04.10.2019

Digital Image Processing (CSE/ECE 478)

Lecture-15: Image Segmentation

Ravi Kiran

Center for Visual Information Technology (CVIT), IIIT Hyderabad

Image Segmentation

Partitioning an image into a collection of connected sets of pixels.

1. into regions, which usually cover the image

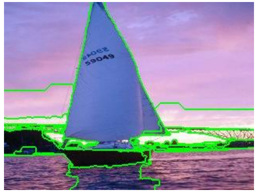
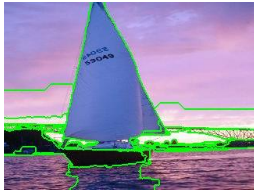


Image Segmentation

Partitioning an image into a collection of connected sets of pixels.

1. into regions, which usually cover the image



2. into linear structures, such as
   - line segments
   - curve segments

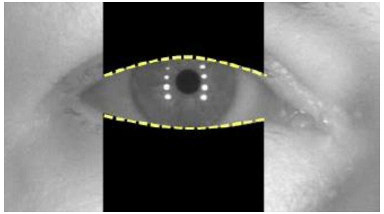
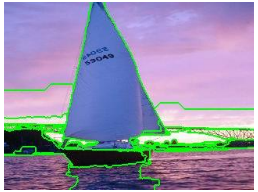


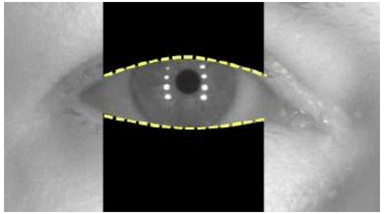
Image Segmentation

Partitioning an image into a collection of connected sets of pixels.

1. into regions, which usually cover the image



2. into linear structures, such as
   - line segments
   - curve segments



3. into 2D shapes, such as
   - circles
   - ellipses
   - ribbons (long, symmetric regions)

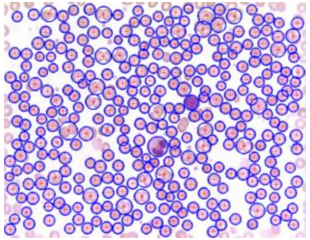


Image Segmentation - Approaches

- Edge-based
- Thresholding
- Region-growing
- Morphological Watersheds
- Motion

Edge-based segmentation → Detection of Discontinuities

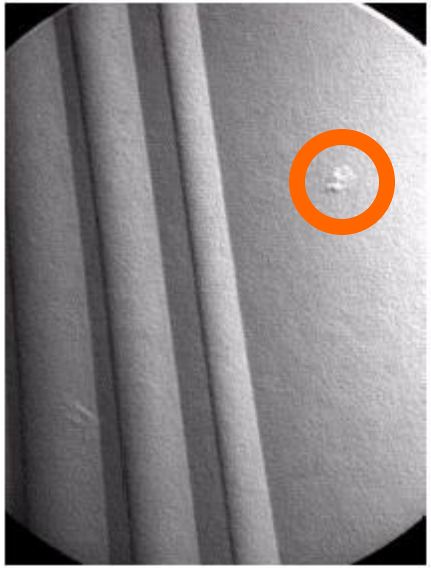
- Three basic types of grey level discontinuities
  - Points / Corners
  - Edges
  - Lines
- Typically find discontinuities using masks and correlation

Point Detection

-1	-1	-1
-1	8	-1
-1	-1	-1

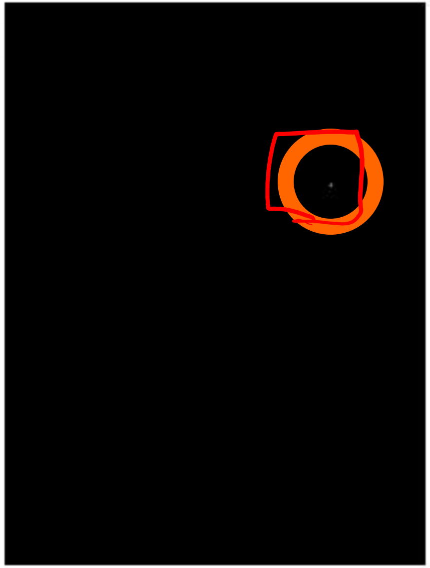
Point Detection (cont…)

-1	-1	-1
-1	8	-1
-1	-1	-1



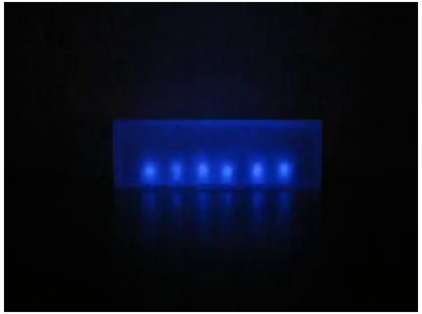
X-ray image of a turbine blade

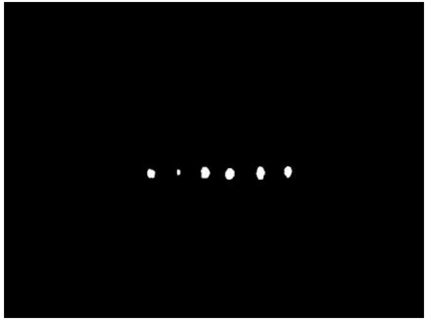
Result of point detection

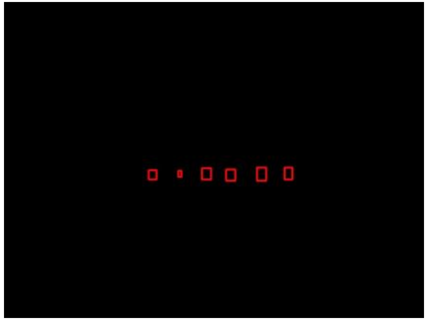


Result of thresholding

Example: Blinking/LED detector







https://stackoverflow.com/questions/10702105/detecting-led-object-status-from-image

Oriented Line Detection

−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

−1	−1	2
−1	2	−1
2	−1	−1

+45°

−1	2	−1
−1	2	−1
−1	2	−1

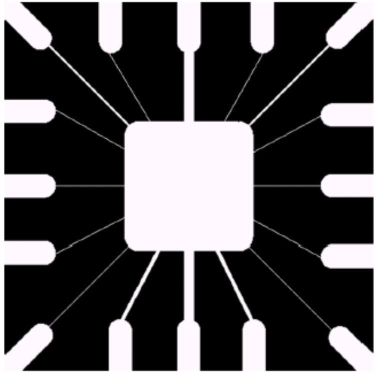
Vertical

2	−1	−1
−1	2	−1
−1	−1	2

−45°

Oriented Line Detection (cont…)

Binary image of a wire bond mask



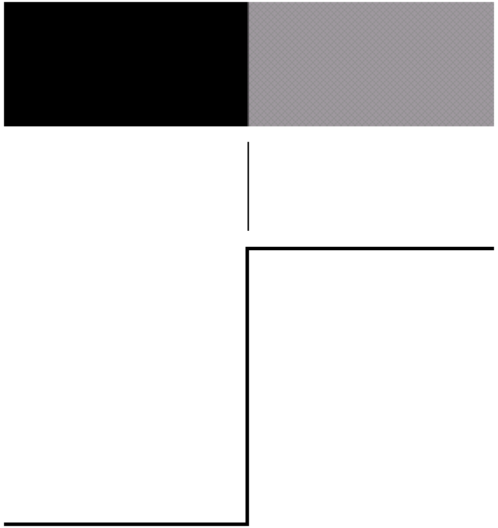
After processing with -45° line detector

Result of thresholding filtering result

Edge Detection

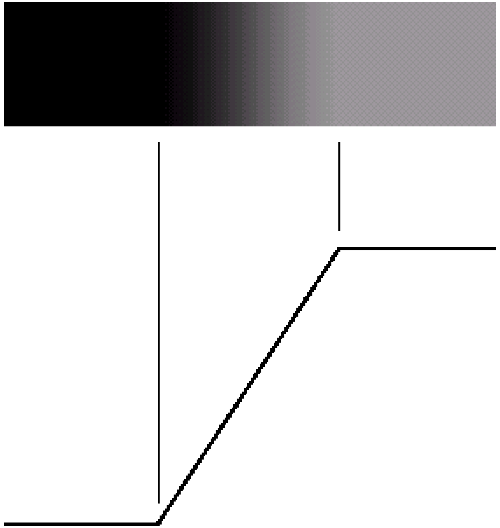
- Edge = <u>set</u> of <u>connected</u> pixels that lie on the boundary between two regions

Model of an ideal digital edge



Gray-level profile
of a horizontal line
through the image

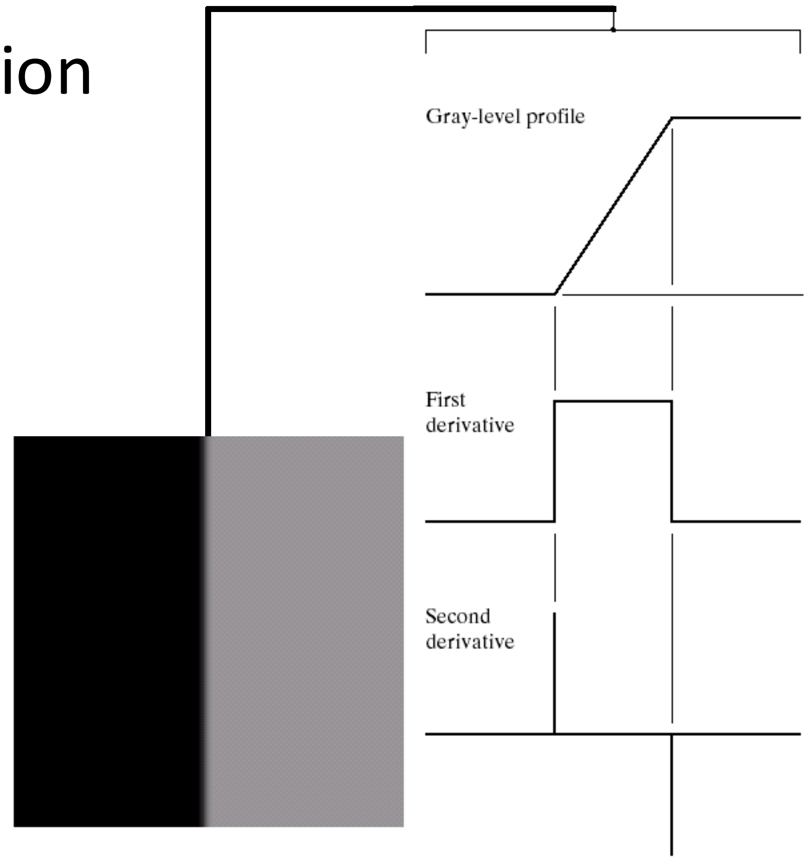
Model of a ramp digital edge



Gray-level profile
of a horizontal line
through the image

Edges & Derivatives

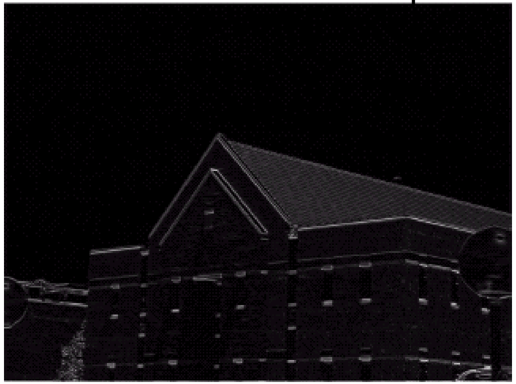
- 1^{st} derivative → edge location
- 2^{nd} derivative →
  - edge location
  - edge direction



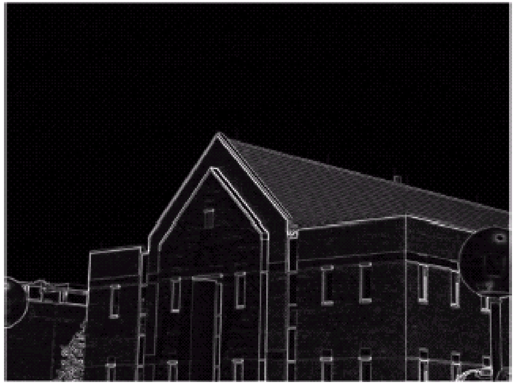
Edge Detection Example With Smoothing

Original Image

Horizontal Gradient Component



Vertical Gradient Component



Combined Edge Image

Laplacian Edge Detection

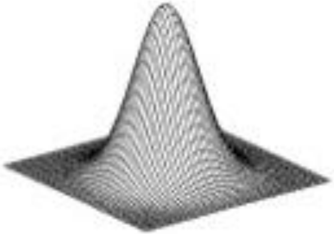
- 2^{nd}-order derivative based Laplacian filter

0	−1	0
−1	4	−1
0	−1	0

−1	−1	−1
−1	8	−1
−1	−1	−1

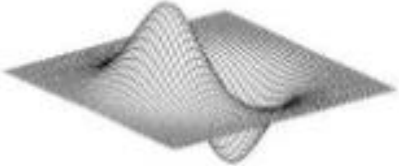
- Typically not used by itself → too sensitive to noise
- Combined with a smoothing Gaussian filter
  - Smooth the image with Gaussian, then apply Laplacian

Laplacian of Gaussian (LoG)



Gaussian

$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



derivative of Gaussian

$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

Laplacian of Gaussian

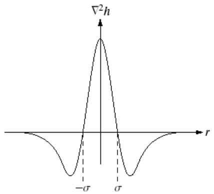
$$\nabla^2 h_\sigma(u,v)$$

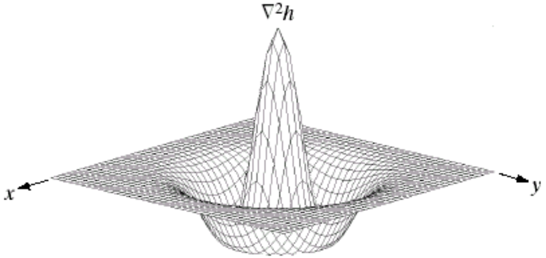
∇^2 is the **Laplacian** operator:

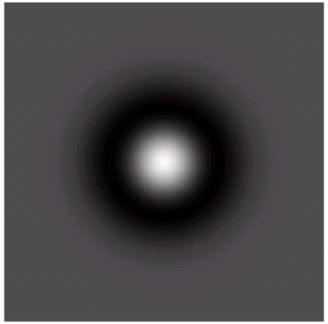
$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

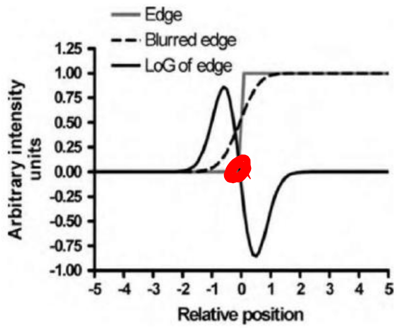
Laplacian Of Gaussian

- The Laplacian of Gaussian (or Mexican hat)
  - Gaussian for noise removal
  - Laplacian for edge detection





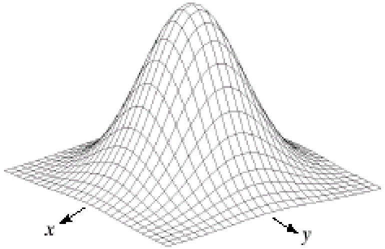




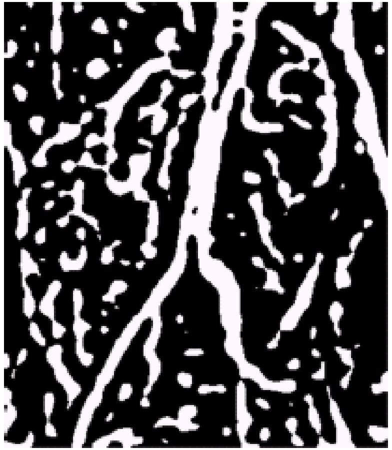
$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\sigma^2}\right]e^{-\frac{x^2+y^2}{2\sigma^2}}$$

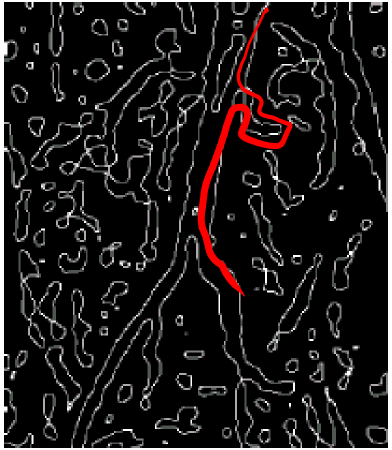
0	0	−1	0	0
0	−1	−2	−1	0
−1	−2	16	−2	−1
0	−1	−2	−1	0
0	0	−1	0	0

Laplacian Of Gaussian Example



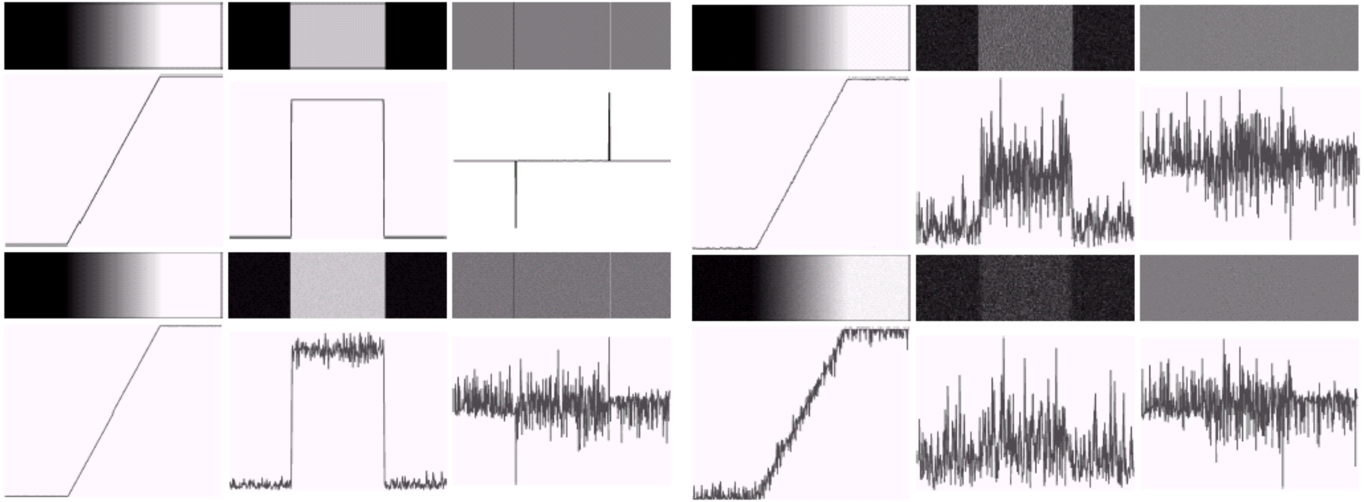
–1	–1	–1
–1	8	–1
–1	–1	–1



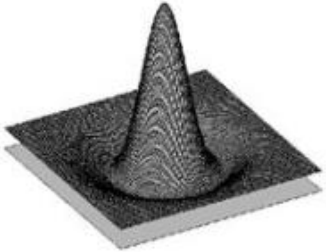


Derivatives & Noise

- Derivative-based edge detectors are extremely sensitive to noise
- We need to keep this in mind



Laplacian ~ Difference of Gaussian

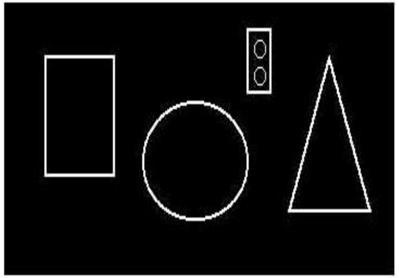


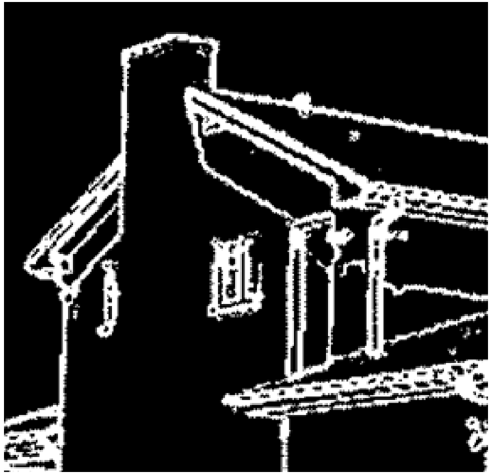
DoG = Difference of Gaussians

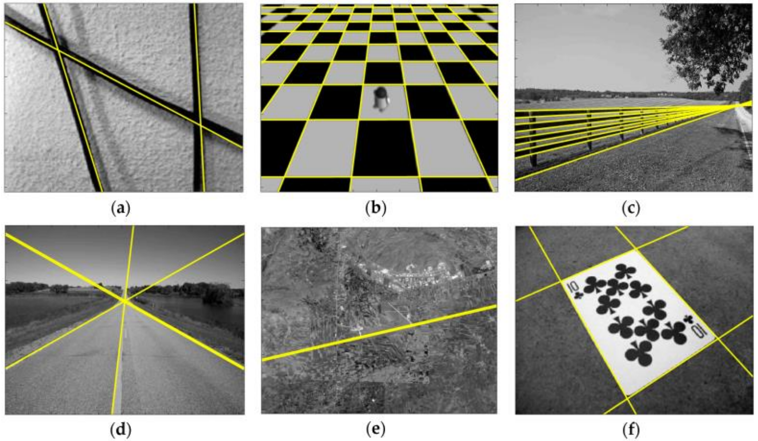
Cheap approximation - no derivatives needed.

Hough Transform

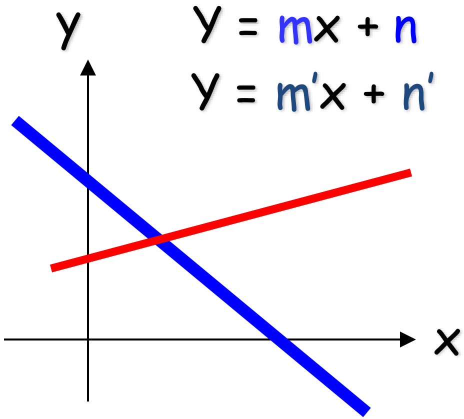
- Straight lines
- Circles
- Algebraic curves
- Arbitrary specific shapes in an image







Hough Transform for Lines: Image and Parameter Spaces



$y_1 = m\, x_1 + n$

$y_2 = m\, x_2 + n$

$y_N = m\, x_N + n$

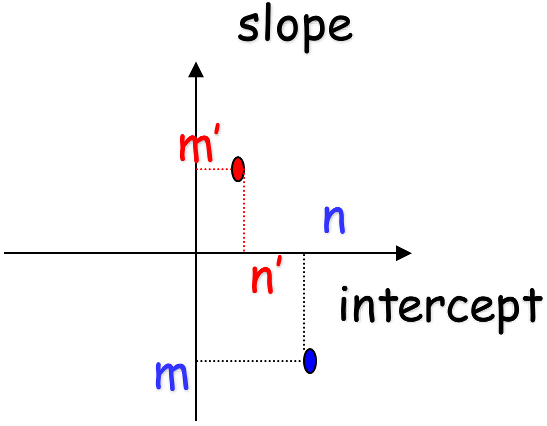


Image Space

Parameter Space

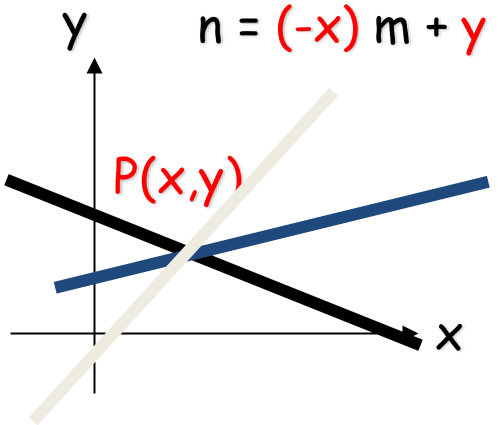
Line in Img. Space ~ Point in Param. Space

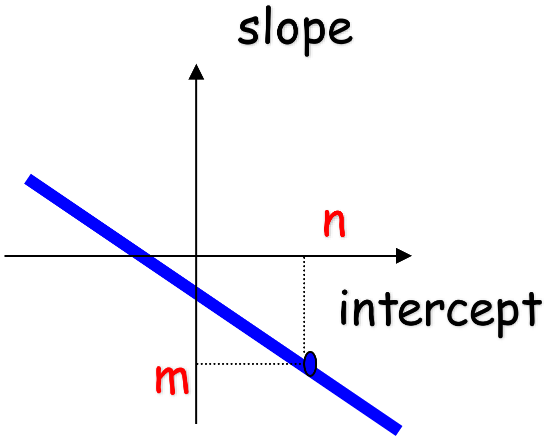
Image Parameter Spaces

- Image Space
  - Lines
  - Points
  - Collinear points
- Parameter Space
  - Points
  - Lines
  - Intersecting lines

Hough Transform Technique

- Given an edge point, there is an infinite number of lines passing through it (Vary m and n).
  - These lines can be represented as a line in parameter space.

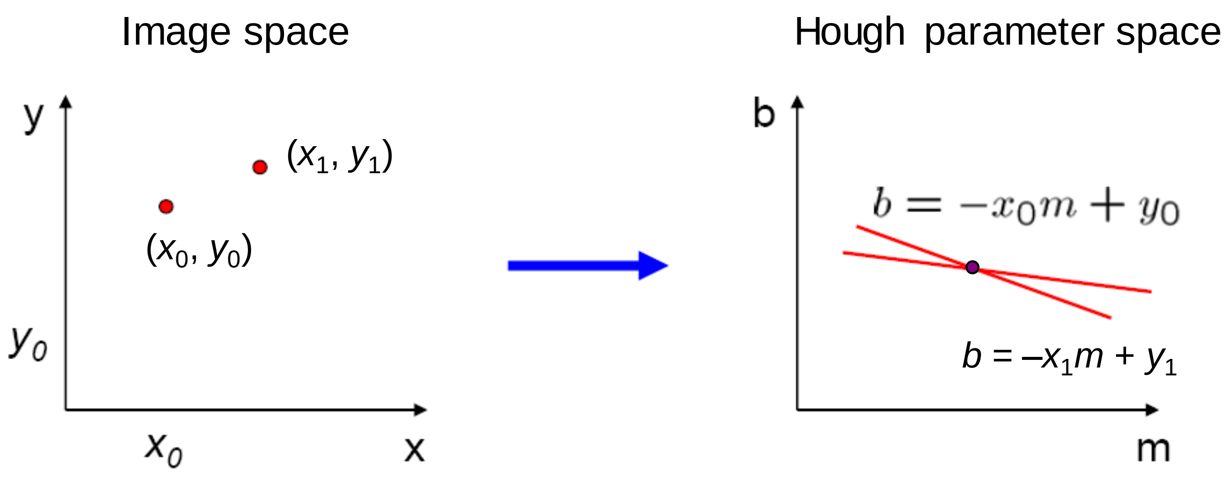


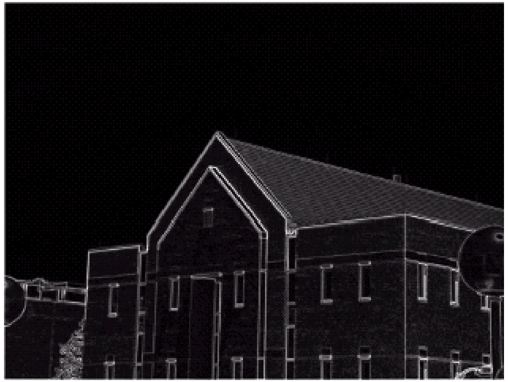


Parameter Space

Parameter space representation

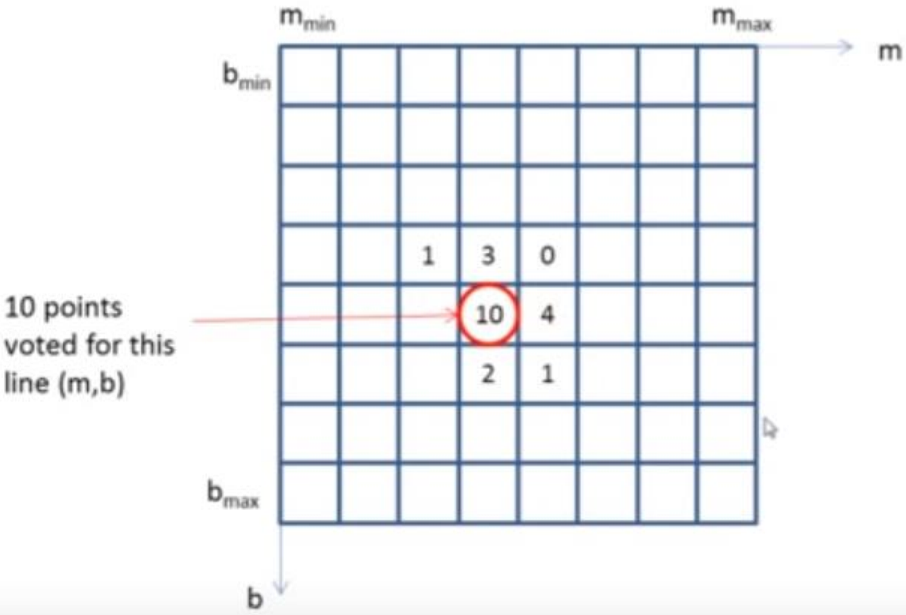
- Where is the line that contains both (x_0, y_0) and (x_1, y_1)?
  - It is the intersection of the lines $b = -x_0 m + y_0$ and $b = -x_1 m + y_1$





Hough Transform Algorithm

- Initialize an accumulator array A(m,b) to zero
- For each edge element (x,y), increment all cells that satisfy b = -x m + y
- Local maxima in A(m,b) correspond to lines

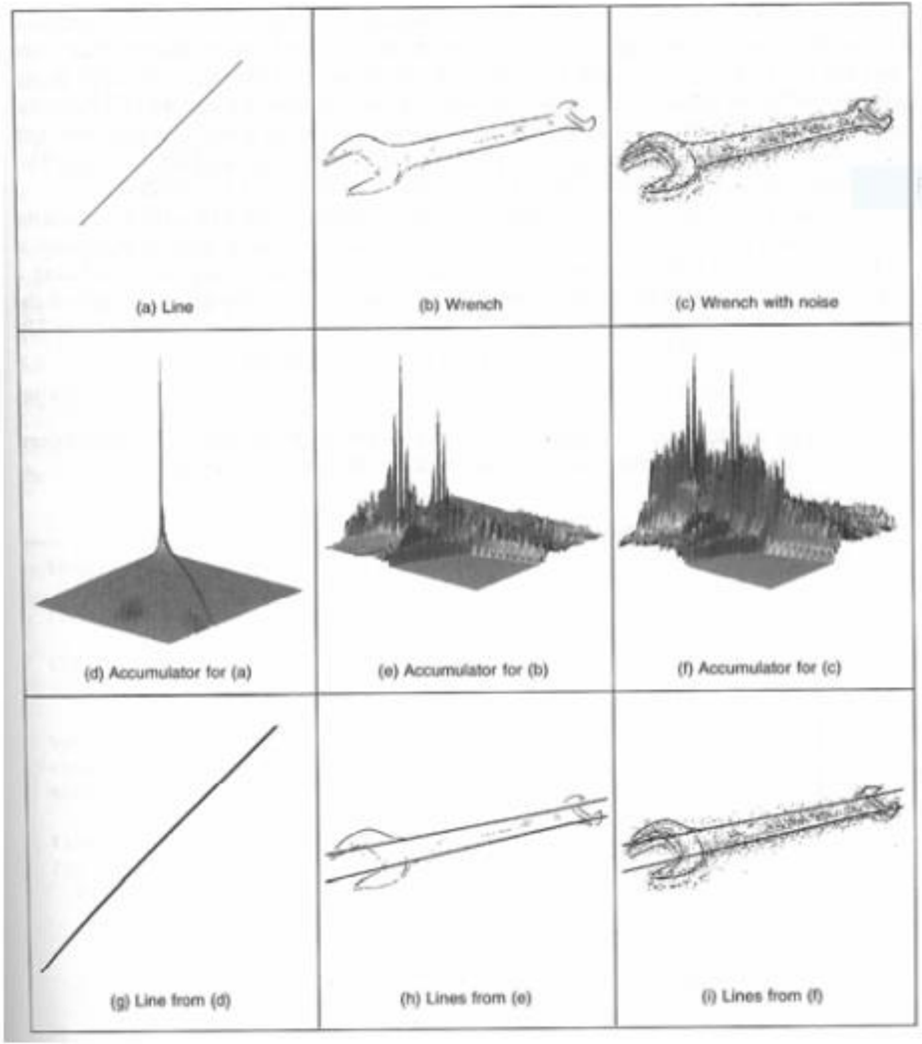


https://www.youtube.com/watch?v=uDB2qGqnQ1g

Thresholded edge images

Visualizing the accumulator space
The height of the peak will be defined by the number of pixels in the line.

Thresholding the accumulator space and superimposing this onto the edge image

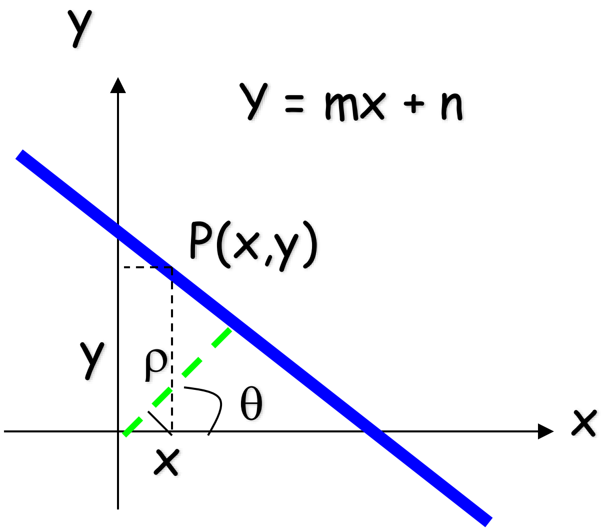


Practical Issues with This Hough Parameterization

- The slope of the line is $-\infty < m < \infty$
  - The parameter space is INFINITE
- The representation $y = mx + n$ does not express lines of the form $x = k$

Solution:

- Use the "Normal" equation of a line:



$\rho = x\cos\theta + y\sin\theta$

θ Is the line orientation

ρ Is the distance between the origin and the line

Consequence:

- A Point in Image Space is now represented as a SINUSOID
  - $\rho = x \cos\theta + y \sin\theta$

New Parameter Space for Hough based on trigonometric functions

- Use the parameter space (ρ, θ)
- The new space is FINITE
  - $0 < \rho < D$, where D is the image diagonal.
  - $-\pi < \theta < \pi$
- The new space can represent all lines
  - $Y = k$ is represented with $\rho = k, \theta=90$
  - $X = k$ is represented with $\rho = k, \theta=0$

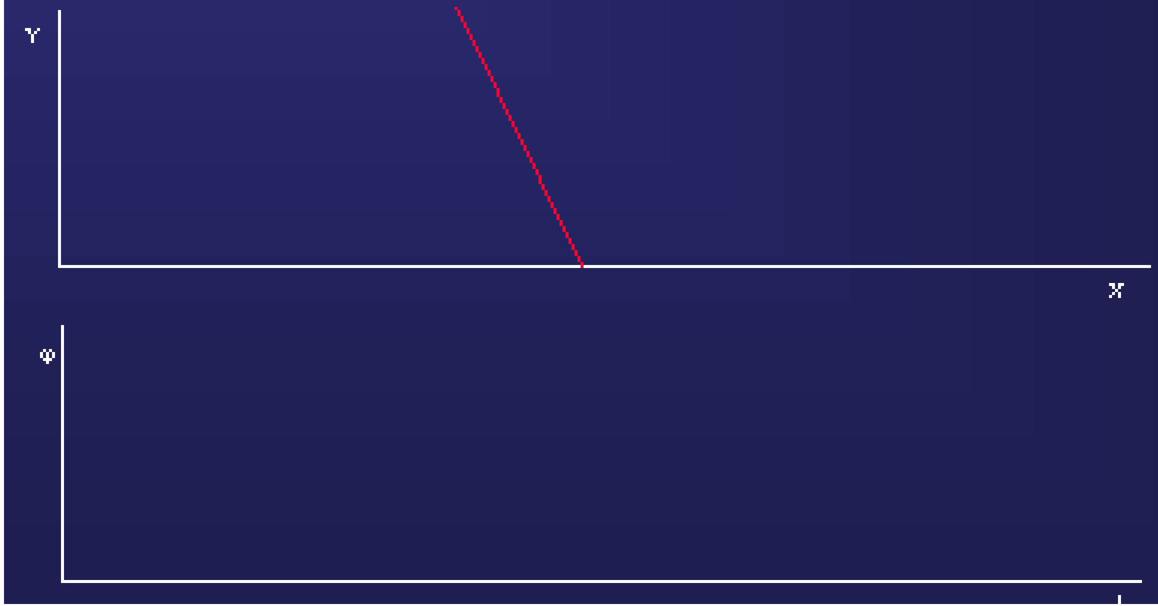
Hough Transform Algorithm

Input is an edge image ($E(i,j)=1$ for edgels)

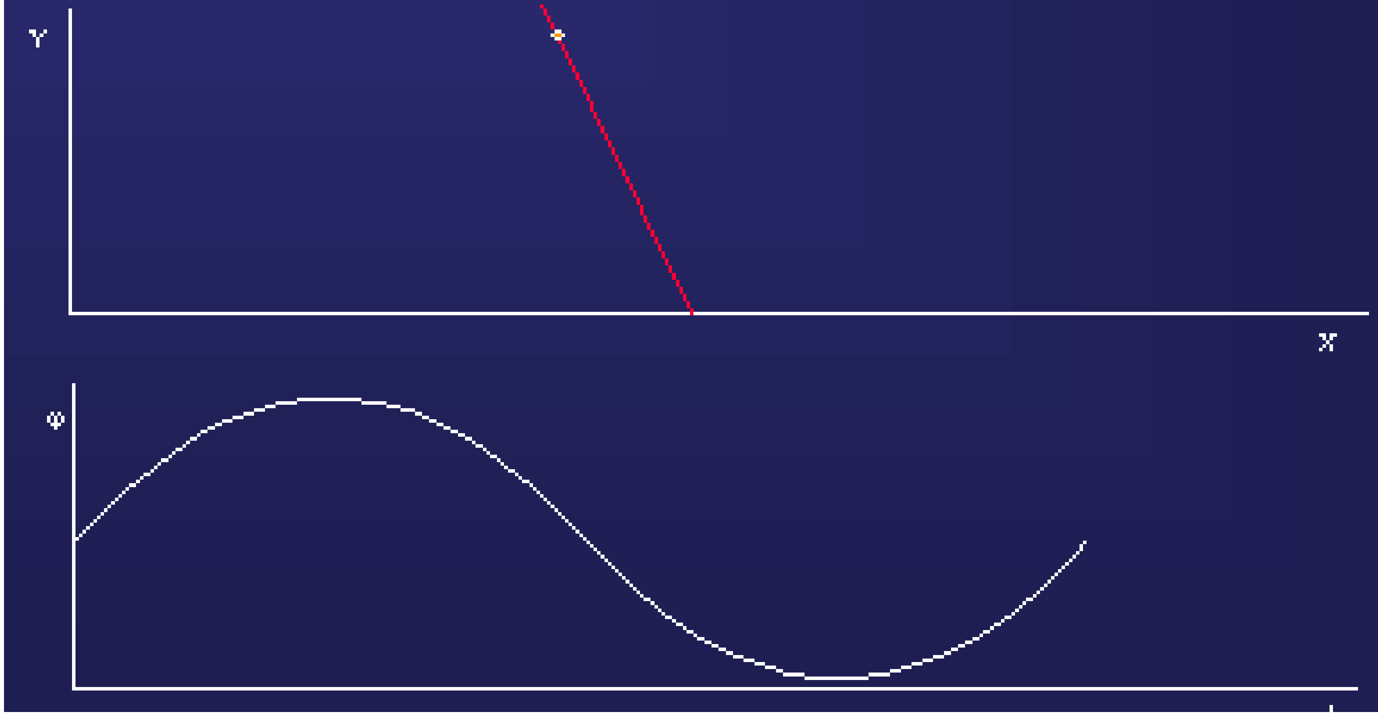
1. Discretize θ and ρ in increments of $d\theta$ and $d\rho$. Let $A(R,T)$ be an array of integer accumulators, initialized to 0.
2. For each pixel $E(i,j)=1$ and $h=1,2,\ldots T$ do
    1. $\rho = j \cos(h * d\theta) + i \sin(h * d\theta)$
    2. Find closest integer k corresponding to ρ
    3. Increment counter $A(h,k)$ by one
3. Find local maxima in $A(R,T)$

SHT: Another Viewpoint

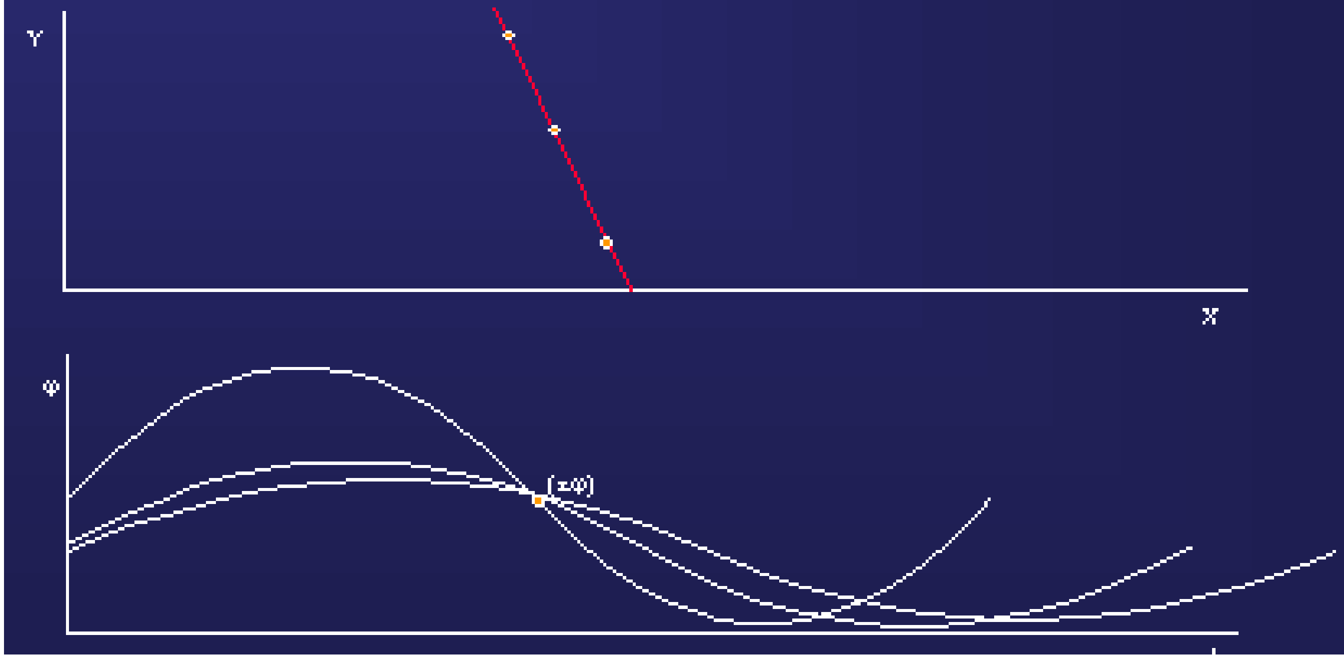
- Since any (r,φ) represents a point in parameter space we may plot all possible (r,φ) points for each (x,y) poi



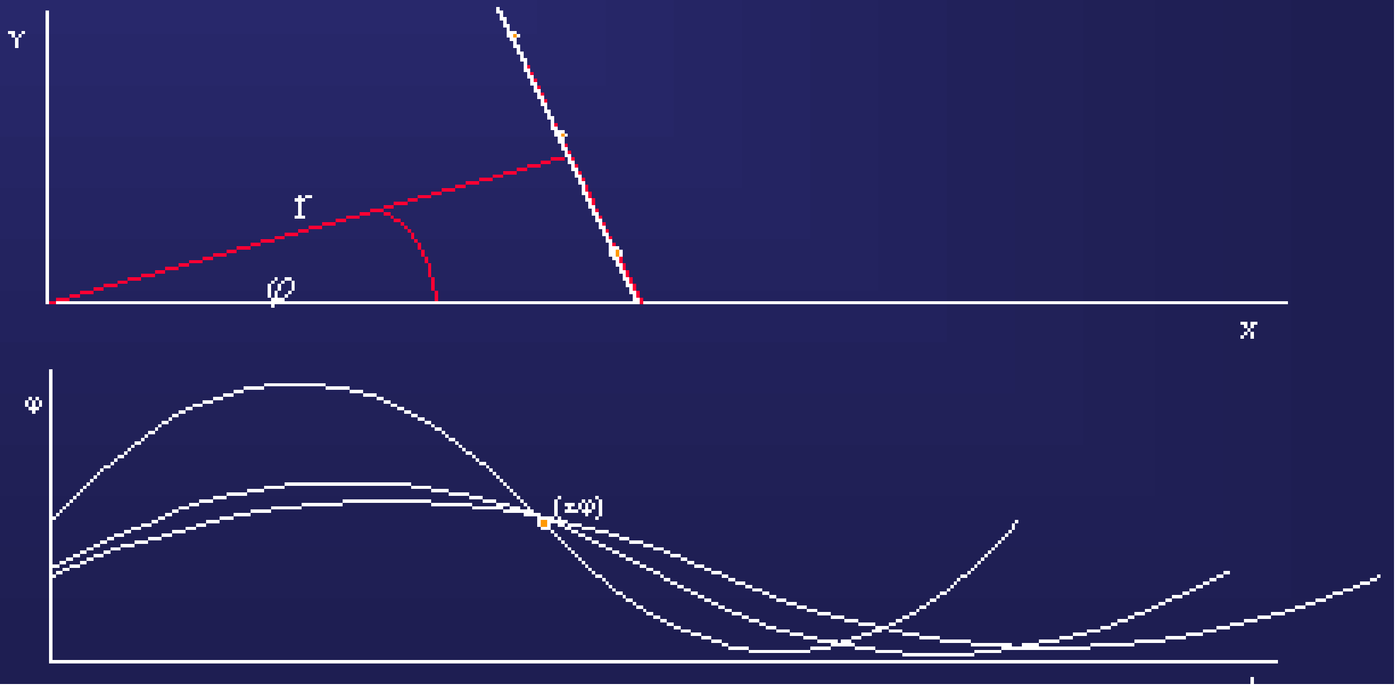
- Taking any particular (x,y) point its set of (r,φ) points is a sinusoid through parameter space



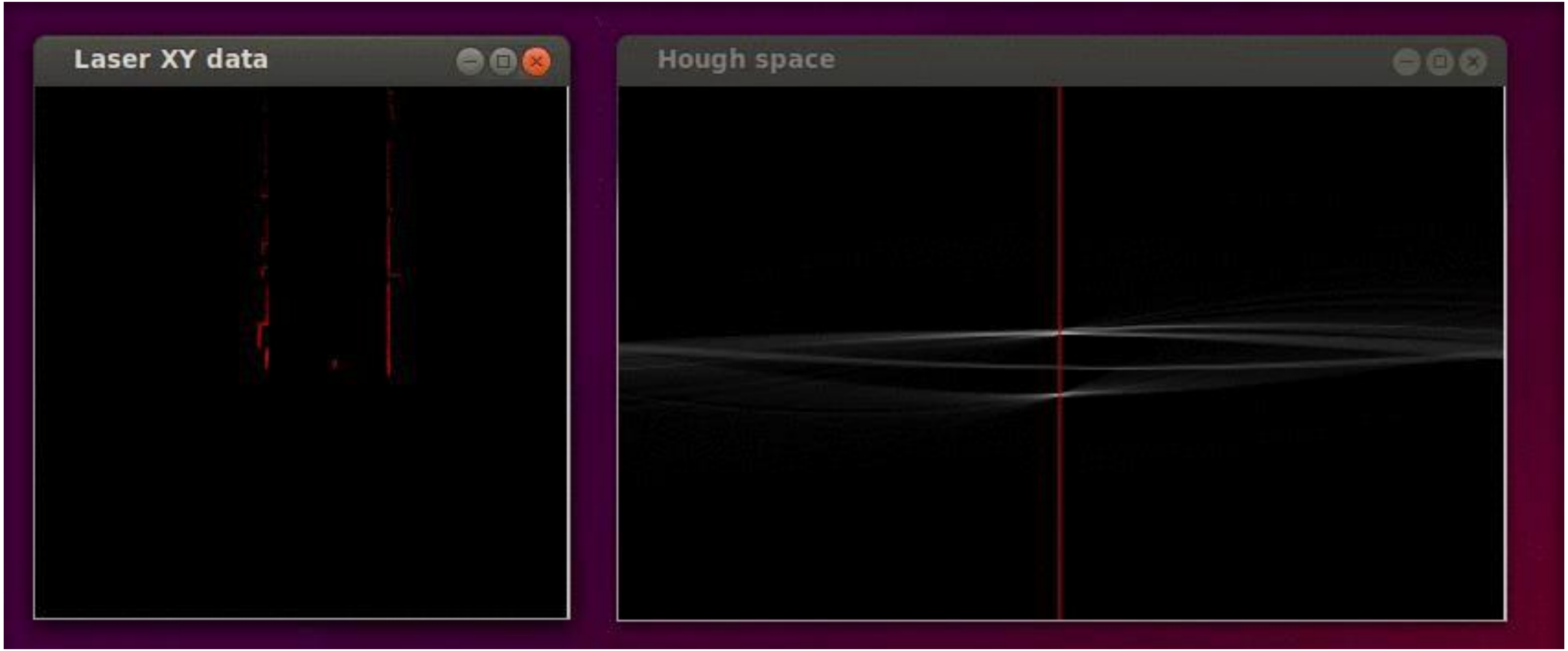
- The line upon which the (x,y) points lie is then given by the (r,φ) pair on which all the sinusoids *agree* (i.e. intersect)



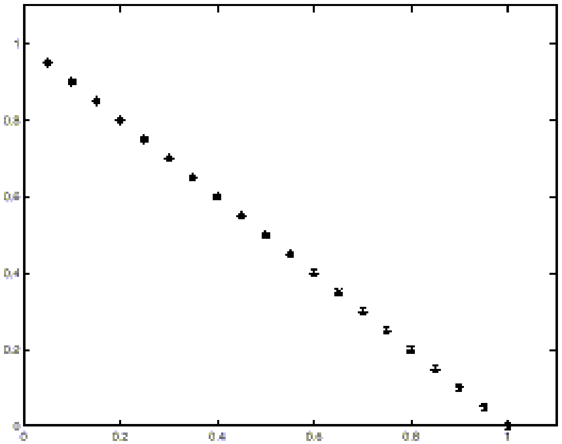
- Finding this intersection we then have the required (r,φ) parameters and therefore the line in the image

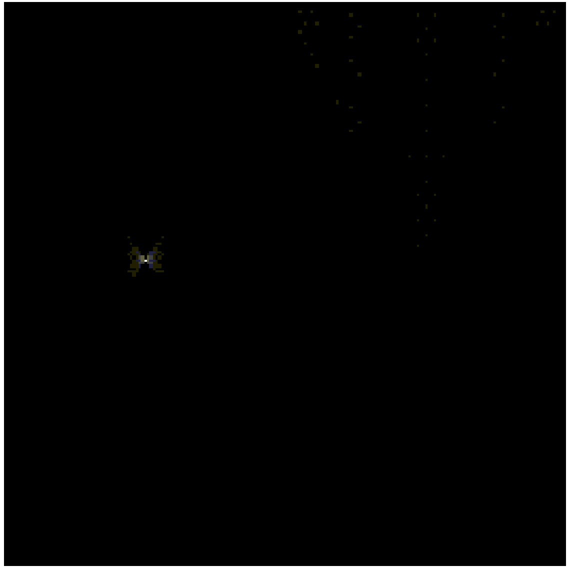


Animation

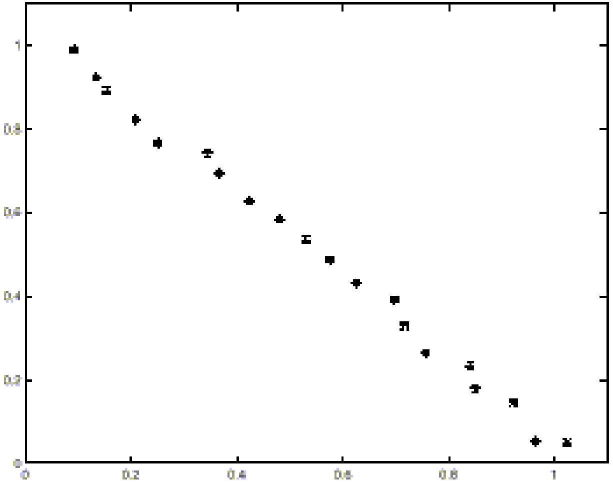


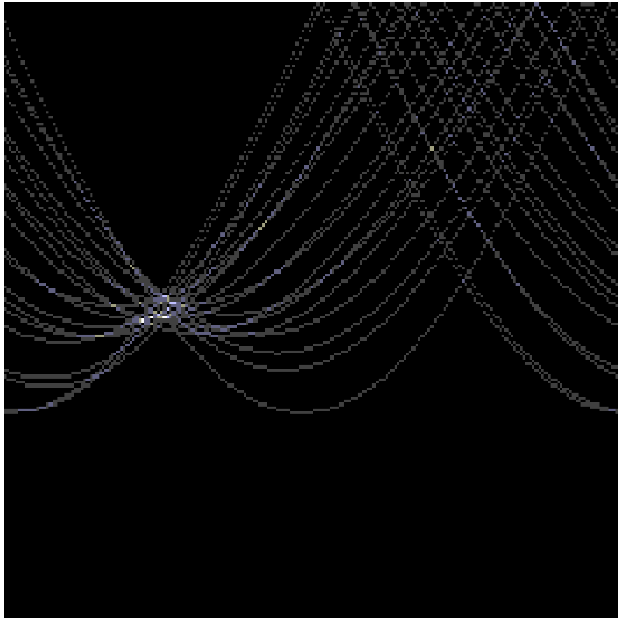
Hough Transform – cont.





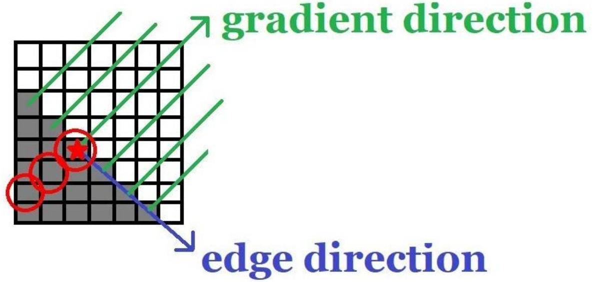
Hough Transform – cont.





Hough Transform Speed Up

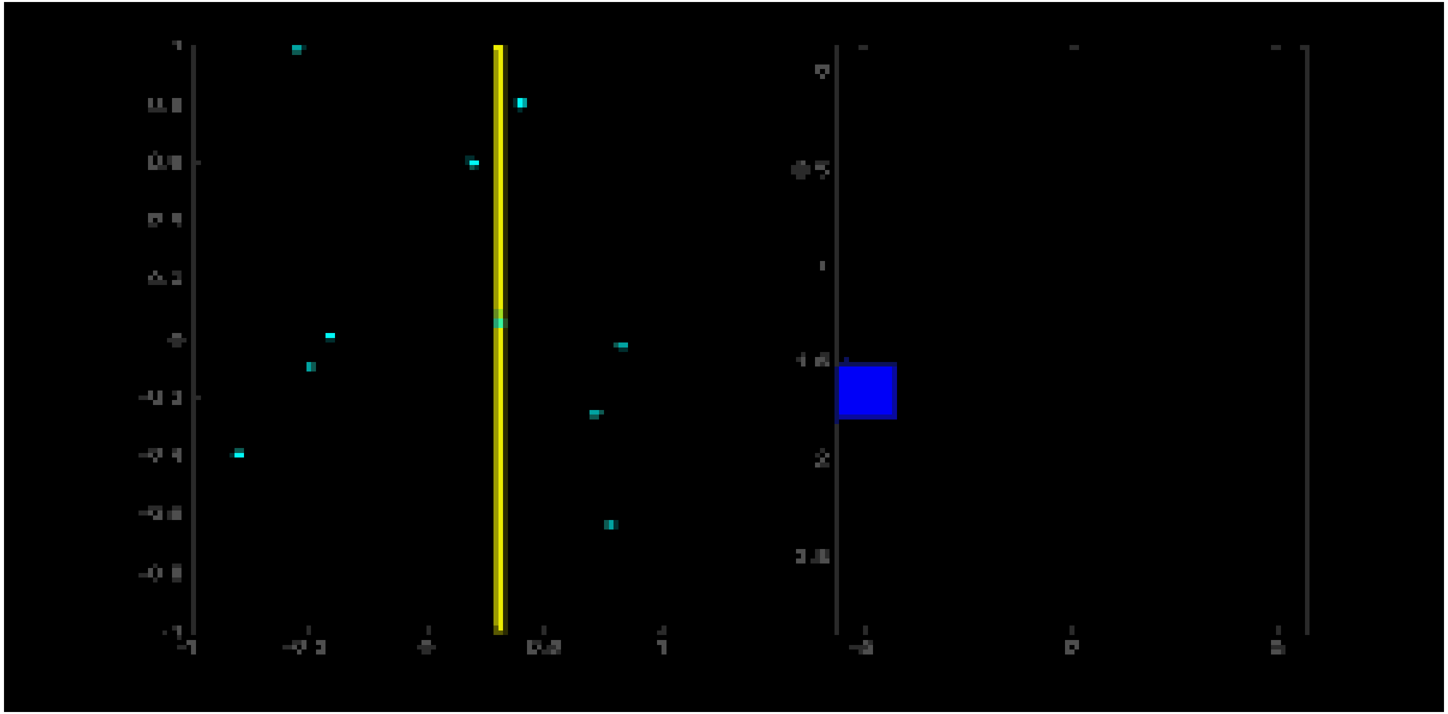
- The orientation of the edge: $\arctan(\frac{G_y}{G_x})$
- If we know the orientation of the edge – usually available from the edge detection step
  - We fix theta in the parameter space and increment **only one** counter!



Hough Transform Speed Up

- If we know the orientation of the edge – usually available from the edge detection step
  - We fix theta in the parameter space and <u>increment **only one** counter</u>!
  - We can allow for orientation uncertainty by incrementing a *few* counters around the "nominal" counter.

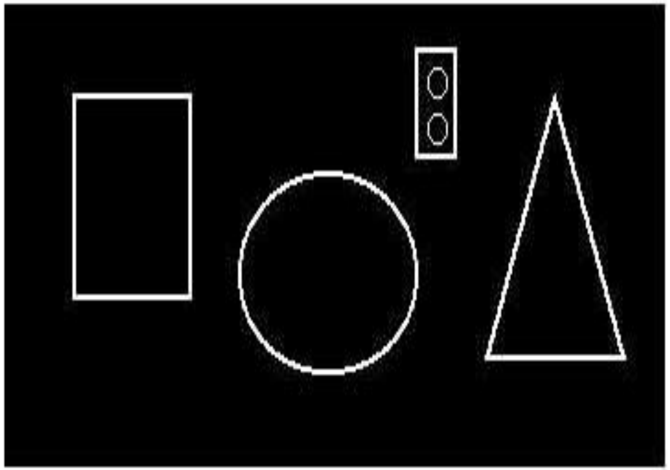
Animation



http://homepages.inf.ed.ac.uk/amos/hough.html

Hough Transform Generalizations

- It locates straight lines (SHT) - standard, simple HT
- It locates straight line intervals
- It locates circles
- It locates algebraic curves
- It locates arbitrary specific shapes in an image
  - **But you pay progressively for complexity of shapes by time and memory usage**



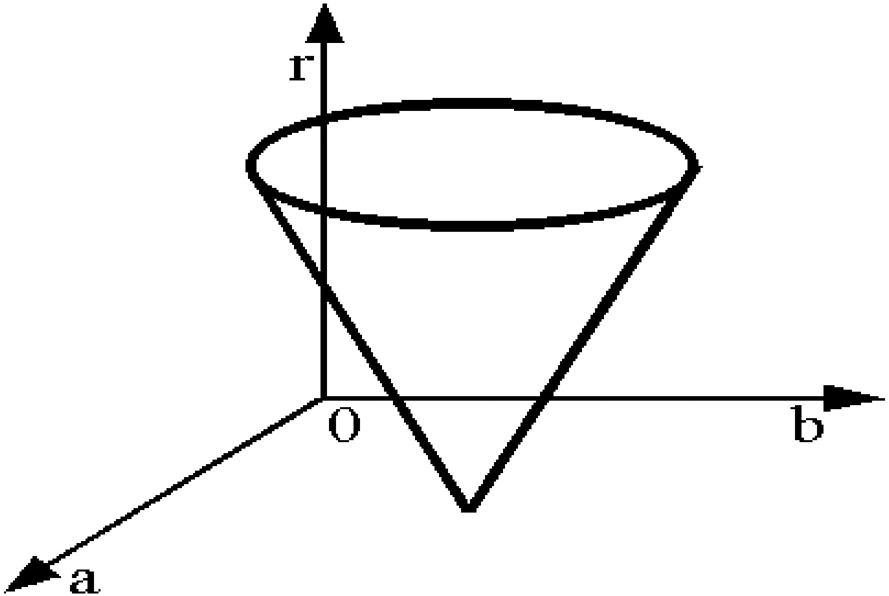
HT for Circles

- Extend HT to other shapes that can be expressed parametrically
- Circle, fixed radius r, centre (a,b)
  - $(x_1-a)^2 + (x_2-b)^2 = r^2$
  - accumulator array must be 3D
  - unless circle radius, r is known
  - re-arrange equation so x_1 is subject and x_2 is the variable
  - for every point on circle edge (x,y) plot range of (x_1,x_2) for a given r

Hough circle Fitting

- Implicit circle equation:
  $(x - a)^2 + (y - b)^2 = r^2$

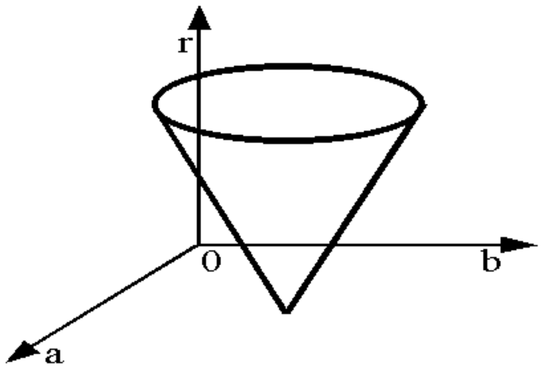
Hough Transform – cont.



A 3-dimensional parameter space for circles in general
Shape = ?

Speedup using gradient info

- For line detection the gradient is @, and so need only to vote for one cell (p,@) where p is
  - $p = x_i \cos @ + y_i \sin @$
- For circle detection
  - the gradient is @, and so need only to vote along a line given by the equations
  - $a = x + r \cos @$, $b = y + r \sin @$



Optimization: HT for circles

- With edge direction
  - edge directions are quantized into 8 possible directions
  - only 1/8 of circle needs take part in accumulator
- Using edge directions
  - a & b can be evaluated from

$$a = x_1 - R\cos(\psi(\mathbf{x}))$$
$$b = x_2 - R\sin(\psi(\mathbf{x}))$$
$$\psi(\mathbf{x}) \in [\phi(\mathbf{x}) - \Delta\phi\,,\ \phi(\mathbf{x}) + \Delta\phi]$$

  - θ = edge direction in pixel x
  - delta θ = max anticipated edge direction error
- Also weight contributions to accumulator A(a) by edge magnitude

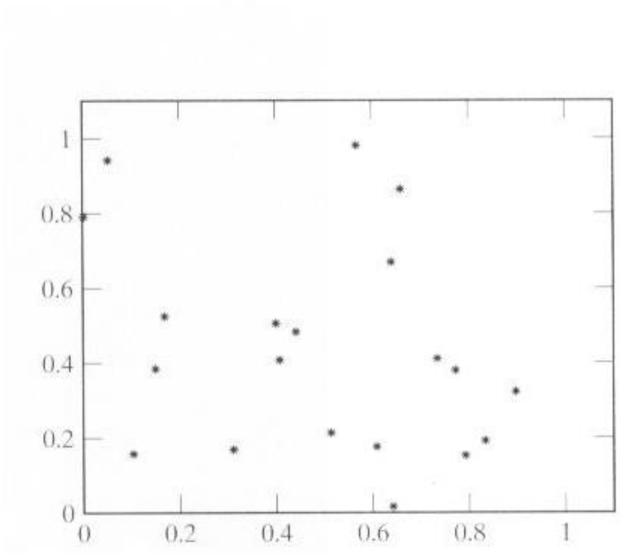
Hough Transform – cont.

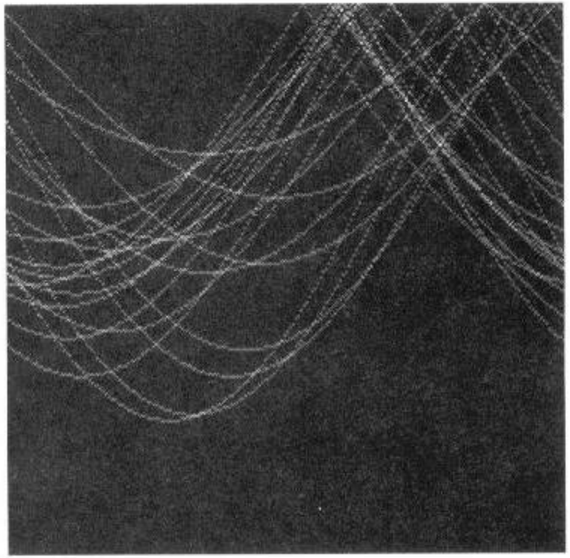
- More complicated shapes
  - Can be used to find shapes with arbitrary complexity
  - … as long as we can describe the shape with some fixed number of parameters
  - The number of parameters required indicates the dimensionality of the accumulator

Generalized Hough Transform

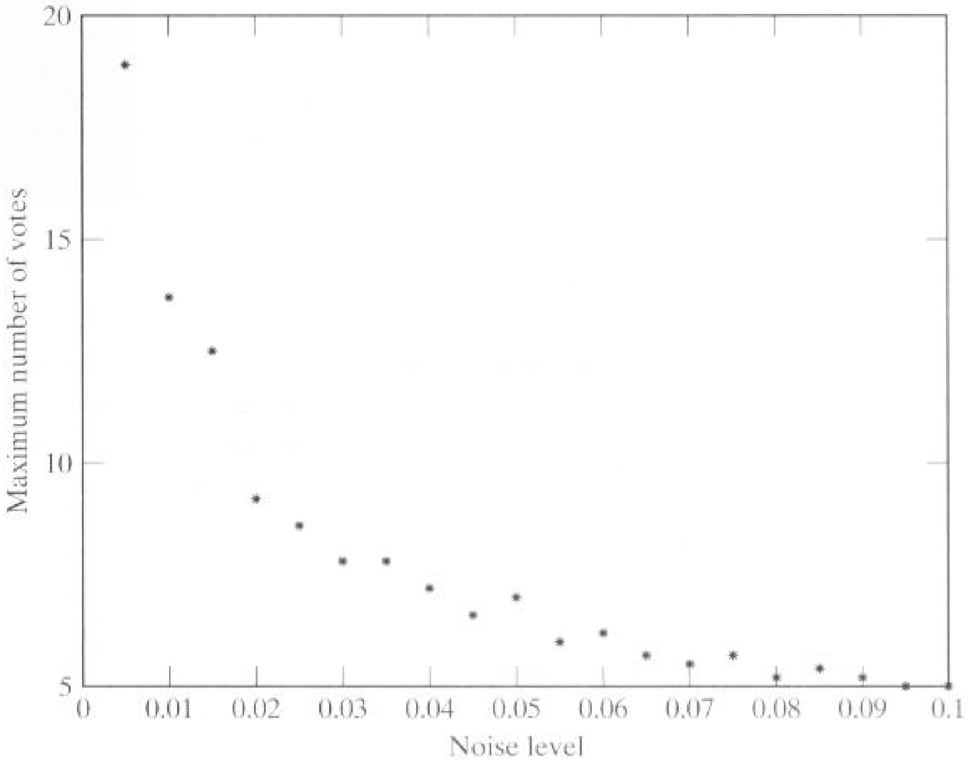
- Some shapes may not be easily expressed using a small set of parameters
  - In this case, we explicitly list all the points on the shape
  - This variation of Hough transform is known as generalized Hough transform

Problems with Hough Transform

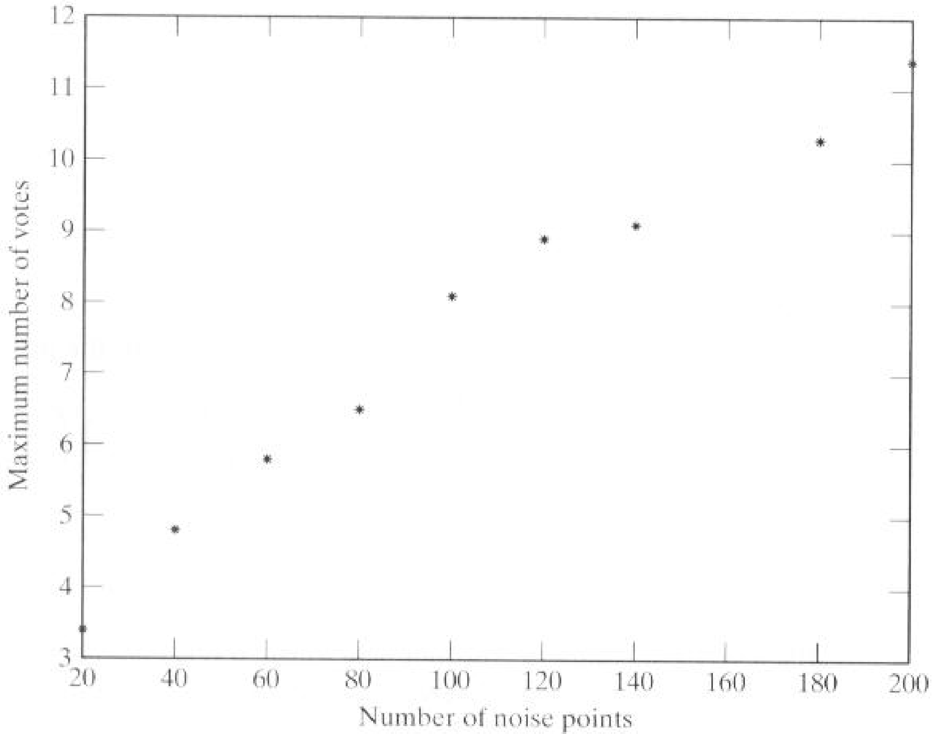




Problems with Hough Transform – cont.



Problems with Hough Transform – cont.



Hough Transform: Philosophy

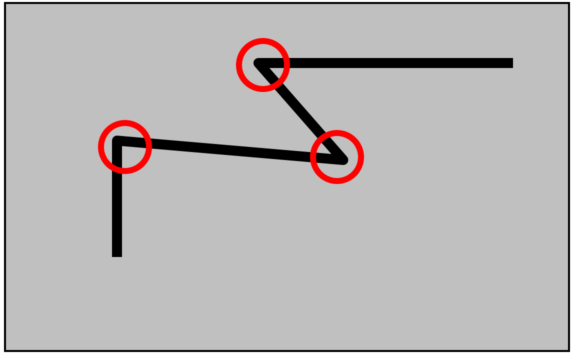
- A method for detecting straight lines, shapes and curves in images.
- Main idea:
  - Map a difficult pattern problem into a simple **peak detection** problem

Hough Transform: Properties

- Advantage - robustness of segmentation results
  - better than edge linking
  - works through occlusion
- Any ***part*** of a straight line can be mapped into parameter space

Harris corner detector

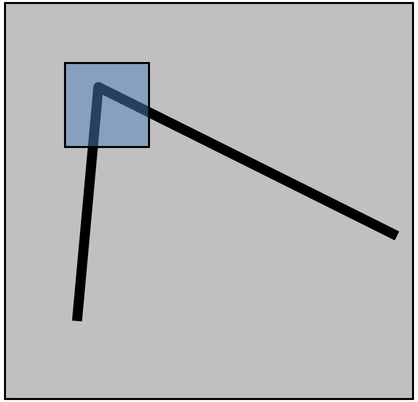
- C.Harris, M.Stephens. “A Combined Corner and Edge Detector”. 1988



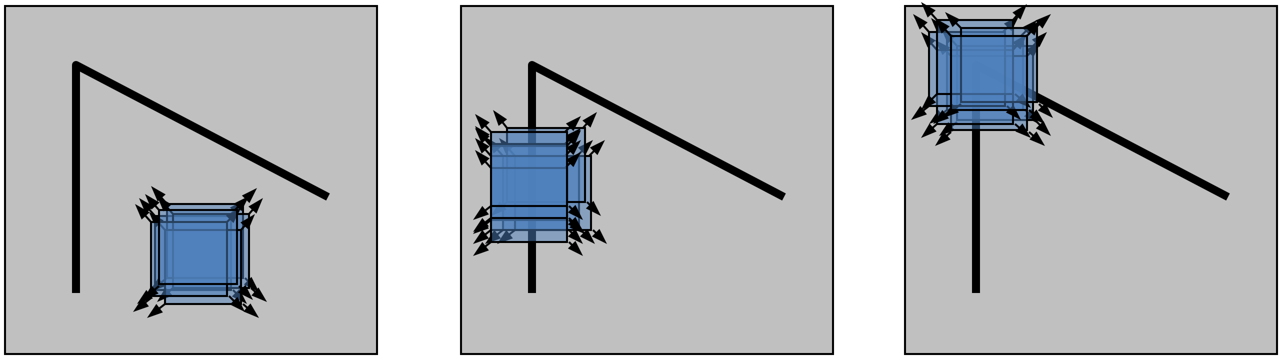
from Rick Szeliski’s lecture notes, CSE576, Spring 05

The Basic Idea

- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



Harris Detector: Basic Idea



"flat" region:
no change in
all directions

"edge":
no change along
the edge direction

"corner":
significant
change in all
directions

Harris Detector: Mathematics

Change of intensity for the shift [u,v]:

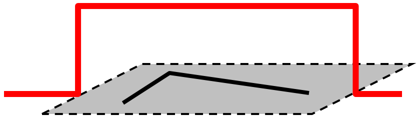
$$E(u,v) = \sum_{x,y} w(x,y) \left[I(x+u, y+v) - I(x,y) \right]^2$$

Window function

Shifted intensity

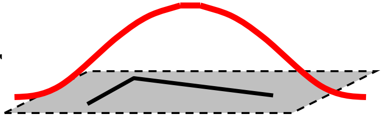
Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

or



Gaussian

Harris Detector: Mathematics

$$E(u,v)=\sum_{x,y} w(x,y)\left[I(x+u,y+v)-I(x,y)\right]^2$$

"I'm sorry, Taylor, but Fourier had one of the best series of 1807."

Harris Detector: Mathematics

$$E(u,v)=\sum_{x,y} w(x,y)\left[I(x+u,y+v)-I(x,y)\right]^2$$

For small shifts $[u,v]$ we have a *bilinear* approximation:

$$E(u,v)\cong[u,v]\ M\begin{bmatrix}u\\v\end{bmatrix}$$

where M is a 2×2 matrix computed from image derivatives:

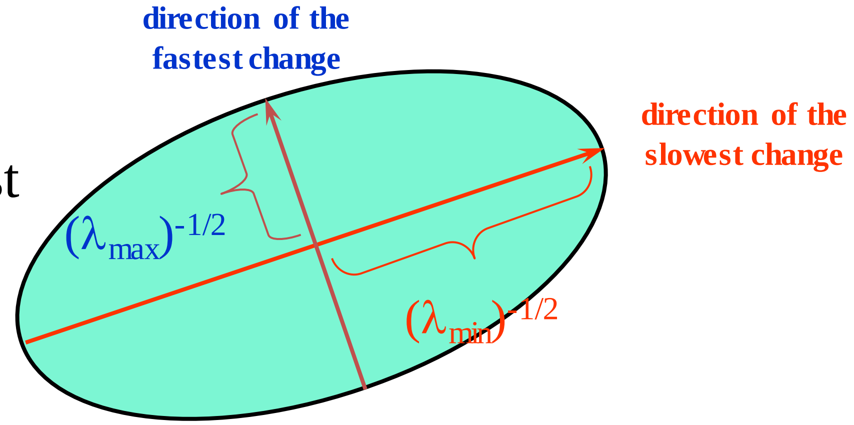
$$M=\sum_{x,y} w(x,y)\begin{bmatrix}I_x^2 & I_xI_y\\ I_xI_y & I_y^2\end{bmatrix}$$

Harris Detector: Mathematics

Intensity change in shifting window: eigenvalue analysis

$$E(u,v) \cong [u,v] \; M \begin{bmatrix} u \\ v \end{bmatrix}$$

λ_1, λ_2 – eigenvalues of M



Ellipse $E(u,v)$ = const

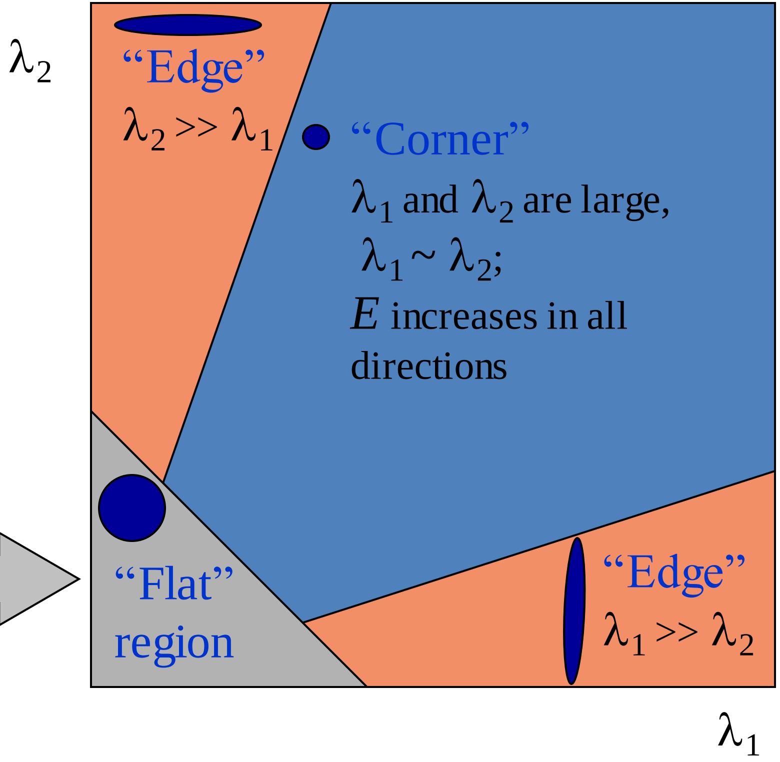
$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Harris Detector: Mathematics

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Classification of image points using eigenvalues of M:

λ_1 and λ_2 are small;
E is almost constant in all directions



Harris Detector: Mathematics

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Measure of corner response:

$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

(k – empirical constant, k = 0.04-0.06)

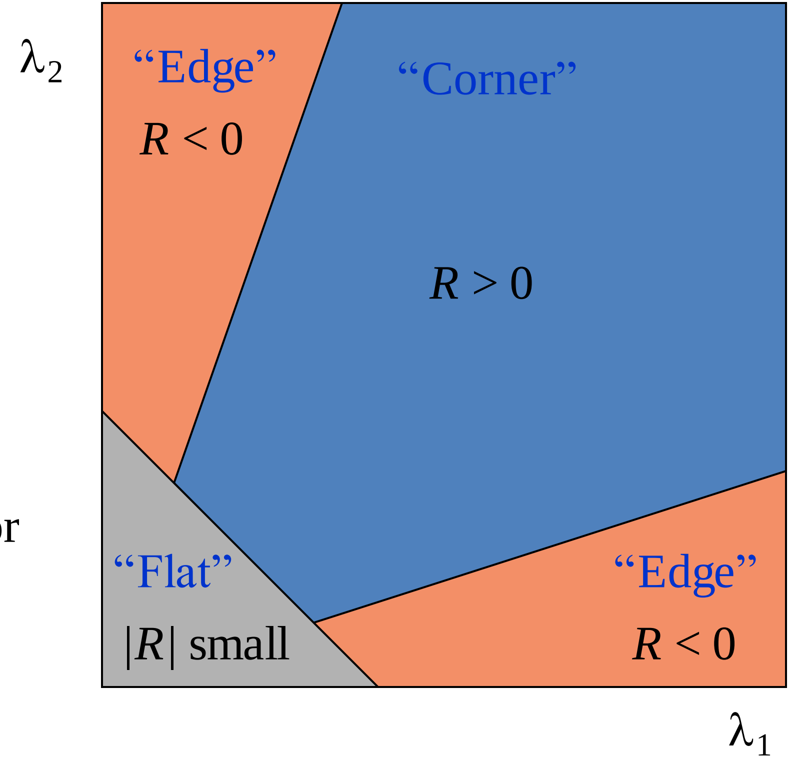
Harris Detector: Mathematics

$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

$$\det M = \lambda_1 \lambda_2$$

$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

- R depends only on eigenvalues of M
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region



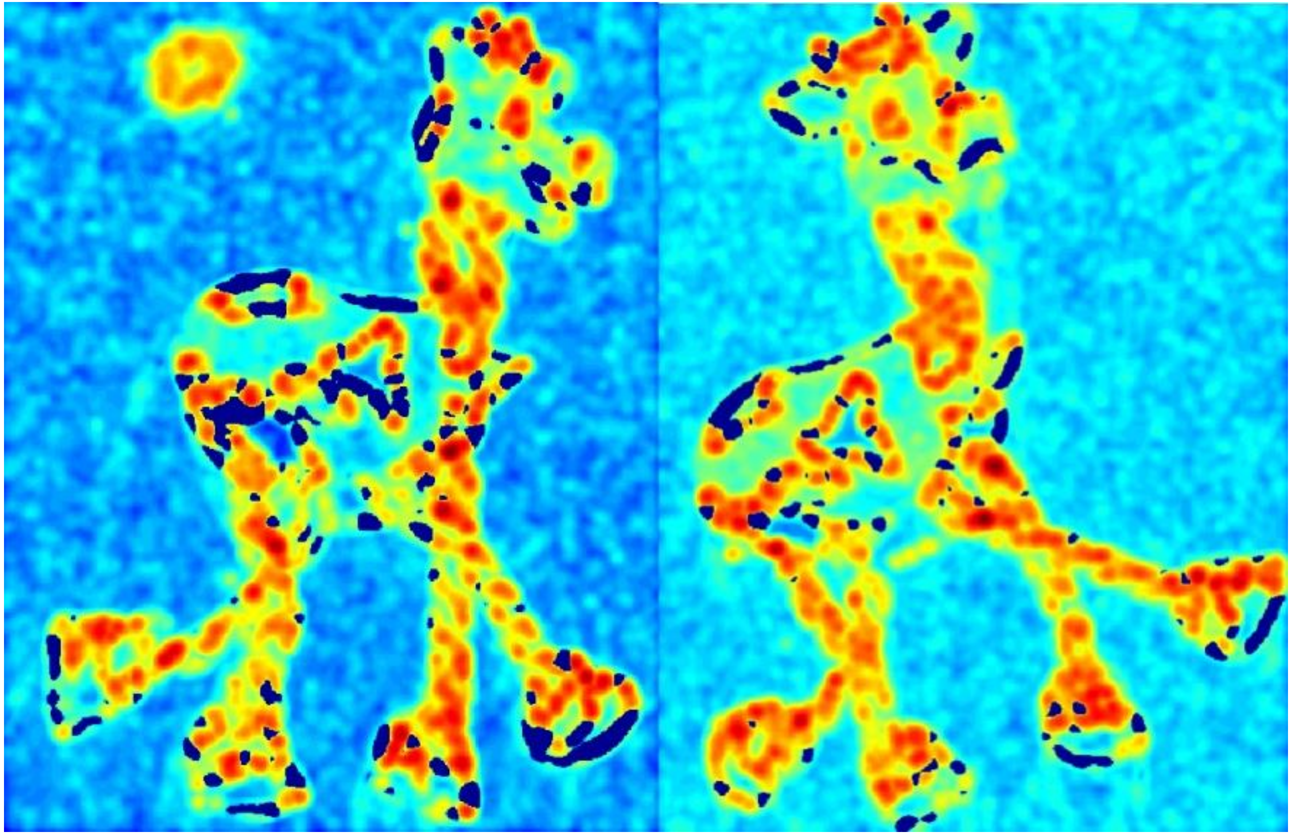
Harris Detector

- The Algorithm:
  - Find points with large corner response function R (R > threshold)
  - Take the points of local maxima of R

Harris Detector: Workflow

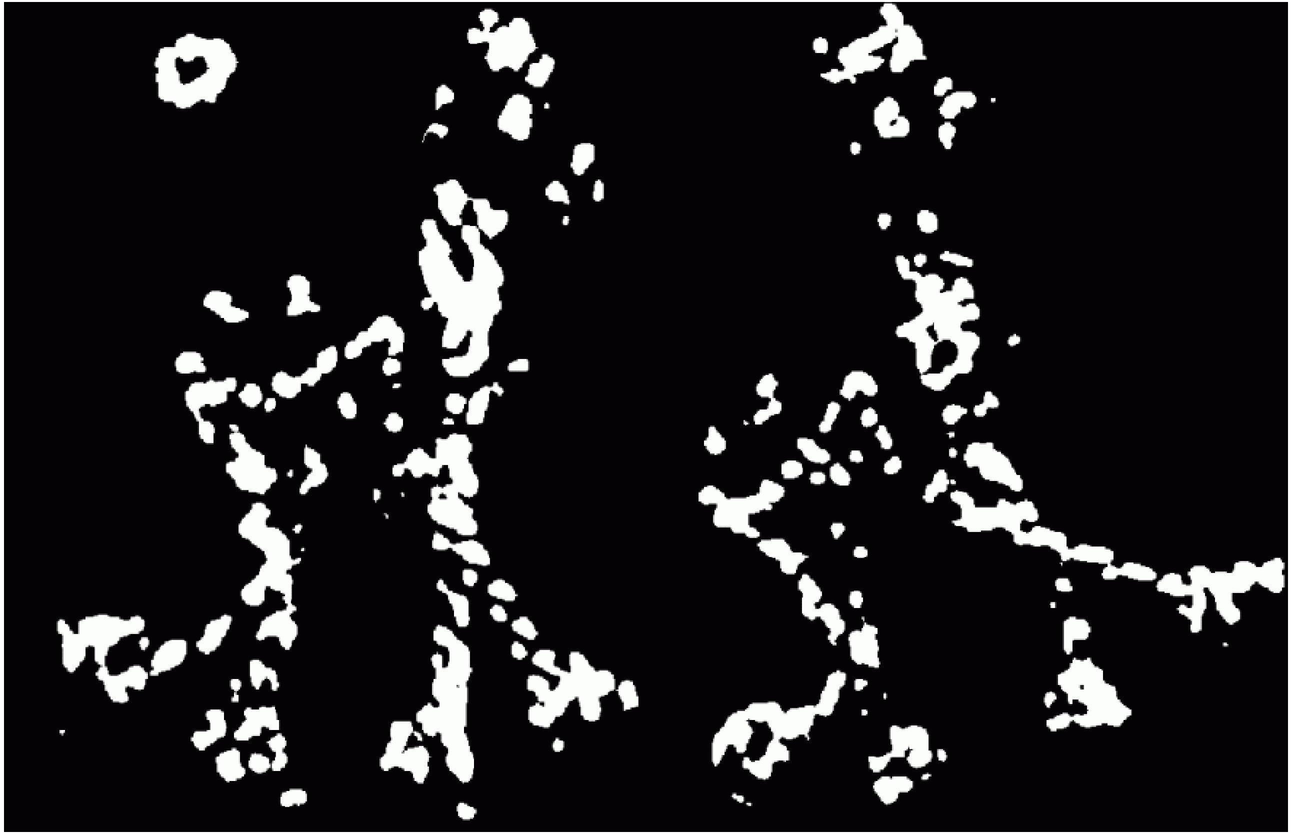
Harris Detector: Workflow

Compute corner response R



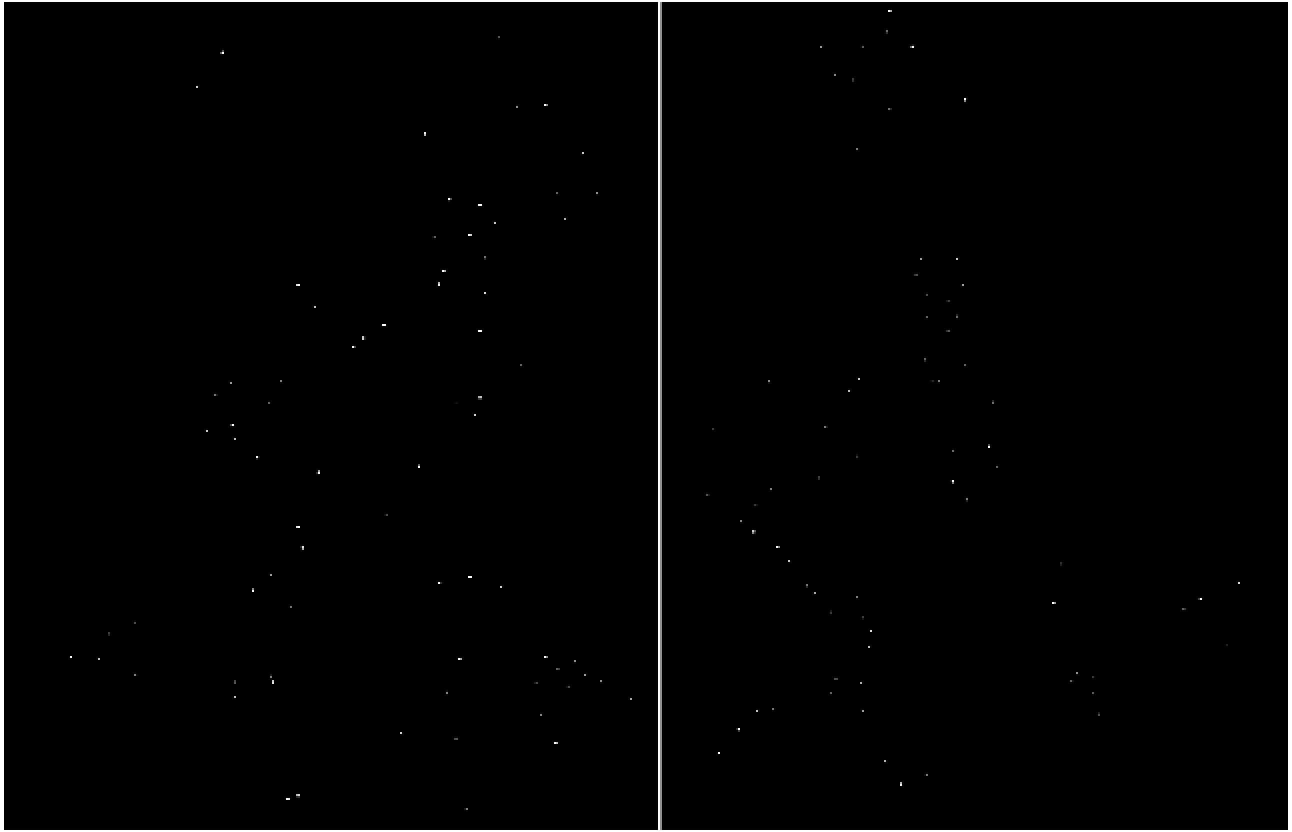
Harris Detector: Workflow

Find points with large corner response: R>threshold



Harris Detector: Workflow

Take only the points of local maxima of R



Harris Detector: Workflow

Harris Detector: Summary

- Average intensity change in direction $[u,v]$ can be expressed as a bilinear form:

$$E(u,v) \cong [u,v] \; M \begin{bmatrix} u \\ v \end{bmatrix}$$

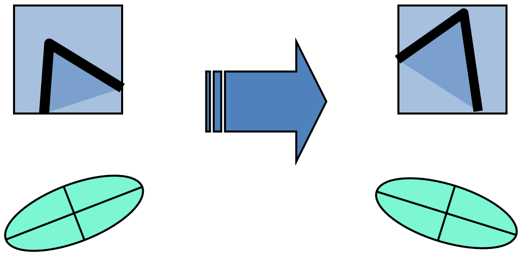
- Describe a point in terms of eigenvalues of M: *measure of corner response*

$$R = \lambda_1 \lambda_2 - k\left(\lambda_1 + \lambda_2\right)^2$$

- A good (corner) point should have a *large intensity change* in *all directions*, i.e. R should be large positive

Harris Detector: Some Properties

- Rotation invariance



Ellipse rotates but its shape (i.e. eigenvalues) remains the same

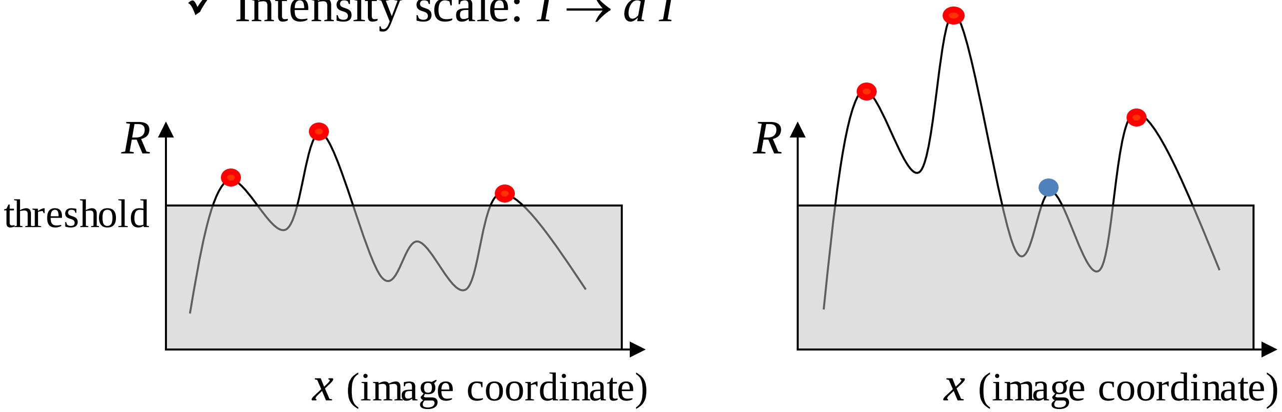
Corner response R is invariant to image rotation

Harris Detector: Some Properties

- Partial invariance to *affine intensity* change

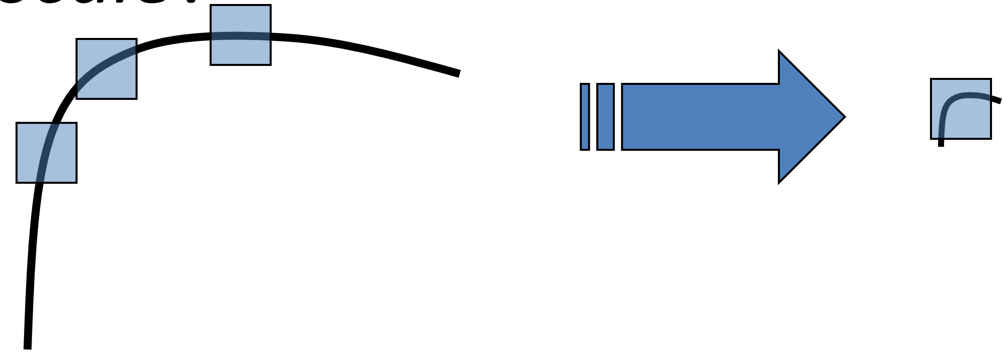
✓ Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$

✓ Intensity scale: $I \rightarrow a\, I$



Harris Detector: Some Properties

- But: non-invariant to *image scale*!



All points will be classified as edges

Corner !