

Using Gaussian Model for Color Classification and Image Segmentation

ECE 276A Sensing & Estimation in Robotics, Project 1

Sheng-Wei Chang

Department of Electrical and Computer Engineering
University of California, San Diego
s7chang@eng.ucsd.edu

Abstract—Color segmentation has always been a common topic for a long time. The related fields including machine, computer vision, etc. This project uses a single Gaussian model to execute color segmentation. The goal in the project is to detect the stop sign in an input image. The result shows that the single Gaussian is a good way to segment color. In many cases, stop signs are detected eventually.

Index Terms—Gaussian model, Color segmentation, Morphological Operations

I. INTRODUCTION

Image color segmentation is a common and important topic. With the technique in computer vision and machine learning, we can find the properties of a given image. Furthermore, we can find areas with the same textures and approach the detection problem. In this project, the object is to segment the stop sign out of given images. This problem is important for self-driving cars to decide whether they should stop at the cross. Basically, we can set a threshold for specific color space (RGB, HSV, YCbCr). After trial, we can moderate the threshold and get a relatively good result. However, there are some disadvantages to the method. First of all, this is an inefficient way. We may have to experiment repeatedly to get a better threshold. Furthermore, for the lighting issue, a fixed threshold may not be able to distinguish light red and dark red. Third, this model may not work well in different environments. For these drawbacks, we need a more proper model to solve this problem.

In this project, I choose the single Gaussian model to approach this problem, using training images to approach parameters and then classify each pixel in a given image into a color class. Then we can get a binary image as mask to distinguish the stop sign red and other colors. After classification, I compute the properties of each region of same color in the mask, trying to score them based on how much they resemble a stop sign shape.

II. PROBLEM FORMULATION

A. Color Classification

I choose the single Gaussian model to approach this problem because I assume that in various situations, the

distribution of the stop sign color in the color space will be a normal distribution. Gaussian function is a common probability density function, of the form:

$$f(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2} \frac{(x - \mu)^2}{\sigma^2}\right)$$

f is the function of x , given μ , the mean and σ^2 , the variance. The equation above is for one-dimensional. The multivariate Gaussian distribution if the function of the form:

$$f_X(x|\mu, \Sigma) = \frac{\exp(-\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu))}{\sqrt{(2\pi)^d |\Sigma|}}$$

f is the function of vector x , given the mean vector: μ and the covariance matrix: Σ d is the dimension of x . Since the color space is three dimensions, no matter RGB, HSV or YCbCr, d is 3. x and μ is 3×1 vector. Σ is the covariance matrix of size 3×3 , the covariance matrix has to be a Positive Semi-Definite (PSD) matrix.

For each x , which means the pixel in the given image, we are going to find the maximum likelihood of a class given a data set.

$$h(x) = \arg \max_y P(x, y) = \arg \max_y P(x|y)P(y)$$

y is the color classes. We pick the color which has the maximum likelihood, and assign the given pixel as that color. Then, we can get the result of the color segmentation.

B. Morphological Operations of Mask

After classifying a color image into our Gaussian model, a color image set is then transformed into a binary image set. We can obtain a mask that the white area in the mask shows the potential stop sign area. However, there might be some noises distracting the model from detecting the stop sign. Also, some areas might not be a stop sign, but they have similar colors with a stop sign, so they are segmented in the mask. To solve these situations, we will use some morphological operation to distinguish stop sign shape. There are three main purposes for applying morphological operation:

- (1) Removing noise
- (2) Isolation of individual elements and joining disparate elements in an image
- (3) Finding of intensity bumps or holes in an image

Two basic methods of Morphological operation are Erosion and Dilation.

(1) Erosion

The erosion operation needs two input data. The first input is the image which is to be eroded, and the second input is a set of coordinate points known as the kernel. The kernel will determine the precise effect of erosion. The center of the kernel would slide through the input image. For each pixel in input image superimpose the origin of the kernel, if the kernel is completely contained by input image the pixel is retained, else deleted. After erosion, what happens is that white pixels near the boundary would be discarded. This method is very useful for removing small noise in the background.

(2) Dilation

The input data for the dilation operation are the same as the erosion. The difference is the operation itself. For each pixel in the input image that has a value of 1, superimpose the kernel, with the center of the kernel aligned with the corresponding pixel in the input image. After dilation, what happens is that black pixels near the boundary would be discarded. This method is very useful for joining broken parts of an object or removing small noise in the foreground.

Implement Erosion and Dilation with a different order, we can realize Opening and Closing operation.

(1) Opening

In morphology, opening is the dilation of the erosion of an input image by a kernel. Opening can remove small objects.

(2) Closing

In morphology, closing is the erosion of the dilation of an input image by a kernel. Closing can remove small holes.

III. TECHNICAL APPROACH

A. Data Labeling

In this project, 200 given training images are given. I collaborated with other classmates to label them. (The full name list is in the Acknowledgment part.) For each image data, we circle the areas we think are the following 6 colors, including stop sign red(sred), other red(ored), orange(orng), blue(blue), brown(brwn), and other color(othr). With the roipoly module, we can easily label training images for each color class to obtain a training data set. For each image, the labeling results will be a ".npz" file which includes 6 ".npy"

files. The ".npy" file contains a n by 3 array, where n is the number of labeled pixels and each pixel is represented in RGB color space.

B. Image Segmentation

With the labeled data, I choose the single Gaussian model to approach this problem. I have tried the method below in both RGB and YCbCr color space. The steps for both color spaces are the same. The only difference is the way to describe colors in R^3 vector spaces. I used 120 images to train the Gaussian model.

Given an input image, the Image Segmentation process will follow the steps below:

- (1) Use labeled data from the training set to compute the mean value μ_y and covariance matrix Σ_y for each class, where $y \in \{sred, ored, orng, blue, brwn, othr\}$. These are the parameters of the single Gaussian classifier.
- (2) Use labeled data from the training set to compute the prior probability.
- (3) Convert the input image $I \in R^{H \times W \times 3}$ (H and W are height and width of input image respectively) to $X \in R^{N \times 3}$ ($N = H \times W$, size of the input image). Each row in X represents a pixel in the input image.
- (4) Put X into the Gaussian model and get the probability for each pixel. For example, if I put X in the the Gaussian model with parameters μ_{blue} and Σ_{blue} , then I will get a vector which represents $P(X|blue)$, the conditional probabilities for each pixel in the input image I in blue color Gaussian model. Repeat the procedure for all y .
- (5) For each pixel of input image, determine which class it belong to by the model:

$$P(x|y; \mu, \Sigma) = \frac{\exp(-\frac{1}{2}(x - \mu)^T \Sigma^{-1}(x - \mu))}{\sqrt{(2\pi)^d |\Sigma|}}$$

- (5) For each pixel of input image, determine which class it belong to by the model:

$$h(x) = \arg \max_y P(x, y) = \arg \max_y P(x|y)P(y)$$

x represents each row in X , as known as each pixel in the input image I .

- (6) If $h(x)$ is "sred (stop sign red)", then mark the pixel as white. If not, mark the pixel as black. Then we can get a mask after reshaping X to $R^{H \times W \times 3}$. Generally, this mask may contain lots of noise and include many areas which are not the stop sign shape. To determine the real stop sign area, we need to implement the morphological operation.

C. Morphological Operation

According to observation, if the stop sign area in the input image is large enough, it is easier for the color segmentation. The white area shape in the mask will look like an octagon with "S", "T", "O", "P" holes, just like the real stop sign. On the other hand, if the stop sign area in the input image is small, the white area in the mask usually be split into two parts: an upper part, and a lower part. This is a difficult situation for the next step, finding bounding boxes. To solve this problem, I applied the morphological operation: Closing. I eroded the mask with a 10 by 10 kernel, then dilated with the same kernel. After the closing operation, the upper part and the lower part of the same stop sign in the mask will connect to each other.

Furthermore, there will be still some noises in the mask. To remove those noises. I applied the Opening morphological operation to remove them, dilating the mask with a 4 by 4 kernel, then eroding with the same kernel.

D. Finding Boundary boxes

Given a mask, I use the python package, "skimage.measure" to enumerate regions in the mask, and I design a method to distinguish whether they are stop signs. At first, I set a threshold on the aspect ratio to decide. After some trial, I found that using only one property is too roughly, so I added a threshold on another property, area proportion. However, these two hard thresholds still can not work pretty well. I believed the reason is that in different conditions, the stop sign shape of regions in the mask may be large or small, complete or incomplete, with holes or not, with rough contour or straight, etc. Thus, I design a method to score them based on how much they resemble a stop sign shape. My method depends on three indicators:

(1) Aspect ratio

For most stop signs in the input images, they face the lens. Therefore, the aspect ratio of the bounding box is approximately one. In some cases, the stop signs are not facing the lens directly. The bounding boxes will look like a rectangle. Based on these characteristics, if the ratio is between 0.5 and 2, add 2 points to the score. If the ratio is between 0.9 and 1.2, add 3 more points to the score.

(2) Area proportion

Generally, in a given mask, a stop sign should be an octagon, with "S", "T", "O", "P" holes. The ratio lies in range 0.45 to 0.85 approximately. If the area proportion is between 0.45 and 0.85, add 3 points to the score. In some cases, the ratio varies. For example, the region contour may be incomplete, or the "S", "T", "O", "P" holes become smaller after closing operation. If the stop sign is not large enough, the "S", "T", "O", "P" holes will be filled after the closing operation. In these

cases, the ratio is approximately 0.8. Thus, if the area is larger than 9000 and the area proportion lies in 0.55 to 0.75, add 3 more points to the score. If the area is smaller than 9000 and the area proportion lies in 0.75 to 0.85, add 3 more points to the score.

(3) Euler Number

Euler characteristic of the region. Computed as a number of objects (should be 1) subtracted by number of holes. The stop sign region in a given mask, there are "S", "T", "O", "P" holes. So the Euler number is negative. According to my observation, the Euler number of most regions that are not the stop sign region is 1: there is no hole in the region. Because the Euler number is more decisive, if the Euler number is negative, add 5 points to the score.

If the final score of a region is larger than 10, I judge it as a stop sign. Use the function `regionprops.bbox` to get the output coordinate. If the final score of a region is not larger than 10, I discard it.

IV. RESULTS

After the experiments, I can successfully detect many stop signs in the validation data set. I also found that the results in RGB are better. In the following pages, I choose 12 validation images to discuss the result in my every step to see what's the differences before and after my procedures, including the color segmentation of the Gaussian model, the mask after the closing operation, the mask after the open operation, and the bounding boxes.

ACKNOWLEDGMENT

In the labeling part, I collaborated with a group to finishing labeling. Thanks to the members in the group for assuring the quality of label data and designing the format of label data for easy reading in python. The name list of my group is below. (List in order of PID)

- A53263730, Jui-Te Lin
- A53295319, Yu-Tsun Yang
- A53295675, Yunhsiu Wu
- A53298402, Ya-Hsiu Hsieh
- A53306080, Yu-Hao Liu
- A53306916, Minhsueh Cheng
- A53308491, David Lu
- A53314199, Chun-Nien Chan
- A53316963, Chun-Yen Liou
- A53317226, Sheng-Wei Chang

REFERENCES

- [1] openCV-Python Tutorials, <https://opencv-python-tutroals.readthedocs.io/en/latest/index.html>
- [2] Eroding and Dilating, https://docs.opencv.org/2.4/doc/tutorials/imgproc/erosion_dilatation/erosion_dilatation.html
- [3] scikit-image, <https://scikit-image.org/docs/dev/api/skimage.measure.html>

Table 1: Results in RGB Color Space (Part 1)







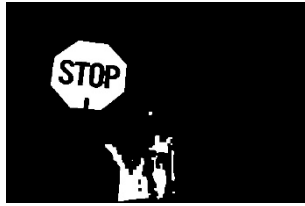








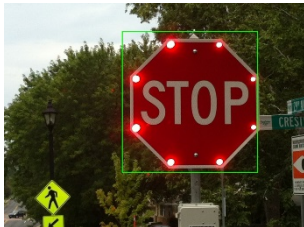
RGB Color Space			
Color Segmentation	Closing Operation	Opening Operation	Bounding Box
			
This is a relatively easy test case. The color segmentation works well after the Gaussian model, and the closing operation removes the small holes in the region. The bounding box can also work well.			
			
This test case includes other areas contain similar colors, so the color segmentation cannot work perfectly. After the morphological operations, my scoring method can find the bounding box.			
			
This lighting is darker in this case. Thus, the color segmentation result is not perfect. However, the closing operation successfully connects the cluster, and my scoring method can find the bounding box.			
			
There are halos around the LED light. They are segmented as the stop sign red also. In this case, the closing operation causes a bad result. It connects the regions that should not be together.			

Table 2: Results in RGB Color Space (Part 2)

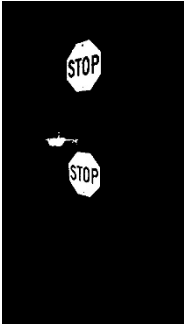
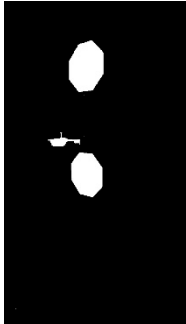


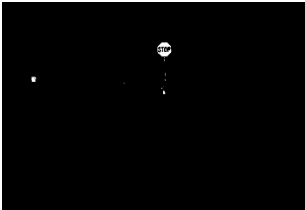
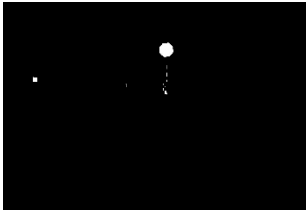
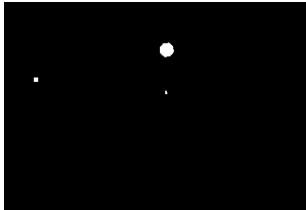



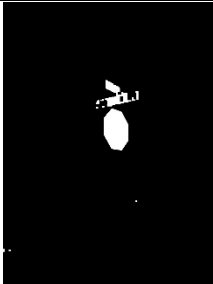





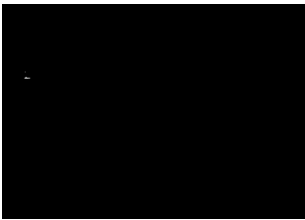
RGB Color Space			
Color Segmentation	Closing Operation	Opening Operation	Bounding Box
			
Although the stop signs are not facing directly to the lens, they still can be detected.			
			
The opening operation can remove many noises.			
			
In this case, the Gaussian model works well. The opening operation remove many noises. Though the closing operations connect the cluster of green signs, my scoring method can detect the stop sign well.			
			
The segmentation takes the stop sign apart. This situation is common when the stop sign is smaller. The closing operation connects the two part of the stop sign.			

Table 3: Results in RGB Color Space (Part 3)

RGB Color Space			
Opening Operation	Bounding Box	Opening Operation	Bounding Box
			
The smaller stop signs can be detected well.			
			
I fails in this test case. The Gaussian model cannot segment the stop sign red regions well. The mask also contains many other parts. No stop signs are detected, and even a red light viewed as a stop sign.			
			
My model fails in this test case. The Gaussian model cannot segment the stop sign red regions. I think the reason is that the color of the stop signs is not similar to other stop signs. The probability in single Gaussian model is not high enough. Then the Bayes rule classifies them as other colors.			
			
My model fails in this test case. The Gaussian model cannot segment the stop sign red regions. I think the reason is that my model classifies the color of the stop sign region as “other red” or “orange”.			