Canny Edge Detection using First Principles

Taneesh
2018csb1186
2018csb1186@iitrpr.ac.in

Indian Institute of Technology
Ropar

Abstract

This document is the report of Part-1 CS518 Computer Vision Assignment 2, where we implement the Canny Edge Detection Algorithm for Edge Detection from first principles. The Canny edge detector uses a multi-stage algorithm to detect a wide range of edges in images. It was developed by John F. Canny in 1986.

1 Introduction

Canny edge detection [**?**] is a technique to extract useful structural information from different vision objects and dramatically reduce the amount of data to be processed. It has been widely applied in various computer vision systems.

Key idea is that Canny used the **calculus of variations** – a technique which finds the function which optimizes a given functional. The optimal function in Canny's detector is described by the sum of four exponential terms, but it can be approximated by the first derivative of a Gaussian.

2 Methodology | Algorithm

The Canny edge detection algorithm is composed of 5 steps :

- Noise reduction and Luminence Extraction
- Gradient calculation
- Non-maximal Suppression
- Double Thresholding to apply labels to the pixels
- Edge Tracking by Hysteresis

  Given below are details of each step of canny edge detector.

2.1 Noise Reduction and Luminence Extraction

Edge Detection results are highly sensitive to image noise, therefore we need to get rid of these noises. We can apply Gaussian Blur (with Kernel of 3x3, 5x5, 7x7 etc.) to smooth the

image. In our example, we have converted the image to float64 and got the luminence image and then convolved with 5x5 Gaussian Kernel.

Algorithm 1 convertToLuminenceImageAndReduceNoise(img)

1: $R \leftarrow img[:,:,0]$
2: $G \leftarrow img[:,:,1]$
3: $B \leftarrow img[:,:,2]$
4: $grayscale \leftarrow (0.2989 * R) + (0.5870 * G) + (0.1140 * B)$
5: $grayscale_blur \leftarrow convolve(luminence\_image, gaussian\_kernel(kernel\_size = 5, sigma = 1))$
6: **return** luminence_image

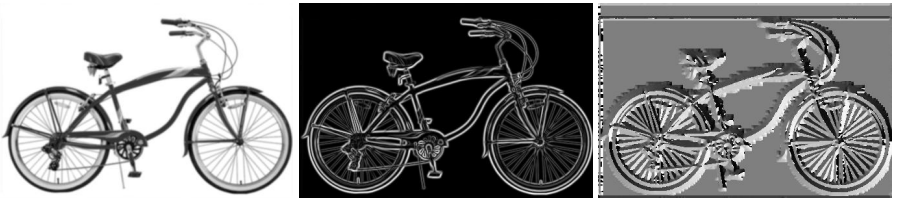
(a) Original Image (b) Luminence Image

Figure 1: Original Image and Luminence Image

2.2 Smoothed Gradient Calculation

Here you can have a look at the gradient calculations which we have done by soble filter.

Now we will use non maximal supression to thin the edges and hence to more localize the output.



(a) Luminence Image (b) Gradient Magnitude Image (c) Gradient Direction Image

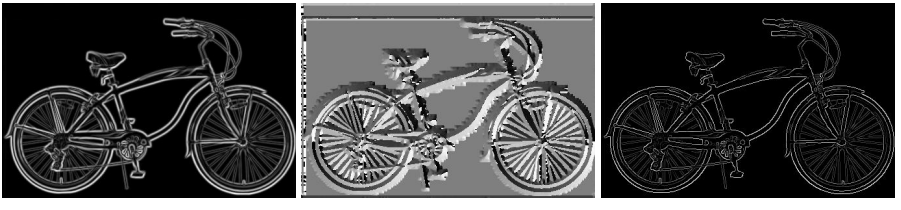
Figure 2: Gradient Images obtained from Luminence Image using Sobel Filters and Convolution

Algorithm 2 getImageGradientAndDirection(img)

1: $sobel\_filter\_x \leftarrow array[[-1,0,1],[-2,0,2],[-1,0,1]]$
2: $sobel\_filter\_y \leftarrow array[[1,2,1],[0,0,0],[-1,-2,-1]]$
3: $gradx \leftarrow convolve(img, sobel\_filter\_x)$
4: $grady \leftarrow convolve(img, sobel\_filter\_y)$
5: $grad\_magnitude \leftarrow \sqrt{gradx^2 + grady^2}$
6: $gradient\_magnitude \leftarrow gradient\_magnitude/gradient\_magnitude.max()$
7: $gradient\_direction \leftarrow tan^{-1}(gradx/grady)$
8: **return** gradient_magnitude, gradient_direction

2.3 Non-Maximal Suppression

Even after this differentiation filter the edges are quite thick but the localisation is a parameter of good image detection for this Non-maximal supression is done, in this in the direction of gradient it is checked if the pixel is largest if it is, then it's marked 1 else it's marked 0. below is the algorithm. As we only have 4 directions available in the real images we just use 4 angles 0 , 45, 90 and 135 and accordingly assign each pixel an angle value according to the range. As in below algorithm.



(a) Gradient Magnitude Image (b) Gradient Direction Image (c) Thinned Edge Image

Figure 3: Thinned Edge Image obtained from Gradient Images using Non-Maximal Suppression

2.4 Double Thresholding

Still there are some extra pixels because of noise and color variations to curb them the following method is used.

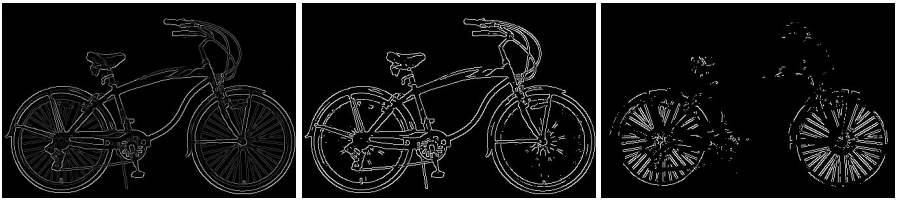
The double threshold step aims at identifying 3 kinds of pixels: strong, weak, and non-relevant (definitely not edge).

- Strong pixels are pixels that have an intensity so high that we are sure they contribute to the final edge.
- Weak pixels are pixels that have an intensity value that is not enough to be considered as strong ones, but yet not small enough to be considered as non-relevant for the edge detection.

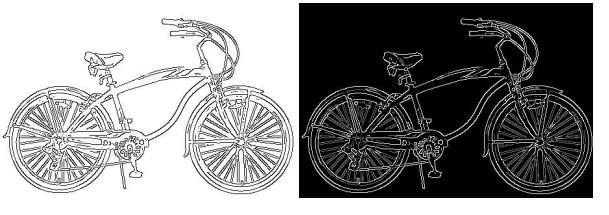
We need to two thresholds T_low and T_high to determine the strong and non-relevant pixels and pixels in b_w the range are weak pixels. Thresholds chosen are T_low = 0.10 and T_high = 0.25 they should be again and again tried until concrete results are acheived.

Algorithm 3 nonMaximalSuppression(gm = gradient_magnitude, gd = gradient_direction)

```
thinned_edge_image ← zeros.shape_like(gm) array
gd[gd < 0] ← gd[gd < 0] + π
for i ← 1 : img_length − 1 do
    for j ← 1 : img_width − 1 do
        x = y = 1.0
        if (0 ≤ gd[i][j] < π/8) or (7π/8 ≤ gd[i][j] ≤ π) then
            x ← gm[i][j+1]
            y ← gm[i][j−1]
        else if (π/8 ≤ gd[i][j] < 3π/8) then
            x ← gm[i+1][j−1]
            y ← gm[i−1][j+1]
        else if (3π/8 ≤ gd[i][j] < 5π/8) then
            x ← gm[i+1][j]
            y ← gm[i−1][j]
        else if (5π/8 ≤ gd[i][j] < 7π/8) then
            x ← gm[i−1][j−1]
            y ← gm[i+1][j+1]
        end if
        if gm[i][j] ≥ x and gm[i][j] ≥ y then
            thinned_edge_image ← gm[i][j]
        else
            thinned_edge_image ← 0
        end if
    end for
end for
return thinned_edge_image
```



(a) Thinned Edge Image (b) Strong Edge Image (c) Weak Edge Image



(d) Definitely Not Edge Image (e) Threshold Image

Figure 4: Threshold Image (Strong Pixels having greater intensity) obtained from Thinned Edge Image using Double Thresholding

Algorithm 4 thresholdingEdges(grad_magnitude, Tlow, Thigh)

```
label ← zeros.shape_like(image) array
for i ← 0 : img_length do
    for j ← 0 : img_width do
        if (grad_magnitude[i][j] ≥ Thigh) then
            label[i][j] ← 1.0
        else if (grad_magnitude[i][j] ≥ Tlow) and (grad_magnitude[i][j] ≤ Thigh) then
            label[i][j] ← 0
        else
            label[i][j] ← −1
        end if
    end for
end for
return label
```

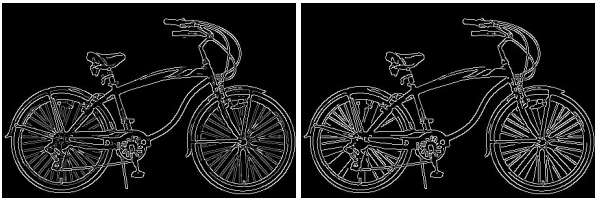
2.5 Edge Tracking by Hysteresis

Based on the threshold results, the hysteresis consists of transforming weak pixels into strong ones, if and only if the weak pixel is directly or indirectly connected to a strong pixel.

For this we use a dfs based approach and apply recursive dfs it is a time consuming step and if the image dimensions are high the computer throws a segmentation fault.

Algorithm 5 hysteresis(image, label)

```
m, m ← image.shape
visited ← zeros.shapeIike(image) array
for i ← 0 : M do
    for j ← 0 : N do
        if (visited[i][j] == False) and (label[i][j] == 1) then
            visited, image = dfs(i, j)
        end if
    end for
end for
return image
```



(a) Threshold Image (b) Canny Image

Figure 5: Canny Edge Detected Image obtained from Threshold Image using Edge Tracking (Hysteresis)

Algorithm 6 dfs(i, j), visited, image -global

```
M,N ← image.shape
visited[i][j] = True
image[i][j] = 1
dx = [−1,−1,−1,0,0,1,1,1]
dy = [−1,0,1,−1,1,−1,0,1]
for k ← 0 : len(dx) do
    if i >= 0 and j >= 0 and i < M and j < N and visited[i][j] == false and (label[i, j] == 1 or label[i, j] == 0) then
        dfs(i, j)
    end if
end for
```

3 Inferences

- For the Toy image, all edges are detected perfectly by the algorithm.
- Sobel's filter direction has a key role in the output.
- Less threshold value detects a lot of edges but it can give non edges as edges too.
- High threshold value guarantees that we are getting an edge but a lot of edges are missed.
- the Canny edge detector is a state-of-the-art edge detector.

4 Walkthrough of the Canny Edge Detector Algorithm

There are definitely some places where for efficient things can be used. Hence inbuilt functions can perform better than our algorithm, implementation.

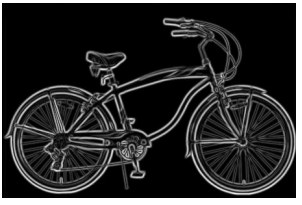
Have a look at the output. The extra lines of code for generating output have been removed and only one image has been shown in the code.

5 Insights and Takeaways

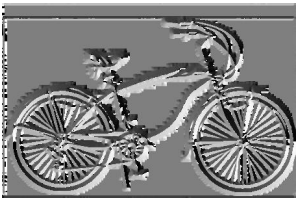
- A big realization of why in-built functions are recommended to use
- In-built are way too much complex and faster and always gives a better output
- My major takeaway was to understand that difficult looking algorithm can actually be simple if understanding is proper
- Canny Edge detector (an algorithm developed in 1986) being used in the current generation, is a sign that sometimes simpler and less complex things can give better output then many complex algorithms

(a) Original Image

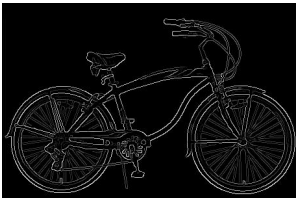
(b) Luminence Image



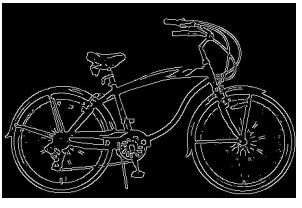
(c) Gradient Magnitude Image



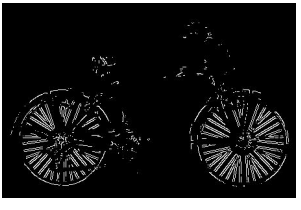
(d) Gradient Direction Image



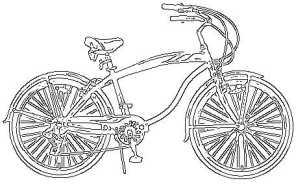
(e) Thinned Edge Image



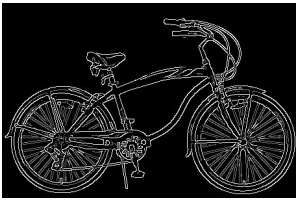
(f) Strong Edge Image



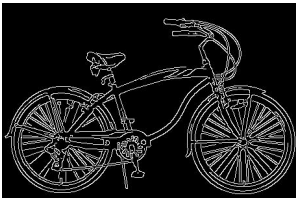
(g) Weak Edge Image



(h) Definitely Not Edge Image

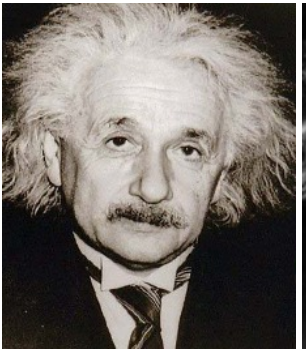


(i) Threshold Image

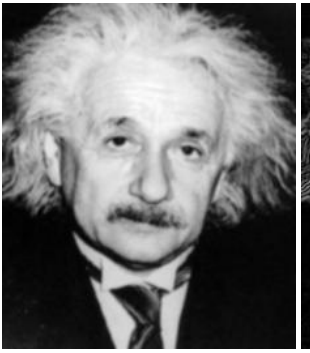


(j) Canny Edge Detected Image

Figure 6: All Intermediate and Final Images through Canny Edge Detector of Bicycle, Tlow = 0.1 Thigh = 0.25



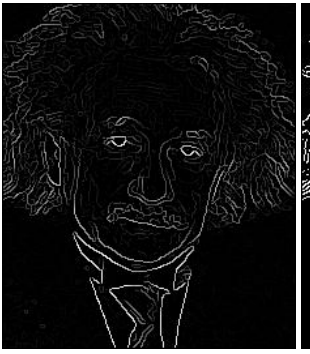
(a) Original Image



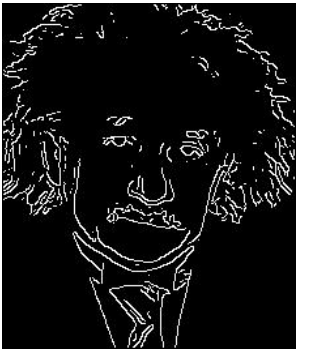
(b) Luminence Image

(c) Gradient Magnitude Image

(d) Gradient Direction Image

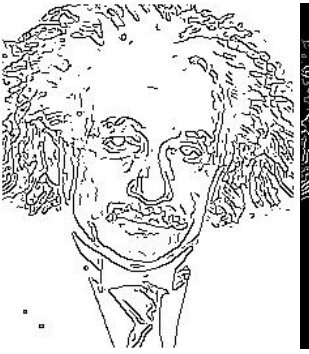


(e) Thinned Edge Image

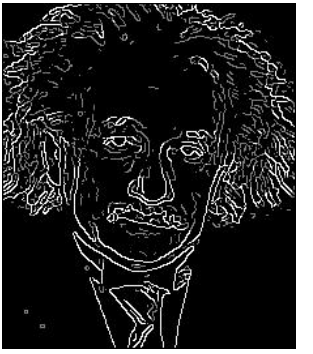


(f) Strong Edge Image

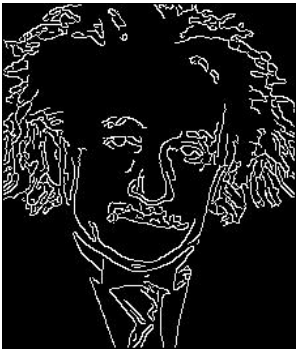
(g) Weak Edge Image



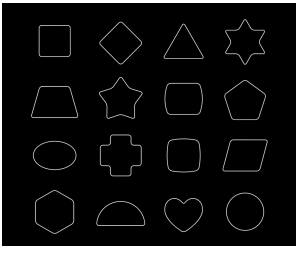
(h) Definitely Not Edge Image



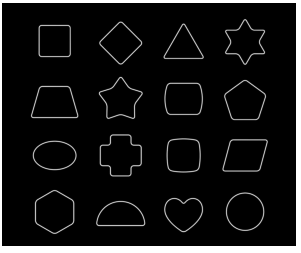
(i) Threshold Image



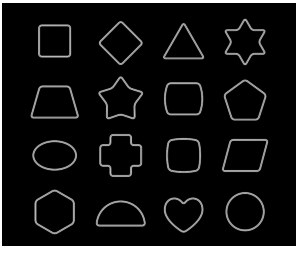
(j) Canny Edge Detected Im-



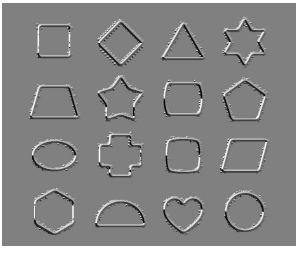
(a) Original Image



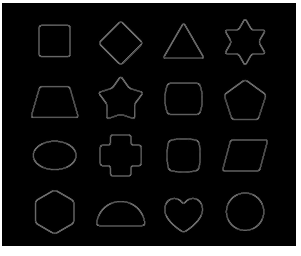
(b) Luminence Image



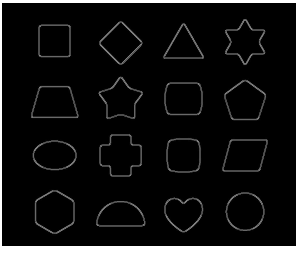
(c) Gradient Magnitude Image



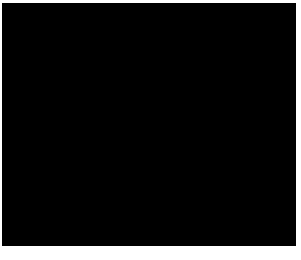
(d) Gradient Direction Image



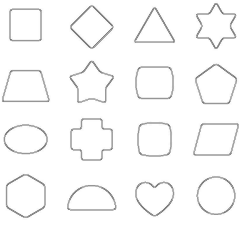
(e) Thinned Edge Image



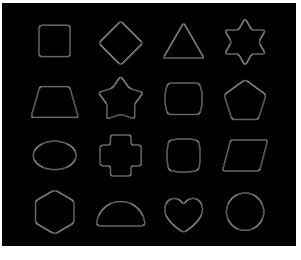
(f) Strong Edge Image



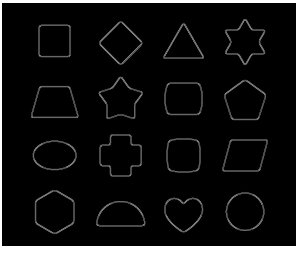
(g) Weak Edge Image



(h) Definitely Not Edge Image

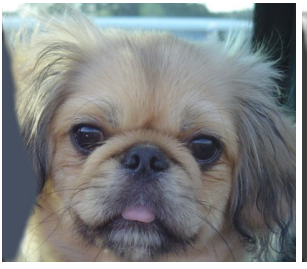


(i) Threshold Image



(j) Canny Edge Detected Image

Figure 8: All Intermediate and Final Images through Canny Edge Detector of Toy, Tlow = 0.1 Thigh = 0.25



(a) Original Image

(b) Luminence Image

(c) Gradient Magnitude Image

(d) Gradient Direction Image

(e) Thinned Edge Image

(f) Strong Edge Image

(g) Weak Edge Image

(h) Definitely Not Edge Image

(i) Threshold Image

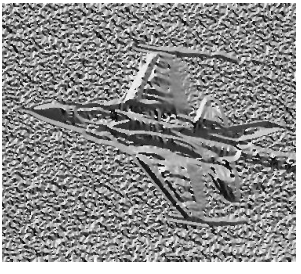
(j) Canny Edge Detected Image

Figure 9: All Intermediate and Final Images through Canny Edge Detector of Dog, Tlow = 0.1 Thigh = 0.25

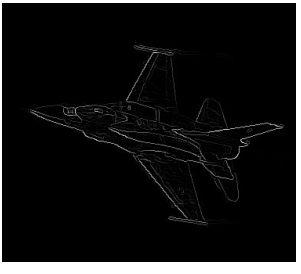
(a) Original Image

(b) Luminence Image

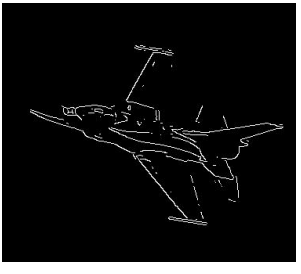
(c) Gradient Magnitude Image



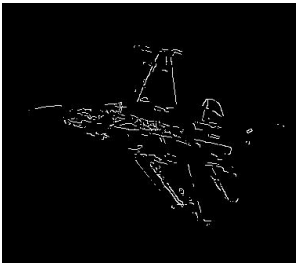
(d) Gradient Direction Image



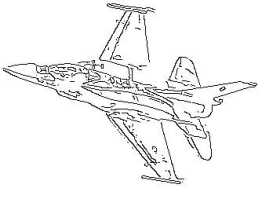
(e) Thinned Edge Image



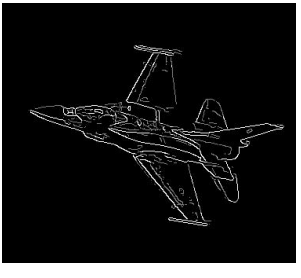
(f) Strong Edge Image



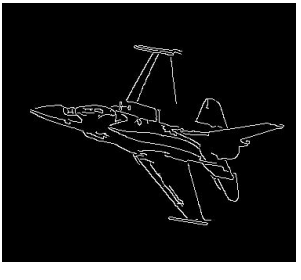
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image



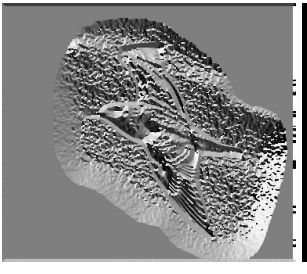
(j) Canny Edge Detected Image

Figure 10: All Intermediate and Final Images through Canny Edge Detector of Plane, Tlow = 0.1 Thigh = 0.25

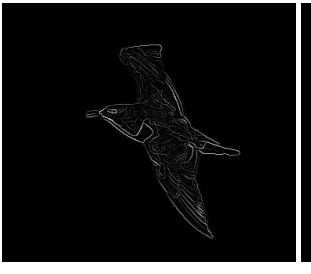
(a) Original Image

(b) Luminence Image

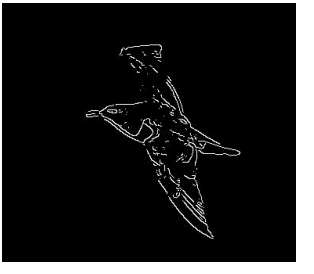
(c) Gradient Magnitude Image



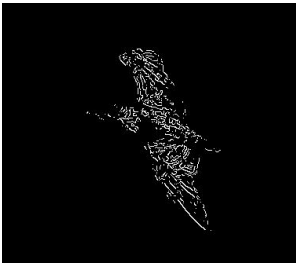
(d) Gradient Direction Image



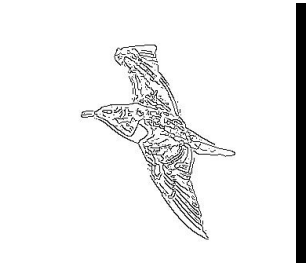
(e) Thinned Edge Image



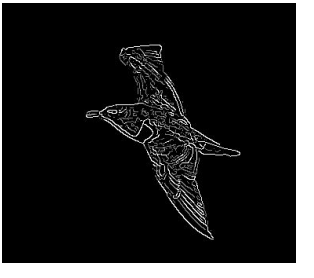
(f) Strong Edge Image



(g) Weak Edge Image



(h) Definitely Not Edge Image



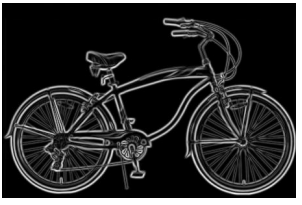
(i) Threshold Image

(j) Canny Edge Detected Image

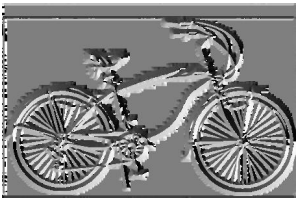
Figure 11: All Intermediate and Final Images through Canny Edge Detector of Bird, Tlow = 0.1 Thigh = 0.25

(a) Original Image

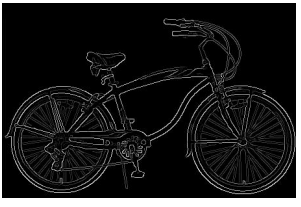
(b) Luminence Image



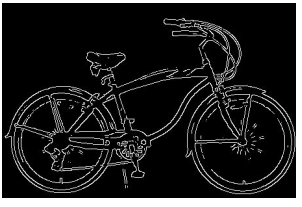
(c) Gradient Magnitude Image



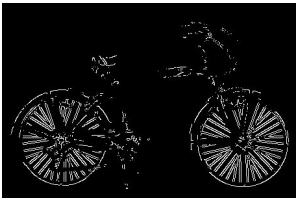
(d) Gradient Direction Image



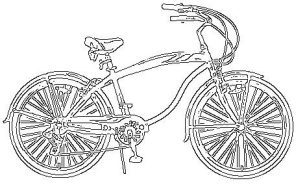
(e) Thinned Edge Image



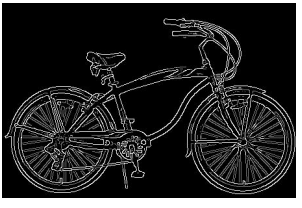
(f) Strong Edge Image



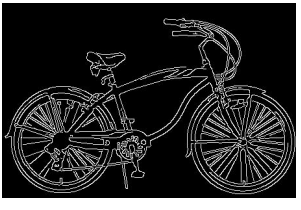
(g) Weak Edge Image



(h) Definitely Not Edge Image

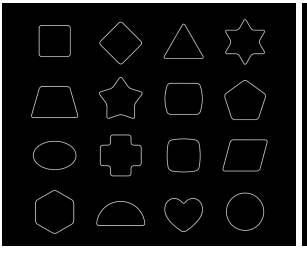


(i) Threshold Image

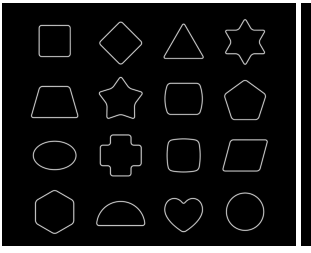


(j) Canny Edge Detected Image

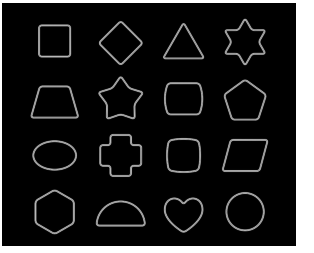
Figure 12: All Intermediate and Final Images through Canny Edge Detector of Bicycle, Tlow = 0.1 Thigh = 0.30



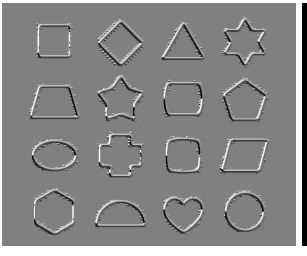
(a) Original Image



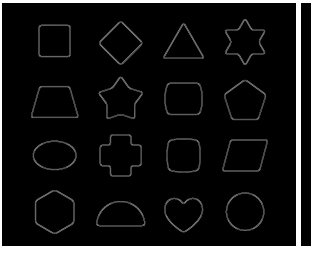
(b) Luminence Image



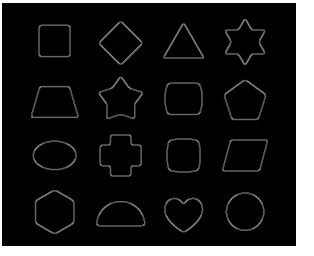
(c) Gradient Magnitude Image



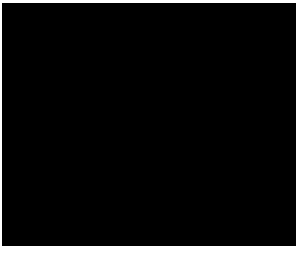
(d) Gradient Direction Image



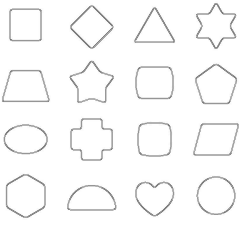
(e) Thinned Edge Image



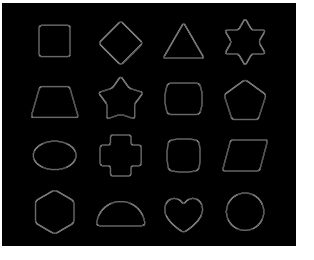
(f) Strong Edge Image



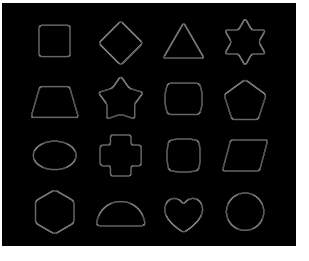
(g) Weak Edge Image



(h) Definitely Not Edge Image

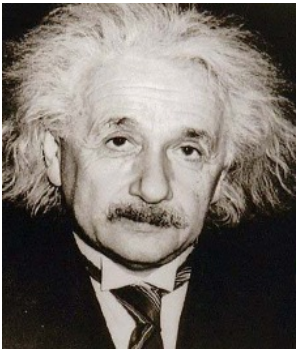


(i) Threshold Image

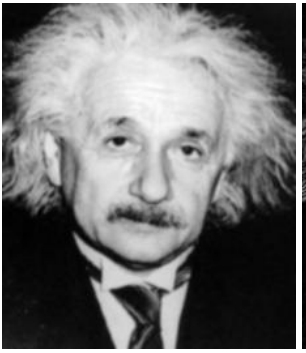


(j) Canny Edge Detected Image

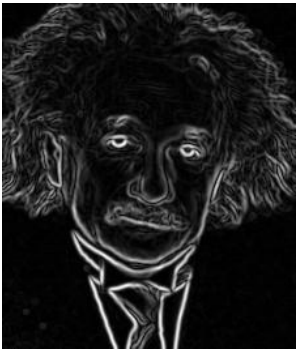
Figure 13: All Intermediate and Final Images through Canny Edge Detector of Toy, Tlow = 0.1 Thigh = 0.30



(a) Original Image

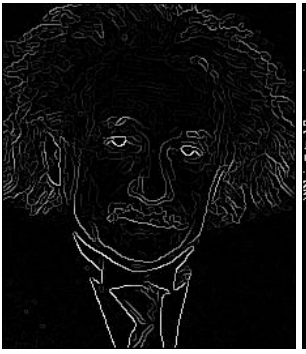


(b) Luminence Image

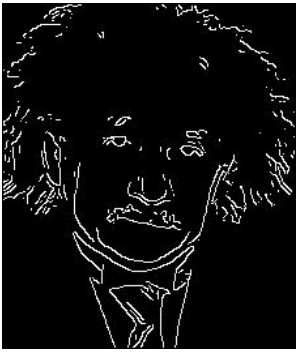


(c) Gradient Magnitude Image

(d) Gradient Direction Image

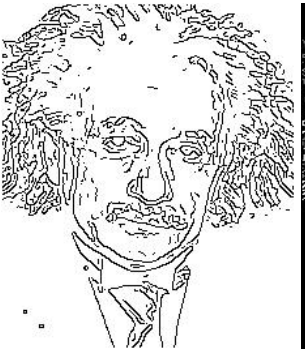


(e) Thinned Edge Image

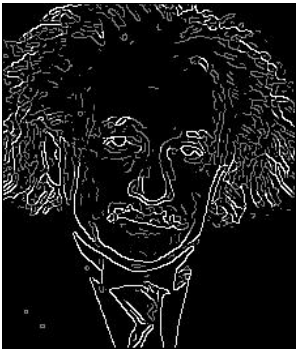


(f) Strong Edge Image

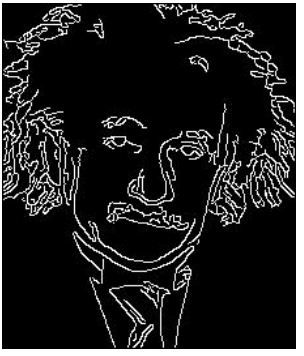
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image

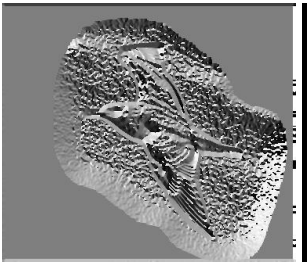


(j) Canny Edge Detected Im-

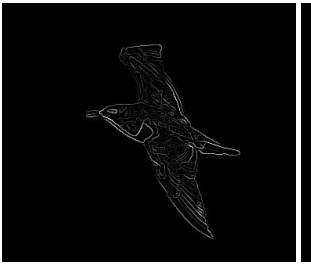
(a) Original Image

(b) Luminence Image

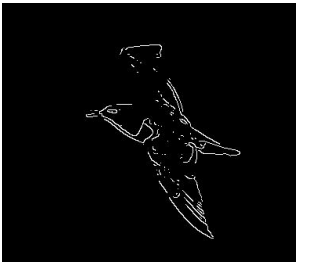
(c) Gradient Magnitude Image



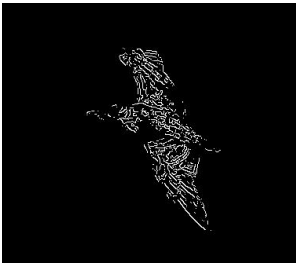
(d) Gradient Direction Image



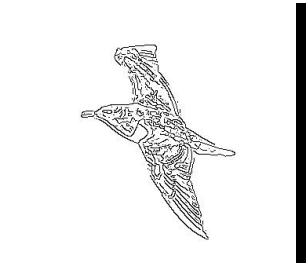
(e) Thinned Edge Image



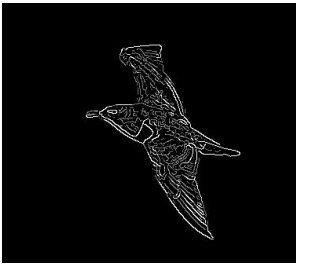
(f) Strong Edge Image



(g) Weak Edge Image



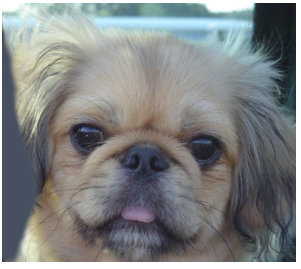
(h) Definitely Not Edge Image



(i) Threshold Image

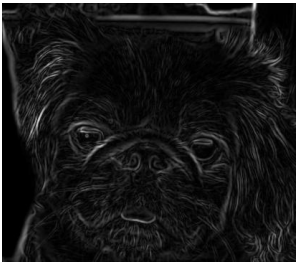
(j) Canny Edge Detected Image

Figure 15: All Intermediate and Final Images through Canny Edge Detector of Bird, Tlow = 0.1 Thigh = 0.30



(a) Original Image

(b) Luminence Image



(c) Gradient Magnitude Image

(d) Gradient Direction Image

(e) Thinned Edge Image

(f) Strong Edge Image

(g) Weak Edge Image

(h) Definitely Not Edge Image

(i) Threshold Image

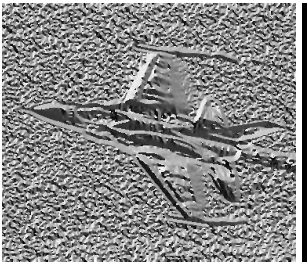
(j) Canny Edge Detected Image

Figure 16: All Intermediate and Final Images through Canny Edge Detector of Dog, Tlow = 0.1 Thigh = 0.30

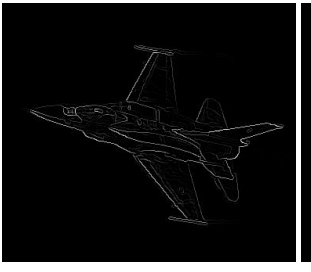
(a) Original Image

(b) Luminence Image

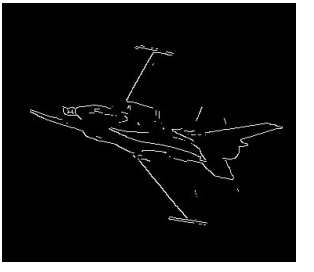
(c) Gradient Magnitude Image



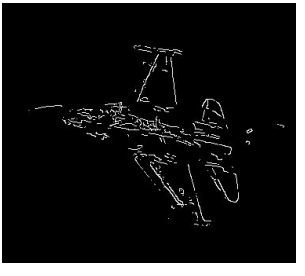
(d) Gradient Direction Image



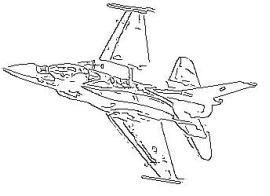
(e) Thinned Edge Image



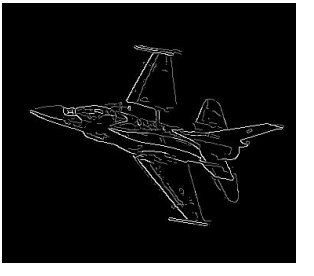
(f) Strong Edge Image



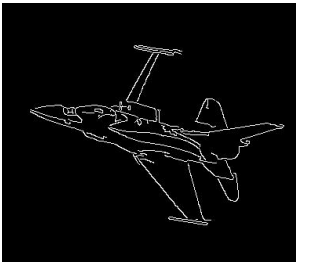
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image

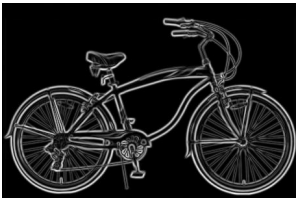


(j) Canny Edge Detected Image

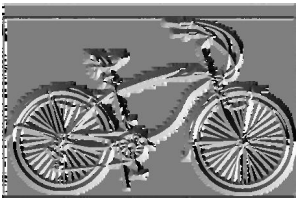
Figure 17: All Intermediate and Final Images through Canny Edge Detector of Plane, Tlow = 0.1 Thigh = 0.30

(a) Original Image

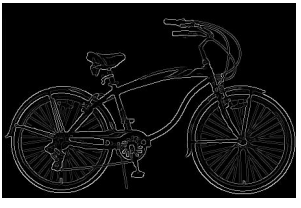
(b) Luminence Image



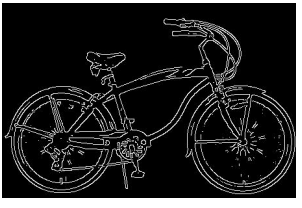
(c) Gradient Magnitude Image



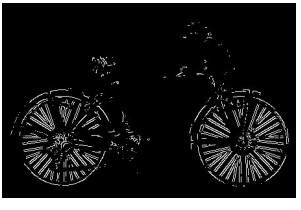
(d) Gradient Direction Image



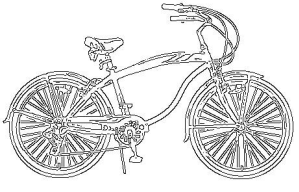
(e) Thinned Edge Image



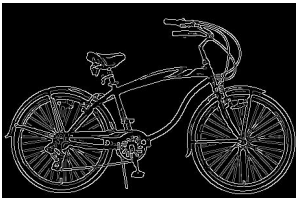
(f) Strong Edge Image



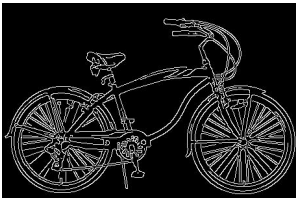
(g) Weak Edge Image



(h) Definitely Not Edge Image

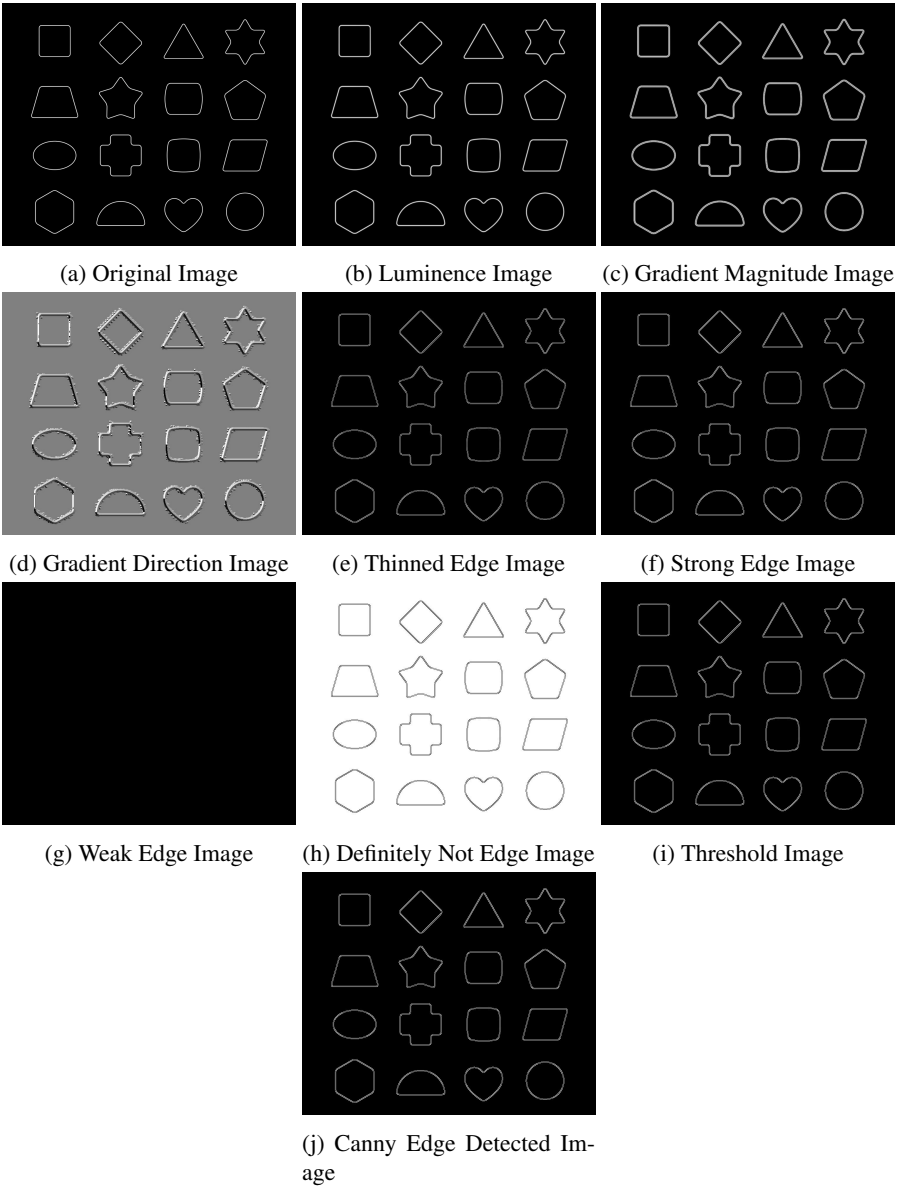


(i) Threshold Image



(j) Canny Edge Detected Image

Figure 18: All Intermediate and Final Images through Canny Edge Detector of Bicycle, Tlow = 0.07 Thigh = 0.25



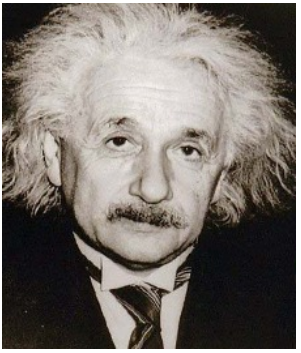
(a) Original Image (b) Luminence Image (c) Gradient Magnitude Image

(d) Gradient Direction Image (e) Thinned Edge Image (f) Strong Edge Image

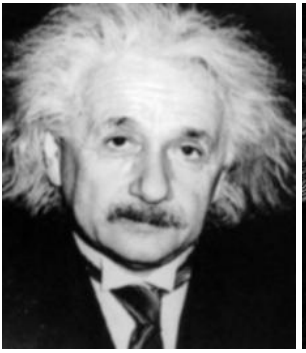
(g) Weak Edge Image (h) Definitely Not Edge Image (i) Threshold Image

(j) Canny Edge Detected Image

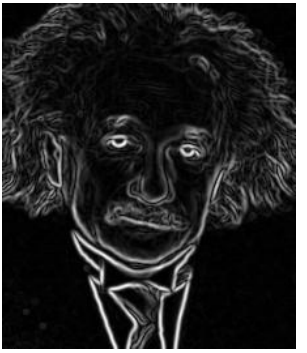
Figure 19: All Intermediate and Final Images through Canny Edge Detector of Toy, Tlow = 0.07 Thigh = 0.25



(a) Original Image

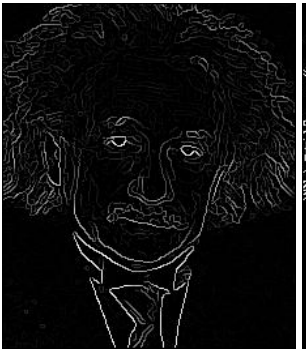


(b) Luminence Image

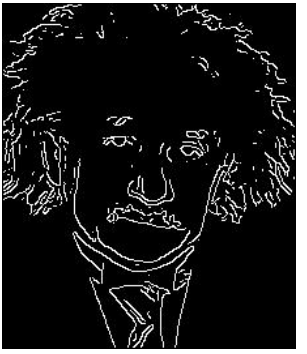


(c) Gradient Magnitude Image

(d) Gradient Direction Image

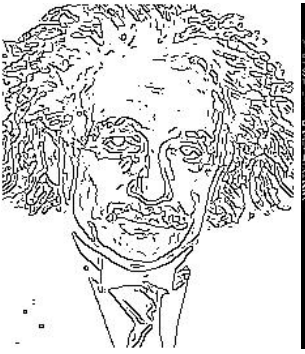


(e) Thinned Edge Image

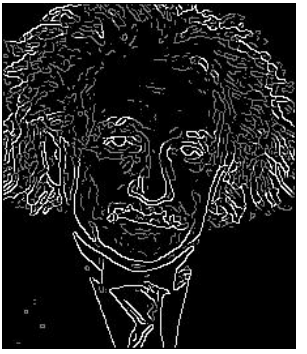


(f) Strong Edge Image

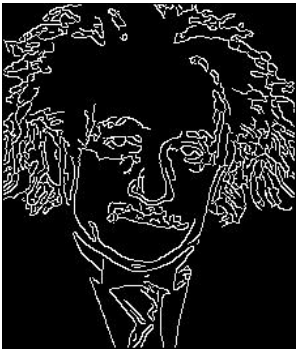
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image

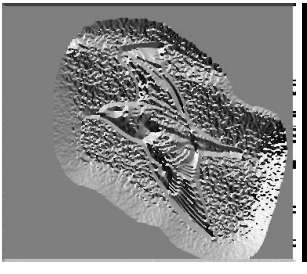


(j) Canny Edge Detected Im-

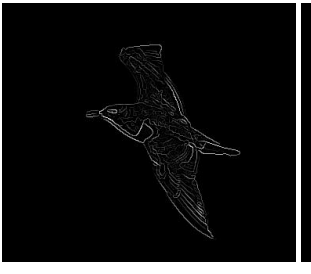
(a) Original Image

(b) Luminence Image

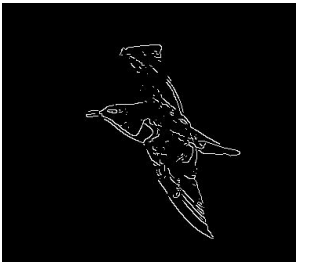
(c) Gradient Magnitude Image



(d) Gradient Direction Image



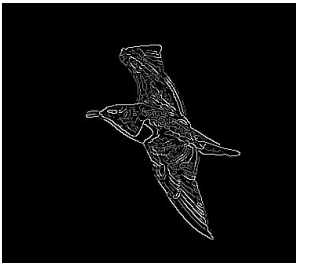
(e) Thinned Edge Image



(f) Strong Edge Image

(g) Weak Edge Image

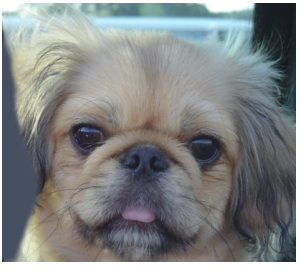
(h) Definitely Not Edge Image



(i) Threshold Image

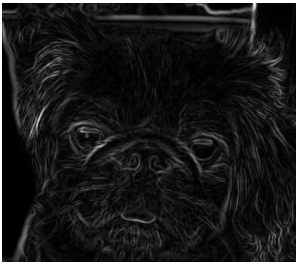
(j) Canny Edge Detected Image

Figure 21: All Intermediate and Final Images through Canny Edge Detector of Bird, Tlow = 0.07 Thigh = 0.25



(a) Original Image

(b) Luminence Image



(c) Gradient Magnitude Image

(d) Gradient Direction Image

(e) Thinned Edge Image

(f) Strong Edge Image

(g) Weak Edge Image

(h) Definitely Not Edge Image

(i) Threshold Image

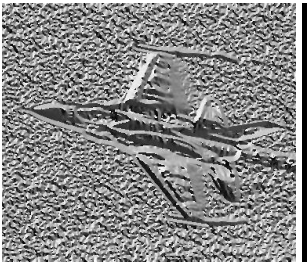
(j) Canny Edge Detected Image

Figure 22: All Intermediate and Final Images through Canny Edge Detector of Dog, Tlow = 0.07 Thigh = 0.25

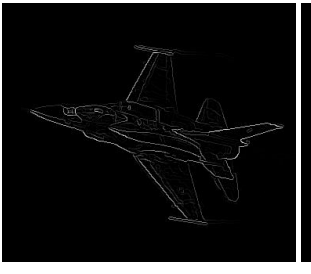
(a) Original Image

(b) Luminence Image

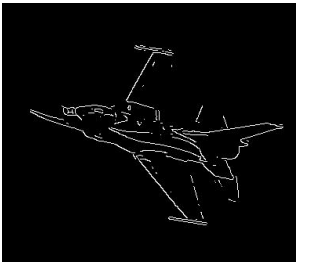
(c) Gradient Magnitude Image



(d) Gradient Direction Image

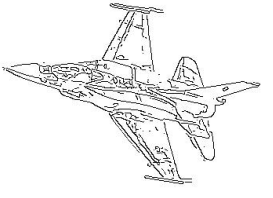


(e) Thinned Edge Image

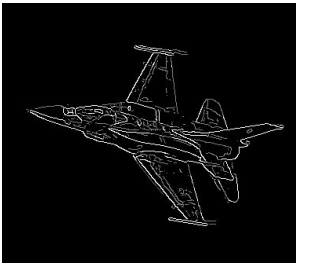


(f) Strong Edge Image

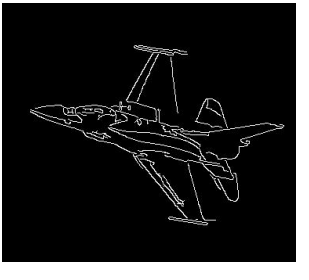
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image



(j) Canny Edge Detected Image

Figure 23: All Intermediate and Final Images through Canny Edge Detector of Plane, Tlow = 0.07 Thigh = 0.25