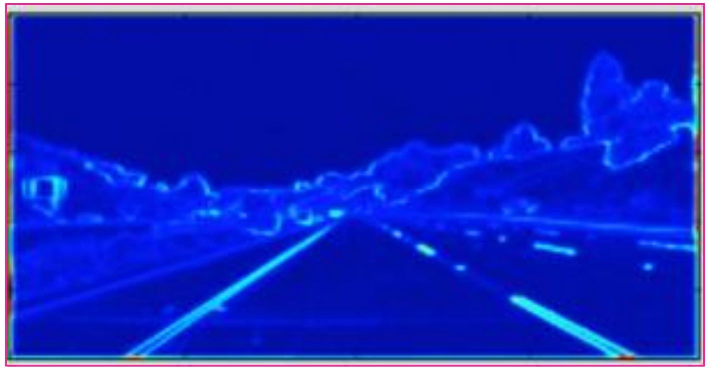
Edge:
Hough transform

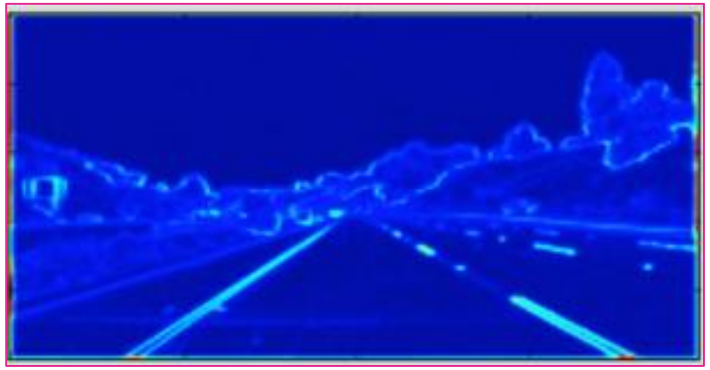
Dr. Tushar Sandhan

Introduction



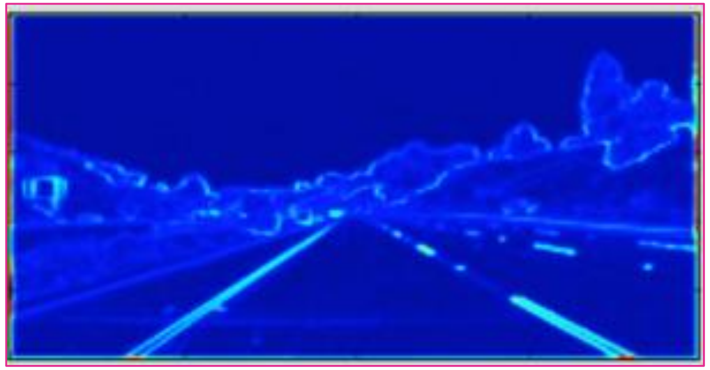
Introduction

- Edges so far (operators, Canny)
  - consider prior info about gradient behaviour
  - do not consider anything about object shape, structure
  - how did we get clear boundary edges previously?



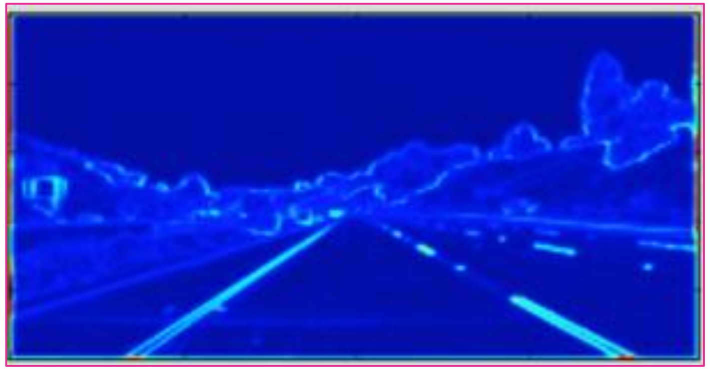
Introduction

- Edges so far (operators, Canny)
  - consider prior info about gradient behaviour
  - do not consider anything about object shape, structure
  - how did we get clear boundary edges previously?



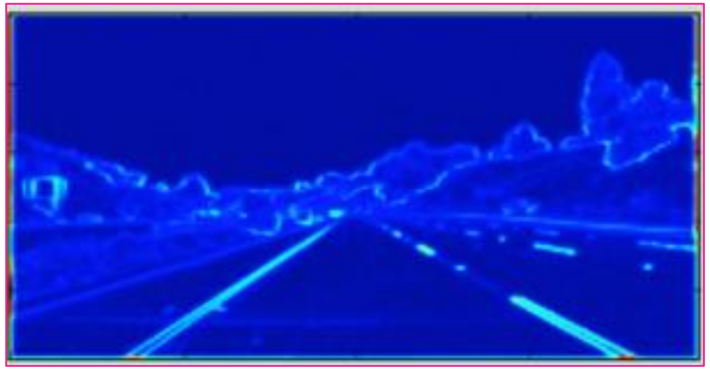
Introduction

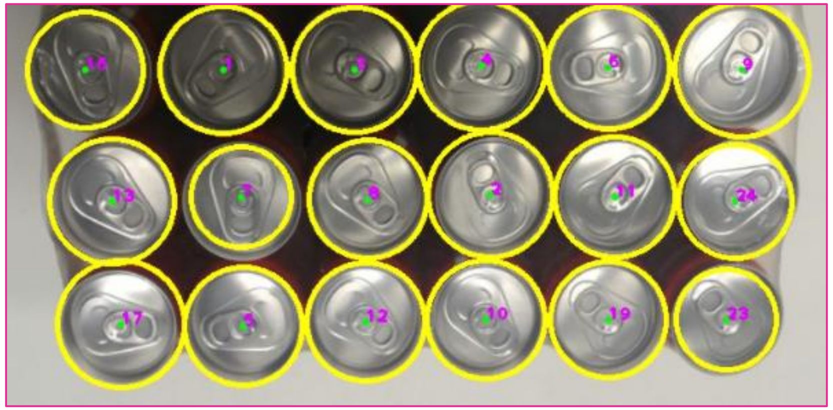
- Edges so far (operators, Canny)
  - consider prior info about gradient behaviour
  - do not consider anything about object shape, structure
  - how did we get clear boundary edges previously?
    - ♪ • Linking



Introduction

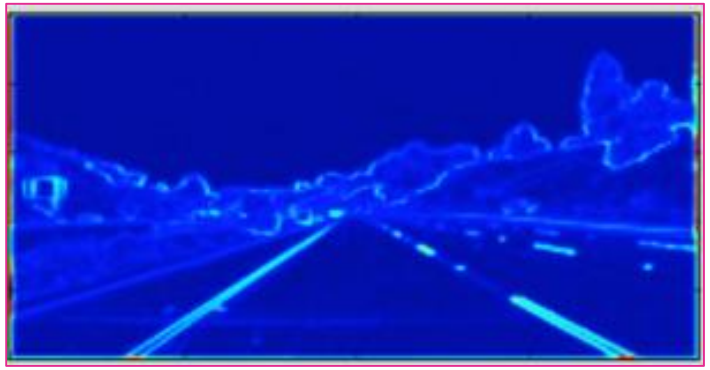
- Edges so far (operators, Canny)
  - consider prior info about gradient behaviour
  - do not consider anything about object shape, structure
  - how did we get clear boundary edges previously?
    - Linking

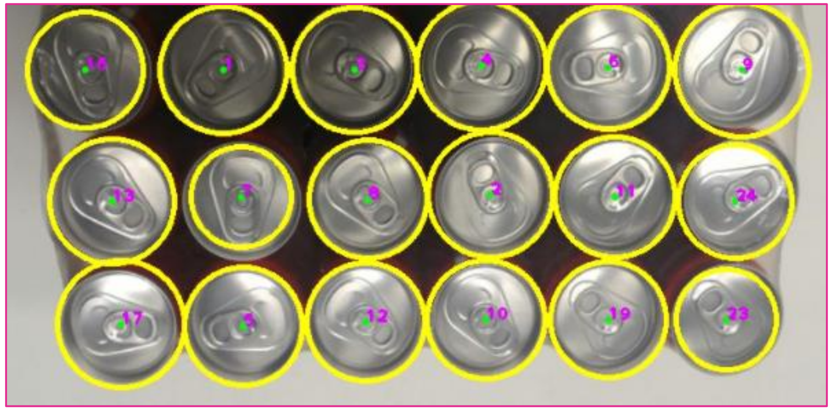


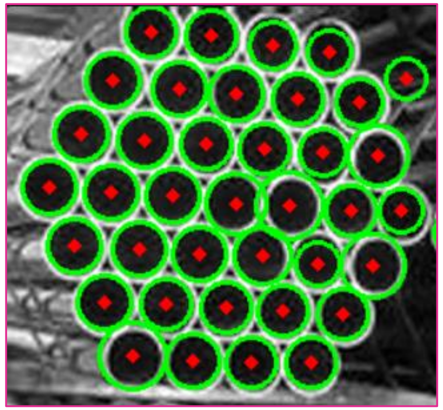


Introduction

- Edges so far (operators, Canny)
  - consider prior info about gradient behaviour
  - do not consider anything about object shape, structure
  - how did we get clear boundary edges previously?
    - Linking

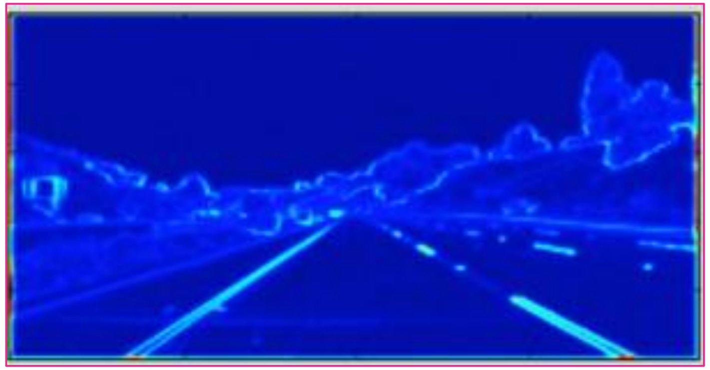


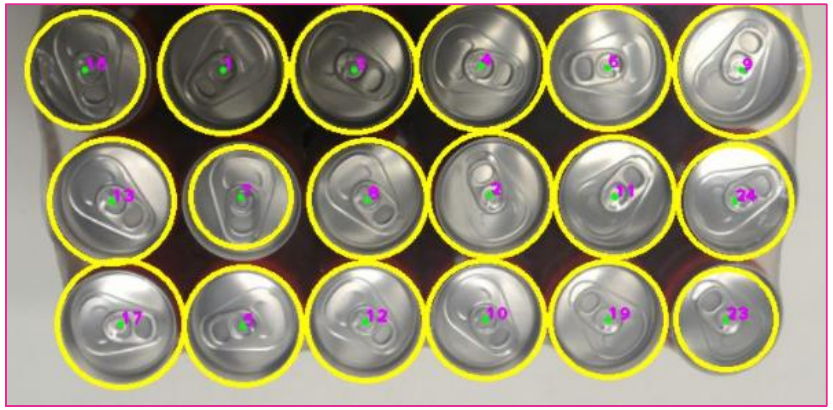


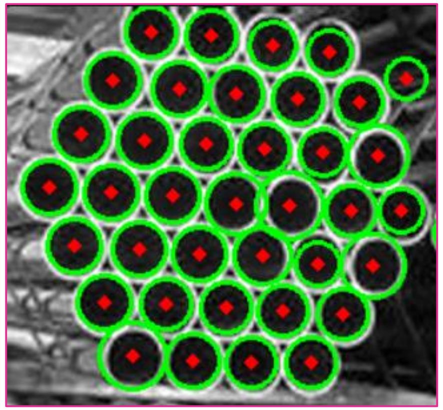


Introduction

- Edges so far (operators, Canny)
  - consider prior info about gradient behaviour
  - do not consider anything about object shape, structure
  - how did we get clear boundary edges previously?
    - Linking

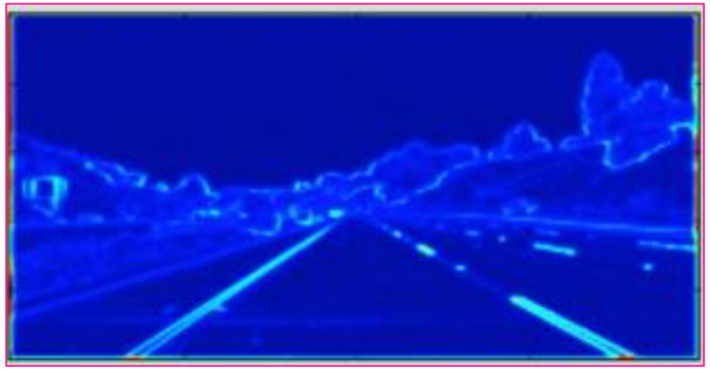


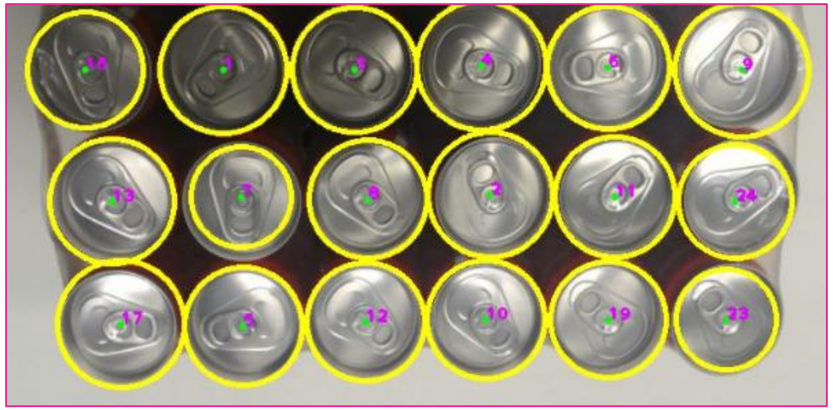


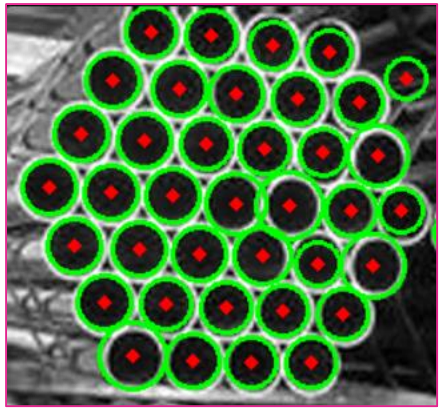


Introduction

- Edges so far (operators, Canny)
  - consider prior info about gradient behaviour
  - do not consider anything about object shape, structure
  - how did we get clear boundary edges previously?
    - Linking







Hough transform

- Image derivatives
  - input image $f(x, y)$
  - (optional) smoothed $f_s(x, y)$
  - get the gradients $g_x(x, y)$, $g_y(x, y)$
  - get thresholded edge map M_T

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

Hough transform

- Image derivatives
  - input image $f(x,y)$
  - (optional) smoothed $f_s(x,y)$
  - get the gradients $g_x(x,y)$, $g_y(x,y)$
  - get thresholded edge map M_T

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

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

Hough transform

- Image derivatives
  - input image $f(x, y)$
  - (optional) smoothed $f_s(x, y)$
  - get the gradients $g_x(x, y)$, $g_y(x, y)$
  - get thresholded edge map M_T

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

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

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

Hough transform

- Image derivatives
  - input image $f(x, y)$
  - (optional) smoothed $f_s(x, y)$
  - get the gradients $g_x(x, y)$, $g_y(x, y)$
  - get thresholded edge map M_T

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

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

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

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

Hough transform

- Image derivatives
  - input image $f(x, y)$
  - (optional) smoothed $f_s(x, y)$
  - get the gradients $g_x(x, y)$, $g_y(x, y)$
  - get thresholded edge map M_T

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

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

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

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

Hough transform

Hough transform

- Hough transform (HT)
  - considers shape of the object as prior info.
  - shape is defined as a function and parametrized

Hough transform

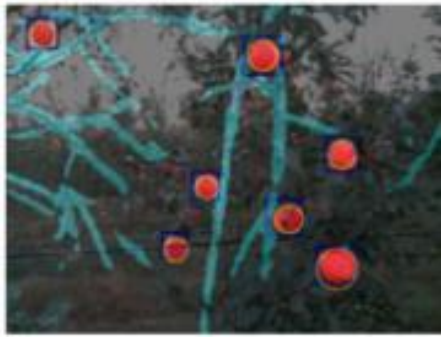
- Hough transform (HT)
  - considers shape of the object as prior info.
  - shape is defined as a function and parametrized

- Shape detection
  - plat fruits plucking autonomous robots

Hough transform

- Hough transform (HT)
  - considers shape of the object as prior info.
  - shape is defined as a function and parametrized

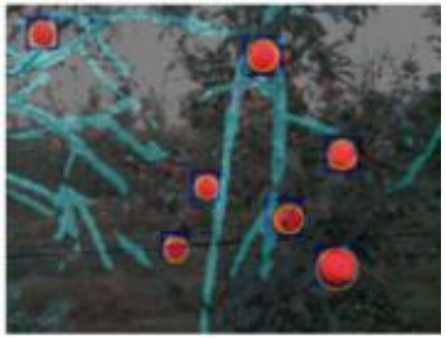
- Shape detection
  - plat fruits plucking autonomous robots



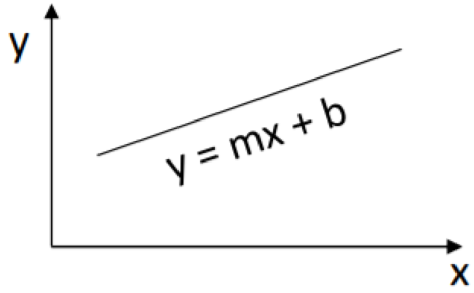
Hough transform

- Hough transform (HT)
  - considers shape of the object as prior info.
  - shape is defined as a function and parametrized

- Shape detection
  - plat fruits plucking autonomous robots



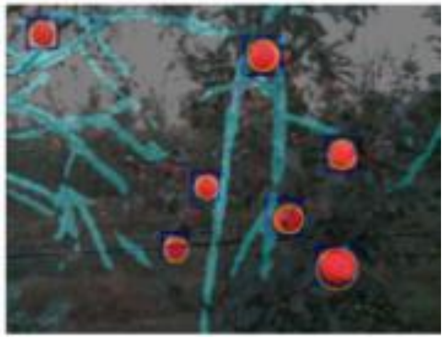
- Lines



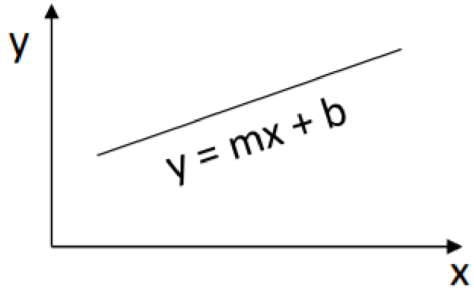
Hough transform

- Hough transform (HT)
  - considers shape of the object as prior info.
  - shape is defined as a function and parametrized

- Shape detection
  - plat fruits plucking autonomous robots



- Lines

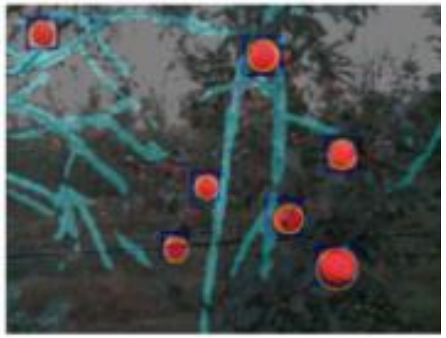


- Hough Transform

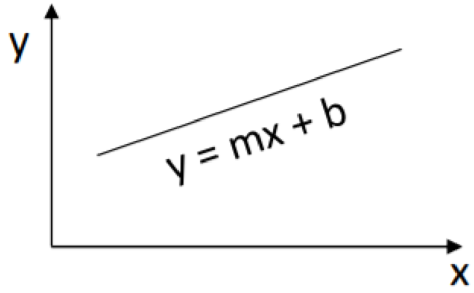
Hough transform

- Hough transform (HT)
  - considers shape of the object as prior info.
  - shape is defined as a function and parametrized

- Shape detection
  - plat fruits plucking autonomous robots



- Lines

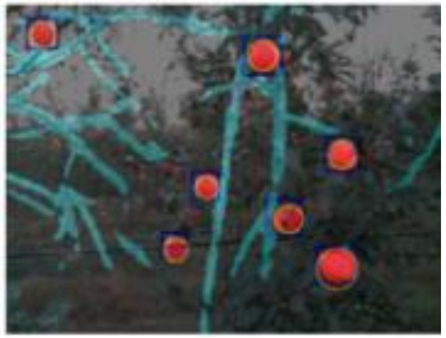


- Hough Transform
  - a space of parameters

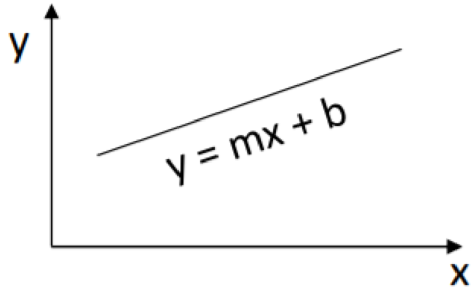
Hough transform

- Hough transform (HT)
  - considers shape of the object as prior info.
  - shape is defined as a function and parametrized

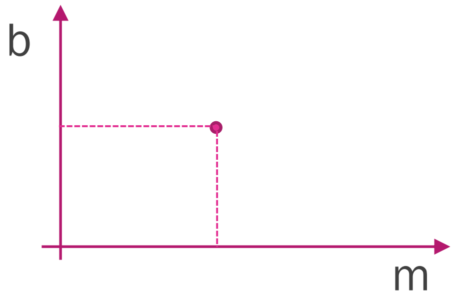
- Shape detection
  - plat fruits plucking autonomous robots



- Lines



- Hough Transform
  - a space of parameters

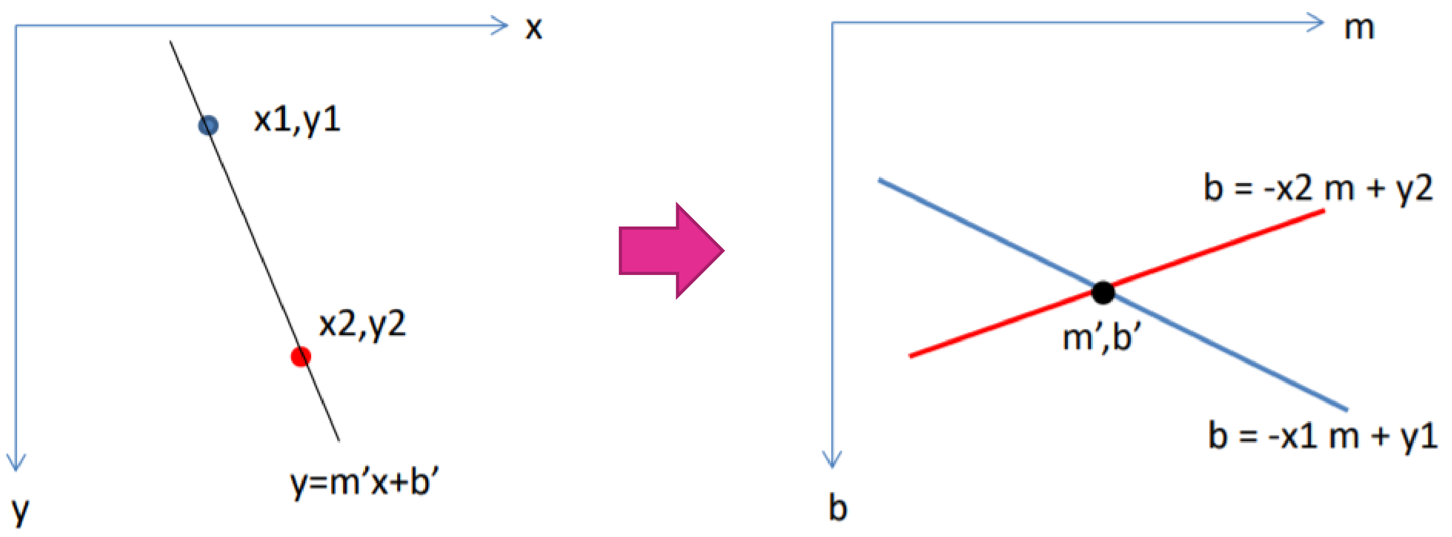


Hough transform

- Hough transform duality
  - Lines in the image space becomes a point in the Hough space
  - A point in the image space becomes ____ in the Hough space

Hough transform

- Hough transform duality
  - Lines in the image space becomes a point in the Hough space
  - A point in the image space becomes ____ in the Hough space



Hough transform

- Hough space voting

courtesy: W. Hoff

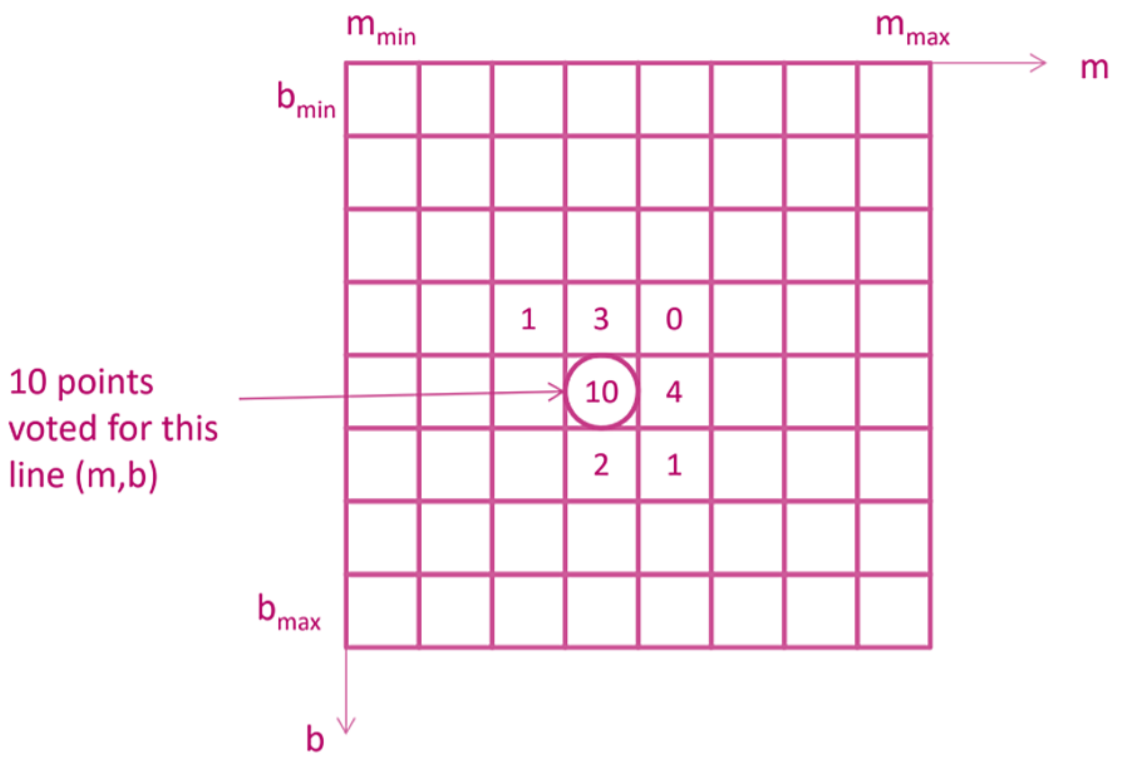
Hough transform

- Hough space voting
  - initialize accumulator $A(m,b) \rightarrow 0$

courtesy: W. Hoff

Hough transform

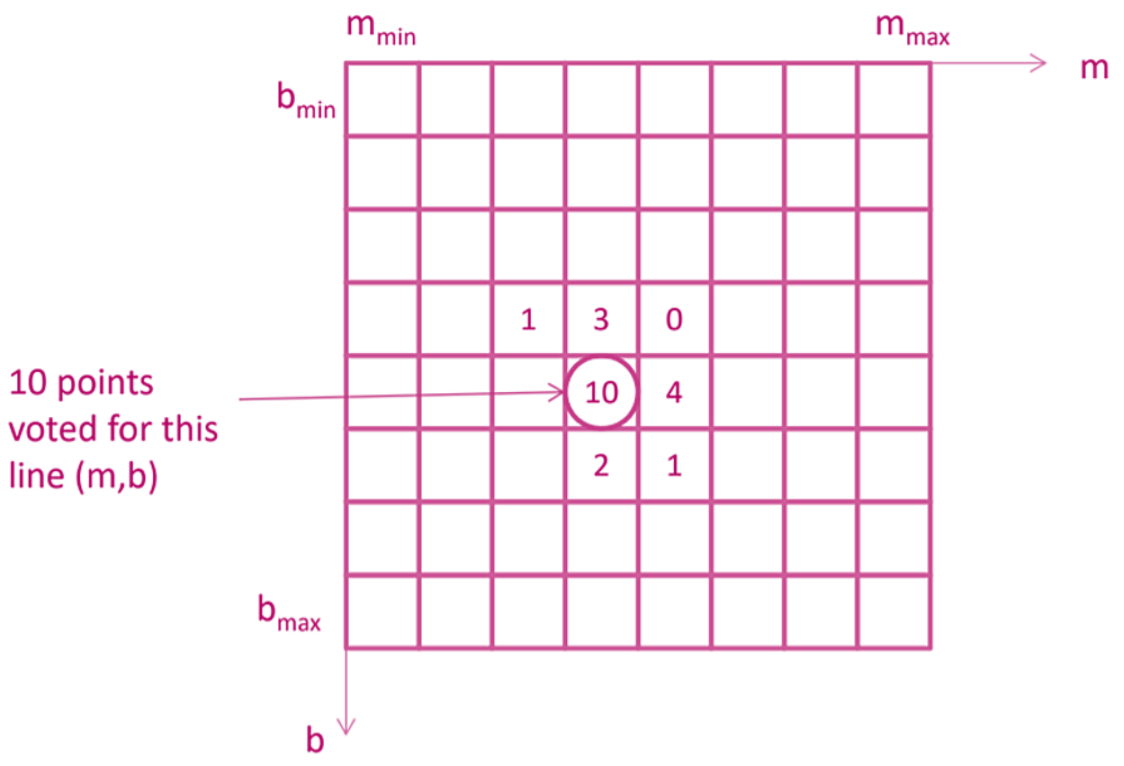
- Hough space voting
  - initialize accumulator A(m,b) → 0



courtesy: W. Hoff

Hough transform

- Hough space voting
  - initialize accumulator A(m,b) → 0
  - for each edge element,
    increment all cells that satisfy $b = -xm + y$

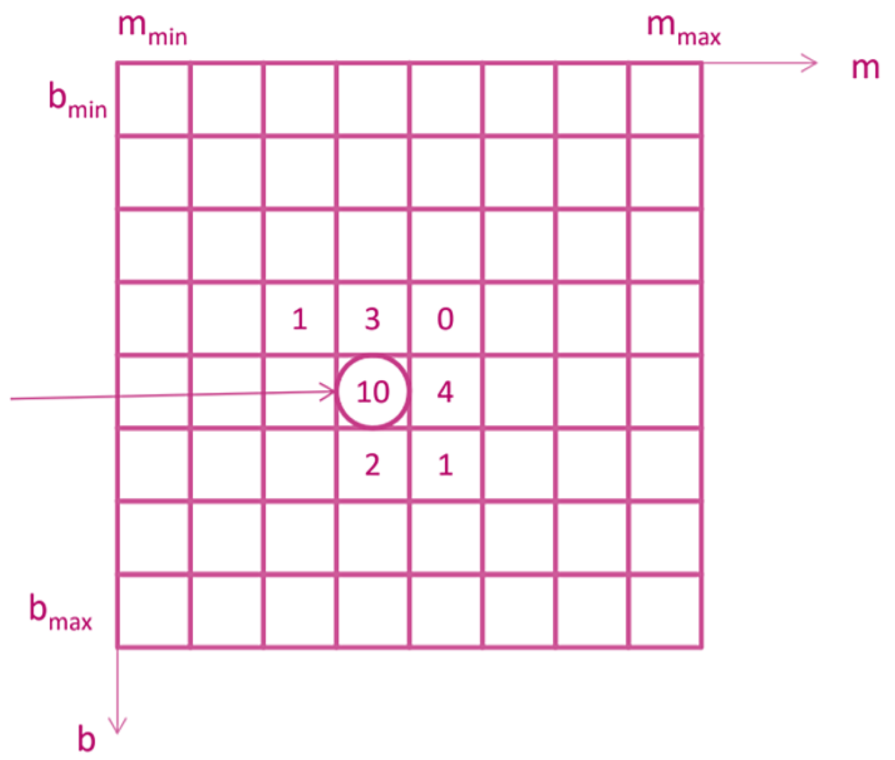


courtesy: W. Hoff

Hough transform

- Hough space voting
  - initialize accumulator A(m,b) → 0
  - for each edge element,
    increment all cells that satisfy $b = -xm + y$
  - local maxima in A(m,b) correspond to lines

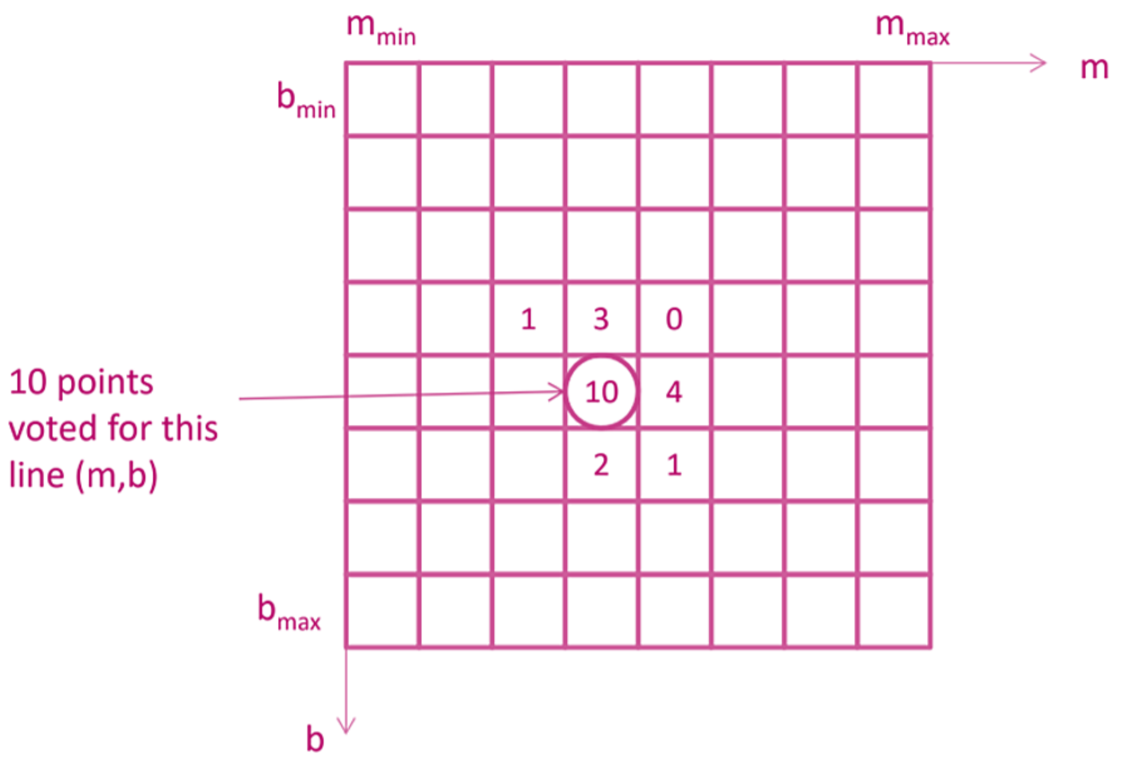
10 points voted for this line (m,b)



courtesy: W. Hoff

Hough transform

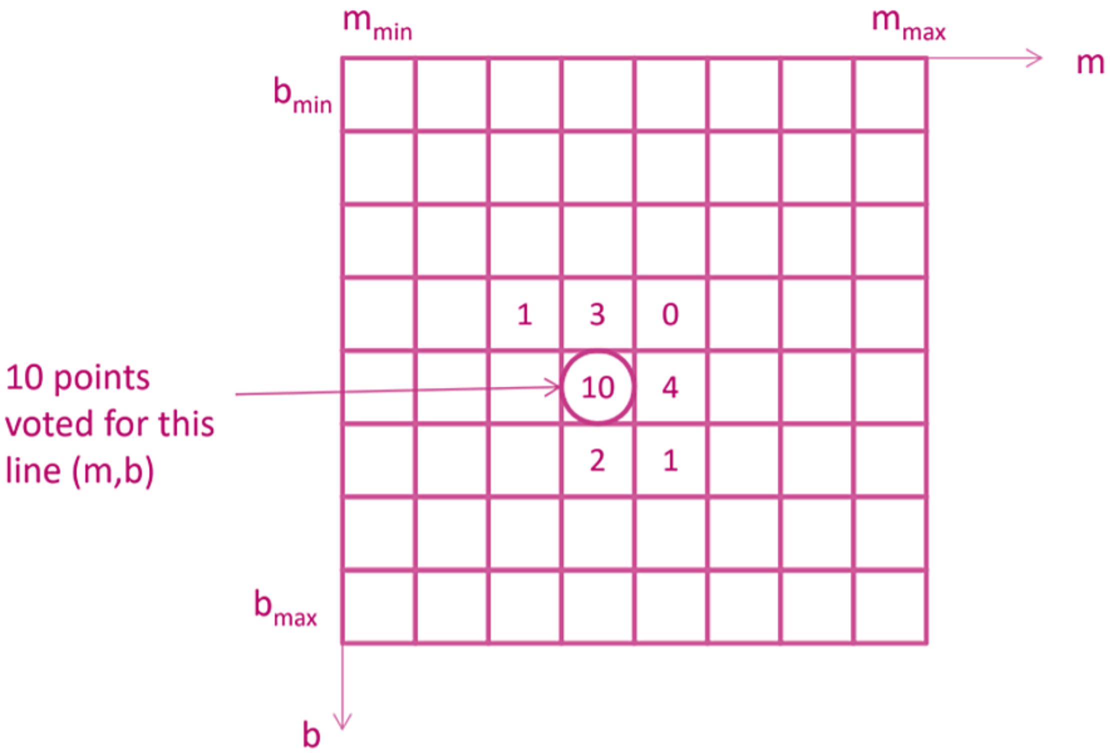
- Hough space voting
  - initialize accumulator A(m,b) → 0
  - for each edge element,
    increment all cells that satisfy $b = -xm + y$
  - local maxima in A(m,b) correspond to lines
    - is there any issue here?



courtesy: W. Hoff

Hough transform

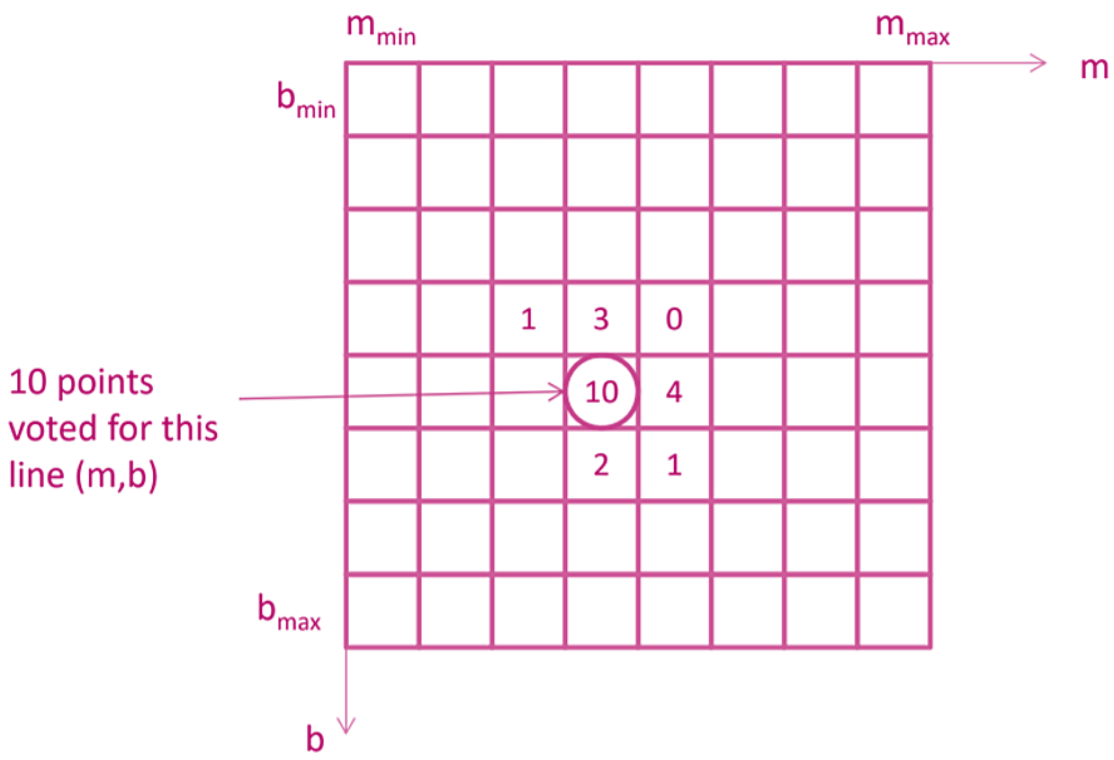
- Hough space voting
  - initialize accumulator A(m,b) → 0
  - for each edge element,
    increment all cells that satisfy $b = -xm + y$
  - local maxima in A(m,b) correspond to lines
    - is there any issue here?
    -



courtesy: W. Hoff

Hough transform

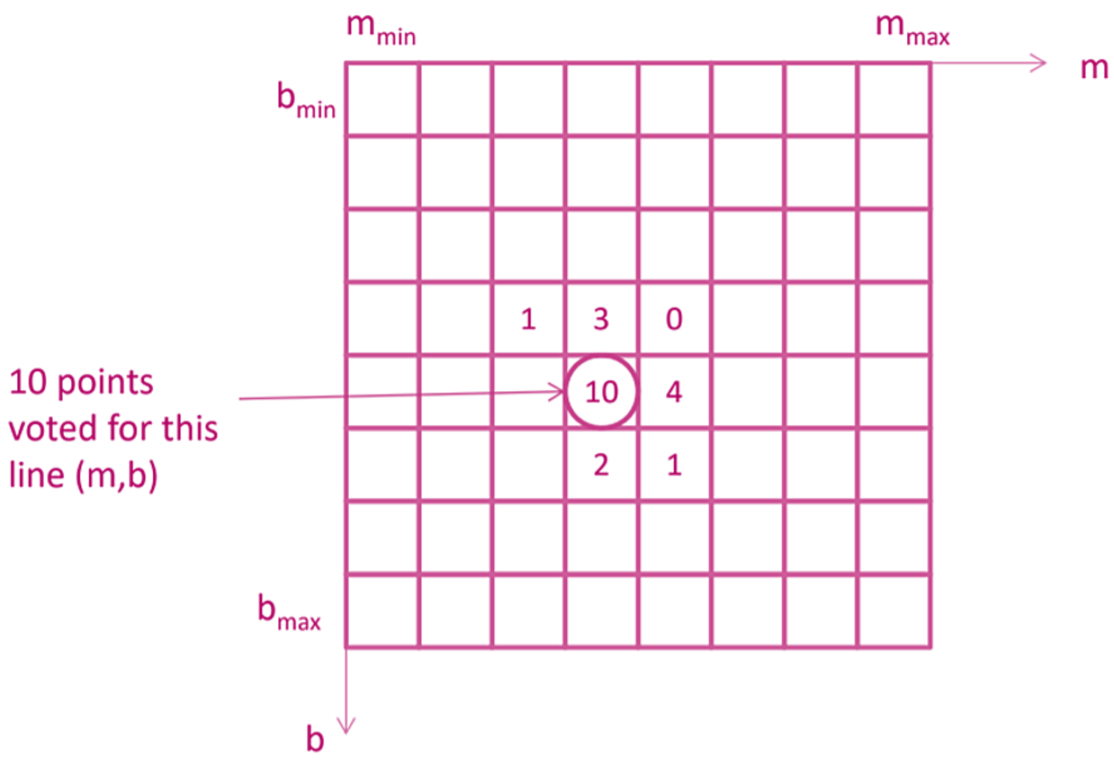
- Hough space voting
  - initialize accumulator A(m,b) → 0
  - for each edge element,
    increment all cells that satisfy $b = -xm + y$
  - local maxima in A(m,b) correspond to lines
    - is there any issue here?
    - for vertical lines



courtesy: W. Hoff

Hough transform

- Hough space voting
  - initialize accumulator A(m,b) → 0
  - for each edge element,
    increment all cells that satisfy $b = -xm + y$
  - local maxima in A(m,b) correspond to lines
    - is there any issue here?
    - for vertical lines
      - m → ∞



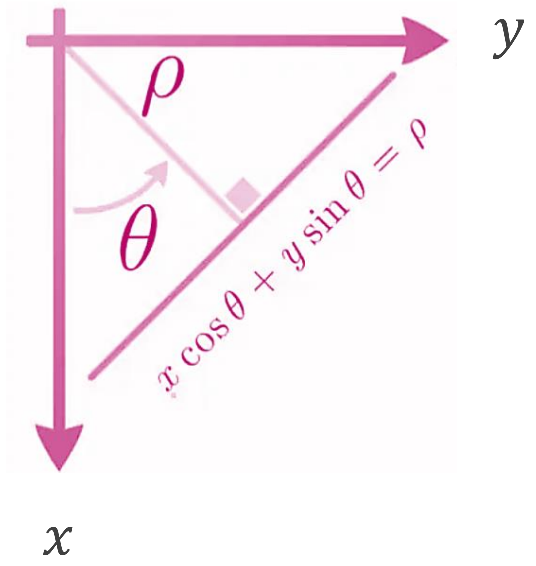
courtesy: W. Hoff

Hough transform

- Horizontal lines
  - $\theta = 0^o$

Hough transform

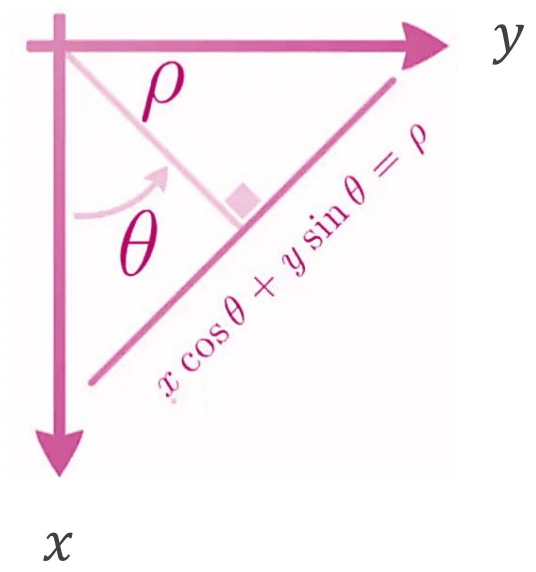
- Horizontal lines
  - $\theta = 0^o$



Hough transform

- Horizontal lines
  - $\theta = 0^o$

- Vertical lines
  - $\theta = 90^o$

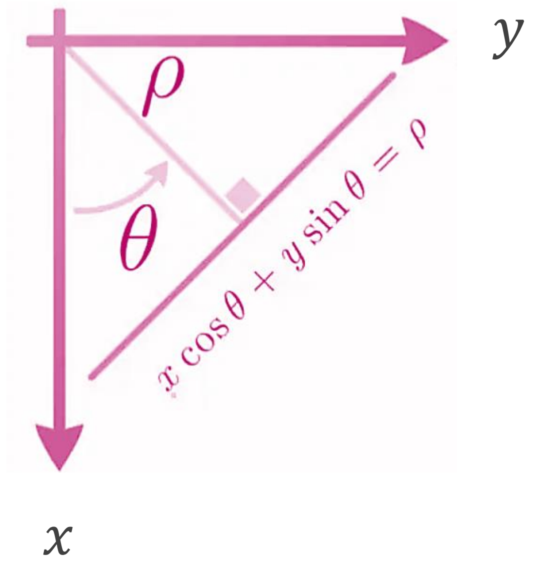


Hough transform

- Horizontal lines
  - $\theta = 0^o$

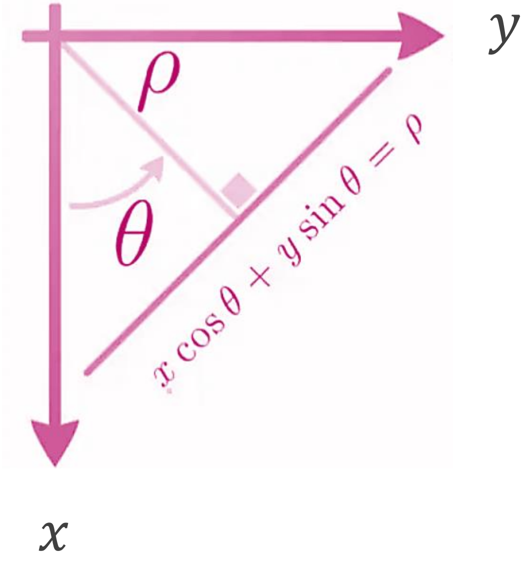
- Vertical lines
  - $\theta = 90^o$

- Ranges:
  - $\theta \in [-90^o, 90)$
  - $\rho \in [-\text{dmax}, +\text{dmax}]$
  - dmax?



Hough transform

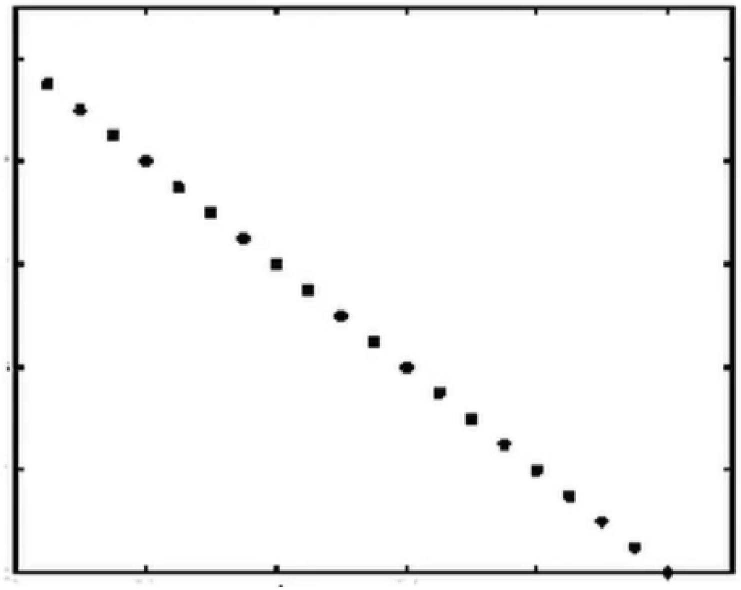
- Horizontal lines
  - $\theta = 0^o$
- Vertical lines
  - $\theta = 90^o$
- Ranges:
  - $\theta \in [-90^o, 90)$
  - $\rho \in [-\text{dmax}, +\text{dmax}]$
  - dmax?



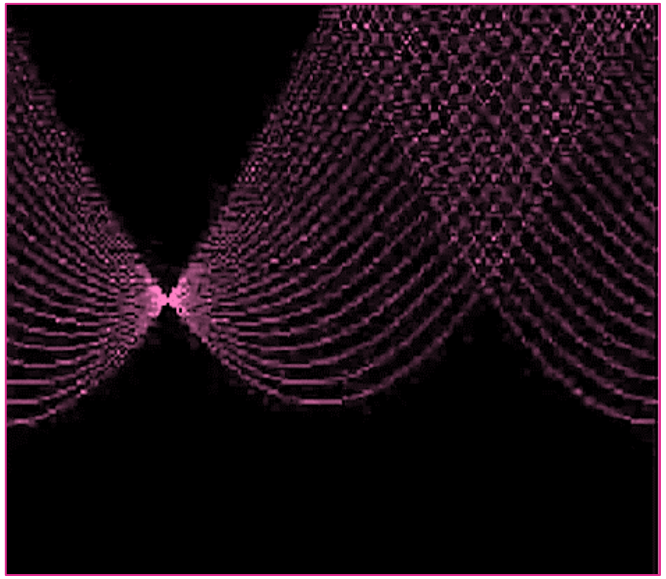
Hough transform

- HT example:

y



x

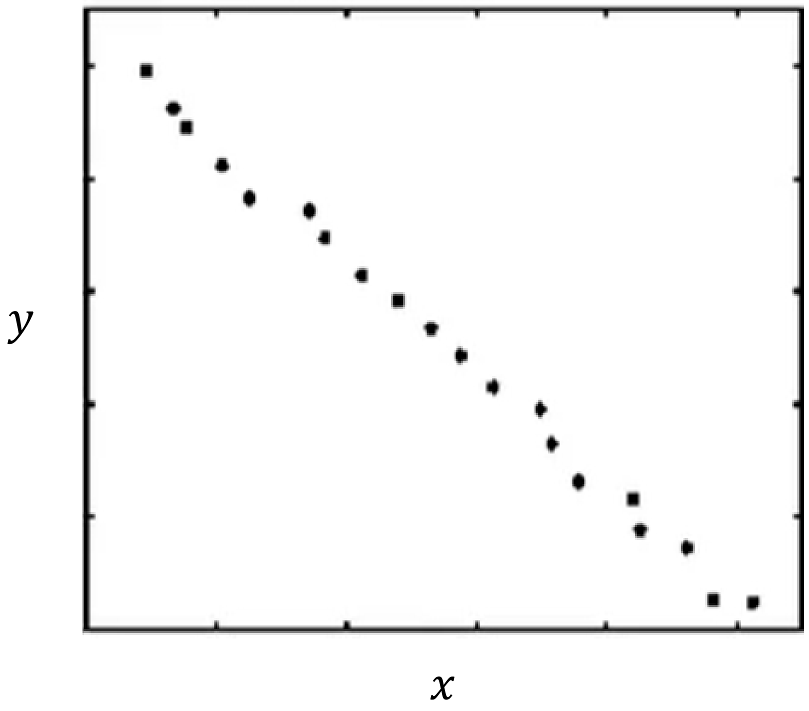


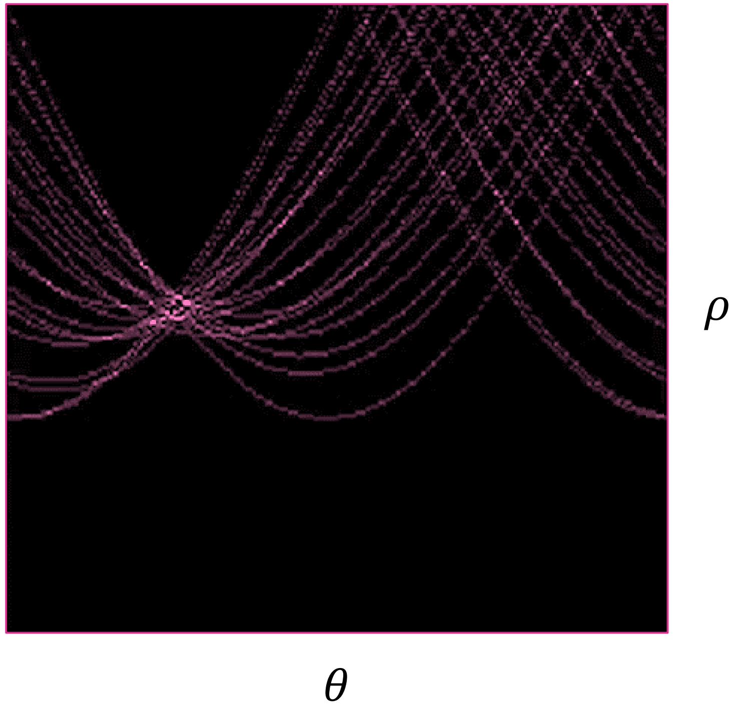
ρ

θ

Hough transform

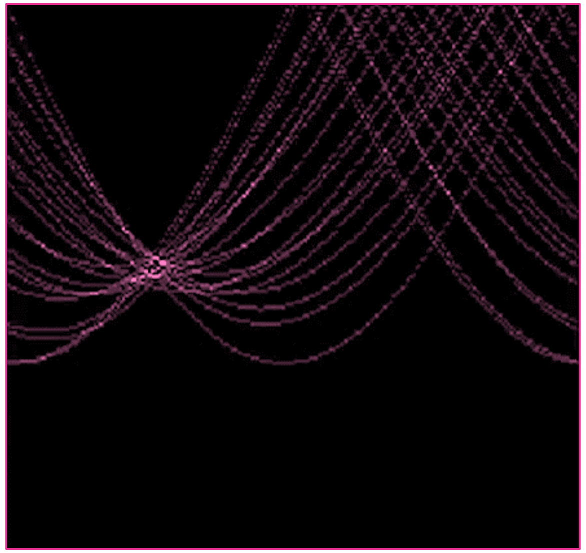
- HT example: with a greater noise





Hough transform

- HT example: with a greater noise
  - tackling the noise

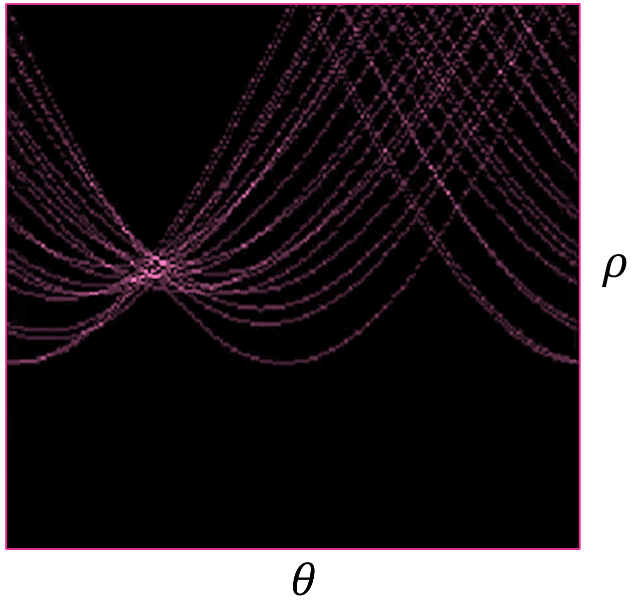


ρ

θ

Hough transform

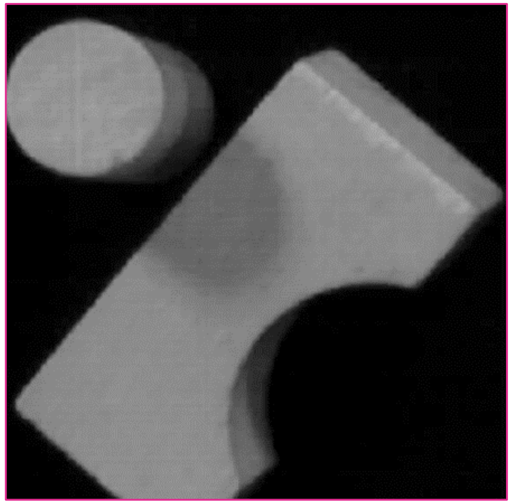
- HT example: with a greater noise
  - tackling the noise



- Image processing in the Hough space
  - smoothing
  - thresholding
  - zoom in the space
  - re-quantize zoomed space
  - redo HT in zoomed space

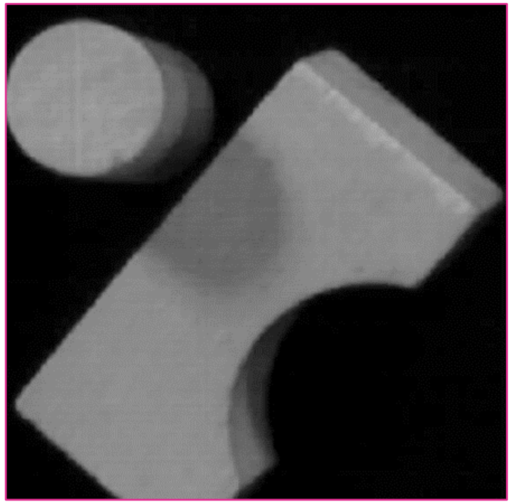
Hough transform

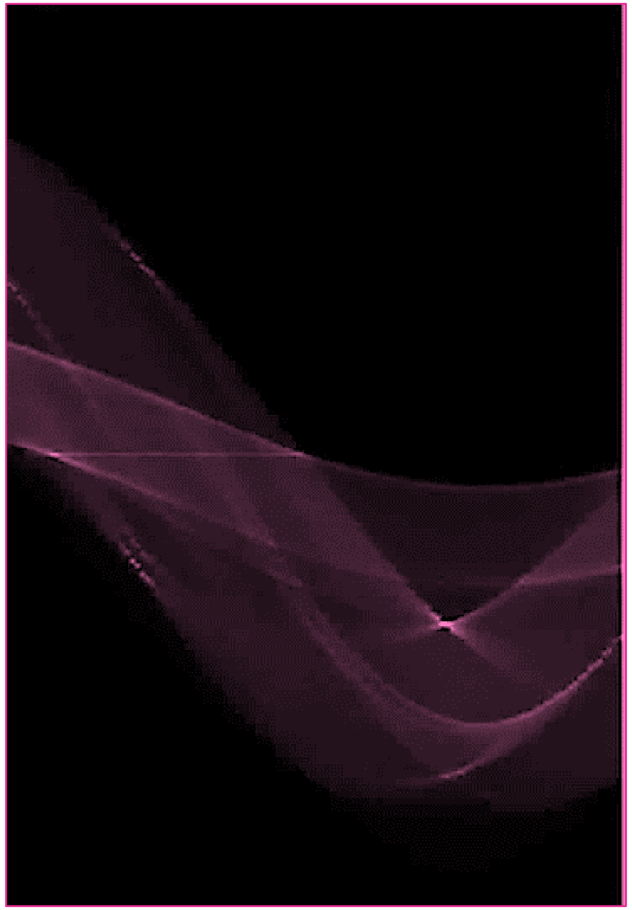
input



Hough transform

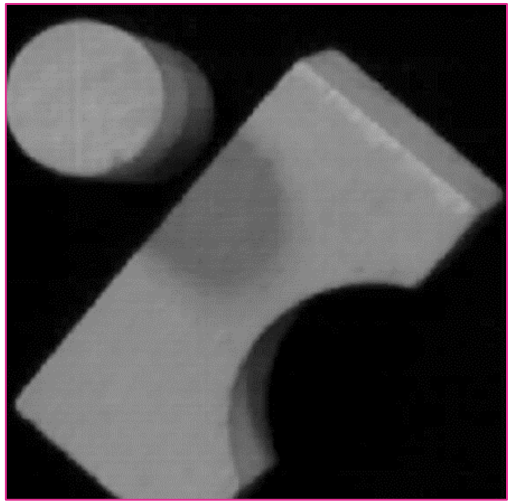
input

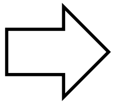




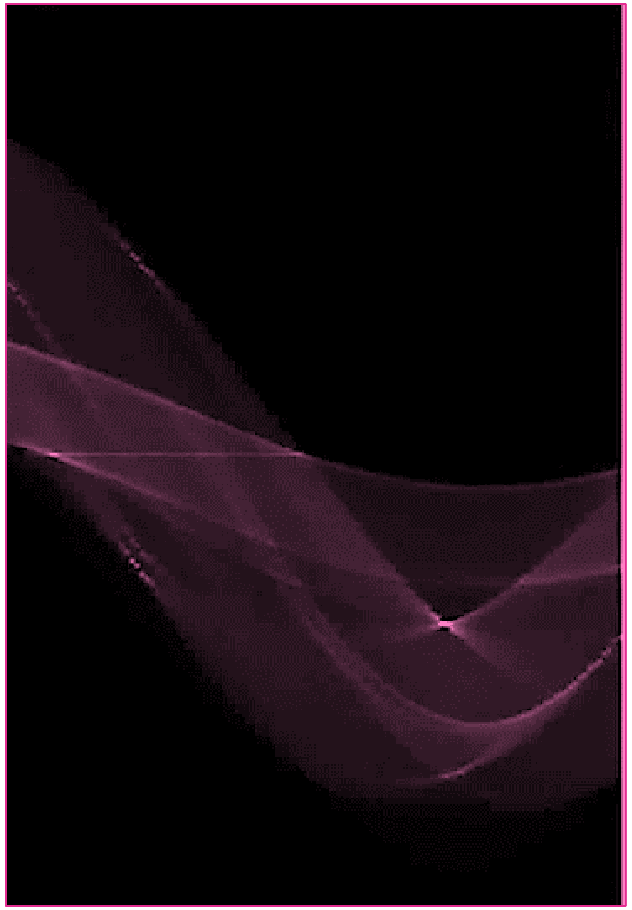
Hough transform

input





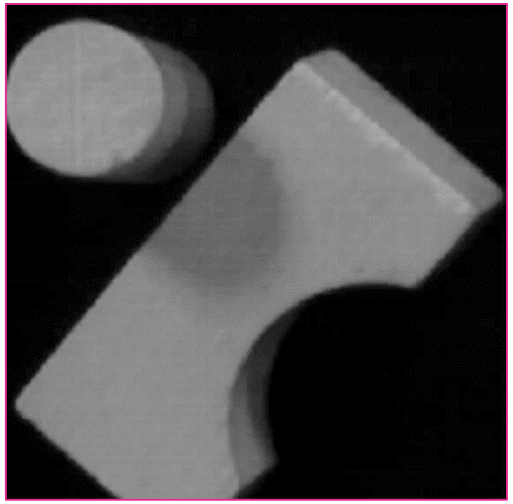
Hough transform

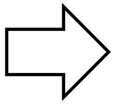


Hough transform

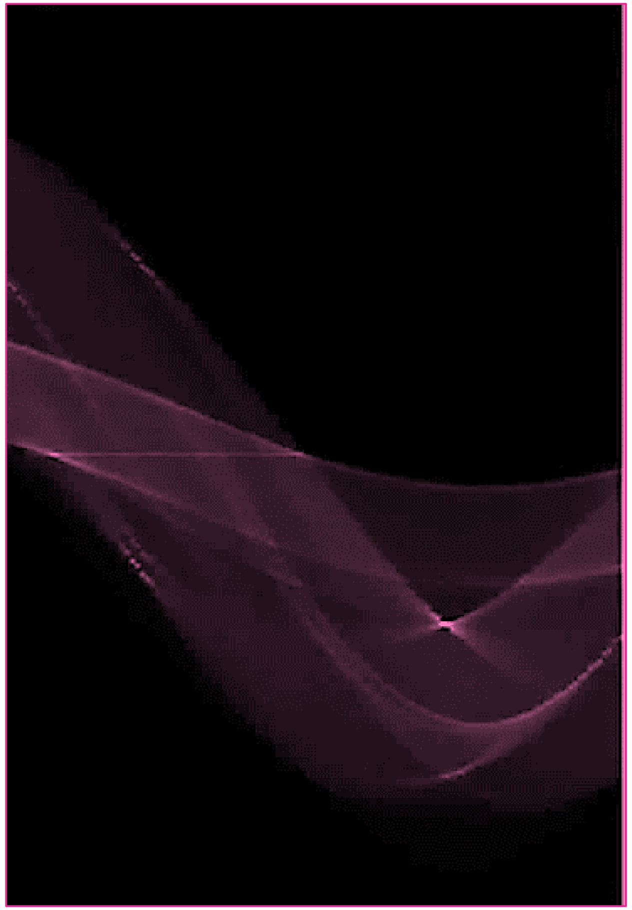
- bright spot in HT corresponds to?

input





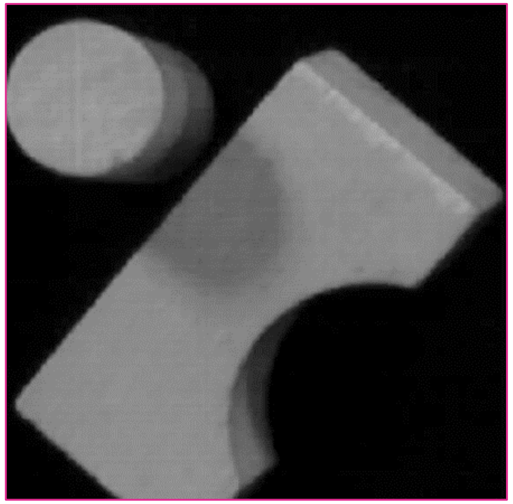
Hough transform

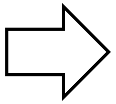


Hough transform

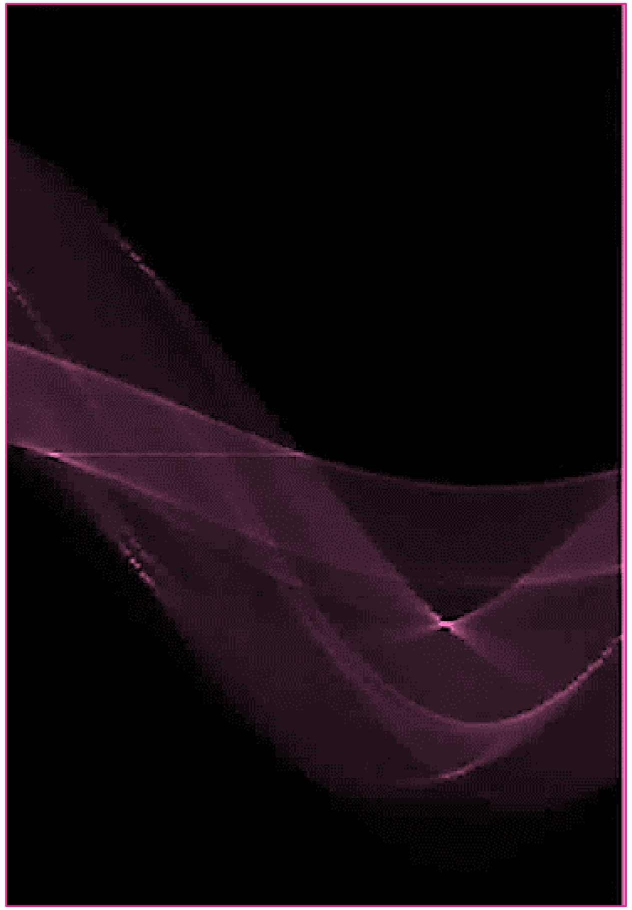
- bright spot in HT corresponds to?
- can circles be detected?

input





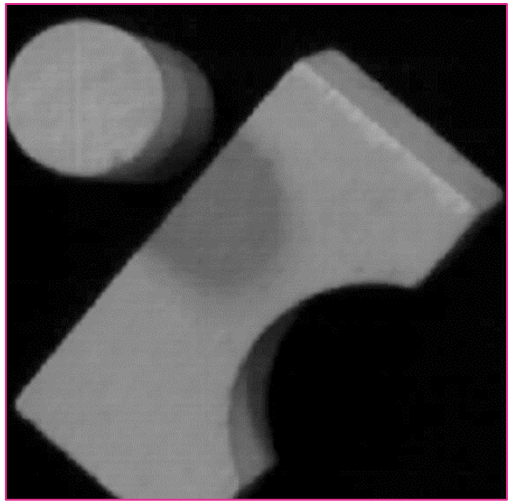
Hough transform

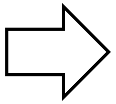


Hough transform

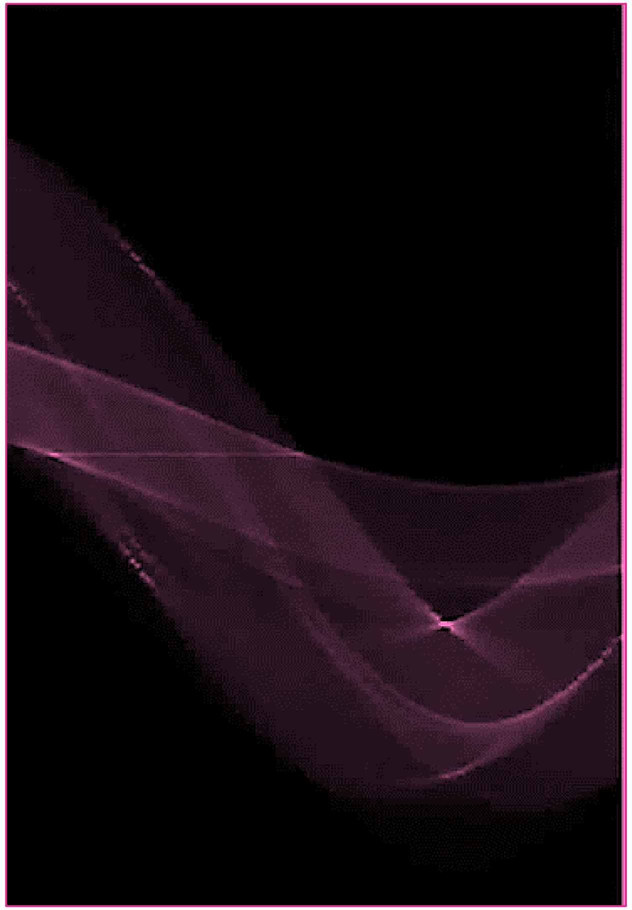
- bright spot in HT corresponds to?
- can circles be detected?
- why does Hough image is not the same size as input?

input





Hough transform



Hough transform

Input

Hough transform

Input


Hough transform

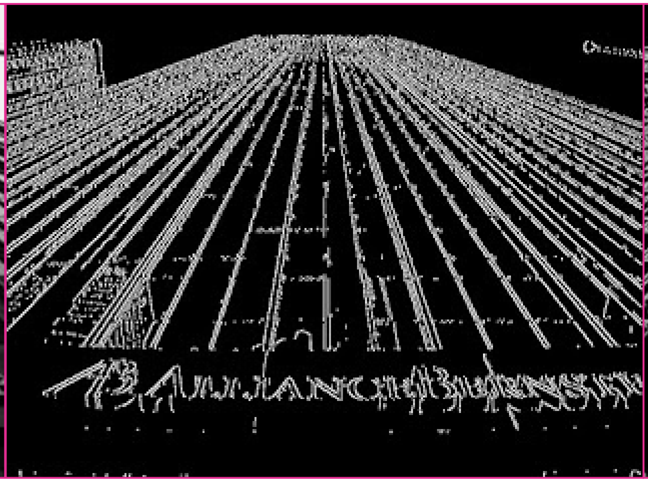
Input

Sobel

Hough transform

Input

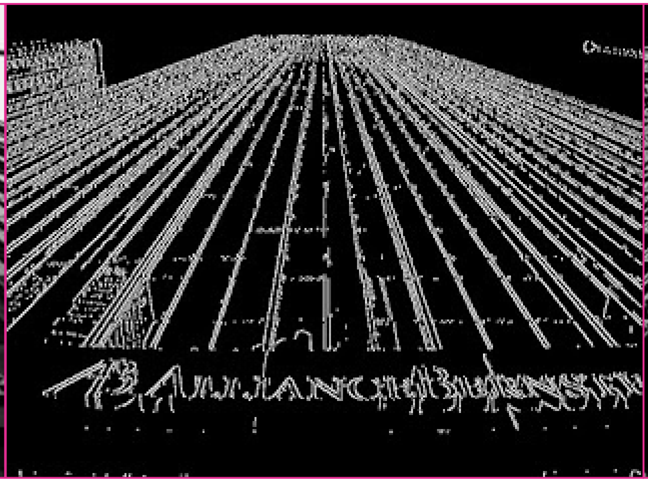
Sobel



Hough transform

Input

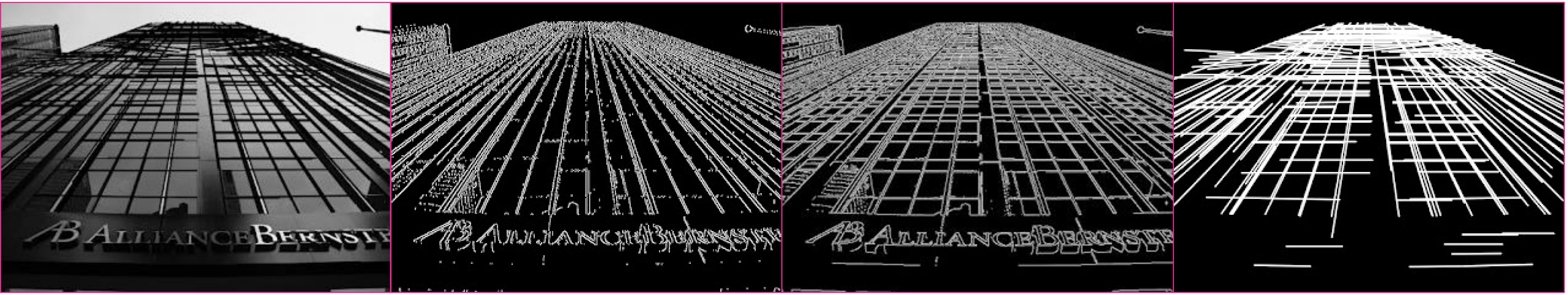
Sobel



Canny

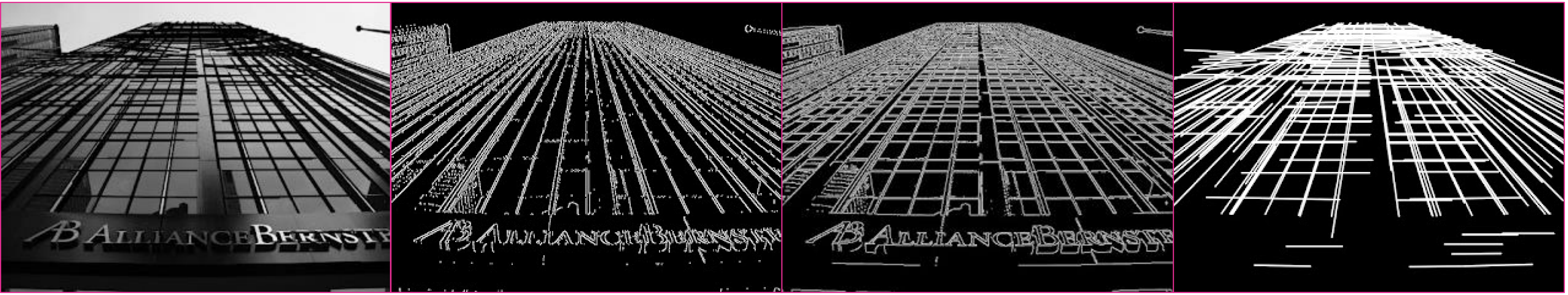
Hough transform

Input Sobel Canny

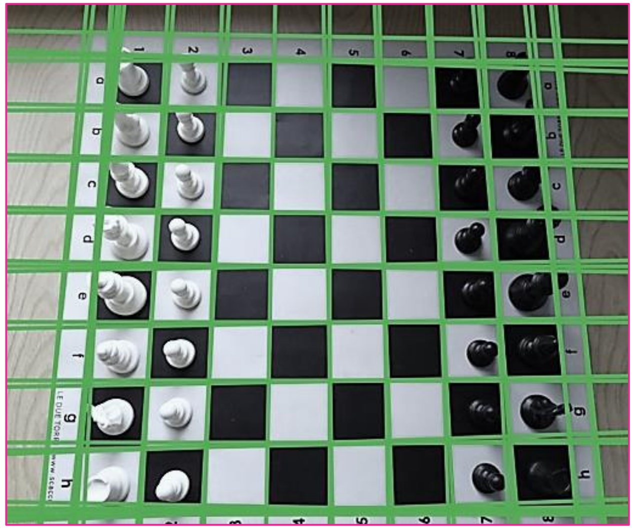


Hough transform

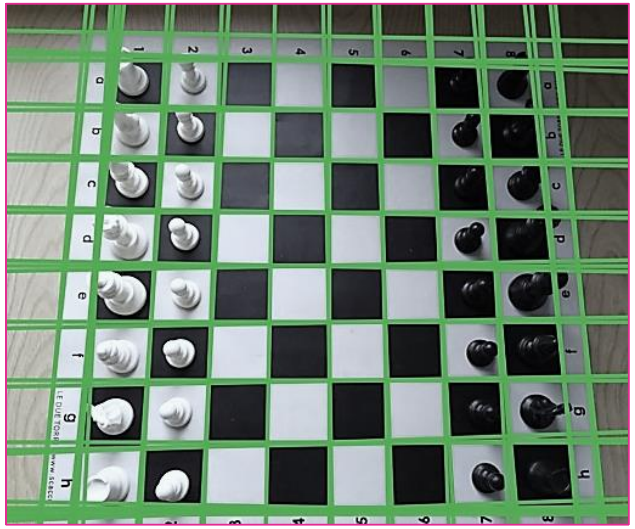
Input | Sobel | Canny | Hough



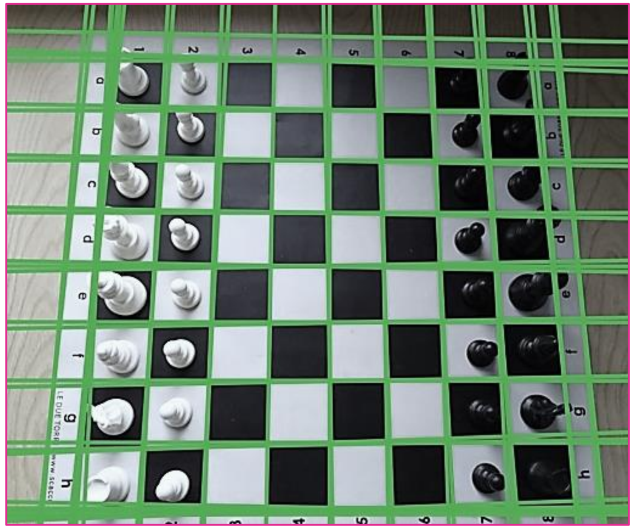
Hough transform



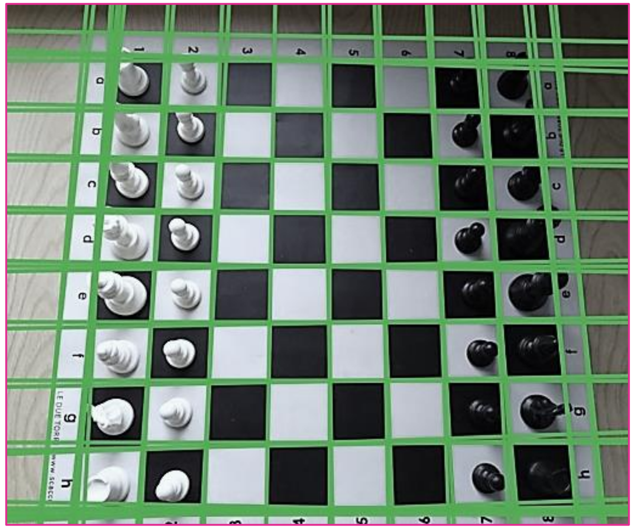
Hough transform



Hough transform

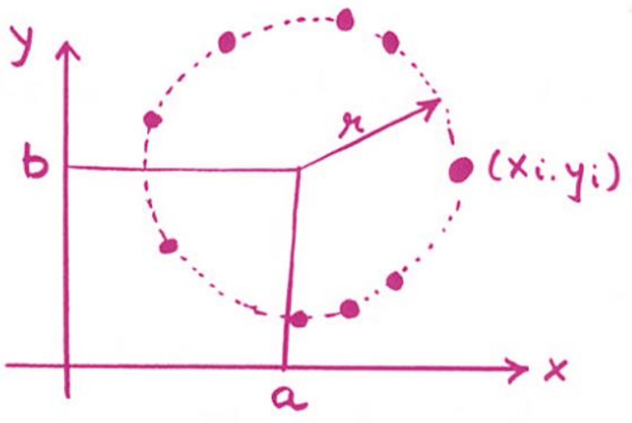


Hough transform



Hough transform

- Equation of circle



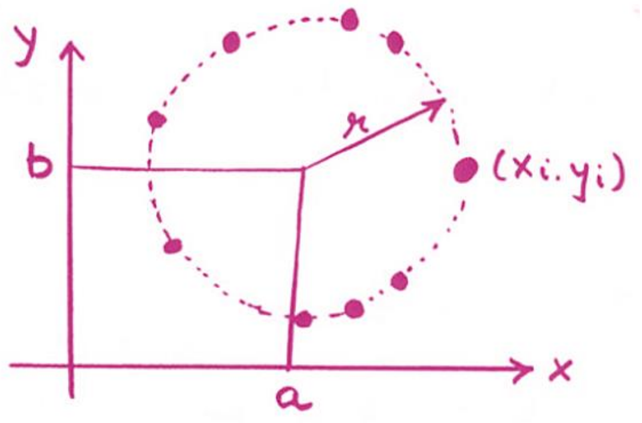
Hough transform

- Equation of circle

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

If radius is known: (2D Hough Space)

Accumulator Array $A(a,b)$



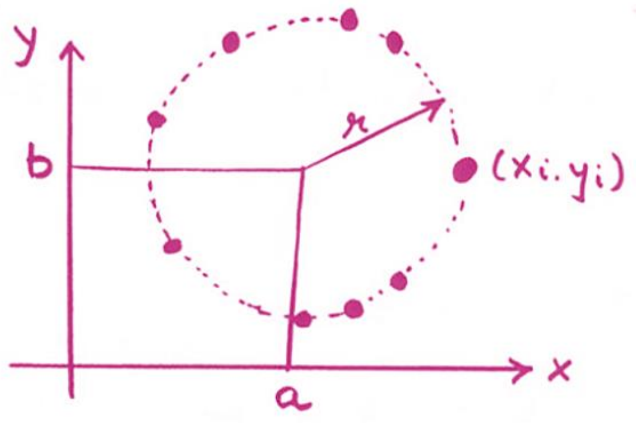
Hough transform

- Equation of circle

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

If radius is known: (2D Hough Space)

Accumulator Array $A(a,b)$



- Approx. object's size are known
    - radius is known

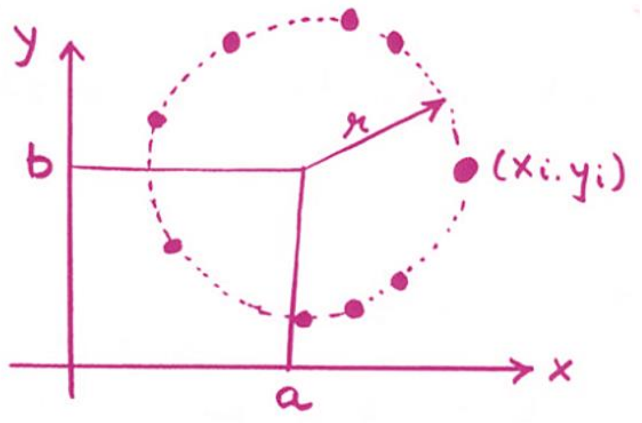
Hough transform

- Equation of circle

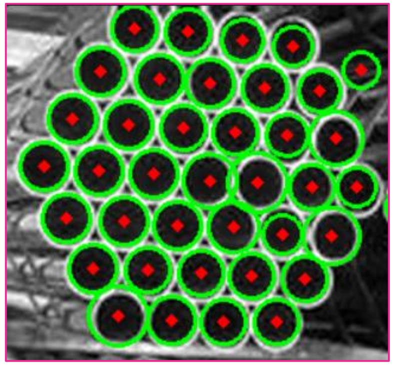
$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

If radius is known: (2D Hough Space)

Accumulator Array $A(a,b)$



- Approx. object's size are known
  - radius is known



Hough transform

- If radii are not known, exhaustively search all possibilities
- Pseudocode:

$: A(\quad) = 0$

$: \forall(x, y)$

$: if\ M_T(x, y)$

$: \forall(a, b)$

$: r = sqrt\{\ (x-a)^2 + (y-b)^2\ \}$

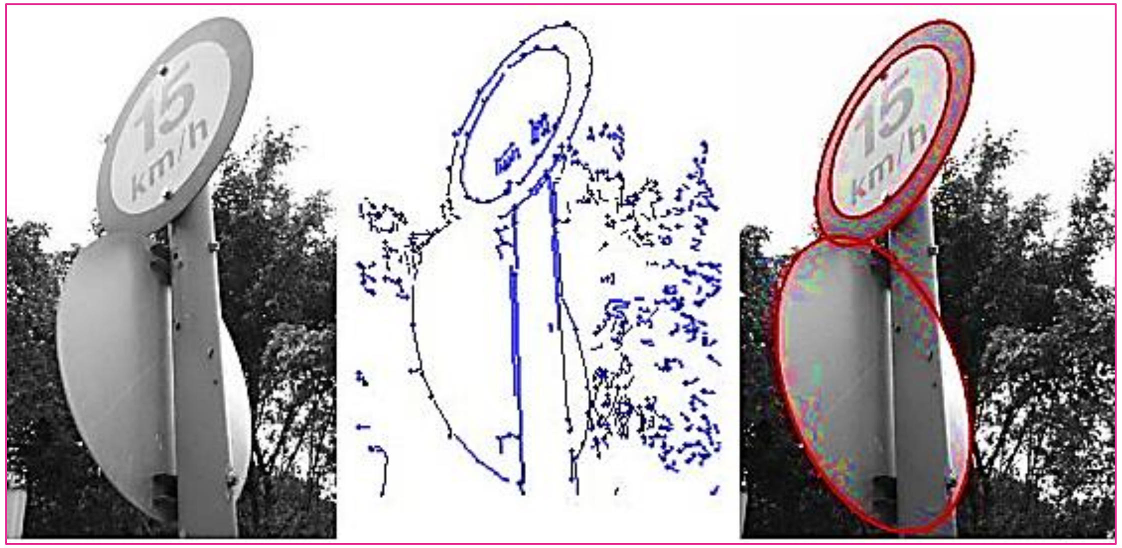
$: A(a, b, r) + +$

$: find\ maximas\ in\ A(\quad)$

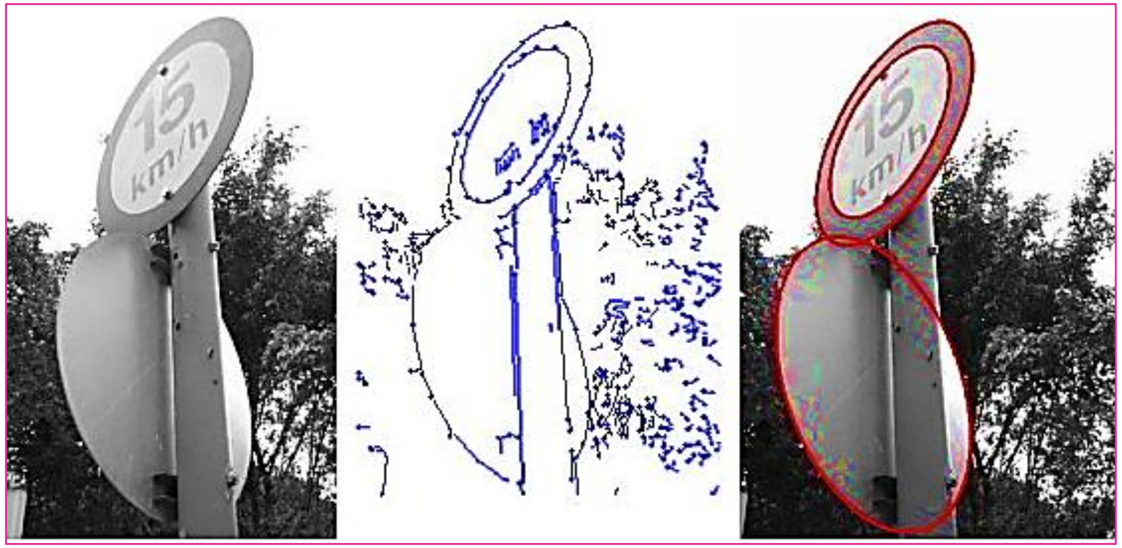
Hough transform

Hough transform

Hough transform



Hough transform



Hough Transform

- Space complexity

Hough Transform

- Space complexity

  - With k quantized bins, it exponentially increases with number of parameters $n \rightarrow k^n$

Hough Transform

- Space complexity

  - With k quantized bins, it exponentially increases with number of parameters $n \rightarrow k^n$

  - Separate quantization for each dimension can be performed

Hough Transform

- Space complexity

  - With k quantized bins, it exponentially increases with number of parameters $n \quad \rightarrow \quad k^n$

  - Separate quantization for each dimension can be performed

- Time complexity

Hough Transform

- Space complexity

  - With k quantized bins, it exponentially increases with number of parameters $n \rightarrow k^n$

  - Separate quantization for each dimension can be performed

- Time complexity

  - Voting is linearly proportional to # of edge points
  - Time complexity is constant in # of edge points detected

Hough transform: other shapes

	parameters	
Line	ρ, θ	$x\cos\theta + y\sin\theta = \rho$
Circle	x_0, y_0, ρ	$(x-x_o)^2+(y-y_0)^2=r^2$
Parabola	x_0, y_0, ρ, θ	$(y-y_0)^2=4\rho(x-x_o)$
Ellipse	x_0, y_0, a, b, θ	$(x-x_o)^2/a^2+(y-y_0)^2/b^2=1$

Conclusion

- Hough Transform

Conclusion

- Hough Transform

- It's tolerant to noise in some extent and highly tolerant to the missing points in an edge

- Space complexity increases with the number of parameters

- Hough space is quantized
  - finer the quantization,
  - more accurate the edge det. will be,
  - but slower will be the procc. speed

Conclusion

- Hough Transform

- It's tolerant to noise in some extent and highly tolerant to the missing points in an edge

- Space complexity increases with the number of parameters

- Hough space is quantized
  - finer the quantization,
  - more accurate the edge det. will be,
  - but slower will be the procc. speed

Hough? →