# ECE 6310- INTRODUCTION TO COMPUTER VISION

# LAB 5: ACTIVE CONTOURS

**Submitted To:**

**Dr. Adam Hoover**


**Ashit Mohanty**

**C13582787**

**amohant@g.clemson.edu**


**Date of submission: 10-27-2020**

## INTRODUCTION

This lab project required the implementation of an active contour algorithm. The main deliverable of the project is an image with the contour points perfectly surrounding the desired object in the image. The inputs to the active contour program are the text file with the coordinates of the contour points and the original ppm image file. The outputs are the Sobel edge gradient magnitude image, the image with the original position of the contour points, the image with the contour points that have been moved due to the active contour algorithm, and a text file with the coordinates of the final coordinates of the contour points after the active contour algorithm. This report gives a brief step by step explanation of the active contour program. The program was written in C and the active contour algorithm was performed on the hawk.ppm image as attached underneath,

*Figure 1: Original hawk.ppm image*

## SOBEL EDGE GRADIENT MAGNITUDE IMAGE

To compute the Sobel edge gradient magnitude image, the original image had to undergo the process of convolution using two filters, one for the horizontal gradient and one for the vertical gradient. The filters used were,

$$f_1 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad f_2 = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

where, $f_1$ is the filter for the horizontal gradient and $f_2$ is the filter for the vertical gradient. This convolution would be performed for every pixel in the image. Therefore, the net value of the pixel, $G$, would be calculated by the formula,

$$G = \sqrt{G_x^2 + G_y^2}$$

where, $G_x$ is the value of the horizontal gradient and $G_y$ is the value of the vertical gradient. After performing the convolution process and after computing the net values, the entire set of values must be normalized to the greyscale range of 0-255. The output of the Sobel filter is as shown in the image below,

*Figure 2: Sobel filter applied to hawk.ppm image. Saved as hawk_sobel.ppm.*

The image pixel intensity values of the Sobel image would be used in the active contour program to compute the external energy.

## ORIGINAL CONTOUR POINTS

The original contour points were read from the text file, “hawk_init.txt”. These contour points were plot on the original image in the form of a “+” symbol with the center of the plus symbol being the coordinate read from the text file. The image with the initial contour points is shown in the image as attached below,

*Figure 3: Original hawk.ppm image with the initial position of contour points (black "+" signs). Saved as hawk_marked_initial.ppm*

**ACTIVE CONTOUR ALGORITHM**

The main objective of the active contour algorithm is to reduce the total energy of the contour points. This is done by considering the pixel with the minimum total energy in a 7x7 contour window around the initial contour point. This pixel with the minimum energy would be the new position of the contour point. This step is repeated for all the contour points and this would be equal to 1 iteration of the active contour algorithm. I have decided to run the algorithm for 50 iterations, as that gives the best result. The total energy for every pixel is dependent on two internal energy terms and one external term such that,

$$E_{total} = E_{internal\_1} + E_{internal\_2} + E_{external}$$

The energy terms are explained in the sections below.

- Internal Energy 1: $E_{internal\_1}$

This term corresponds to the first internal energy term. This energy is equal to the square of the distance between the point in the 7x7 window around the current contour point and the next contour point. The formula looks something like,

$$E_{internal\_1} = (x_i - x_{i+1})^2 + (y_i - y_{i+1})^2$$

In the above equation $x_{i+1}, y_{i+1}$ are the coordinates of the next contour point and $x_i, y_i$ have to be iterated for every pixel in the 7x7 window centered around the current contour point.

- Internal Energy 2: $E_{internal\_2}$

This term corresponds to the second internal energy term. This energy is equal to the square of the difference between the average distance and the distance between a pixel in the window centered around the current contour point and the next contour point. The formula is shown in the equation below.

$$E_{internal\_2} = \left(dist_{avg} - \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2}\right)^2$$

The average distance is computed by the average of all distances between the consecutive contour points as per their initial positions. The second part that is under the square root is the same as $E_{internal\_1}$.

- External Energy: $E_{external}$

This term is the external energy. This term is the negative of the square of the Sobel edge gradient for that pixel position, as computed above. The formula is shown below,

$$E_{external} = -(I\_sobel_i)^2$$

In the above formula, the external energy is to be computed for every pixel in the 7x7 window for which the total energy is being computed.

The total energy is computed by the formula,

$$E_{total} = a * E_{internal\_1} + b * E_{internal\_2} + c * E_{external}$$

The variables $a, b, c$ in the above equation are the weights associated with each energy term. To ensure that the weights are the only ones driving each energy term and not their magnitude, all the energy terms were normalized to a value between a range of 0-1. The normalization was performed by using the formula,

$$I_N = (I - Min)\frac{newMax - newMin}{Max - Min} + newMin$$

where,

$I$ = Energy value, but not normalized

$Min$ = Minimum energy value in the particular energy term, but not normalized

$Max$ = Maximum energy value in the particular energy term, but not normalized

$newMin$ = New Minimum energy value in the particular energy term, in this case, 0

$newMax$ = New Maximum energy value in the particular energy term, in this case, 1

$I_N$ = New energy value after being normalized.

The active contour algorithm was run for 50 iterations and the values of the weights were taken as

| Weight | Value |
|---|---|
| a (weight for internal energy 1) | 1 |
| b (weight for internal energy 2) | 1.6 |
| c (weight for external energy) | 0.8 |

So, the energy equation becomes,

$$E_{total} = 1 * E_{internal\_1} + 1.6 * E_{internal\_2} + 0.8 * E_{external}$$

After running the contour algorithm for 50 iterations, the final positions of the contour points were printed in a text file, "hawk_points_final.txt". These positions are shown in a tabular form as attached underneath,

| Point No. | Column coordinate | Row coordinate |
|---|---|---|
| 1 | 277 | 126 |
| 2 | 279 | 137 |
| 3 | 279 | 148 |
| 4 | 279 | 159 |
| 5 | 275 | 169 |
| 6 | 272 | 178 |
| 7 | 268 | 188 |
| 8 | 264 | 197 |
| 9 | 260 | 207 |
| 10 | 255 | 216 |
| 11 | 255 | 228 |
| 12 | 247 | 237 |
| 13 | 238 | 236 |
| 14 | 226 | 240 |
| 15 | 224 | 252 |
| 16 | 220 | 262 |
| 17 | 210 | 267 |
| 18 | 199 | 265 |
| 19 | 196 | 255 |
| 20 | 195 | 245 |
| 21 | 189 | 252 |
| 22 | 183 | 243 |
| 23 | 176 | 237 |

| | | |
|---|---|---|
| 24 | 181 | 228 |
| 25 | 182 | 216 |
| 26 | 182 | 205 |
| 27 | 183 | 195 |
| 28 | 184 | 184 |
| 29 | 185 | 171 |
| 30 | 187 | 159 |
| 31 | 190 | 145 |
| 32 | 194 | 134 |
| 33 | 198 | 122 |
| 34 | 204 | 112 |
| 35 | 217 | 104 |
| 36 | 225 | 99 |
| 37 | 235 | 90 |
| 38 | 247 | 84 |
| 39 | 260 | 88 |
| 40 | 265 | 96 |
| 41 | 267 | 107 |
| 42 | 274 | 116 |

The final image with the position of the final contour points is shown in the image as attached underneath. The contour points are shown by the black “+” signs.

*Figure 4: hawk.ppm image with the final positions of contour points after 50 iterations. Saved as hawk_final_image.ppm.*

## CONCLUSION

The choice of weights indicates that the energy is most biased on the second internal energy term. The weight associated with the second energy term is 1.6. This indicates that the contour points will prioritize being equidistant to each other most over anything else. As a result, it is clearly visible from the final image that the contour points are equidistant to each other.

The first internal energy term has a weight of 1 multiplied to it, indicating that its weight is normal. This would have a normal influence on the contour positions. As this energy is only responsible for the shrinking action, like that of a rubber band, I decided not to tweak this weight as it is perfectly imitating the shrinking action.

The third energy term is the external energy, which has a weight of 0.8 multiplied by it. The reason for choosing the weight to be the least among all other energy terms is because as can be seen in Figure 2, the Sobel gradient image, there are a few extremely bright/ white pixels inside the hawk's body. The contour points would get pulled to these points as well, but the main objective was to form a chain of contour points around the hawk's outline. Hence, a weight of 0.8 gives the desired output as seen in the final image, that tracks the outline of the hawk quite successfully.

The final image as shown in the previous section is reattached again below, but with the contour points colored white, for better visibility.

*Figure 5: hawk.ppm image with the final positions of contour points, as white "+" symbols, after 50 iterations.*