ECS174 / Seunghyup Alex Oh (916201993)

<PS#2>

Short answer problems [15 points]

1. When using the Hough Transform, we often discretize the parameter space to collect votes in an accumulator array. Alternatively, suppose we maintain a continuous vote space. Which grouping algorithm (among k-means, mean-shift, or graph-cuts) would be appropriate to recover the model parameter hypotheses from the continuous vote space? Briefly describe and explain.

Among the grouping algorithms, the mean-shift algorithm would be appropriate algorithm to recover the model parameter hypotheses from the continuous vote space. The mean-shift algorithm seeks modes or local maxima density in the feature space by taking only one parameter of a window size and does not have to process over all votes place in the vote space. This makes easier to find the maxima of the Hough space.
While the k-means algorithm with given continuous vote space, it is impossible to perform the algorithm as the k-means algorithm needs to calculate the distance between cluster center to all of the bins. The segmentation by graph-cuts algorithm tries to delete links that cross between segments into pixel units which does not appropriately correspond to this case.

2. Consider Figure 2 below. Each small dot denotes an edge point extracted from an image. Say we are going to use k-means to cluster these points' positions into k=2 groups. That is, we will run k-means where the feature inputs are the (x,y) coordinates of all the small dots. What is a likely clustering assignment that would result? Briefly explain your answer.

First of all, as k=2 groups, the algorithm will result two clusters. K-mean clustering aims to cluster observations into k clusters where each observation belongs to the cluster with the nearest mean – using Euclidian distance and minimization of the SSD (Sum of squared differences) among all the dots to the cluster centers. Thus, unlike how we human would cluster this Figure 2 into bigger, outer circle and smaller, inner circle, using k-means algorithm to cluster these points' positions into k=2 groups will result two circles to be straight cut in half – right in the concentric center and the cluster centers to be on both sides, each clustering result having two half of the circles.

3. Suppose we have access to multiple foreground 'blobs' acquired using connected components within a binary image. Write pseudocode showing how to group the blobs according to the similarity of their area (# of pixels) into some specified number (k) of groups. Define clearly any variables you introduce.

Simply speaking, we have to cluster the blobs by the size of area into k numbers of groups.

Steps

1) Find the center of mass of each blob.

2) Compute the radius of each blob, the area of each blob using the computed radius, and extract the circularity features by using computed result of area.

3) By using K-means algorithm, cluster blobs into k numbers of groups according to the similarity of their area, their circularity features.

Variables

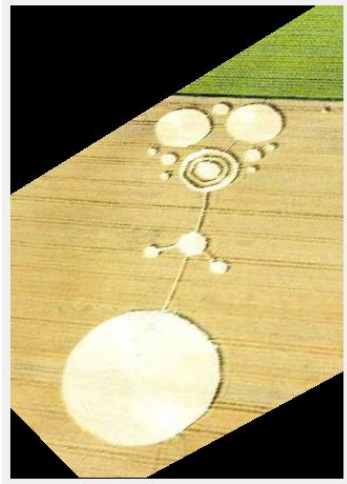
sum = sum of the position pixels
boundary = boundary pixel of the blob
//Omitted variables with obvious naming

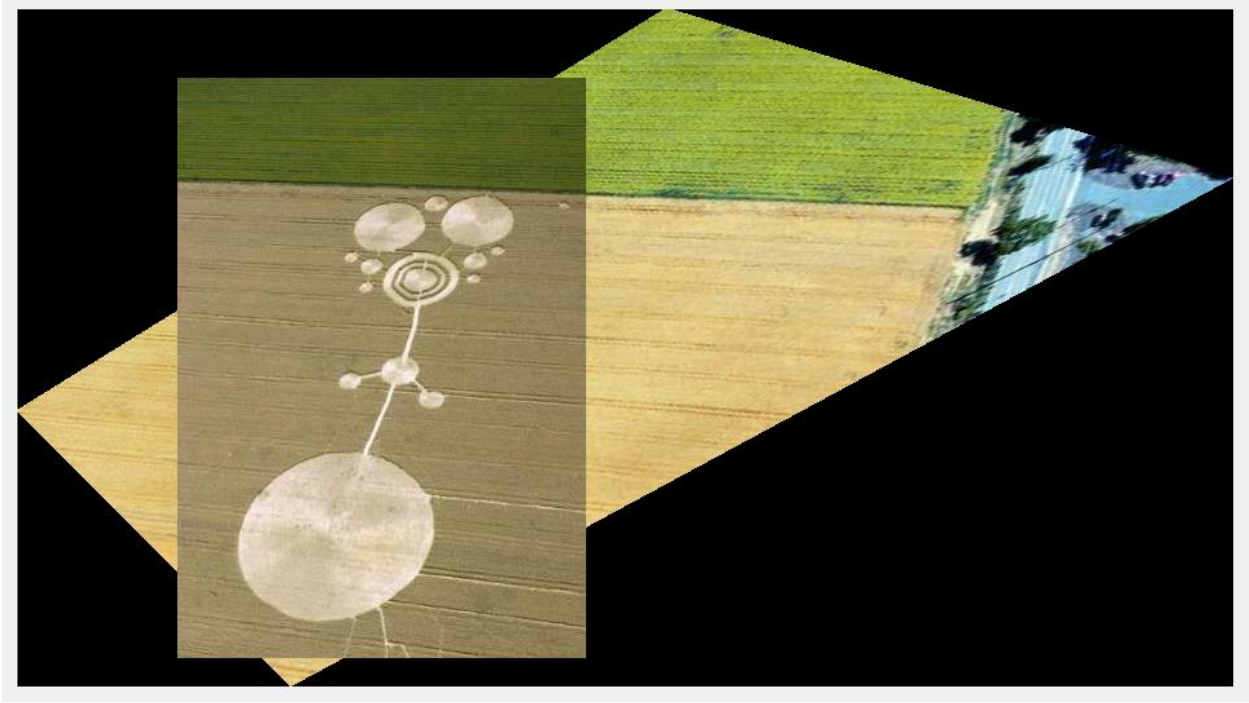
Pseudocode

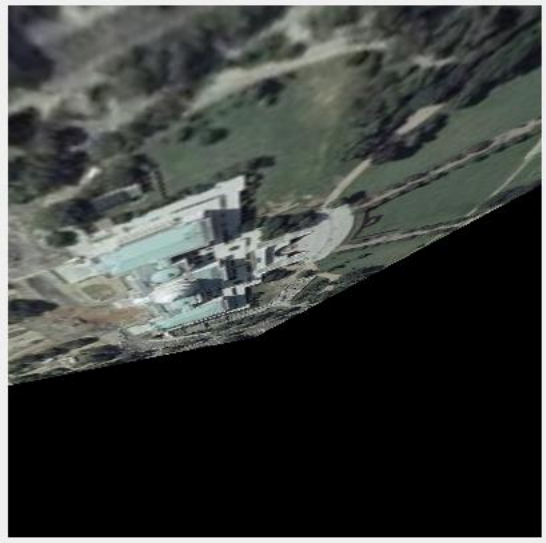
```
get-Center-Of-Mass (blob) {
    for every pixel in blob:
        sum += position(pixel);
    center-Of-Mass = sum / size(blob);
    return center-Of-Mass;
}

get-Circularity-features (blob) {
    for i = 0 to size(blob) in every blob:
        radius = boundary – (center-Of-Mass);
        area = (math.pi) * (radius^2);
        circularity[i] = area / size(blob);
    return circularity;
}

k-Means (circularity [], k) {
    return k-means = circularity [];
}
```

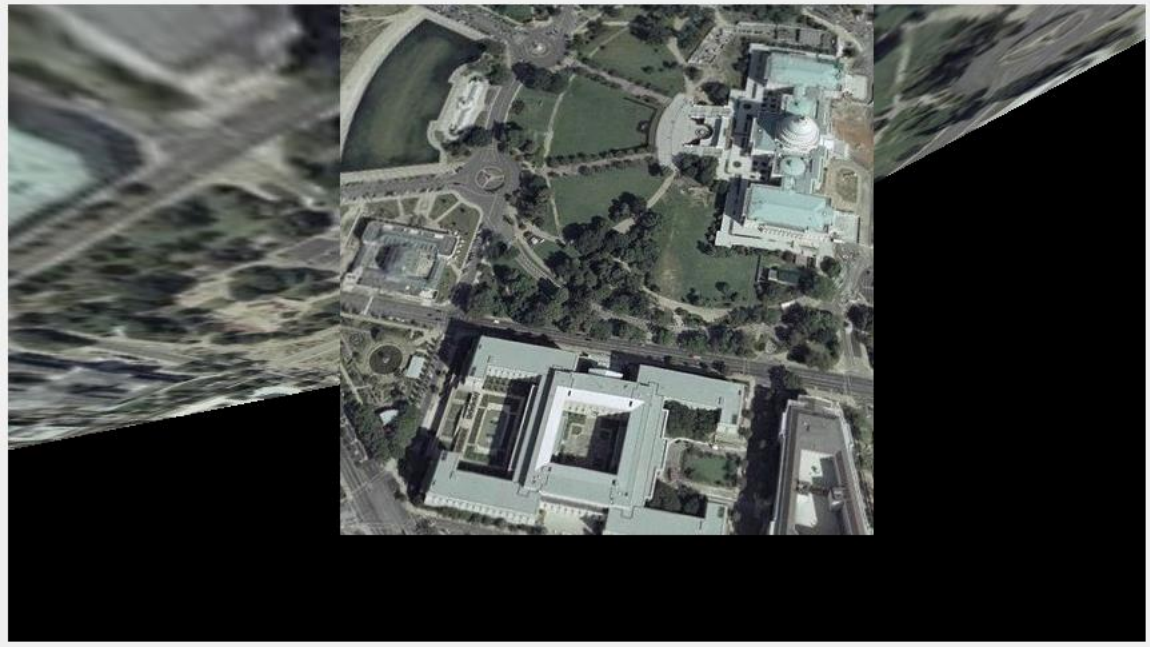
Programming [85 points]

2 (d).

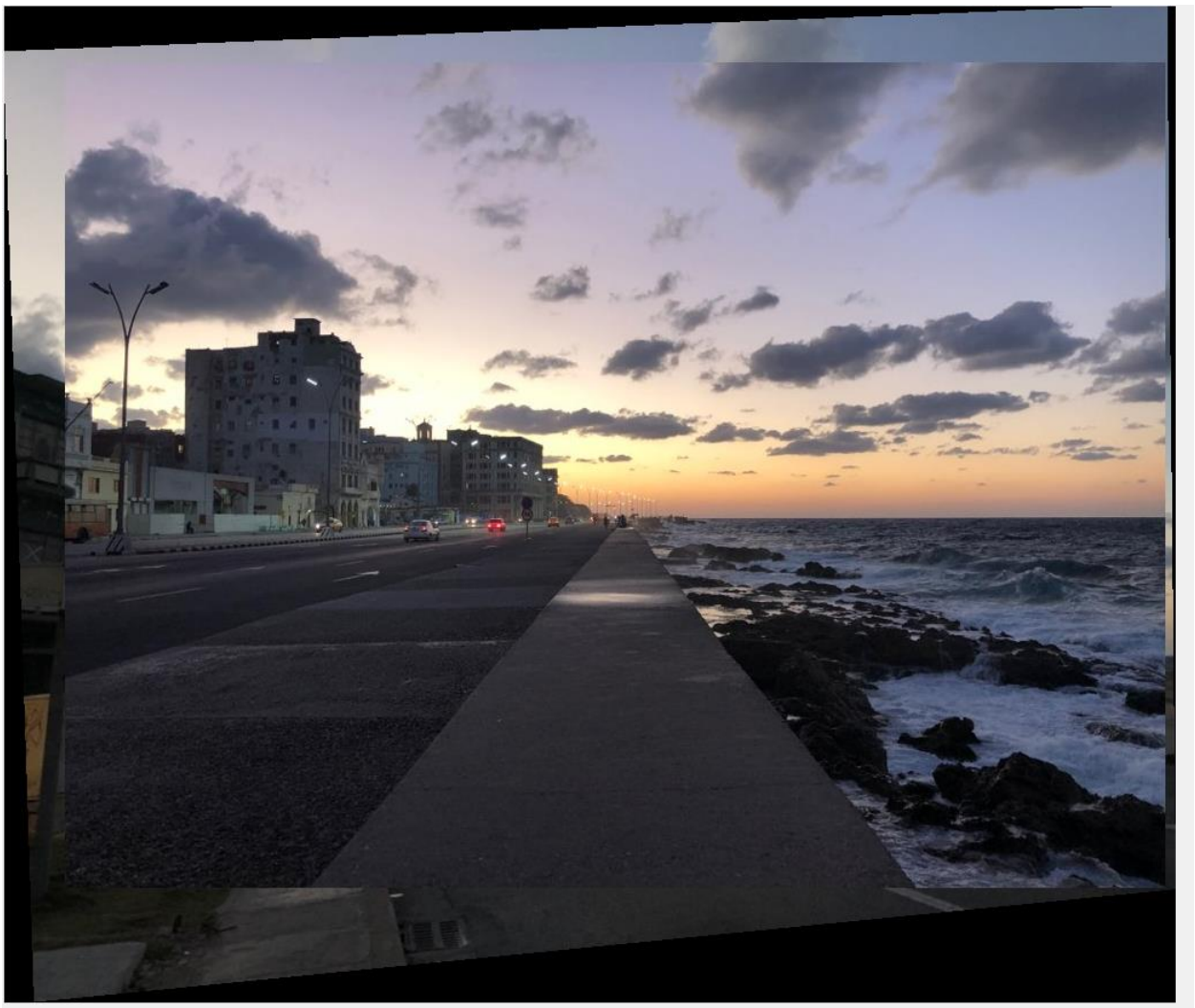








2 (e).



2 (f).