CS44053 Assignment2

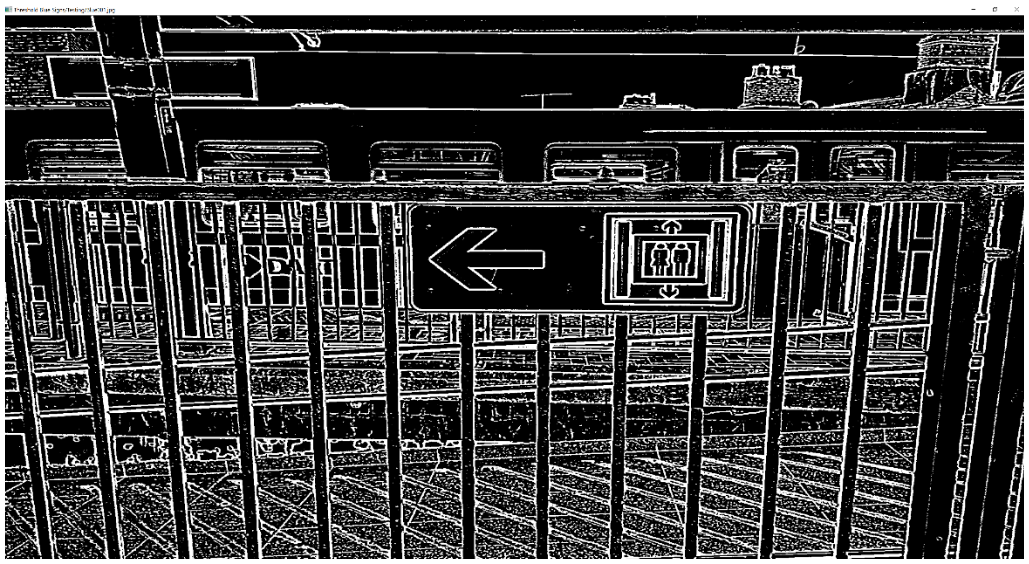
Introduction:

For this particular task of blue sign location and recognition, I was able to achieve a best F1 score of 0.887 with precision of 0.957 and recall of 0.827. The model that I have implemented is very simple and inefficient. I have tried to implement more sophisticated models but was not able to successfully implement them, which I will talk about in the end of this report. The program is divided into two main parts, segmentation and recognition. For segmentation, the original image is first converted to grayscale image. Then a gaussian filter is applied to the grayscale image. The resulting image is converted to a binary image using adaptive thresholding. The binary image is used to find all of the contours in the image. For each of the contours found. Contours are coarsely filtered to give a small set of possible rectangular bounding boxes which may contain a sign object. The possible set of boxes are used as input to the recognition stage. The recognition stage uses template matching to determine which boxes contain blue signs.

Procedures:

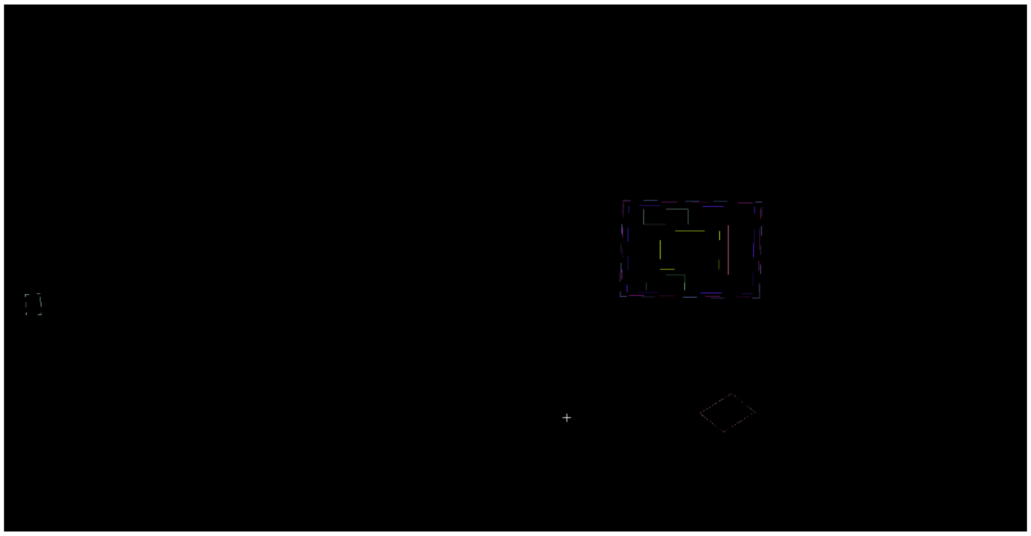
Segmentation:

The original image is converted to a grayscale image so that I only need to work with pixel intensity values and not care about the color of the image. This step improves thresholding in the later steps by ignoring color factor. This simplifies the model. Gaussian blur is performed on the grayscale image with filter size 31 since the images are relatively big and sigma 1. This in effect blurs the image to reduced gaussian noise which helps with edge detection and thresholding later in the segmentation stage. The smoothed image is converted to a binary image using adaptive thresholding with filter size 11.



Adaptive thresholding is used as it works with images with strong illumination gradients since it takes into account the intensity value of neighboring pixels to determine the threshold dynamically. Binary images created this way does not contain patches of area with strong lighting which gives better segmentations of regions. From the resulting binary image. I am able to distinguish regions of the images clearly. Lots of white spots and small holes are present from the image. This could have been dealt with by using opening and closing after. Opening is erosion followed by dilation which should remove noise from the image, closing is dilation followed by erosion which should remove holes in the image. However, I found in practice, doing such operations did not improve the result much. The noise and holes are not effectively removed with small filter sizes while bigger filter sizes resulted in lost of information especially the joining of boundaries of closely placed signs. The binary image alone is sufficient enough have good performance on segmenting out regions.

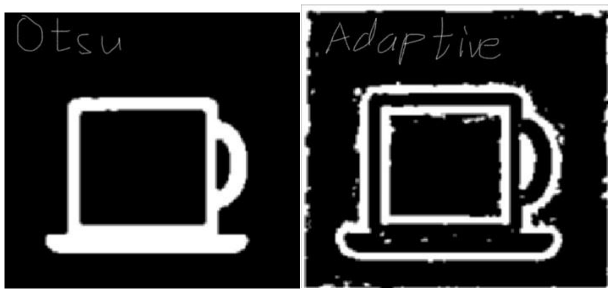
The resulting binary image is used to find all of the contours in the image. This is effectively found by following the borders of the binary image. Two pixels are connected if they are 8 connected neighbors and the second pixel must be of 3 neighbors in the direction of the contour. Contour segmentation resulted in all of the contours in the binary image. The polygonal segments of the contours are then found by using 10% of arc length and contours must be closed to find approximate shape segmentation regions created by the contours. This removes contours which does not form polygonal shapes which are definitely not contours of a sign object. The remaining polygonal segments are further filtered by only selecting those that forms a rectangular shape such that only segments formed by 4 vertices are considered for next step. The final boxes used for recognition also does not include boxes which are elongated or extremely small. This is done by excluding those that with large width and length ratio, and the area of the minimum bounding rectangle is less a threshold. Below is the boxes of the areas possibly containing the sign object.



The minimum rotated bounding rectangle of the remaining polygonal segments are passed on for recognition. By deducing the minimum rotated bounding rectangle, we can extract the object easily in the recognition step. The reason that I included boxes within one another at this point is to prevent case where boxes of a sign is within a very large box, and to increase performance of recognition by taking making multiple tries with different sized regions within a sign object.

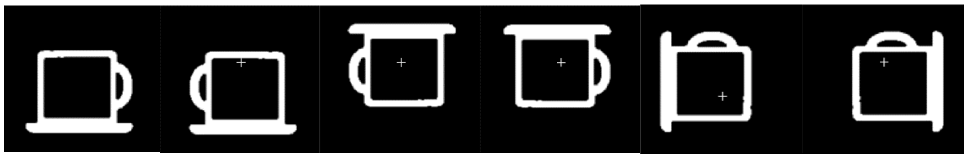
Recognition stage:

The recognition stage involves simple template matching steps. The training object images are processed with grayscale, gaussian blur, Otsu thresholding. This is again to make the model simple by considering only intensity values of the pixel. The resulting binary image also makes template matching perform better by limiting pixel values to 0 and 1. Otsu thresholding can effectively distinguish regions of the sign since there is a strong gradient between the blue background and the white sign drawing. Comparing to adaptive thresholding which produces more noise and holes which makes recognition more difficult, Otsu gives clearly distinguishable regions without much noise.



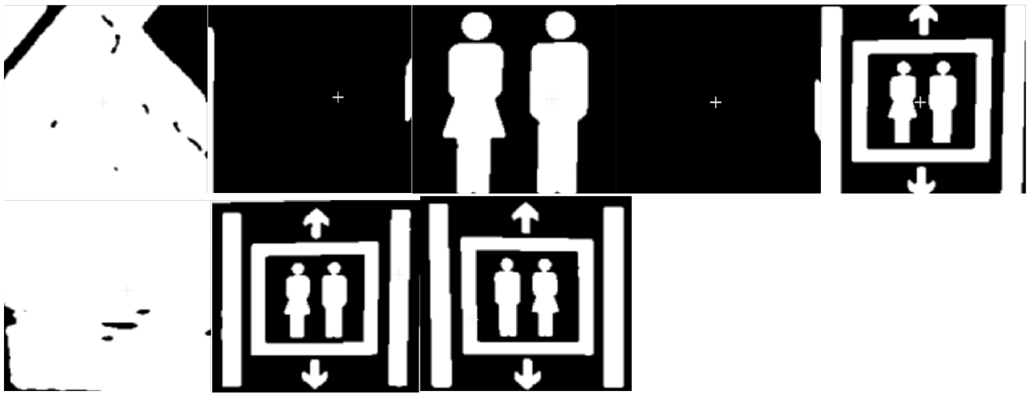
The training images are resized to 200 by 200 and the borders of the image is copped out to disregard the white box surrounding the sign body. This improves performance by decreasing the complexity of features that need to be analyzed. Some test images may have distorted white borders. By disregarding the borders, we can recognize some images by only looking at the actual body of the sign. The training images are also flipped in all directions and rotated by 90, 180, 270 degrees to try to create multiple copies of the training image with different orientations. This improved performance since some images are contains signs with different orientations of the training images.

Training images are like these:



The rotated bounding rectangles are used so that we can crop out a more or less up right object. For each of the rotated bounding rectangle, the bounding area in the original image is cropped out by calculating the rotational angle of the box and rotating the original image by that angle to make the rotated rectangle up right. Each cropped object are passes through the same processing pipeline as the training images.

Testing cropped images from image001 the last two was matched by template matching:



Each of the testing crops are template matched with each of the training images in one single step of template matching since the cropped image and template training image are resized to the same dimension. Template matching was carried out with normalized squared difference. This limited result between values 0 and 1 which is easier to find a good performing threshold. The template matching algorithm calculates in terms of the binary image, effectively the percentage of pixels that have the same value as binary images only have 1 and 0. Through testing, a threshold of 0.5 was to give optimal performance on the result. For each of the test object images that have been matched with a class, most overlapping recognitions are filtered out by ignoring those that are within another recognized object.

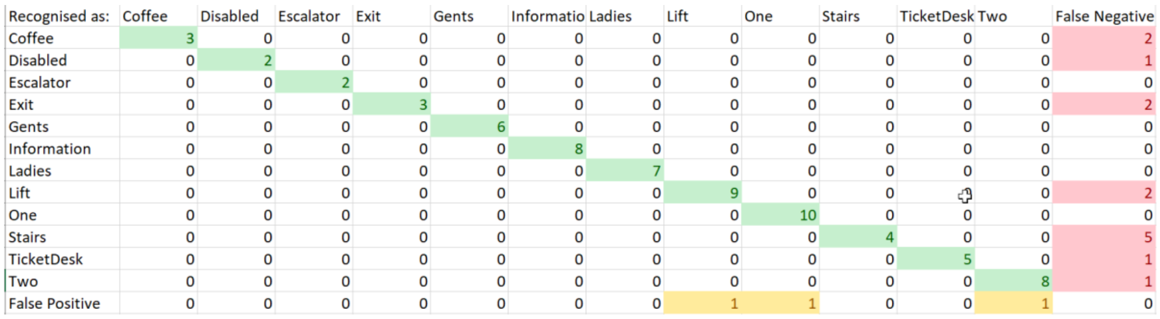
Results:

The results are:

Precision = 0.957143

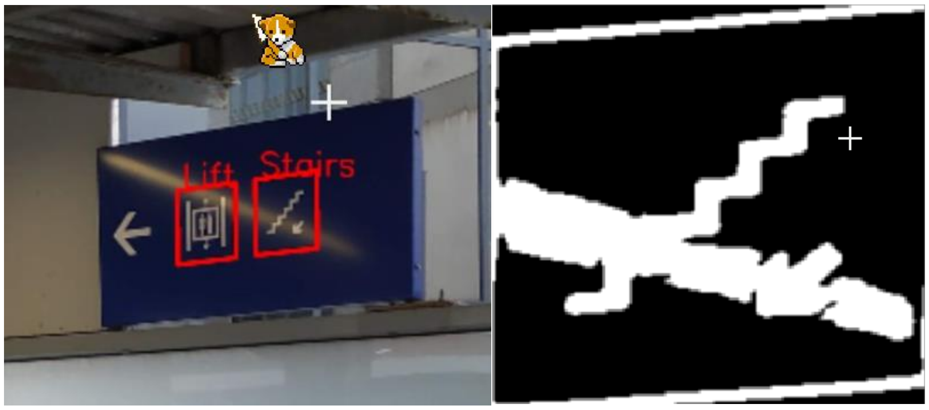
Recall = 0.82716

F1 = 0.887417

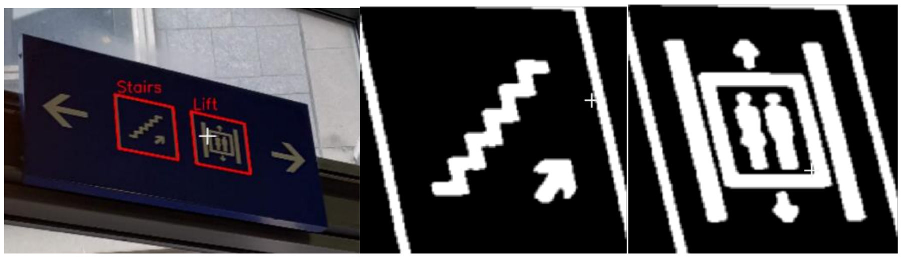


Recognised as:	Coffee	Disabled	Escalator	Exit	Gents	Informatio	Ladies	Lift	One	Stairs	TicketDesk	Two	False Negative
Coffee	3	0	0	0	0	0	0	0	0	0	0	0	2
Disabled	0	2	0	0	0	0	0	0	0	0	0	0	1
Escalator	0	0	2	0	0	0	0	0	0	0	0	0	0
Exit	0	0	0	3	0	0	0	0	0	0	0	0	2
Gents	0	0	0	0	6	0	0	0	0	0	0	0	0
Information	0	0	0	0	0	8	0	0	0	0	0	0	0
Ladies	0	0	0	0	0	0	7	0	0	0	0	0	0
Lift	0	0	0	0	0	0	0	9	0	0	[illegible]	0	2
One	0	0	0	0	0	0	0	0	10	0	0	0	0
Stairs	0	0	0	0	0	0	0	0	0	4	0	0	5
TicketDesk	0	0	0	0	0	0	0	0	0	0	5	0	1
Two	0	0	0	0	0	0	0	0	0	0	0	8	1
False Positive	0	0	0	0	0	0	0	1	1	0	0	1	0

In general, a F1 score of 0.887 is generally doing well with this simple model. Precision is really high in this model which means of those signs that was identified by the model, 95% of those were actually the sign recognized. This means the model is correctly identifying the meaning of signs with high precision. The slightly lower recall means that the model is not identifying some signs. This problem is clearly identified in some images but not so clearly in others.



In this image in particular, the signs are covered by extreme lighting, this gave a big patch in the resulting binary image after thresholding. This is very difficult to address in template matching recognition system such that it will only be able to match images with high similarity. If I try increasing threshold for matching, unrelated images will get recognized as signs.



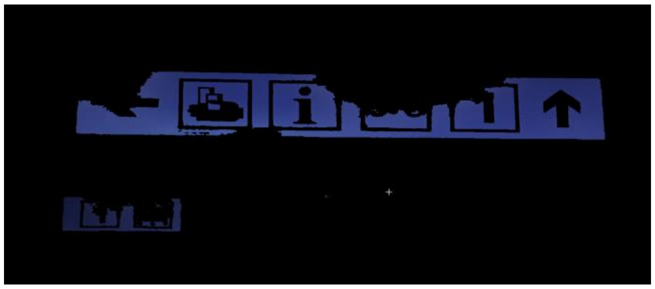
In images like this, the signs are greatly distorted by the orientation of the sign. This means the image I crop from the rectangle box will be strangely rotationally distorted. These images can't be properly template matched by this model. This might be solvable by introducing more transformations for the training or test image set to be template matched or further reduce borders to minimize the regions for template matching to reduce the effect of these distortions. However, this model is already relatively slow and inefficient, introducing these solutions will further worsen efficiency exponentially while does not provide great increase for performance.

Most of the false negative signs appear to be stairs. I believe this is because the stair signs in the training images does not contain all of the orientations of the stair signs actually used in the train station. This means that template matching sometimes cannot find a stair sign in the given training set. Also, many of the stairs signs in the test set are also in a distorted orientation as described above which lead to the model not able to identify them properly. One of the stairs signs is also affected by the strong lighting effect.

In conclusion the model implemented can be simple and mostly effective but is greatly affected by the surrounding environment.

Extras:

In the earlier tries, I have tried to use color segmentations in HSV color space which should be less influenced by lighting in the environment. I have tried to tune HSV color space to find the blue segments which should give me the regions in the image with the blue signs. However, I found this to be very difficult as if you tune saturation and value too broadly, it includes too much irrelevant regions but if tight boundaries were used, signs with dark regions are not identified or bright regions of the signs are left with a huge hole.



For recognition I have tried to use SVM with the preprocessed images like the steps in the actual model implemented as feature vectors, however I was not able to get useful classification result. I have tried to create an SVM classifier for each of the classes. With other classes being the negative example however I was not able to successful classify images. I was getting the same result from the classifier not matter what image I put in or tuning the hyperparameters. Perhaps this might be because the data is too skewed which means always guessing a single value can be a good estimate. Also using the whole image as feature vector may have also contributed to poor result, maybe using Hough Transform to find feature vectors for the signs might be a better approach.