Edge:
Canny

Dr. Tushar Sandhan

Introduction

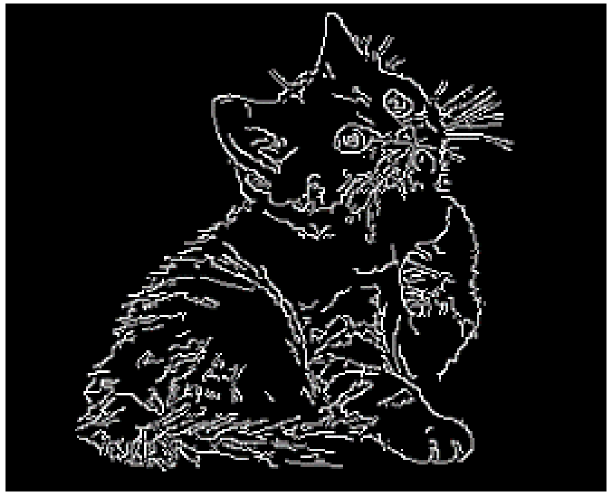
- Single point thick edges

input

Introduction

- Single point thick edges

input

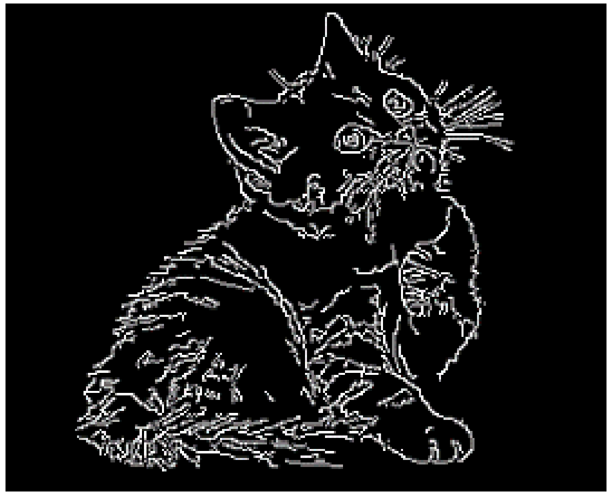


Introduction

- Single point thick edges

input

Canny edges

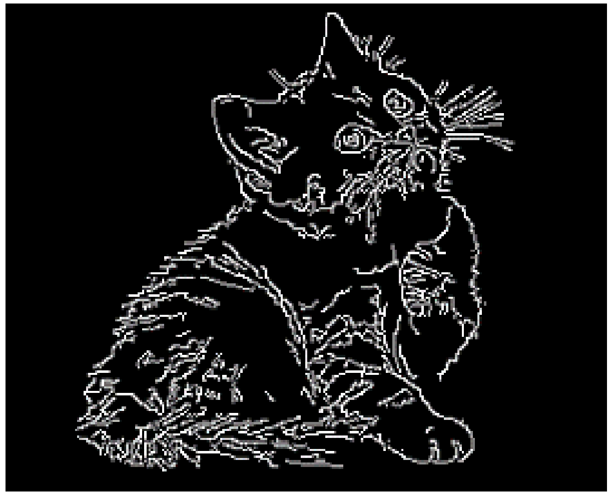


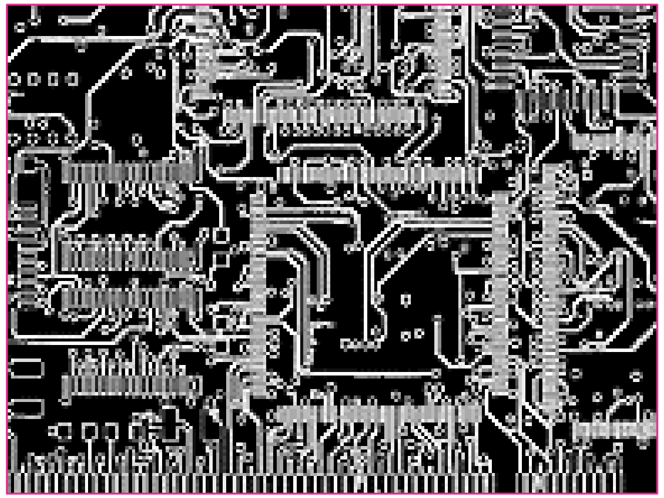
Introduction

- Single point thick edges

input

Canny edges



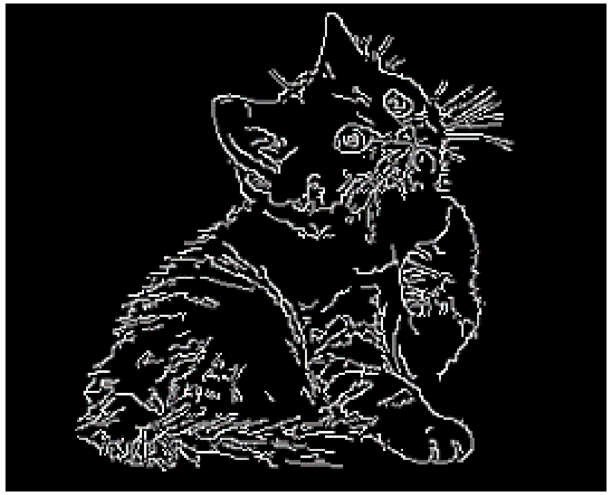


Introduction

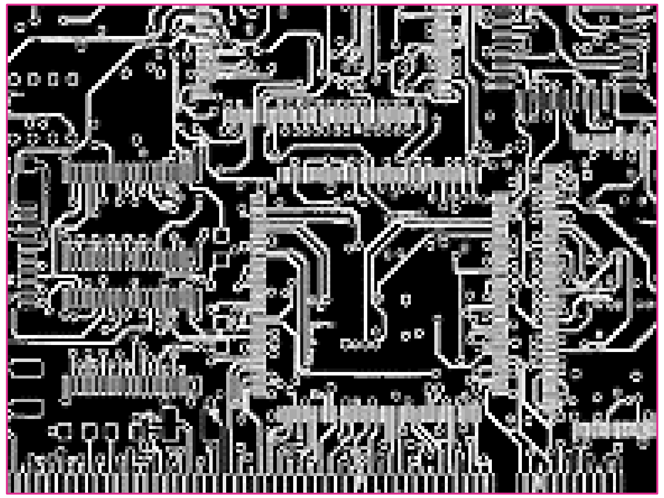
- Single point thick edges

input

Canny edges



Canny PCB edges



Edge

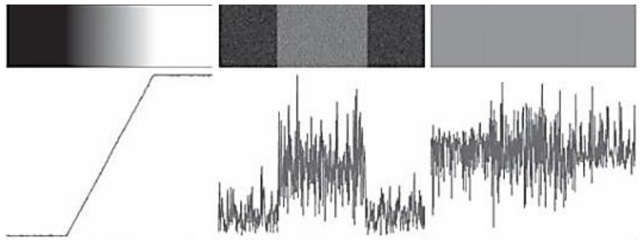
- What would be important steps in edge det.

Edge

- What would be important steps in edge det.
    - Smooth derivatives

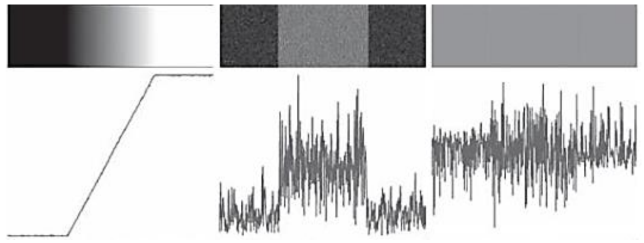
Edge

- What would be important steps in edge det.
  - Smooth derivatives



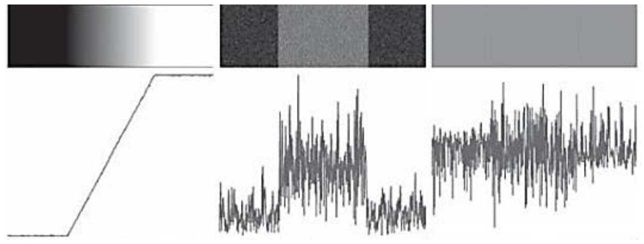
Edge

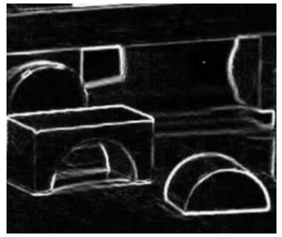
- What would be important steps in edge det.
  - Smooth derivatives
  - Thresholding



Edge

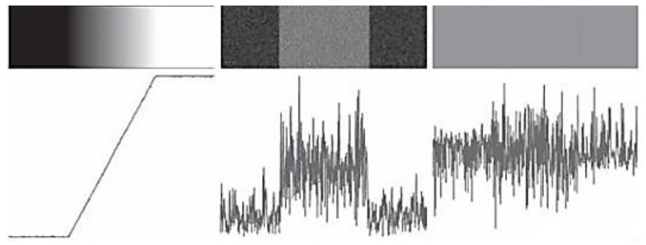
- What would be important steps in edge det.
  - Smooth derivatives
  - Thresholding

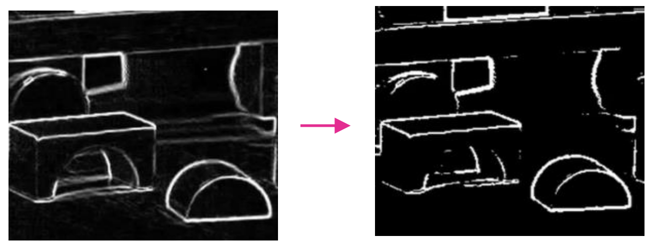




Edge

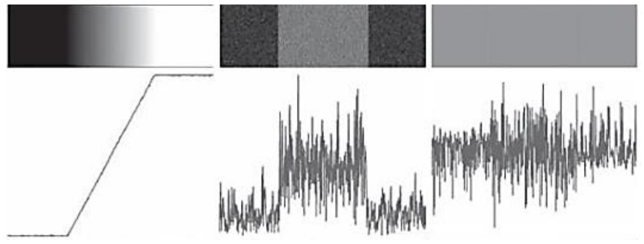
- What would be important steps in edge det.
  - Smooth derivatives
  - Thresholding

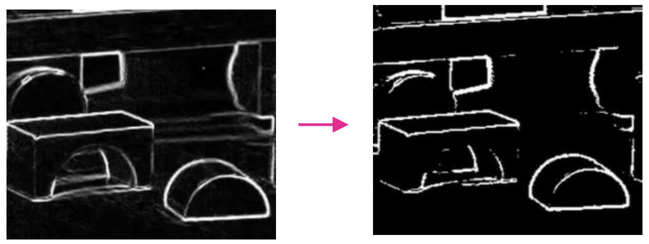




Edge

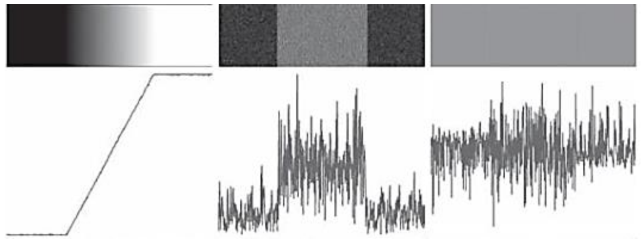
- What would be important steps in edge det.
  - Smooth derivatives
  - Thresholding
  - Thinning

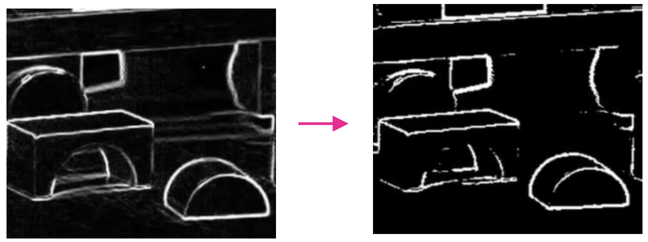




Edge

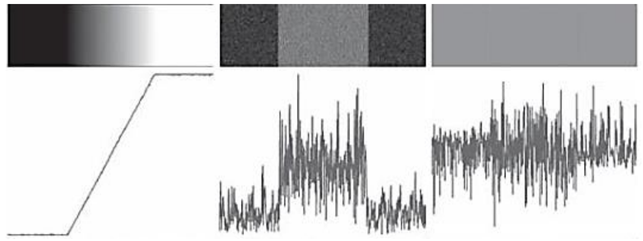
- What would be important steps in edge det.
  - Smooth derivatives
  - Thresholding
  - Thinning
  - Linking

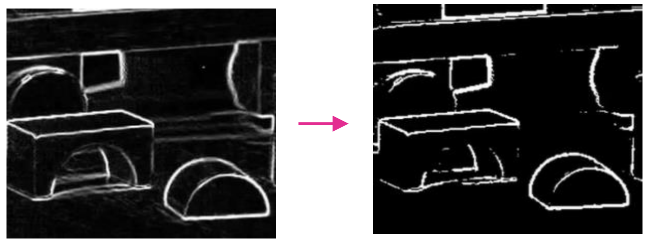




Edge

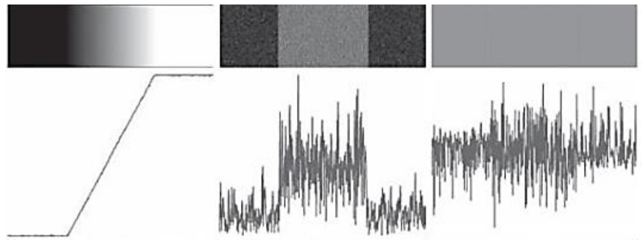
- What would be important steps in edge det.
  - Smooth derivatives
  - Thresholding
  - Thinning
  - Linking

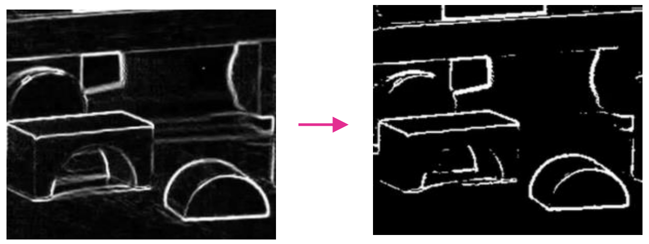


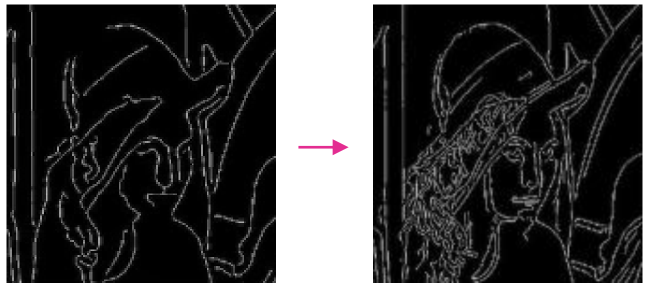


Edge

- What would be important steps in edge det.
  - Smooth derivatives
  - Thresholding
  - Thinning
  - Linking







Canny edge detector

- Objectives

Canny edge detector

- Objectives
  - low error rate
    - all edges should be found

Canny edge detector

- Objectives
  - low error rate
    - all edges should be found
  - good localization of edges
    - centre of true edge at i^{th} pixel : c_i
    - obtained edge point at i^{th} pixel: e_i
    - minimize the distance $||c_i - e_i||_2$

Canny edge detector

- Objectives
  - low error rate
    - all edges should be found
  - good localization of edges
    - centre of true edge at i^{th} pixel : c_i
    - obtained edge point at i^{th} pixel: e_i
    - minimize the distance $\|c_i - e_i\|_2$
  - single point edge response
    - 1 point for each true edge point

Canny edge detector

- Image derivatives
  - input image $f(x,y)$
  - smoothed $f_s(x,y)$
  - any operator can be used to get $g_x(x,y)$, $g_y(x,y)$

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Canny edge detector

- Image derivatives
  - input image $f(x, y)$
  - smoothed $f_s(x, y)$
  - any operator can be used to get $g_x(x, y)$, $g_y(x, y)$

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$f_s(x,y) = G(x,y) \star f(x,y)$$

Canny edge detector

- Image derivatives
  - input image $f(x, y)$
  - smoothed $f_s(x, y)$
  - any operator can be used to get $g_x(x, y)$, $g_y(x, y)$

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$f_s(x,y) = G(x,y) \star f(x,y)$$

$$g_x(x,y) = \partial f_s(x,y)/\partial x \qquad g_y(x,y) = \partial f_s(x,y)/\partial y$$

Canny edge detector

- Image derivatives
  - input image $f(x, y)$
  - smoothed $f_s(x, y)$
  - any operator can be used to get $g_x(x, y)$, $g_y(x, y)$

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$f_s(x,y) = G(x,y) \star f(x,y)$$

$$g_x(x,y) = \partial f_s(x,y)/\partial x \qquad g_y(x,y) = \partial f_s(x,y)/\partial y$$

$$M_s(x,y) = \|\nabla f_s(x,y)\| = \sqrt{g_x^2(x,y) + g_y^2(x,y)}$$

Canny edge detector

- Image derivatives
  - input image $f(x,y)$
  - smoothed $f_s(x,y)$
  - any operator can be used to get $g_x(x,y)$, $g_y(x,y)$

$$G(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$f_s(x,y) = G(x,y) \star f(x,y)$$

$$g_x(x,y) = \partial f_s(x,y)/\partial x \qquad g_y(x,y) = \partial f_s(x,y)/\partial y$$

$$M_s(x,y) = \|\nabla f_s(x,y)\| = \sqrt{g_x^2(x,y) + g_y^2(x,y)} \qquad \alpha(x,y) = \tan^{-1}\left[\frac{g_y(x,y)}{g_x(x,y)}\right]$$

Canny edge detector

- Thinning
  - $M_S(x, y)$ wide ridges around local maxima
  - ridges thinning is needed
  - non-max suppression
    - suppress where?

$$M_s(x,y) = \|\nabla f_s(x,y)\| = \sqrt{g_x^2(x,y) + g_y^2(x,y)}$$

$$\alpha(x,y) = \tan^{-1}\left[\frac{g_y(x,y)}{g_x(x,y)}\right]$$

Canny edge detector

- Thinning
  - $M_s(x, y)$ wide ridges around local maxima
  - ridges thinning is needed
  - non-max suppression
    - suppress where?
    - on the ridges

$$M_s(x,y) = \|\nabla f_s(x,y)\| = \sqrt{g_x^2(x,y) + g_y^2(x,y)}$$

$$\alpha(x,y) = \tan^{-1}\left[\frac{g_y(x,y)}{g_x(x,y)}\right]$$

Canny edge detector

- Thinning
  - $M_S(x,y)$ wide ridges around local maxima
  - ridges thinning is needed
  - non-max suppression
    - suppress where?
    - on the ridges
    - how?

$$M_s(x,y) = \|\nabla f_s(x,y)\| = \sqrt{g_x^2(x,y) + g_y^2(x,y)}$$

$$\alpha(x,y) = \tan^{-1}\left[\frac{g_y(x,y)}{g_x(x,y)}\right]$$

Canny edge detector

- Thinning
  - $M_S(x,y)$ wide ridges around local maxima
  - ridges thinning is needed
  - non-max suppression
    - suppress where?
    - on the ridges
    - how?
    - walking along edge normals?

$$M_s(x,y) = \|\nabla f_s(x,y)\| = \sqrt{g_x^2(x,y) + g_y^2(x,y)}$$

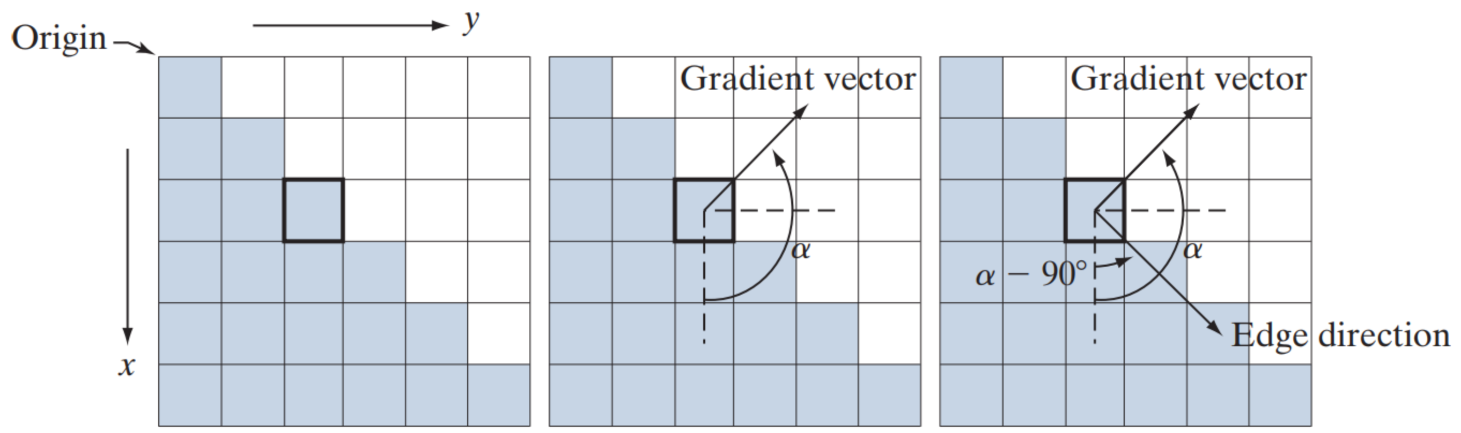
$$\alpha(x,y) = \tan^{-1}\left[\frac{g_y(x,y)}{g_x(x,y)}\right]$$

Canny edge detector

- Thinning
  - $M_s(x, y)$ wide ridges around local maxima
  - ridges thinning is needed
  - non-max suppression
    - suppress where?
    - on the ridges
    - how?
    - walking along edge normals?

$$M_s(x,y) = \|\nabla f_s(x,y)\| = \sqrt{g_x^2(x,y) + g_y^2(x,y)}$$

$$\alpha(x,y) = \tan^{-1}\left[\frac{g_y(x,y)}{g_x(x,y)}\right]$$

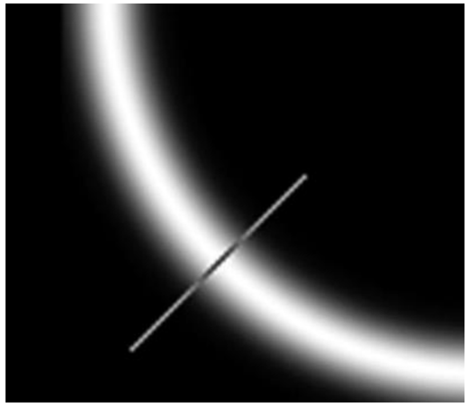


Canny edge detector

- Thinning
  - non-max suppression:
    checks whether pixel is local maxima
    in grad direction
  - linear interpolation for missing
    locations e.g. r, p

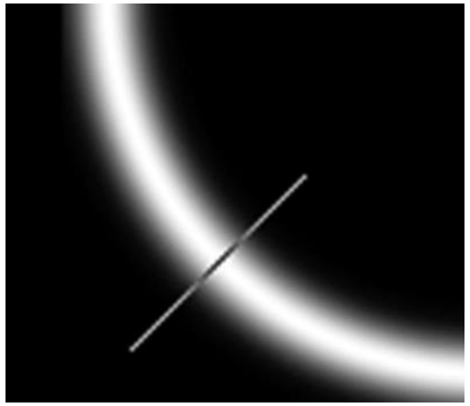
Canny edge detector

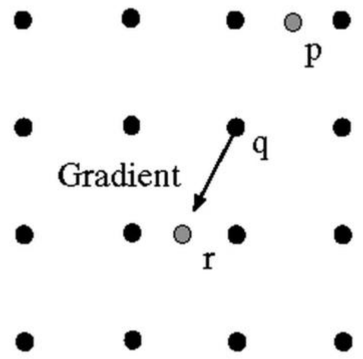
- Thinning
  - non-max suppression:
    checks whether pixel is local maxima in grad direction
  - linear interpolation for missing locations e.g. r, p



Canny edge detector

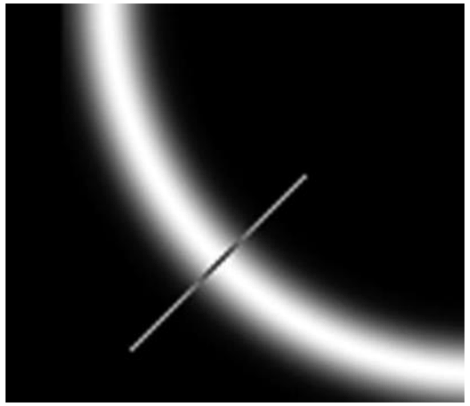
- Thinning
  - non-max suppression:
    checks whether pixel is local maxima in grad direction
  - linear interpolation for missing locations e.g. r, p

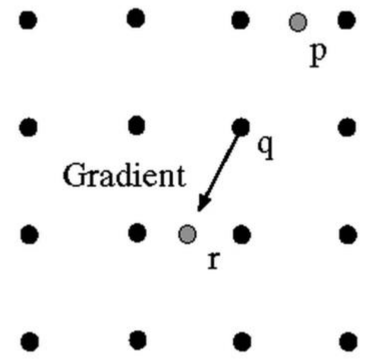


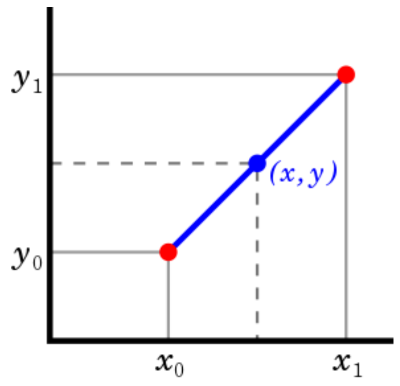


Canny edge detector

- Thinning
  - non-max suppression:
    checks whether pixel is local maxima in grad direction
  - linear interpolation for missing locations e.g. r, p

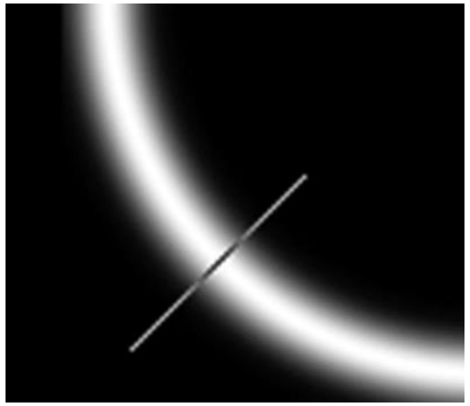


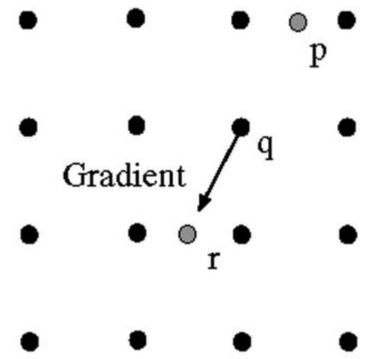




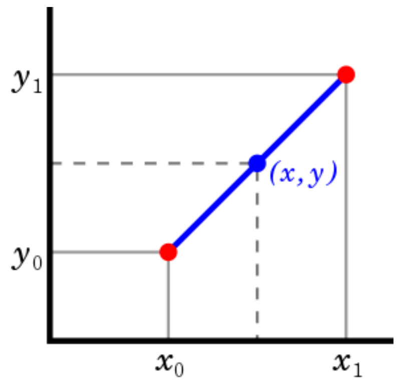
Canny edge detector

- Thinning
  - non-max suppression:
    checks whether pixel is local maxima in grad direction
  - linear interpolation for missing locations e.g. r, p



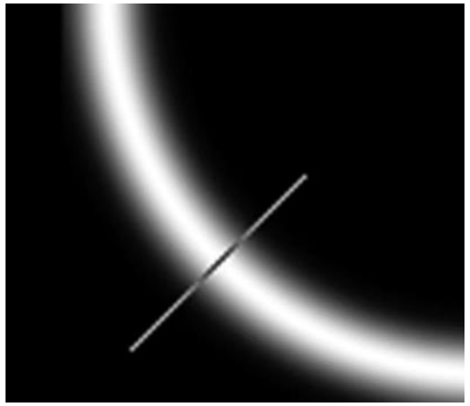


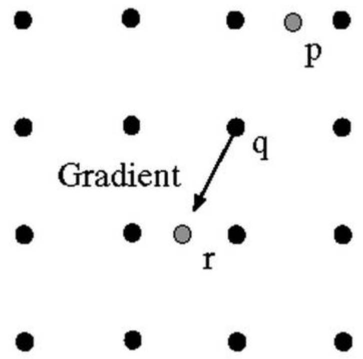
$$\frac{y - y_0}{x - x_0} = \frac{y_1 - y_0}{x_1 - x_0}$$



Canny edge detector

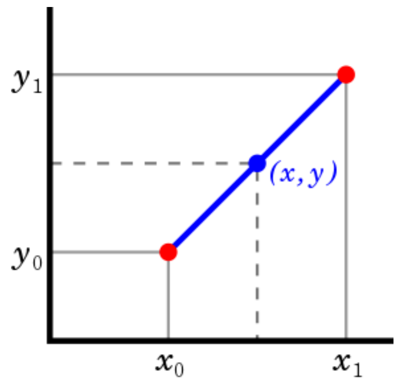
- Thinning
  - non-max suppression:
    checks whether pixel is local maxima in grad direction
  - linear interpolation for missing locations e.g. r, p





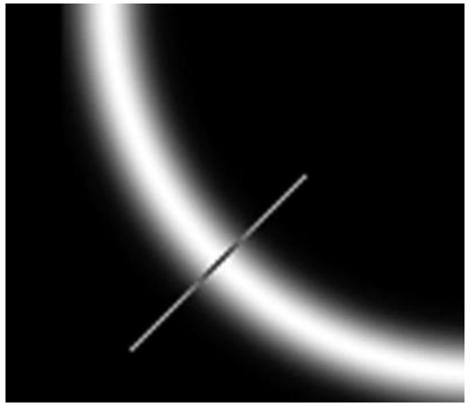
$$\frac{y - y_0}{x - x_0} = \frac{y_1 - y_0}{x_1 - x_0}$$

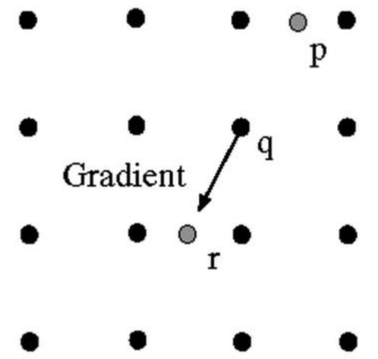
$$y = y_0 + (x - x_0)\frac{y_1 - y_0}{x_1 - x_0}$$



Canny edge detector

- Thinning
  - non-max suppression:
    checks whether pixel is local maxima in grad direction
  - linear interpolation for missing locations e.g. r, p

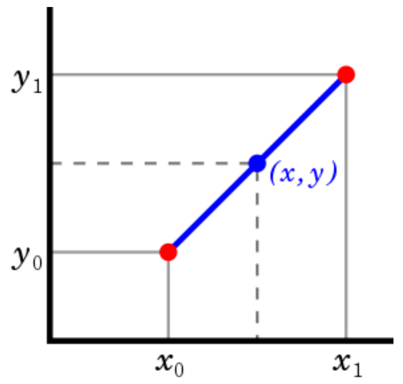




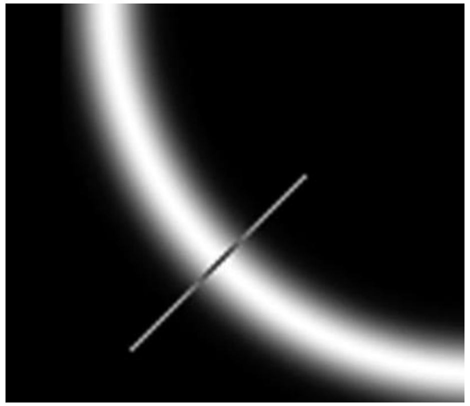
$$\frac{y - y_0}{x - x_0} = \frac{y_1 - y_0}{x_1 - x_0}$$

$$y = y_0 + (x - x_0)\frac{y_1 - y_0}{x_1 - x_0}$$

$$= y_0 \left(1 - \frac{x - x_0}{x_1 - x_0}\right) + y_1 \left(\frac{x - x_0}{x_1 - x_0}\right)$$

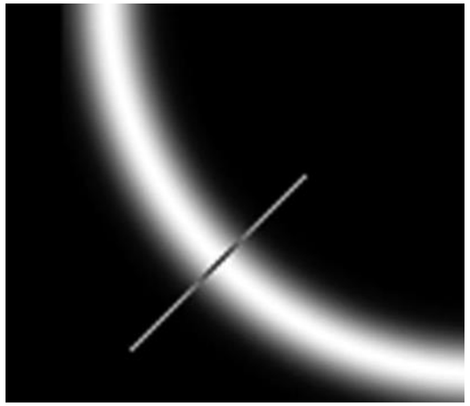


Canny edge detector



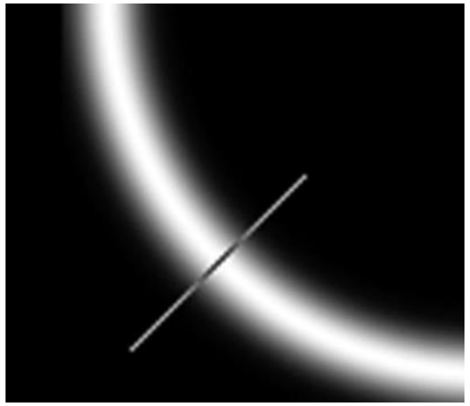
Canny edge detector

- LINKINg Points
  - Canny edge detector
    - It starts with one thing: gradients
    - In the end, it doesn't even matter: which operators have been used
  - at last we have to link the non-suppressed ones!



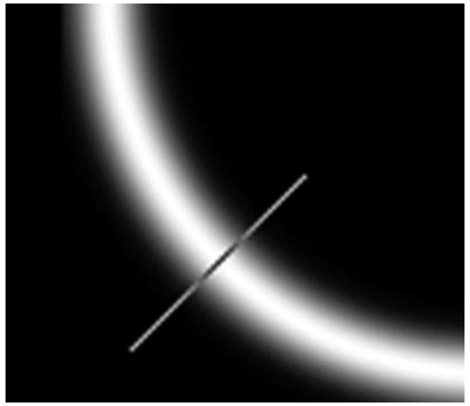
Canny edge detector

- LINKINg Points
  - Canny edge detector
    - It starts with one thing: gradients
    - In the end, it doesn't even matter: which operators have been used
  - at last we have to link the non-suppressed ones!



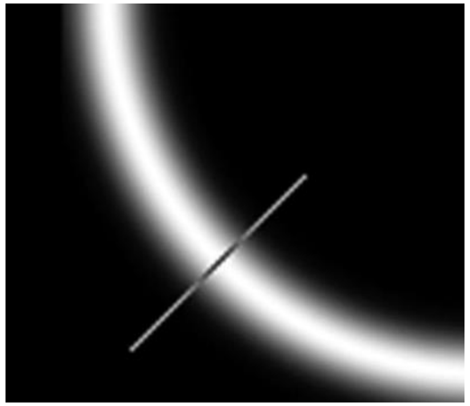
Canny edge detector

- LINKINg Points
  - Canny edge detector
    - It starts with one thing: gradients
    - In the end, it doesn't even matter: which operators have been used
  - at last we have to link the non-suppressed ones!



Canny edge detector

- LINKINg Points
  - Canny edge detector
    - It starts with one thing: gradients
    - In the end, it doesn't even matter: which operators have been used
  - at last we have to link the non-suppressed ones!
  - two instruments of thresholds: Hysteresis

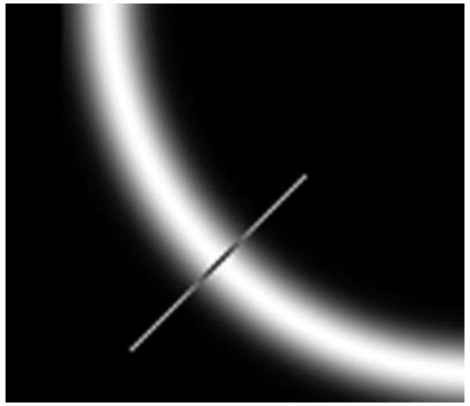


Canny edge detector

- LINKINg Points
  - Canny edge detector
    - It starts with one thing: gradients
    - In the end, it doesn't even matter: which operators have been used
  - at last we have to link the non-suppressed ones!

  - two instruments of thresholds: Hysteresis

    a. find all edge points using TH^{high}
    b. from each strong point follow the both side direction ⊥ to the edge normal
    c. in that directions, construct the contours of connected edge points
    d. mark all points greater than TH_{low}



Canny edge detector

- Entire algorithm composition:

Canny edge detector

- Entire algorithm composition:

1. Filter image with derivatives of Gaussian
2. Get M, α
3. Non-max suppression
   - thin multi-pixel wide edges to a single pixel widths
4. Linking: the hysteresis
   - 2 thresholds: TH_{low} , TH^{high}
   - TH^{high}: to start an edge
   - TH_{low} : continue started edge

Canny edge detector

Canny edge detector

- Speeding up the beats of operations
  - binning the α (angles)
  - 4 directions

Canny edge detector

- Speeding up the beats of operations
  - binning the α (angles)
  - 4 directions

    Horizontal

    +45 degrees

    Vertical

    -45 degrees

Canny edge detector

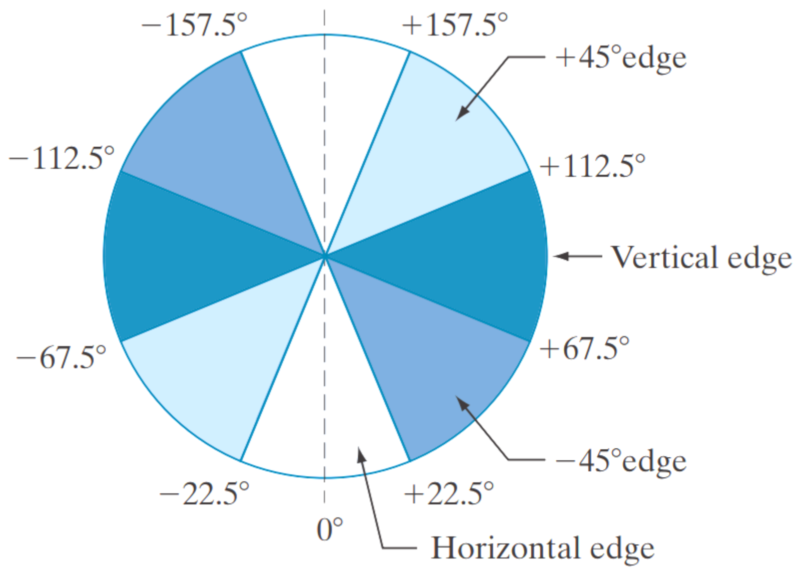
- Speeding up the beats of operations
  - binning the α (angles)
  - 4 directions

    Horizontal

    +45 degrees

    Vertical

    -45 degrees



Canny edge detector

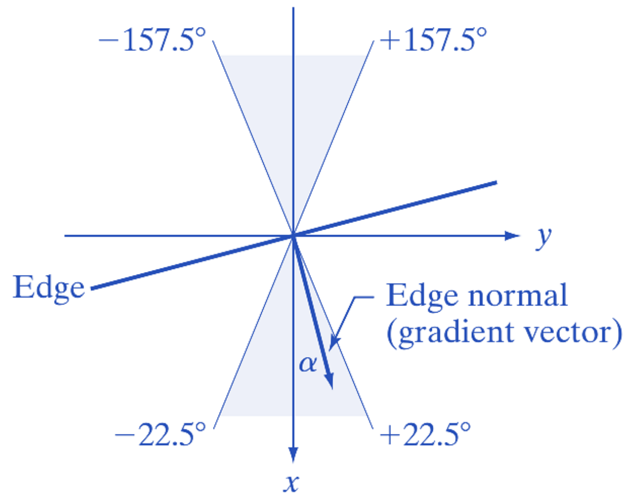
- Speeding up the beats of operations
  - binning the α (angles)
  - 4 directions

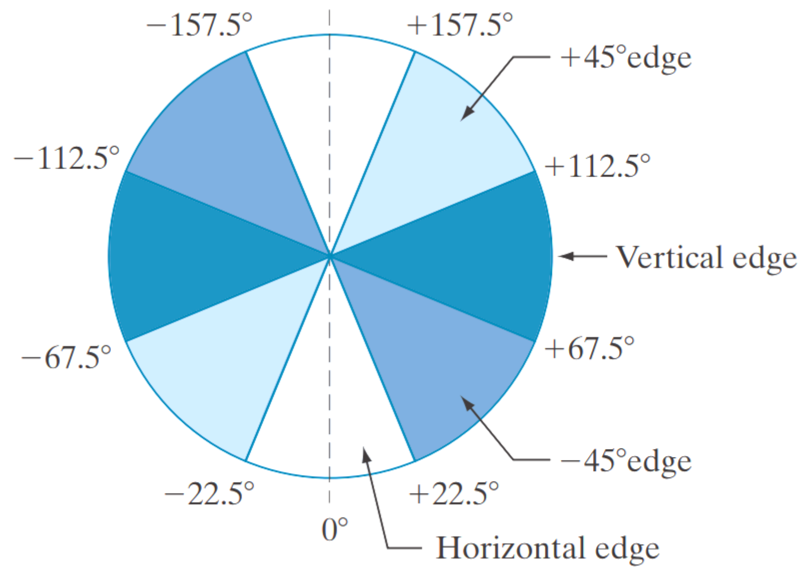
Horizontal

+45 degrees

Vertical

-45 degrees

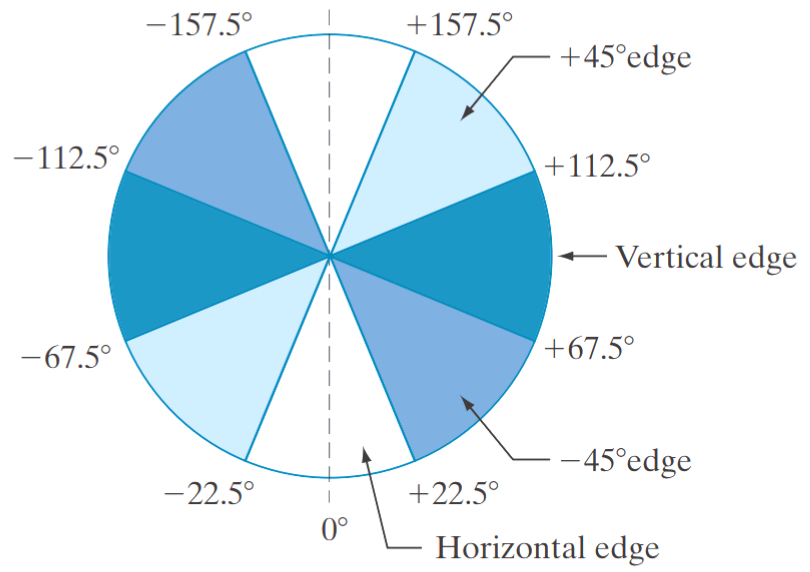




Canny edge detector

- Speeding up the beats of operations
  - binning the α (angles)
    - get the directional bin $Bin()$ closest to α
    - from previous operations edge: $M(x,y)$
    - suppression
      - If $M(x',y') > M(x,y)$ then

        $M(x,y) \rightarrow 0$
      - where neighbors $x',y' \leftarrow Bin(x,y)$



Canny edge detector

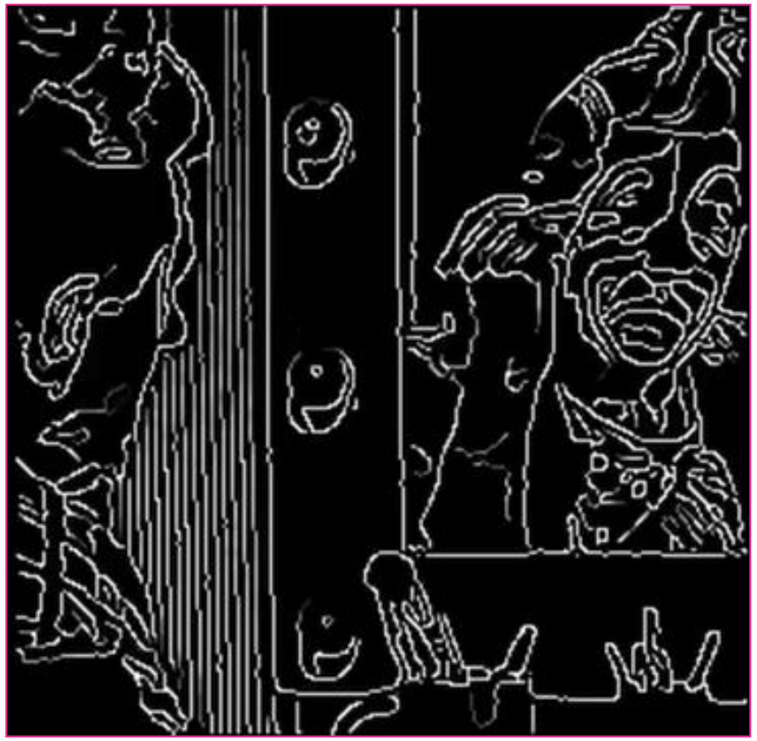
- Varying σ

input

Canny edge detector

- Varying σ

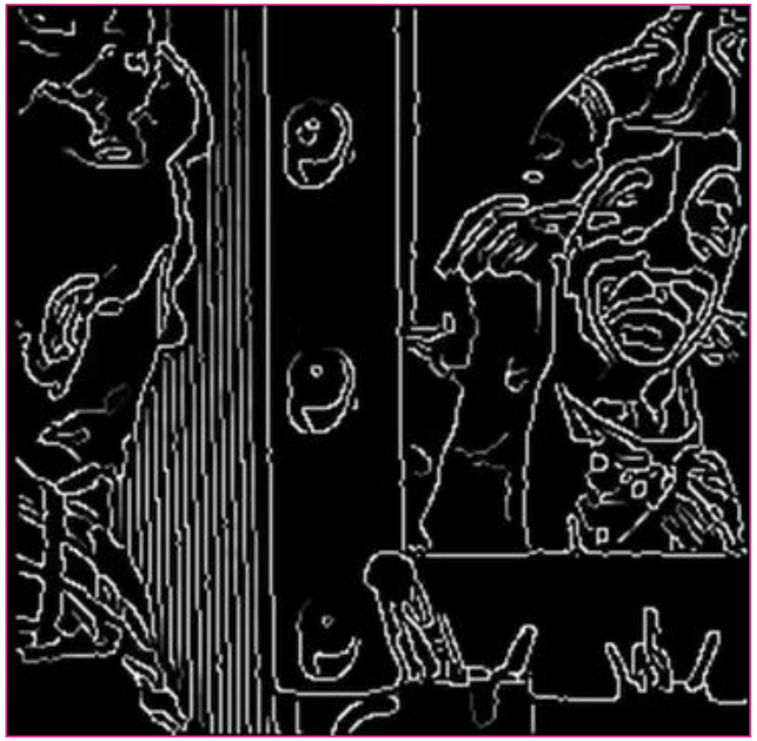
input

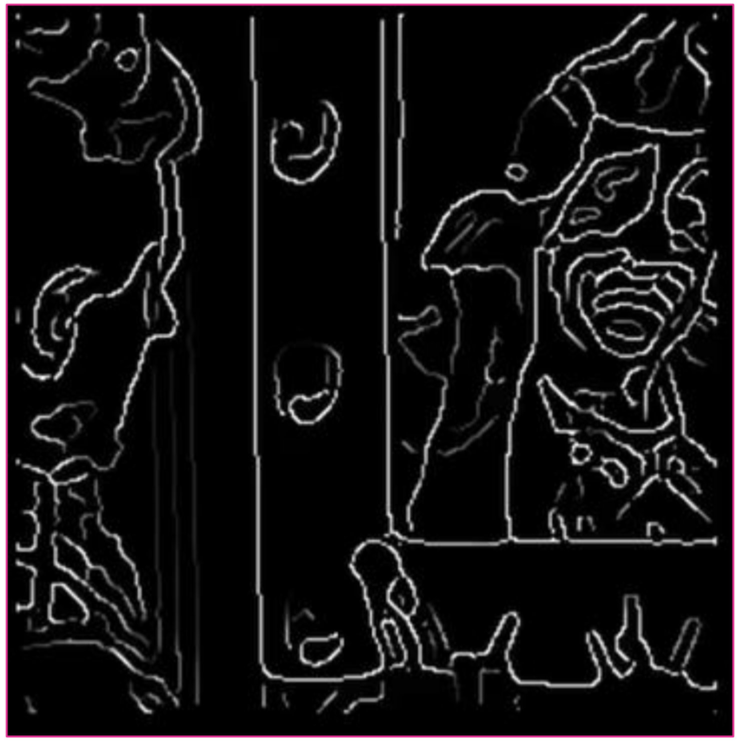


Canny edge detector

- Varying σ

input



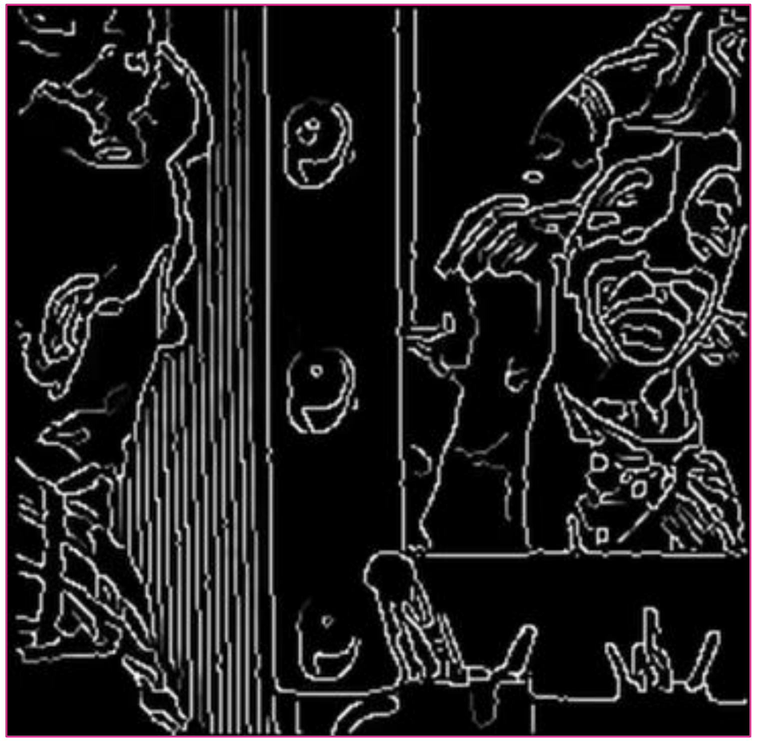


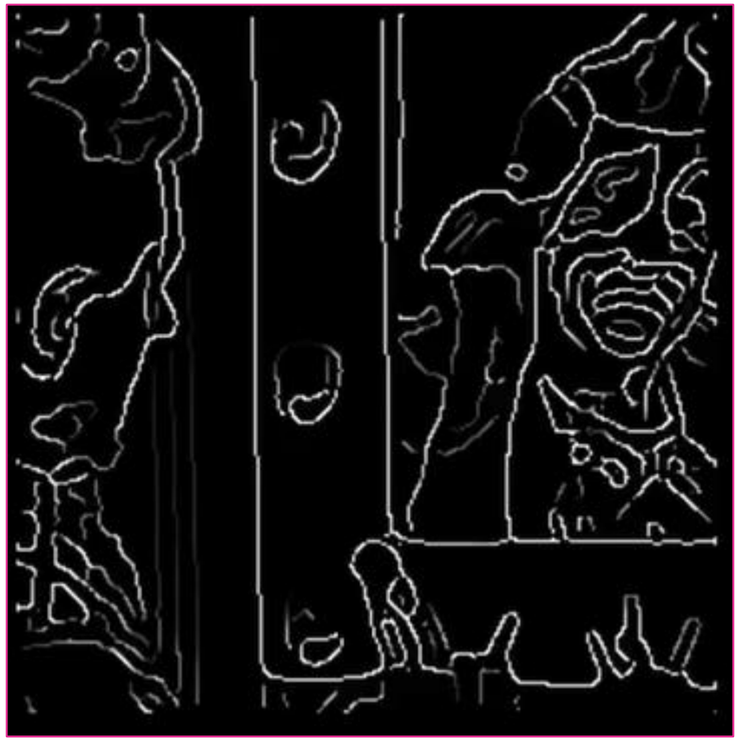
Canny edge detector

- Varying σ

input

σ small



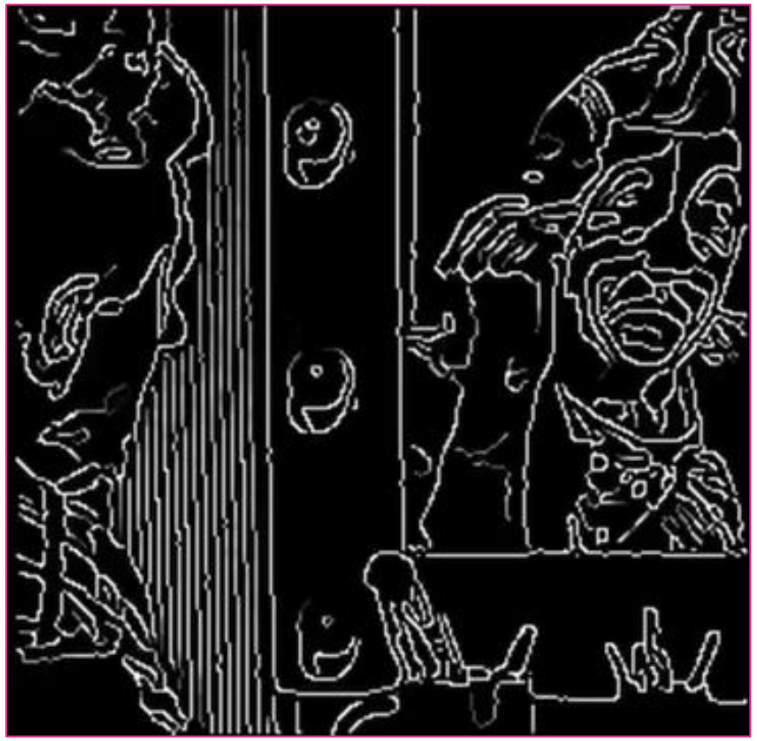


Canny edge detector

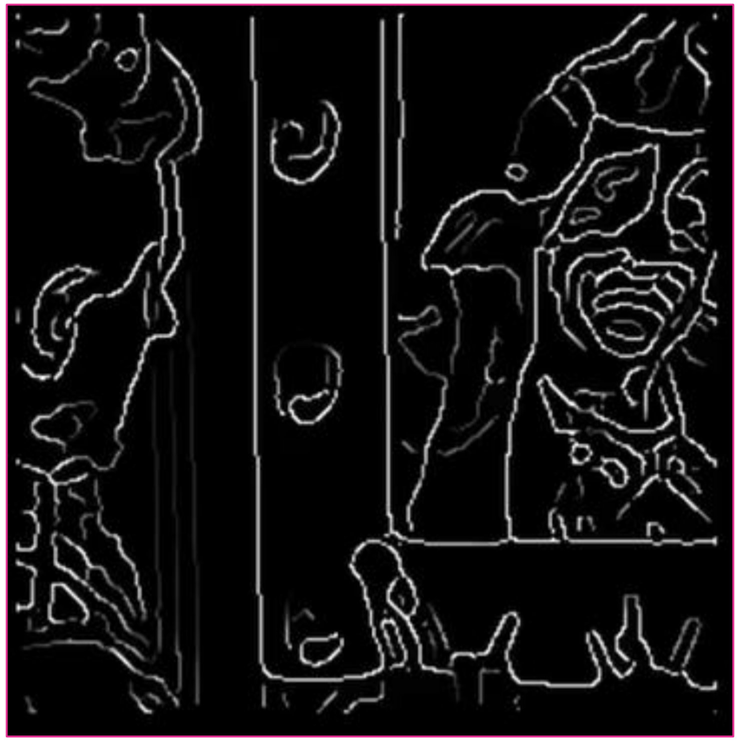
- Varying σ

input

σ small



σ large



Canny edge detector

- Comparing other edge detectors

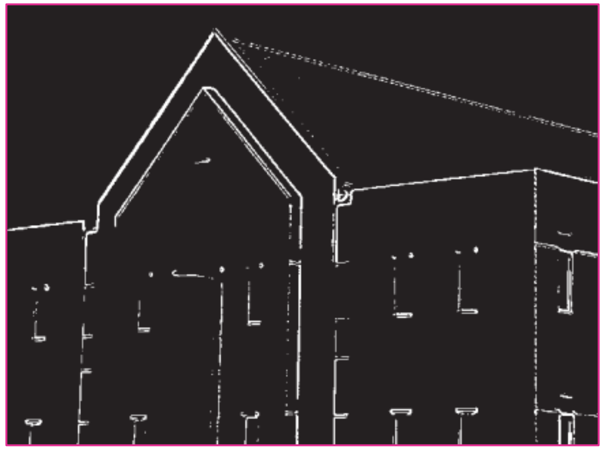
input

Canny edge detector

- Comparing other edge detectors

input

Sobel with TH

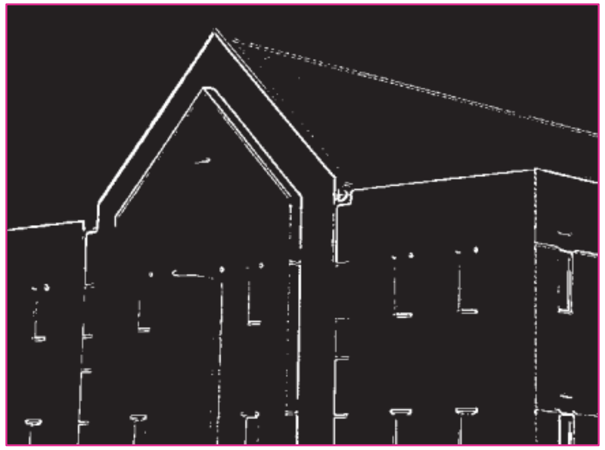


Canny edge detector

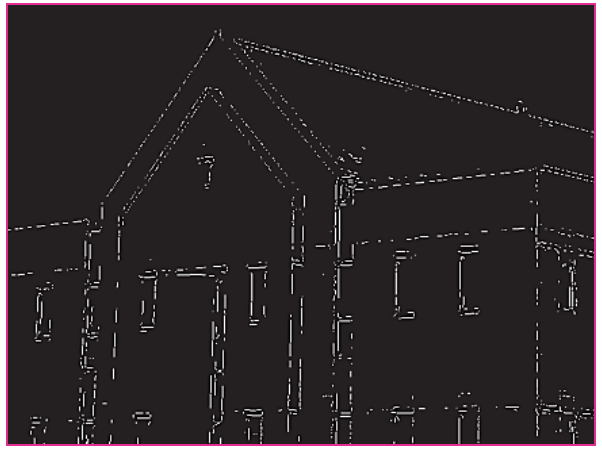
- Comparing other edge detectors

input

Sobel with TH



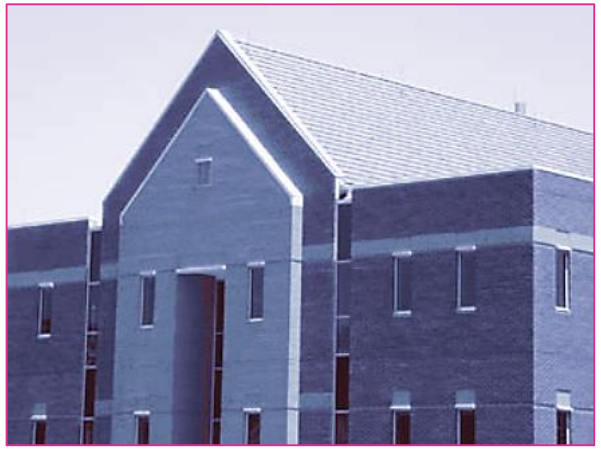
LoG zero crossings



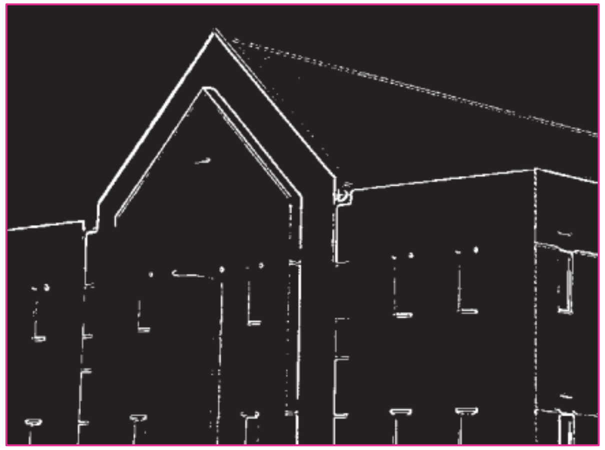
Canny edge detector

- Comparing other edge detectors

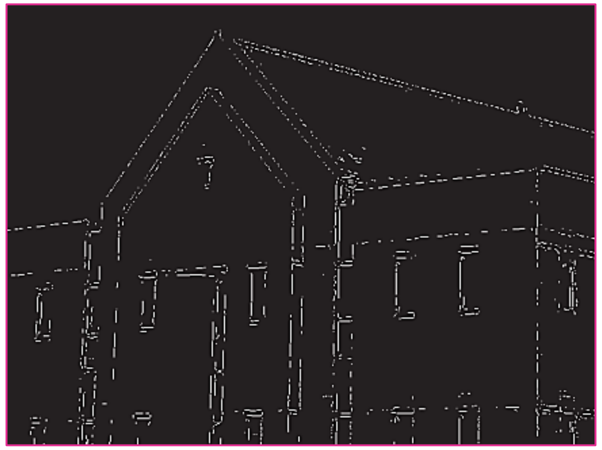
input



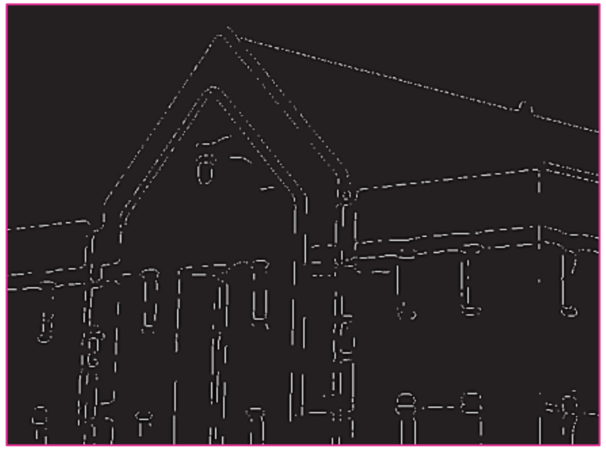
Sobel with TH



LoG zero crossings



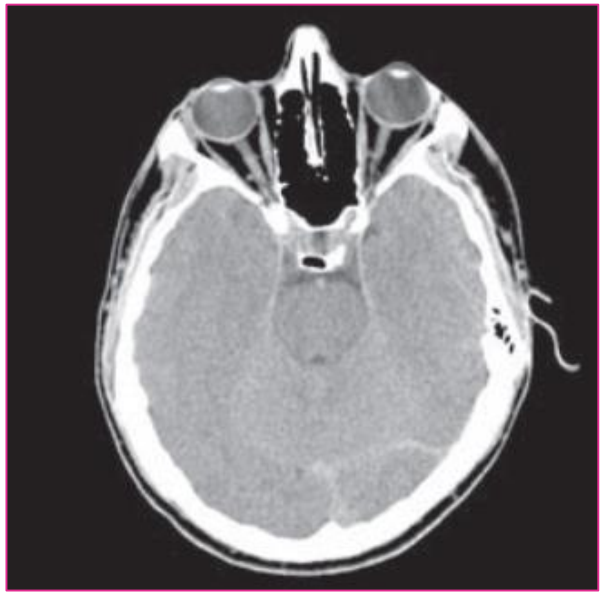
Canny



Canny edge detector

- Comparing other edge detectors

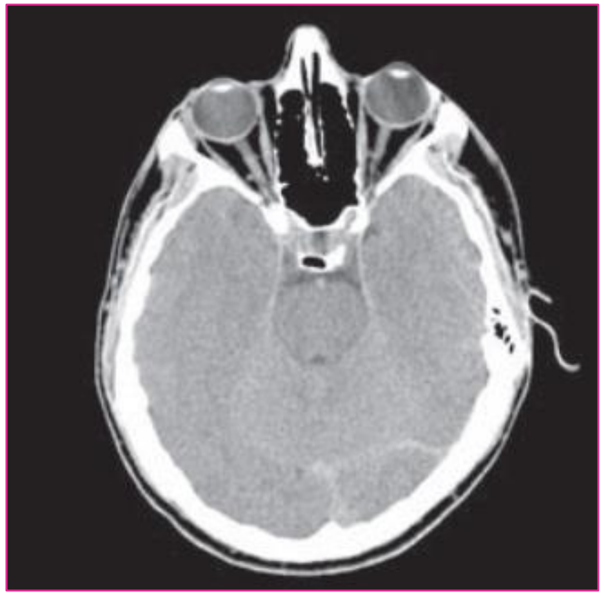
input



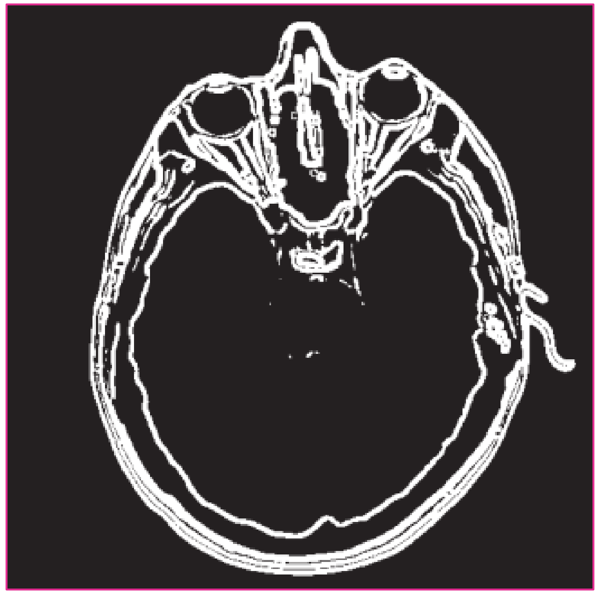
Canny edge detector

- Comparing other edge detectors

input



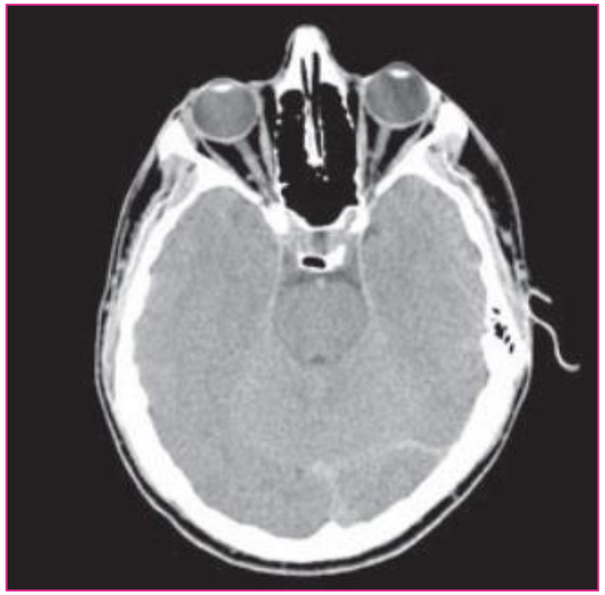
Sobel with TH



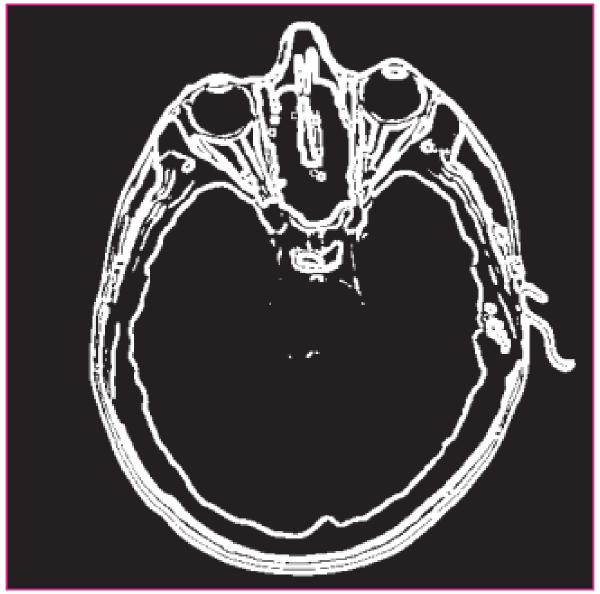
Canny edge detector

- Comparing other edge detectors

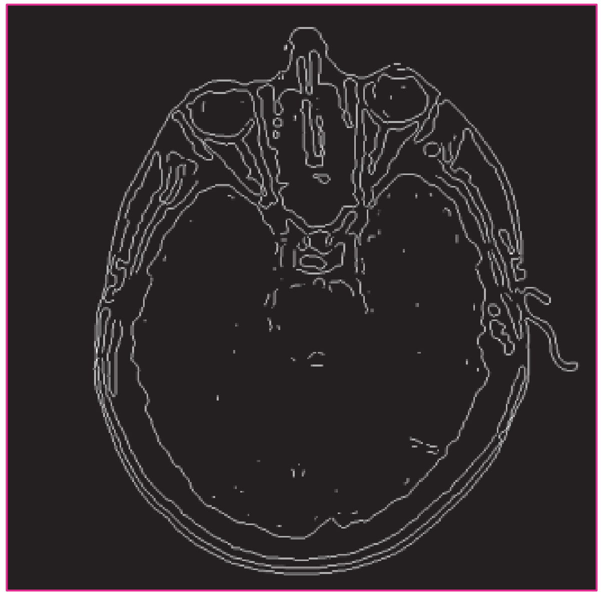
input



Sobel with TH



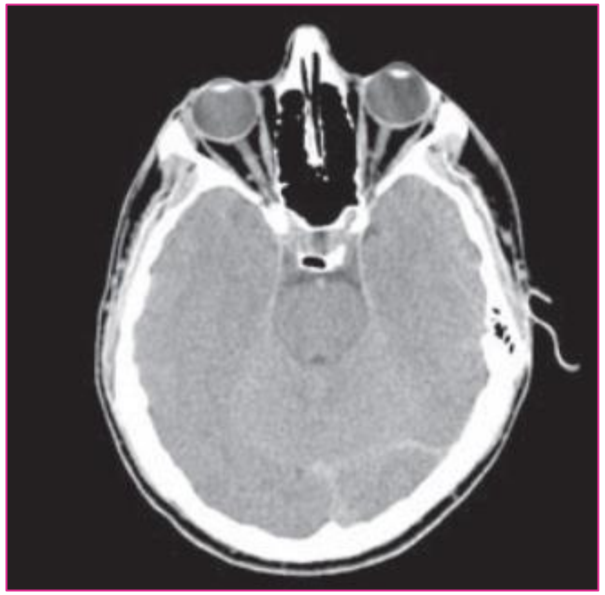
LoG zero crossings



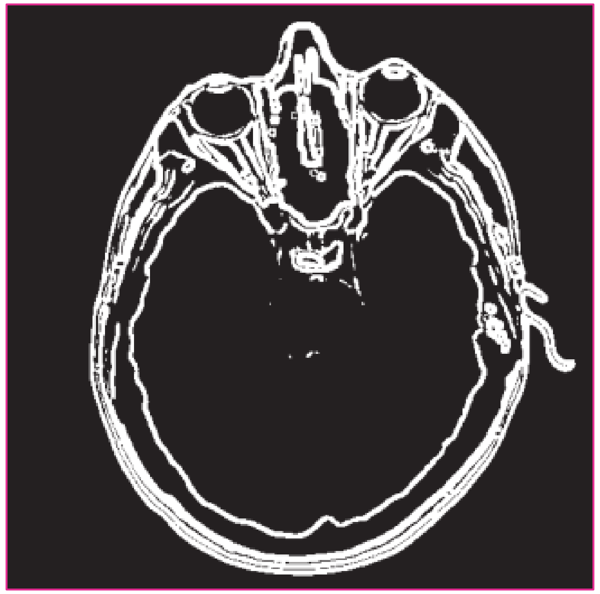
Canny edge detector

- Comparing other edge detectors

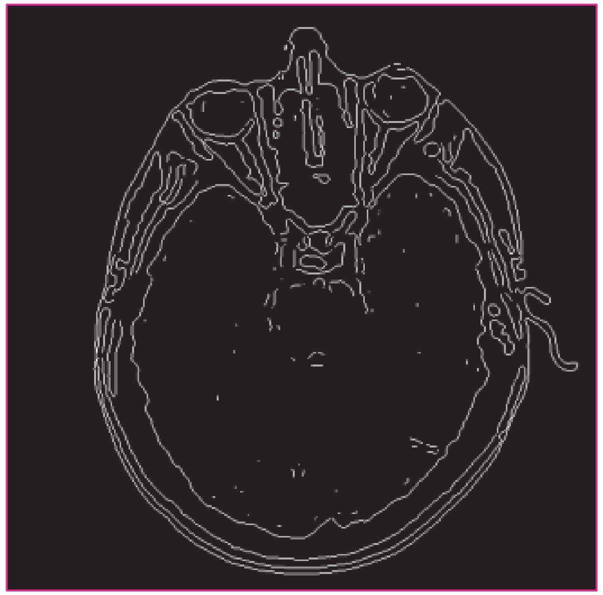
input



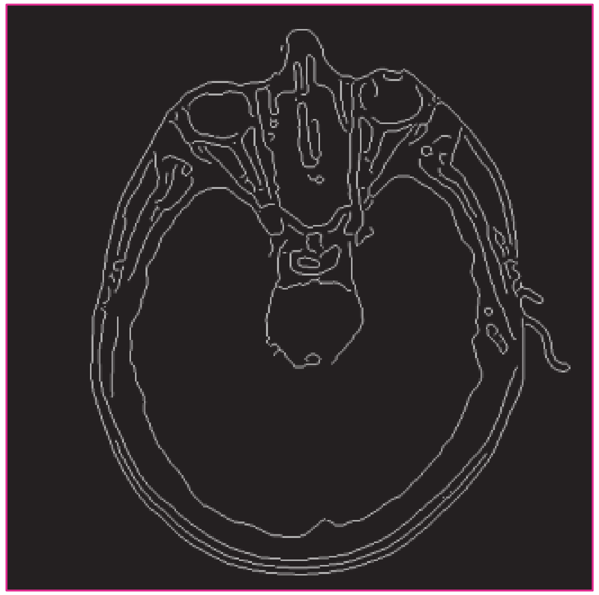
Sobel with TH



LoG zero crossings



Canny



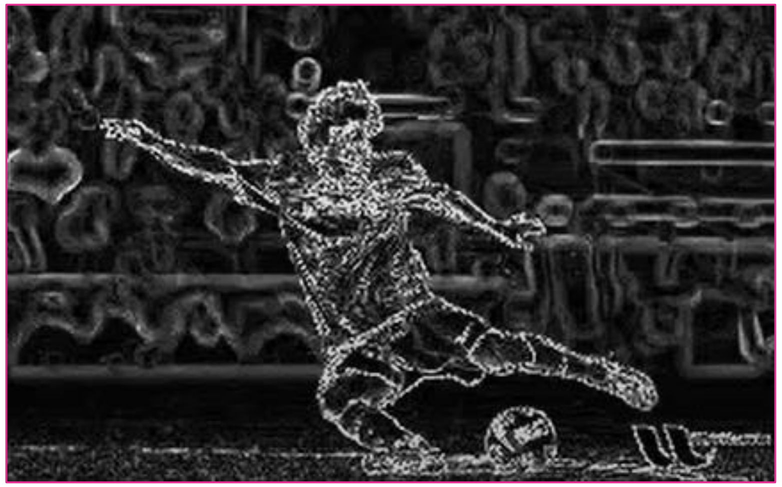
A canny player with a Canny edge!

A canny player with a Canny edge!

Messi

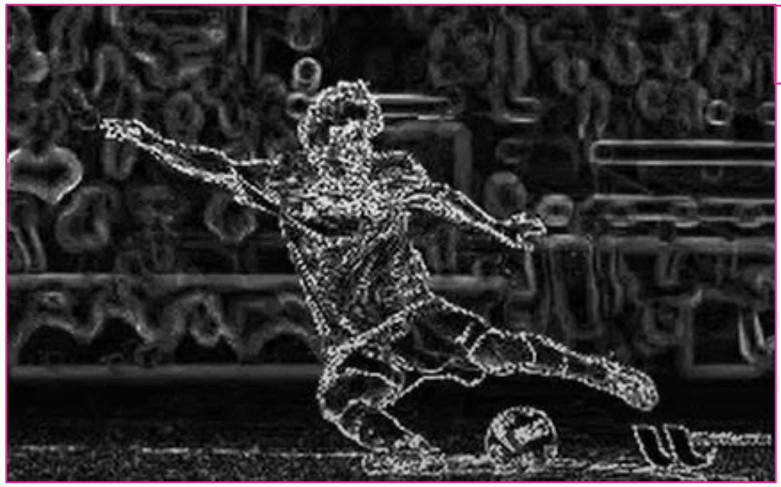
A canny player with a Canny edge!

Messi



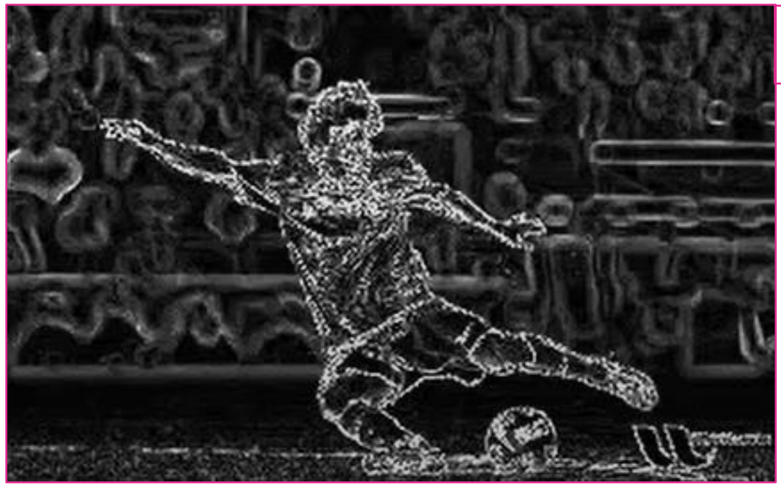
A canny player with a Canny edge!

Messi

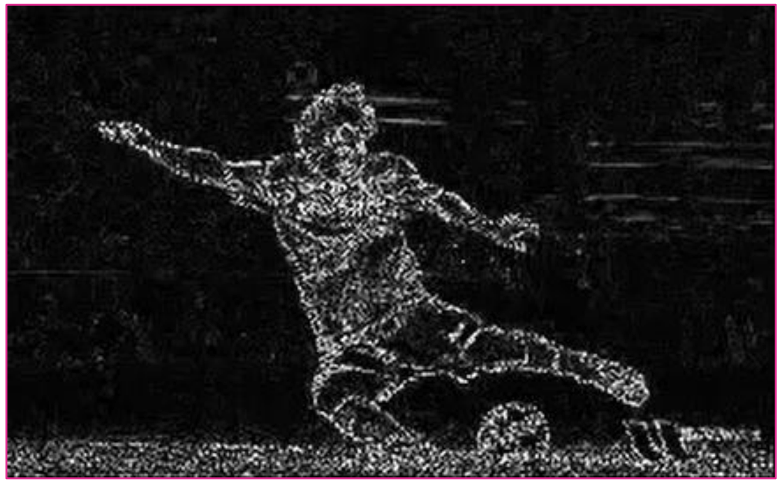

Sobel

A canny player with a Canny edge!

Messi

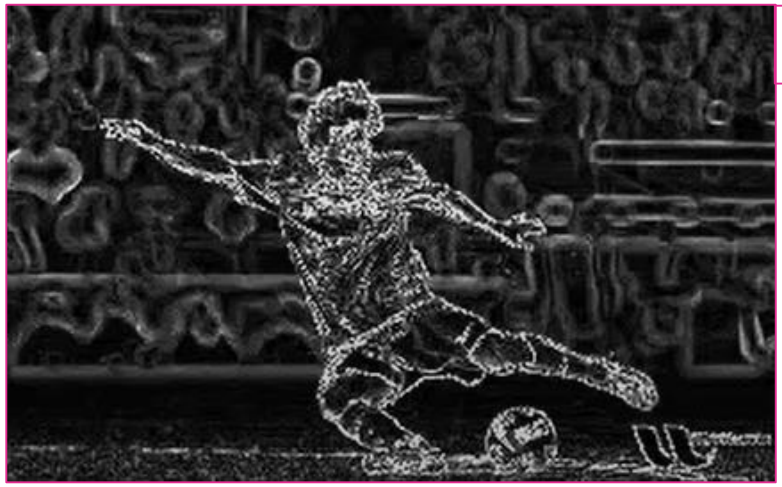


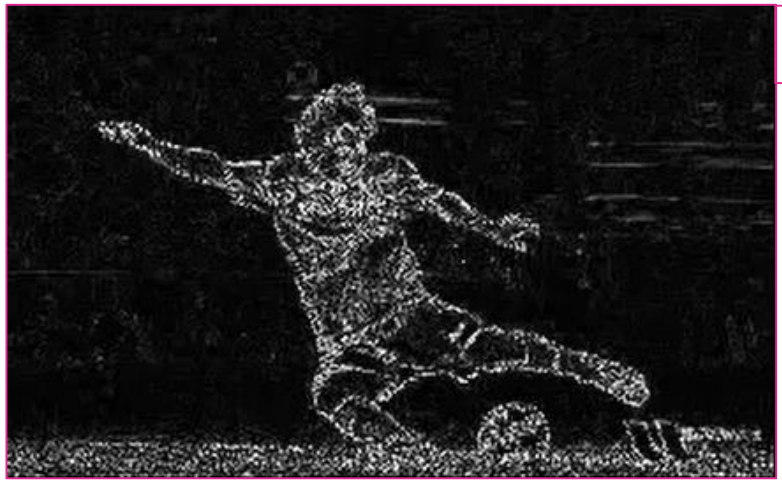
Sobel



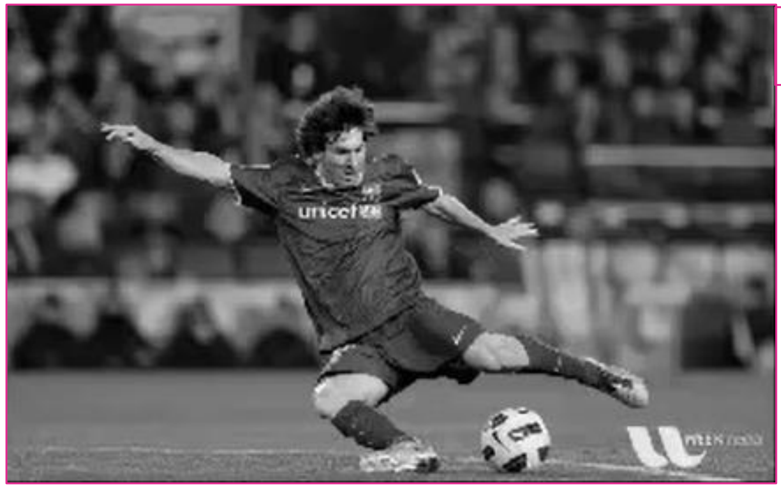
A canny player with a Canny edge!

Messi

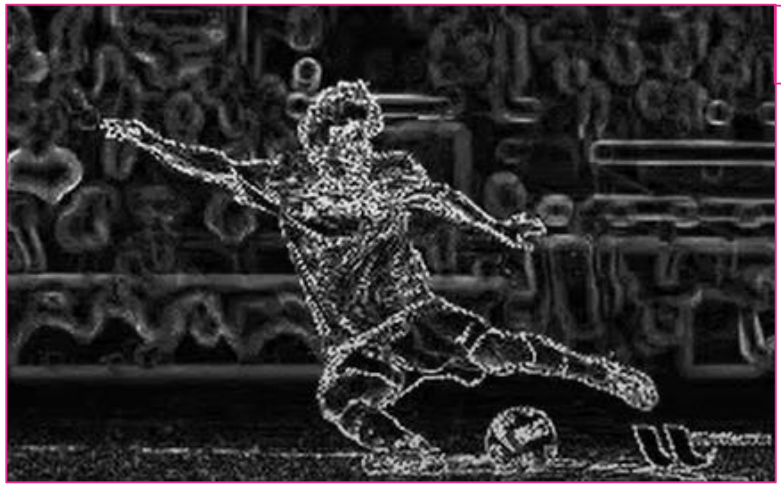

Sobel


Laplacian

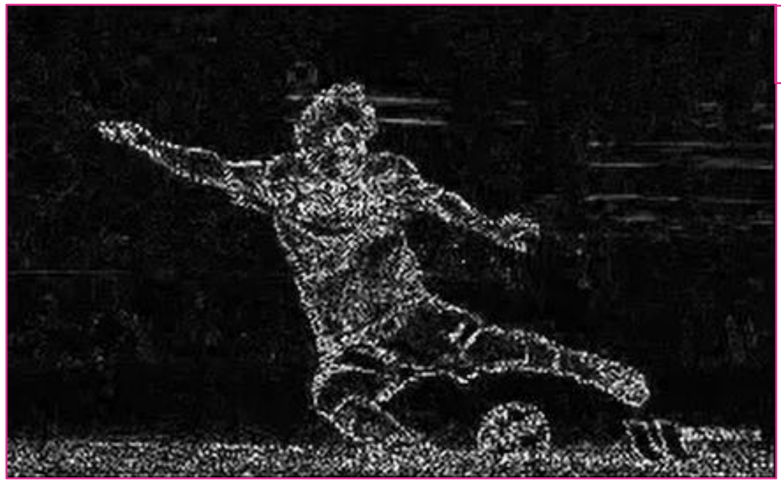
A canny player with a Canny edge!



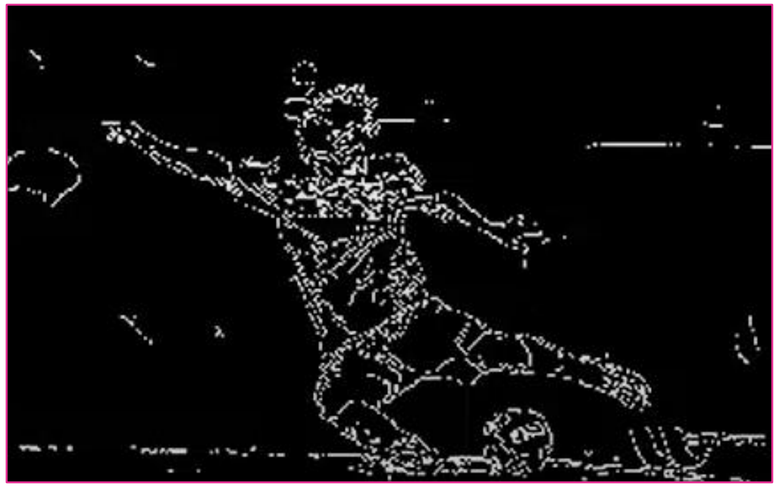
Messi



Sobel

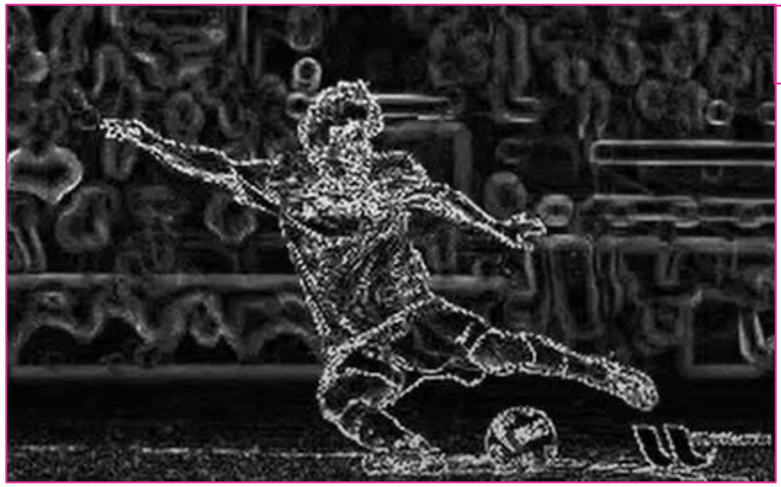


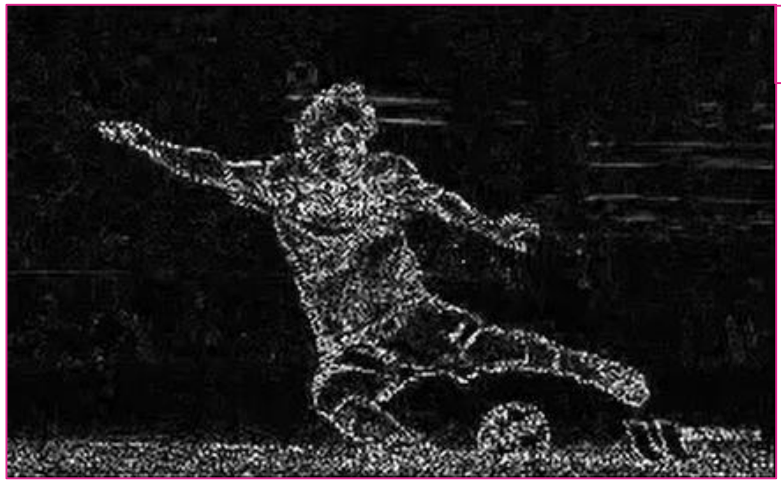
Laplacian

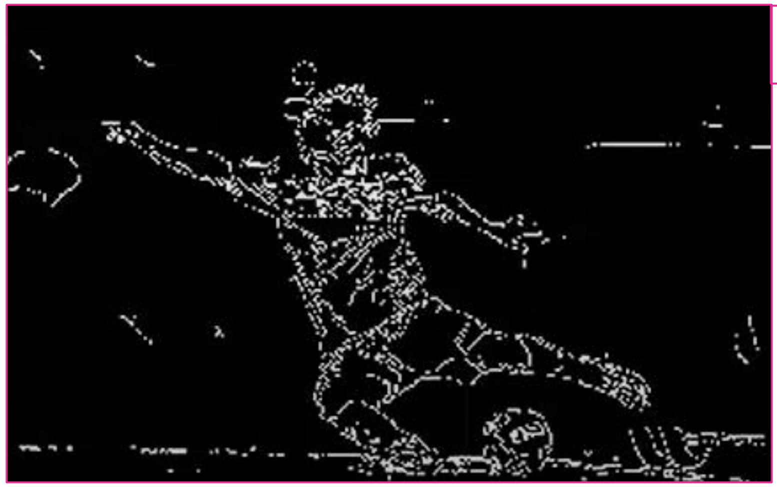


A canny player with a Canny edge!

Messi


Sobel

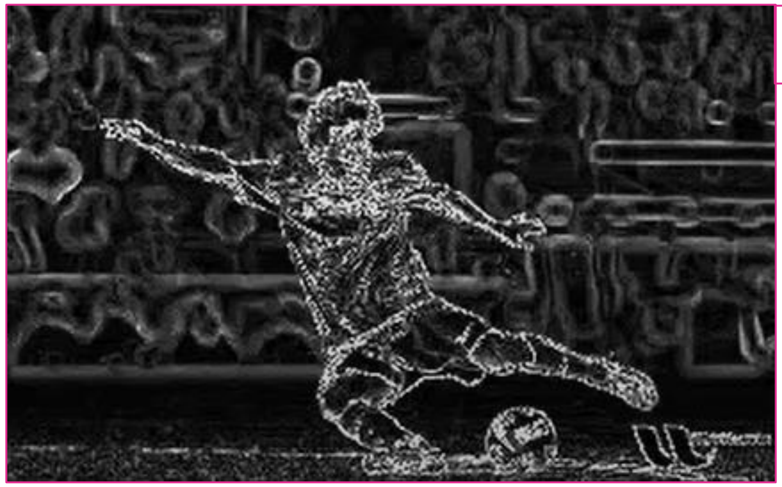

Laplacian

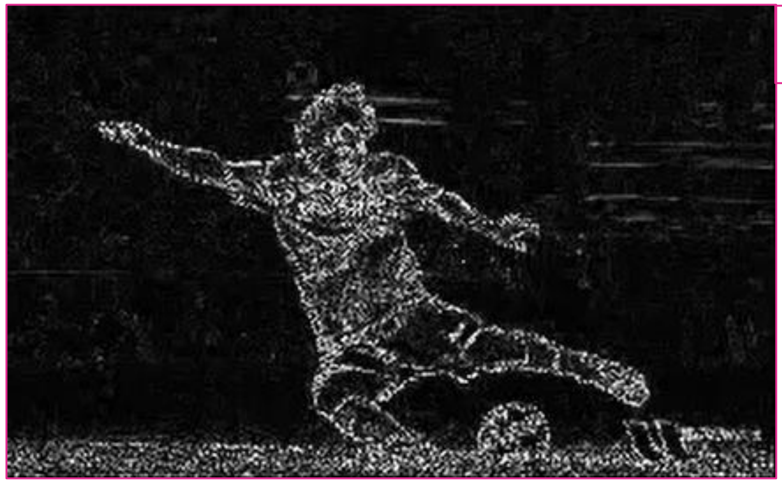

Canny

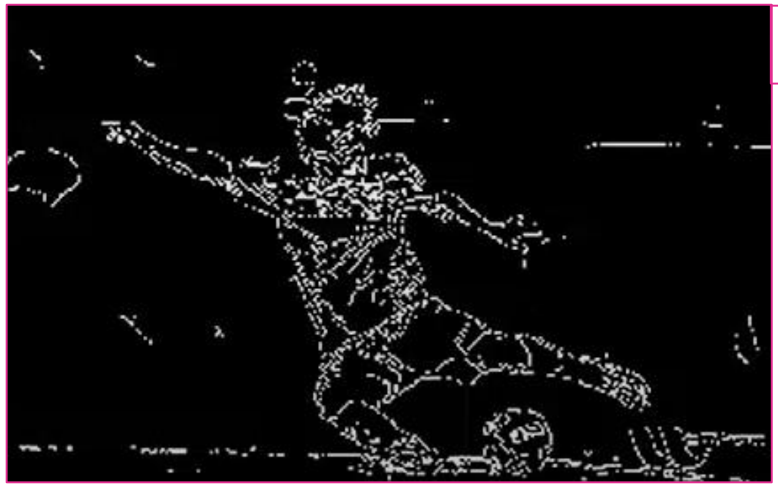
A canny player with a Canny edge!

Messi

who will you go with?


Sobel


Laplacian


Canny

Conclusion

- Canny edge detector

Conclusion

- Canny edge detector

❑ Single point thick edges

❑ Canny operations

- Thinning: non-max suppression
- Linking: double TH hysteresis
- High accuracy is paid via computational expenses

Conclusion

- Canny edge detector

- Single point thick edges

- Canny operations
  - Thinning: non-max suppression
  - Linking: double TH hysteresis
  - High accuracy is paid via computational expenses

Conclusion

- Canny edge detector

- Single point thick edges

- Canny operations
  - Thinning: non-max suppression
  - Linking: double TH hysteresis
  - High accuracy is paid via computational expenses

Go with a 'canny' player, for accuracy

Conclusion

- Canny edge detector

- Single point thick edges

- Canny operations
  - Thinning: non-max suppression
  - Linking: double TH hysteresis
  - High accuracy is paid via computational expenses

Go with a ‘canny’ player, for accuracy

And with uSΘin BΘLt, for speed!

Conclusion

- Canny edge detector

❑ Single point thick edges

❑ Canny operations
  - Thinning: non-max suppression
  - Linking: double TH hysteresis
  - High accuracy is paid via computational expenses

Go with a ‘canny’ player, for accuracy

And with uSΘin BΘLt, for speed!