Aaron Lo
atl2374
7 October 2023

CS 376 Assignment 2

I. Short Answer Problems
   A. Compare the effects of 1) Dilation + Erosion against 2) Erosion + Dilation. Do they have the same effects? Why?
      1. These do not have the same effect. Dilation essentially expands the boundaries of the foreground while Erosion is an operation that reduces the boundaries of the foreground.
      2. When Dilation is performed before Erosion, gaps are expanded before shrunken. This has the effect of eliminating small gaps and smoothening the boundaries of objects.
      3. When Erosion is performed by Dilation, the gaps are shrunken before expanded. This removes small details from the image.
   B. List two examples of regular texture and two examples of near-regular texture.
      1. Regular Texture: Checkerboard Pattern, Black and White Stripes
      2. Near-Regular Texture: Brick Wall, Basket Weaving
   C. What are the cases where optical flow is not well-defined? Please given two concrete examples.
      1. One case where optical flow is not well-defined is a lack of features in a given space. For example, a uniformly painted wall would present challenges as there are no points to be able to track.
      2. Another case is when motion between frames is fast. Rapid panning of a Camera could lead to non-constant motion and a break between features, leading to bad results from optical flow.
   D. What are the advantages of RANSAC when compared with Hough Transform?
      1. RANSAC has several advantages when compared with Hough Transform. One is that RANSAC is more robust at handling data with significant amounts of noise as it fits based on the inliers, and ignores outliers while Hough Transform takes into account both. Another is that RANSAC is more efficient when dealing with a sparse set of points. Hough Transform needs an array of the entire space while RANSAC only needs to store a small subset of the points at each iteration. One more is RANSAC is more adaptive than Hough Transforms, as RANSAC does not need the predefined parametric models of Hough Transforms.

II. Circle Detection
   A. Explain your implementation in concise steps (English, not code).
      1. Hough Transform

a) I first define my hyperparameters theta_step and a threshold percentage. Then, for each point on an edge, it visits a circle around that point, adding one to each bin it visits. The circle is defined by a theta 0 to 2π in steps of theta_step. After all edges have been visited, the greatest value in the array is calculated. A threshold of the max value * threshold_percentage is calculated. Then, any value greater than this threshold is declared a center.
b) Step 1: Define hyperparameters for steps around edges / threshold percentage
c) Step 2: Find the edges in the image
d) Step 3: Visit a circle of distance *radius* away from each edge point, adding 1 to the corresponding accumulator array
e) Step 4: Define threshold has max value in accumulator array * threshold percentage. Centers are defined as all points above the threshold

2. RANSAC Function
    a) My RANSAC circle detection function is split into two parts. One part is a RANSAC single circle detection that detects a single circle. This works by running the RANSAC algorithm and using least squares to determine a circle's center point with three chosen points. After running for an adaptive number of iterations, it will return the center. The second part simply takes the center produced and removes all the circle points corresponding to that center. Then, it reruns the single circle detector until not enough points are produced from the certain circle.
    b) Step 1: Run RANSAC on the image
        (1) Step 1a: Randomly choose 3 points and fit a circle to it
        (2) Step 1b: Find the number of inlier points to that circle
        (3) Step 1c: Repeat until number of iterations has been completed and return the circle center with most number of inliers
    c) Step 2: End function if the returned circle center has too few inliers to it. Else, remove all the edge points that correspond with that circle and repeat Step 1.

B. HT Functions

Figure 1: Hough Transform on coins.jpg

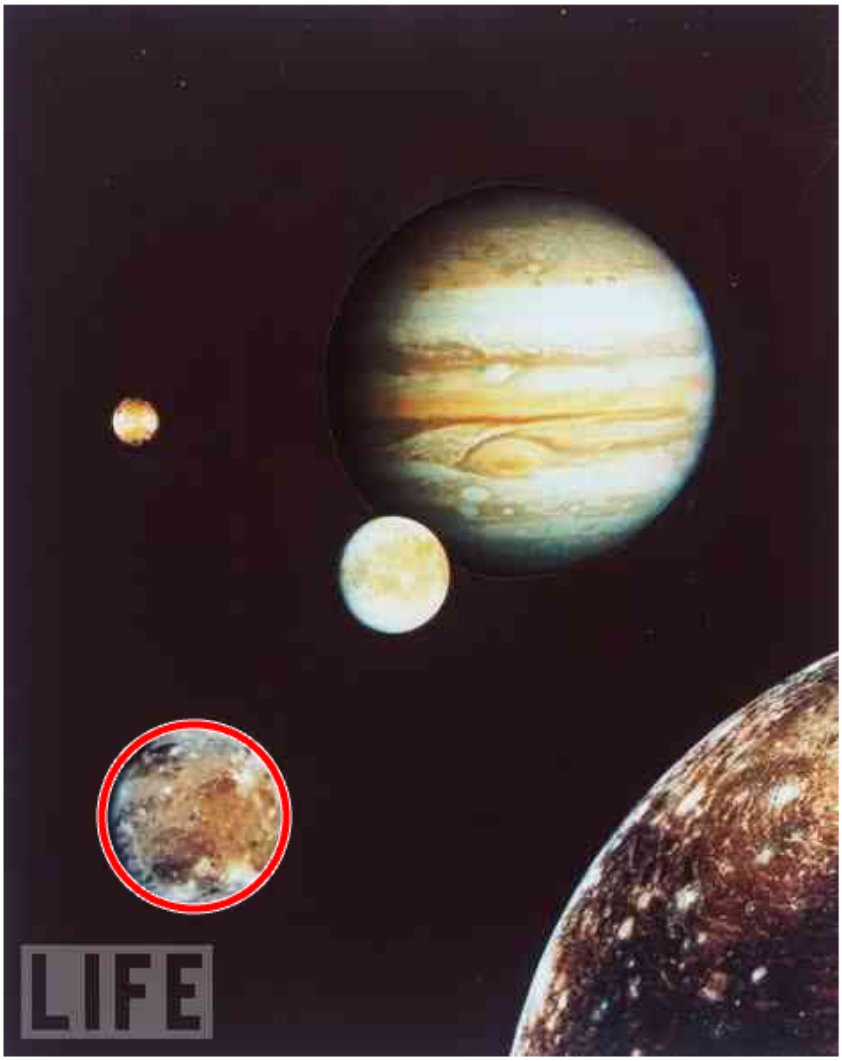

Figure 2: Hough Transformation on planets.jpg

Figure 3: Hough Transformation on gumballs

C. RANSAC Functions

Figure 4: RANSAC on coins.jpg

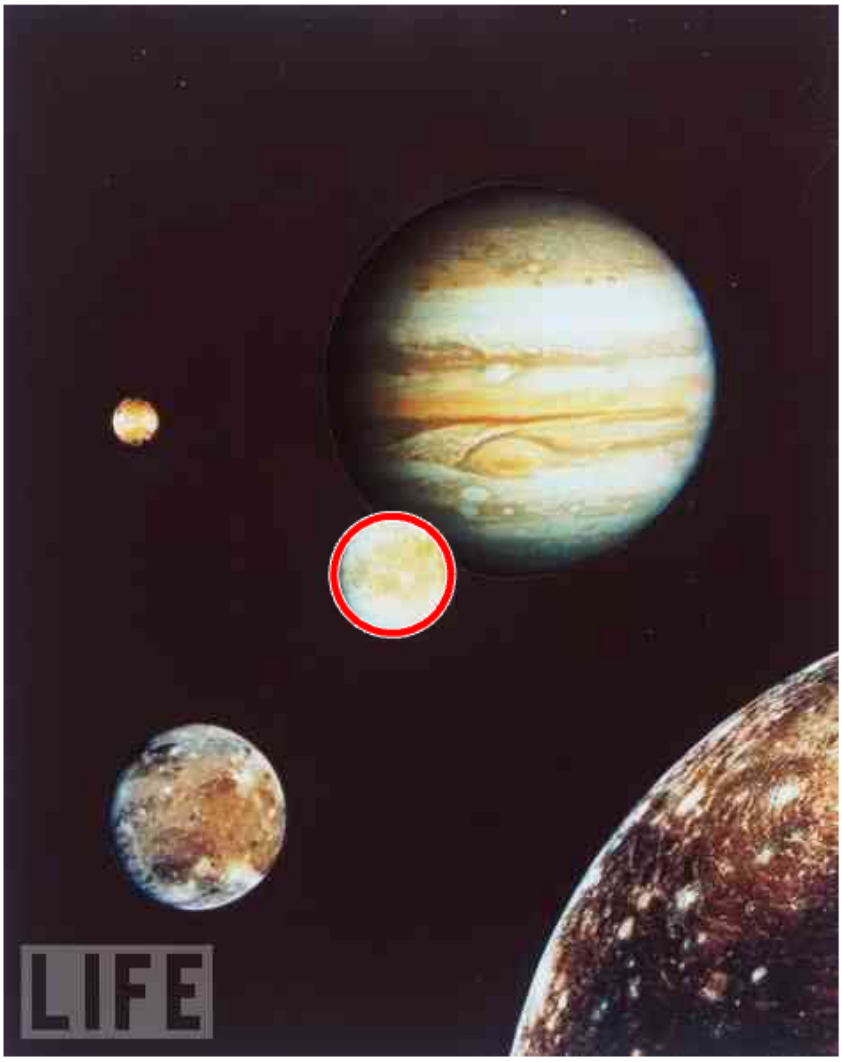

Figure 5: RANSAC on planets.jpg

Figure 6: RANSAC on gumballs

D. Hough Transform Accumulator Array

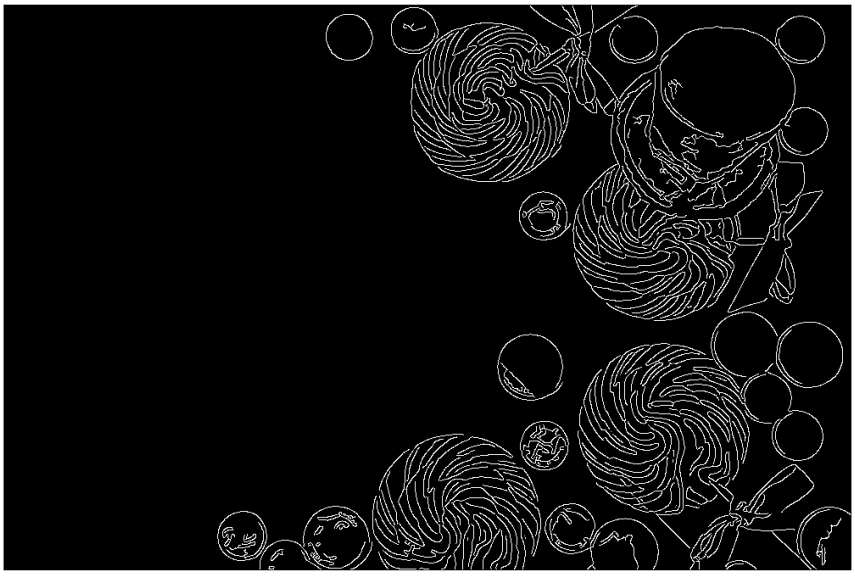

Figure 7: Edges Graph of coins.jpg

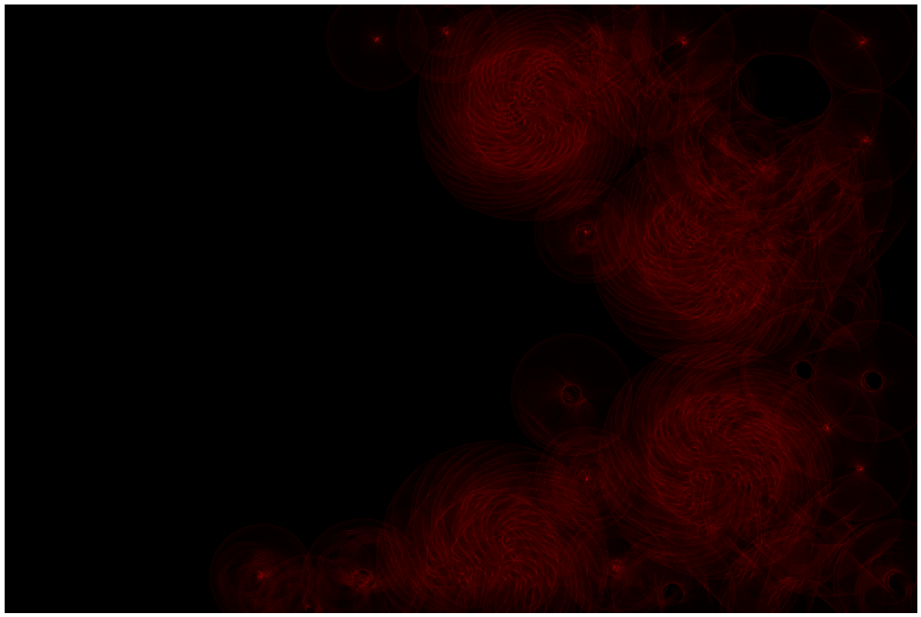

Figure 8: Hough Space Accumulator Array for coins.jpg

After the accumulation array has been created, my implementation finds the maximum value in the accumulation array. The number of circles found present is declared as points in the accumulation array with values greater than a threshold. This threshold is the max value in the array multiplied by a threshold percentage.

Figure 8 shows the accumulator array for coins.jpg. As can be seen, there are multiple concentrations of dots where there are centers of circles. In some places, the value is a lot higher due to the circle having a cleaner circle (allowing for better accumulation). Other places, due to the coin's edges not being fully perfect, it has a more "fuzzy" center (where values fall into other bins).

E. RANSAC Circle Fitting

1. For circle fitting, I use the least squares method explained in class to fit

$$\begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ & \vdots & \\ x_N & y_N & 1 \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} = \begin{bmatrix} x_1^2 + y_1^2 \\ x_2^2 + y_2^2 \\ \vdots \\ x_N^2 + y_N^2 \end{bmatrix}$$

$$\boldsymbol{Ap} = \boldsymbol{b}$$

my circles. When given 3 points by my RANSAC method, I use least squares to calculate p1, p2, and p3, where p1 = 2x, p2 = 2y, and p3 = r^2 - x^2 - y^2. Since radius is already defined for the method, I only look at the calculated centers.

F. BIN Sizes

Figure 9: Hough Transform with bin_size = 1 on coins.jpg

Figure 10: Hough Transform with bin_size = 3 on coins.jpg

Figure 11: Hough Transform with bin_size = 5 on coins.jpg

1. As seen in Figure 9, 10, and 11, as the bin_size used in the Hough Transform cicle detection increased, the number of false positive circles also drastically increased. This occurs because the increased quantization reduces the precision of the circle detection. This leads to non-circular patterns accumulating votes for non-circular patterns of our declared radius. Of course, this causes the noise and false positives as can be seen in edge dense areas to go way up.

G. Progress of RANSAC as number of tries increase

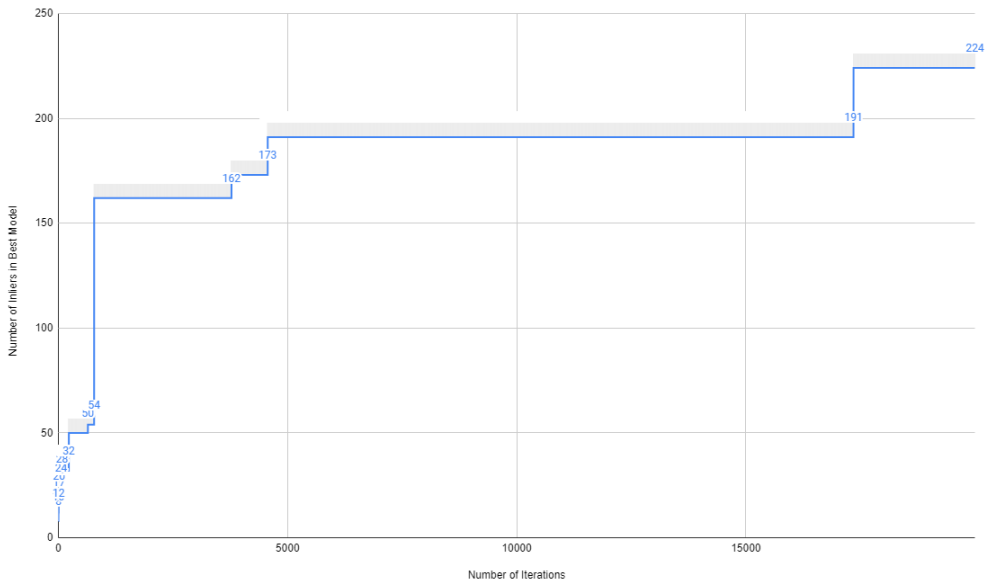

Figure 12: Number of iterations of RANSAC vs Number of Inliers for best model

1. Figure 12 shows the progress of RANSAC as the number of iterations increases. Because of how I implemented RANSAC, each iteration only tests for a single circle. It can be seen that as the number of iterations increases, at the start, there is a rapid increase in the best model found for the circle. Then, it tapers off as it becomes harder to find a better circle.

III. Image Segmentation with K-Means

A. Cluster Pixels

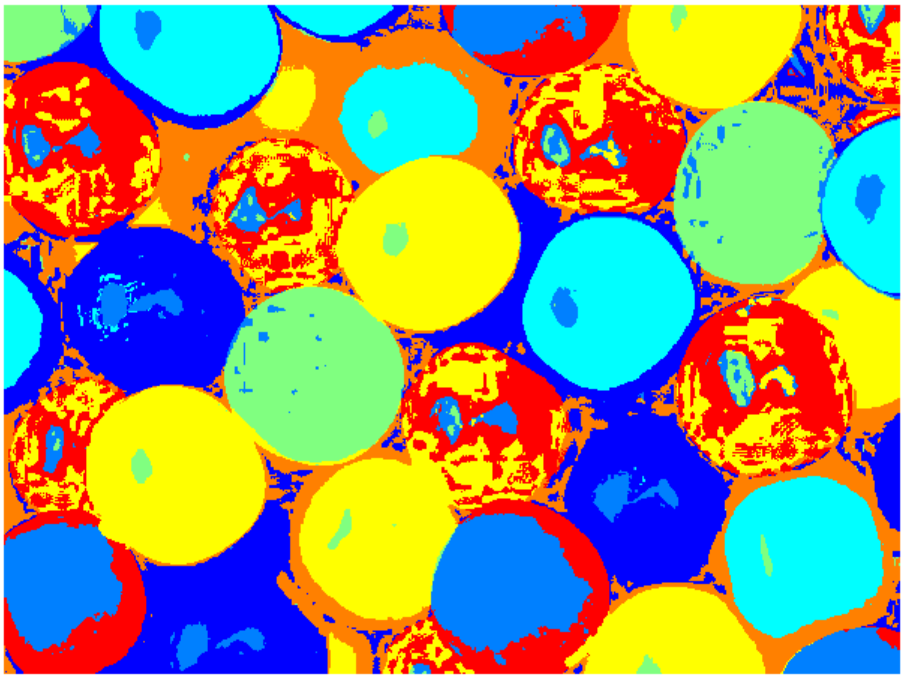

Figure 13: K-Means Clustering $k = 5$ of gumballs.jpg

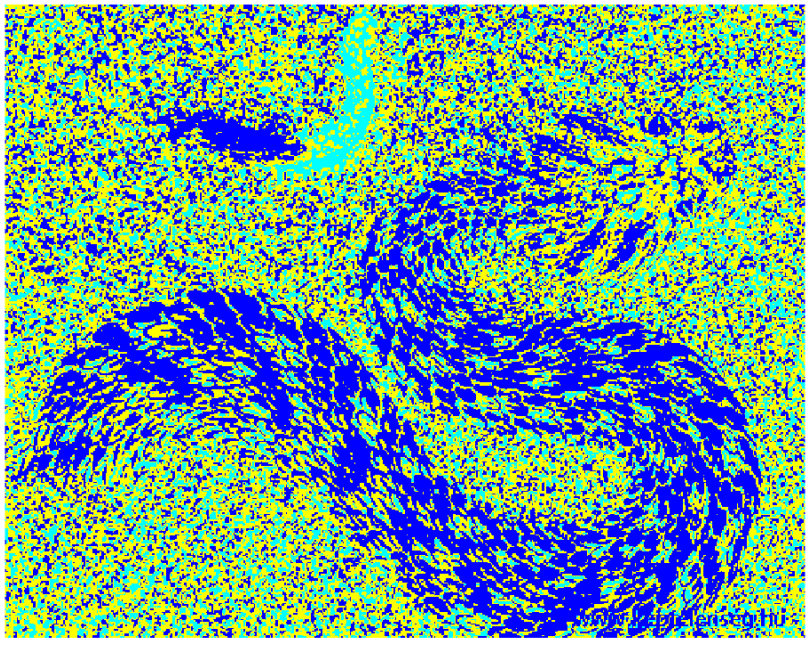

Figure 14: K-Means Clustering $k = 3$ of snake.jpg

Figure 15: K-Means Clustering $k = 5$ of twins.jpg

Figure 16: Picture of Eagle

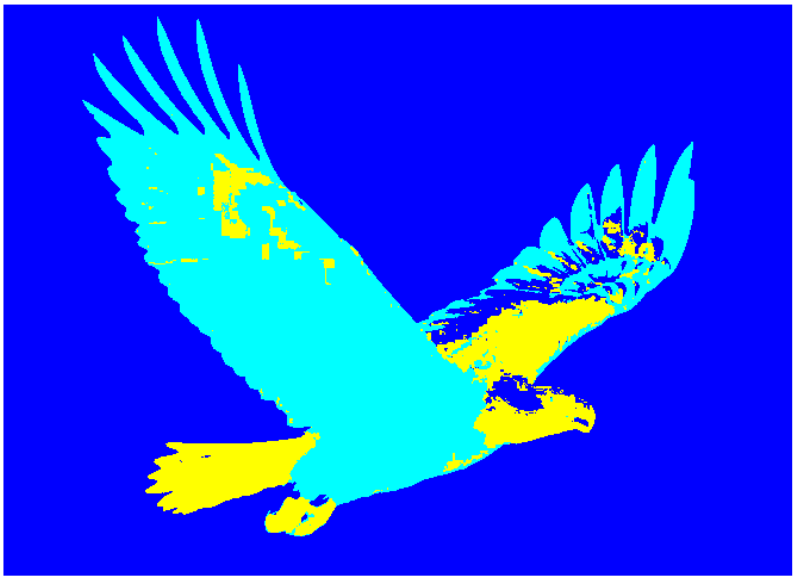

Figure 17: K-Means Clustering $k = 3$ of eagle

B. boundaryPixels

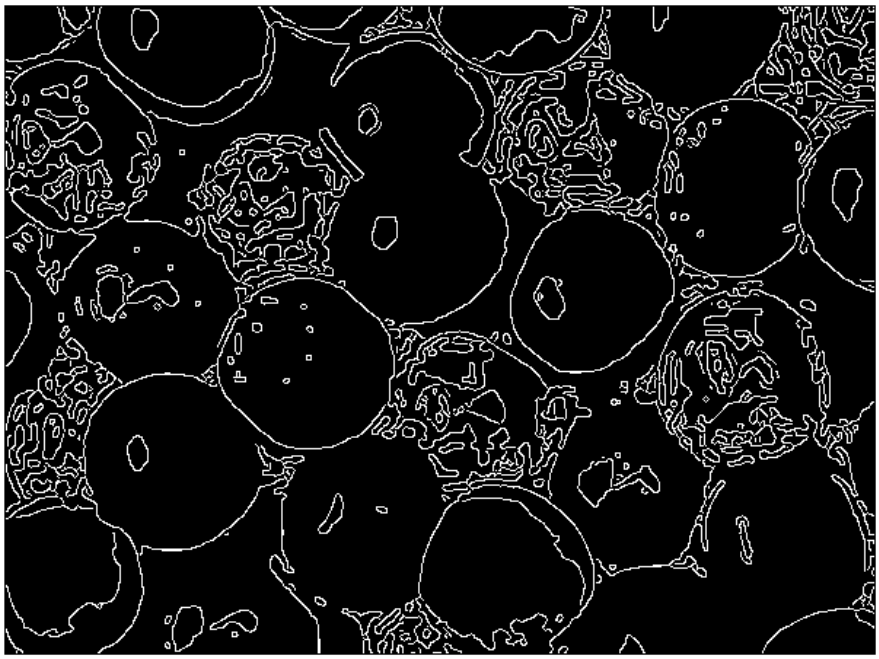

Figure 18: Edges to K-Means of gumball.jpg

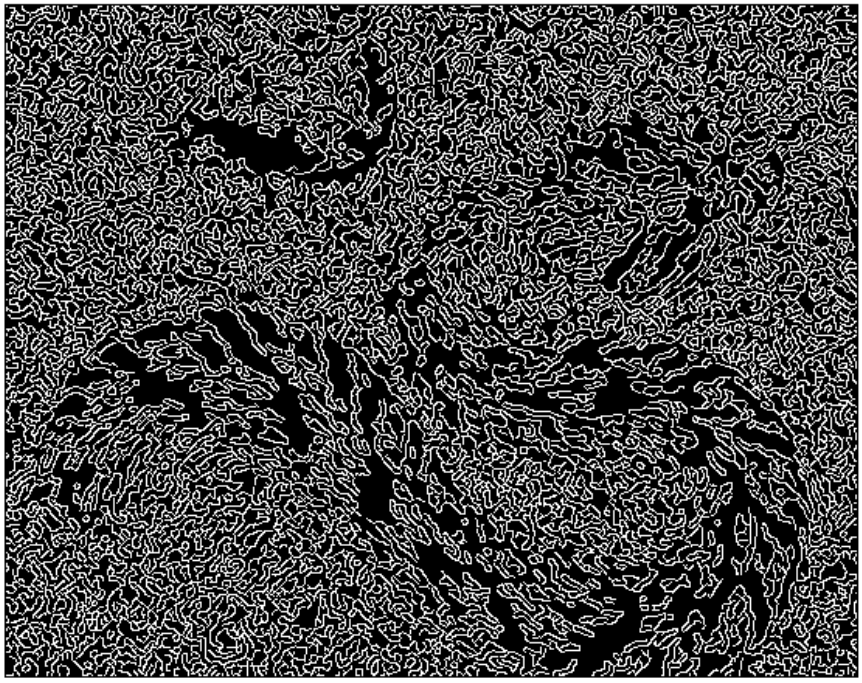

Figure 19: Edges to K-Means of snake.jpg

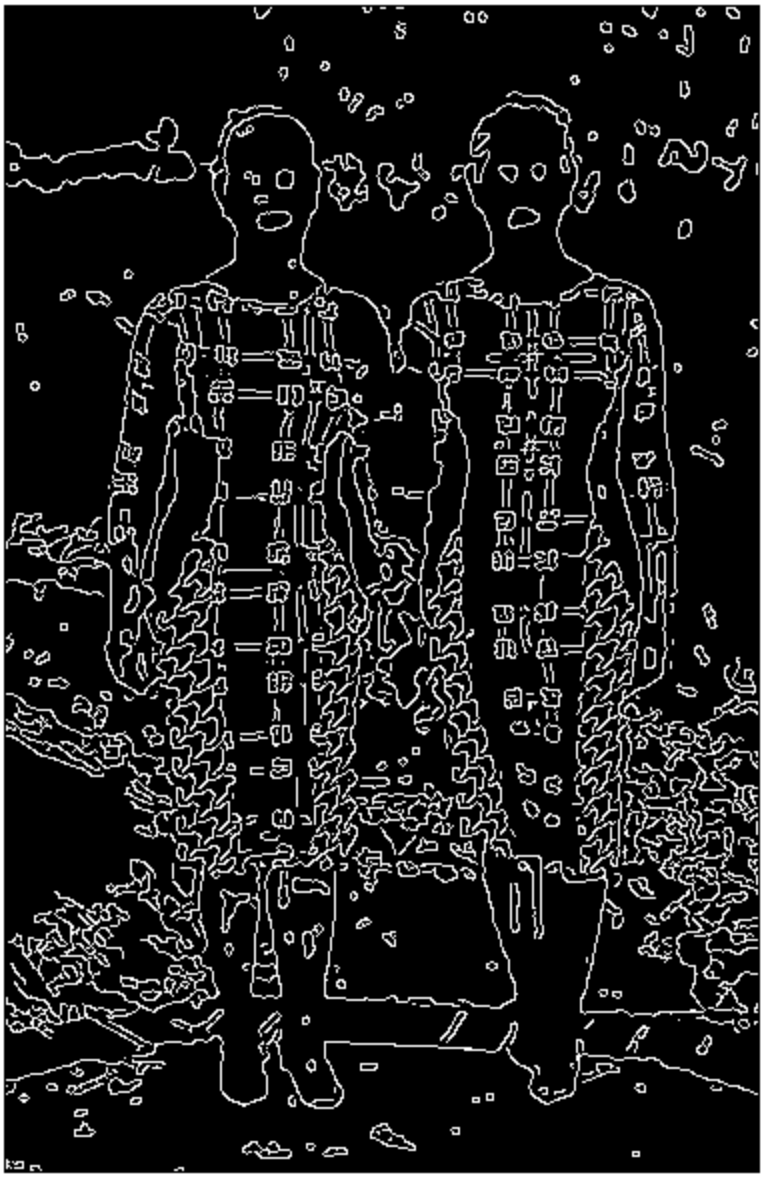

Figure 20: Edges to K-Means of twins.jpg

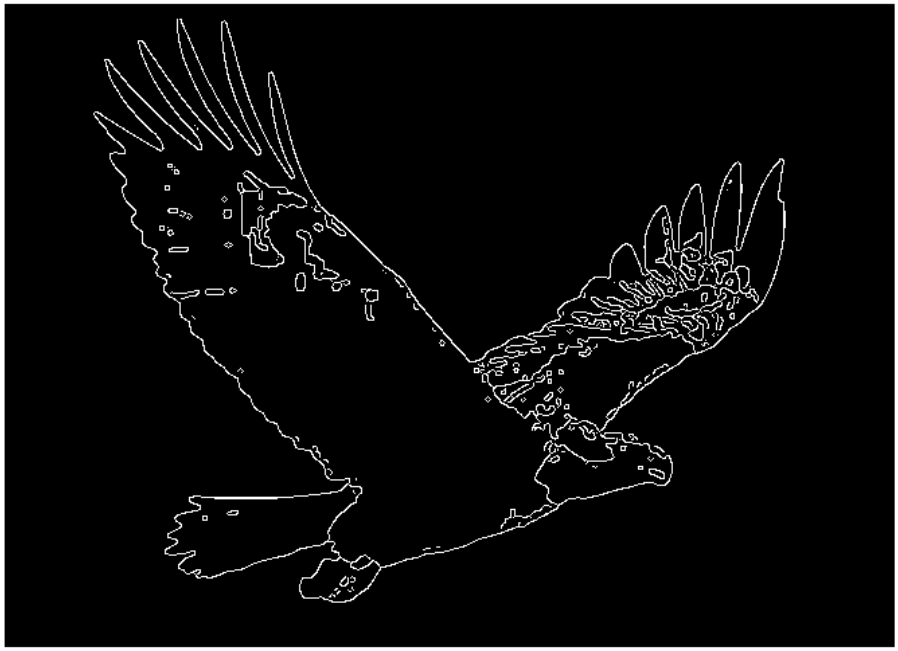

Figure 21: Edges to K-Means of eagle