ECE 6310 – Introduction to Computer Vision – LAB 5 REPORT

ACTIVE CONTOURS

TASKS and REFERENCE

- Load greyscale image "hawk.ppm" and list of contour points.
- Process contour points through contouring algorithm.
- Draw "+" centered around point with greyscale value = 0.
- Algorithm:
  - Two internal energies:
    - Square of distances between the points.
    - Square of average distance between points.
  - External energy:
    - Square of the image gradient magnitude calculated using convolution with Sobel template.
  - Energies normalized to scale 0-1.
  - Energies weighted equally.
  - 30 iterations of algorithm must take place
- Outputs to produce:
  - Copy of image with initial contour points
  - Copy of image with final contour points
  - Result showed for the Sobel edge gradient magnitude image (entire image)
  - Pixel coordinates of final contour provided in a table

Contour Point Reference

	C0	C1		C2	C3	C4	C5	C6
R0								
R1								
R2								
R3								
R4								
R5								
R6								

DISCUSSION

Program

The following methods were written to achieve an efficient program:

1. Find_distance
2. Sqr_distance
3. Sqr_deviation
4. Mark_contour_point
5. Find_gradient_magnitude
6. Normalize

Varying Parameters

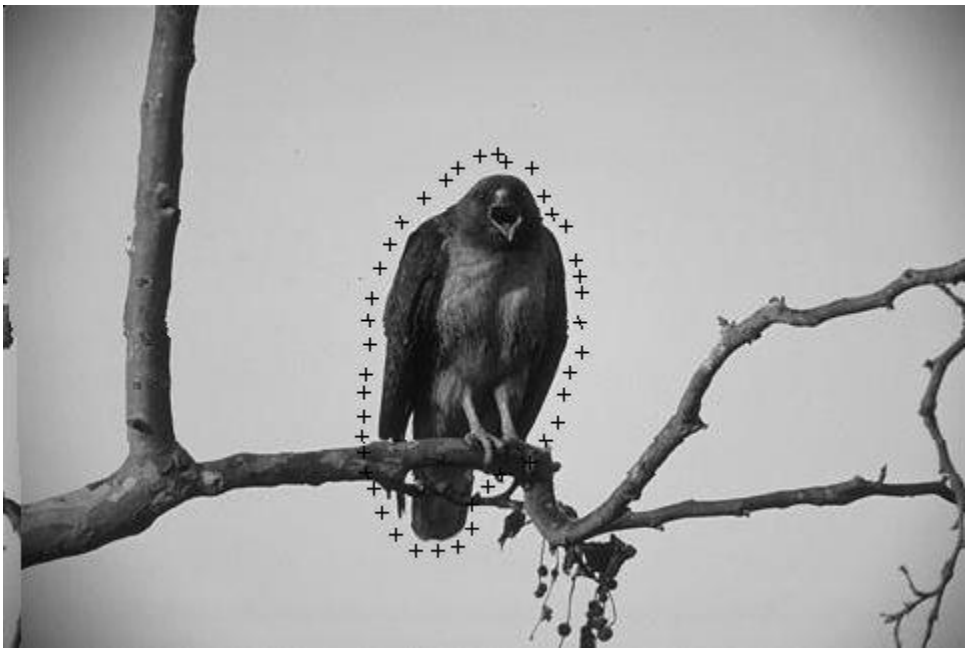
There are three energies driving the contour points inward towards the hawk- distance energy, deviation energy and external gradient magnitude energy. Initially, all three weights were set to 1 and the window size was 7x7. However, the final contour image showed that the points had gaps, and some were far from the boundary of the hawk. Below is an example output that shows this:

From a logical standpoint, it was clear that the deviation energy needed to be increased and the distance energy needed to be decreased in order to fill the gaps and have equidistant contour points. Additionally, the points that were off the boundary of the hawk were brought closer to the hawk by increasing the window size from 7x7 to various different sizes. After trial and error, the optimal parameter values chosen to obtain the best output are given below:

Parameter	**Value**
Distance Energy	0.75 (weight)
Deviation Energy	3.81 (weight)
Gradient Magnitude Energy	2 (weight)
Window Size	13x13
Iterations	35

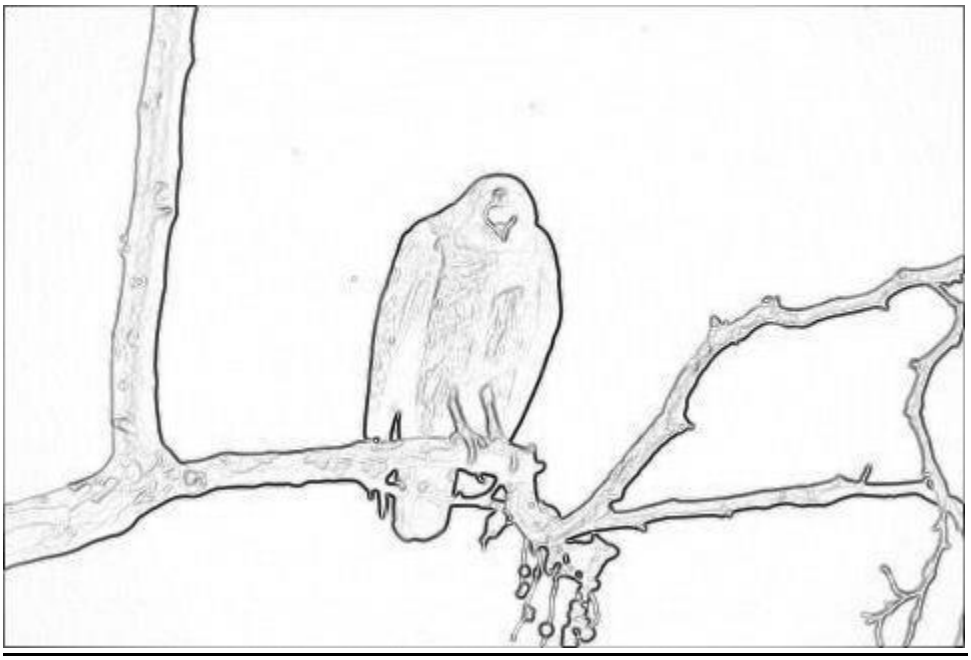
OUTPUTS

Initial Contour Image



Sobel Filter Image

Inverted Sobel Image



Final Contour Image

Output Discussion

At the optimal parameters, the output image clearly shows equidistant points that outline the hawk's body almost perfectly well. The small deviation that occurs is near the branch of the tree which is of the same pixel intensity as the hawk. However, this is minimal, and the hawk's tail is also outlined neatly.

Final Contour Points

Row	Col
98	265
108	266
117	273
127	276
138	278
149	278
160	277
170	274
180	270
190	266
200	262
210	257
220	257
226	266
228	254
238	226
249	228
257	222
265	215
266	206
263	196
254	195
244	193
233	194
217	187
209	195
206	181
196	182
186	183
175	184
164	185
153	187
142	190
131	194
122	197
112	203
106	211
100	222
93	231
85	242
85	253
89	260

Difficulties Encountered and Rectifications

1. Indexing: Invalid index points caused my output image to crash. Fixed by offsetting the row and column variables by loop variables.
2. Warped sobel image: Not writing the image as “wb” binary caused the sobel image to be warped. Code ran perfectly in Linux, but not in windows. Fixed by changing “w” to “wb.”
3. Contour points not on hawk’s border: varied window size to determine optimal window size at 13x13.
4. Contour points accumulated at one side: After trying multiple combinations, it was discovered that increasing deviation energy, external energy and decreasing distance energy spread the contour points out at equal distances and moved points away from boundary towards hawk’s body.
5. Segmentation fault: not allocating memory to the right type of data for a certain variable caused this.
6. Invalid writes: When varying the window size, the memory allocation was not changed, causing the errors “invalid writes” at multiple points of the code. Fixed by changing the size of memory allocated to these variables with respect to change in window size.

The following images show contour points of algorithm at different iterations:

Iteration 1:

Iteration 12:

Iteration 25:

Iteration 35 (final):