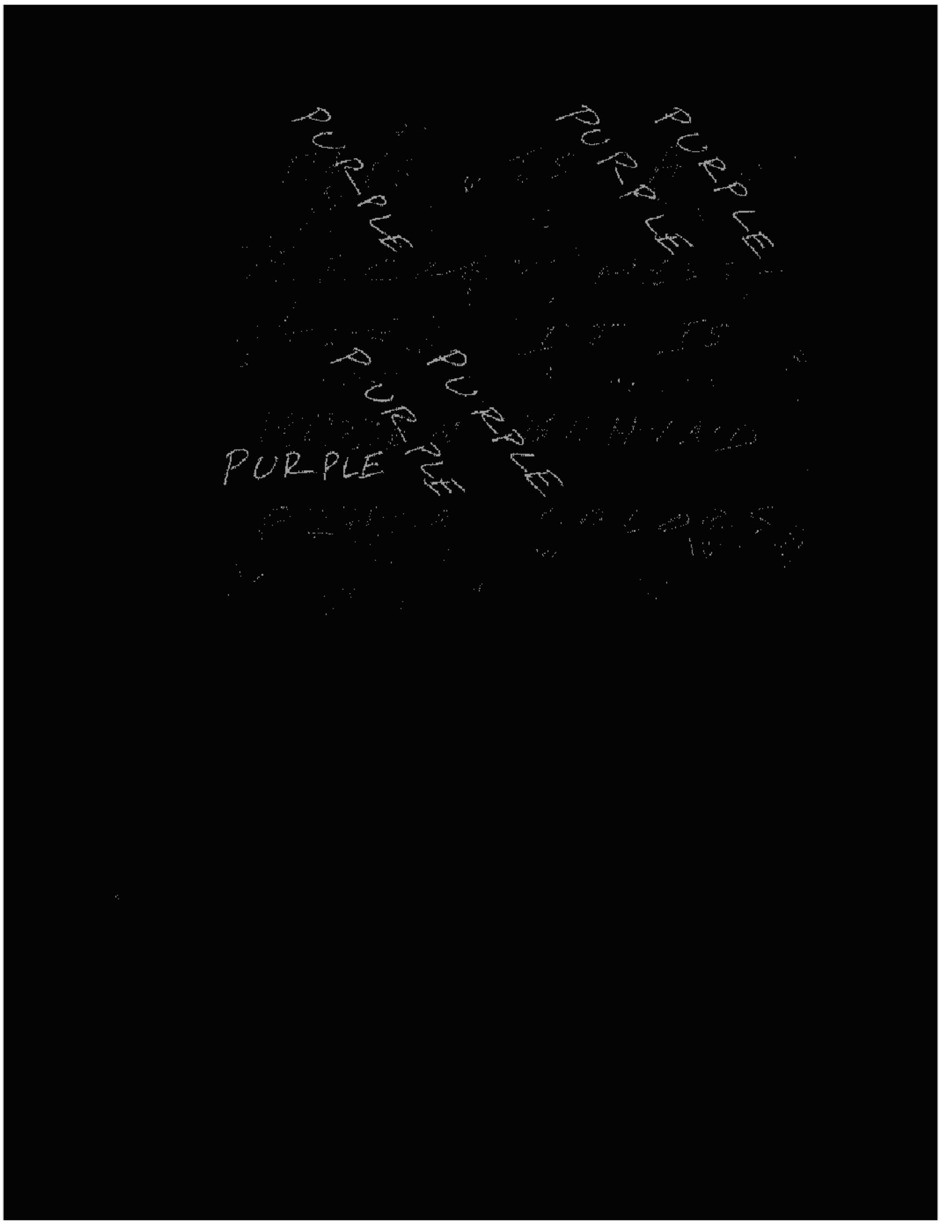
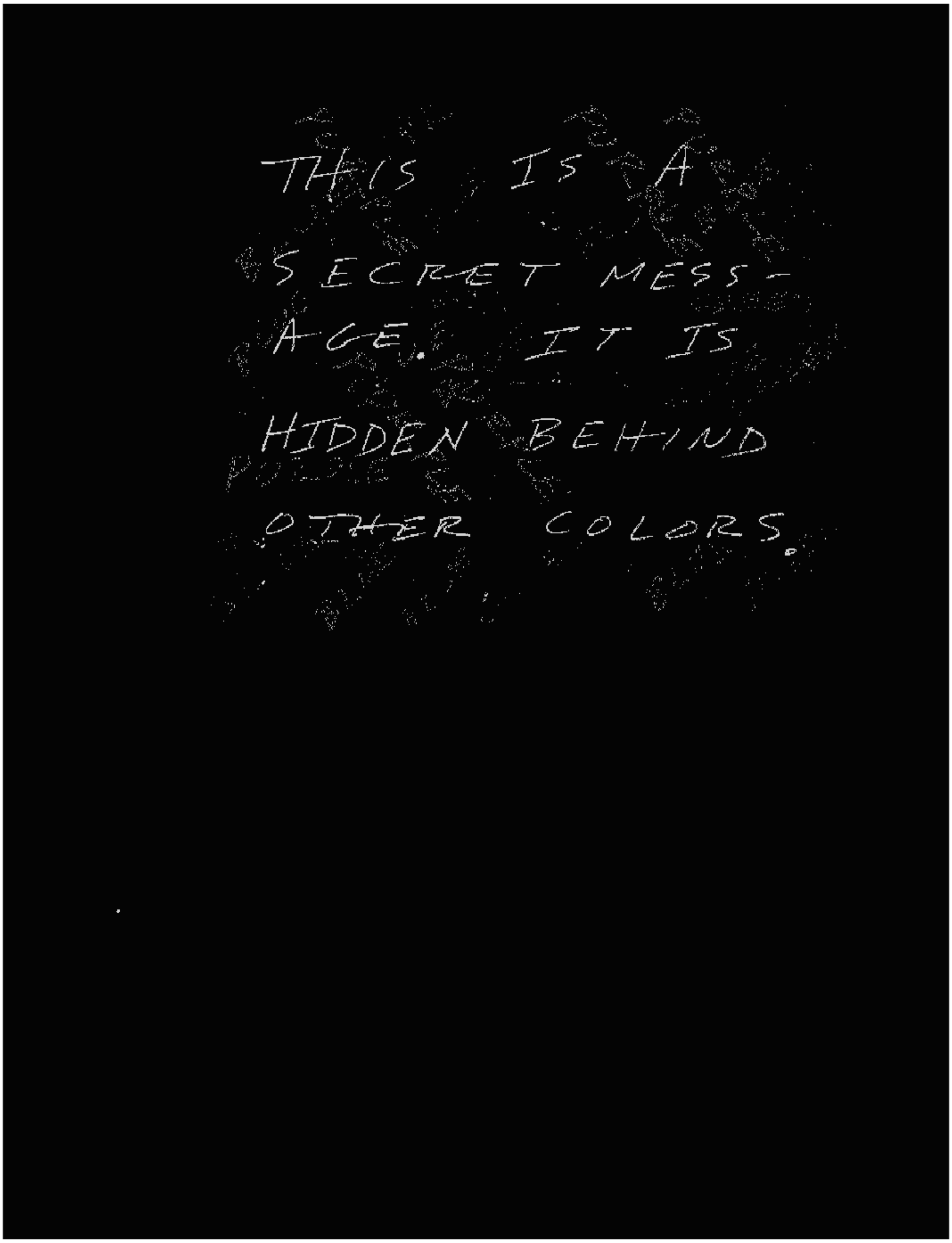
Name: Divyank Kulshrestha

Cluster Colors

The output shows text of different colors as white on a black background. I sub sampled the image in order to reduce computations. And then displayed each k-means cluster as white on a black background image. I used 10 clusters, instead of 8, to account for all the colors in the image, because using 8 clusters was not able to distinguish between green and blue inks. Some of the results had noise in them. I tried to clean up the noise using morphological operations, but it also erased the desired text, so I had to abort. I also did not try seeding, as the output I got was good enough without it.

OUTPUT IMAGES:





THIS IS A SECRET MESSAGE. IT IS HIDDEN BEHIND OTHER COLORS.

RED IS A DISTRACTION COLOR. RED IS QUICKLY READ IF YOU READ RED. BE WELL READ. READ TO LE AD.

THANCK YOU FOR READING.

BLUE INK

BLUE BLUE BLUE BLUE

BLUE

BLUE BLUE BLUE

BLUE BLUE BLUE BLUE.

GREEN

GR EE N GREEN

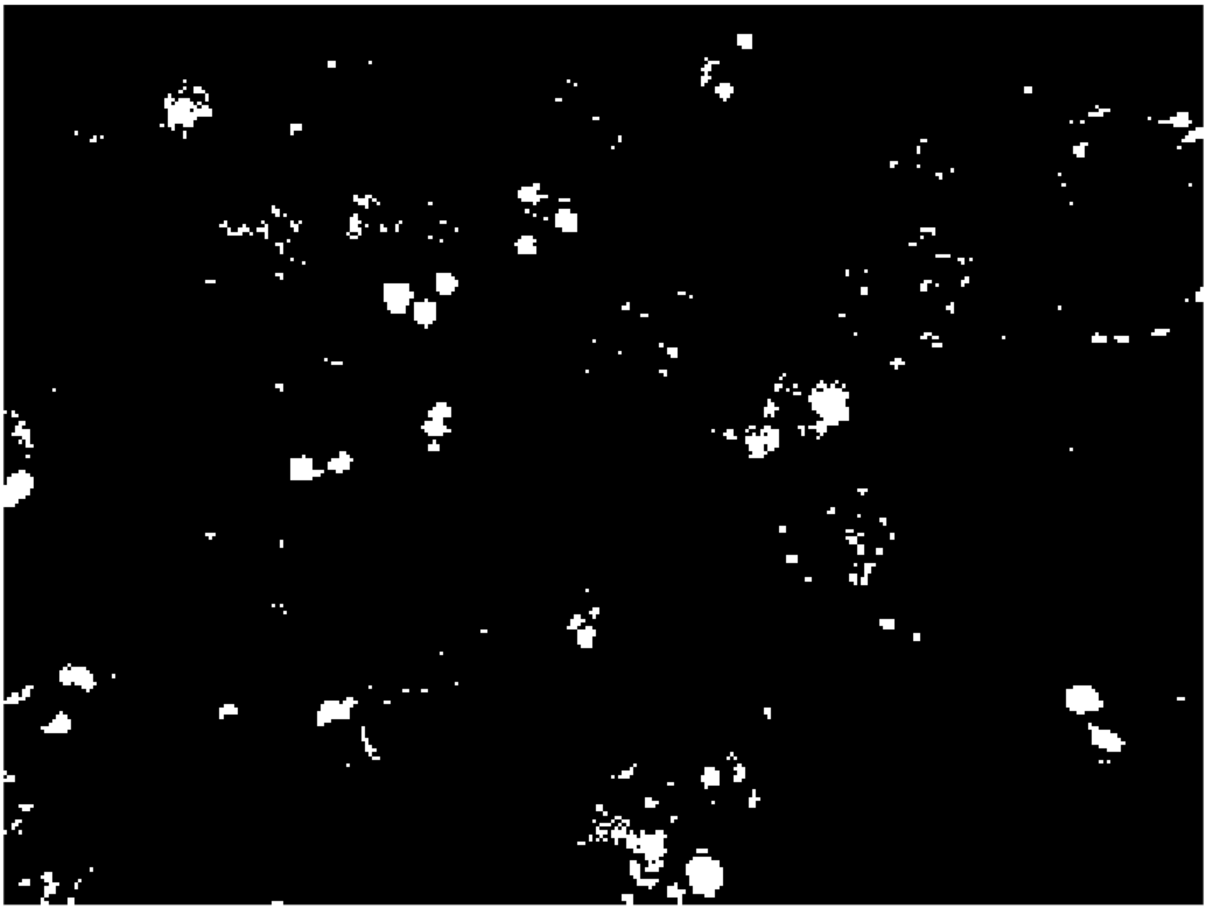
GREEN GREEN.

GREEN GREEN GREEN

Find Raspberries

I sub-sampled the image by skipping every fourth row/column for faster computations before getting clicks from the user using ginput(). Then I computed the mahalanobis distance for the image using the function ‘**mahal**’. To find a threshold value, I started with 5 which was producing insufficient results. Finally I tried 10 as the threshold, which seemed to give the most desired output. As I was still a little curious, I also tried using 20 for the threshold, but that ended up detecting the majority of the image as raspberries, which was again incorrect. I displayed the output as a binary image, where white color represents the pixels within the threshold distance (the raspberries in the image).

OUTPUT IMAGE:



Find Oranges

First I rotated the image to align it straight and then sub-sampled the image by skipping every tenth row/column for faster computations. The rest of the procedure is the same as for finding raspberries, with the threshold value used as 10 again.

OUTPUT IMAGE:

Conclusion

I realized the importance of converting the image to double and sub-sampling in this homework. Using K-means takes computations, and sub-sampling helps perform K-means much faster. Using K-means without sub-sampling took a lot of time to finish. Using the function **ginput()** again served as a refresher on one of the ways to get input from the user.

It took me some time to figure out how to **mahal()** function with the user-clicked inputs. But once I was able to calculate that, I used a zeros vector as the black image (pixel value 0), and updated pixels in that to white (pixel value 1), based on whether a pixel is within the threshold distance of 10 from the user-clicked input pixels. I used the same procedure to detect both raspberries as well as oranges.