上海科技大学
ShanghaiTech University

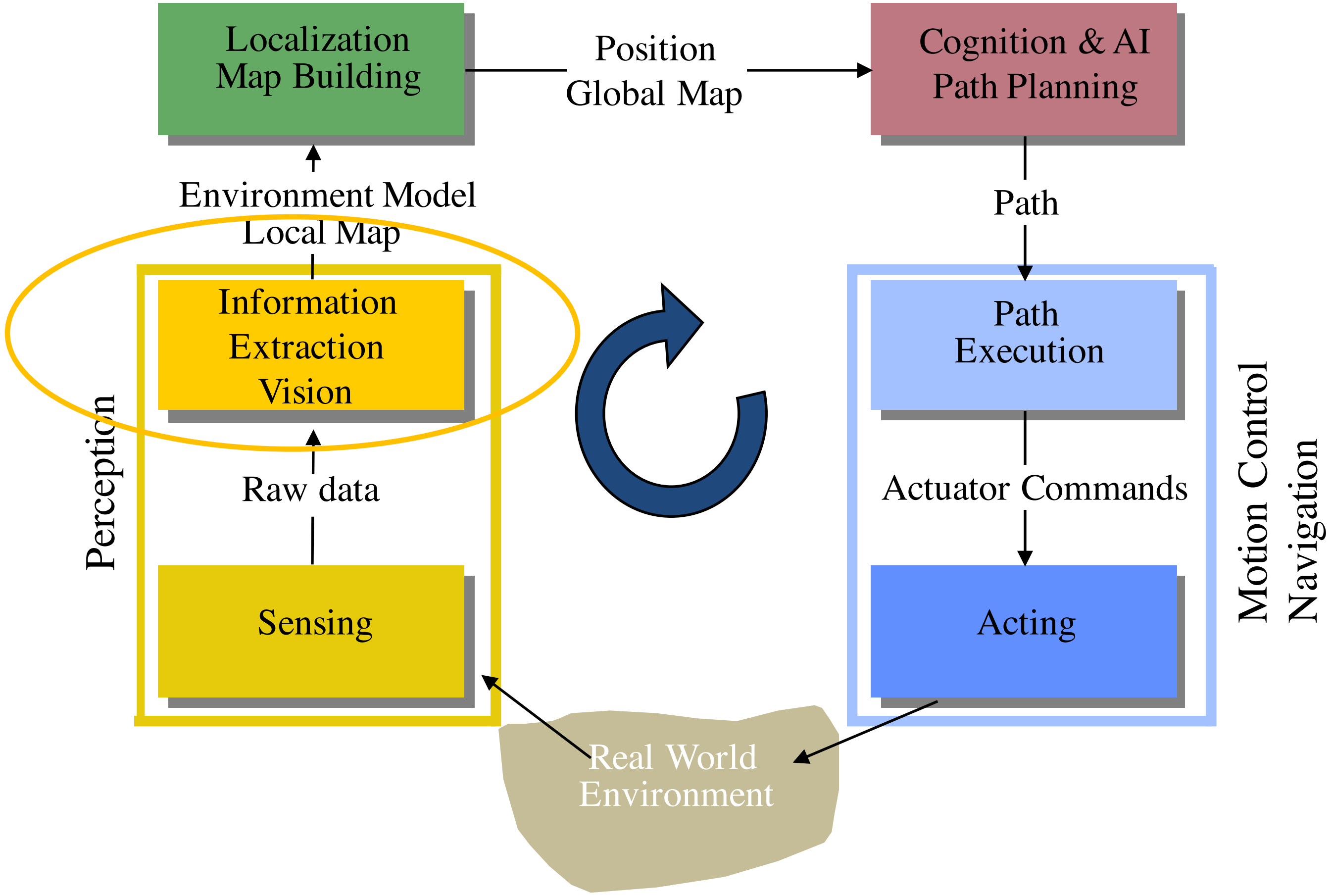
Introduction to Information Science and Technology (IST)
Part IV: Intelligent Machines and Robotics
Vision

Sören Schwertfeger / 师泽仁

ShanghaiTech University

General Control Scheme for Mobile Robot Systems

Lecture 5: Vision

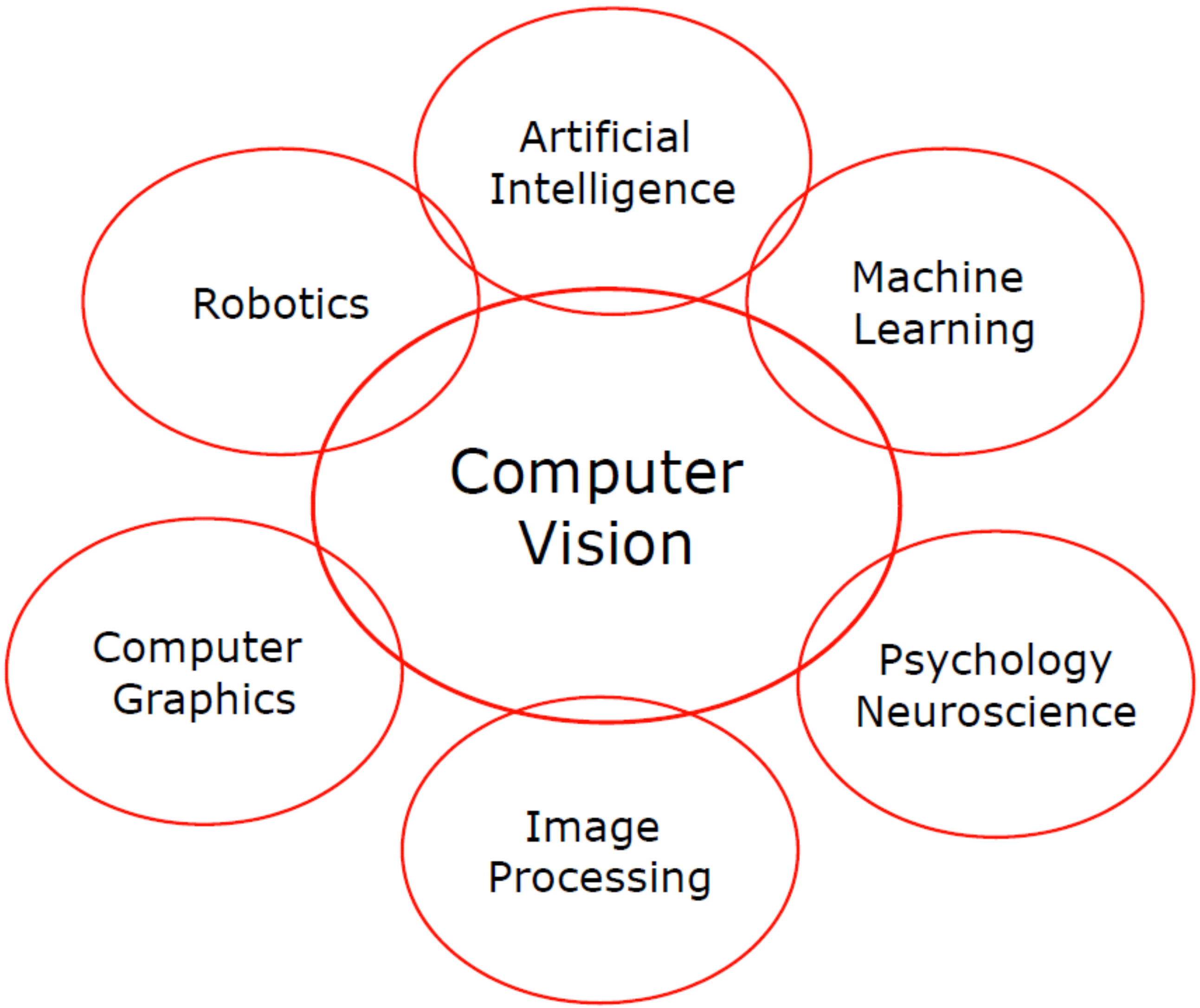


With material from Roland Siegwart and Davide Scaramuzza, ETH Zurich

COMPUTER VISION

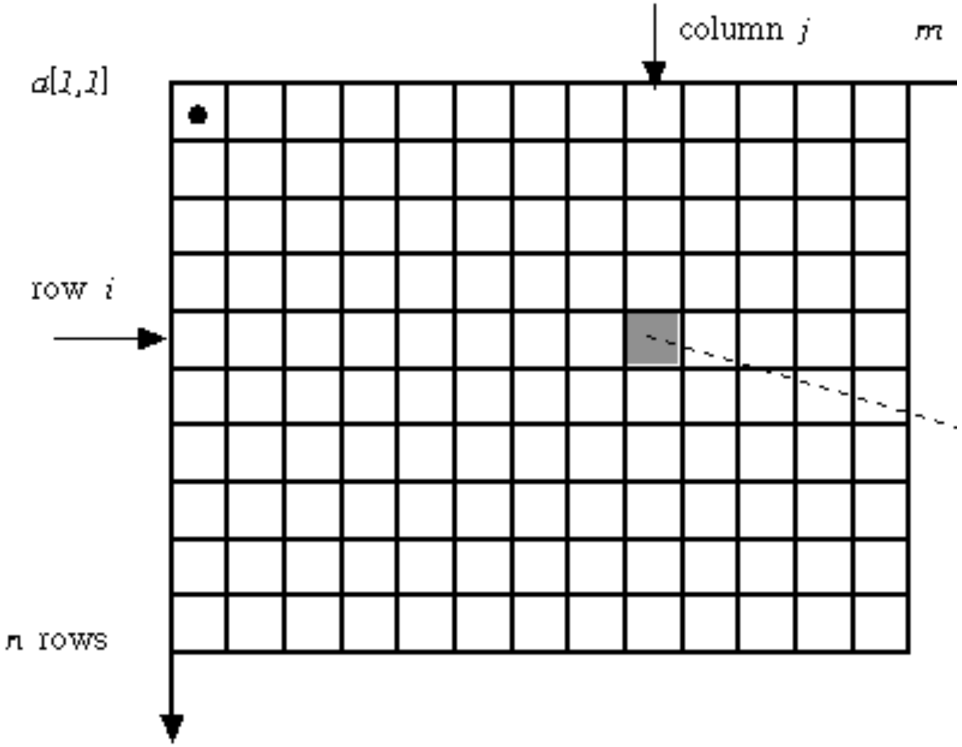
Camera Model

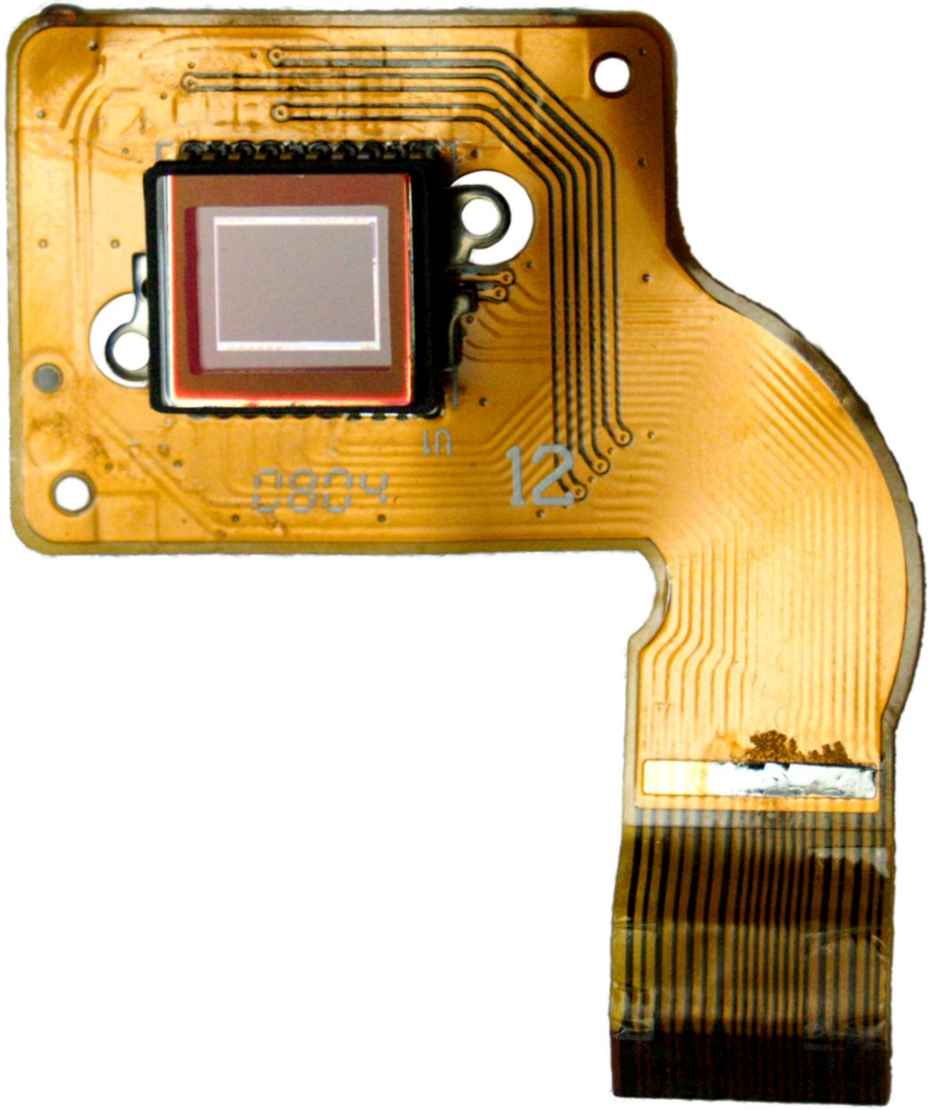
Connection to other disciplines



Digital Image

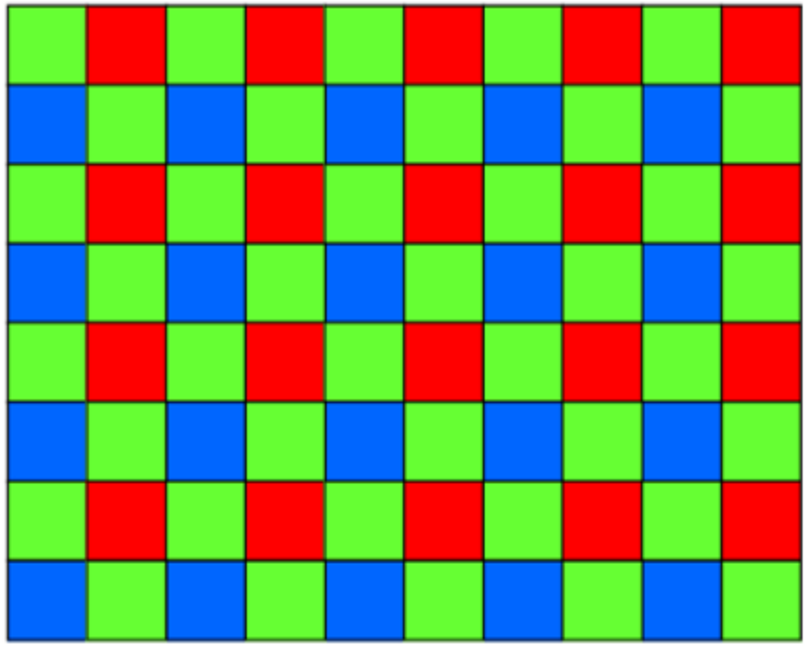
- Image : a two-dimensional array of pixels
- The indices [i, j] of pixels : integer values that specify the rows and columns in pixel values

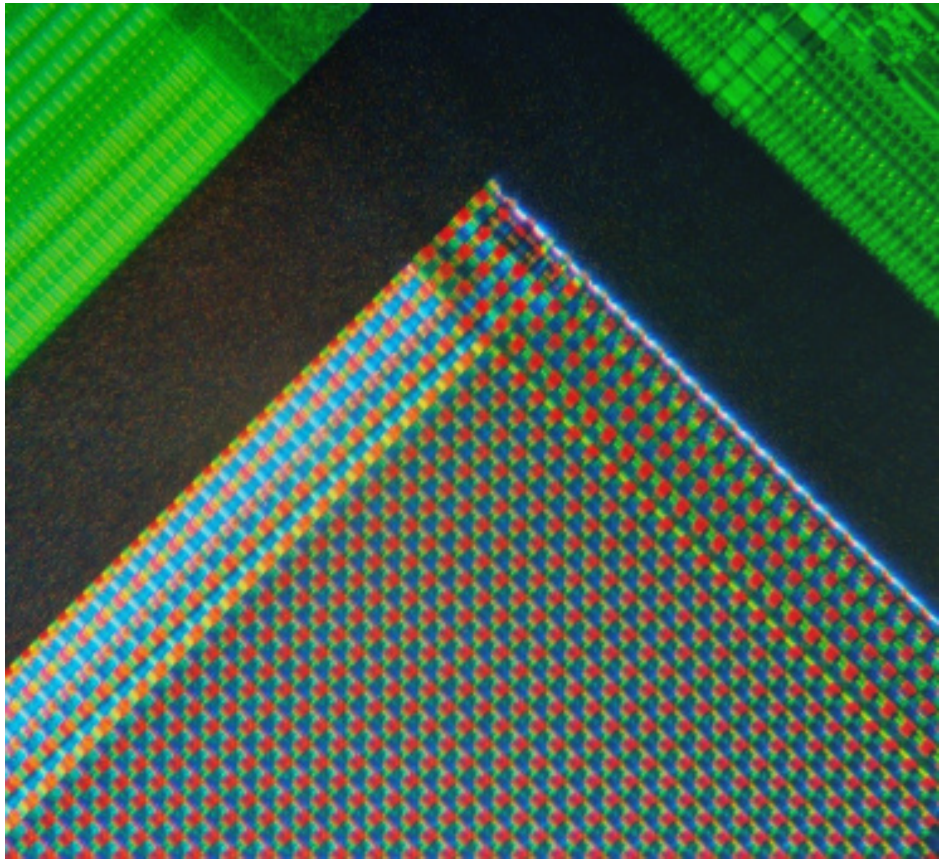




Digital Color Camera

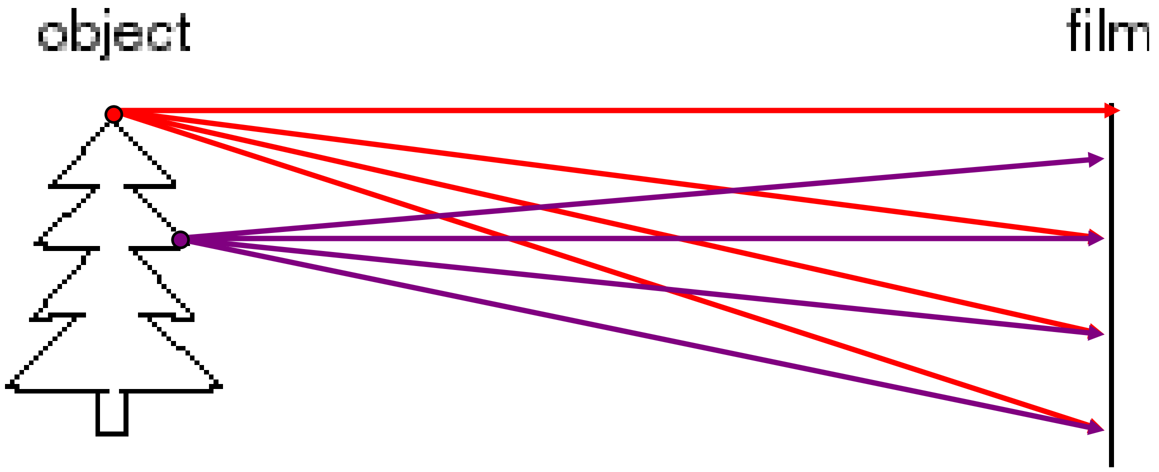
- Bayer Pattern:
  - 50% green, 25% red and 25% blue =>
  - RGBG or GRGB or RGGB.
  - 1 Byte per square
  - 4 squared per 1 pixel
  - More green: eyes are more sensitive to green (nature!)





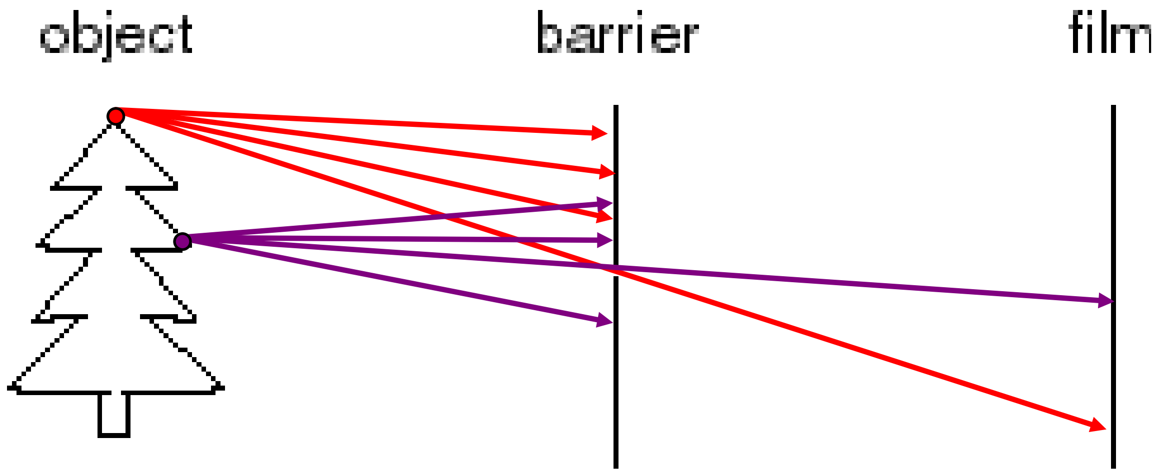
A micrograph of the corner of the photosensor array of a ‘webcam’ digital camera. (Wikimedia)

How do we see the world?



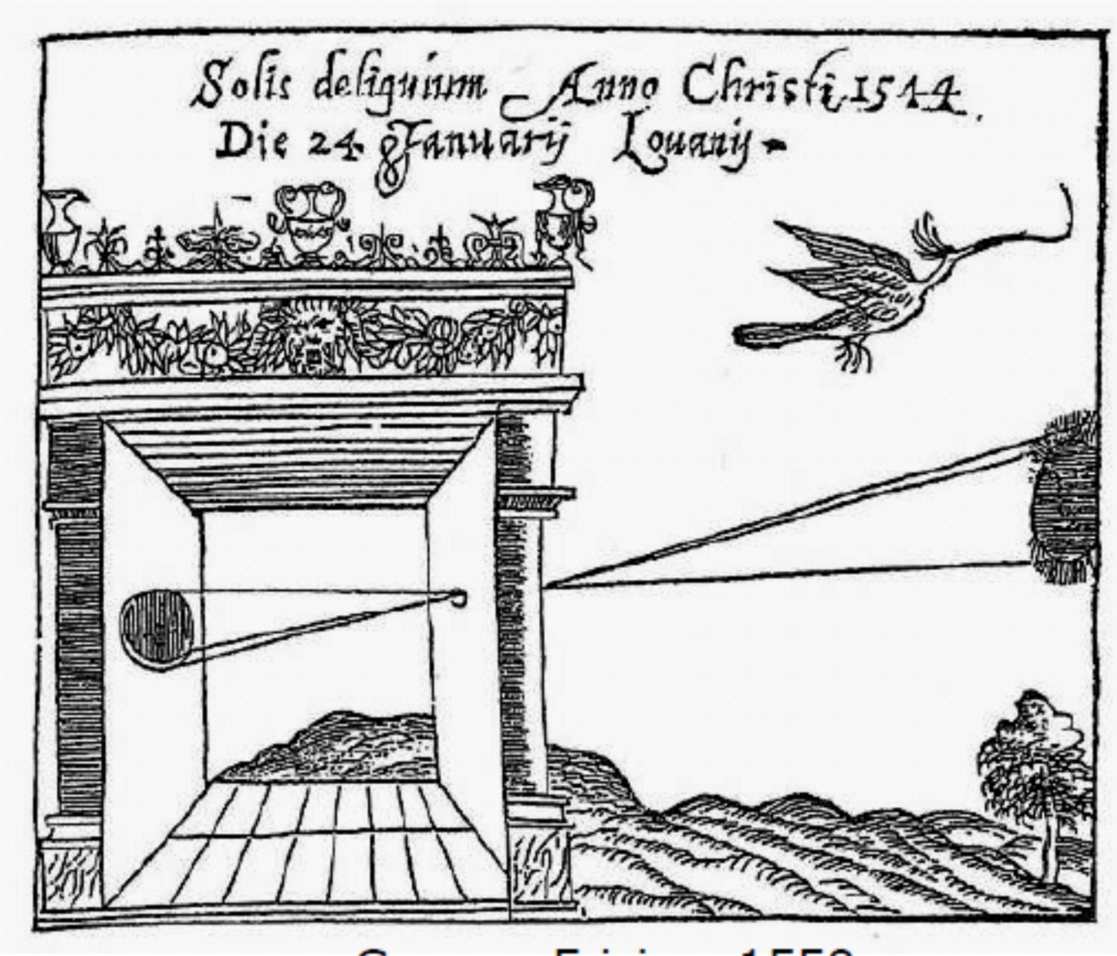
- Let’s design a camera
  - Idea 1: put a piece of film in front of an object
  - Do we get a reasonable image?

Pinhole camera



- Add a barrier to block off most of the rays
  - This reduces blurring
  - The opening known as the **aperture**

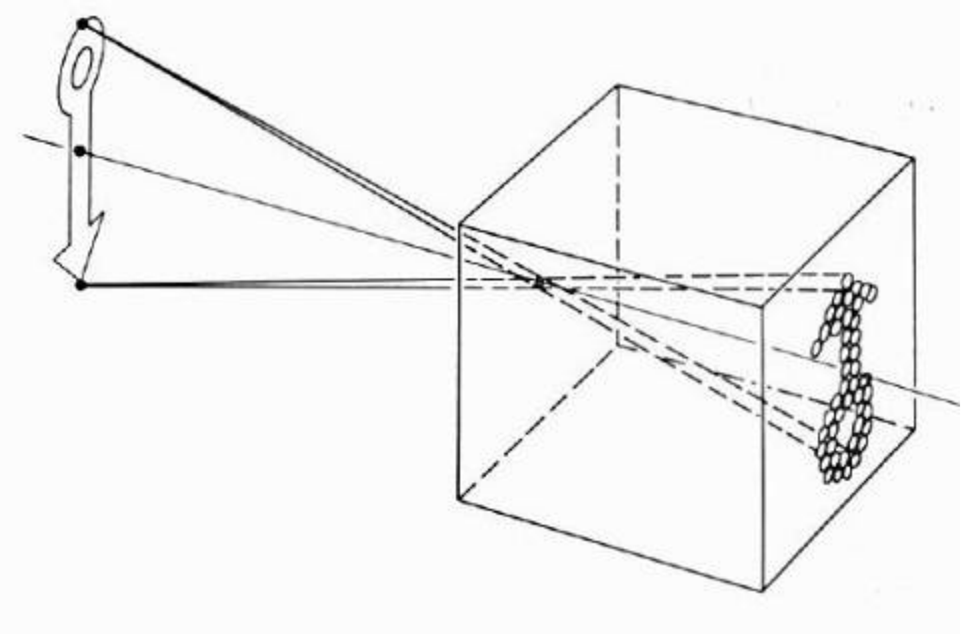
Camera obscura



Gemma Frisius, 1558

- Basic principle known to Mozi (470-390 BC), Aristotle (384-322 BC)
- Drawing aid for artists: described by Leonardo da Vinci (1452-1519)
- Depth of the room (box) is the effective focal length

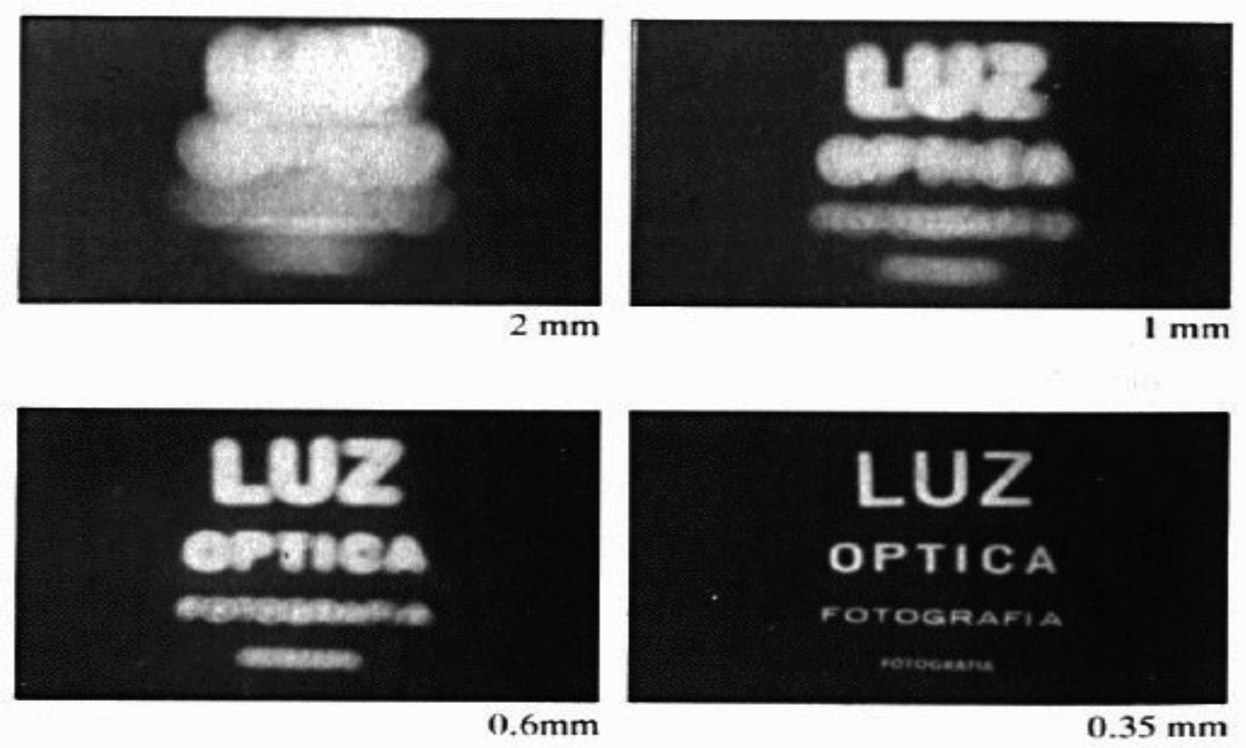
Pinhole camera model



- Pinhole model:
  - Captures **pencil of rays** – all rays through a single point
  - The point is called **Center of Projection**
  - The image is formed on the **Image Plane**

Slide by Steve Seitz

Shrinking the aperture (size of hole)

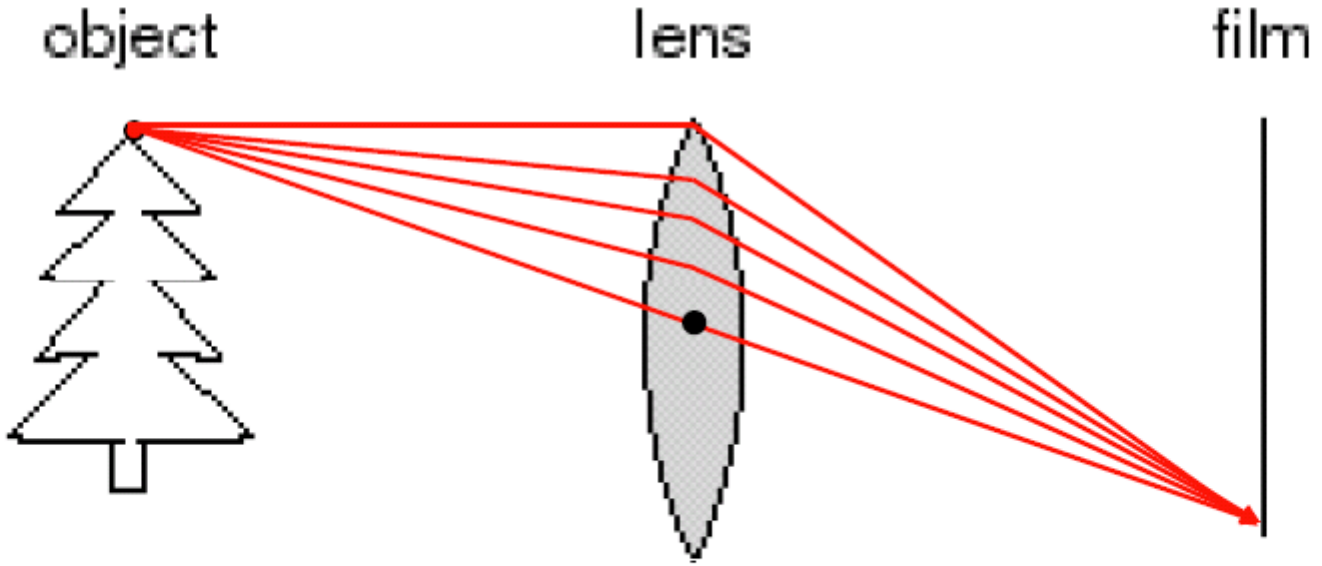


- Why not make the aperture as small as possible?
  - Less light gets through (must increase the exposure)
  - Diffraction effects…

Slide by Steve Seitz

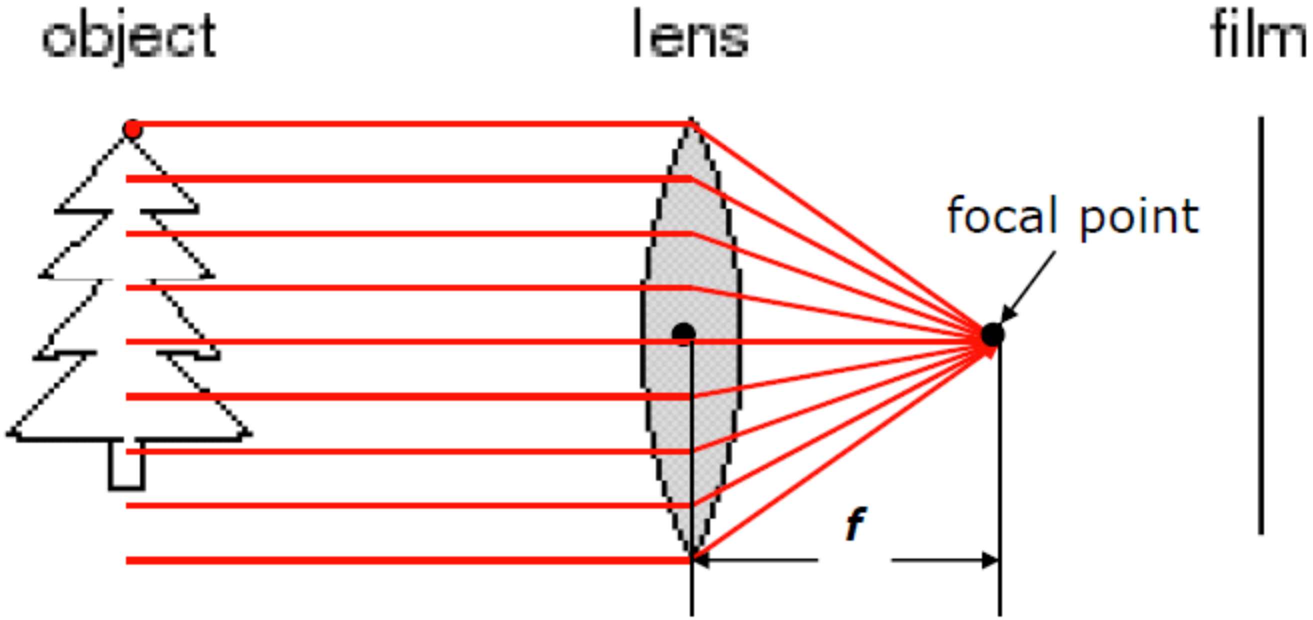
Shrinking the aperture (size of hole)

Solution: adding a lens



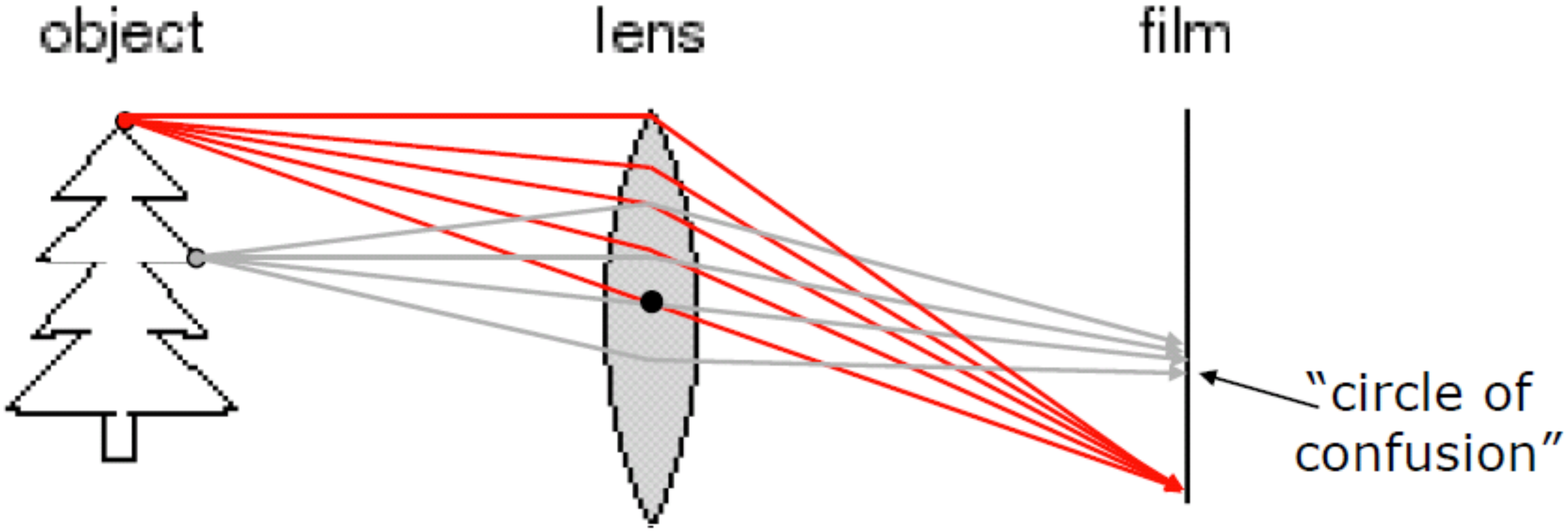
- A lens focuses light onto the film
  - Rays passing through the center are not deviated

Solution: adding a lens



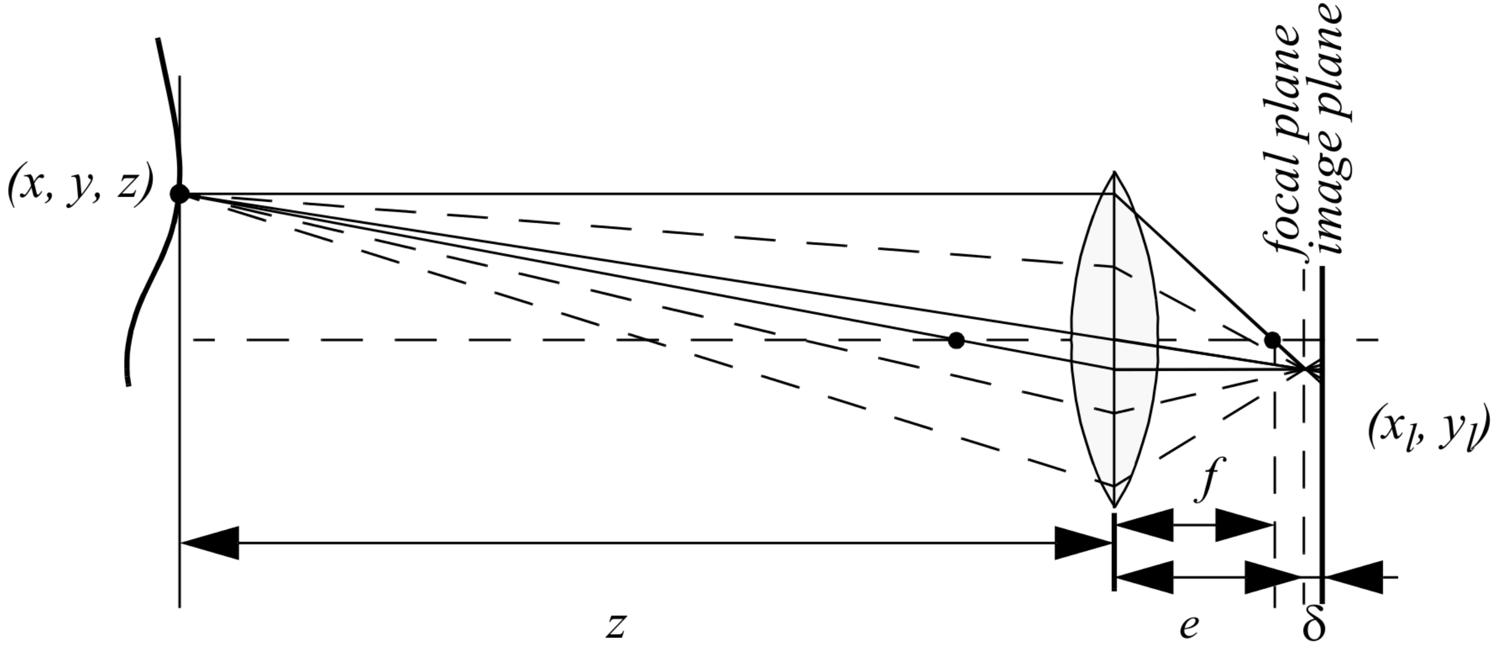
- A lens focuses light onto the film
  - Rays passing through the center are not deviated
  - All parallel rays converge to one point on a plane located at the *focal length f*

Solution: adding a lens



- A lens focuses light onto the film
  - There is a specific distance at which objects are "in focus"
    - other points project to a "circle of confusion" in the image

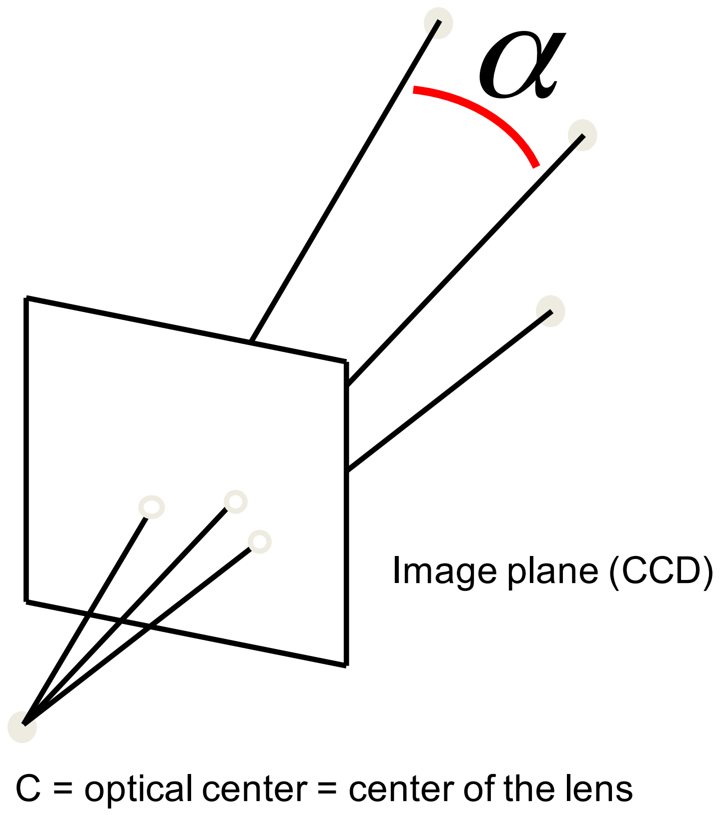
Thin lenses



- Thin lens equation: $$\frac{1}{z} + \frac{1}{e} = \frac{1}{f}$$
  - Any object point satisfying this equation is in focus
  - This formula can also be used to estimate roughly the distance to the object (“Depth from Focus”)

Pin-hole Model

Perspective camera



- For convenience, the image plane is usually represented in front so that the image preserves the same orientation (i.e. not flipped)
- **Notice: a camera does not measure distances but angles! Therefore it is a "bearing sensor"**

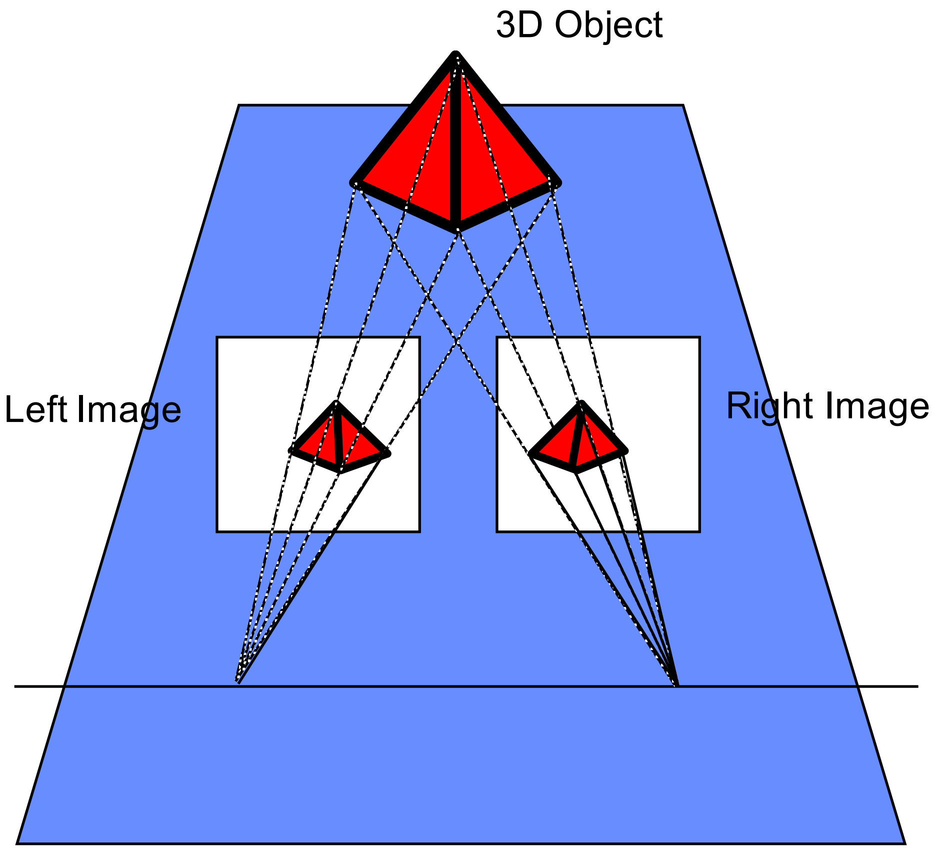
How do we measure distances with cameras?

- Structure from stereo (Stereo-vision):
  - ➢use two cameras with known relative position and orientation

- Structure from motion:
  - ➢use a single moving camera: both 3D structure and camera motion can be estimated up to a scale

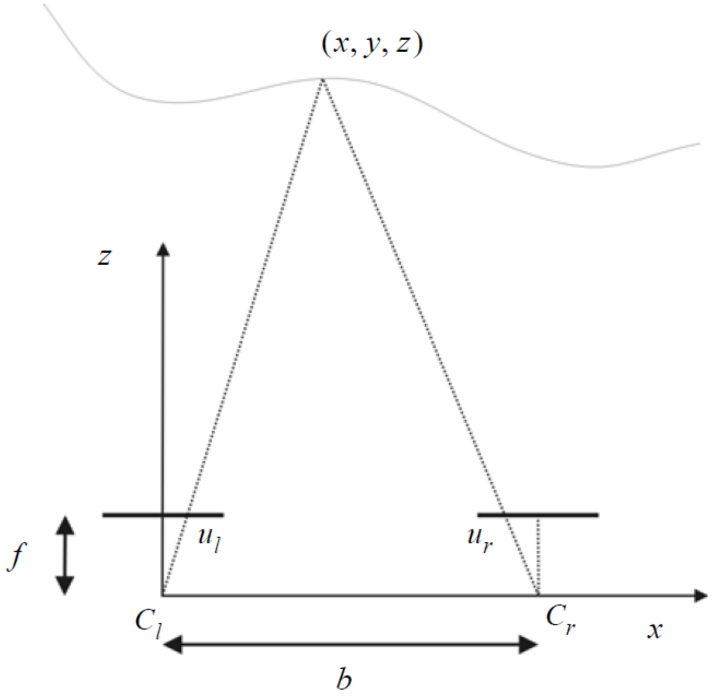
Stereo Vision

- Allows to reconstruct a 3D object from two images taken at different locations



Stereo Vision - The simplified case

- The simplified case is an ideal case. It assumes that both cameras are identical and are aligned on a horizontal axis



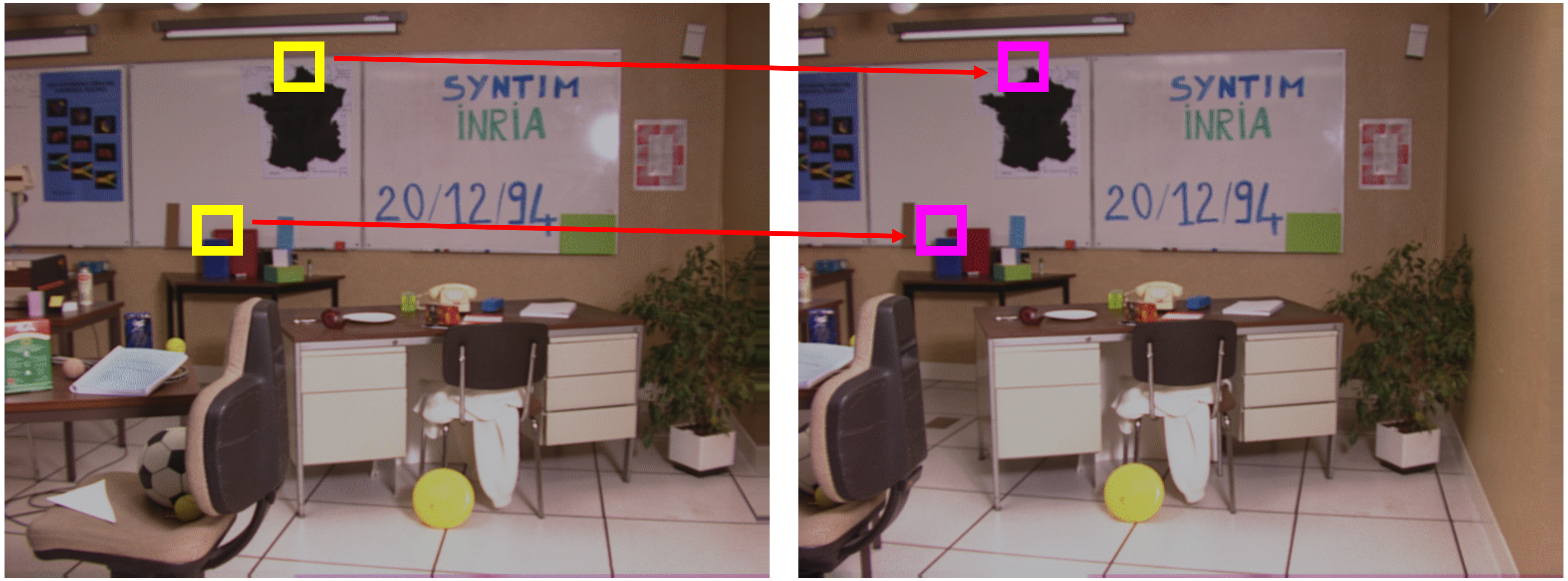
$$\frac{f}{z} = \frac{u_l}{x}, \qquad \frac{f}{z} = \frac{u_r}{b-x} \quad \Rightarrow \quad z = b\frac{f}{u_l - u_r}$$

Distance

- ***b*** = baseline, distance between the optical centers of the two cameras
- ***f*** = focal length
- $\boldsymbol{u_l}$-$\boldsymbol{u_r}$ = disparity

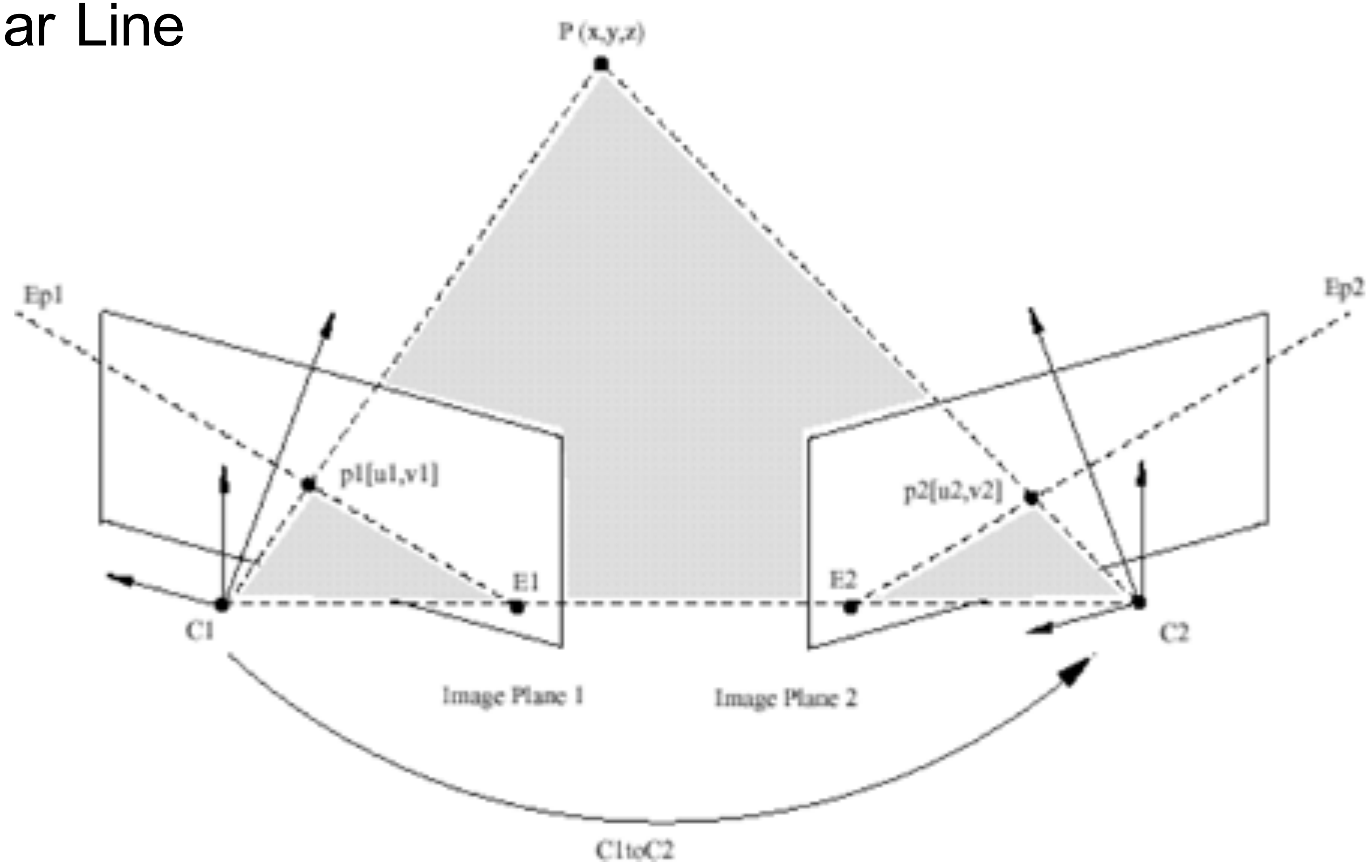
Stereo Vision: Correspondence Problem

- Matching between points in two images that are projection of same 3D real point
- Correspondence search:
  - Compare this point to all other points in other image?
  - Computationally very expensive!
  - => make correspondence search 1 dimensional…



Correspondence Problem: Epipolar Constraint

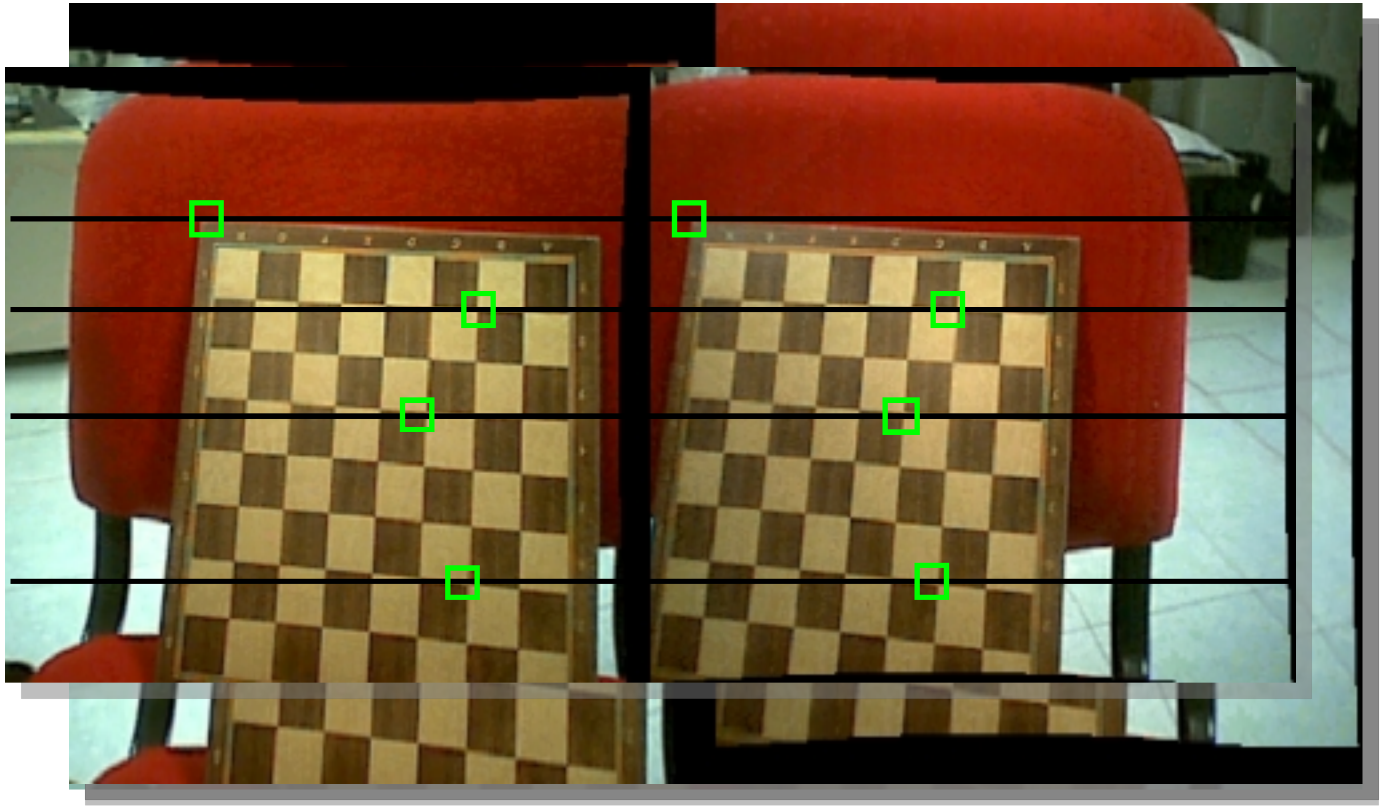
- The correspondent of a point in an image must lie on a line in the other image, called Epipolar Line



Epipolar Rectification

- Determines a transformation of each image plane so that pairs of conjugate epipolar lines become collinear and parallel to one of the image axes (usually the horizontal one)

Rotation
⇩
Focal lengths
⇩
Lens Distortion
⇩
Translation



Stereo Vision Output 1 – Disparity map

- Find the correspondent points of all image pixels of the original images
- For each pair of conjugate points compute the disparity ***d* = *v*-*v'***
- ***d(x,y)*** is called Disparity map.

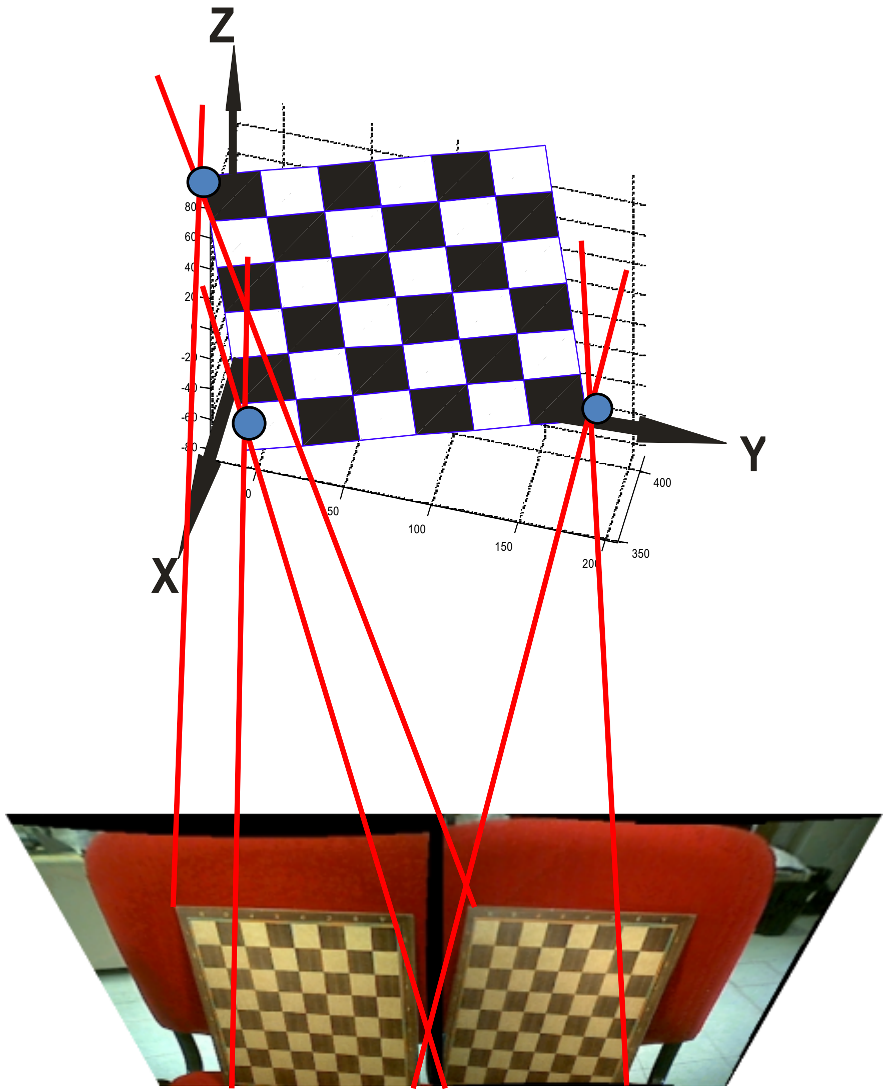
Left image

Right image

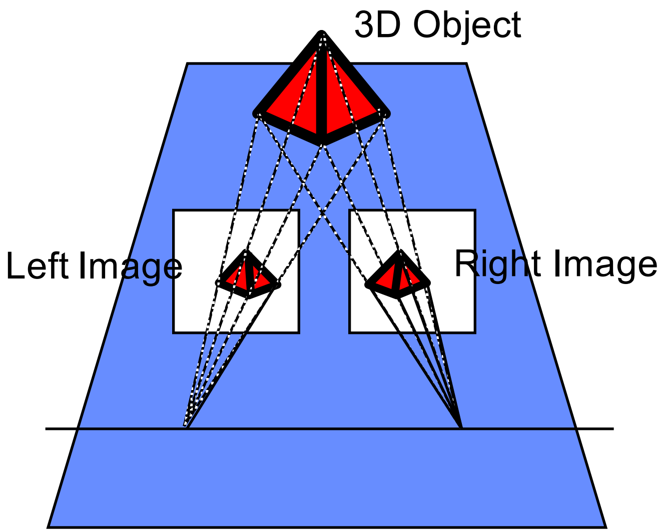
Disparity map

- Disparity maps are usually visualized as grey-scale images. Objects that are closer to the camera appear lighter, those who are further appear darker.

Stereo Vision Output 2 - 3D Reconstruction via triangulation



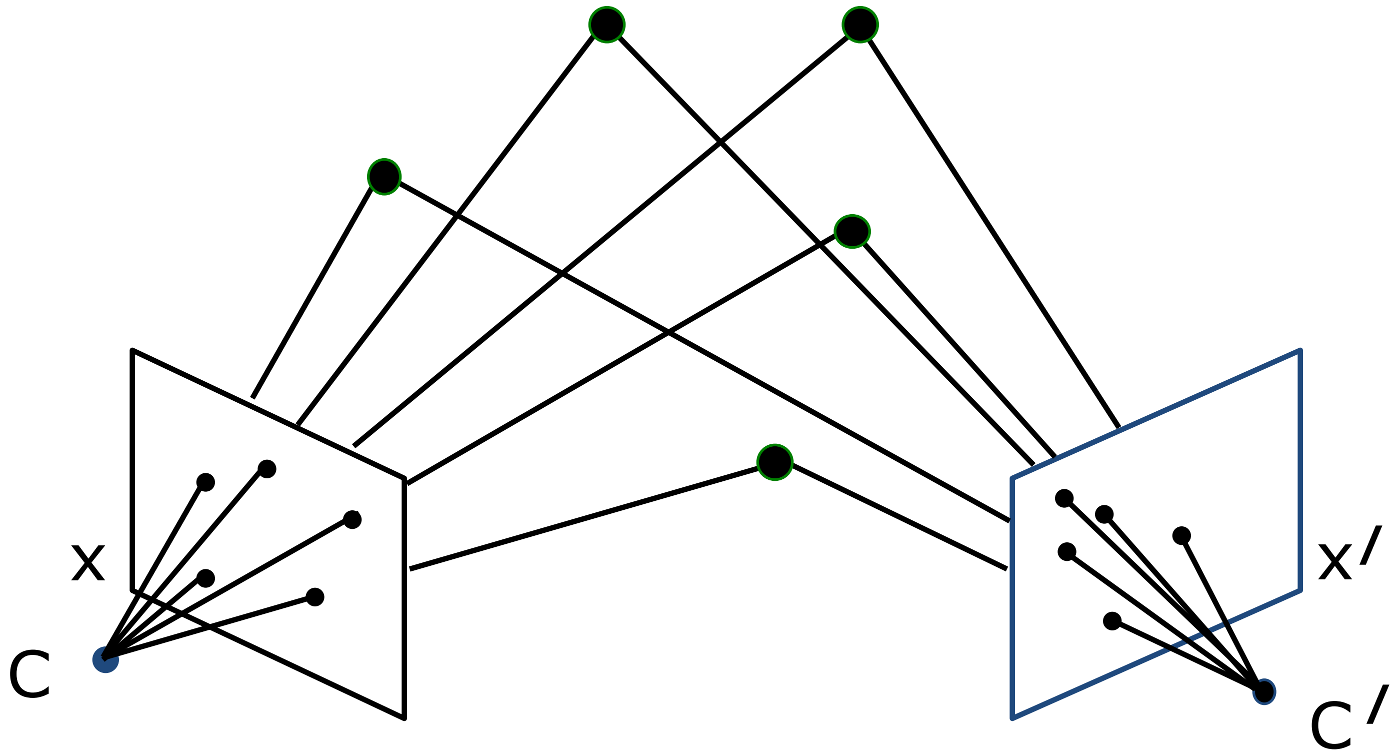
Stereo Vision - summary



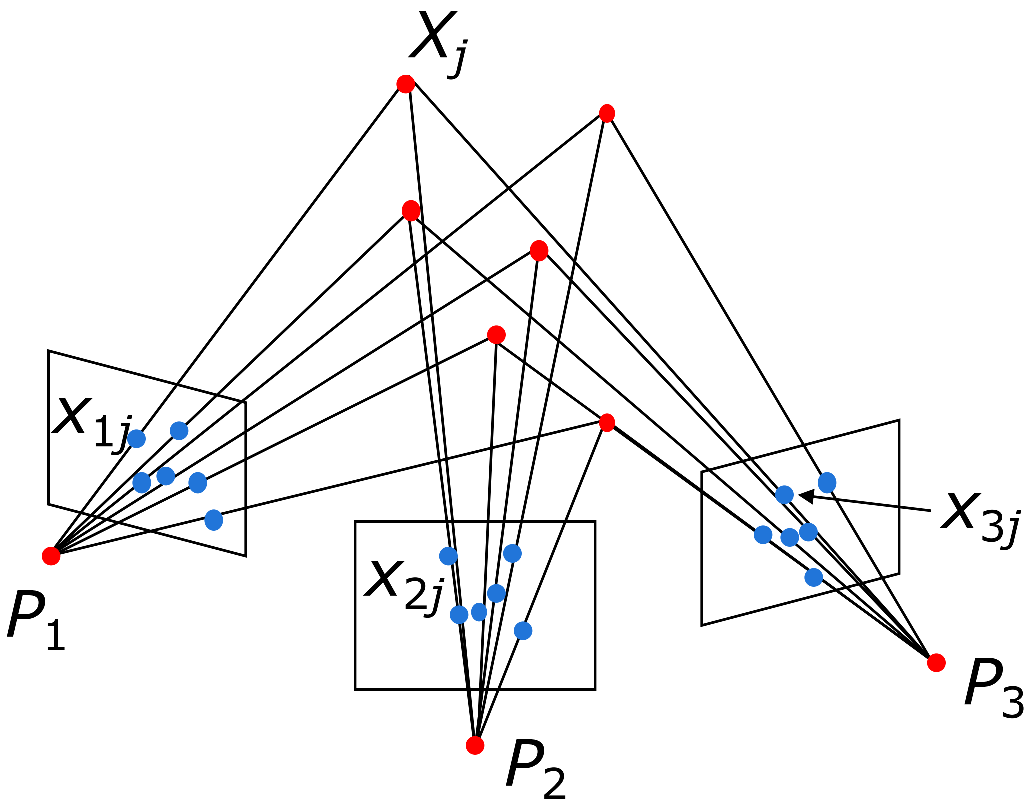
1. Stereo camera calibration -> compute camera relative pose
2. Epipolar rectification -> align images
3. Search correspondences
4. Output: compute stereo triangulation or disparity map
5. Consider baseline and image resolution to compute accuracy!

Structure from motion

- Given image point correspondences, $\mathbf{x}_i \leftrightarrow \mathbf{x}_i^{\prime}$, determine R and T
- Rotate and translate camera until stars of rays intersect
- At least 5 point correspondences are needed

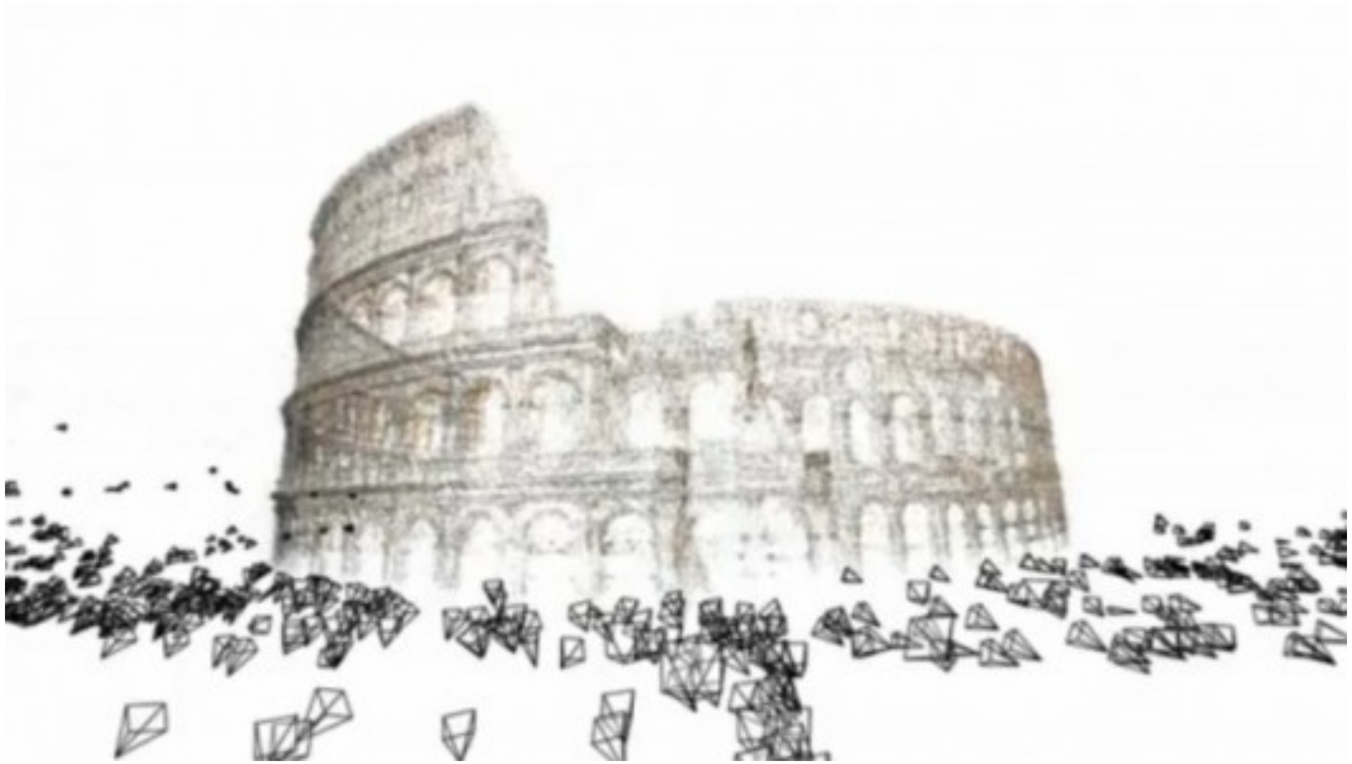


Multiple-view structure from motion



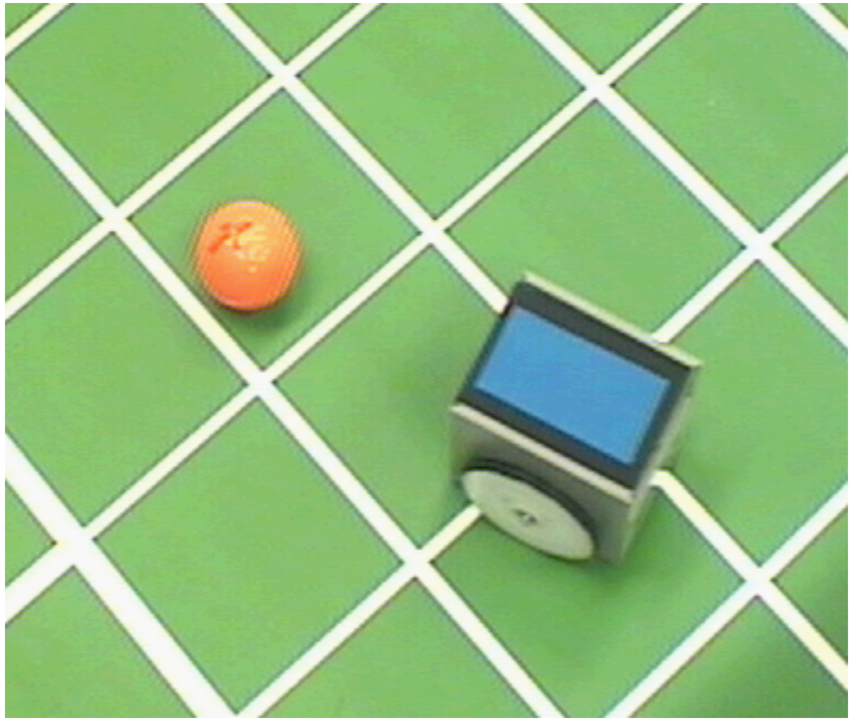
Multiple-view structure from motion

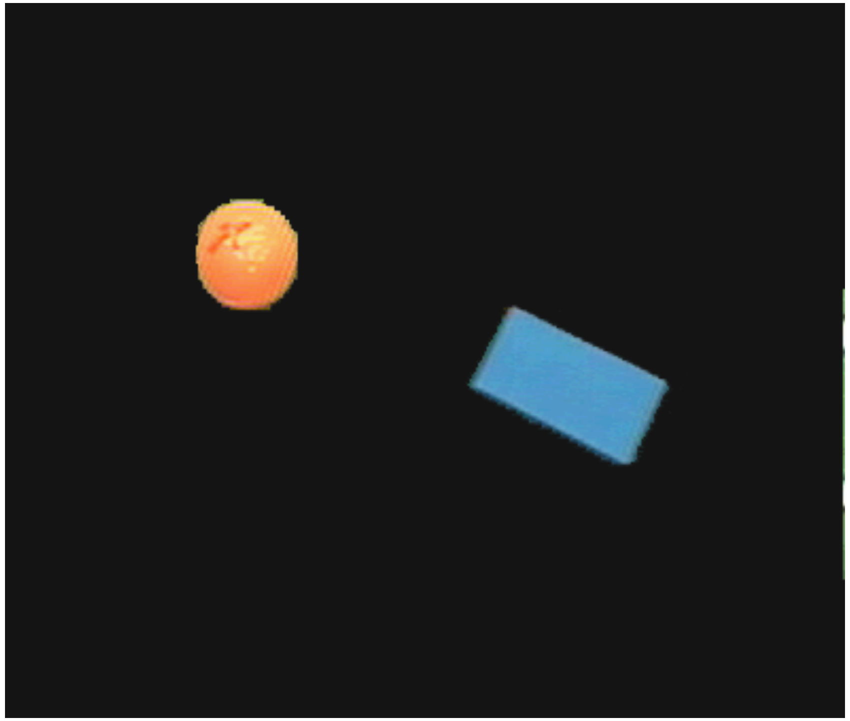
- Results of Structure from motion from 2 million user images from flickr.com



Color Tracking

- Motion estimation of ball and robot for soccer playing using color tracking





Color segmentation with fixed thesholds

- Simple: constant thresholding:
  - selection only **iff** RGB values (r,g,b) simultaneously in R, G, and B ranges:
  - six thresholds [Rmin,Rmax], [Gmin,Gmax], [Bmin,Bmax]:

$$R_{min} < r < R_{max} \text{ and } G_{min} < g < G_{max} \text{ and } B_{min} < b < B_{max}$$

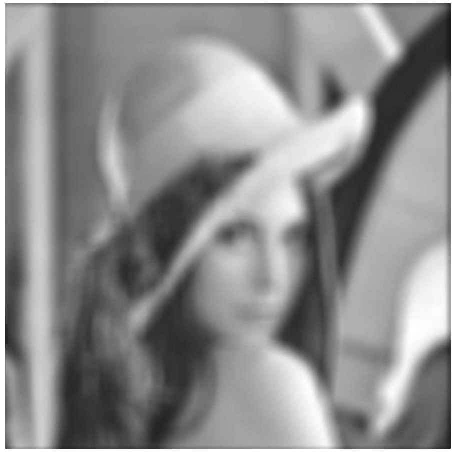
- Alternative: YUV color space
  - RGB values encode intensity of each color
  - YUV:
  - U and V together color (or chrominance)
  - Y brightness (or luminosity)
  - bounding box in YUV space => greater stability wrt. changes in illumination

VISION

Image Processing

Image filtering

- Filter: frequency domain processing where “filtering” refers to the process of accepting or rejecting certain frequency components. E.g.:
  - Lowpass filer: pass only low frequencies => blur (smooth) an image
  - **spatial filters** (also called masks or kernels): same effect



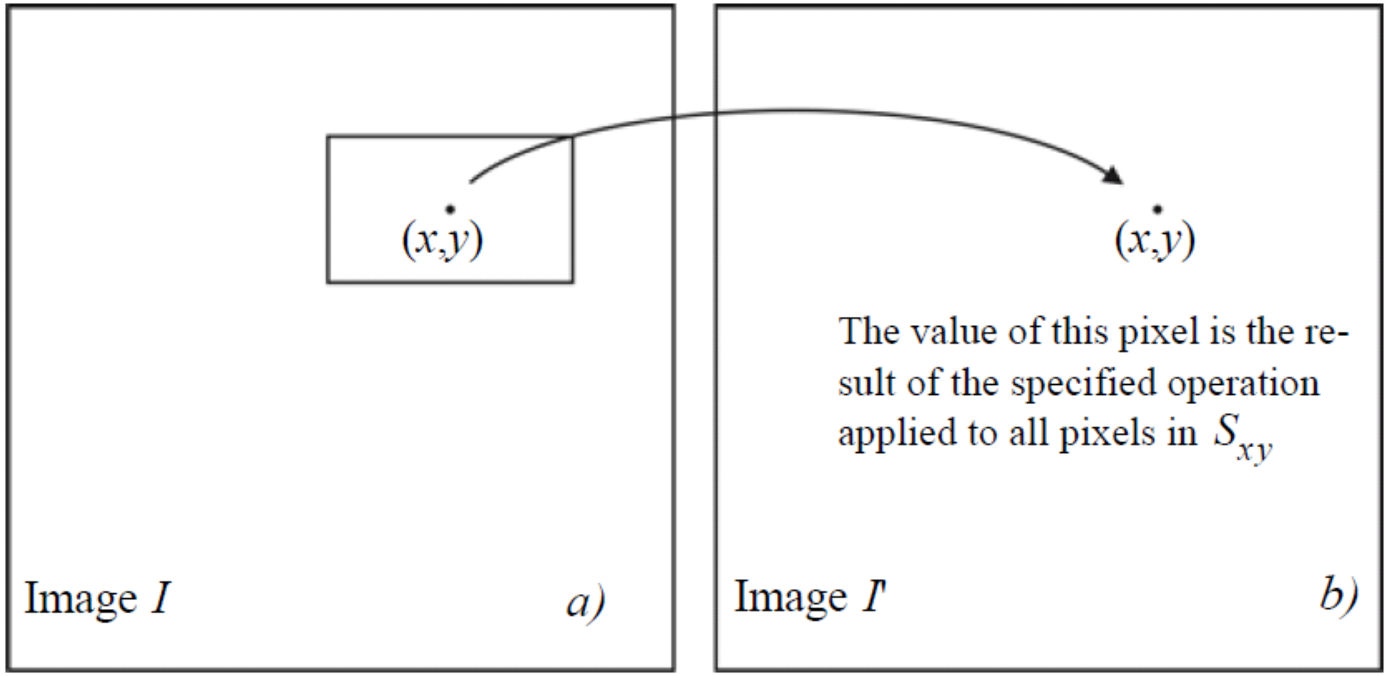
Lowpass filtered image

Highpass filtered image

Lena: Image processing standard test picture (512x512) since 1972

Spatial filters

- Let *Sxy* denote the set of coordinates of a neighborhood centered on an arbitrary point (x,y) in an image *I*
- Spatial filtering generates a corresponding pixel at the same coordinates in an output image *I'* where the value of that pixel is determined by a specified operation on the pixels in *Sxy*



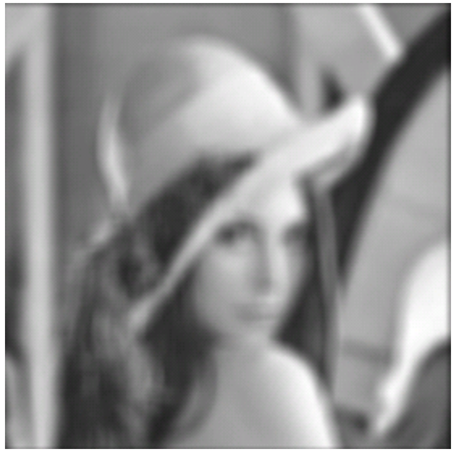
- For example, an averaging filter is: $$I' = \frac{1}{mn} \sum_{(r,c) \in S_{xy}} I(r,c)$$

Smoothing filters (1)

- A constant averaging filter yields the standard average of all the pixels in the mask. For a 3x3 mask this writes:

$$w = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

- all coefficients sum to 1 => normalization
- Normalization important: keep the same value as the original image if region of filter is is uniform



This example was generated with a 21x21 mask

Smoothing filters (2)

- A Gaussian averaging write as

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

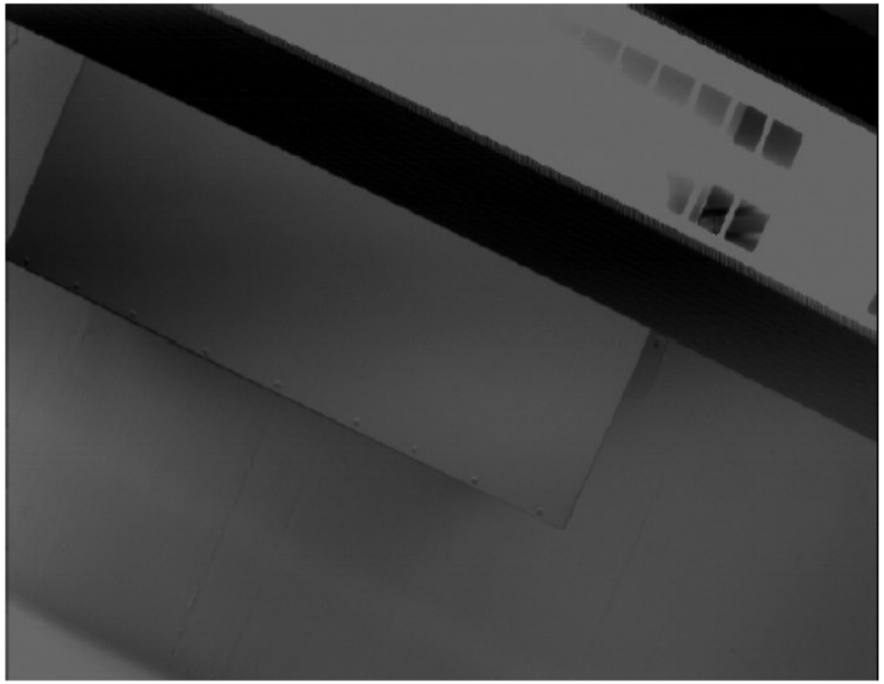
- To generate, say, a 3x3 filter mask from this function, we sample it about its center. For example, with σ=0.85, we get

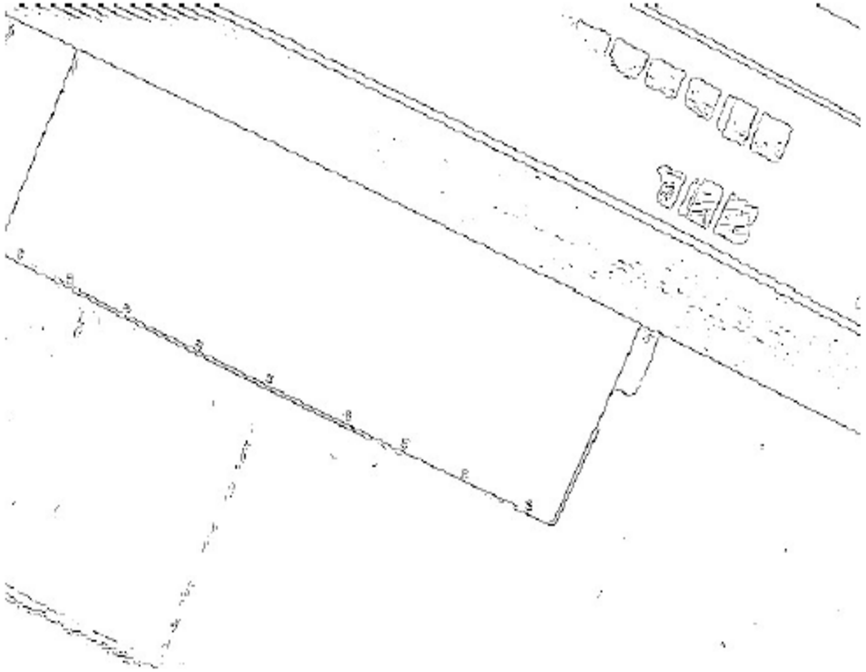
$$G = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- Very popular: Such low-pass filters effectively removes high-frequency noise =>
- Gradients and derivatives very important in image processing =>
- Gaussian smoothing preprocessing popular first step in computer vision algorithms

Edge Detection

- Ultimate goal of edge detection:
  - an idealized line drawing
- Edge contours in the image correspond to important scene contours.





Edge is Where Change Occurs

- Edges correspond to sharp changes of intensity
- Change is measured by 1^{st} derivative in 1D
- Biggest change, derivative has maximum magnitude
- Or 2^{nd} derivative is zero.

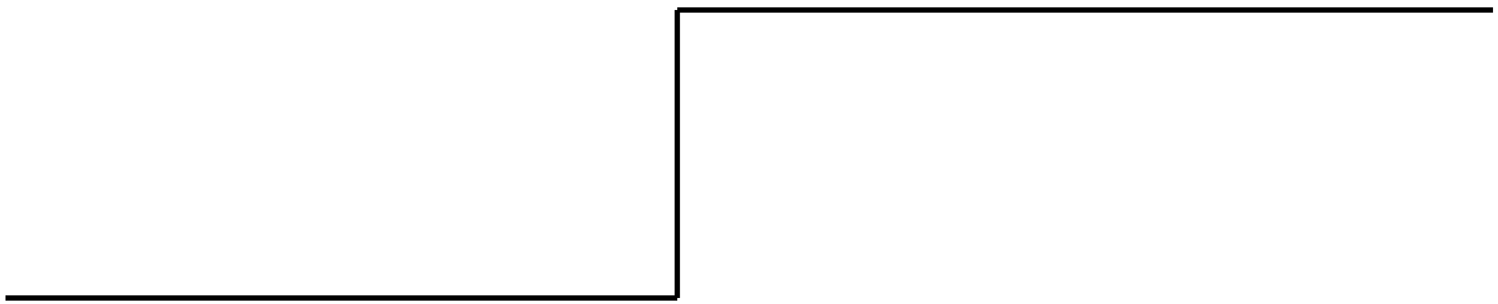
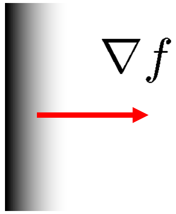


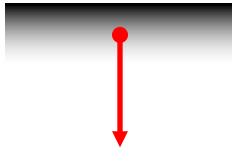
Image gradient

- The gradient of an image: $$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

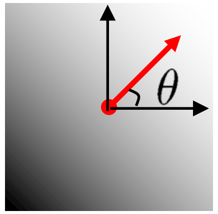
- The gradient points in the direction of most rapid change in intensity



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



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



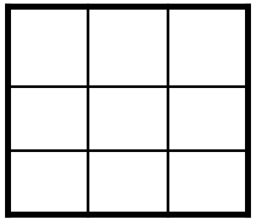
$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

- The gradient direction is: $\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$
- The gradient magnitude is: $\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$

The discrete gradient

- How can we differentiate a *digital* image f[x,y]?
  - Option 1: reconstruct a continuous image, then take gradient
  - Option 2: take discrete derivative (finite difference)

$$\frac{\partial f}{\partial x}[x,y] \approx f[x+1,y] - f[x,y]$$



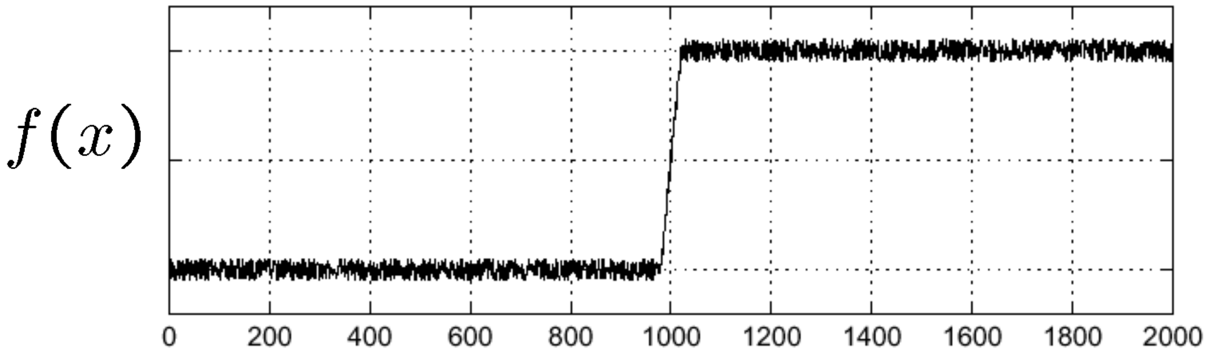
w

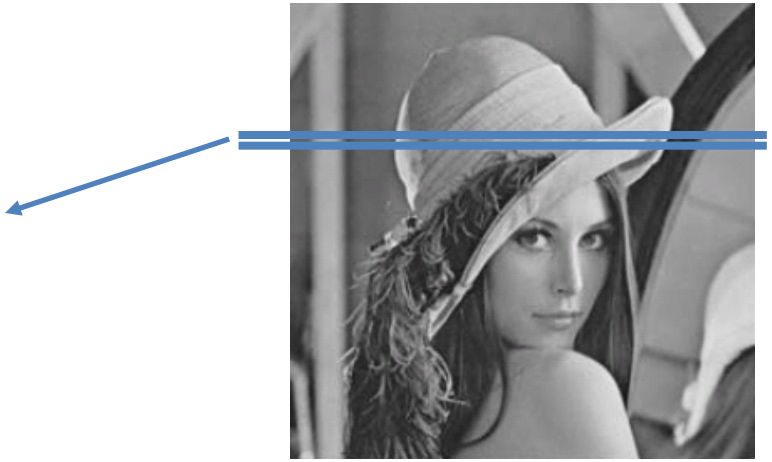
Gradient Edge Detectors

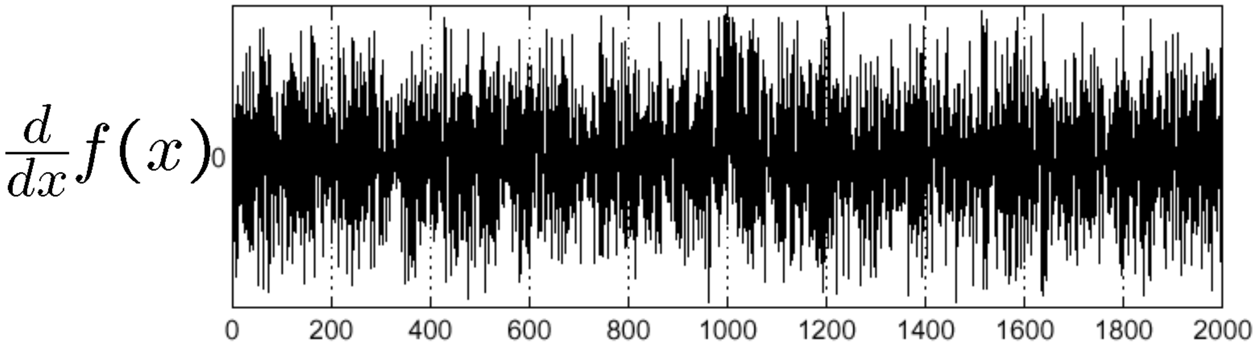
- **Roberts** $|G| \cong \sqrt{r_1^2 + r_2^2}$; $r_1 = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$; $r_2 = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$
- **Prewitt** $|G| \cong \sqrt{p_1^2 + p_2^2}$; $\theta \cong \operatorname{atan}\left(\frac{p_1}{p_2}\right)$; $p_1 = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$; $p_2 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$
- **Sobel** $|G| \cong \sqrt{s_1^2 + s_2^2}$; $\theta \cong \operatorname{atan}\left(\frac{s_1}{s_2}\right)$; $s_1 = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$; $s_2 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$

Effects of noise

- Consider a single row or column of the image
  - Plotting intensity as a function of position gives a signal







- Where is the edge?

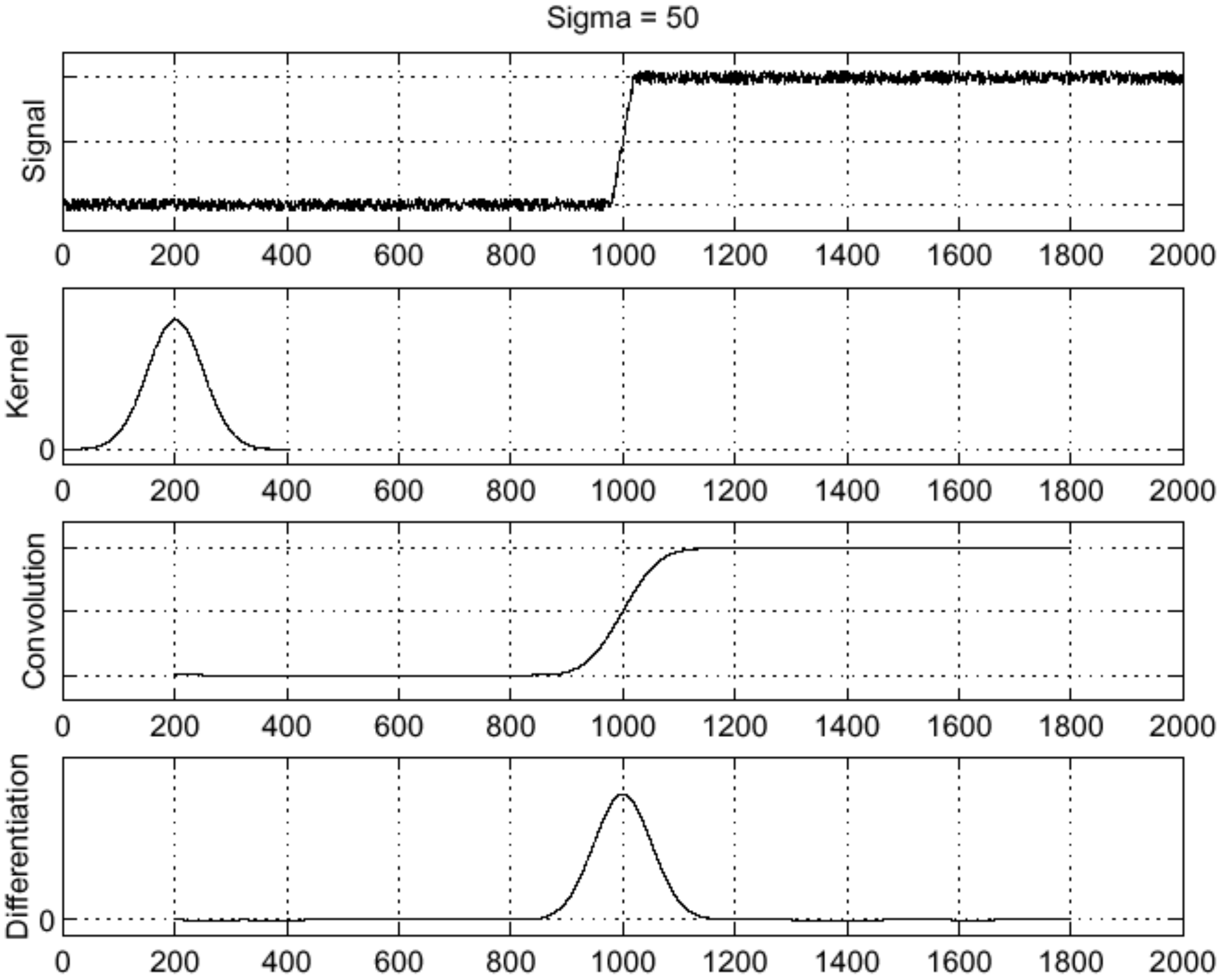
Solution: smooth first

f

h

$h \star f$

$\frac{\partial}{\partial x}(h \star f)$



- Where is the edge?
- Look for peaks in $\frac{\partial}{\partial x}(h \star f)$

Derivative theorem of convolution

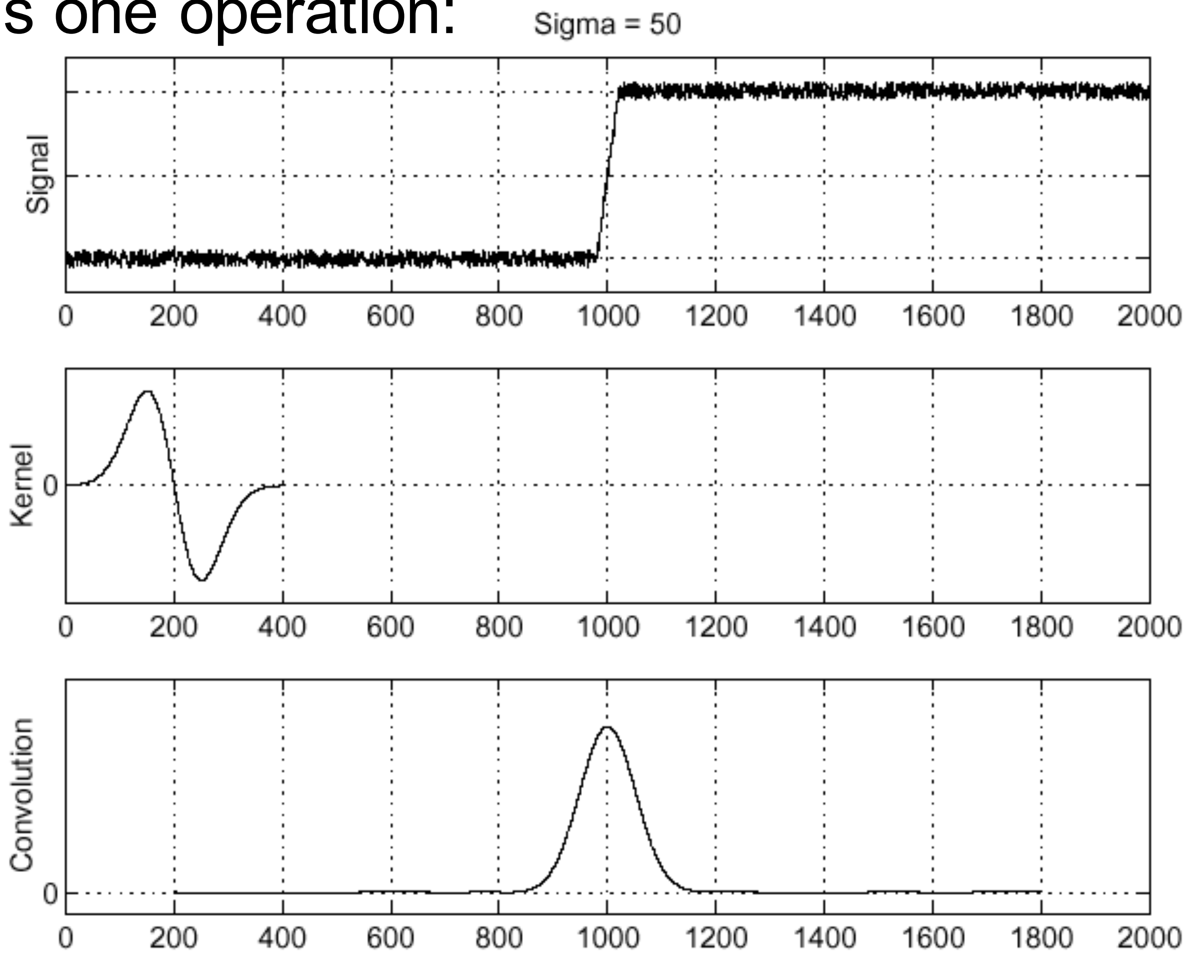
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

- This saves us one operation:

f

$\frac{\partial}{\partial x}h$

$(\frac{\partial}{\partial x}h) \star f$



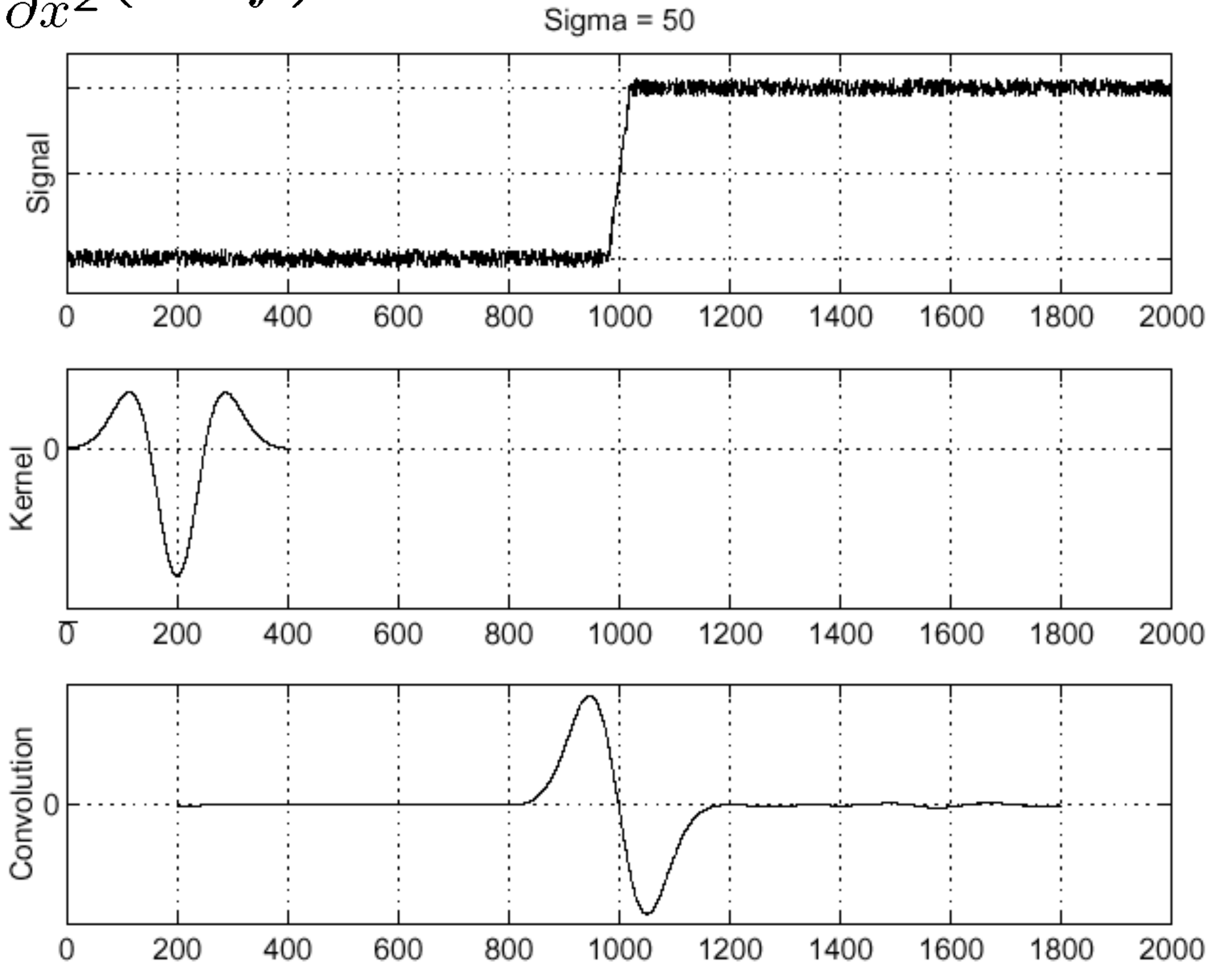
The Canny Edge Detector

- Consider $\frac{\partial^2}{\partial x^2}(h \star f)$

f

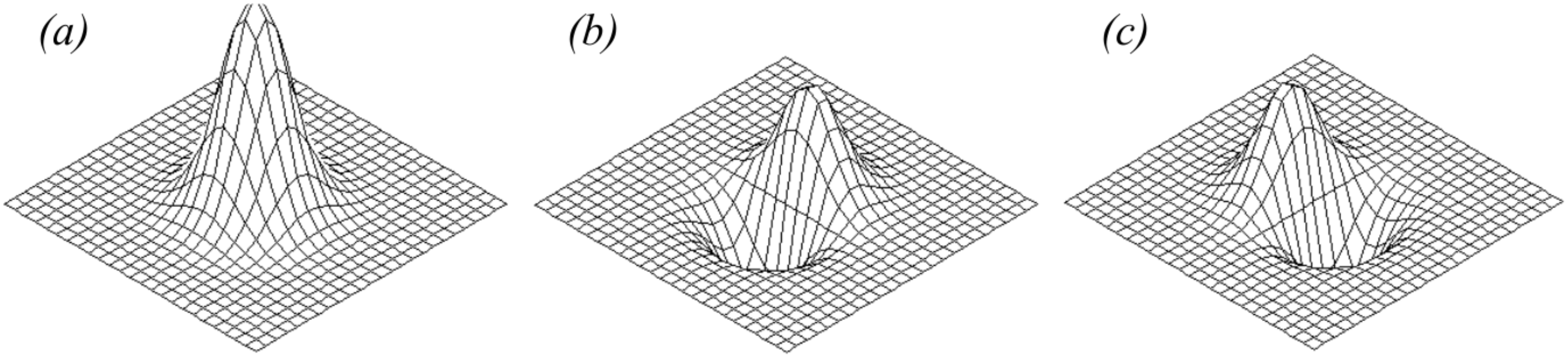
Laplacian of Gaussian operator $\frac{\partial^2}{\partial x^2}h$

$(\frac{\partial^2}{\partial x^2}h) \star f$



- Where is the edge?
- Zero-crossings of bottom graph

2D Canny edge detector



$G_\sigma(x, y) = G_\sigma(x)G_\sigma(y)$ $\quad f_V(x, y) = G'_\sigma(x)G_\sigma(y)$ $\quad f_H(x, y) = G'_\sigma(y)G_\sigma(x)$

- Two perpendicular filters:
  - Convolve imge $I(x,y)$ with $f_V(x,y)$ and $f_H(x,y)$ – obtaining $R_V(x,y)$ and $R_H(x,y)$
  - Use square of gradient magnitude: $R(x,y) = R_V^2(x,y) + R_H^2(x,y)$
  - Mark peaks in $R(x,y)$ above a threshold

The Sobel edge detector

original image (Lena image)

The Sobel edge detector

norm of the gradient

The Sobel edge detector

thresholding

The Sobel edge detector

thinning

(non-maxima suppression)

IMAGE FEATURES

- Lines
- Points
    - Harris
    - SIFT

Example: Build a Panorama

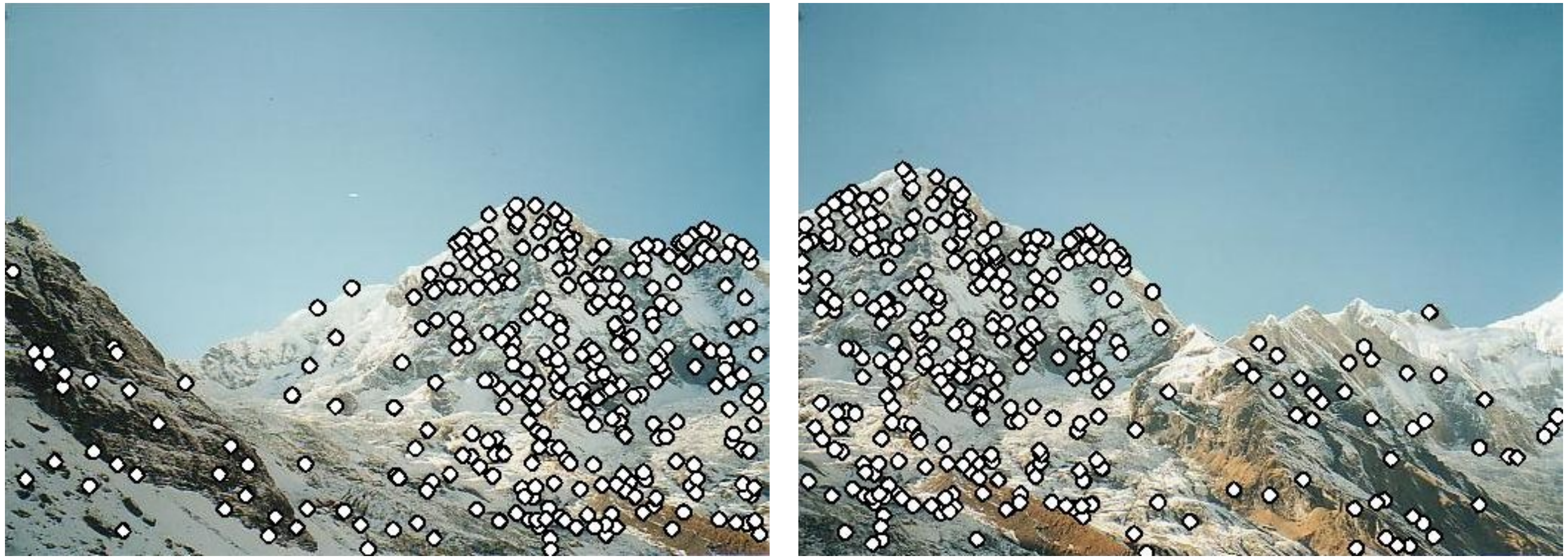
This panorama was generated using **AUTOSTITCH** (freeware), available at
http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html

How do we build panorama?

- We need to match (align) images

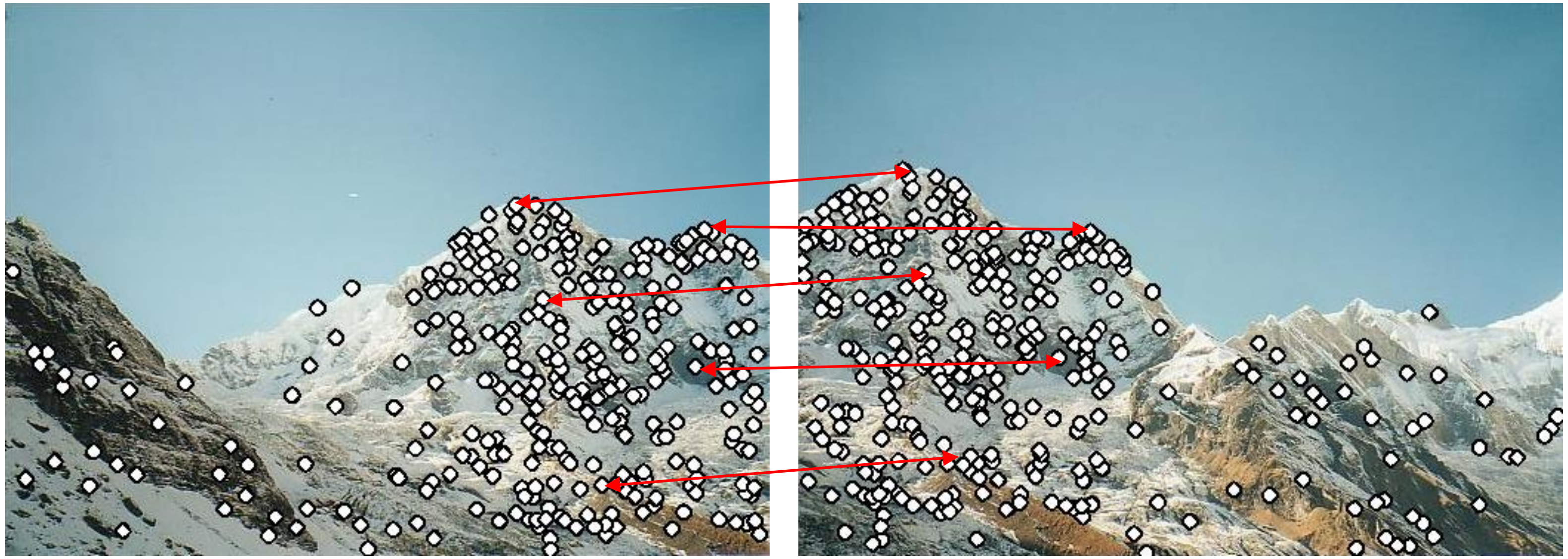
Matching with Features

- Detect feature points in both images



Matching with Features

- Detect feature points in both images
- Find corresponding pairs



Matching with Features

- Detect feature points in both images
- Find corresponding pairs
- Use these pairs to align images

Matching with Features

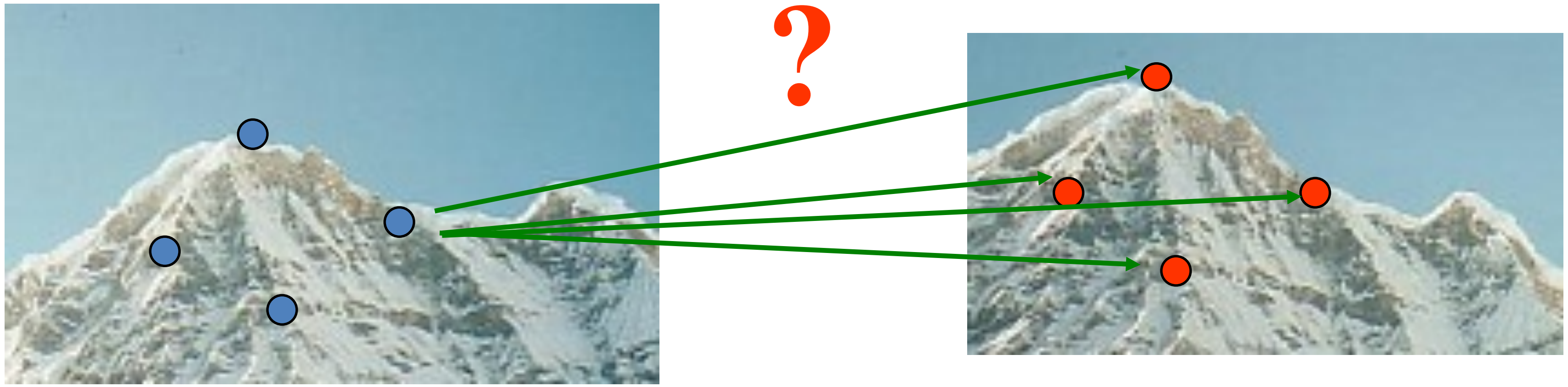
- Problem 1:
  - Detect the *same* point *independently* in both images

no chance to match!

We need a repeatable detector

Matching with Features

- Problem 2:
  - For each point correctly recognize the corresponding one



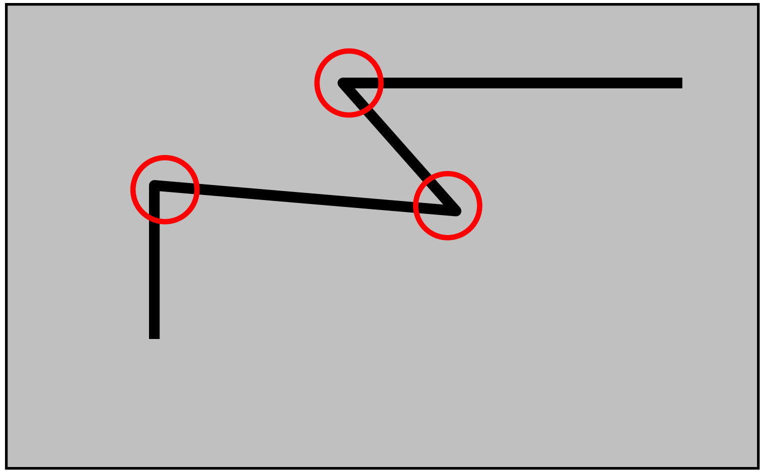
We need a reliable and distinctive descriptor

More motivation…

- Feature points are used also for:
    - Robot navigation
    - Object recognition
    - Image alignment (panoramas)
    - 3D reconstruction
    - Motion tracking
    - Indexing and database retrieval -> Google Images

… other

HARRIS CORNER DETECTOR



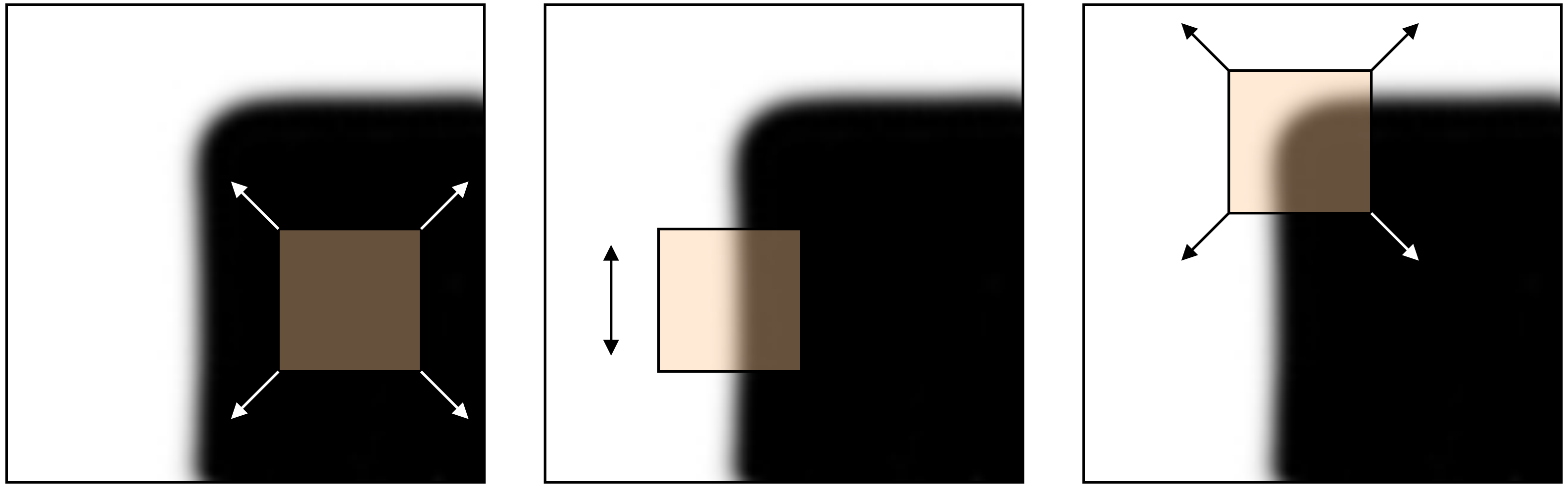
C.Harris, M.Stephens. “A Combined Corner and Edge Detector”. 1988

Finding Corners

- Key property: in the region around a corner, image gradient has two or more dominant directions
- Corners are repeatable and distinctive

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
- => define a corner response function



"flat" region:
no change in all directions

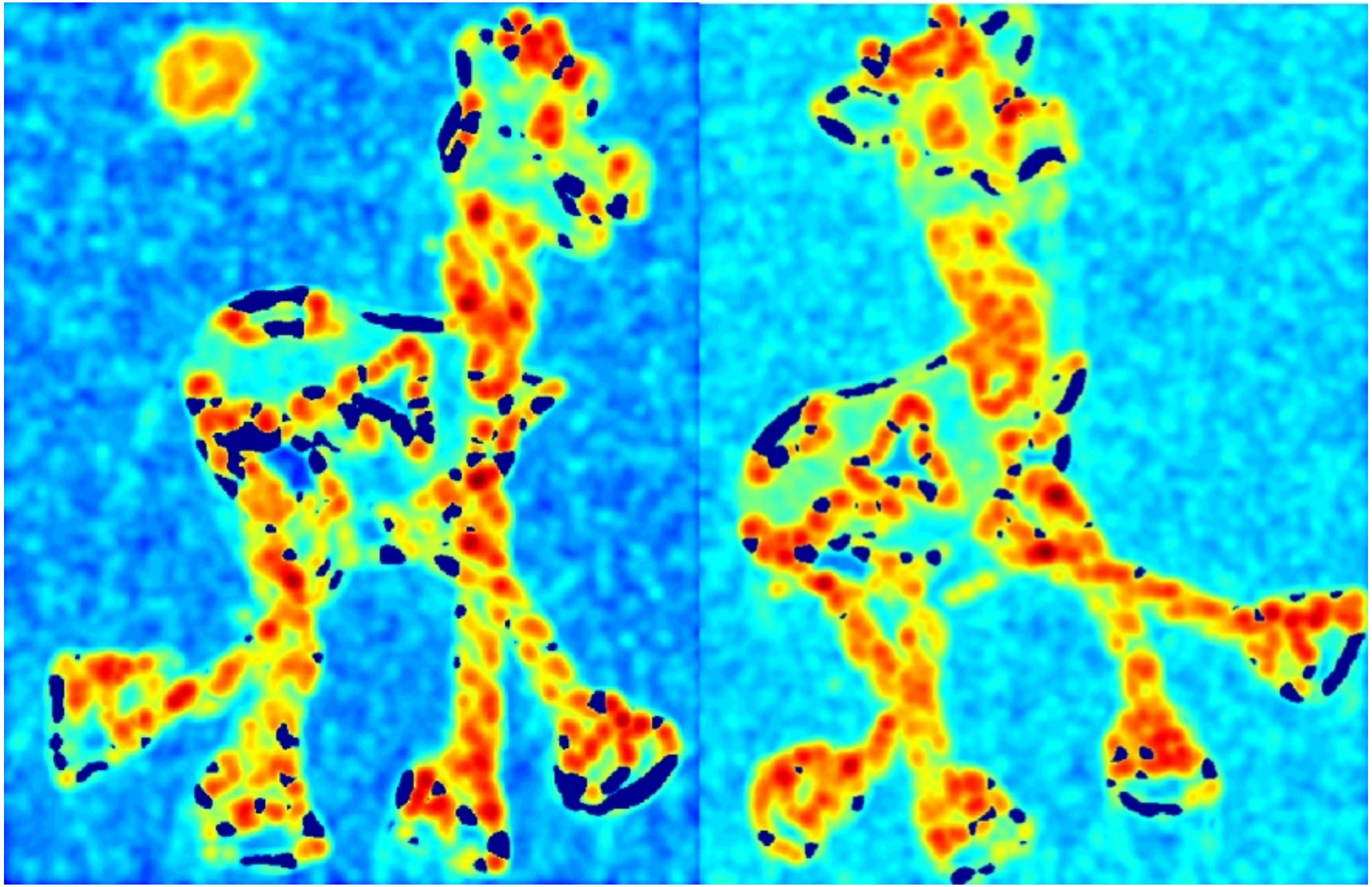
"edge":
no change along the edge direction

"corner":
significant change in all directions

Harris Detector: Workflow

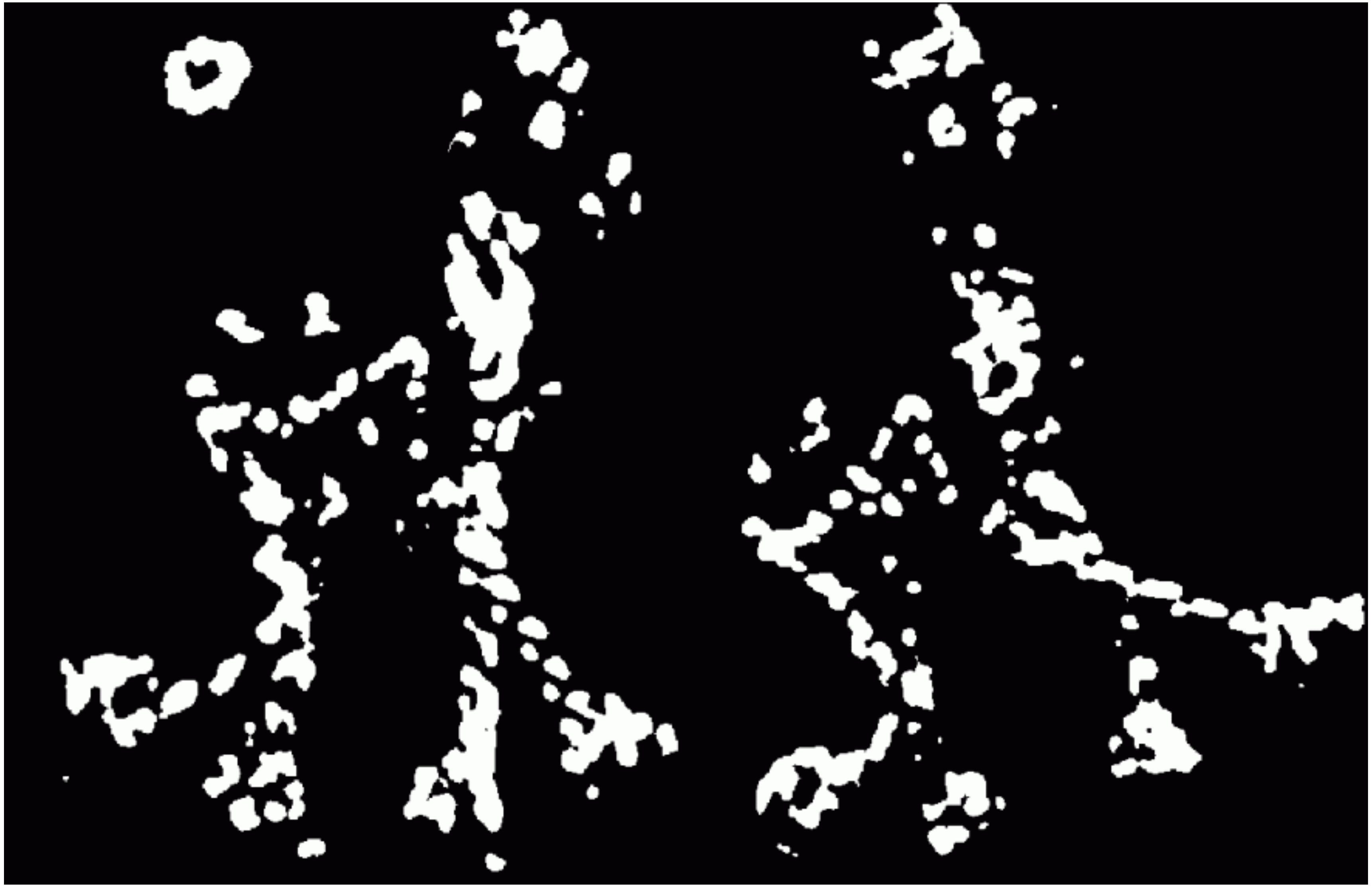
Harris Detector: Workflow

- Compute corner response *R*



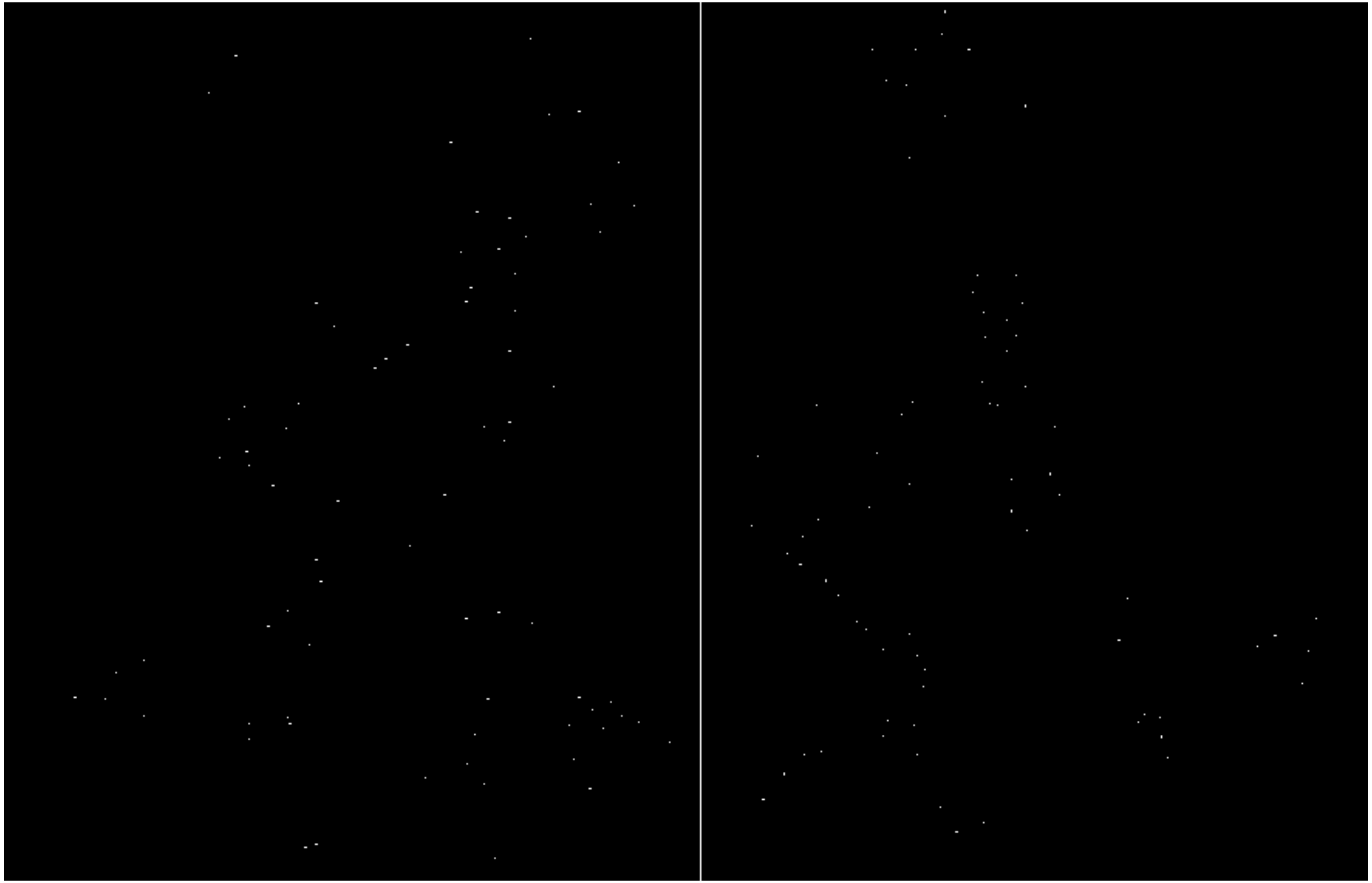
Harris Detector: Workflow

- Find points with large corner response: $R >$ threshold



Harris Detector: Workflow

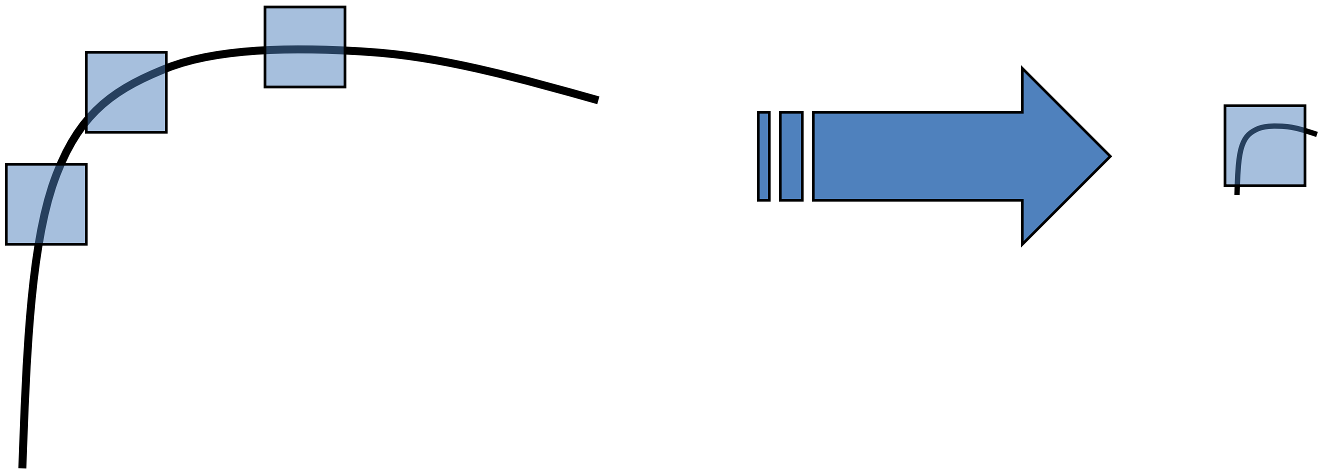
- Take only the points of local maxima of *R*



Harris Detector: Workflow

Harris Detector: Some Properties

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



All points will be classified as edges

Corner !

Summary on Harris properties

- Harris detector is an approach for detecting and extracting corners (i.e. points with high intensity changes in all directions)

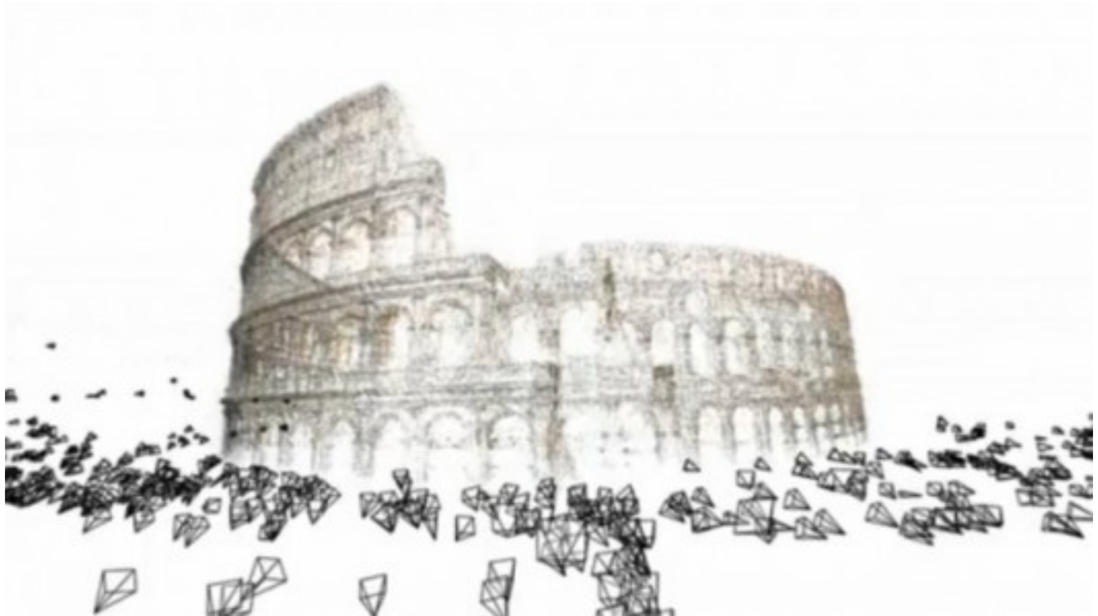
- The detection is Invariant to
  - Rotation
  - Linear intensity changes

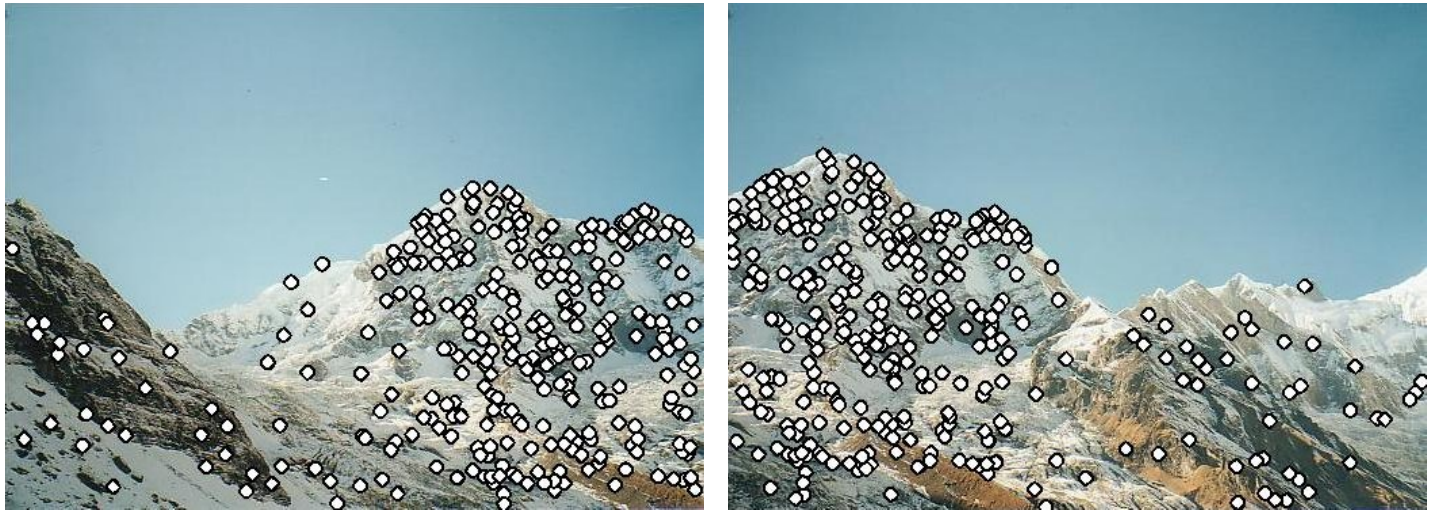
- The detection is NOT invariant to
  - Scale changes
  - Geometric affine changes (e.g. look at the image from the side)

Scale Invariant Feature Transform: SIFT

- Approach for detecting and extracting local feature descriptors
- They are reasonably invariant to changes in =>

- rotation
- scaling
- small changes in viewpoint
- illumination



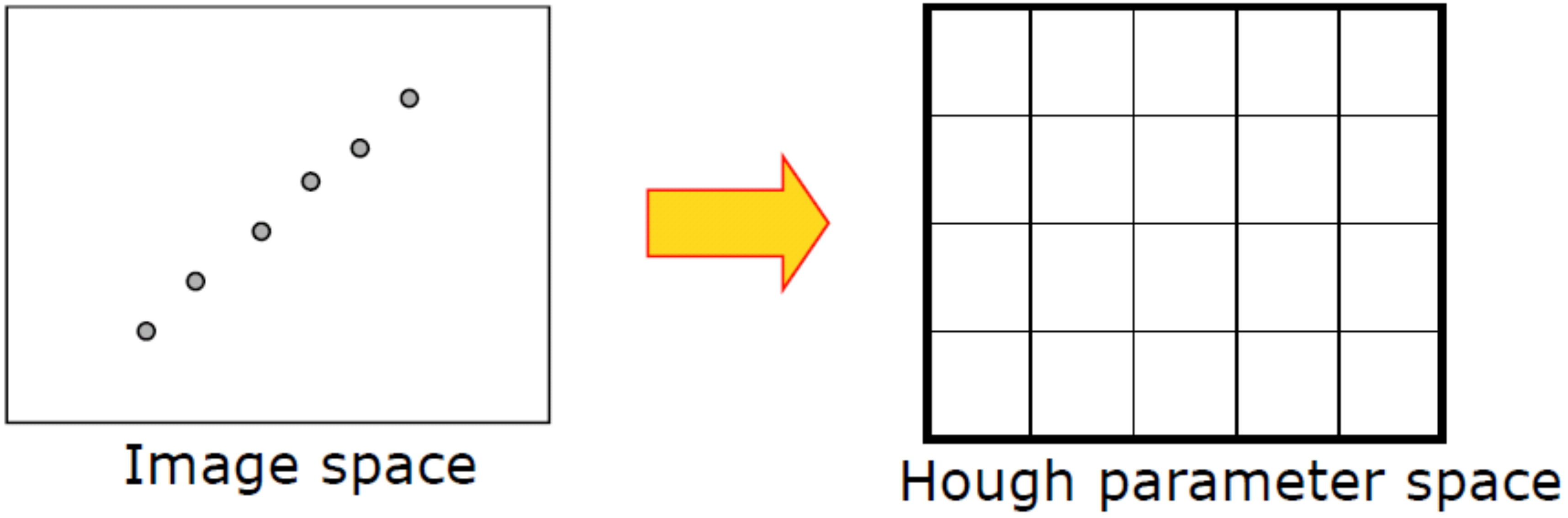


COMPUTER VISION

Hough Transform

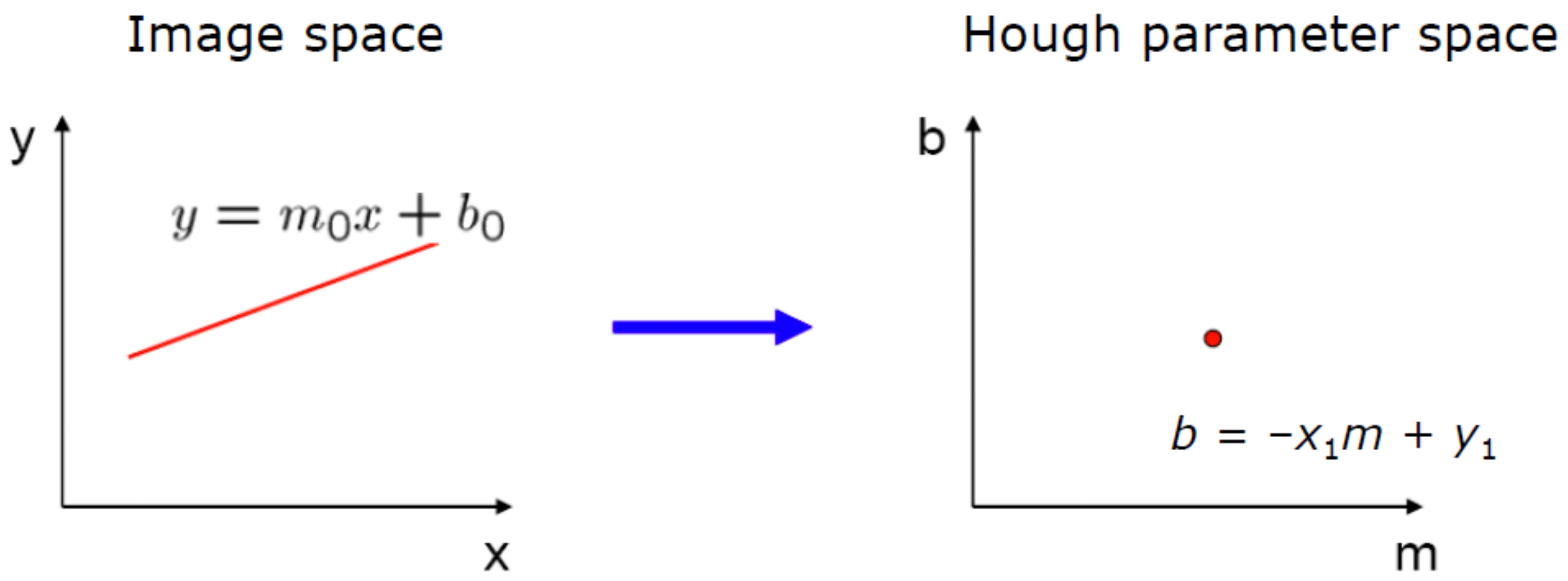
Algorithm 4: Hough-Transform

- Hough Transform uses a voting scheme



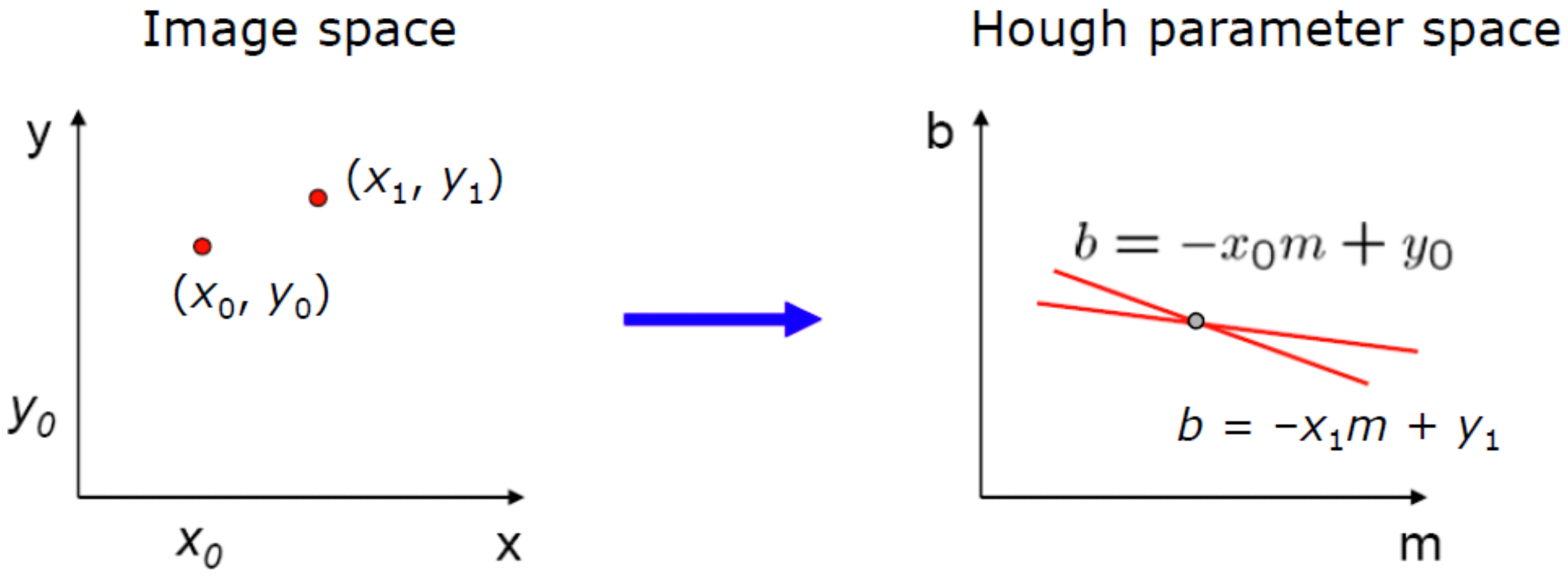
Algorithm 4: Hough-Transform

- Line detection example
- A line in the image corresponds to a point in Hough space



Algorithm 4: Hough-Transform

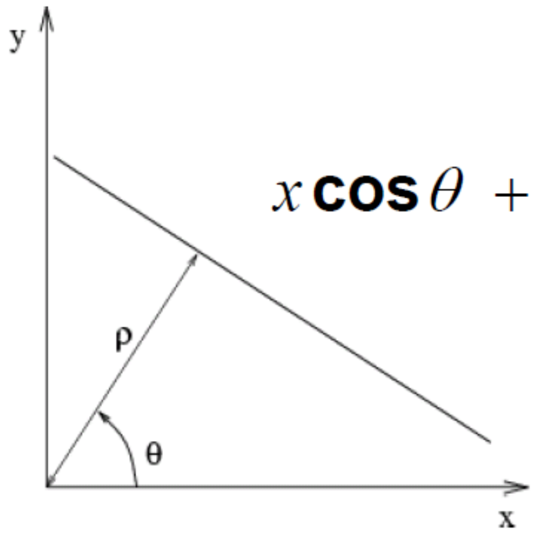
- Where is the line that contains both (x_0, y_0) and (x_1, y_1)?
  - It is the intersection of the lines $b = -x_0m + y_0$ and $b = -x_1m + y_1$



Algorithm 4: Hough-Transform

- Problems with the (m,b) space:
  - Unbounded parameter domain
  - Vertical lines require infinite m

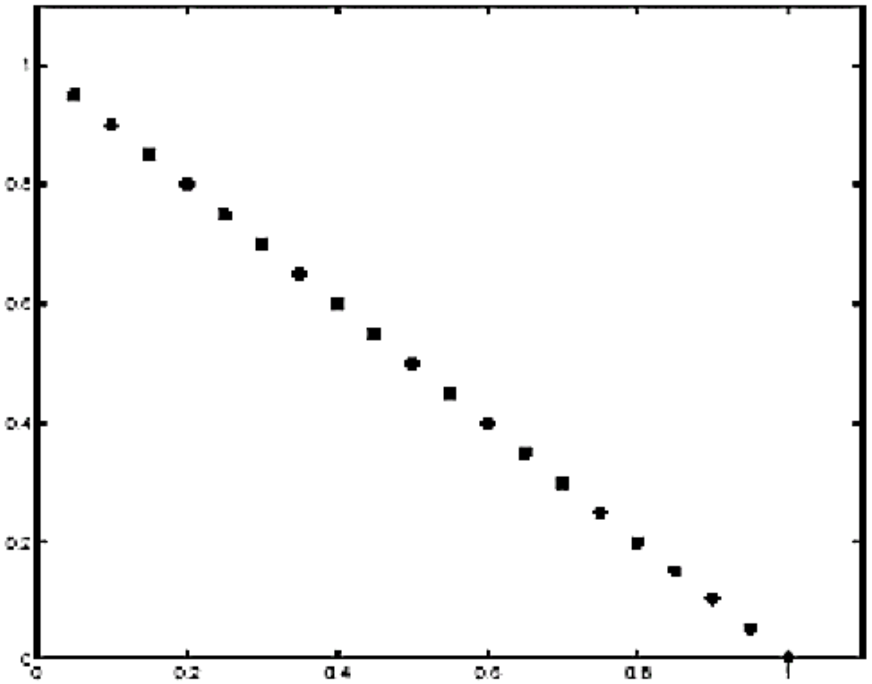
- Alternative: polar representation



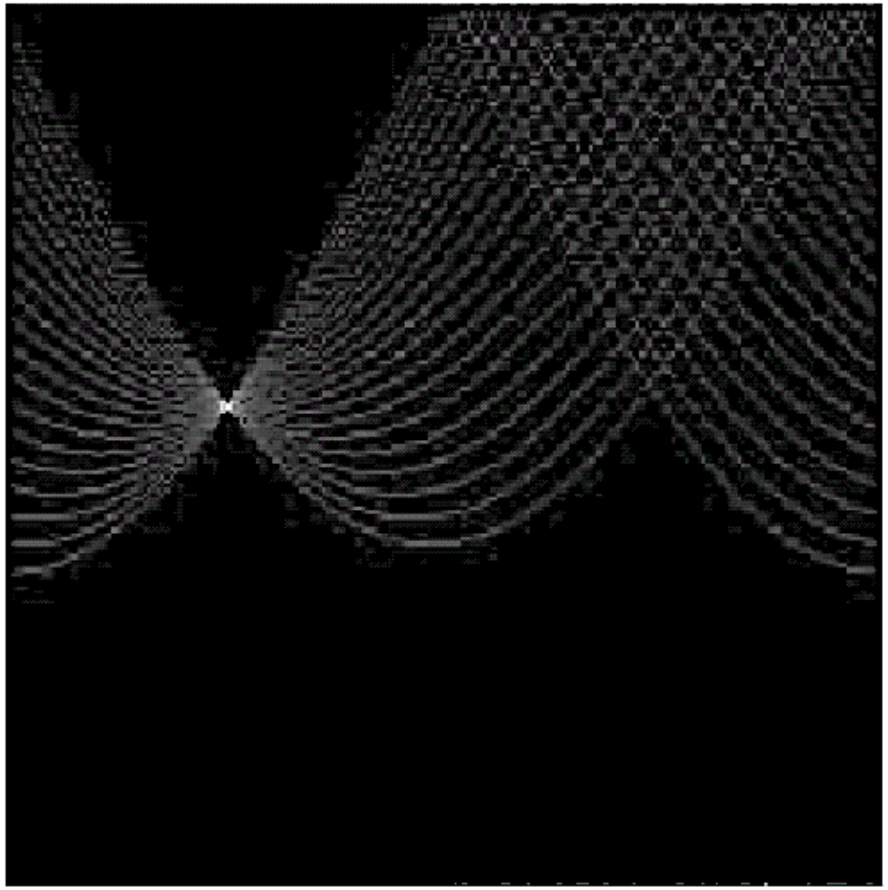
$$x\cos\theta + y\sin\theta = \rho$$

Each point will add a sinusoid in the (θ,ρ) parameter space

Algorithm 4: Hough-Transform



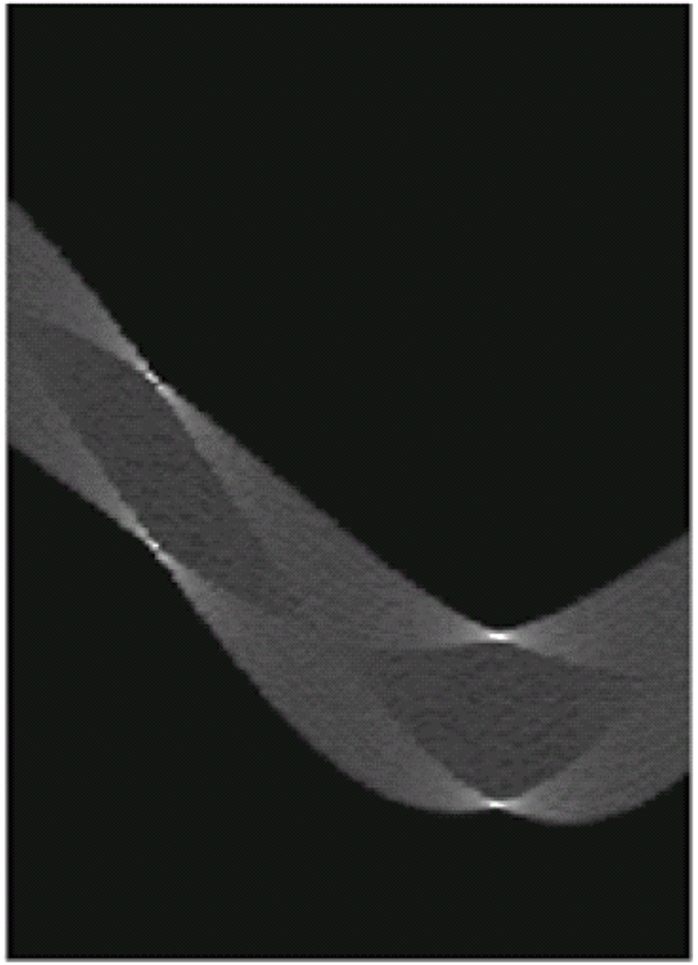
features



votes

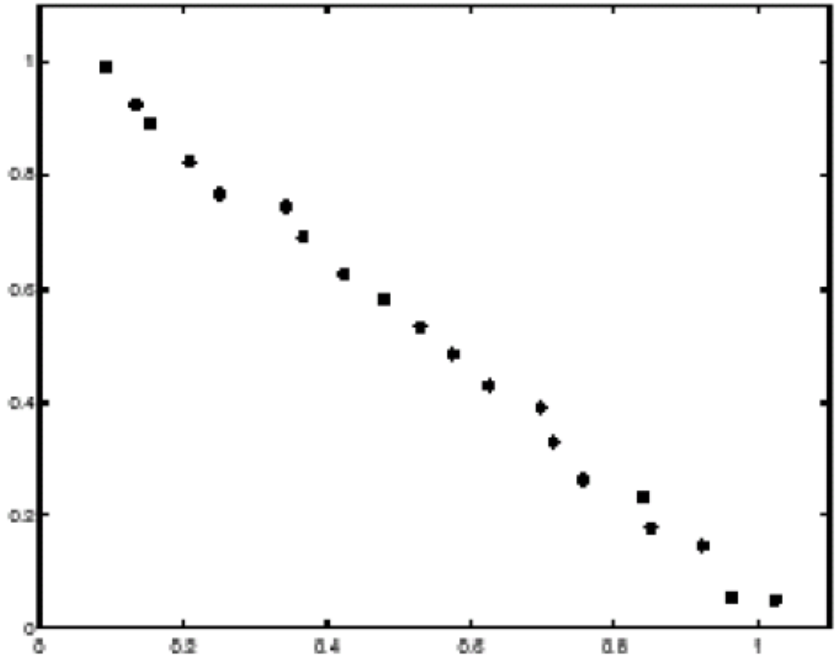
Algorithm 4: Hough-Transform

Square

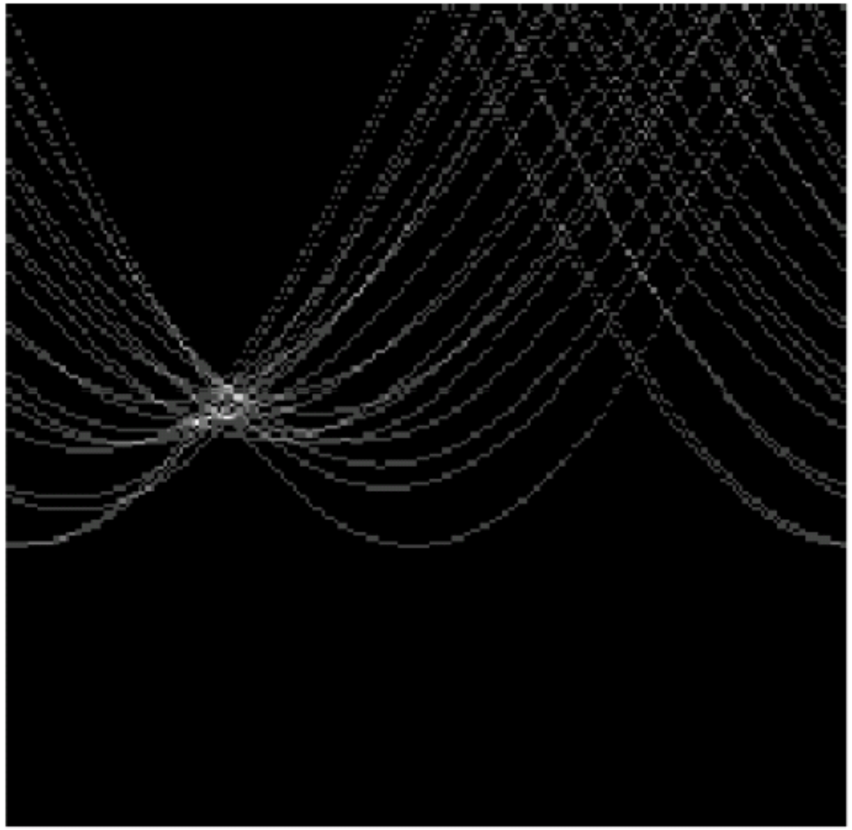


Algorithm 4: Hough-Transform

Effect of Noise



features



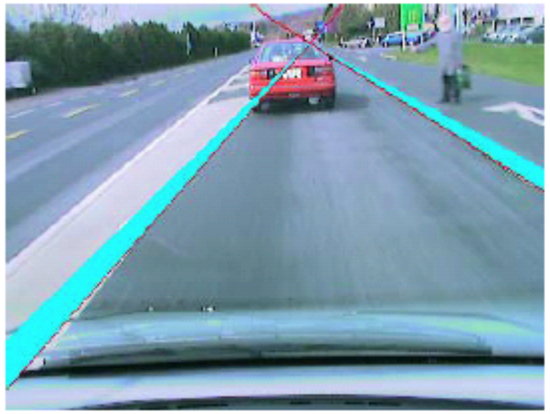
votes

- Peak gets fuzzy and hard to locate

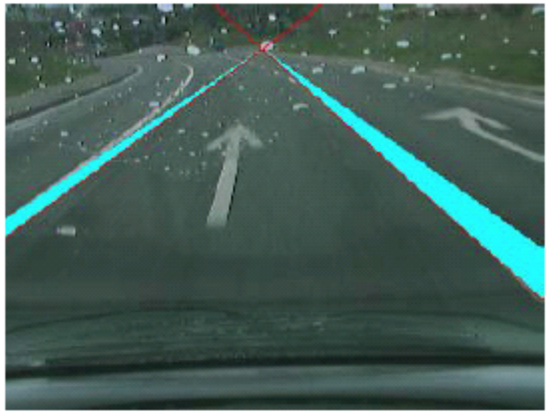
Algorithm 4: Hough-Transform

Application: Lane detection

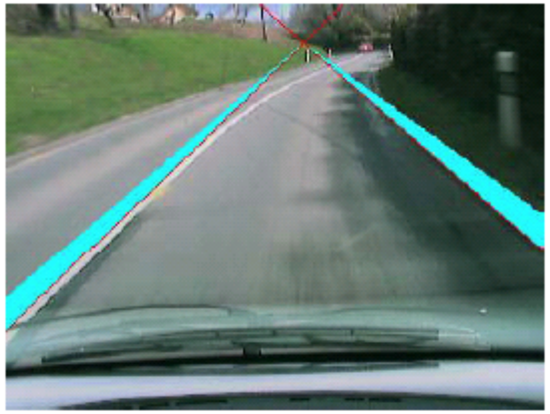
Inner city traffic



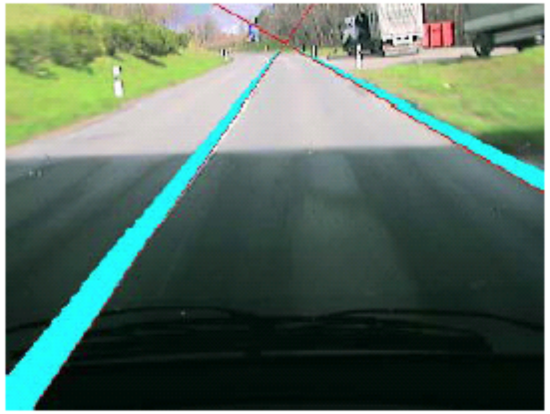
Ground signs



Country-side lane

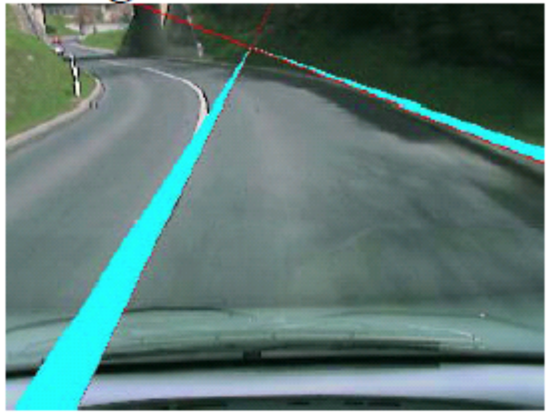


Tunnel exit

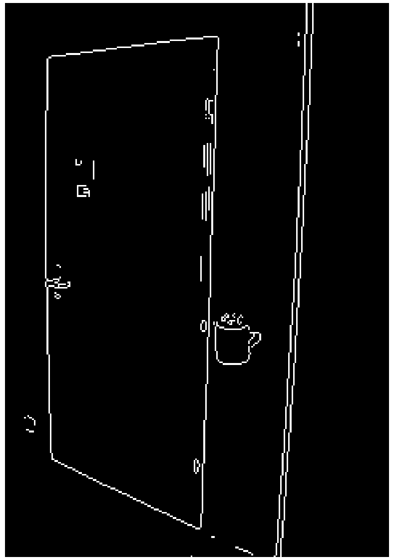


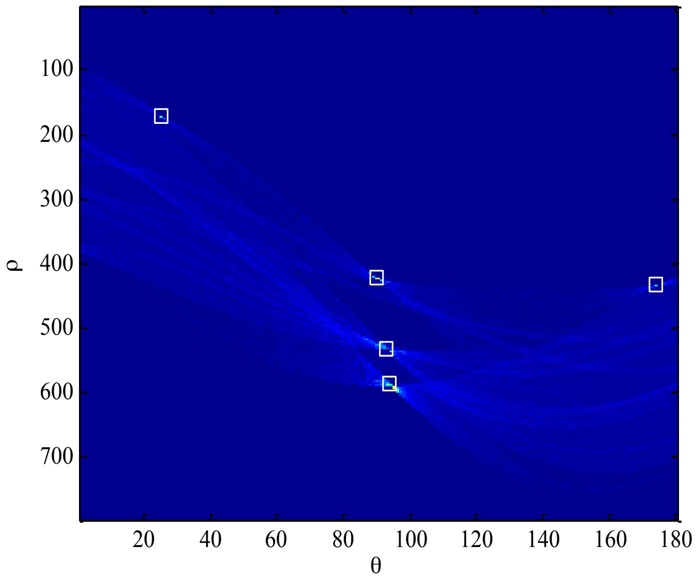
Obscured windscreen

High curvature

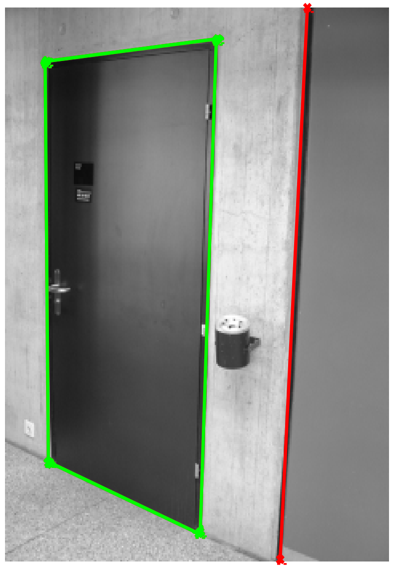


Example – Door detection using Hough Transform





Hough Transform



Hough Transform

- Advantages
  - Noise and background clutter do not impair detection of local maxima
  - Partial occlusion and varying contrast are minimized
- Negatives
  - Requires time and space storage that increases exponentially with the dimensions of the parameter space