CANNY EDGE DETECTOR

Vishal Yadav
2018csb1128
2018csb1128@iitrpr.ac.in

Indian Institute of Technology
Ropar, 140001
Punjab, India

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[1] uses a multi-stage algorithm to detect a wide range of edges in images. It was developed by John F. Canny in 1986. It is a multi-stage algorithm and we will go through each stages.

1 Introduction

The Canny edge detector is an edge detection operator that uses a multi-stage algorithm to detect a wide range of edges in images. It was developed by John F. Canny in 1986. Canny has found that the requirements for the application of edge detection on diverse vision systems are relatively similar. Thus, an edge detection solution to address these requirements can be implemented in a wide range of situations. 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
- Gradient calculation
- Non-maximal Suppression
- Double Thresholding
- Edge Trecking by Hysteresis

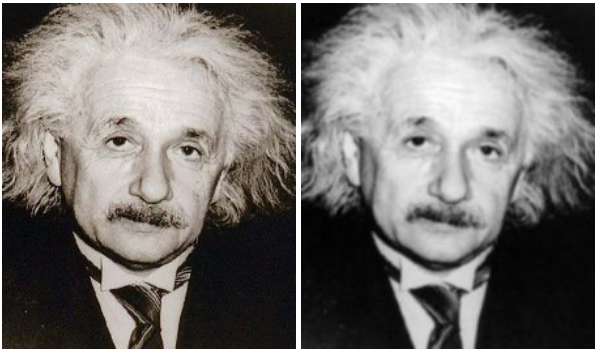
© 2020. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

2.1 Noise Reduction

Since the mathematics involved behind the scene are mainly based on derivatives, edge detection results are highly sensitive to image noise, first step is to remove the noise in the image with a Gaussian filter. We can apply Gaussian Blur (with Kernel of 3x3, 5x5, 7x7 etc.) to smooth the image. Since all edge detection results are easily affected by the noise in the image, it is essential to filter out the noise to prevent false detection caused by it. To smooth the image, a Gaussian filter kernel is convolved with the image. This step will slightly smooth the image to reduce the effects of obvious noise on the edge detector.

Algorithm 1 convertToLuminenceImageAndReduceNoise(img)

1: $R_channel \leftarrow img[:,:,0]$
2: $G_channel \leftarrow img[:,:,1]$
3: $B_channel \leftarrow img[:,:,2]$
4: $luminence_image \leftarrow (0.2989 * R_Channel) + (0.5870 * G_Channel) + (0.1140 * B_Channel)$
5: $luminence_image \leftarrow convolve(luminence_image, gaussian_kernel(kernel_size = 5, sigma = 1))$
6: **return** luminence_image



(a) Original Image (b) Luminence Image

Figure 1: Original Image and Luminence Image of Einstein

2.2 Smoothed Gradient Calculation

The Gradient calculation step detects the edge intensity and direction by calculating the gradient of the image using edge detection operators. Edges correspond to a change of pixels' intensity. Smoothened image is then filtered with a Sobel kernel in both horizontal and vertical direction to get first derivative in horizontal direction (G(x)) and vertical direction (G(y)). From these two images, we can find edge gradient and direction for each pixel. In our example, we have applied the Sobel Filter (approximated Gaussian derivative) in both directions and then calculated Gradient Magnitude and Gradient Direction.

Note: The result is almost the expected one, but we can see that some of the edges are thick and others are thin. Non-Max Suppression step will help us mitigate the thick ones.

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

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

Moreover, the gradient intensity level is between 0 and 255 which is not uniform. The edges on the final result should have the same intensity (i-e. white pixel = 255).

Algorithm 2 get_img_grad(img)

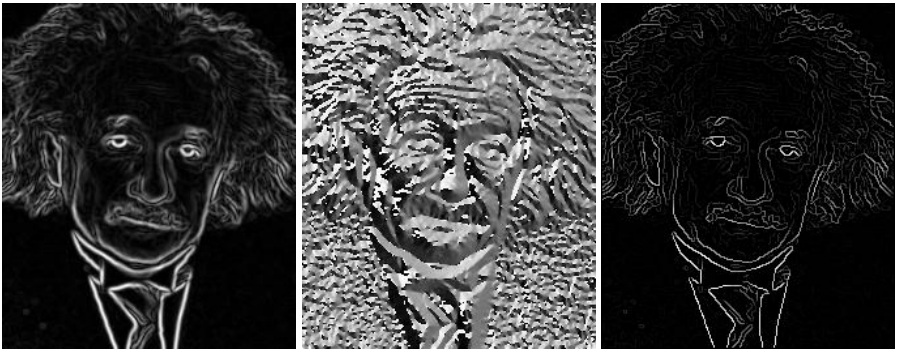
$sobel_x \leftarrow array[[-1,0,1],[-2,0,2],[-1,0,1]]$
$sobel_y \leftarrow array[[1,2,1],[0,0,0],[-1,-2,-1]]$
$Fx \leftarrow convolve(img, sobel_x)$
$Fy \leftarrow convolve(img, sobel_y)$
$grad_magnitude \leftarrow \sqrt{Fx^2 + Fy^2}$
$grad_magnitude \leftarrow grad_magnitude / grad_magnitude.max()$
$grad_direction \leftarrow tan^{-1}(Fy/Fx)$
return grad_magnitude, grad_direction

2.3 Non-Maximal Suppression

Ideally, the final image should have thin edges. After getting gradient magnitude and direction, a full scan of image is done to remove any unwanted pixels which may not constitute the edge. For this, at every pixel, pixel is checked if it is a local maximum in its neighborhood in the direction of gradient. For example, let Point A is on the edge (in vertical direction), Gradient direction is normal to the edge and Point B and C are in gradient directions. So, point A is checked with point B and C to see if it forms a local maximum. If so, it is considered for next stage, otherwise, it is suppressed (put to zero).

In our example, we find for each pixel the Direction D* in (0, π/4, π/2, 3π/4) that is closest to the orientation D at that pixel in *gradient_magnitude* image.

The result is the same image with thinner edges. We can however still notice some variation regarding the edges' intensity: some pixels seem to be brighter than others, and we will try to cover this shortcoming with the two final steps.



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

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

2.4 Double Thresholding

The edge-pixels remaining after the non-maximum suppression step are (still) marked with their strength pixel-by-pixel. Many of these will probably be true edges in the image, but some may be caused by noise or color variations for instance due to rough surfaces. The simplest way to discern between these would be to use a threshold, so that only edges stronger that a certain value would be preserved. The Canny edge detection algorithm uses double thresholding.
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.

- Other pixels are considered as non-relevant for the edge.

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.

Algorithm 3 non_maximal_suppression(gm = grad_magnitude, gd = grad_direction)

```
thinned_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_image ← gm[i][j]
        else
            thinned_image ← 0
        end if
    end for
end for
return thinned_image
```

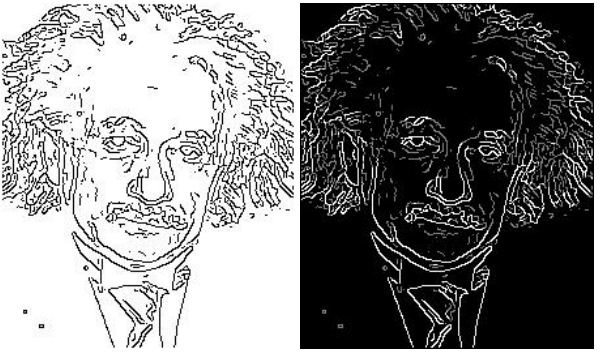
Algorithm 4 thresholdingEdges(image, Tlow, Thigh)

```
threshold_image ← zeros.shape_like(image) array
for i ← 0 : img_length do
    for j ← 0 : img_width do
        if (img[i][j] ≥ Thigh) then
            threshold_image[i][j] ← 1.0
        else if (img[i][j] ≥ Tlow) and (img[i][j] ≤ Thigh) then
            threshold_image[i][j] ← 0.5
        end if
    end for
end for
return threshold_image
```

2.5 Edge Tracking by Hysteresis

This stage decides which are all edges are really edges and which are not. For this, we need two threshold values, minVal and maxVal. Any edges with intensity gradient more than

(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) of Einstein obtained from Thinned Edge Image of Einstein using Double Thresholding

maxVal are sure to be edges and those below minVal are sure to be non-edges, so discarded. Those who lie between these two thresholds are classified edges or non-edges based on their connectivity. If they are connected to “sure-edge” pixels, they are considered to be part of edges. Otherwise, they are also discarded. Then, based on the threshold results, the hysteresis consists of transforming weak pixels into strong ones, if and only if at least one of the pixels around the one being processed is a strong one. We visit pixels in chains starting from the strong pixels. For each strong pixel, we recursively visit the weak pixels that are in the 8 connected neighborhood around the strong pixel, and label those also as strong (and as edge). Also, we label as "not edge" any weak pixels that are not visited by this process. Finally, hysteresis was used as a means of eliminating streaking.

Algorithm 5 hysteresisEdgeTracking(image)

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

(a) Threshold Image (b) Canny Image

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

Algorithm 6 hysteresisEdgeTrackingUtil(i, j, visited, image)

```
M,N ← image.shape
if i < 0 or j < 0 or i ≥ M or j ≥ N or visited[i][j] == True then
    return visited, image
end if
visited[i][j] = True
if image[i][j] == 0 then
    return visited, image
end if
image[i][j] = 1
staticRowN = [−1, −1, −1, 0, 0, 1, 1, 1]
staticColN = [−1, 0, 1, −1, 1, −1, 0, 1]
for k ← 0 : len(RowN) do
    visited, image ← hysteresis_util(i + RowN[k], j + ColN[k], visited, image)
end for
return visited, image
```

3 Observations

We have seen that canny edge detector gives better result. It is less sensitive to noise, adaptive in nature, resolved the problem of streaking, provides good localization and detects sharper edges. It is considered as optimal edge detection technique hence lot of work and improvement on this algorithm has been done and further improvements are possible in future as an improved canny algorithm can detect edges in color image without converting to gray image, improved canny algorithm for automatic extraction of moving object in the image guidance. It finds practical application in Runway Detection and Tracking for Unmanned Aerial Vehicle, in brain MRI image, cable insulation layer measurement, Real-time facial expression recognition, edge detection of river regime, automatic multiple Faces Tracking and Detection. Direction of Sobel Filter can change the output drastically. The greater the T_high threshold is, the less the edges we detect (but definitely an edge). The lesser the T_high threshold is, the more the edges (and possibly many non-edges) are detected. We have used 8 neighborhood connectivity, 4 or 6 neighborhood connectivity could also have been used (but best output comes with 8-way).
Although John Canny's work was done in the early days of computer vision, the Canny edge detector (including its variations) is still a state-of-the-art edge detector. The greater the T_low threshold is, the worser is the output, because it that way we might not get a smooth edge (as many pixels in that edge might be marked as definitely_not_edge because of the high low_threshold). For the Toy image, all edges are detected perfectly by the algorithm. Also there are no weak edges (all are strong).

4 Insights and Takeaways

Canny has shown that the first derivative of the Gaussian closely approximates the operator that optimizes the product of signal-to-noise ratio and localization.

We have used Gaussian operator to blur an image and suppress the noise, it could be seen as a perfect function which is easy to specify. 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.Corners are often missed because the 1D gradient at corners is usually small. This can cause considerable difficulties for line labeling schemes because they rely on having corners and junctions being marked properly.
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. Upon trying a lot of thresholds, Thresholds with best results are T_low = 0.10 and T_high = 0.25.

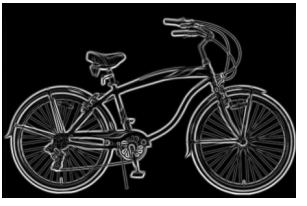
References

[1] John Canny. A computational approach to edge detection ieee transactions on pattern analysis and machine intelligence. NOVEMBER 1985.

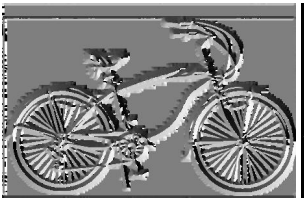
5 Results

(a) Original Image

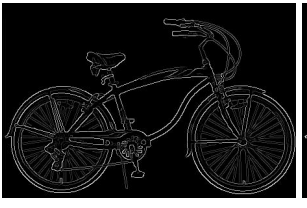
(b) Luminence Image



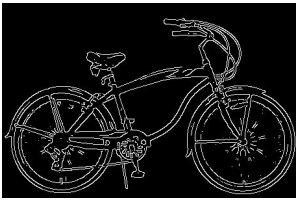
(c) Gradient Magnitude Image



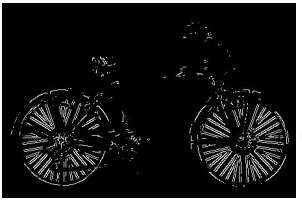
(d) Gradient Direction Image



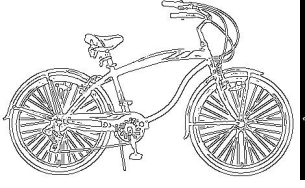
(e) Thinned Edge Image



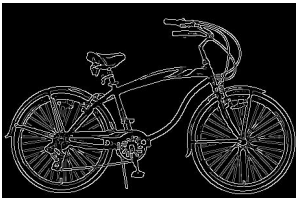
(f) Strong Edge Image



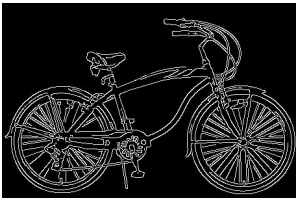
(g) Weak Edge Image



(h) Definitely Not Edge Image

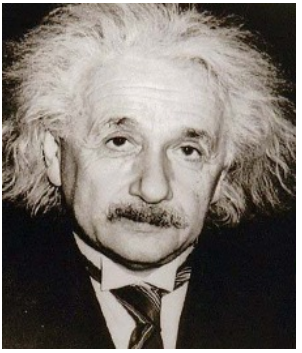


(i) Threshold Image

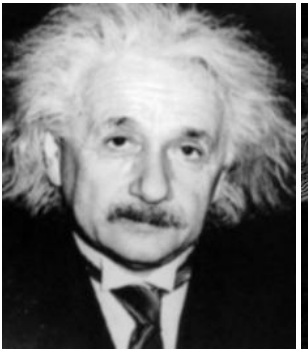


(j) Canny Edge Detected Image

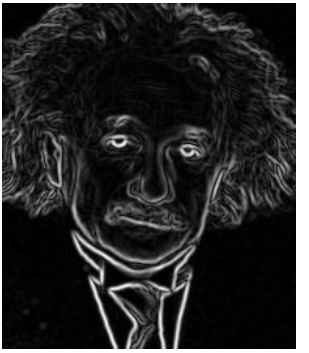
Figure 6: All Intermediate and Final Images through Canny Edge Detector of Bicycle, Tlow = 0.1 Thigh = 0.25



(a) Original Image

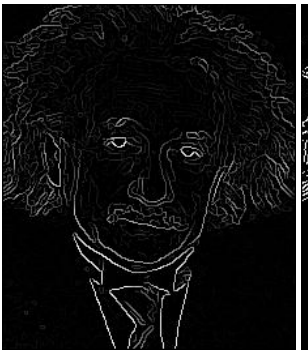


(b) Luminence Image

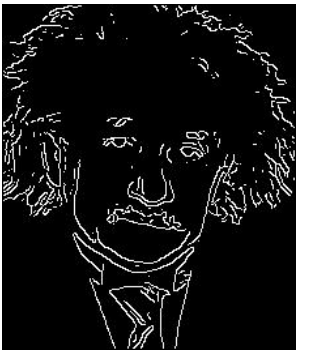


(c) Gradient Magnitude Image

(d) Gradient Direction Image

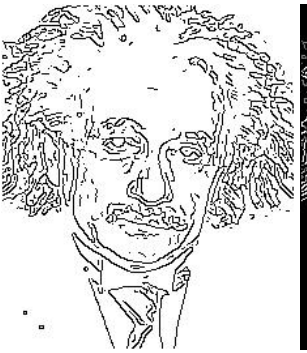


(e) Thinned Edge Image

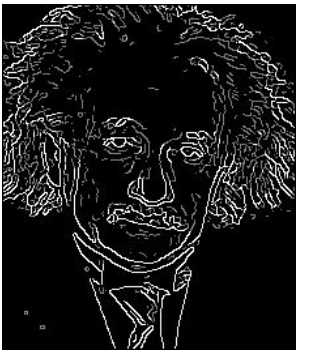


(f) Strong Edge Image

(g) Weak Edge Image



(h) Definitely Not Edge Image



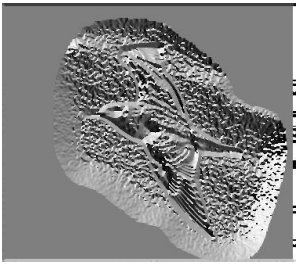
(i) Threshold Image

(j) Canny Edge Detected Im-

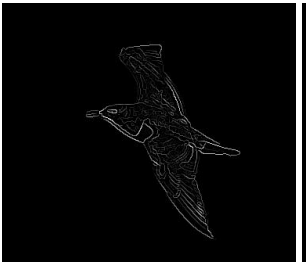
(a) Original Image

(b) Luminence Image

(c) Gradient Magnitude Image



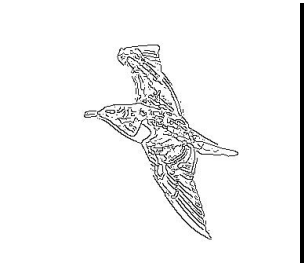
(d) Gradient Direction Image



(e) Thinned Edge Image

(f) Strong Edge Image

(g) Weak Edge Image

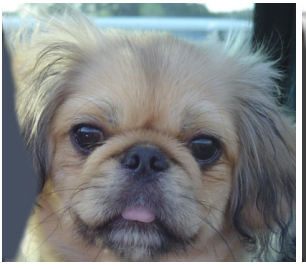


(h) Definitely Not Edge Image

(i) Threshold Image

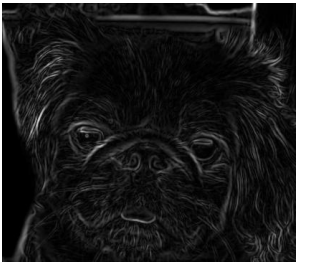
(j) Canny Edge Detected Image

Figure 8: All Intermediate and Final Images through Canny Edge Detector of 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

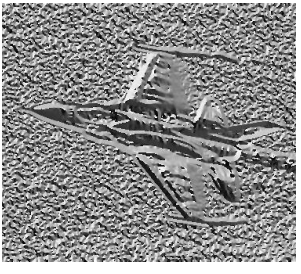
(j) Canny Edge Detected Image

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

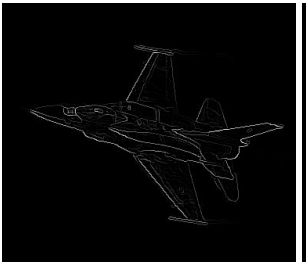
(a) Original Image

(b) Luminence Image

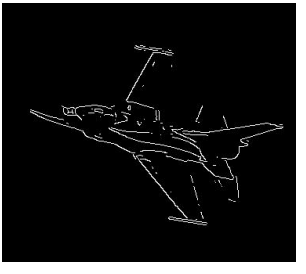
(c) Gradient Magnitude Image



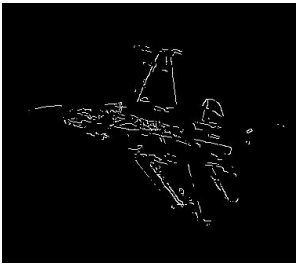
(d) Gradient Direction Image



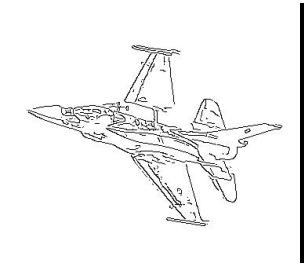
(e) Thinned Edge Image



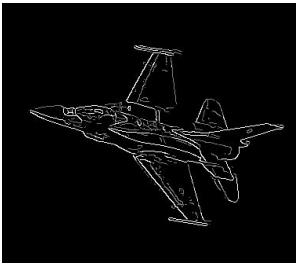
(f) Strong Edge Image



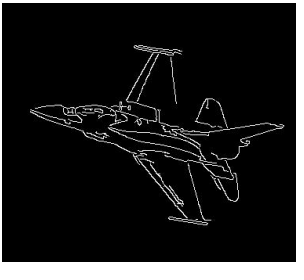
(g) Weak Edge Image



(h) Definitely Not Edge Image

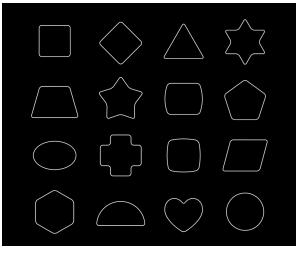


(i) Threshold Image

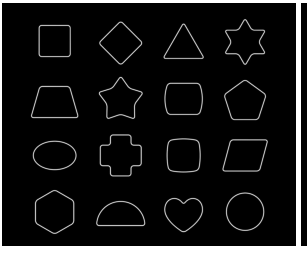


(j) Canny Edge Detected Image

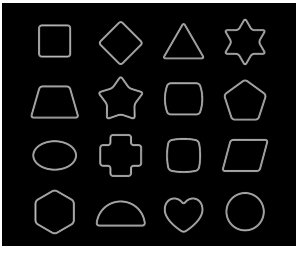
Figure 10: All Intermediate and Final Images through Canny Edge Detector of Plane, Tlow = 0.1 Thigh = 0.25



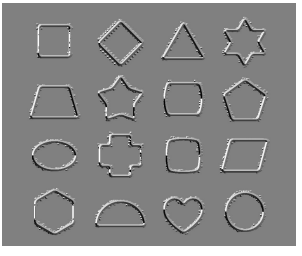
(a) Original Image



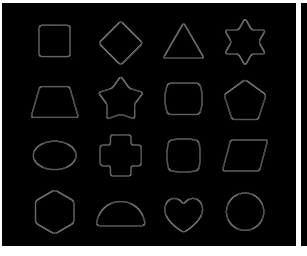
(b) Luminence Image



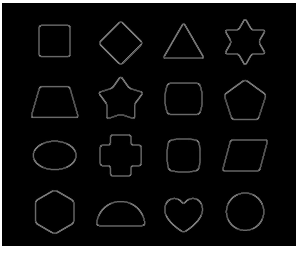
(c) Gradient Magnitude Image



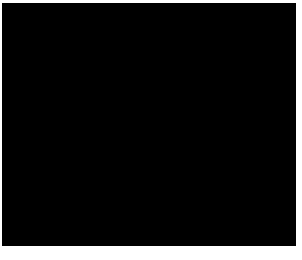
(d) Gradient Direction Image



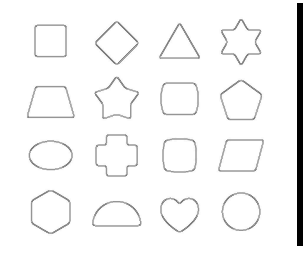
(e) Thinned Edge Image



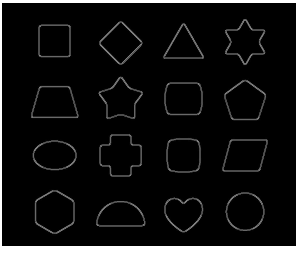
(f) Strong Edge Image



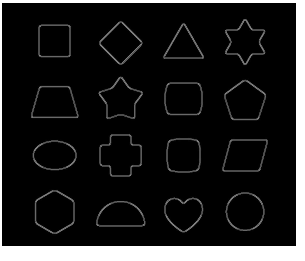
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image

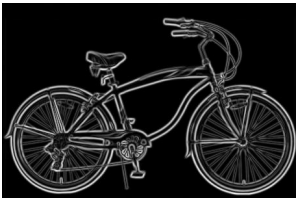


(j) Canny Edge Detected Image

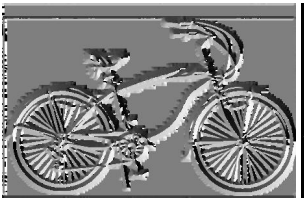
Figure 11: All Intermediate and Final Images through Canny Edge Detector of Toy, Tlow = 0.1 Thigh = 0.25

(a) Original Image

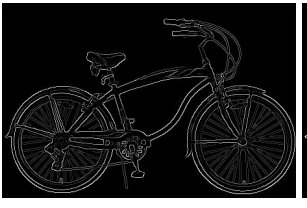
(b) Luminence Image



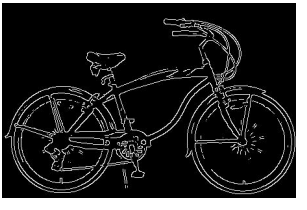
(c) Gradient Magnitude Image



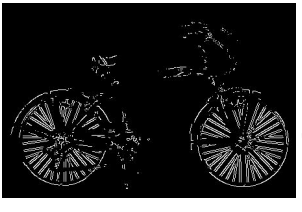
(d) Gradient Direction Image



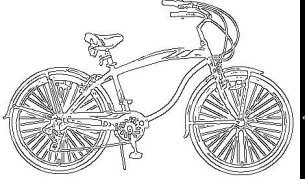
(e) Thinned Edge Image



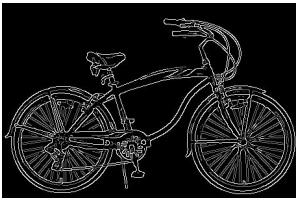
(f) Strong Edge Image



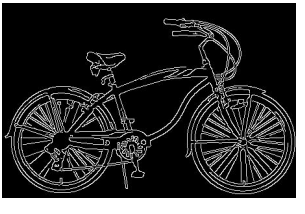
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image



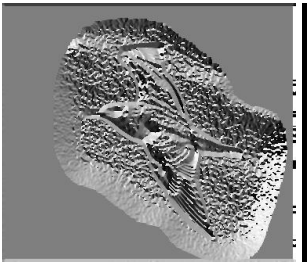
(j) Canny Edge Detected Image

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

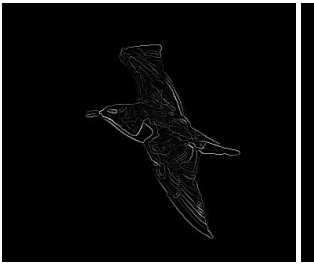
(a) Original Image

(b) Luminence Image

(c) Gradient Magnitude Image



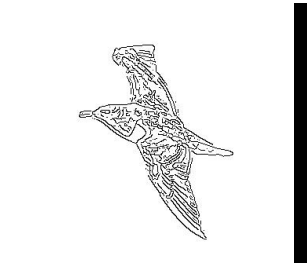
(d) Gradient Direction Image



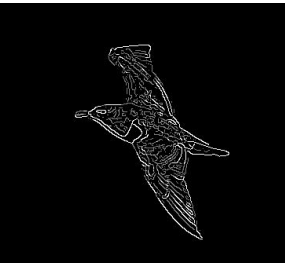
(e) Thinned Edge Image

(f) Strong Edge Image

(g) Weak Edge Image



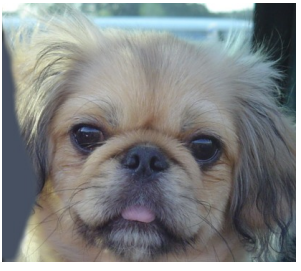
(h) Definitely Not Edge Image



(i) Threshold Image

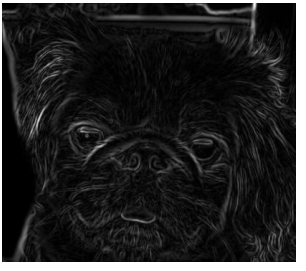
(j) Canny Edge Detected Image

Figure 13: All Intermediate and Final Images through Canny Edge Detector of 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

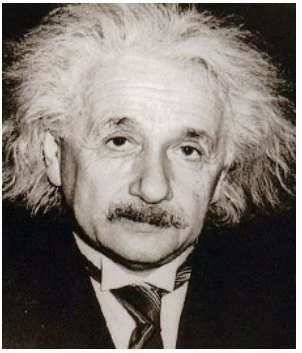
(g) Weak Edge Image

(h) Definitely Not Edge Image

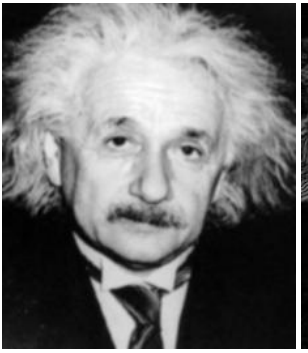
(i) Threshold Image

(j) Canny Edge Detected Image

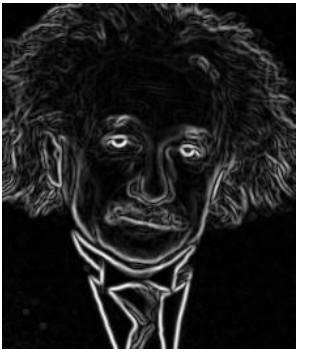
Figure 14: All Intermediate and Final Images through Canny Edge Detector of Dog, Tlow = 0.1 Thigh = 0.30



(a) Original Image

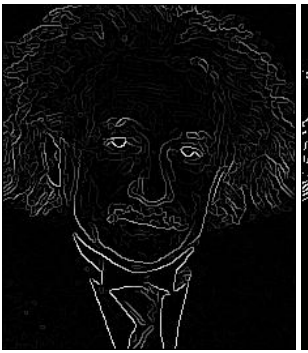


(b) Luminence Image

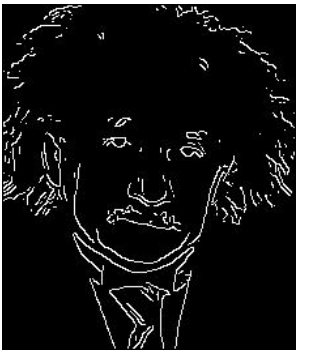


(c) Gradient Magnitude Image

(d) Gradient Direction Image

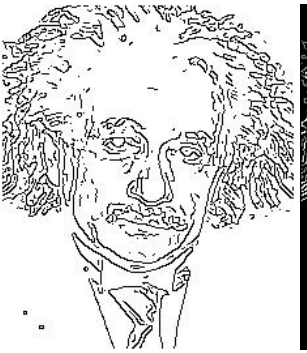


(e) Thinned Edge Image

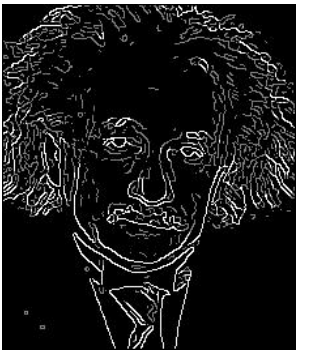


(f) Strong Edge Image

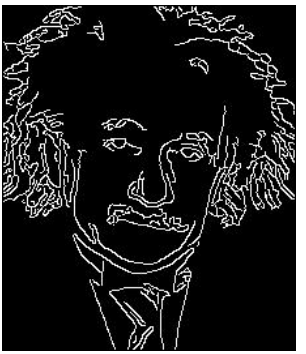
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image

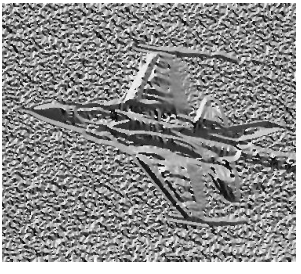


(j) Canny Edge Detected Im-

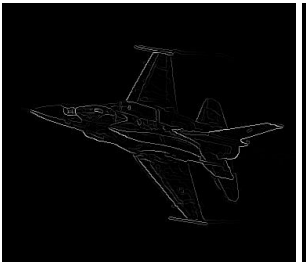
(a) Original Image

(b) Luminence Image

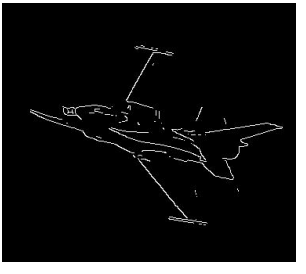
(c) Gradient Magnitude Image



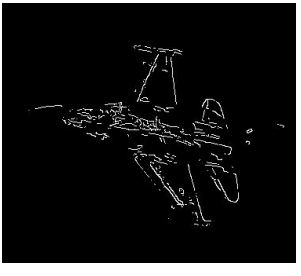
(d) Gradient Direction Image



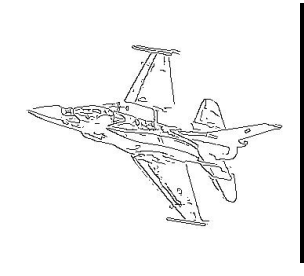
(e) Thinned Edge Image



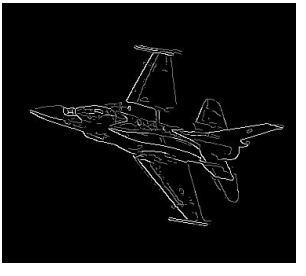
(f) Strong Edge Image



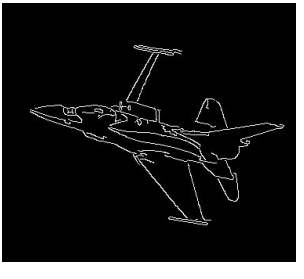
(g) Weak Edge Image



(h) Definitely Not Edge Image

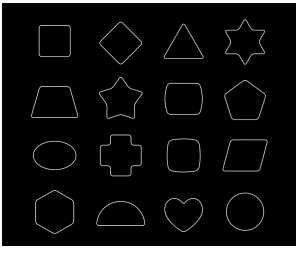


(i) Threshold Image

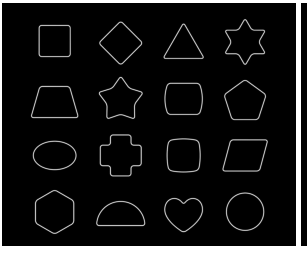


(j) Canny Edge Detected Image

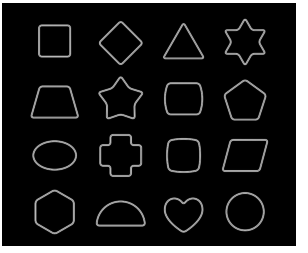
Figure 16: All Intermediate and Final Images through Canny Edge Detector of Plane, Tlow = 0.1 Thigh = 0.30



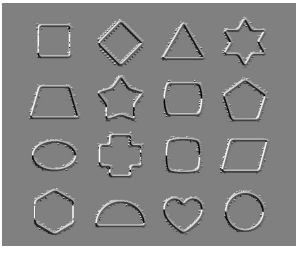
(a) Original Image



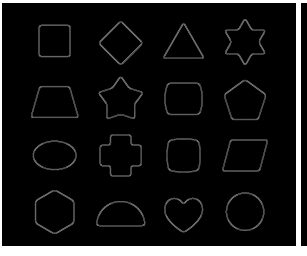
(b) Luminence Image



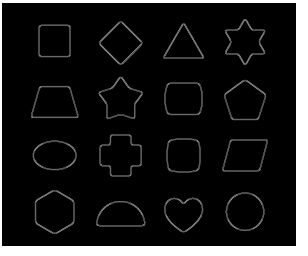
(c) Gradient Magnitude Image



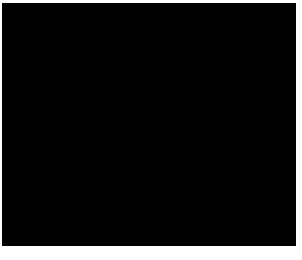
(d) Gradient Direction Image



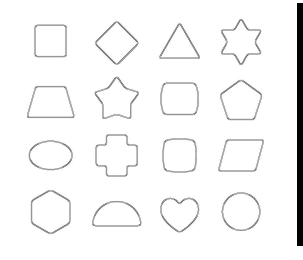
(e) Thinned Edge Image



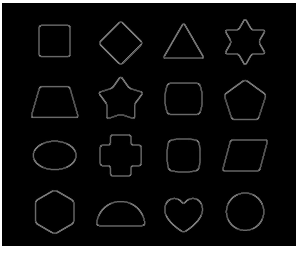
(f) Strong Edge Image



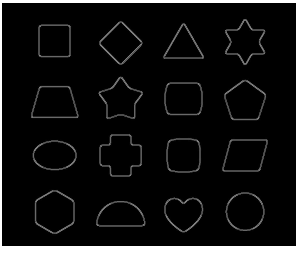
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image

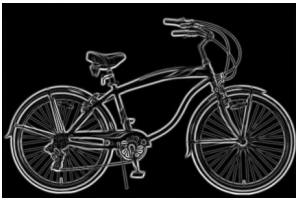


(j) Canny Edge Detected Image

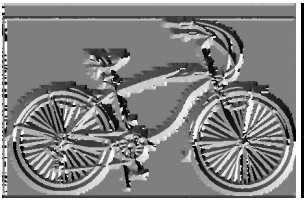
Figure 17: All Intermediate and Final Images through Canny Edge Detector of Toy, Tlow = 0.1 Thigh = 0.30

(a) Original Image

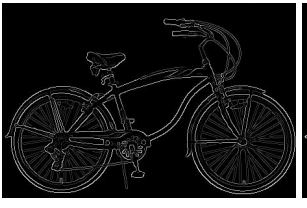
(b) Luminence Image



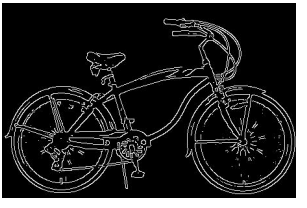
(c) Gradient Magnitude Image



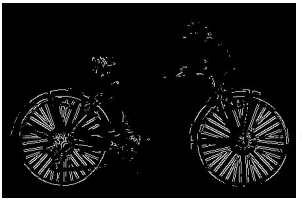
(d) Gradient Direction Image



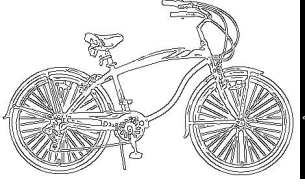
(e) Thinned Edge Image



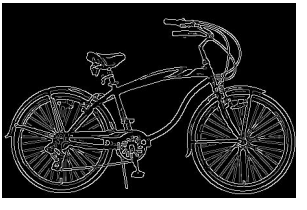
(f) Strong Edge Image



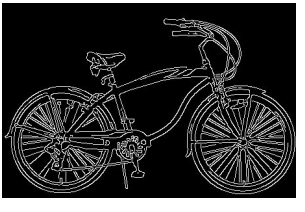
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image



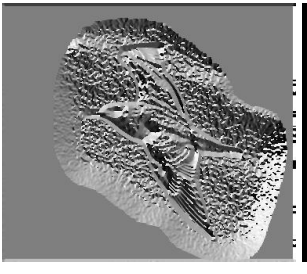
(j) Canny Edge Detected Image

Figure 18: All Intermediate and Final Images through Canny Edge Detector of Bicycle, Tlow = 0.05 Thigh = 0.25

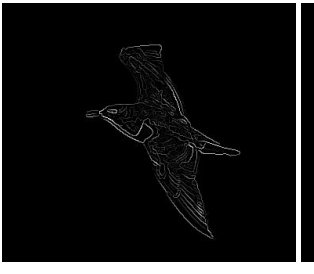
(a) Original Image

(b) Luminence Image

(c) Gradient Magnitude Image



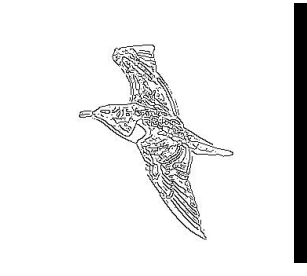
(d) Gradient Direction Image



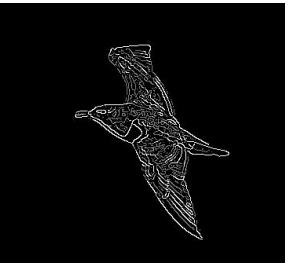
(e) Thinned Edge Image

(f) Strong Edge Image

(g) Weak Edge Image



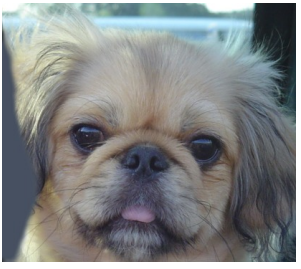
(h) Definitely Not Edge Image



(i) Threshold Image

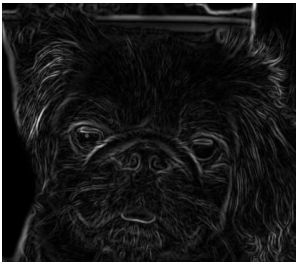
(j) Canny Edge Detected Image

Figure 19: All Intermediate and Final Images through Canny Edge Detector of Bird, Tlow = 0.05 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

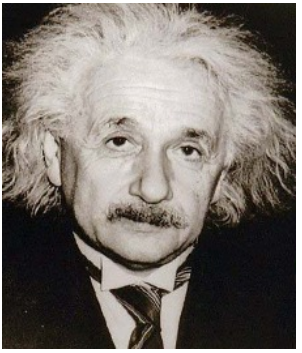
(g) Weak Edge Image

(h) Definitely Not Edge Image

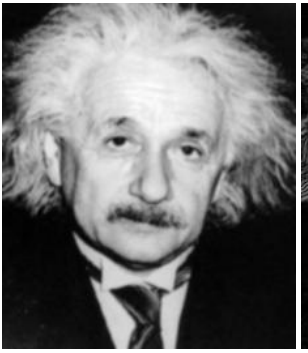
(i) Threshold Image

(j) Canny Edge Detected Image

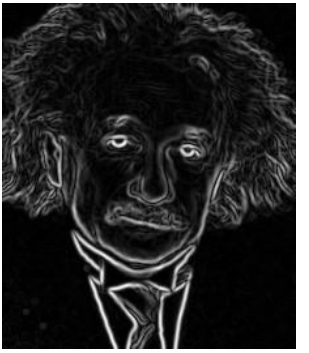
Figure 20: All Intermediate and Final Images through Canny Edge Detector of Dog, Tlow = 0.05 Thigh = 0.25



(a) Original Image

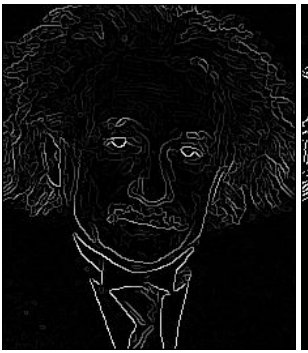


(b) Luminence Image

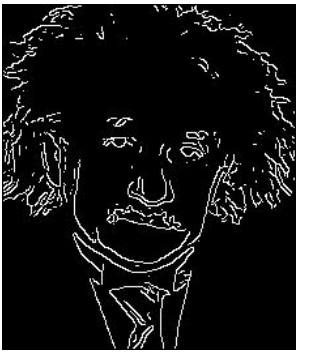


(c) Gradient Magnitude Image

(d) Gradient Direction Image

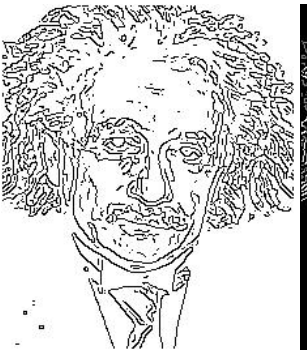


(e) Thinned Edge Image

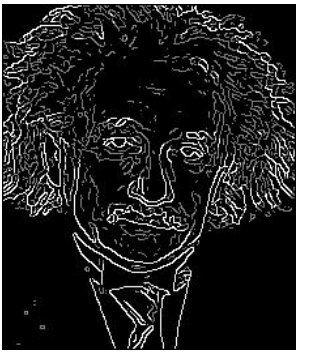


(f) Strong Edge Image

(g) Weak Edge Image



(h) Definitely Not Edge Image



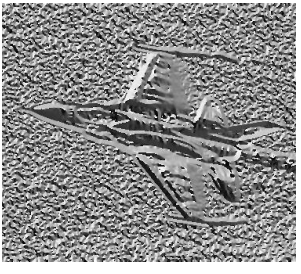
(i) Threshold Image

(j) Canny Edge Detected Im-

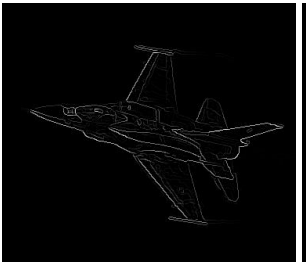
(a) Original Image

(b) Luminence Image

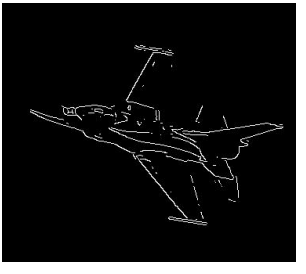
(c) Gradient Magnitude Image



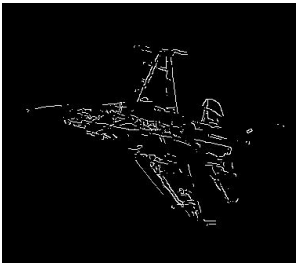
(d) Gradient Direction Image



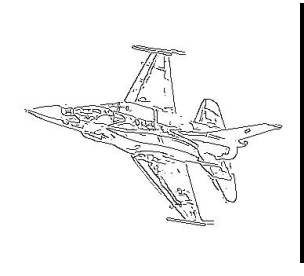
(e) Thinned Edge Image



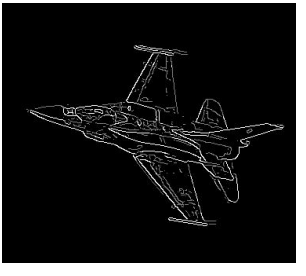
(f) Strong Edge Image



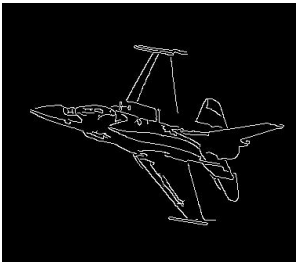
(g) Weak Edge Image



(h) Definitely Not Edge Image

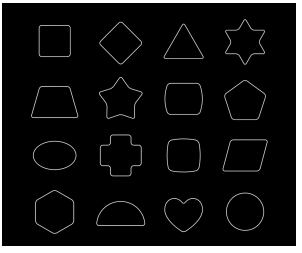


(i) Threshold Image

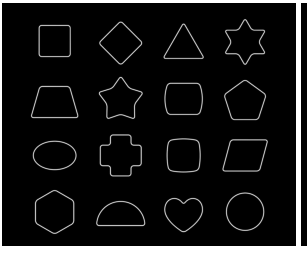


(j) Canny Edge Detected Image

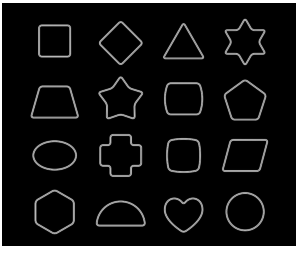
Figure 22: All Intermediate and Final Images through Canny Edge Detector of Plane, Tlow = 0.05 Thigh = 0.25



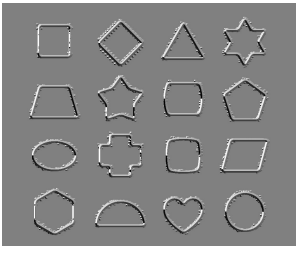
(a) Original Image



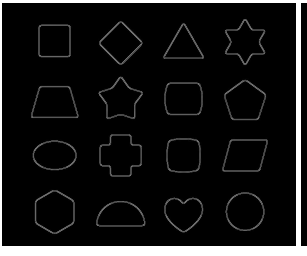
(b) Luminence Image



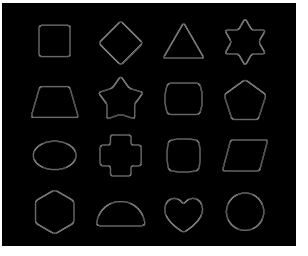
(c) Gradient Magnitude Image



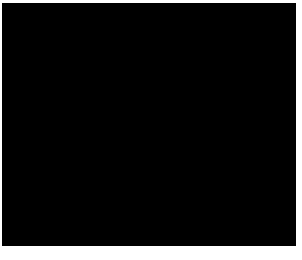
(d) Gradient Direction Image



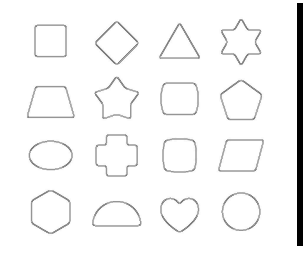
(e) Thinned Edge Image



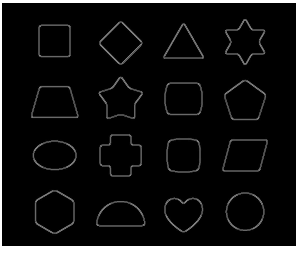
(f) Strong Edge Image



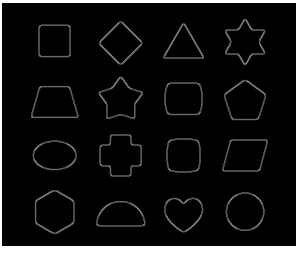
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image

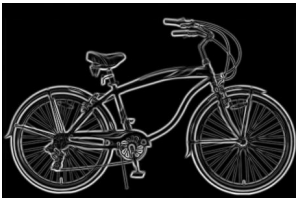


(j) Canny Edge Detected Image

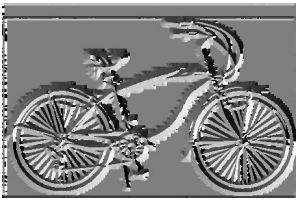
Figure 23: All Intermediate and Final Images through Canny Edge Detector of Toy, Tlow = 0.05 Thigh = 0.25

(a) Original Image

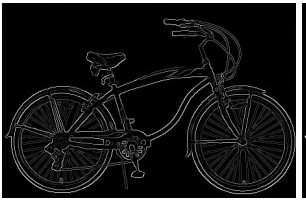
(b) Luminence Image



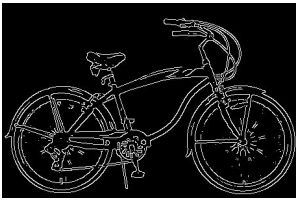
(c) Gradient Magnitude Image



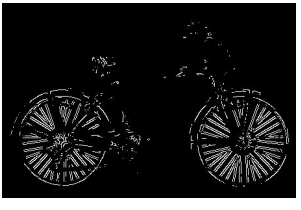
(d) Gradient Direction Image



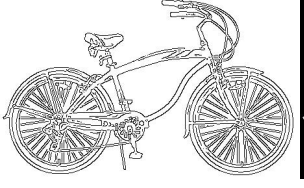
(e) Thinned Edge Image



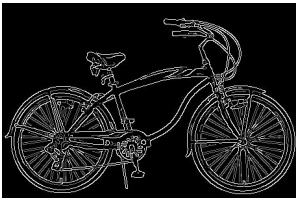
(f) Strong Edge Image



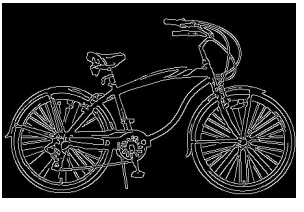
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image



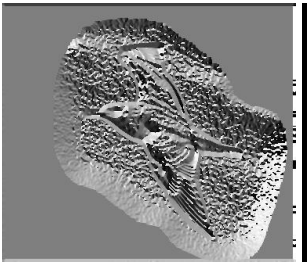
(j) Canny Edge Detected Image

Figure 24: All Intermediate and Final Images through Canny Edge Detector of Bicycle, Tlow = 0.06 Thigh = 0.25

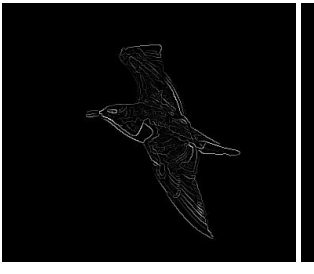
(a) Original Image

(b) Luminence Image

(c) Gradient Magnitude Image



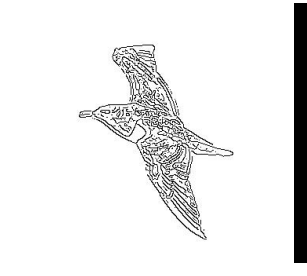
(d) Gradient Direction Image



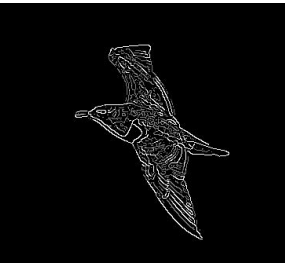
(e) Thinned Edge Image

(f) Strong Edge Image

(g) Weak Edge Image



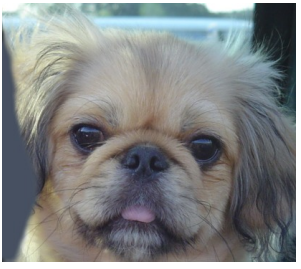
(h) Definitely Not Edge Image



(i) Threshold Image

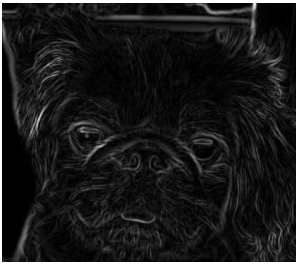
(j) Canny Edge Detected Image

Figure 25: All Intermediate and Final Images through Canny Edge Detector of Bird, Tlow = 0.06 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

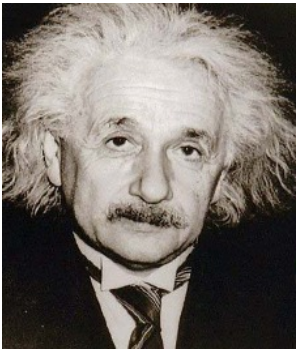
(g) Weak Edge Image

(h) Definitely Not Edge Image

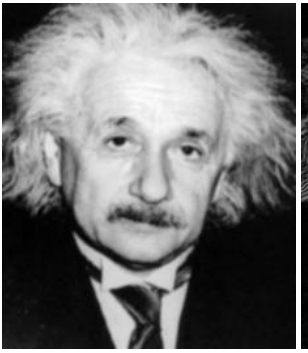
(i) Threshold Image

(j) Canny Edge Detected Image

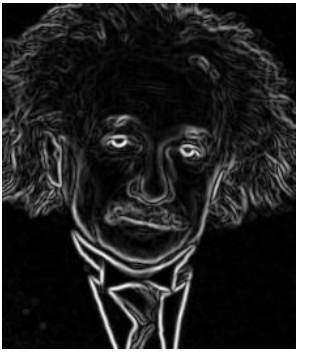
Figure 26: All Intermediate and Final Images through Canny Edge Detector of Dog, Tlow = 0.06 Thigh = 0.25



(a) Original Image

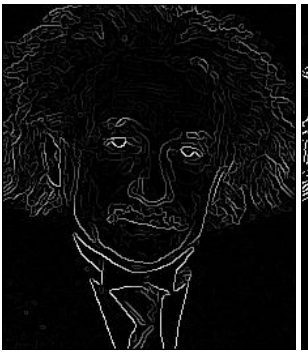


(b) Luminence Image

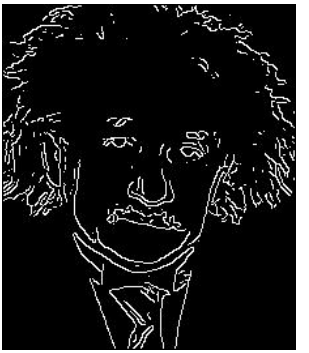


(c) Gradient Magnitude Image

(d) Gradient Direction Image

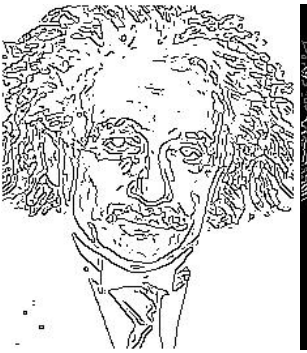


(e) Thinned Edge Image

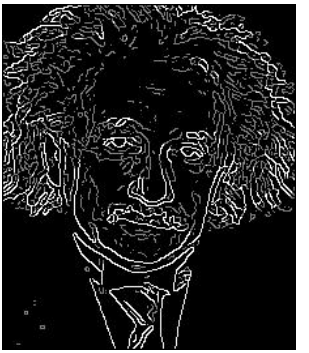


(f) Strong Edge Image

(g) Weak Edge Image



(h) Definitely Not Edge Image



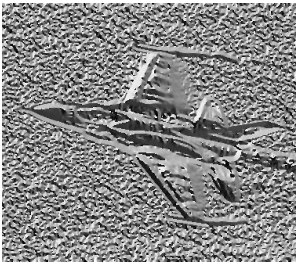
(i) Threshold Image

(j) Canny Edge Detected Im-

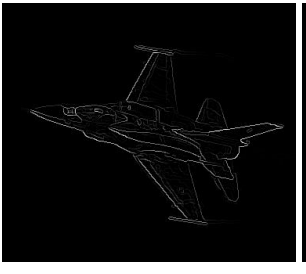
(a) Original Image

(b) Luminence Image

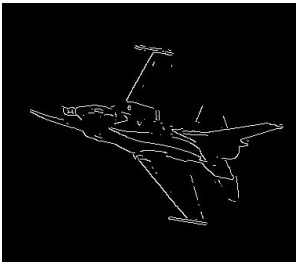
(c) Gradient Magnitude Image



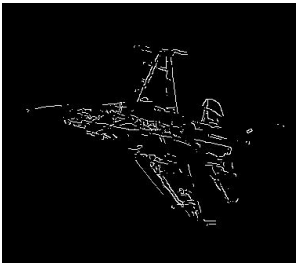
(d) Gradient Direction Image



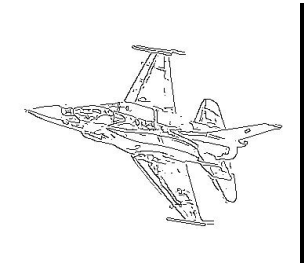
(e) Thinned Edge Image



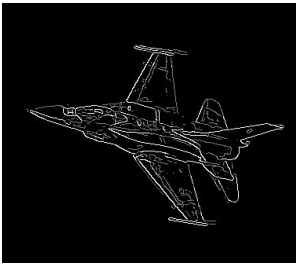
(f) Strong Edge Image



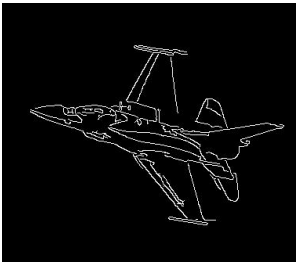
(g) Weak Edge Image



(h) Definitely Not Edge Image

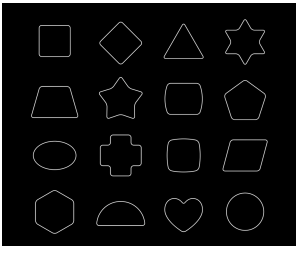


(i) Threshold Image

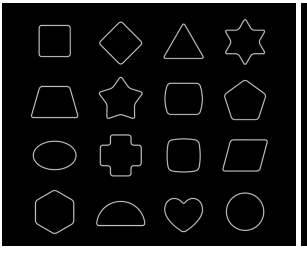


(j) Canny Edge Detected Image

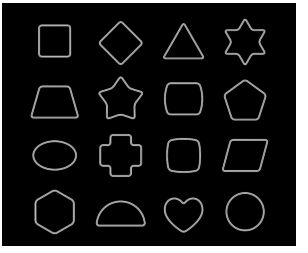
Figure 28: All Intermediate and Final Images through Canny Edge Detector of Plane, Tlow = 0.06 Thigh = 0.25



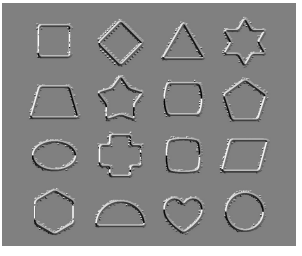
(a) Original Image



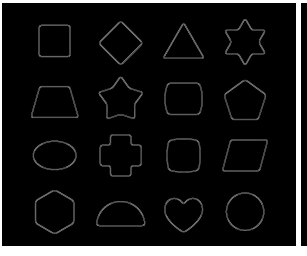
(b) Luminence Image



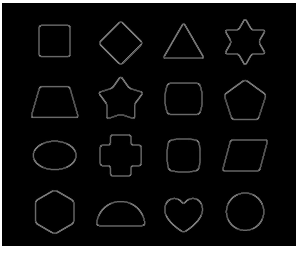
(c) Gradient Magnitude Image



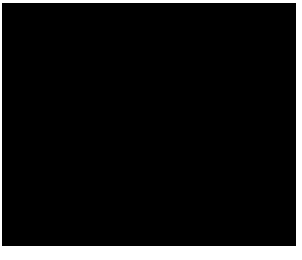
(d) Gradient Direction Image



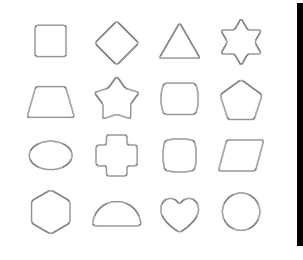
(e) Thinned Edge Image



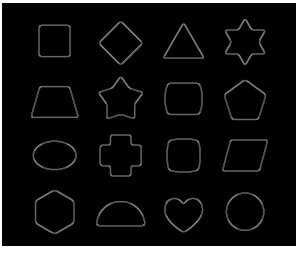
(f) Strong Edge Image



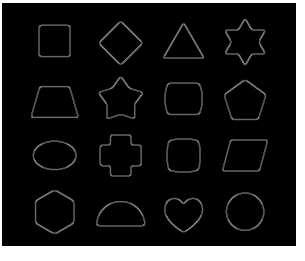
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image

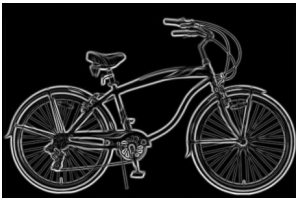


(j) Canny Edge Detected Image

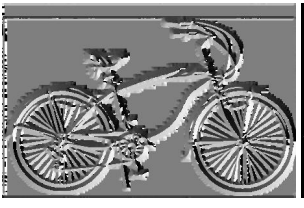
Figure 29: All Intermediate and Final Images through Canny Edge Detector of Toy, Tlow = 0.06 Thigh = 0.25

(a) Original Image

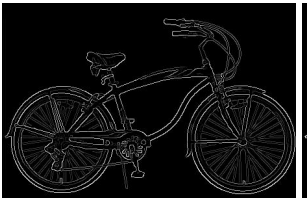
(b) Luminence Image



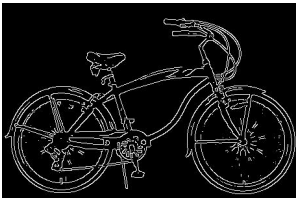
(c) Gradient Magnitude Image



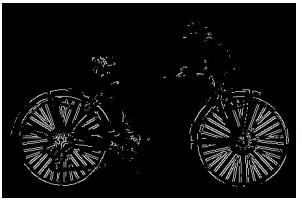
(d) Gradient Direction Image



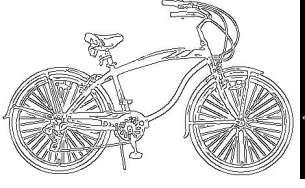
(e) Thinned Edge Image



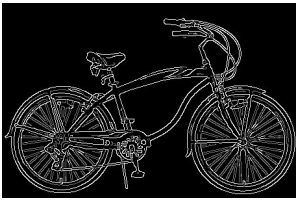
(f) Strong Edge Image



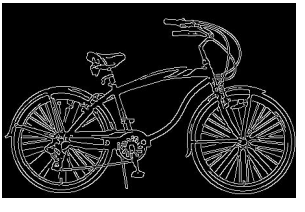
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image



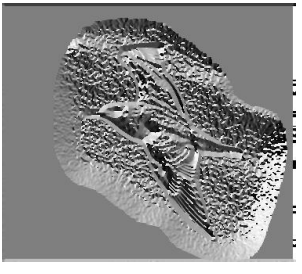
(j) Canny Edge Detected Image

Figure 30: All Intermediate and Final Images through Canny Edge Detector of Bicycle, Tlow = 0.07 Thigh = 0.25

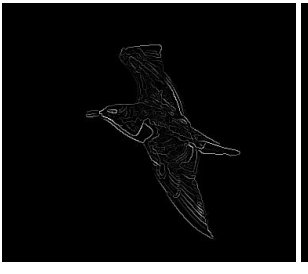
(a) Original Image

(b) Luminence Image

(c) Gradient Magnitude Image



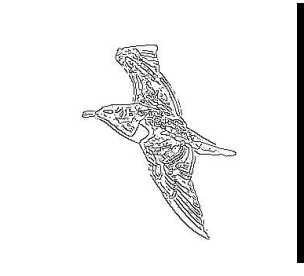
(d) Gradient Direction Image



(e) Thinned Edge Image

(f) Strong Edge Image

(g) Weak Edge Image

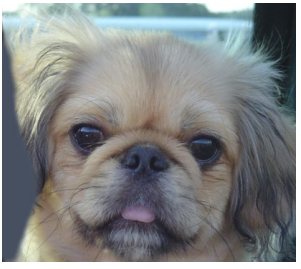


(h) Definitely Not Edge Image

(i) Threshold Image

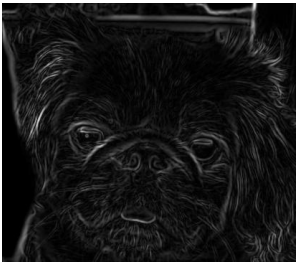
(j) Canny Edge Detected Image

Figure 31: 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

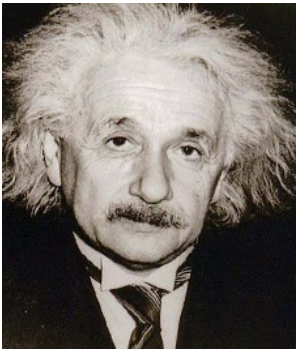
(g) Weak Edge Image

(h) Definitely Not Edge Image

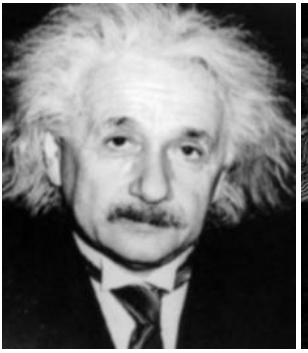
(i) Threshold Image

(j) Canny Edge Detected Image

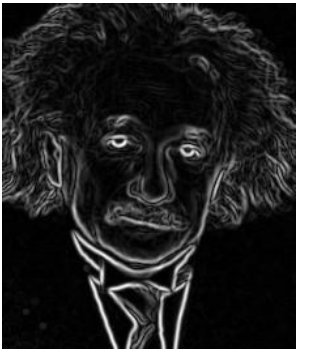
Figure 32: All Intermediate and Final Images through Canny Edge Detector of Dog, Tlow = 0.07 Thigh = 0.25



(a) Original Image

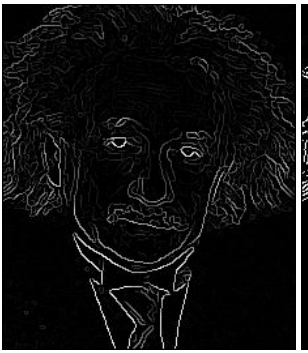


(b) Luminence Image

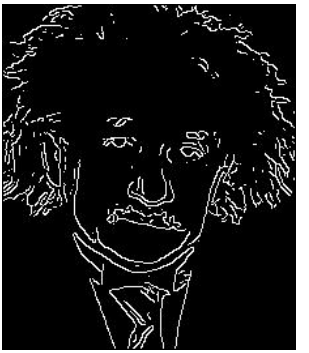


(c) Gradient Magnitude Image

(d) Gradient Direction Image

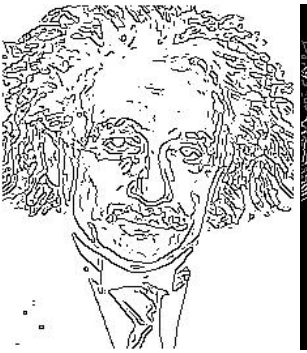


(e) Thinned Edge Image

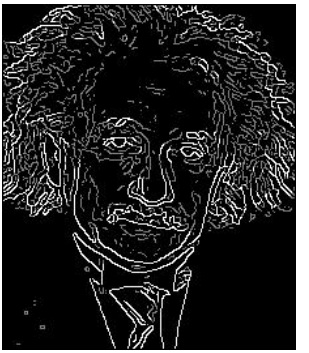


(f) Strong Edge Image

(g) Weak Edge Image



(h) Definitely Not Edge Image



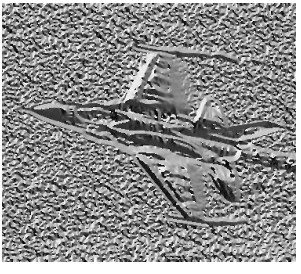
(i) Threshold Image

(j) Canny Edge Detected Im-

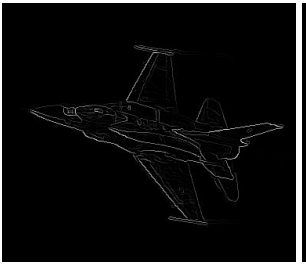
(a) Original Image

(b) Luminence Image

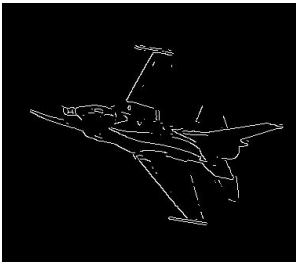
(c) Gradient Magnitude Image



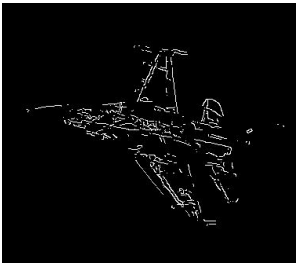
(d) Gradient Direction Image



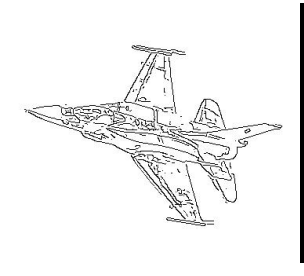
(e) Thinned Edge Image



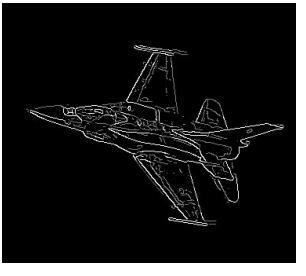
(f) Strong Edge Image



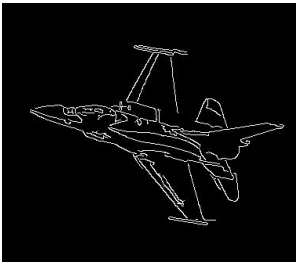
(g) Weak Edge Image



(h) Definitely Not Edge Image

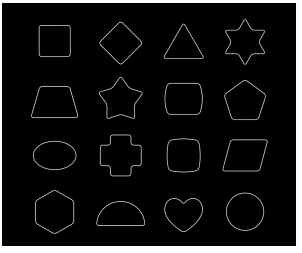


(i) Threshold Image

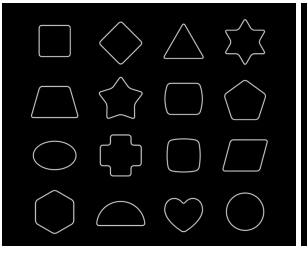


(j) Canny Edge Detected Image

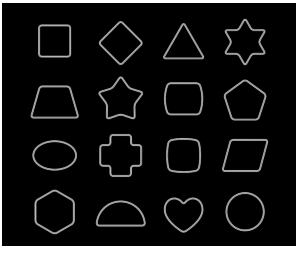
Figure 34: All Intermediate and Final Images through Canny Edge Detector of Plane, Tlow = 0.07 Thigh = 0.25



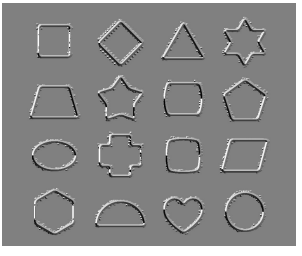
(a) Original Image



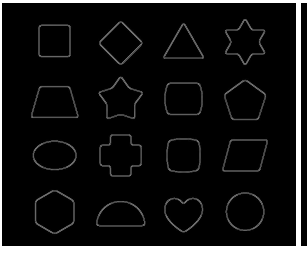
(b) Luminence Image



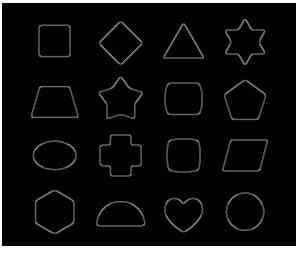
(c) Gradient Magnitude Image



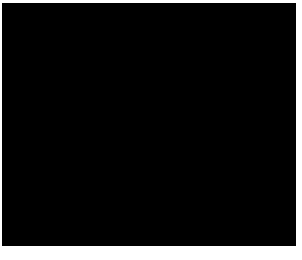
(d) Gradient Direction Image



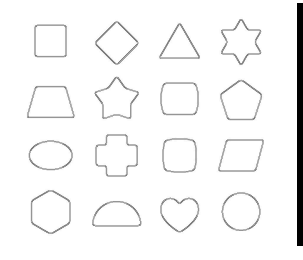
(e) Thinned Edge Image



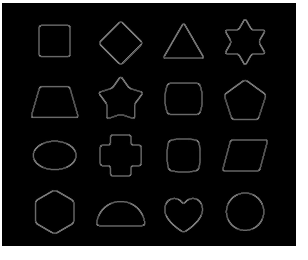
(f) Strong Edge Image



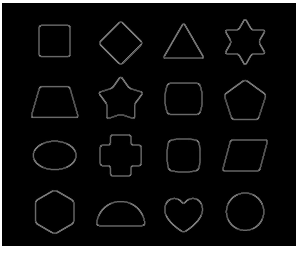
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image

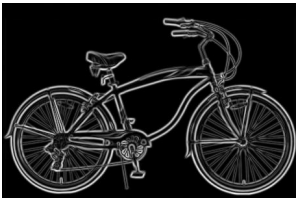


(j) Canny Edge Detected Image

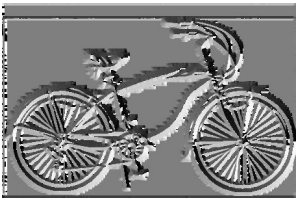
Figure 35: All Intermediate and Final Images through Canny Edge Detector of Toy, Tlow = 0.07 Thigh = 0.25

(a) Original Image

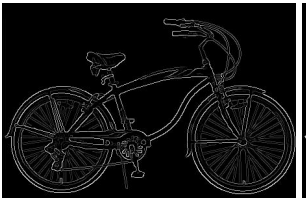
(b) Luminence Image



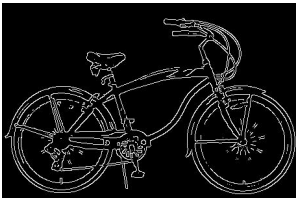
(c) Gradient Magnitude Image



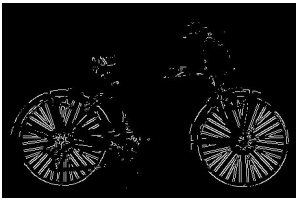
(d) Gradient Direction Image



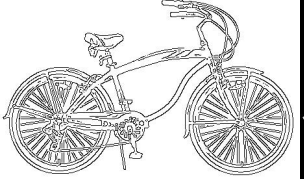
(e) Thinned Edge Image



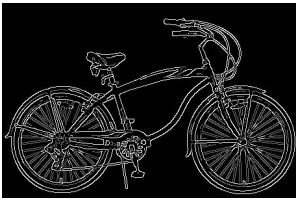
(f) Strong Edge Image



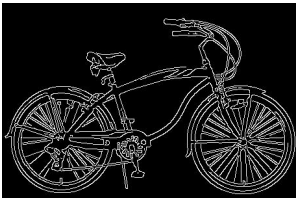
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image



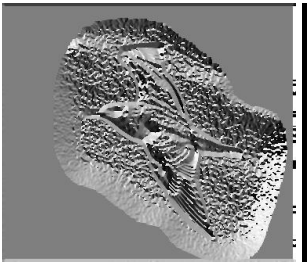
(j) Canny Edge Detected Image

Figure 36: All Intermediate and Final Images through Canny Edge Detector of Bicycle, Tlow = 0.09 Thigh = 0.27

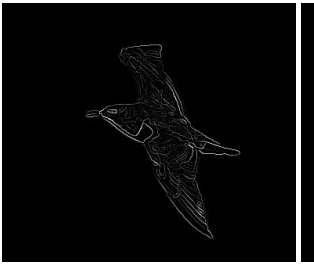
(a) Original Image

(b) Luminence Image

(c) Gradient Magnitude Image



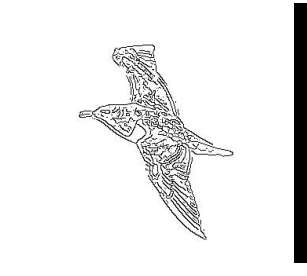
(d) Gradient Direction Image



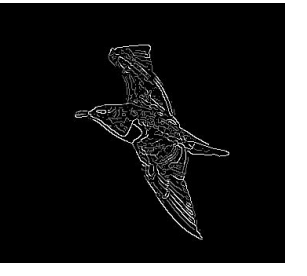
(e) Thinned Edge Image

(f) Strong Edge Image

(g) Weak Edge Image



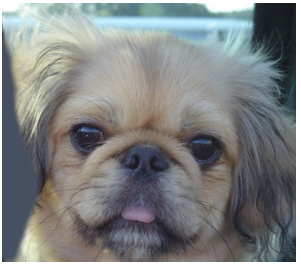
(h) Definitely Not Edge Image



(i) Threshold Image

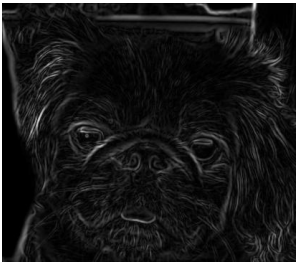
(j) Canny Edge Detected Image

Figure 37: All Intermediate and Final Images through Canny Edge Detector of Bird, Tlow = 0.09 Thigh = 0.27



(a) Original Image

(b) Luminence Image



(c) Gradient Magnitude Image

(d) Gradient Direction Image

(e) Thinned Edge Image

(f) Strong Edge Image

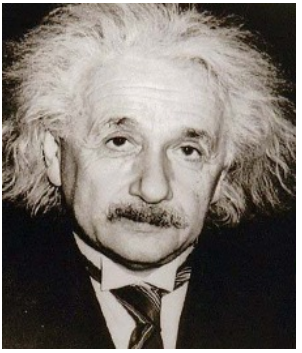
(g) Weak Edge Image

(h) Definitely Not Edge Image

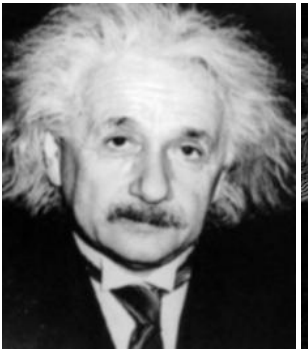
(i) Threshold Image

(j) Canny Edge Detected Image

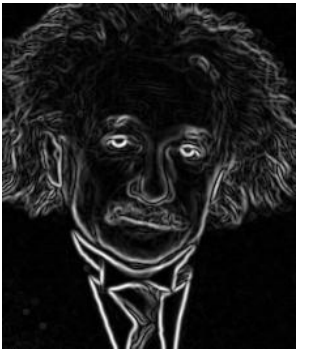
Figure 38: All Intermediate and Final Images through Canny Edge Detector of Dog, Tlow = 0.09 Thigh = 0.27



(a) Original Image

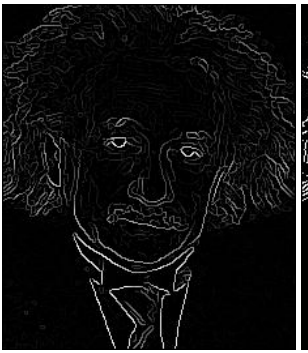


(b) Luminence Image

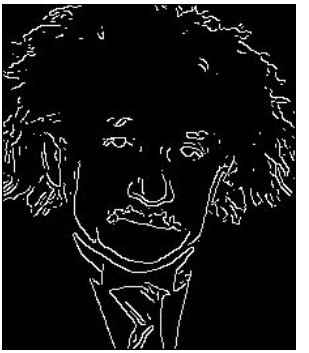


(c) Gradient Magnitude Image

(d) Gradient Direction Image

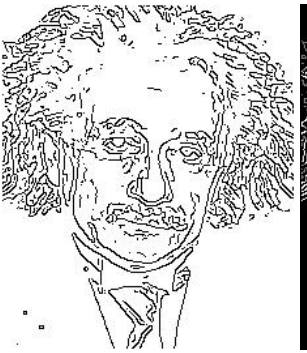


(e) Thinned Edge Image

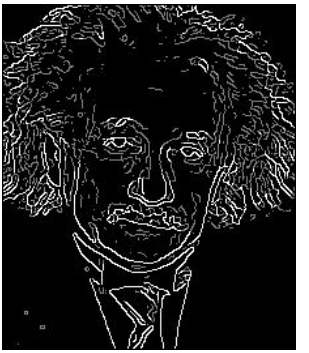


(f) Strong Edge Image

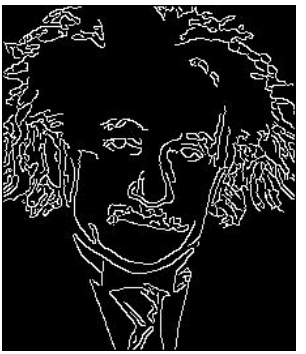
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image

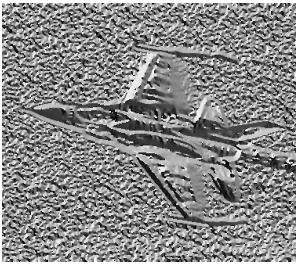


(j) Canny Edge Detected Im-

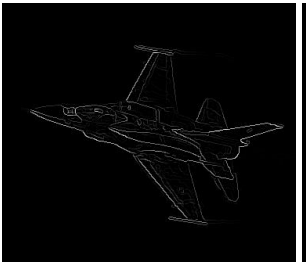
(a) Original Image

(b) Luminence Image

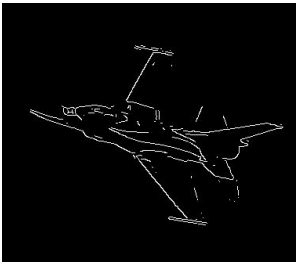
(c) Gradient Magnitude Image



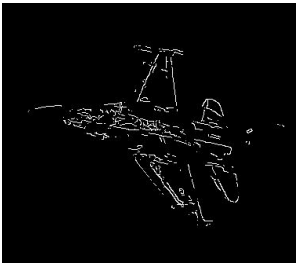
(d) Gradient Direction Image



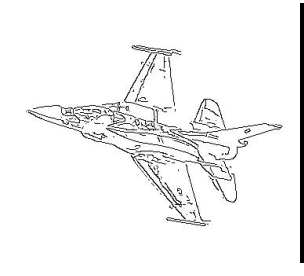
(e) Thinned Edge Image



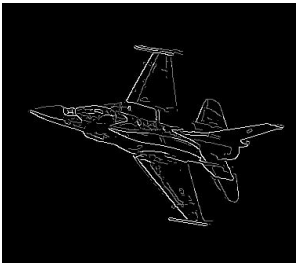
(f) Strong Edge Image



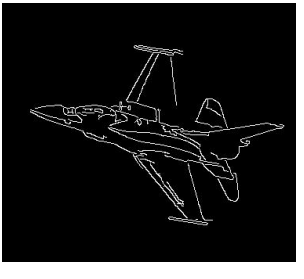
(g) Weak Edge Image



(h) Definitely Not Edge Image

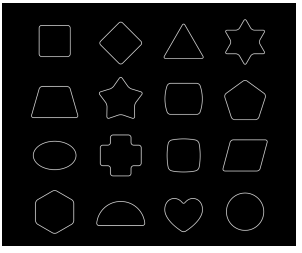


(i) Threshold Image

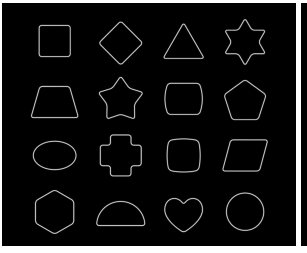


(j) Canny Edge Detected Image

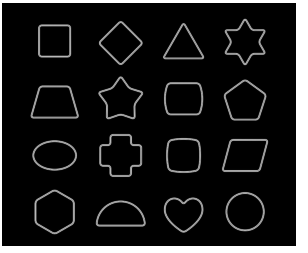
Figure 40: All Intermediate and Final Images through Canny Edge Detector of Plane, Tlow = 0.09 Thigh = 0.27



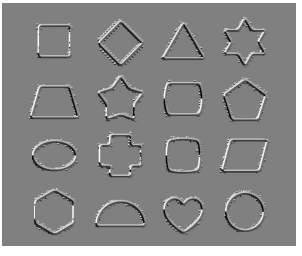
(a) Original Image



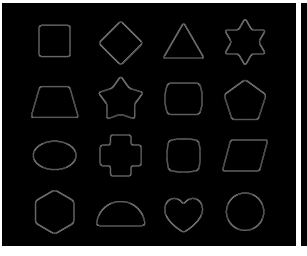
(b) Luminence Image



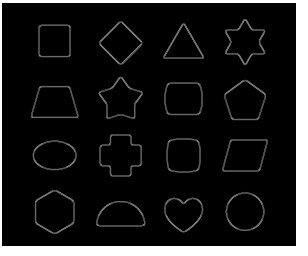
(c) Gradient Magnitude Image



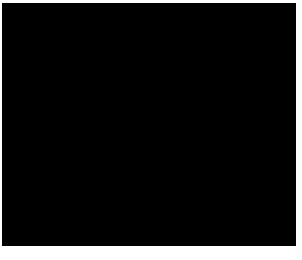
(d) Gradient Direction Image



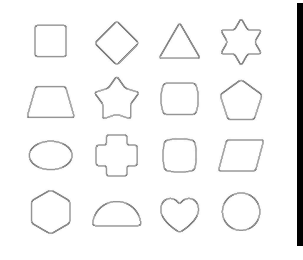
(e) Thinned Edge Image



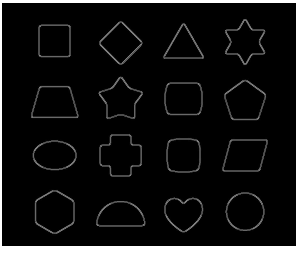
(f) Strong Edge Image



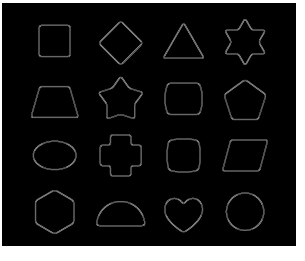
(g) Weak Edge Image



(h) Definitely Not Edge Image



(i) Threshold Image



(j) Canny Edge Detected Image

Figure 41: All Intermediate and Final Images through Canny Edge Detector of Toy, Tlow = 0.09 Thigh = 0.27