ECE 276A
Project 1

Color Segmentation

Due Date: Oct 25, 2017

Professor: Nikolay Atanasov

Chuqiao Song

A53239614

Introduction

Computer vision is a discipline science, and it includes picture recognition, identification, verification and other analysis. It teaches computer to understand picture, which is a normal ability for human being, however it is hard for computer to gain that skill. There are lots of ways for computer to identify picture, and to some extent, the color segmentation is a way to do this.

In this color segmentation project, the first step is to classify the picture to four segments, which is achieved by trained Gaussian distribution. Then by using some algorithm, it can be easy to find the red barrel segment. For the distance of the test graph, linear regression method is used in this projected, which is trained by test data (X=red area of barrel, Y = known distance of each graph). Finally, to find the bounding box of the red barrel and showing it on graph, using the function minArearect with found contour is a way to do this.

Problem Formulation

1) using labeled training set to get Gaussian model

X is the pixel with RGB value of each color segmentation

Mean of one segment, 3*1 vetcor

$\mu = \frac{1}{N}\sum_{i=1}^{N} X_i$, where N is number of samples in that segment

Covariance of one segment, 3*3 matrix

$$\mathcal{E} = \frac{1}{N-1}\sum_{i=1}^{N}(X_i - \mu)(X_i - \mu)^T$$

2)identify the probability of each pixel of test image in each color segments

$$p(x) = \frac{1}{(2\pi)^{\frac{n}{2}}|\Sigma|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

P(class|X)=P(X|class)*P(class)

Class = argmax(class) [P(class|X)]

3) training the linear regression model to find the distance by using test data.

$\begin{matrix} a \\ b \end{matrix} = (X^T X)^{-1} X^T y$, where X is m*2 matrix, and the first column of it is area of red barrel, and second column is all one column. Y is m*1 distance vector given by project.

y = ax + b the formula to find the distance given test image, where x is the area of red barrel in test image

Technical Approach

Color segmentation

Step one: Get training data. Find the training data for each color segmentation, and this step is realized by labeling each picture 4 times by using given function roipoly, and carefully label the red barrel. After doing this, I get YCBCR training data for red barrel, not so red, brown, and yellow class. **Step two:** get mean and covariance value. Using these four training sets to train the 3D Gaussian model, such that find the mean and covariance value for each class. **Step three:** classify each pixel of the test image. By using the trained Gaussian model to identify whether the pixel is belonging to red, not so red, brown, or yellow class. Meanwhile, create a matrix that has the same size of the test image, and assign associated position of classified pixel a specific value. To simplify, if the pixel is classified as red, I assign the matrix at associate position RGB [255, 0, 0], and other not barrel red pixel I set to white color RGB [255, 255, 255]. Finally, I get the final segmented graph from the matrix.

Barrel detection

By using the segmented graph, I use the findcontour() function to find the contour of the red segment, and in this case this function will return the contour (array of position) for each red segment. After doing this, I should filter irrelevant contours to leave the red barrel contours. To realize this, I use the contourArea() function to find the area for each contour, and by find the index of maximum area, I can associate its index to contour array to find the red barrel contour. Up to now, I have found the specific value for red barrel, and then I use minAreaRect() to find the bounding box for the red barrel such that I can achieve the barrel detection.

Result

After I convert the original image to 1200*900 graph, I resize the graph again in my code, and shrink it 10 times which is to 120*90 for running speed. Thus, the following coordinate should be 10 times smaller than original one; however, modify the area of the bounding box 100 times, so the distance after linear regression should at same scale.

Picture	001	002	003	004	005	006	007	008	009	010
Distance	5.31	6.38	1.63	0.668	6.14	- 16.33	6.4	3.56	5.95	6.46

ImageNo = [01], BottomLeftX = 56, BottomLeftY =53, TopRightX = 67, TopRightY = 36

ImageNo = [02], BottomLeftX = 57, BottomLeftY = 54, TopRightX = 64, TopRightY = 45

ImageNo = [03], BottomLeftX = 63, BottomLeftY = 54, TopRightX = 80, TopRightY = 8

ImageNo = [04], BottomLeftX = 63, BottomLeftY = 64, TopRightX = 63, TopRightY = 24

ImageNo = [05], BottomLeftX = 61, BottomLeftY =52, TopRightX = 68, TopRightY = 41

ImageNo = [06], BottomLeftX = 51, BottomLeftY 79, TopRightX = 85, TopRightY = 1

ImageNo = [07], BottomLeftX = 60, BottomLeftY 19, TopRightX = 66, TopRightY = 11

ImageNo = [08], BottomLeftX = 52, BottomLeftY 40, TopRightX = 71, TopRightY = 32

ImageNo = [09], BottomLeftX = 65, BottomLeftY 71, TopRightX = 69, TopRightY = 56

ImageNo = [10], BottomLeftX = 62, BottomLeftY 31, TopRightX = 68, TopRightY = 23

Discussion

The failed test graph is figure 3, 6, 8. In the figure3, my algorithm cannot identify whether it is red barrel or vending machine, because the color of them are nearly same. To some extent, it is because my training set are too smaller, and my Gaussian model it not much precise to identify the red of barrel, and the red vending machine. The failed test for image 6 is same as above, since the non-precise Gaussian model. Also, the reason of the failed test for image 8 is my non-precise bounding box algorithm. I can only find the maximum red region to draw the contour on it. Maybe, the way to modify this, I can compare these three red regions, and if these areas are similar, I can find a way to draw the contour to include all three red regions.

Covariance and mean

```
cov_red = [[ 983.45516294, -20.82693649, 251.59740008], [-20.82693649, 18.27097824, -39.23327015],
           [ 251.59740008, -39.23327015, 458.49355809]]
mean_red = [[60.21777405206787], [121.75193536269101], [187.3117150894875]]

cov_nsred = [[ 1995.54352586, -145.25809121, -111.5877214 ], [ -145.25809121, 83.23549837, -117.67712958],
             [ -111.5877214, -117.67712958, 493.246937  ]]
mean_nsred = [[94.15792807168283], [115.255097581932], [172.50993494537866]]

cov_brown = [[ 1714.41542918, -125.00920446, 382.1762053 ], [ -125.00920446, 83.38935013, -121.41269209],
             [  382.1762053, -121.41269209, 269.16857731]]
mean_brown = [[75.56044612666769], [114.84189721740381], [153.4726336741301]]

cov_yellow = [[ 1907.97039881, -462.80228832, 144.84046221], [ -462.80228832, 249.39126596, -114.30270918],
              [  144.84046221, -114.30270918, 105.07373941]]
mean_yellow = [[120.61519891291778], [96.64161703480976], [153.83091679195095]]
```

linear regression

```
param = [-9.43267079e-05, 6.83717067e+00]
```

first number is a, second is b, y = ax+b

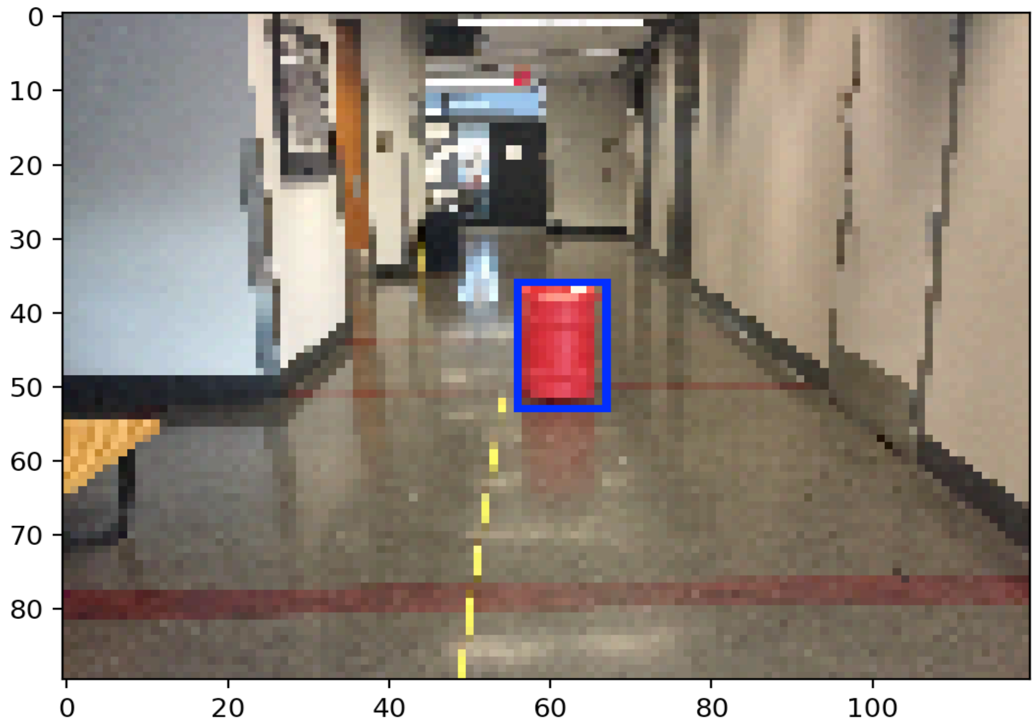

figure1, from bottom left clockwise coordinates [56 53], [56 36], [67 36], [67 53]

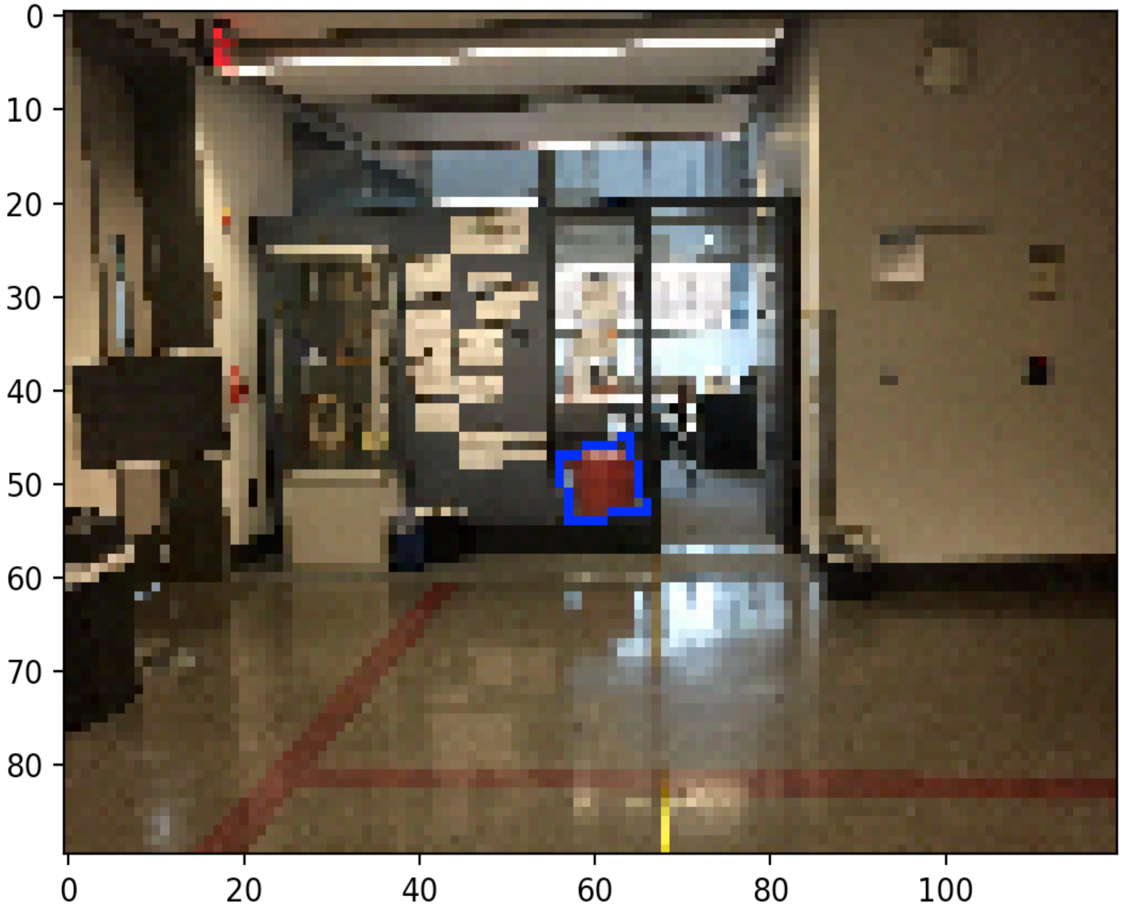

figure2, from bottom left clockwise coordinates [57 54], [56 47], [64 45], [66 53]

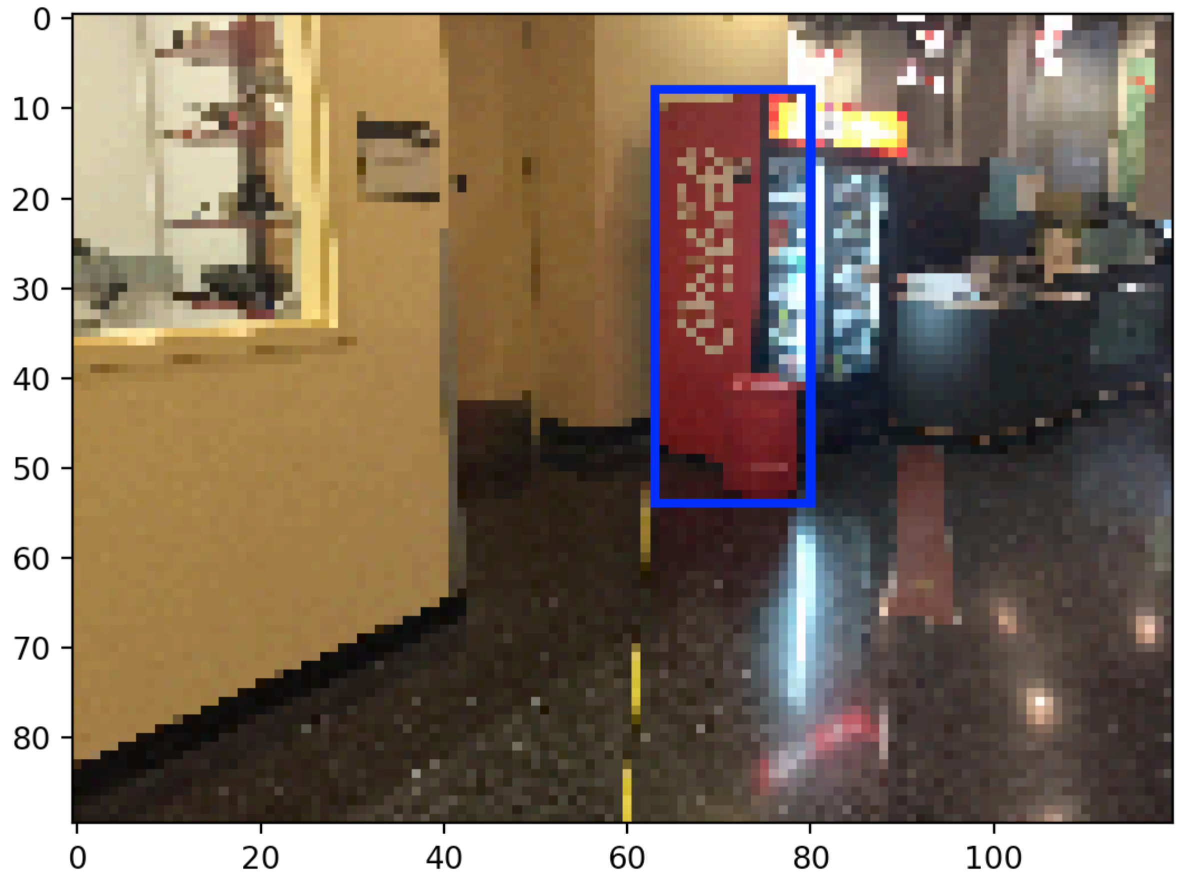

figure3, from bottom left clockwise coordinates [63 54], [63 8], [80 8], [80 54]

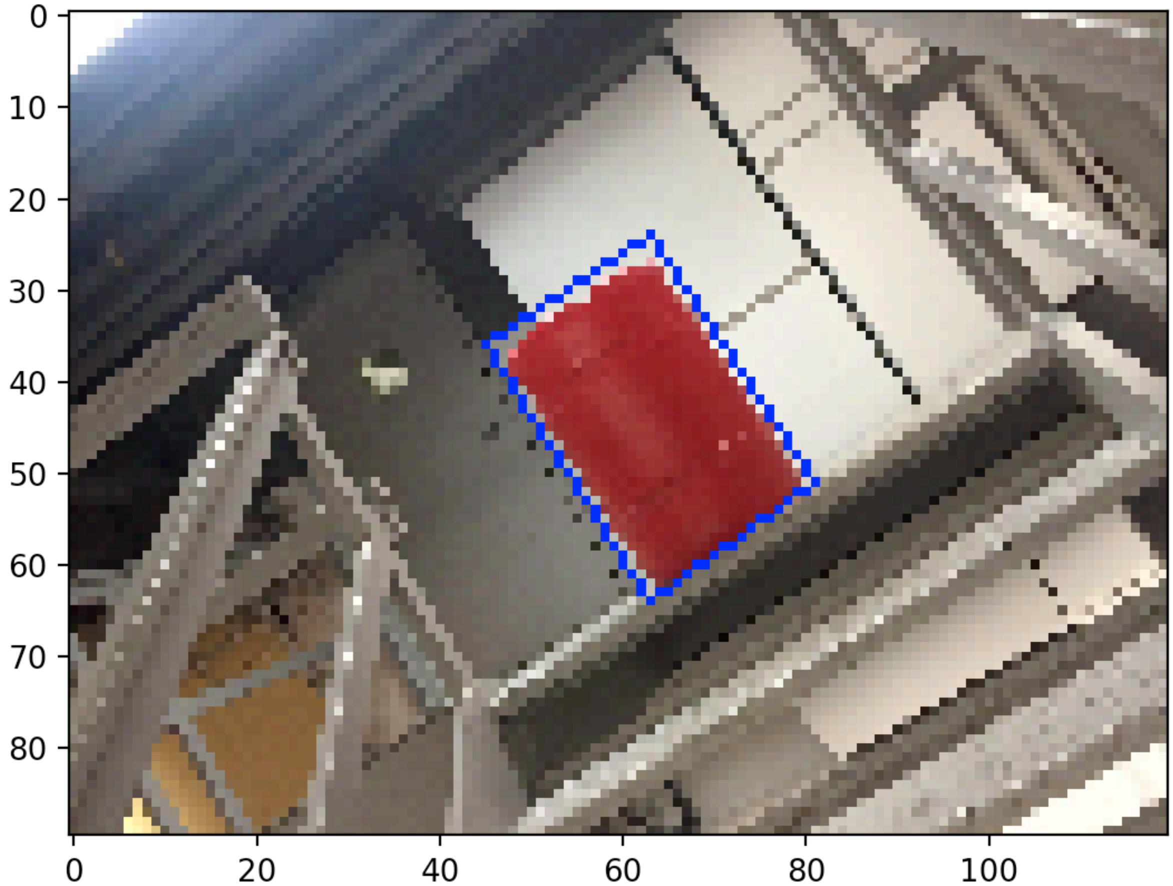

figure4, from bottom left clockwise coordinates [63 64], [45 36], [63 24], [81 51]

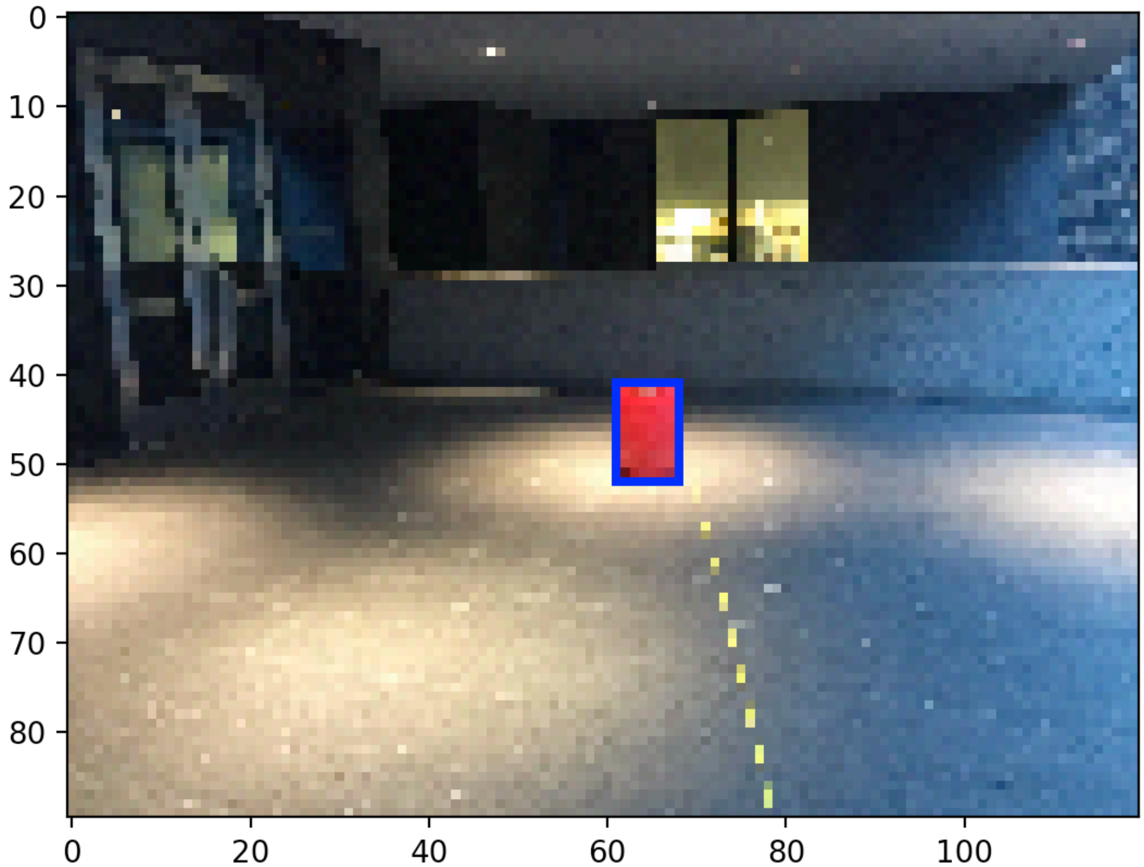

figure5, from bottom left clockwise coordinates [61 52], [61 41], [68 41], [68 52]

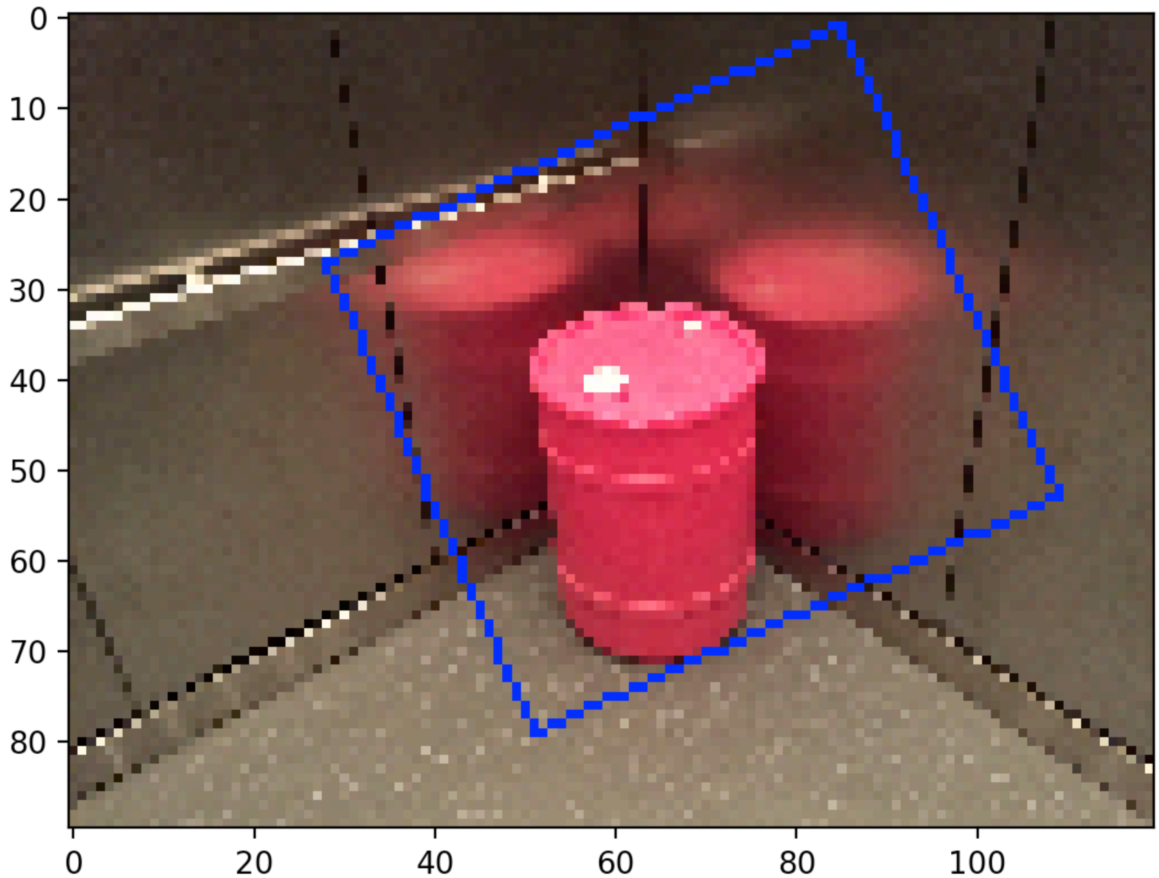

figure6, from bottom left clockwise coordinates [51 79], [28 27], [85   1], [109 53]

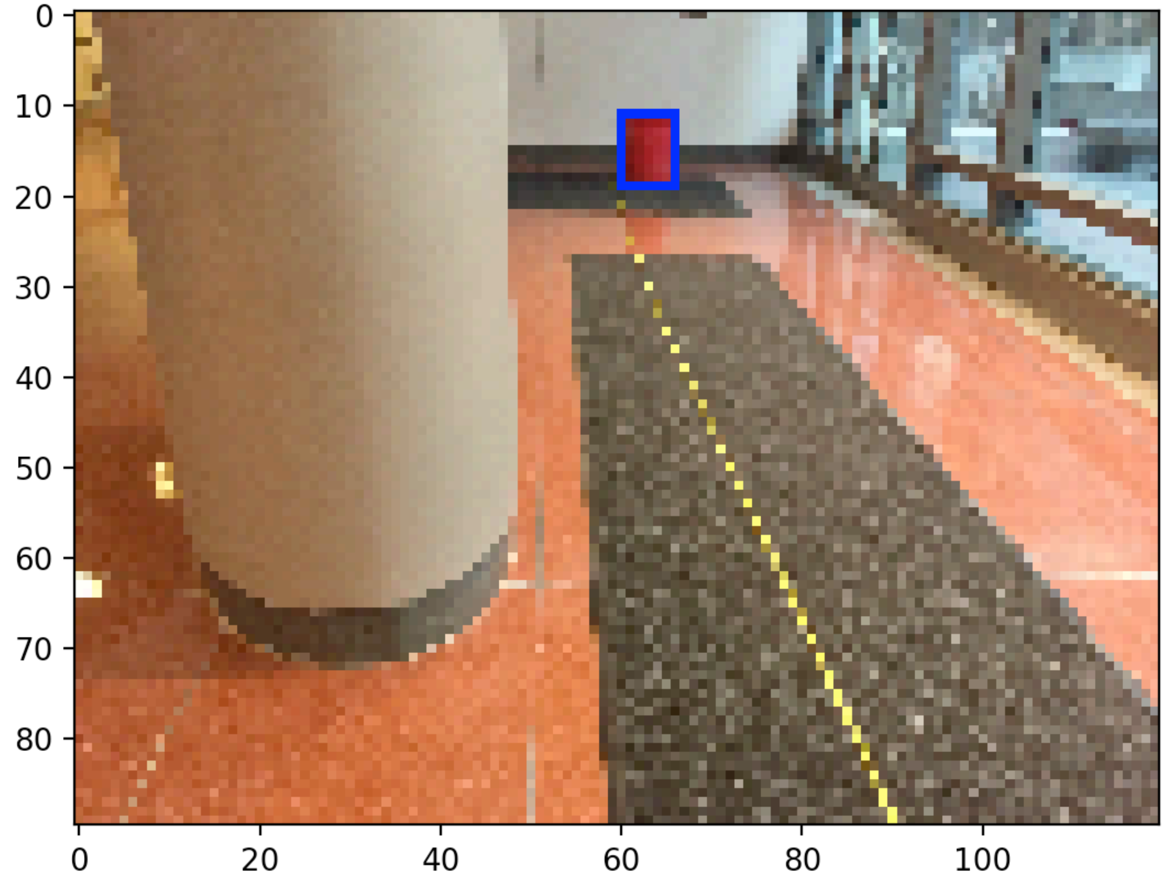

figure7, from bottom left clockwise coordinates [60 19], [60 11], [66 11], [66 19]

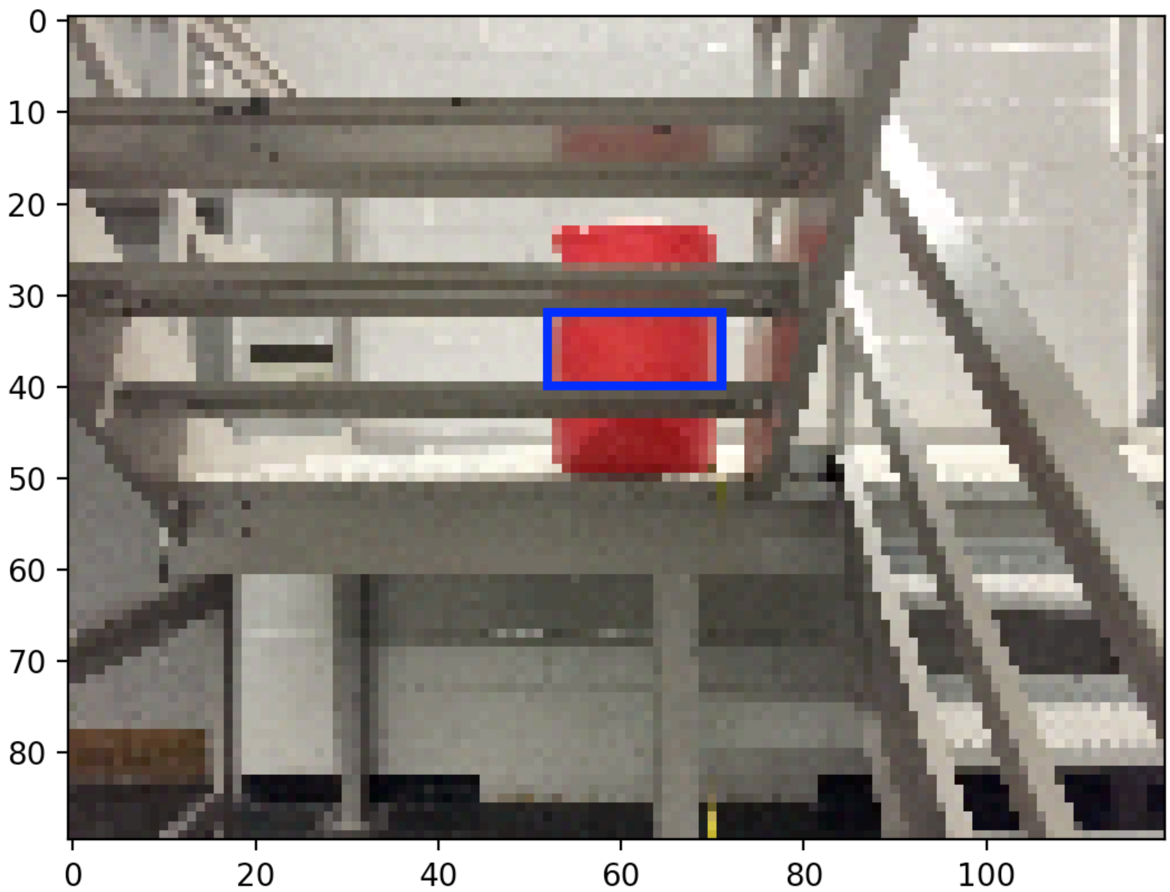

figure8, from bottom left clockwise coordinates [52 40], [52 32], [71 32] [71 40]

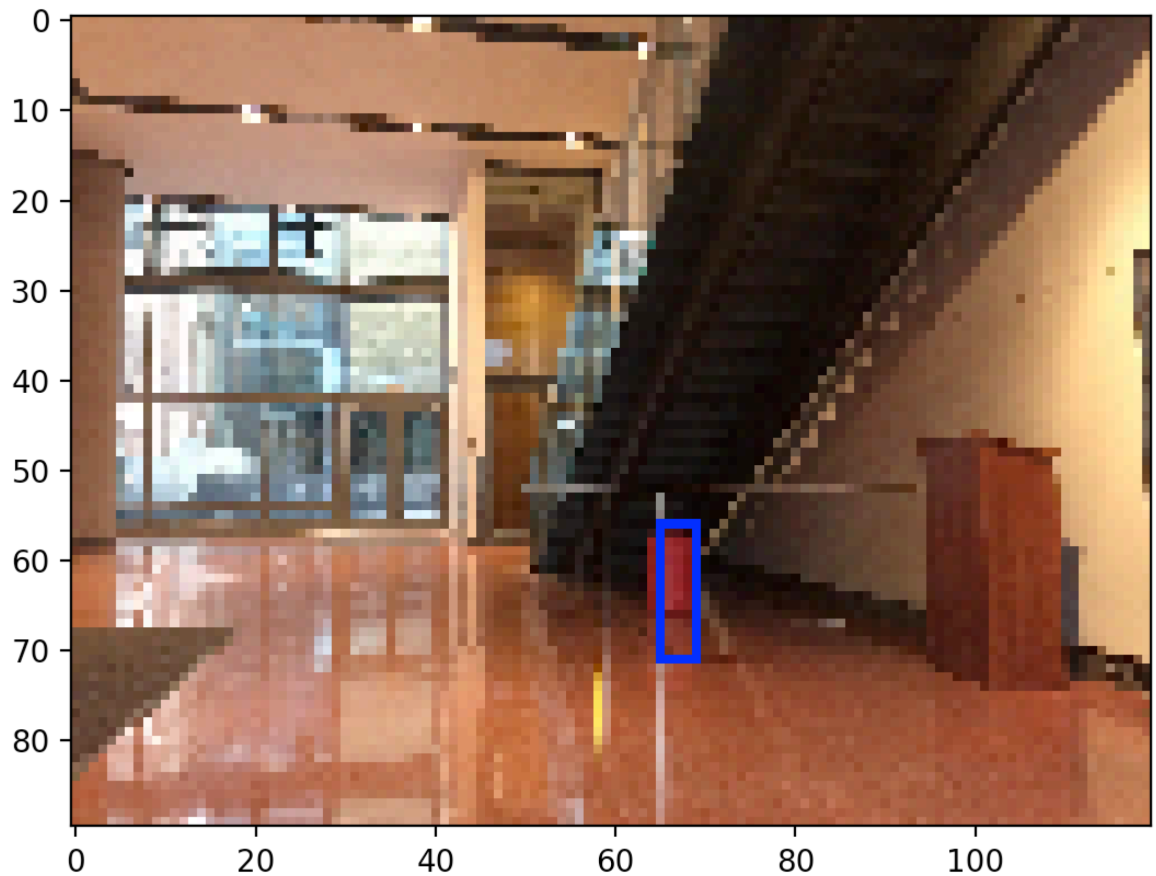

figure9, from bottom left clockwise coordinates [65 71], [65 56], [69 56], [69 71]

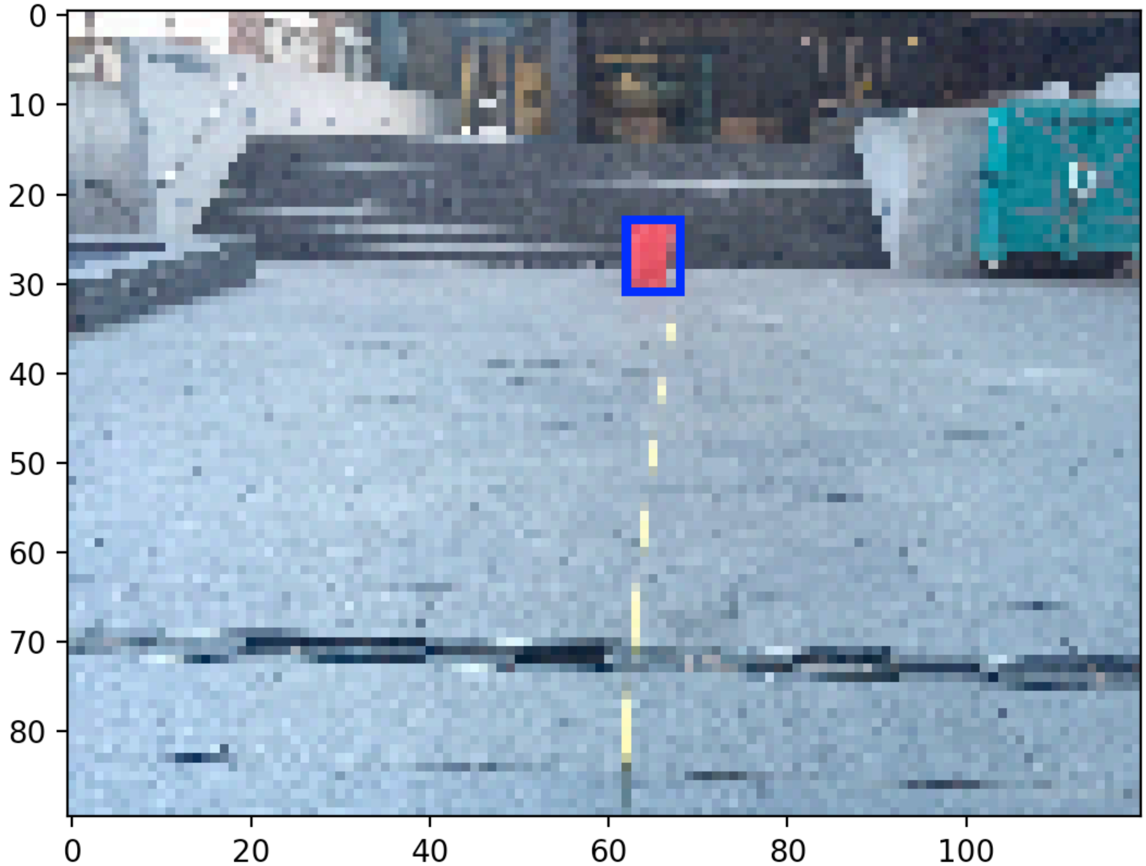

figure10, from bottom left clockwise coordinates [62 31], [62 23], [68 23], [68 31]