UNIVERSITÀ DI PARMA

Edges Detection
Lines Detection

Summary

UNIVERSITÀ DI PARMA

- Edge detection
  - What is an edge?
    - Are they important?
  - Canny Edge Detector
- Lines
  - Hough Transform
  - Generalized Hough Transform

UNIVERSITÀ DI PARMA

Edges Detection

Edge filtering

UNIVERSITÀ DI PARMA

- Q: What is an edge? A: A discontinuity in intensity!

Edge detection

UNIVERSITÀ DI PARMA

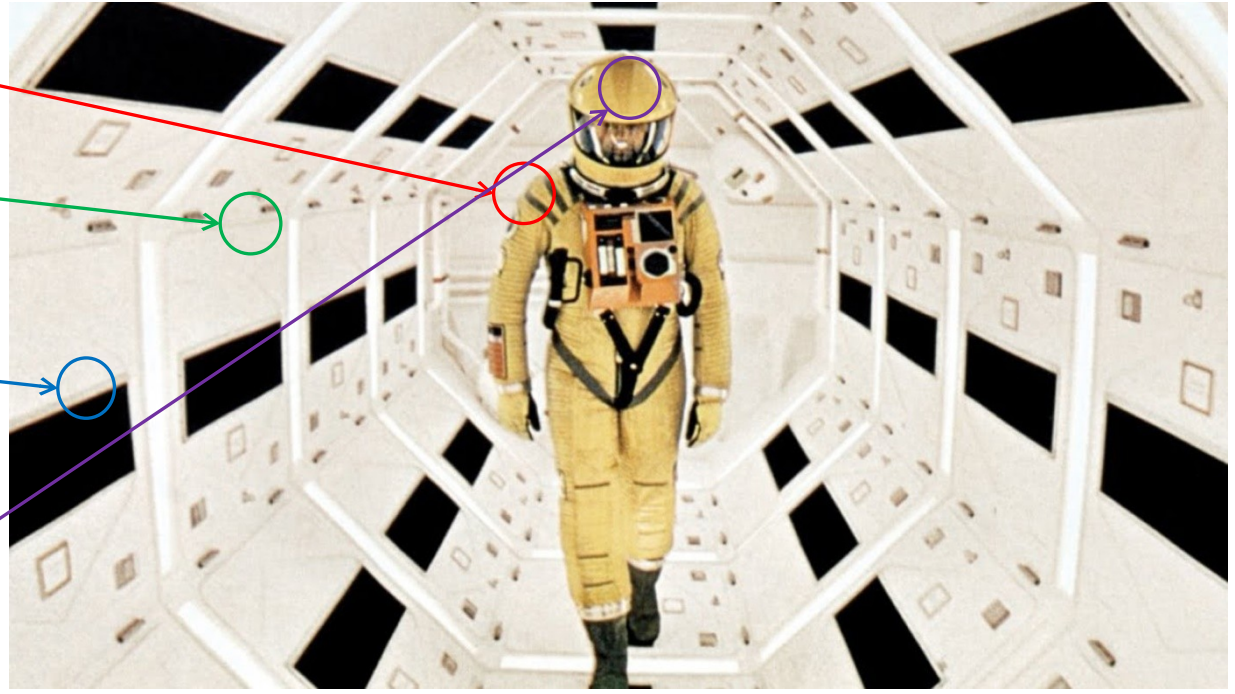
- Q: From where intensity discontinuities (aka edges) come from?

Depth differences

Orientations of surfaces

Properties of surfaces

Different illuminations (i.e. shadows or lights)



Types of edges

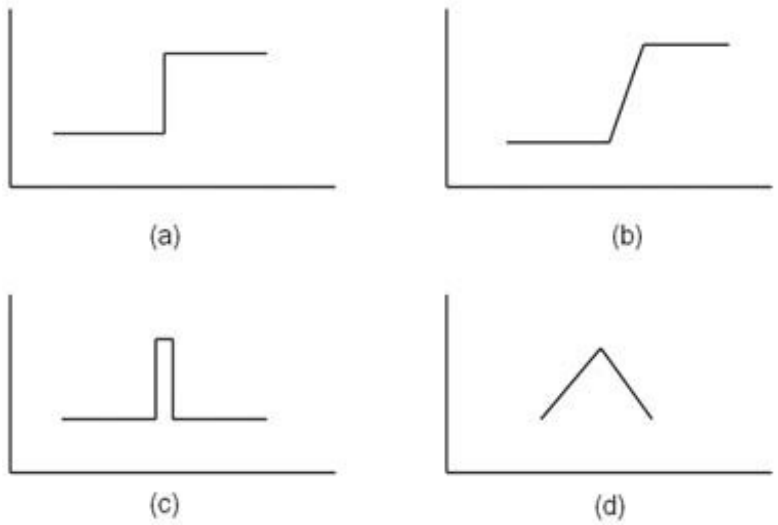
UNIVERSITÀ DI PARMA

(a) Step edge: abrupt change of intensity

(b) Ramp edge: smooth change

(c) Ridge/line edge: abrupt change followed by a return to starting value

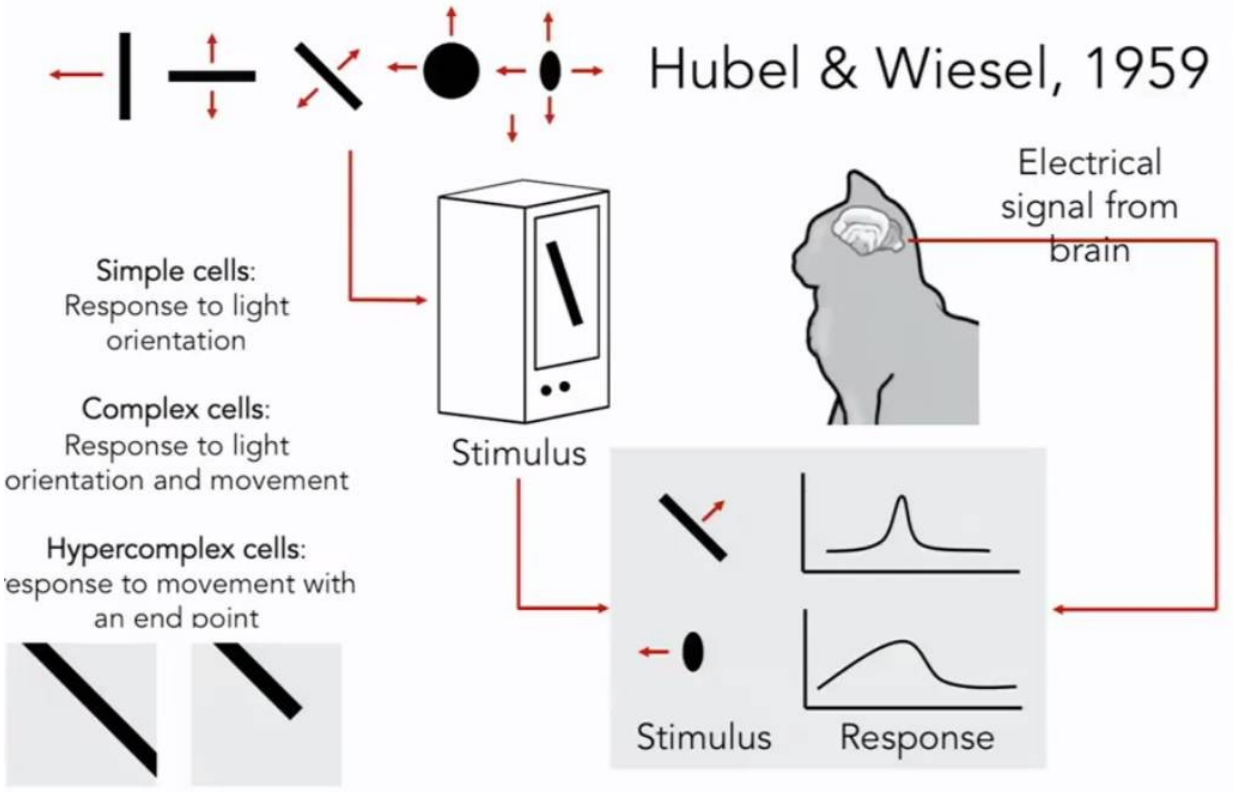
(d) Roof edge: a ridge edge with a smoother change



Edge Importance

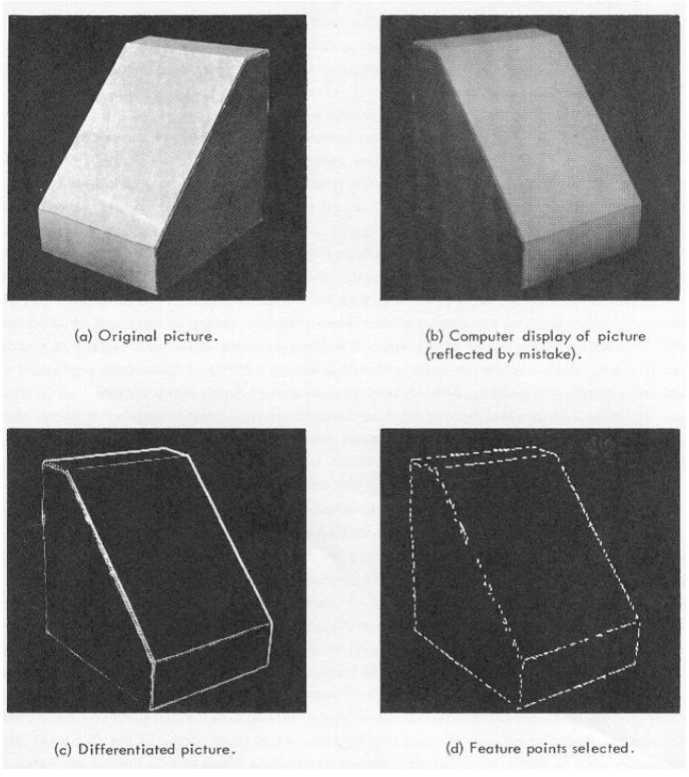
UNIVERSITÀ DI PARMA

- Why edges are important?



3D from edges

- Why edges are important?



Larry Roberts, 1963

Edge Detection

UNIVERSITÀ DI PARMA

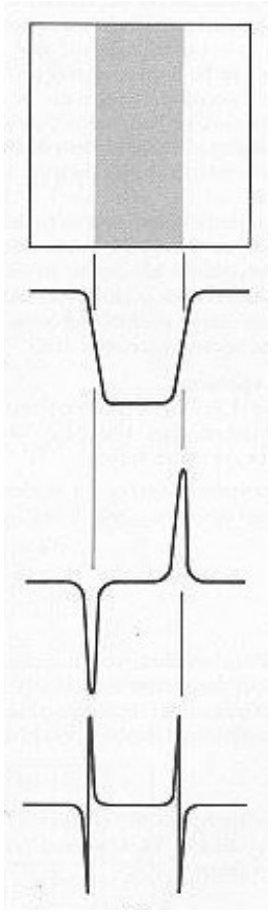
- Edges can help:
  - Extract information
  - Detect objects
  - Detect object shapes

Simple Edge Detection

UNIVERSITÀ DI PARMA

- Already (partially) discussed
- Small recap:
  - Smoothing ⟶ to reduce noise
  - Derivative Horizontal & Vertical filtering ⟶ extract gradient

Gradient



Image

Horizontal Intensity

First derivattive

Second derivattive

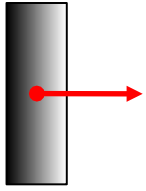
Edges are the maximum/minimum in first derivative

Edges corresponds to 0 crossings in second derivative

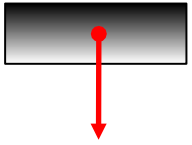
How we can compute gradient?

UNIVERSITÀ DI PARMA

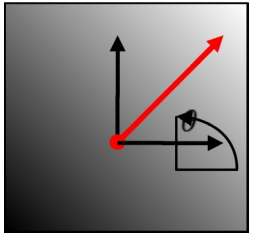
- Gradient of image:



$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$



$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$



$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

- Gradient is a vector:
  - Magnitude/Intensity
  - Direction

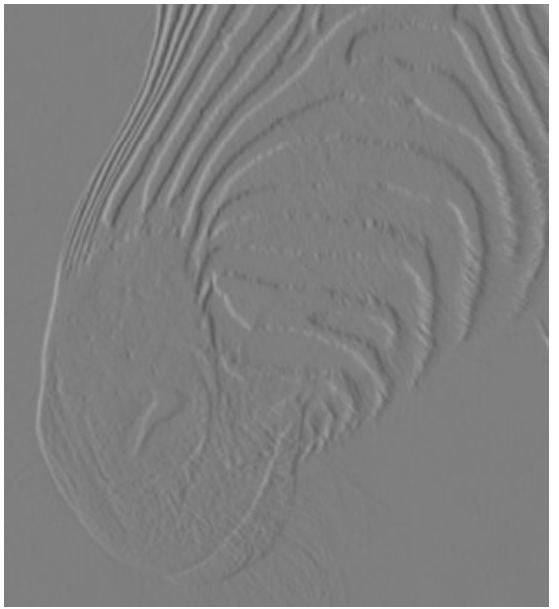
$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

Example

f

$\frac{\partial f}{\partial y}$



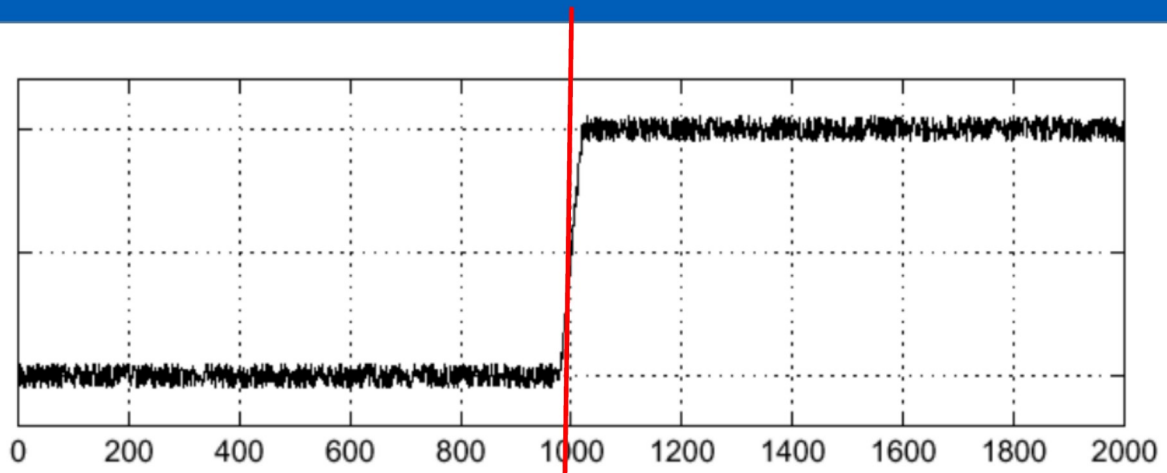
$\frac{\partial f}{\partial x}$

Forsyth & Ponce – *Computer Vision A Modern Approach*

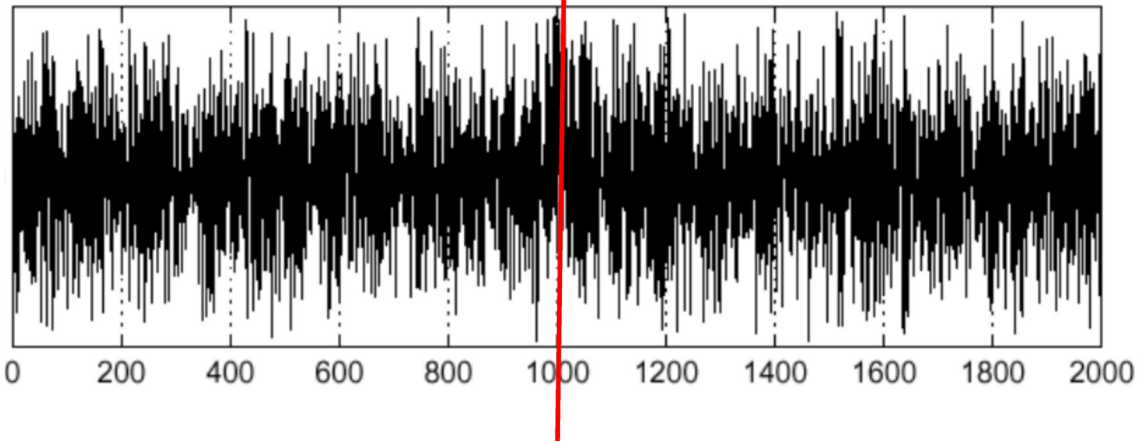
Gradient and Noise

UNIVERSITÀ DI PARMA

$f(x)$

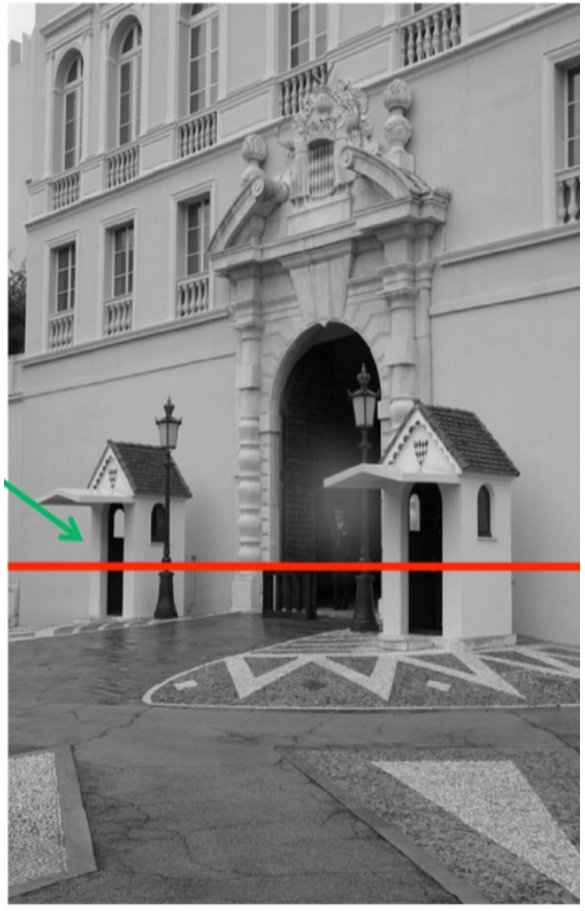


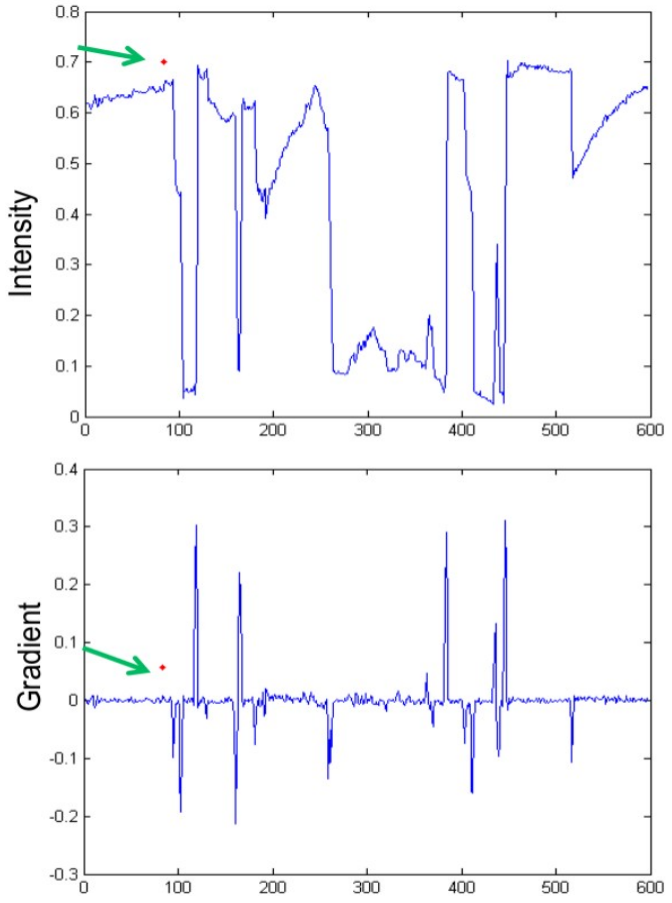
$\frac{d}{dx} f(x)$



From: S. Seitz

Gradient in real images





We have large peaks but also some noise

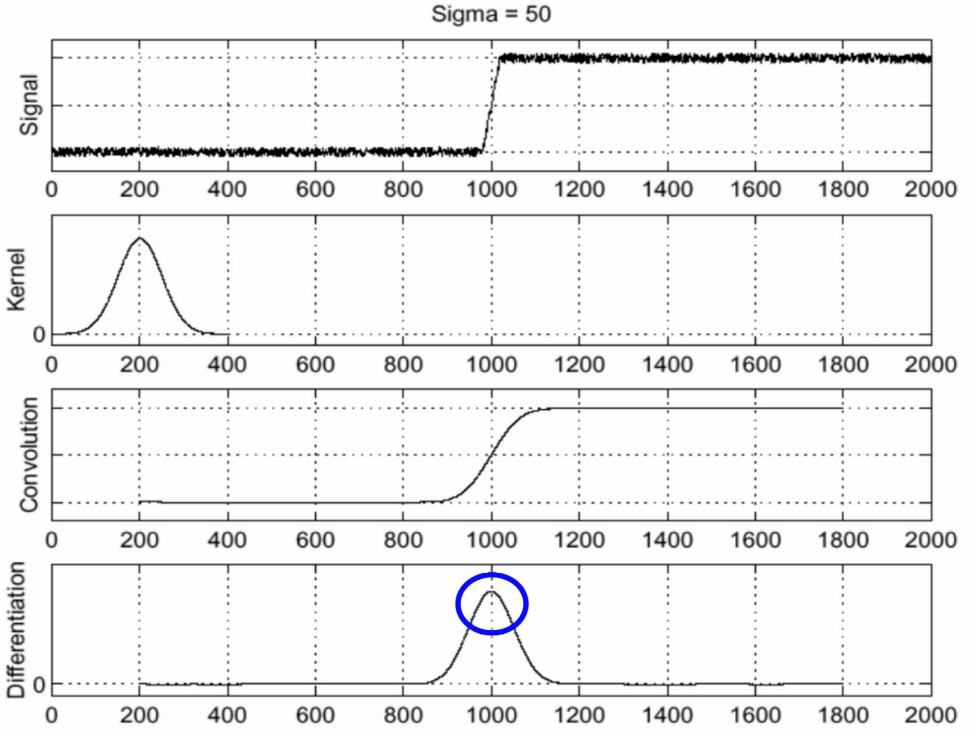
From: Fei Fei Li

Noise effect

UNIVERSITÀ DI PARMA

- Derivative filter have a great sensitivity to noise
  - Noise $\longrightarrow$ uniform areas are not so uniform
  - Pixels can be very different wrt neighboors
- Smoothing filtering allows to reduce noise effects

Smoothing & Derivative filtering



f

g

$f * g$

$\frac{d}{dx}(f * g)$

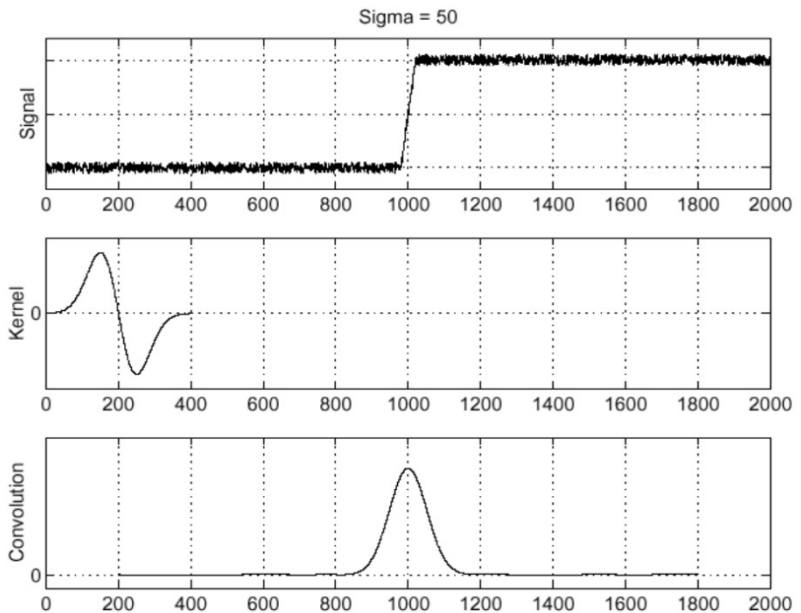
Edges correspond to peaks in derivative of f and g convolution

From: S. Seitz

Convolution properties

- Convolution is an associative operation
- We can optimize

$$\frac{\partial}{\partial x}(f * g) = f * \frac{\partial}{\partial x} g$$



f

$\frac{d}{dx} g$

$f * \frac{d}{dx} g$

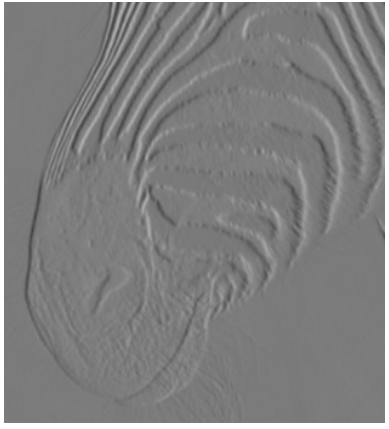
From: S. Seitz

Tradeoff

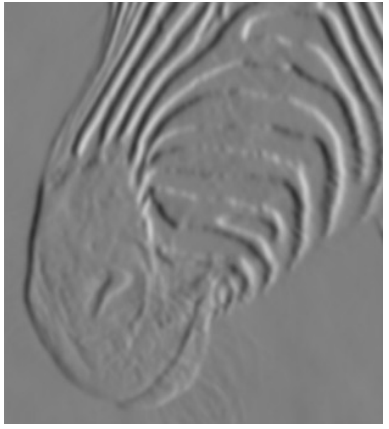
UNIVERSITÀ DI PARMA

- Between smoothing and localization

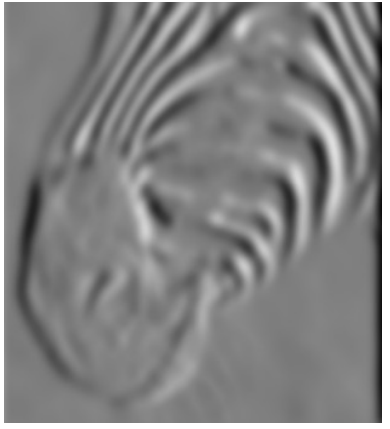
Input image



$\sigma = 1px$



$\sigma = 3px$



$\sigma = 7px$

From: Forsyth & Ponce – *Computer Vision A Modern Approach*

Sobel Edge Detector

UNIVERSITÀ DI PARMA

- Composition of a derivative operator and an averaging one
- Anyway a little rough for high frequency variations

-1	-2	-1
0	0	0
1	2	1

$$\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

-1	0	1
-2	0	2
-1	0	1

$$\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$$

Sobel Edge Detector: threshold

UNIVERSITÀ DI PARMA

- Sobel operator allows to compute the gradient
  - Not actual edges
- We need a threshold

Sobel Edge Detector: edge thickness

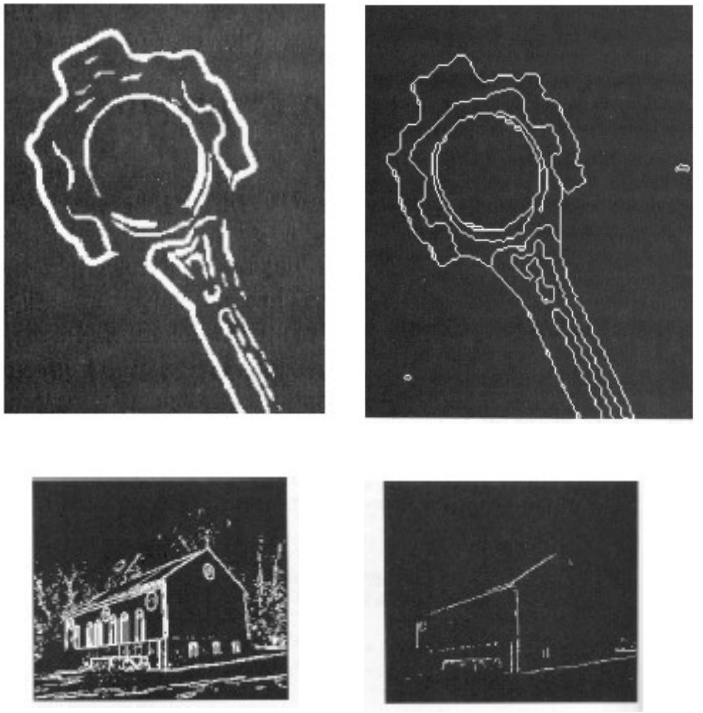
UNIVERSITÀ DI PARMA

- Each edge is “fat”

Canny Edge Detector

UNIVERSITÀ DI PARMA

- Sobel is a quick&dirt solution with some issue:
  - Threshold
  - Fat edges
- There is a better approach?
- Yes ⟶ Canny Edge Detector
  - Gradient
  - Thinning
  - Thresholding



Canny Edge Detector

UNIVERSITÀ DI PARMA

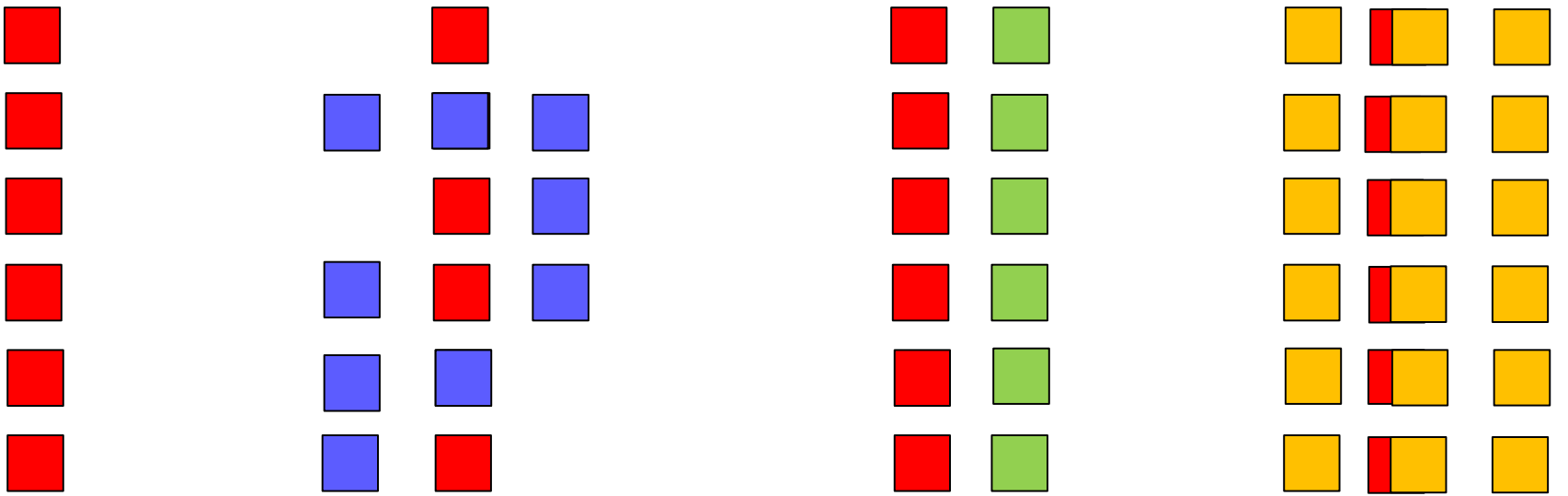
- Widely used
  - It performs very well!
- Criteria
  - Accuracy: reduce false positives (also negatives)
  - Localization: detected edges should match real edges as much as possible
  - Uniqueness: only one detection for each real edge

J. Canny, *A Computational Approach to Edge Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol 8, No. 6, Nov 1986

Canny Edge Detection

UNIVERSITÀ DI PARMA

- Examples:



Real edge

Accuracy

Localization

Uniqueness

Canny Edge Detector

UNIVERSITÀ DI PARMA

- 3 Main steps
  - Gradient detection: compute horizontal & vertical derivatives
    - Magnitude & Direction
  - Thinning: non local maxima suppression
    - 1 pixel wide gradients
  - Thresholding: hysteresis approach
    - Final result

Input

Gradient Detection

UNIVERSITÀ DI PARMA

- Sobel is ok!

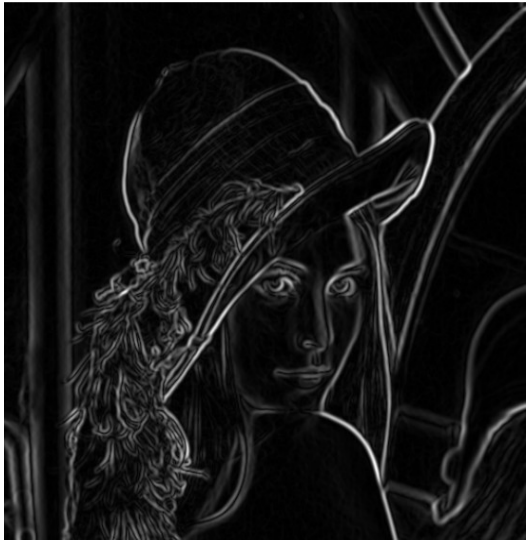
Horizontal Gradient G_x

Vertical Gradient G_y

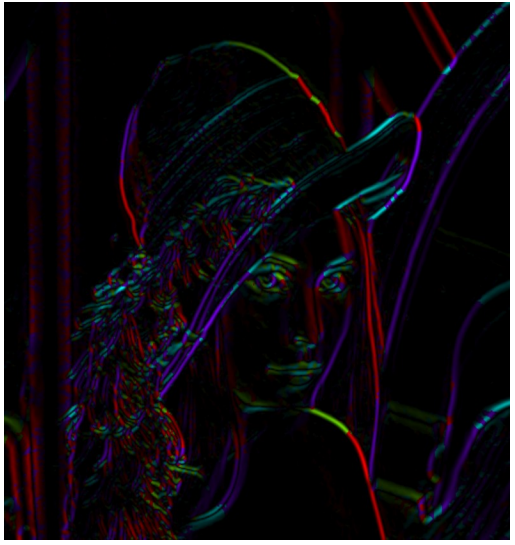
Gradient Detection

UNIVERSITÀ DI PARMA

- We have 2 results: gradient magnitude and gradient direction



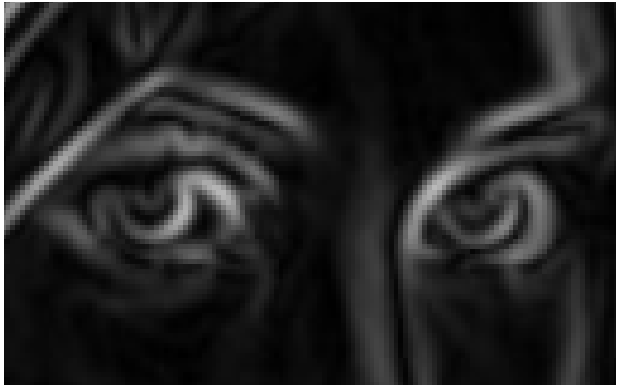
Magnitude

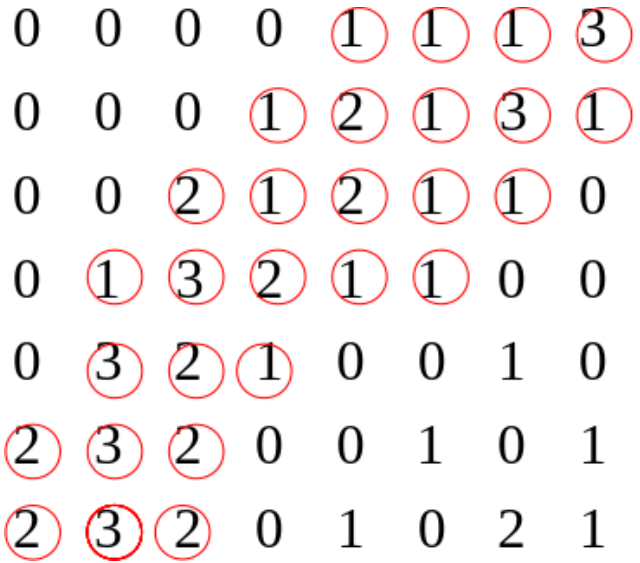


Orientation

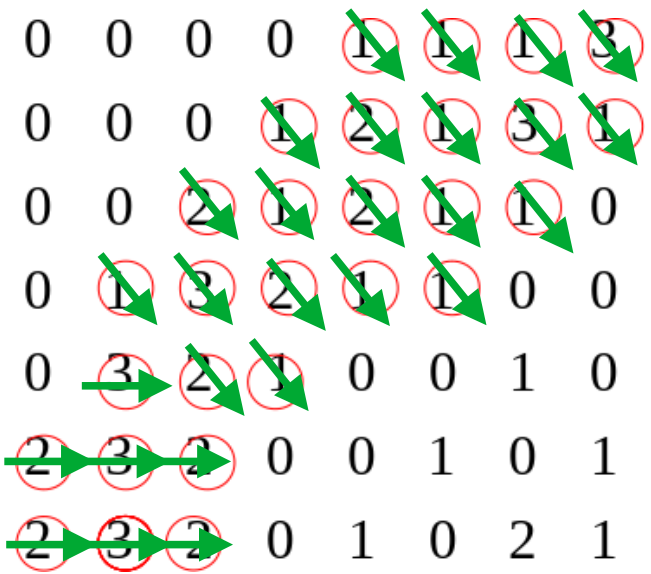
$$M = \sqrt{G_x^2 + G_y^2}$$

$$\Theta = \tan^{-1}\left(\frac{G_y}{G_x}\right) = atan2\left(G_y, G_x\right)$$





- Non Maxima Suppression
  - Only local maxima must survive
  - Orientation and gradient are used to detect local maxima
    - Bilinear interpolation or
    - A 3×3 grid



- Non Maxima Suppression
  - Only local maxima must survive
  - Orientation and gradient are used to detect local maxima
    - Bilinear interpolation or
    - A 3×3 grid

- Non Maxima Suppression
  - Only local maxima must survive
  - Orientation and gradient are used to detect local maxima
    - Bilinear interpolation or
    - A 3×3 grid

Non Maxima Suppression

UNIVERSITÀ DI PARMA

- Local Maxima does not mean “strong edges”
- One threshold is not enough
- Two thresholds:
  - Under T_{low} $\longrightarrow$ eliminate edge
  - Over T_{high} $\longrightarrow$ keep edge
  - Between T_{low} and T_{high}?

Hysteresis

UNIVERSITÀ DI PARMA

- Edges above T_{high} are iteratively grown using adjacent edges above T_{low}
  - Similar to a labeling technique
- At the end edges < T_{high} are removed

Hysteresis

UNIVERSITÀ DI PARMA

Lines Detection

Summary

UNIVERSITÀ DI PARMA

- Edges and lines
- Hough transform
- Not only lines
  - Hough transform can be generalized

Beyond edges

UNIVERSITÀ DI PARMA

- Edges are not lines… still a low level feature

Lines Detection

UNIVERSITÀ DI PARMA

- First approach: determine pixel alignment
  - Expensive!
  - We have to match each pixel to other pixels ⟶ straight lines
    - Check whether other pixels are on those lines
  - For n pixels we have $n(n-1)/2 \sim n^2$ potential straight lines
    - $(n-2)[n(n-1)/2] \sim n^3$ matches
- Real approach: Hough Transform

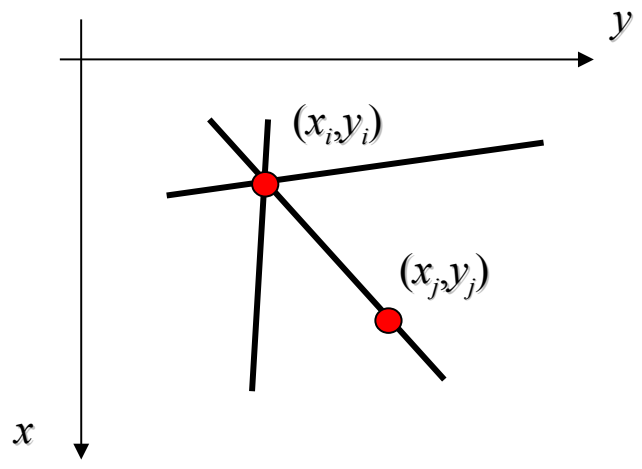
Hough Transform

UNIVERSITÀ DI PARMA

- The family of straight lines that pass through a given point (x_i, y_i) is defined as:

  $y_i = ax_i + b$

- For each parameter set (a, b) we have a given straight line
- Actually x_i and y_i are constant, while a and b are the unknowns



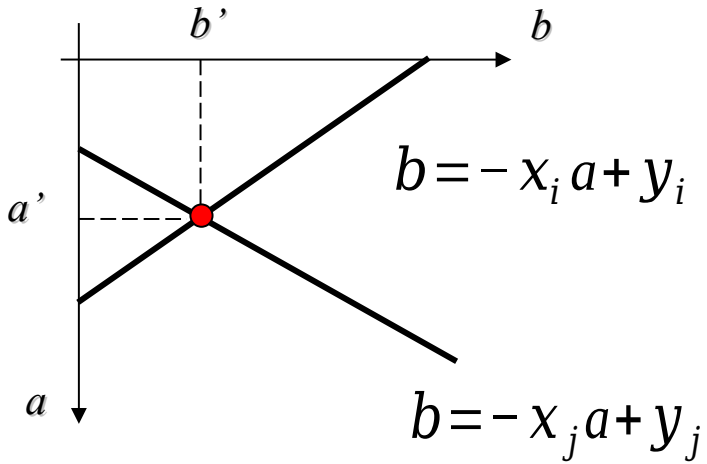
Hough Transform

UNIVERSITÀ DI PARMA

- Therefore, the previous equation can be written as:

  $b = -x_i a + y_i$

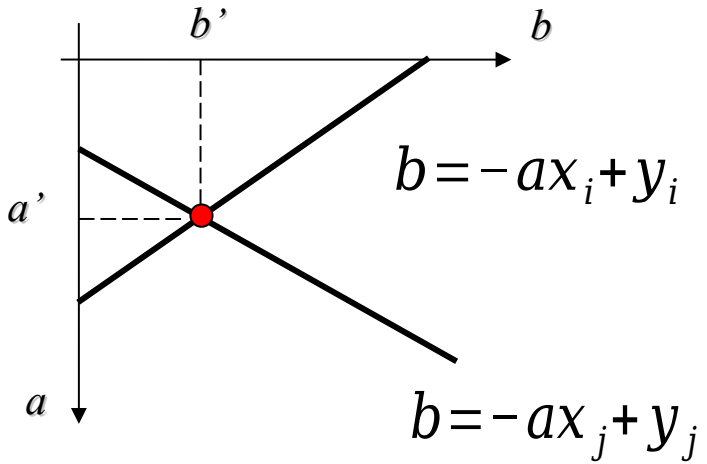
- That represents all parameters values for a given point (x_i, y_i)
  - Still a straight line but in parameter space!
- This can be done also for other points, i.e. given point (x_j, y_j)
- The two straight lines intersect in (a', b')



Hough Transform

UNIVERSITÀ DI PARMA

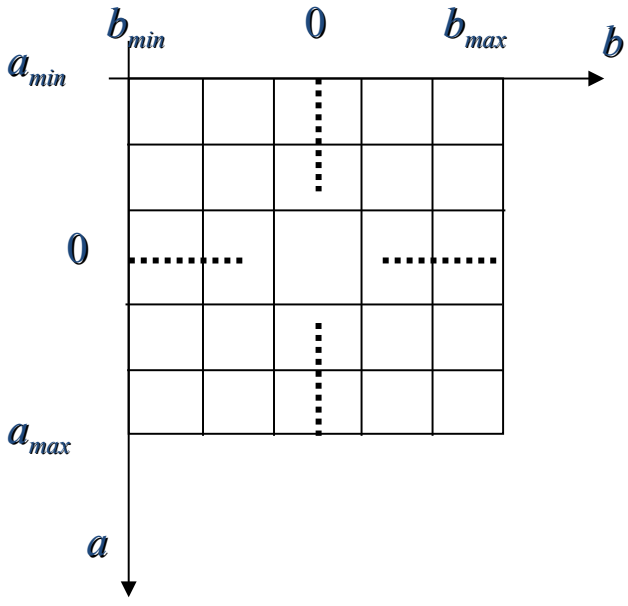
- (a', b') are then the parameters that define the straight line that pass through both (x_i, y_i) and (x_j, y_j)
- More generally, all points that lie on the same line in the x,y space generate straight lines that intersect in a specific point in the a,b space
- This is the underlying rationale of the Hough Transform



Hough Transform

UNIVERSITÀ DI PARMA

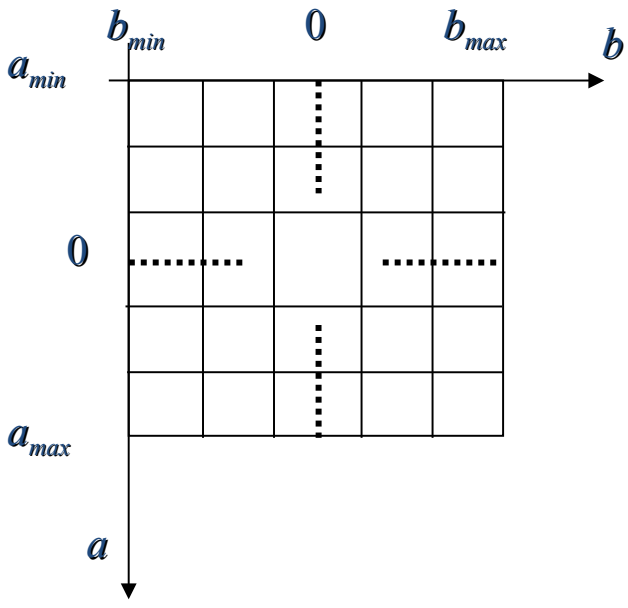
- Main steps:
  - Define a discrete accumulator matrix for the a,b space
  - For each (x_k, y_k) compute all possible b values using
    - All a values in the a_{min} - a_{max} range
    - $b = -x_k a + y_k$
  - Increase the corresponding a,b cells in the accumulator matrix



Hough Transform

UNIVERSITÀ DI PARMA

- At the end of the process each accumulation cell gives us information on how much pixels lie on the same straight line
- i.e. if a cell (a_i, b_i) contains the value M this means that in the image we have M points that lie on the $y=a_ix+b_i$ line
- A threshold is generally used to extract biggest values
  - Namely detected straight lines



Hough Transform

UNIVERSITÀ DI PARMA

- If the a axis is subdivided in k intervals for each (x, y) point we get k values for b
- For n points we need to compute $n \times k$ values
  - O(n) better than $O(n^3)$
- Open issues:
  - Which value for k?
  - What range for a?
  - Vertical lines?

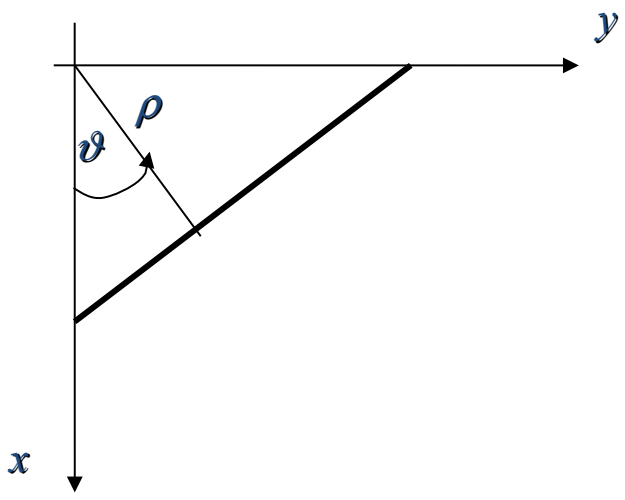
Hough Transform

UNIVERSITÀ DI PARMA

- To solve some issues Euclidean space is used instead of Cartesian one

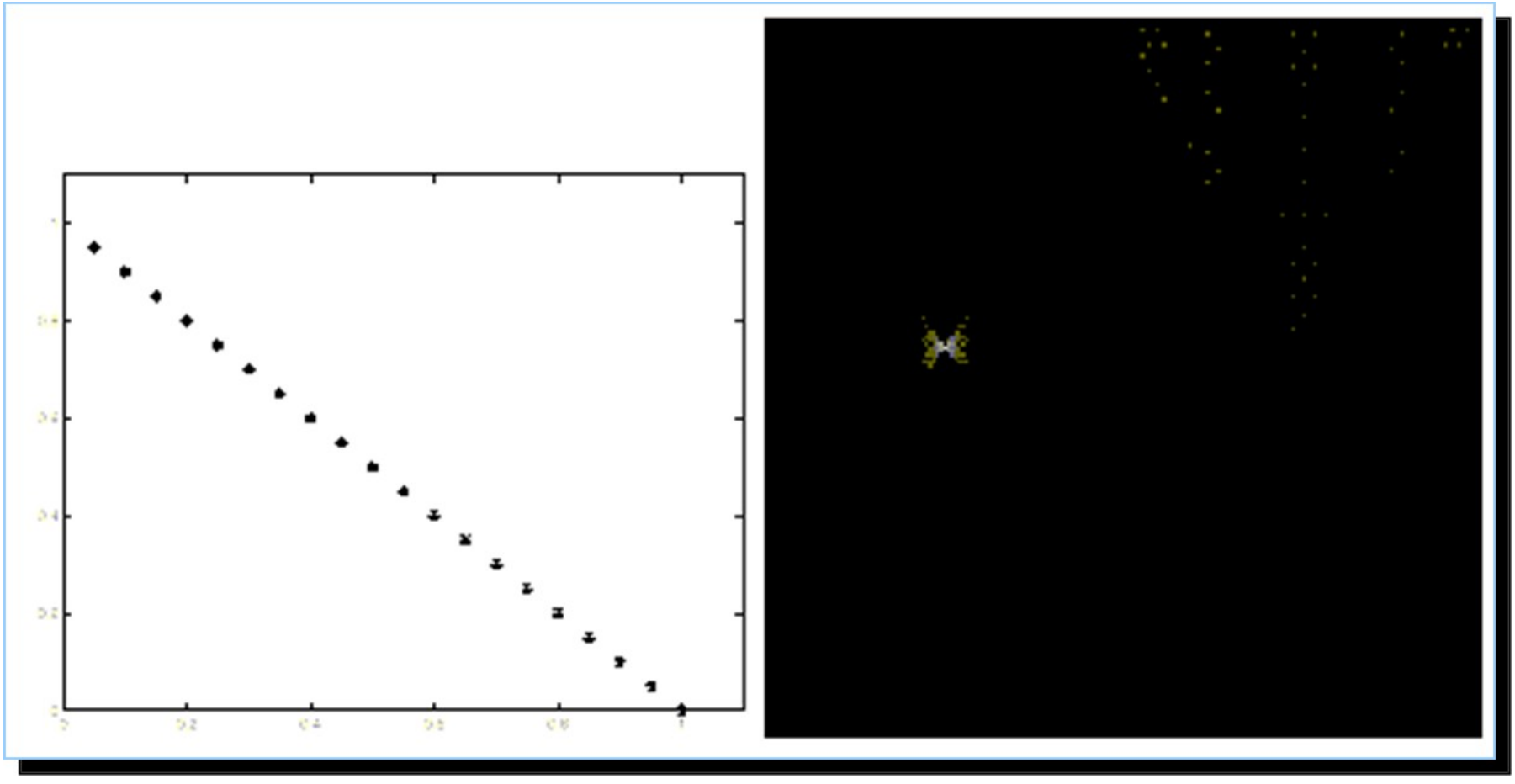
$$x\cos\theta + y\sin\theta = \rho$$

- Both parameters have a limited range
  - θ is an angle
  - ρ is limited by image diagonal
- Similar procedure as before
- In the parameters space we have curves



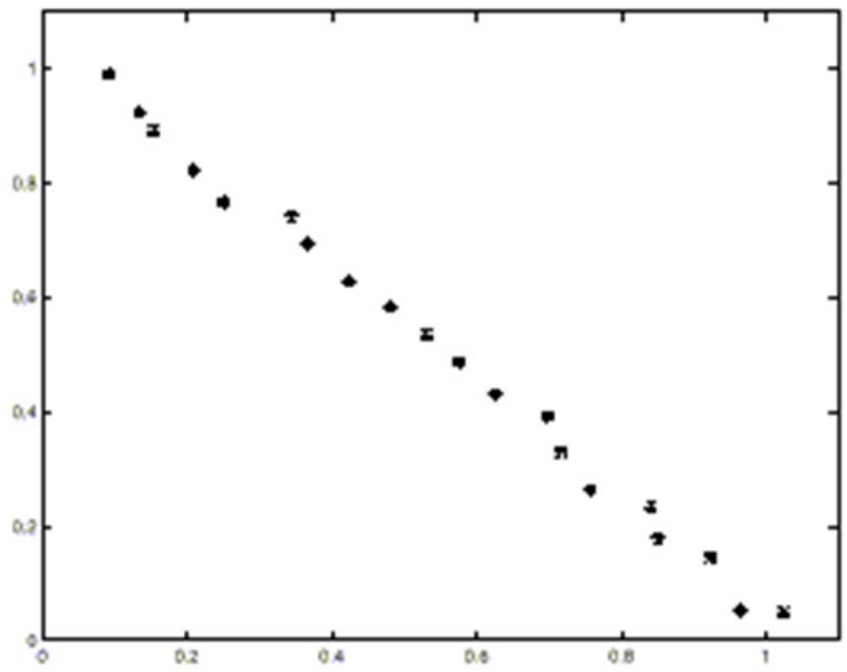
Hough Transform

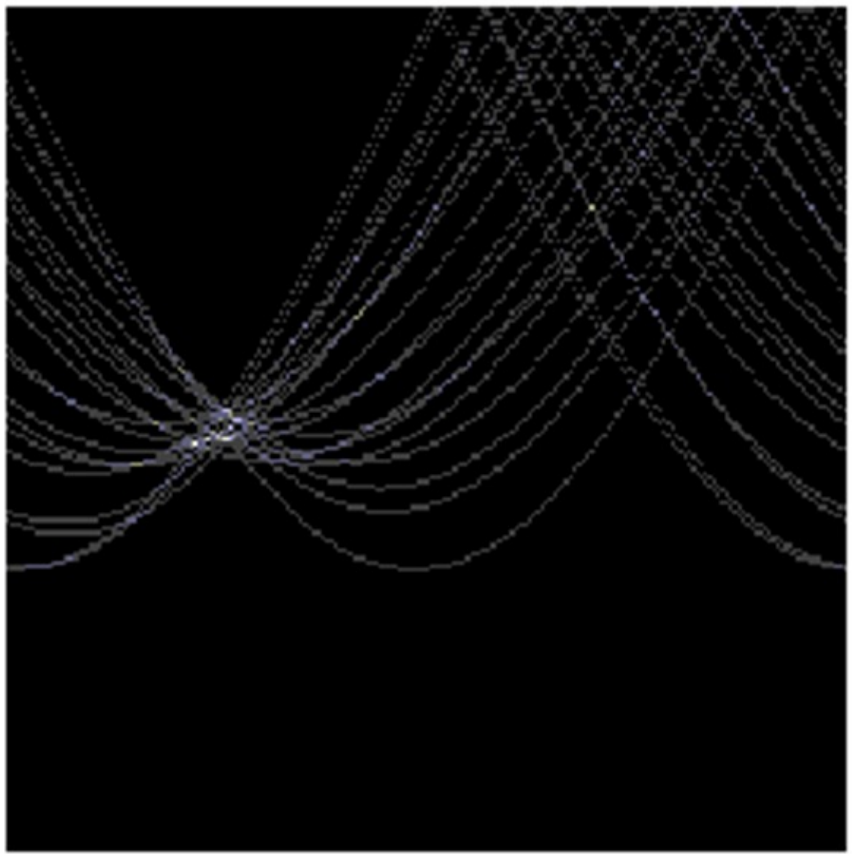
UNIVERSITÀ DI PARMA



Hough Transform

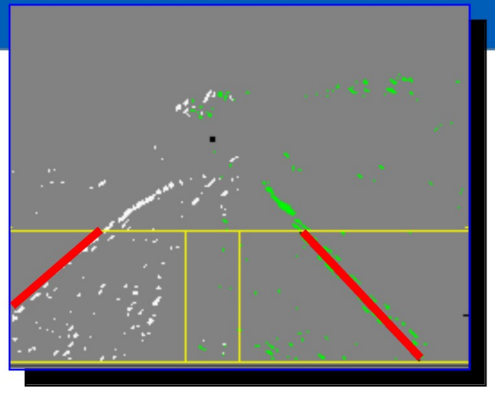
UNIVERSITÀ DI PARMA

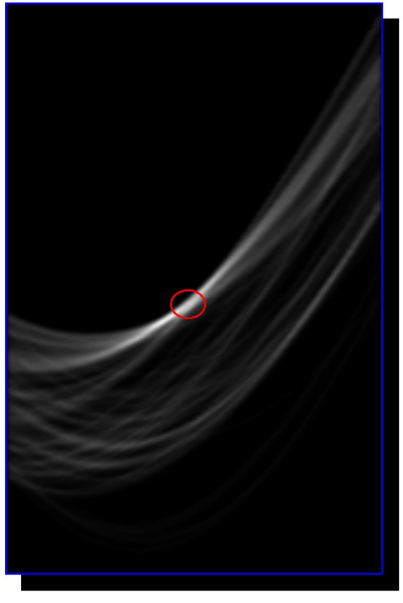


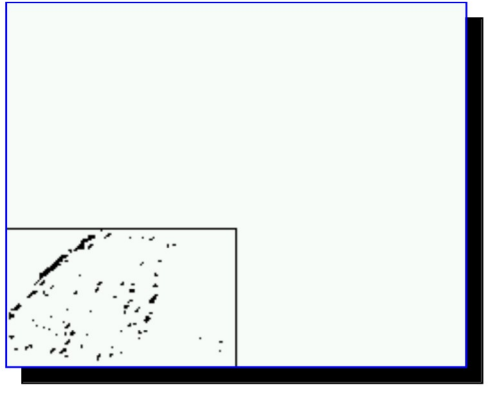


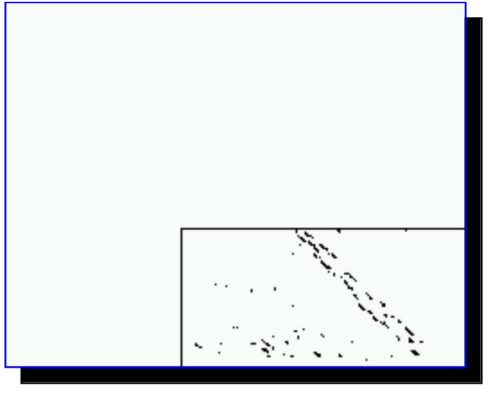
Hough Transform

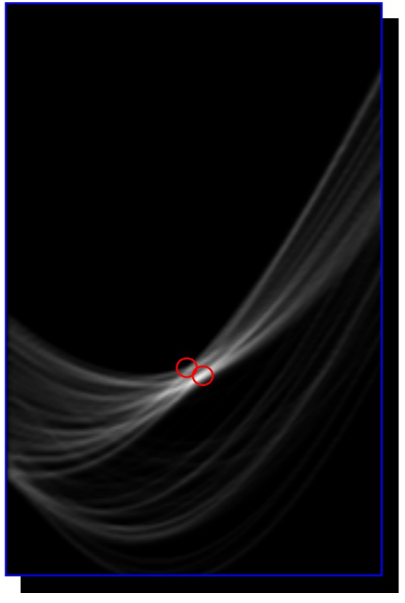
UNIVERSITÀ DI PARMA











Generalized Hough Transform

UNIVERSITÀ DI PARMA

- Described approach is for straight lines
- Anyway it can be applied to any analytical curve
- The only differences are the number and ranges of parameters
  - i.e. for a circle $(x-c_x)^2+(y-c_y)^2=r^2$
  - We need to use a 3D accumulator
  - Similar approach ($O(n^2)$)