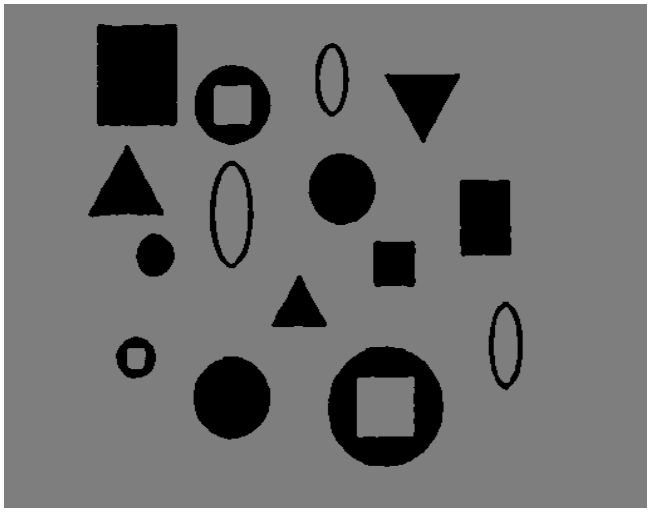
CS 650
Pattern Recognition
Jae Lee

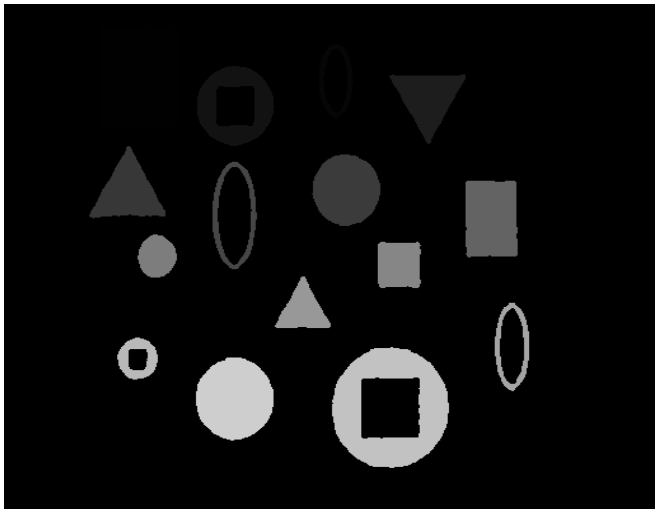
1. Isolation of individual shapes

As suggested, I implemented the connected component labeling algorithm. There are two parts in my implementation. The first part is to assign labels to foreground pixels in ascending order. New label is assigned to a foreground pixel that their left and top neighboring pixels are background pixels. When either left or top neighboring pixel has already a label assigned, the current pixel is also assigned to the label. When both left and top neighboring pixels have labels, the current pixel is assigned to a smaller label. The second part is to merge these labels so that one object is assigned to only one label. I used the map data structure to merge labels. The result turned out to be successful. First couple shapes are not shown in the connected images, but there are still there with different pixel values from background pixel. Since they have the smallest label in number, their color is just very close to black.

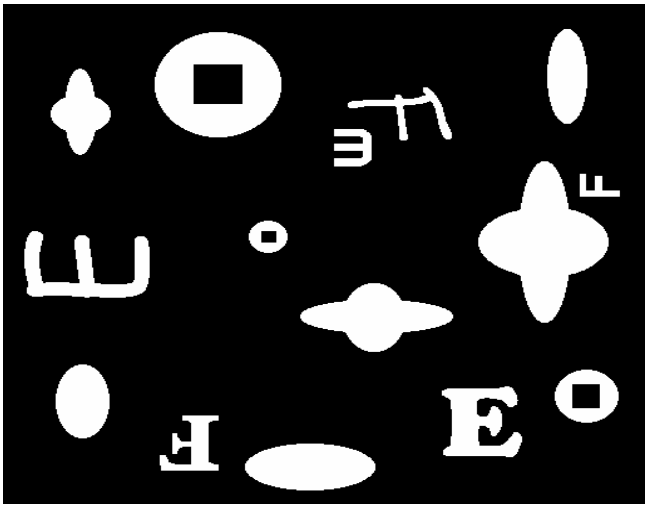
Original



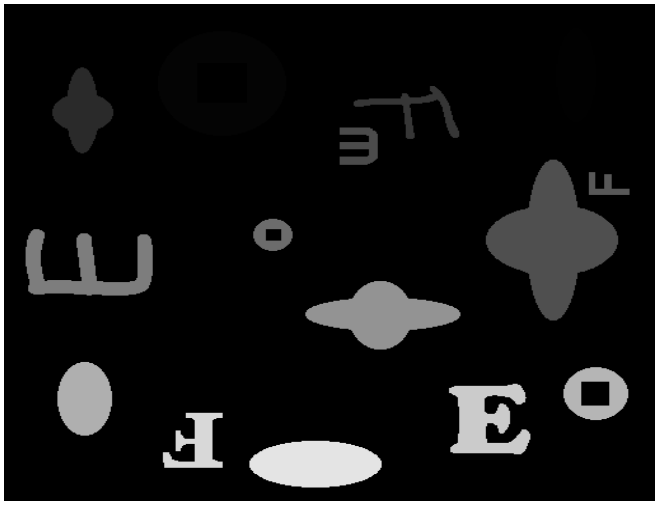
Connected



Original



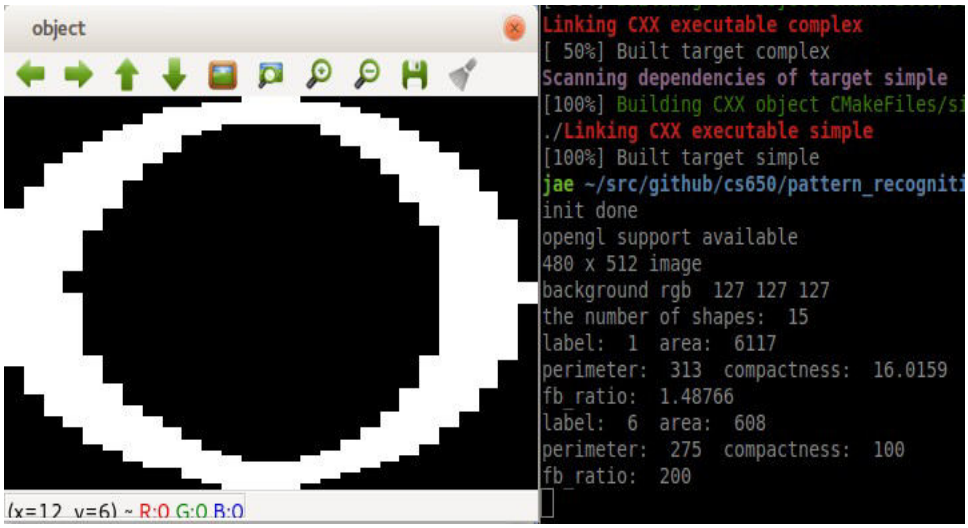
Connected

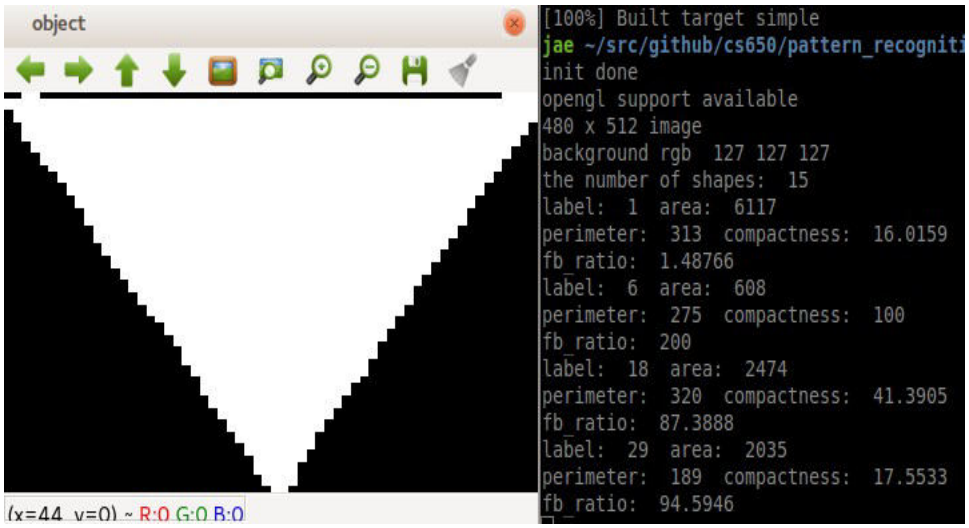


2. Generation of Features / Patterns

I tried to extract 3 features for this project: compactness, foreground and background ratio, and color crossing. The first thing I did to extract feature was to construct sub-image of each shape object. This sub-image was made so that it can keep each shape perfectly in the smallest possible frame. From these sub-images, I extracted foreground and background ratio which is calculated as # of background pixels / # of foreground pixels. This info helped me a lot to distinguish between shapes. In fact, I am using only these two features for shapes.pgm and testshapes.pgm.

Here are some samples of sub-image and corresponding features.





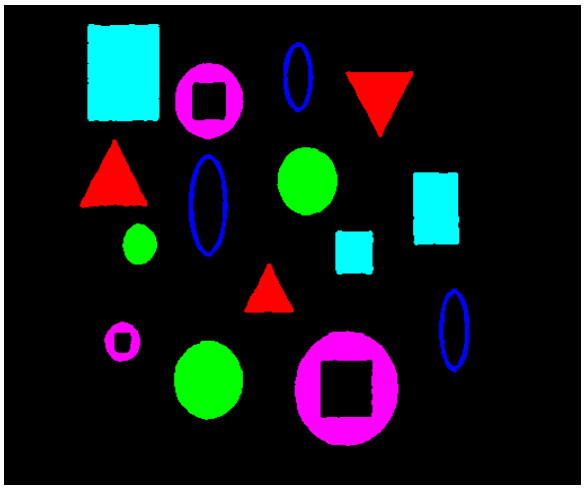
These two features worked perfectly for simple shapes as in shapes.pgm, but didn't work well for train1.pgm. So I added a feature that counts how many different foreground chunks there are on the virtual lines that bisect a sub-image vertically and horizontally. For example, 'E' shape without rotation will have 3 vertical foreground chunks, 1 horizontal chunk. I just named this feature as color crossing. The impact of this feature will be discussed further in the following section.

3,4. Generation of 5 separate pattern classes and Pattern classification

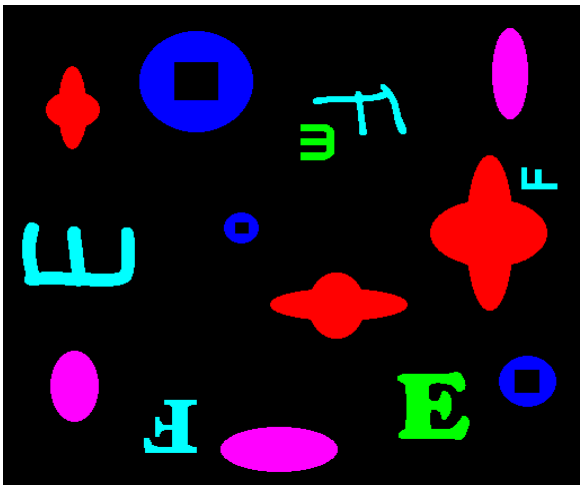
I used the K-mean clustering algorithm. In order to provide with inputs for K-mean algorithm, I created pattern/feature vectors from the features described above. One thing that I noticed was that I had to give proper weights on features. Otherwise, some features were under represented while other features were over represented resulting in having dominant effects on the result of K-mean. After I feed in thoughtfully weighted feature vectors, the outputs from K-mean are cluster numbers that each feature vector belongs to and centroids of each clusters. These centroids are used to create linear discriminants.

The following shows the result of the K-mean clustering. As shown below, color crossing helped to distinguish between 'E' and 'F', but not perfectly. On the third image, the red 'F' turned out to be 3 color crossing because the bottom part of 'F' sticks out to the center which caused one more count for color crossing. This made this 'F' to be considered as 'E'.

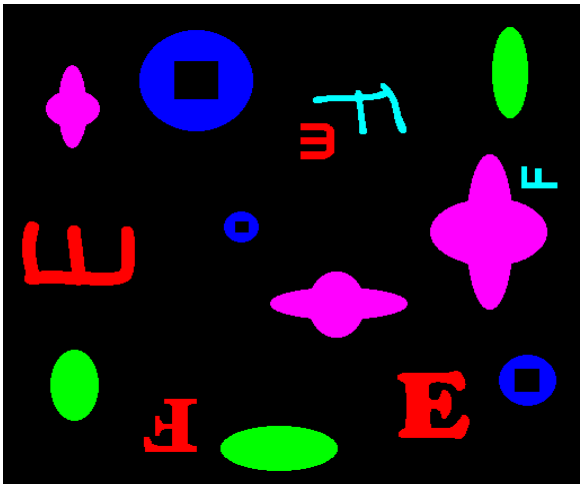
Shapes.pgm



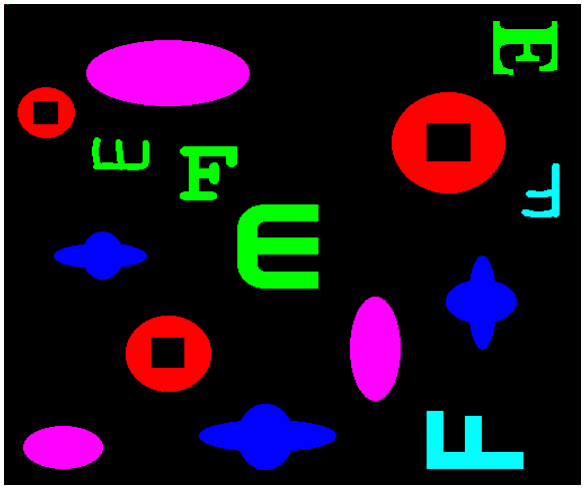
Train1.pgm without color crossing



Train1.pgm with color crossing(weight=10)



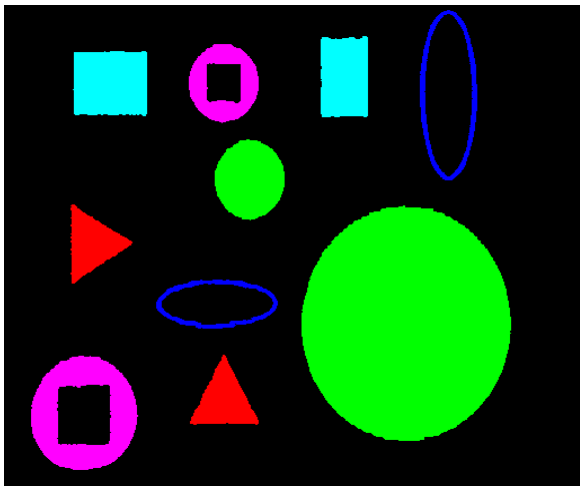
Train2.pgm with color crossing(weight=10)



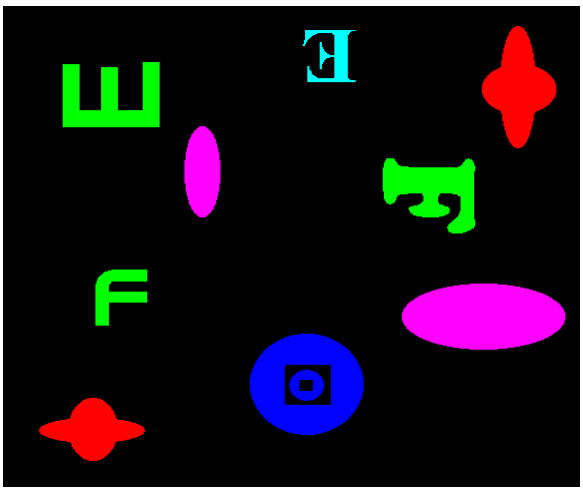
5. Testing

As mentioned earlier, linear discriminants were created for each class with the centroids from K-mean algorithm. The difference between K-mean and linear discriminant is that K-mean algorithm keeps adjusting clusters' centroids whereas linear discriminant just classifies an input vector. One thing I found out was that even if the color crossing feature helped to distinguish 'E' and 'F' in the training image, this didn't necessarily mean that it would help the linear discriminants too. I purposely chose the weight for color crossing to be 5 because with the weight being 10, it started to mess up correct clusters.
The following shows the result of my linear discriminants.

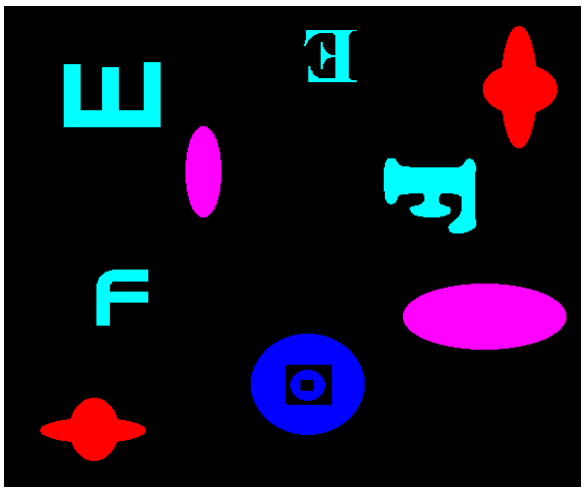
Testshapes.pgm

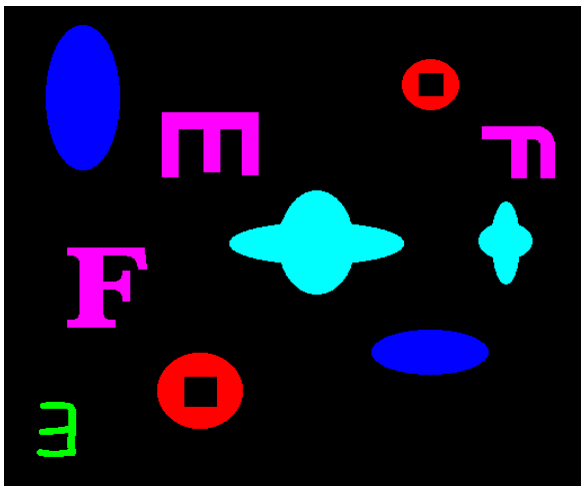
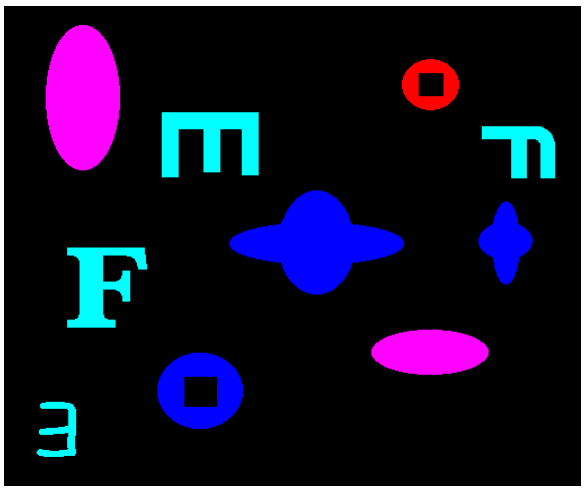


Match1.pgm without color crossing



Match1.pgm with color crossing(weight=5)



Match2.pgm without color crossing	Match2.pgm with color crossing(weight=5)
	

6. Conclusion

I learned so much by doing this project. I learned the connected component labeling algorithm from scratch and also learned to be able to extract descriptors for shapes. This project made me realize that computers can be smart with algorithms that will make them smart. Also, this project was a good one for me to make a connection between machine learning and computer vision in my mind. Even though my implementation is not 100% perfect, I am glad that I got this far. This project definitely stretched my ability and intelligence.