Muhammed Halbutogullari

ECS 174 PS2

1) If we maintain a continuous vote space, we can rule out graph cuts since the array is not discrete & binary (ideal for graph cuts). K-Means would also be less than ideal here because it would for the most part disregard the actual values of the votes, as it would only attempt to minimize Euclidian distance between the points. Mean-shift would be the most ideal as it would group the points according to their average values, and thus would be the best utilization of the continuous vote space.

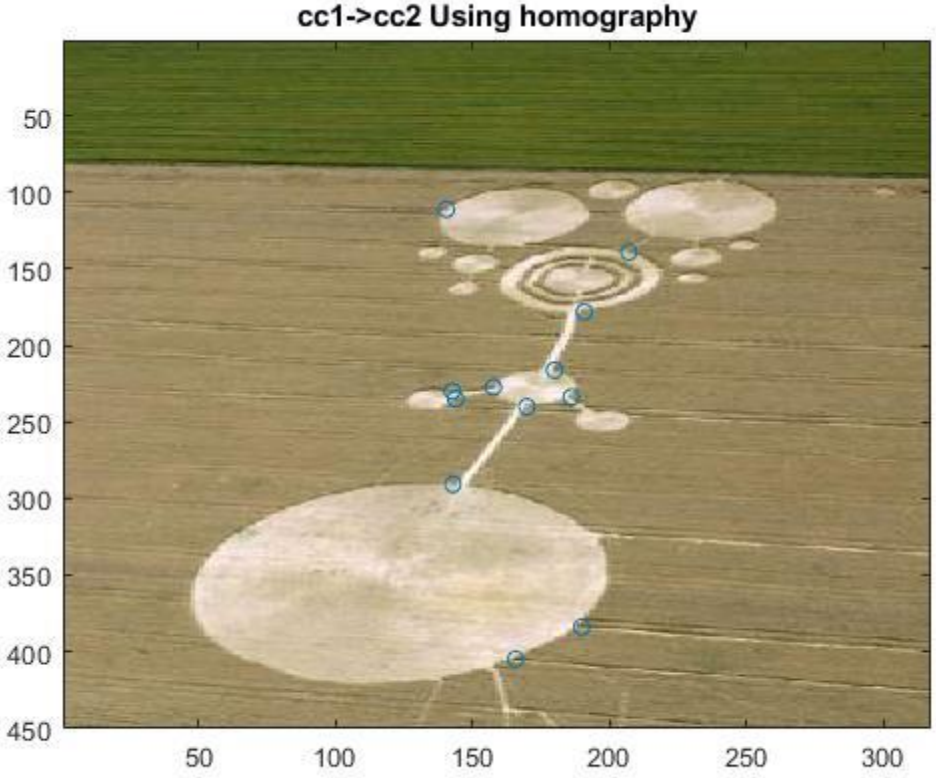
2) Depending on the initial location of the clusters, what will likely occur is that roughly half of each circle will be closest to one cluster and the rest will be close to the other cluster. Since K means minimizes the distance from the cluster to all the nearest (considering Euclidian distance) points, the clusters will keep moving toward the centers of one side of the figure. The final center assignments of the clusters will end up being somewhere near the inner circle, on opposite sides, as the clusters would roughly split all the points into two halves. A human would likely assign both cluster centers to the shared center of the concentric circles, however the K means algorithm only cares about Euclidian distance and thus cannot "see" the shapes, and thus it will simply split all the points into two halves. The only room for variability is how the graph is split, and the final split can be predicted by connecting the two initial cluster locations via a line, and then finding a perpendicular line* around halfway between them. Once we run the k means we will find the resulting clustering approximately split in two by this initial line*.

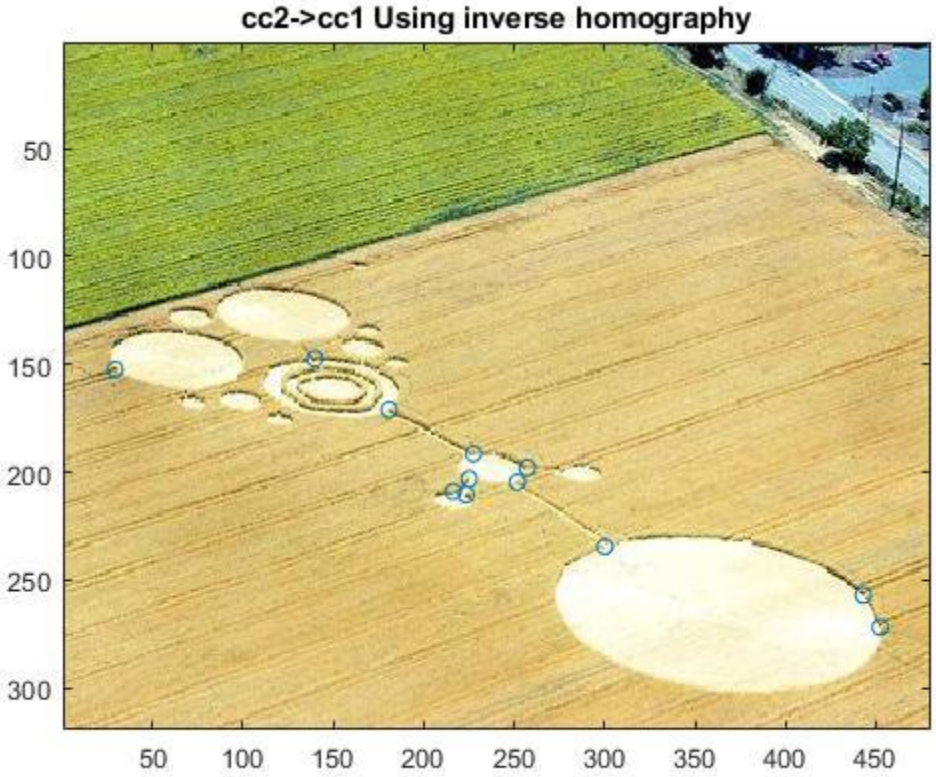
3 K=specified number of groups.

Since the question is asking in terms of similarity of area, and we have a binary image, we will group blobs according to their Euclidean distance from the blob centers. First, we initialize k points randomly in coordinate system. Then, match each point on image to nearest group. Once everything is grouped, take the average coordinate values for each group, and assign this as new center of that group. Recompute the nearest points using the new center. Repeat this until all points in k are not moving anymore, and you will have identified all the 'blobs' in this binary image based on their distance, their centers being the elements of k.

2A) Work contained in appropriately titled script in zip.

2B) Testing my homography code by extrapolating cc2 using the homography and cc1, and also vice versa (Script included in zip)


cc1->cc2 Using homography
50
100
150
200
250
300
350
400
450
50
100
150
200
250
300


cc2->cc1 Using inverse homography
50
100
150
200
250
300
50
100
150
200
250
300
350
400
450

2C) Worked very hard but I could not complete this code. Please look at the code because I actually had most of it down, including comments and overall understanding of what I did. I was so close to finishing except for a very pesky bug I could not figure out. Please consider generous, yet partial credit for this question considering I could not fully complete the rest of the problems.

2D) I did a portion of part D, by selecting the points.

2E)I also did a portion of part E by taking the pictures and selecting key points.