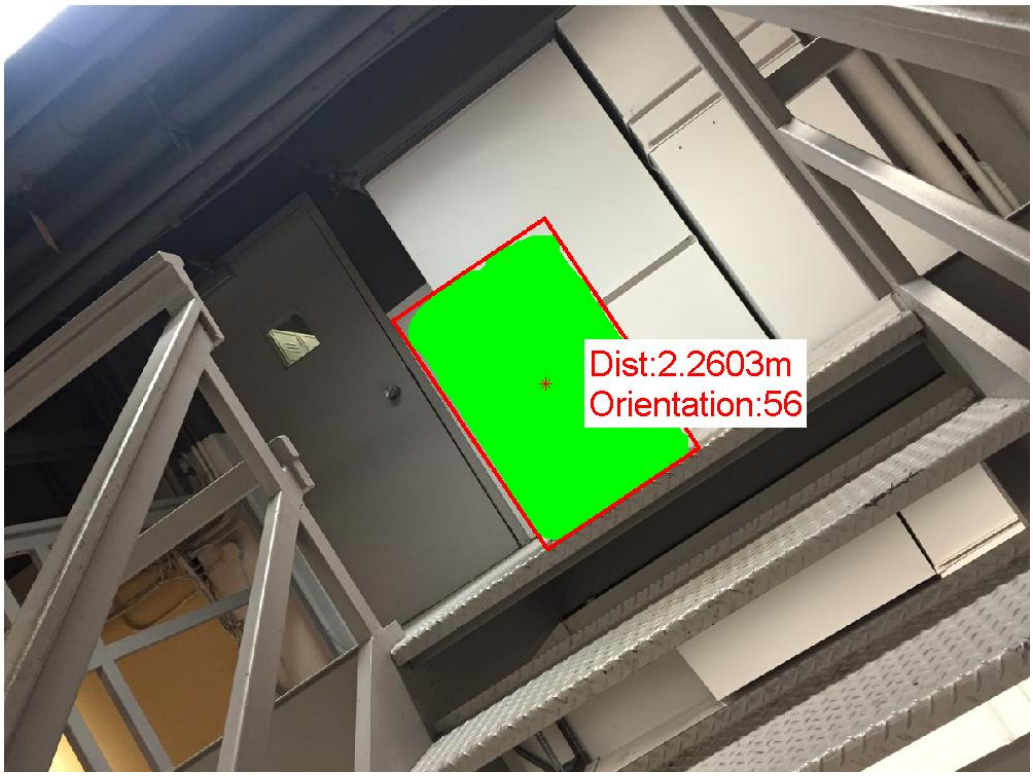

Fig. 29. Output rYb 4.

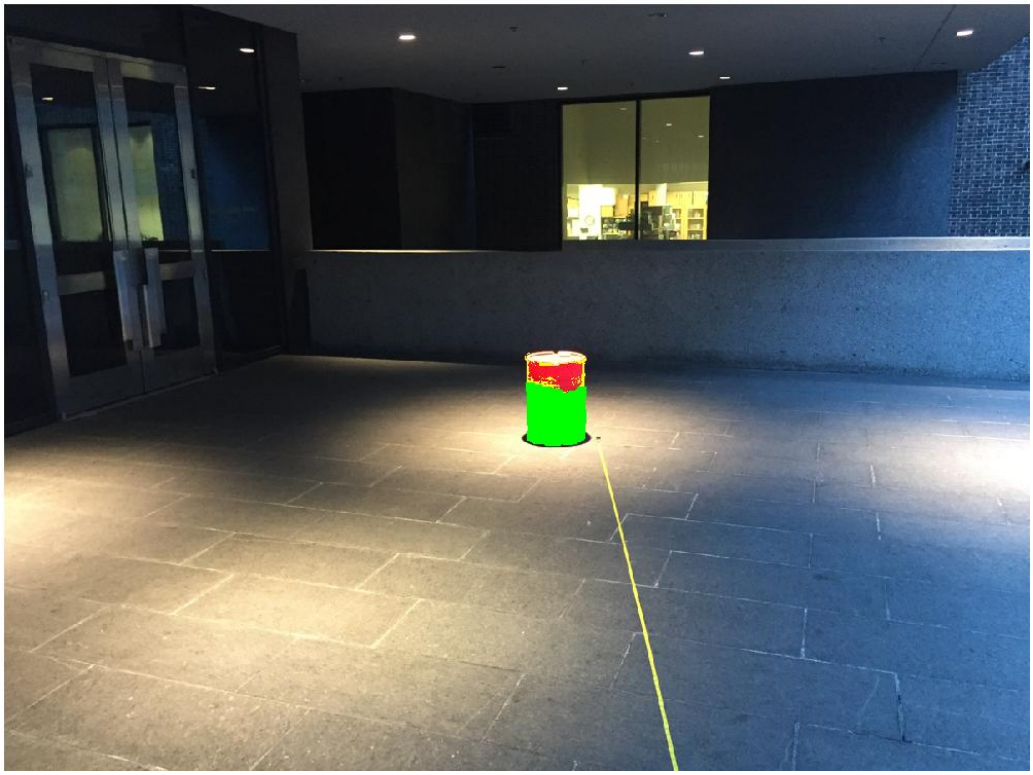

Fig. 30. Output rYb 5.

Fig. 31. Output rYb 6.

Fig. 32. Output rYb 7.

Fig. 33. Output rYb 8.

Fig. 34. Output rYb 9.

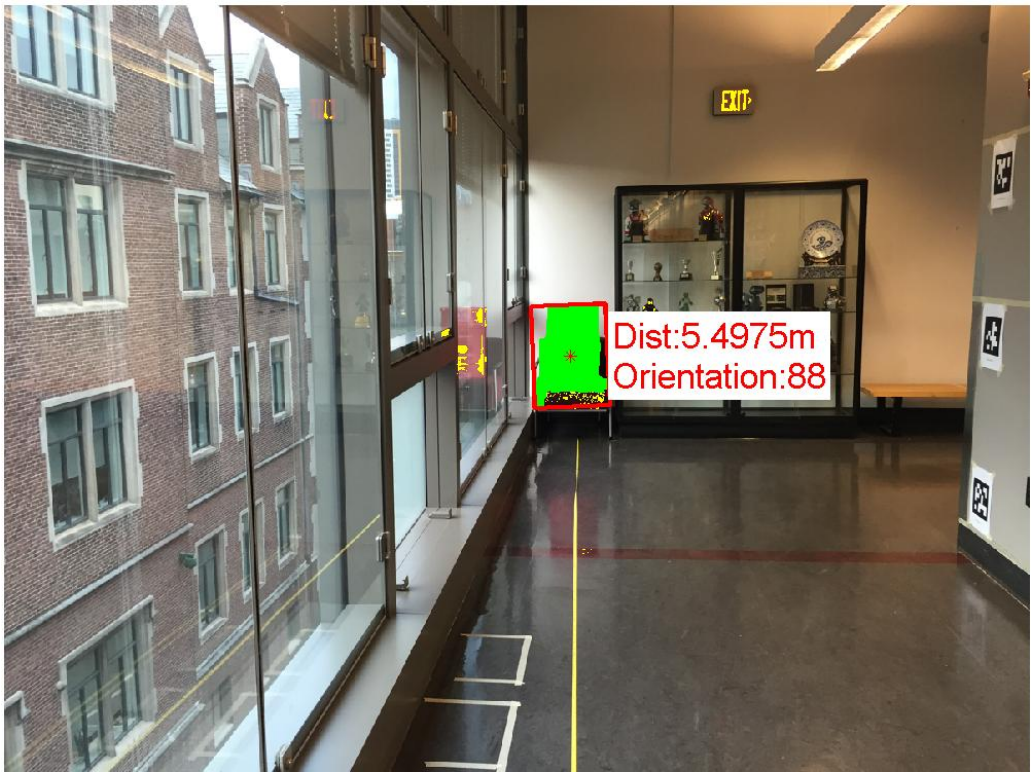

Fig. 35. Output rYb 10.

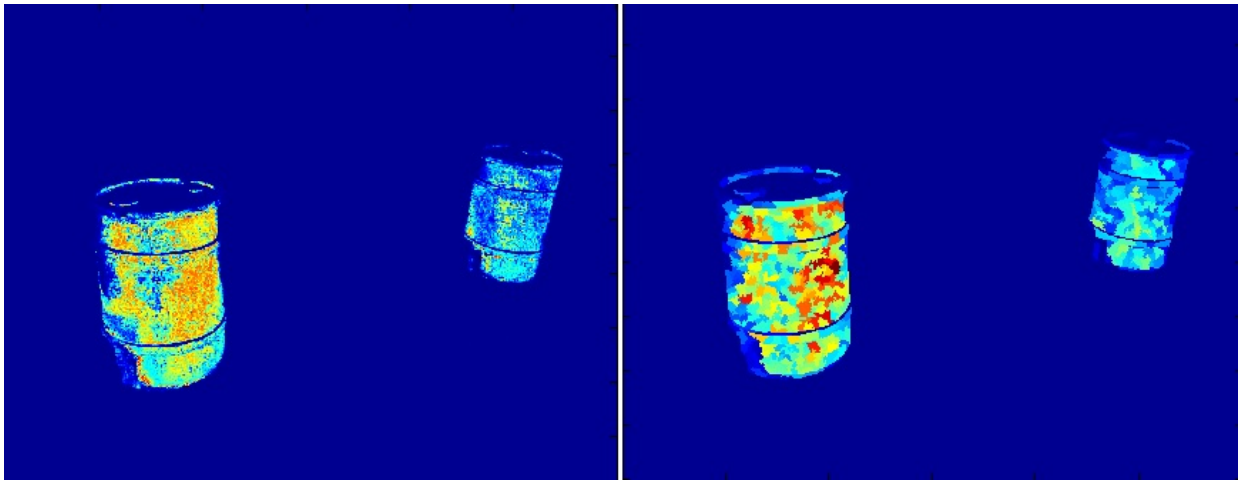

Fig. 36. Left to Right: $P(x|c_l)$ computed using pixels and super-pixels.

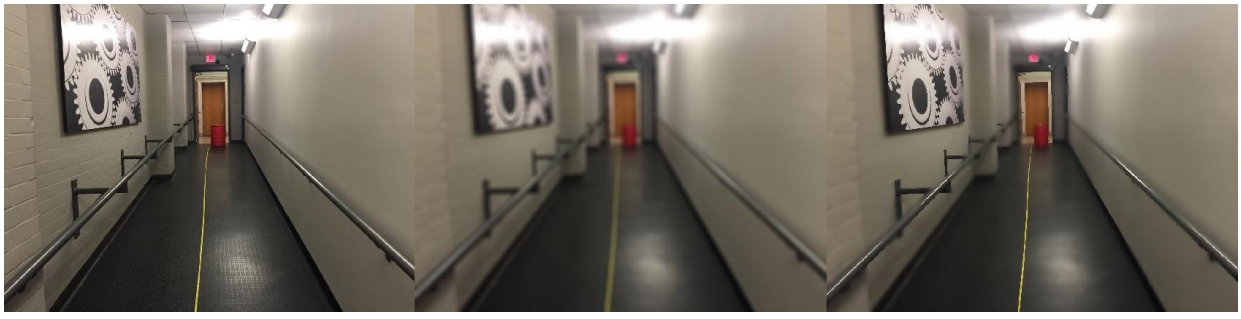

Fig. 37. Left to Right: Original Image, Image blurred using average/box filter, Image blurred using anisotropic diffusion.

A. Analysis of the Results

If the blobs were too big, the combined bounding box had a wierd shape like Figs. 17 and 27, this is due to the combination algorithm and the way I generated the convex hull using oriented bounding boxes. Both the color spaces failed in the image where the barrel was too bright, this is because my training set did not have very bright images (Refer to Figs. 20 and 30). Both my color spaces took the chair red color thereby underestimating the distance because my training images had more dark images (Refer to Figs. 25 and 35).

V. Other Interesting Stuff I did

A. Superpixel Segmentation to improve probability scores

If the illumination variation is severe on the barrel, the pixel probabilities can jump a lot, blurring the image will change the pixel values to something other than red. For eg. i.e., if light is shining directly on the barrel and the barrel is shiny, the pixels might be white, blurring it will give us pink or orange which is undesirable. So a more robust probability assignment can be obtained by the usage of superpixels. 5000 superpixels generated from Entrpoy Rate Superpixel Segmentation [2] was used to achieve this. A sample probability score output using pixels and super pixels are shown in Fig. 36. We can clearly observe the improvement in scores in the super-pixel based assignment. In each super-pixel all the pixels get multiplied by the ratio of maximum value in that super-pixel and mean value in that super-pixel. This greatly improved the overall results by making all the thresholds robust, but it was too slow for a slight improvement. The code without super-pixels ran in 3secs and with super-pixels ran in 90s. A 30X slowdown for a slight improvement was not justifiable.

B. High Frequency Noise Removal using Anisotropic Diffusion

Simple blurring with a gaussian or a box filter will generate wrong pixel values for training as the pixels are blurred across the edge. Hence, an adaptive blurring technique called Anisotropic Diffusion was used, which uses a heuristic to blur within similar regions and not across. A sample output comparing box filter and anisotropic diffusion is shown in Fig. 37.

VI. Important Lessons Learnt

Overfitting is a big problem and cross validation helps a lot (which I did).
Learn to accept a few false positives to improve overall precision (which I didn't do).
Keep thresholds as dynamic as possible (which I did).
Sometimes silliest of the methods work the best (Which I did).

VII. Conclusions

Overall, both rYb and YCbCr colorspaces outperformed RGB, with rYb performing slightly better than YCbCr albeit the numerical issues in MATLAB. I could segment the barrel colored pixels from a crowded scene, reject non-barrel blobs and combine split barrel blobs successfully. Occlusion posed a bit of a challenge but was combated to a great extent. The distance estimates on the training set were accurate to about $\pm 0.5m$.

Acknowledgment

The author would like to thank Dr. Daniel D. Lee and all the Teaching Assistants of ESE 650 course for all the help in accomplishing this project.

References

[1] Expectation-Maximization Theory, http://www.informit.com/articles/article.aspx?p=363730&seqNum=2.

[2] Liu, Ming-Yu, et al., *Entropy rate superpixel segmentation*, IEEE Conference on Computer Vision and Pattern Recognition, 2011.