MTX4700: Experimental Robotics
Robot Vision and Image Processing

Dr. Thierry Peynot
Dr. Stefan B. Williams

[with material from Gonzalez & Woods, Chieh-Chih (Bob) Wang, Martial Hebert, Sebastian Thrun,…]

Course Outline

Unit of Study Program:

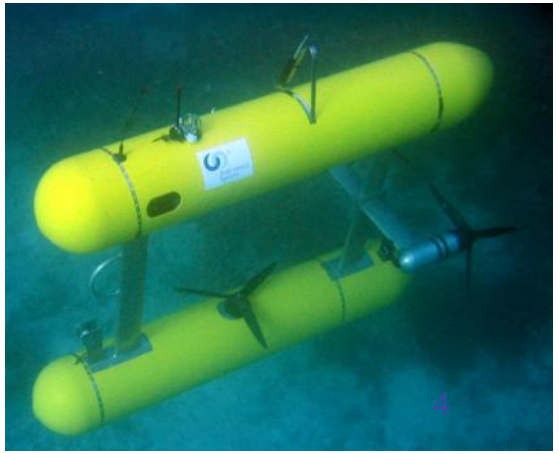
Week	Date	Content	Labs	Due Dates
1	5 Mar	Introduction, history & philosophy of robotics		
2	12 Mar	Robot kinematics & dynamics	Kinematics/Dynamics Lab	
3	19 Mar	Sensors, measurements and perception	"	
4	26 Mar	Robot vision and vision processing.	*No Tute (Good Friday)*	**Kinematics Lab**
	2 Apr	BREAK		
5	9 Apr	Localization and navigation	Sensing with lasers	
6	16 Apr	Estimation and Data Fusion	Sensing with vision	
7	23 Apr	Extra tutorial session (sensing)	Robot Navigation	**Sensing Lab**
8	30 Apr	Obstacle avoidance and path planning	Robot Navigation	
9	7 May	Extra tutorial session (nav demo)	Major project	**Navigation Lab**
10	14 May	Robotic architectures, multiple robot systems	"	
11	21 May	Robot learning	"	
12	28 May	Case Study	"	
13	4 June	Extra tutorial session (Major Project)	"	**Major Project**
14		Spare		

ACFR
AUSTRALIAN CENTRE FOR FIELD ROBOTICS

Goals

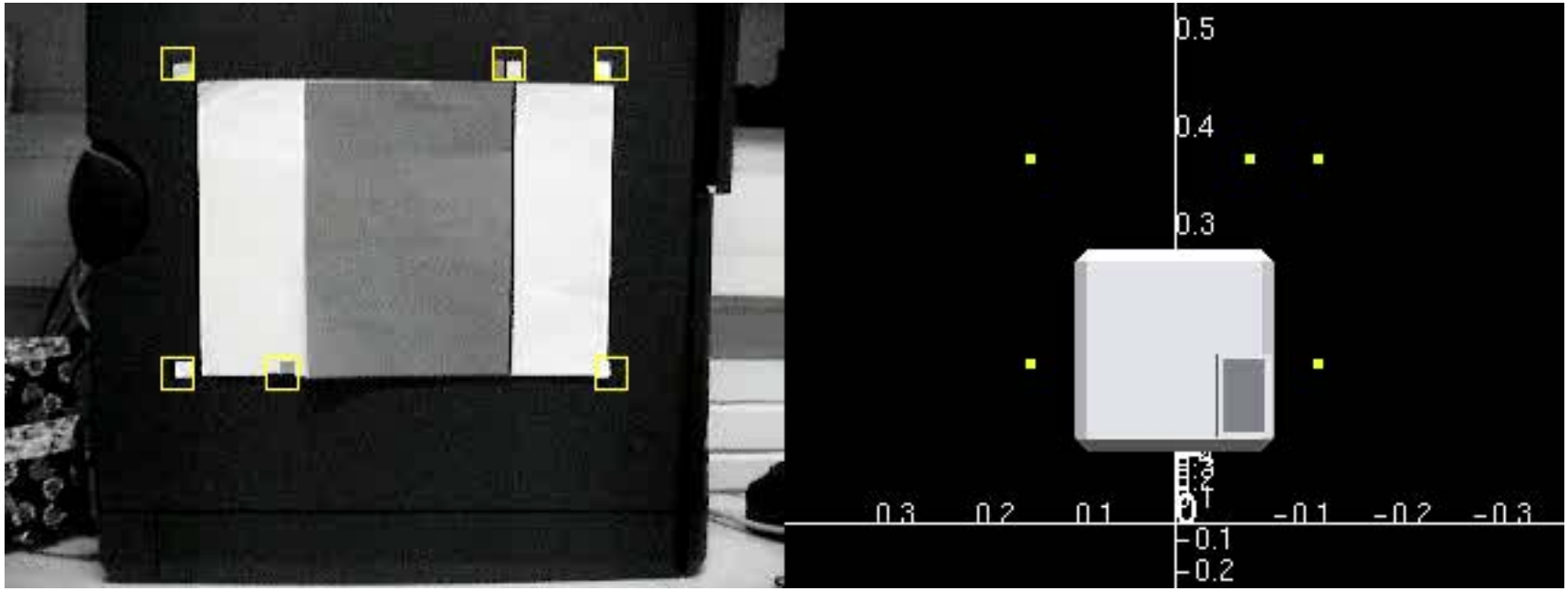
- To familiarize you with the basic techniques of Robot/ Computer Vision.
- To get you excited!
- To let you experience (and appreciate!) the difficulties of real-world computer vision

ACFR
AUSTRALIAN CENTRE FOR FIELD ROBOTICS



Vision in Robotics: Applications

- SLAM



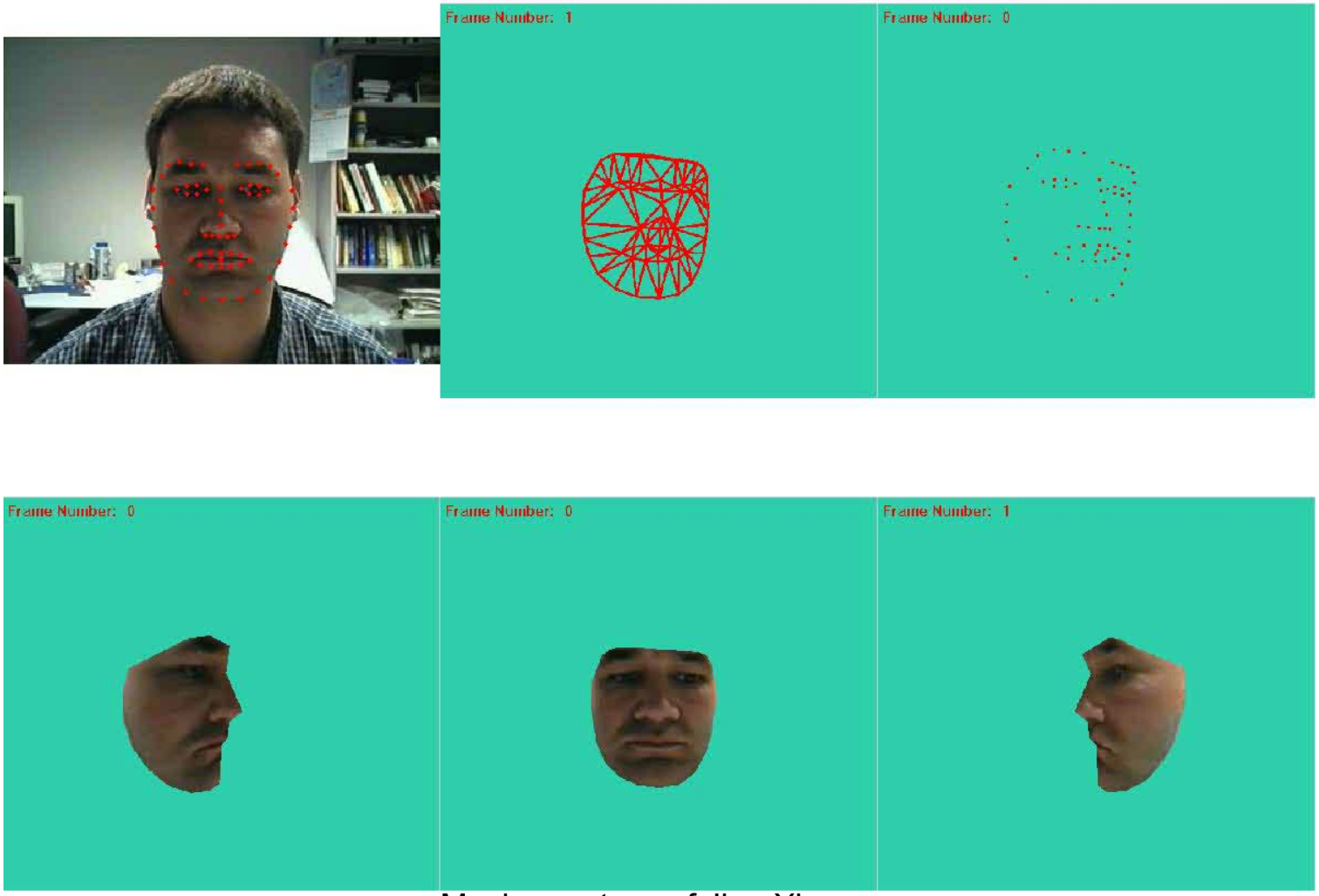
Movie courtesy of Andrew Davison

Vision in Robotics: Applications

- Multiview Structure from Motion

Movie courtesy of N. Snavely et al. at the University of Washington

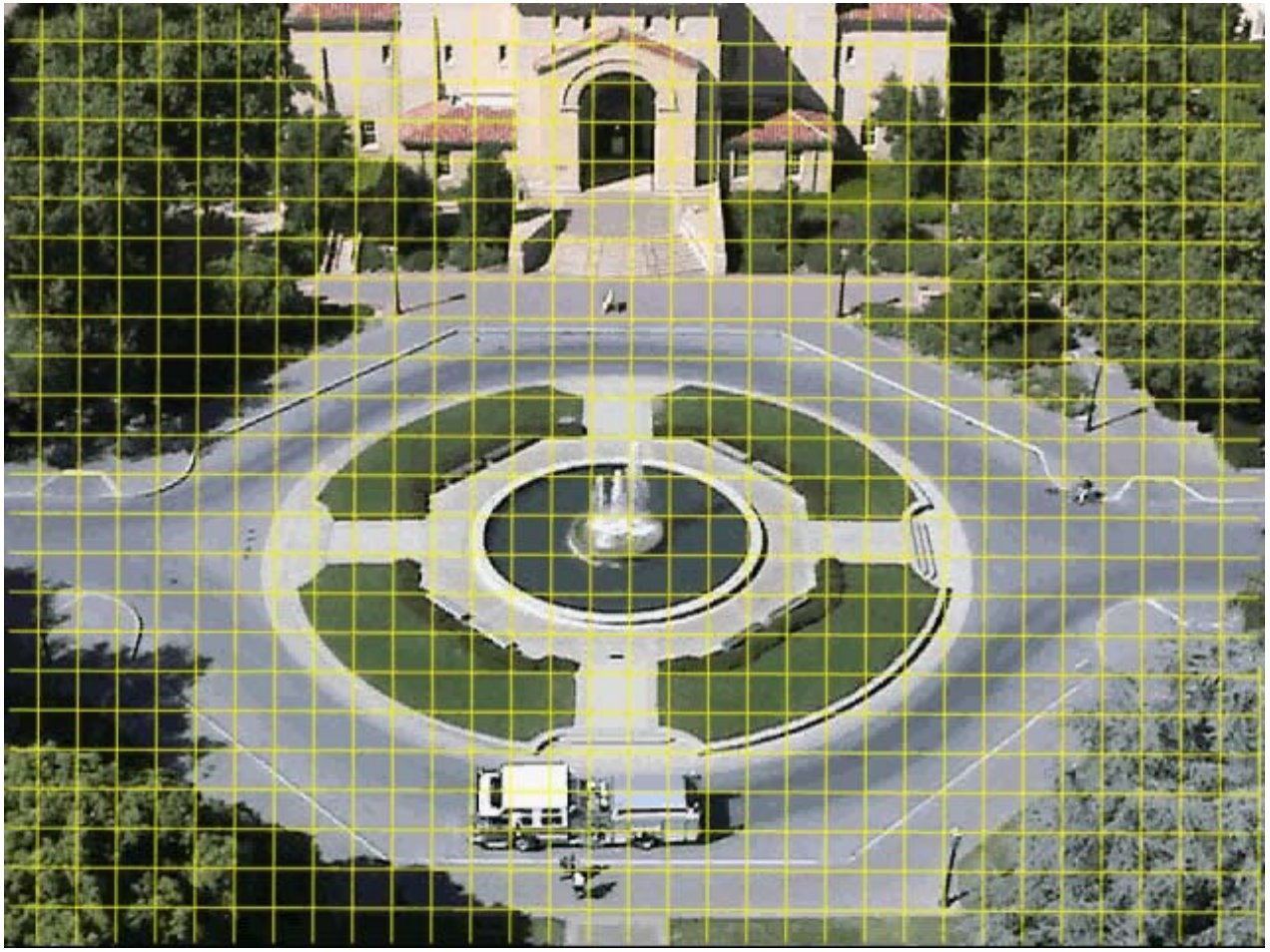
Vision in Robotics: Applications



Movie courtesy of Jing Xiao

Vision in Robotics: Applications

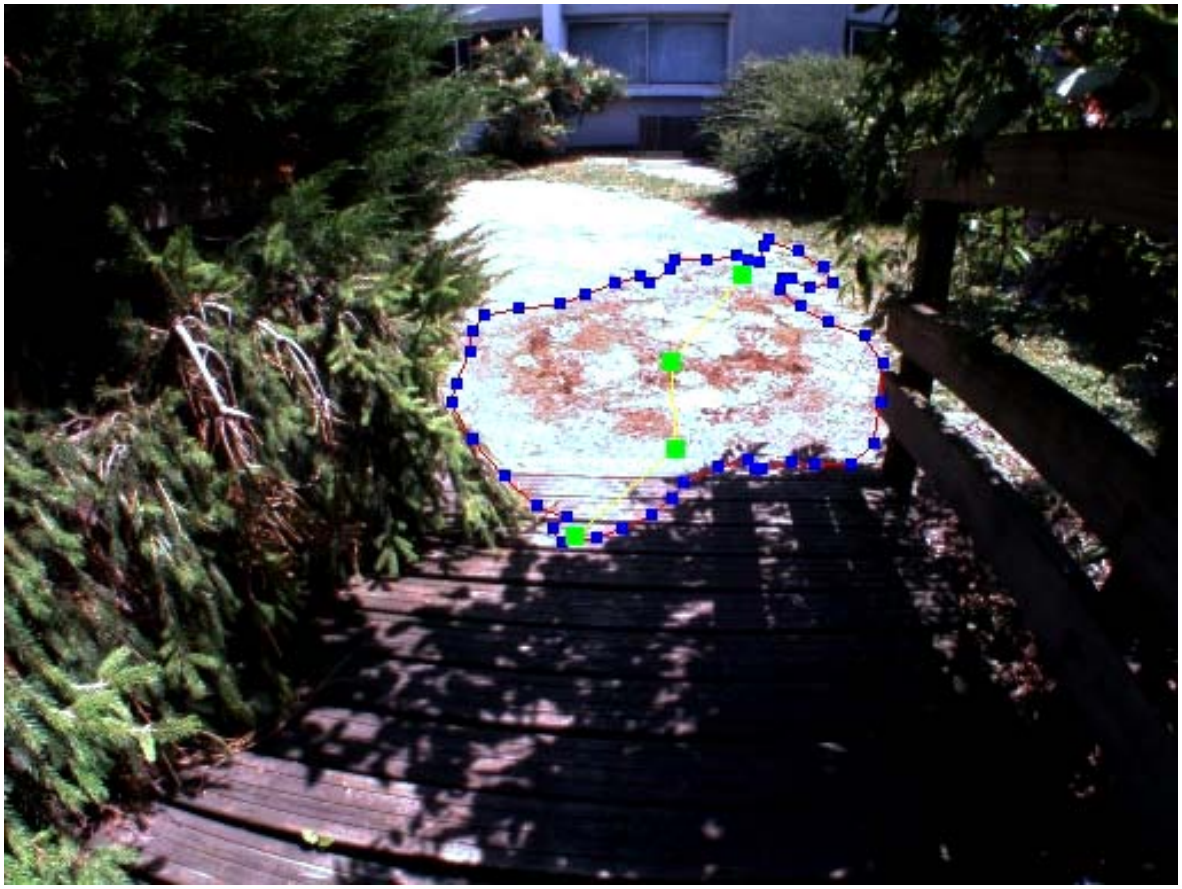
- Tracking



Lookingbill, Lieb, Stavens and Thurn, ICRA2005

Vision in Robotics: Applications

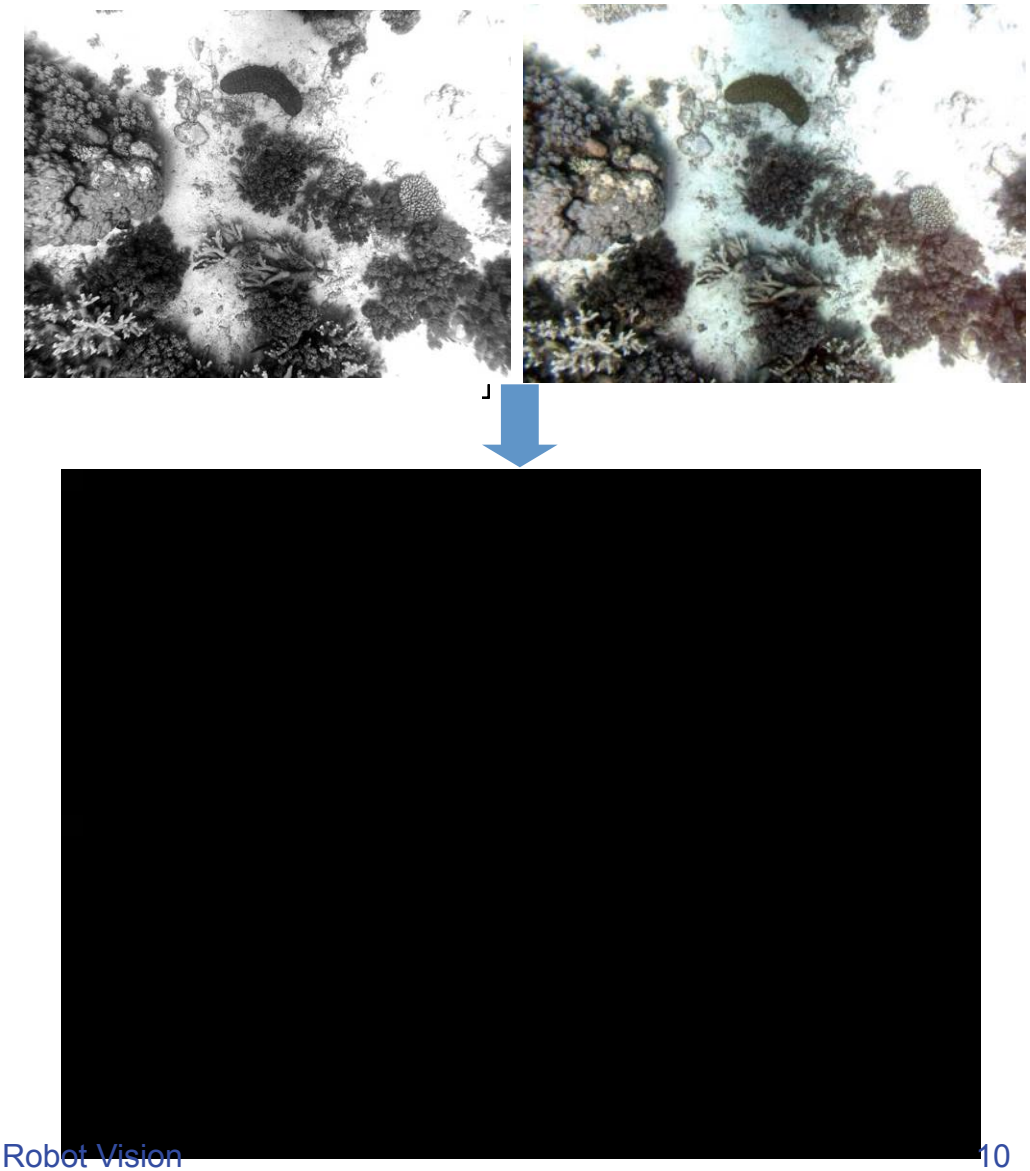
- Path-following



[2004, LAAS-CNRS]

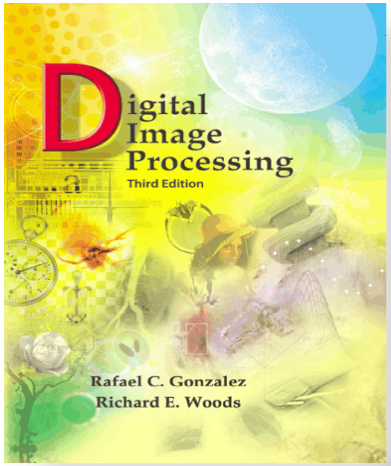
Vision in Robotics: Applications

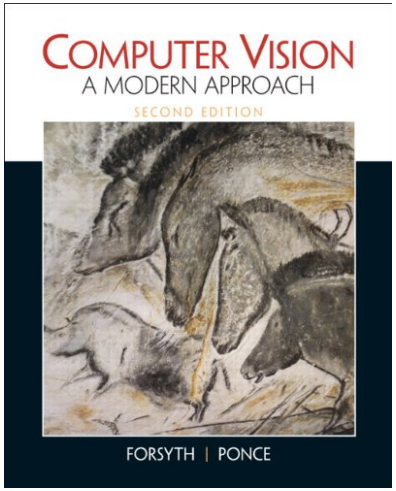
- (Dense)
  3D Mapping
  from Stereovision

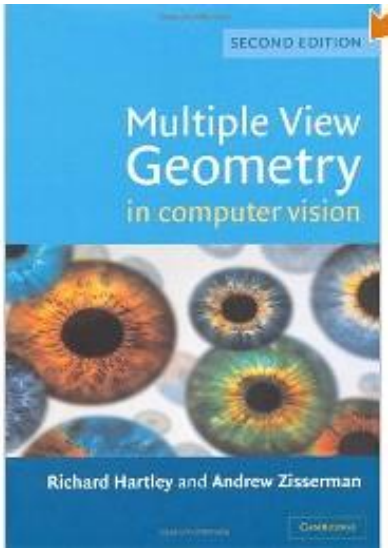


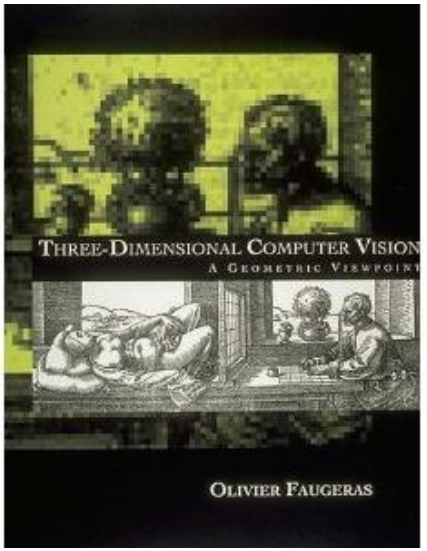
[Johnson-Robertson et al., 2009]

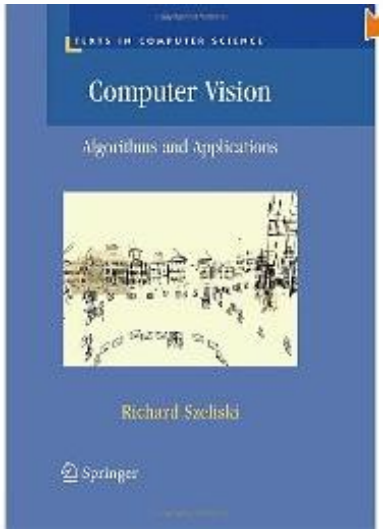
(Some) Major References: Vision

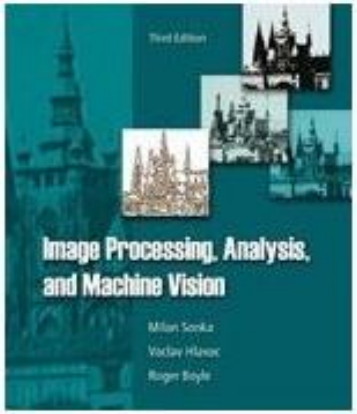




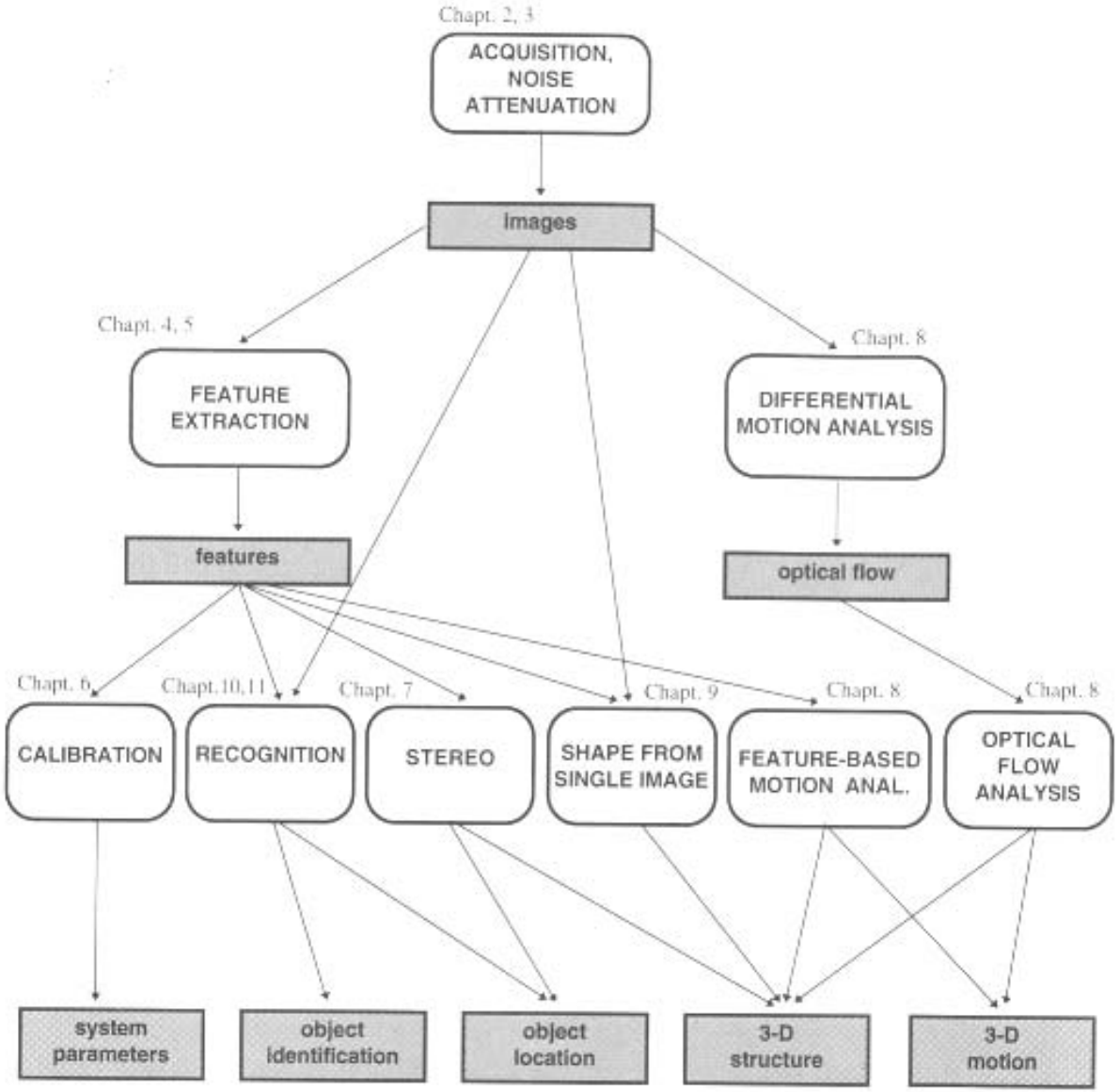








Computer Vision [Trucco&Verri'98]

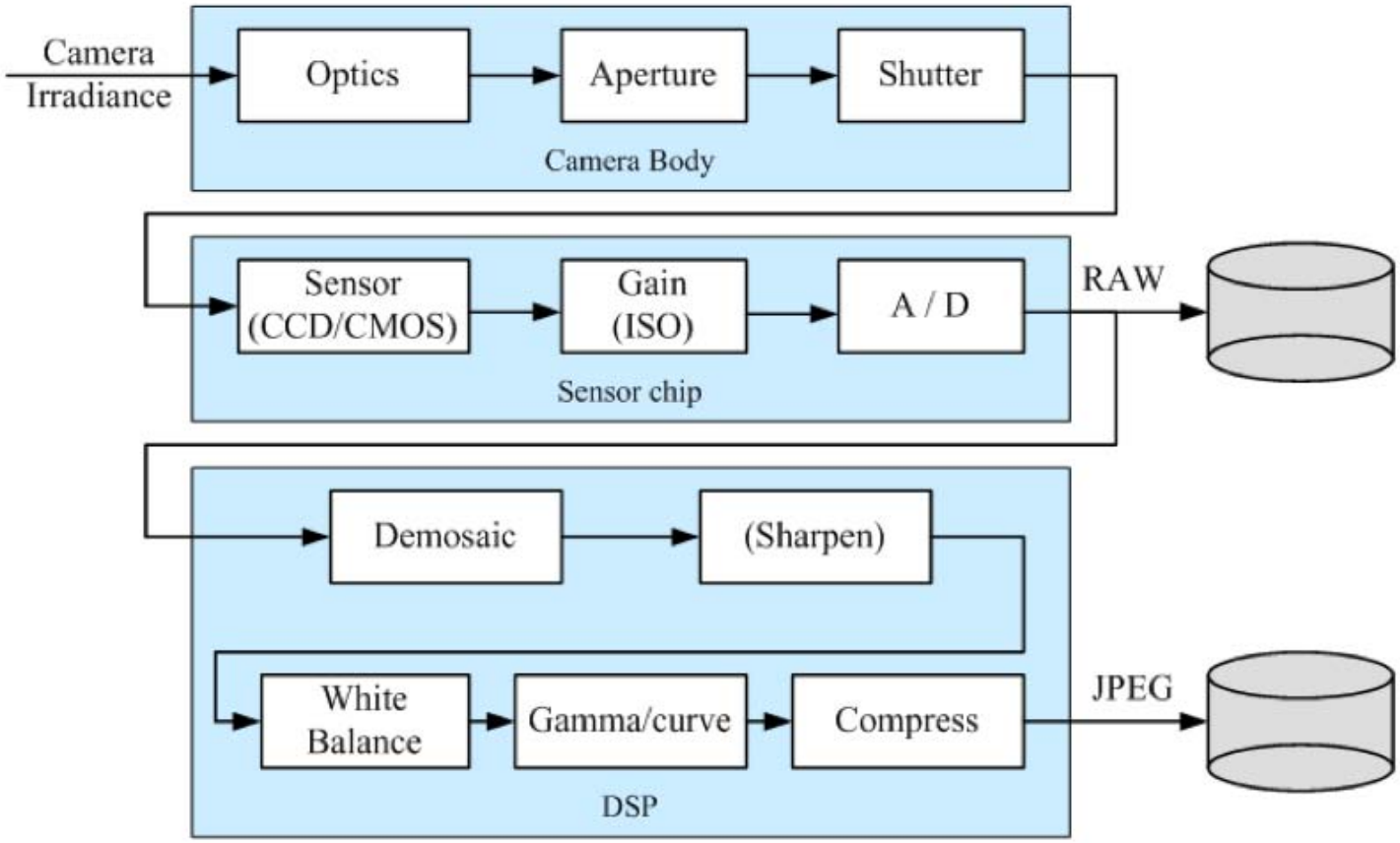


ACFR
AUSTRALIAN CENTRE FOR FIELD ROBOTICS

Topics we will (hardly) cover…

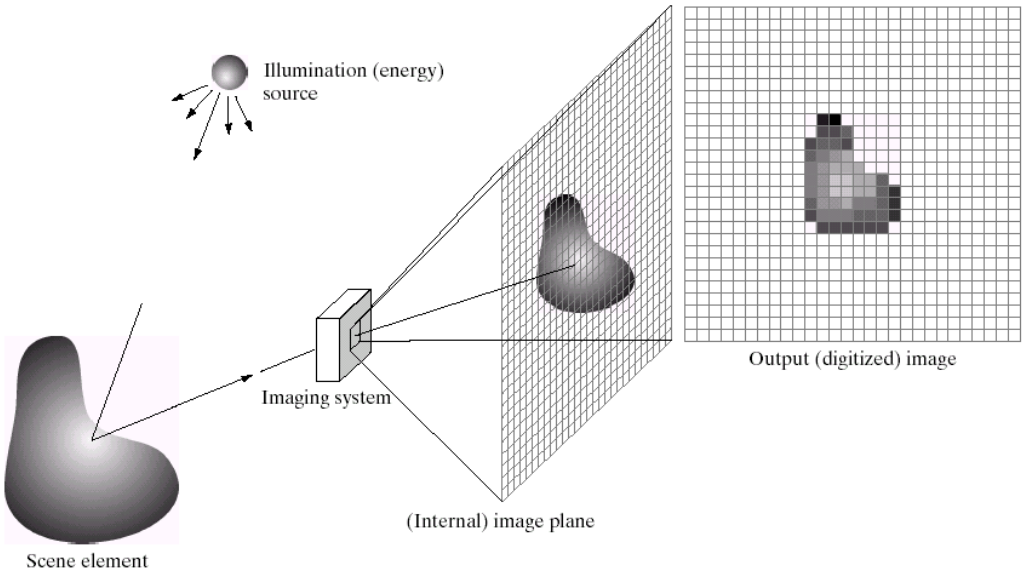
- Digital Imaging: Acquisition, Basic Definitions
- Camera Geometry/Model
  - Ideal pinhole model
  - Geometric models
- Introduction to Camera Calibration
- Feature Extraction (Edge & Corner)
  - Sobel and Canny Edge Detectors
  - Harris Corners
  - Interest Points
- Image Matching using detected corners

Image Acquisition



Digital Image Fundamentals

- Digital image = 2D function: (x,y) ➔ f(x,y) (1D or 3D)

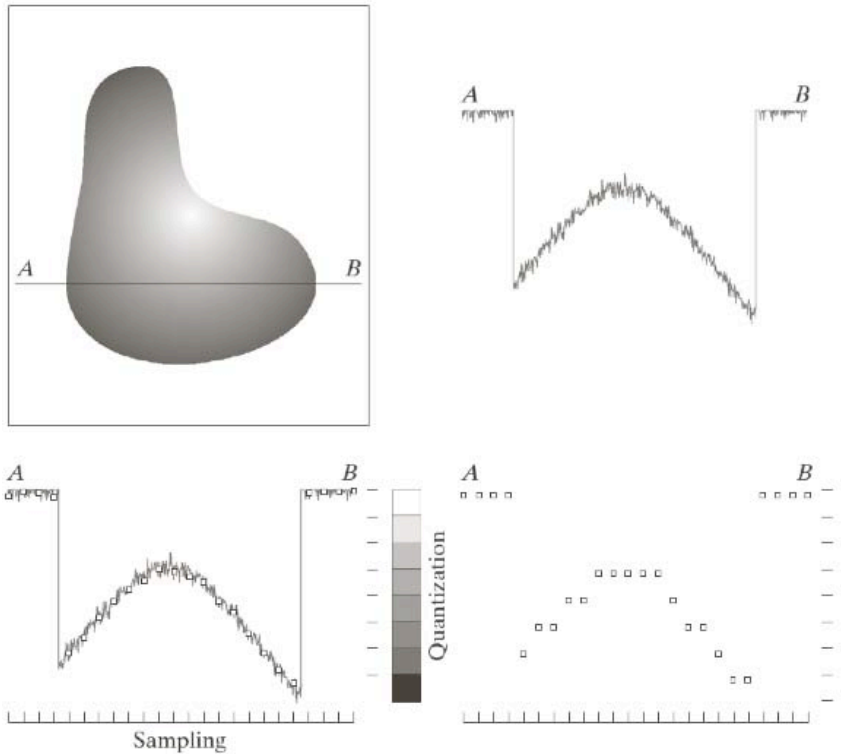


a
b c d e

FIGURE 2.15 An example of the digital image acquisition process. (a) Energy ("illumination") source. (b) An element of a scene. (c) Imaging system. (d) Projection of the scene onto the image plane. (e) Digitized image.

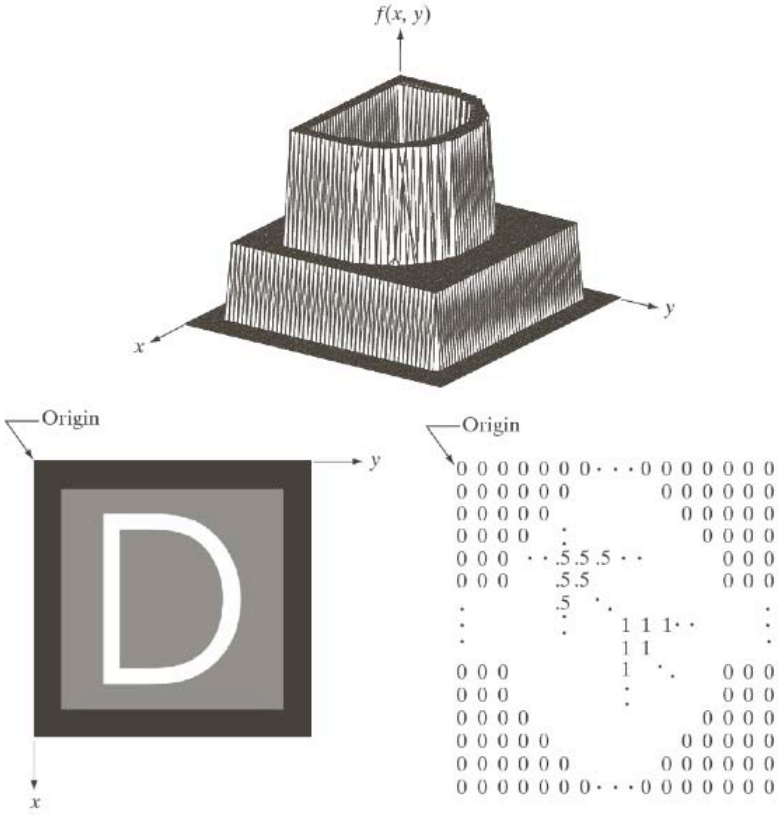
Digital Image Fundamentals

- Sampling and Quantization



Digital Image Fundamentals

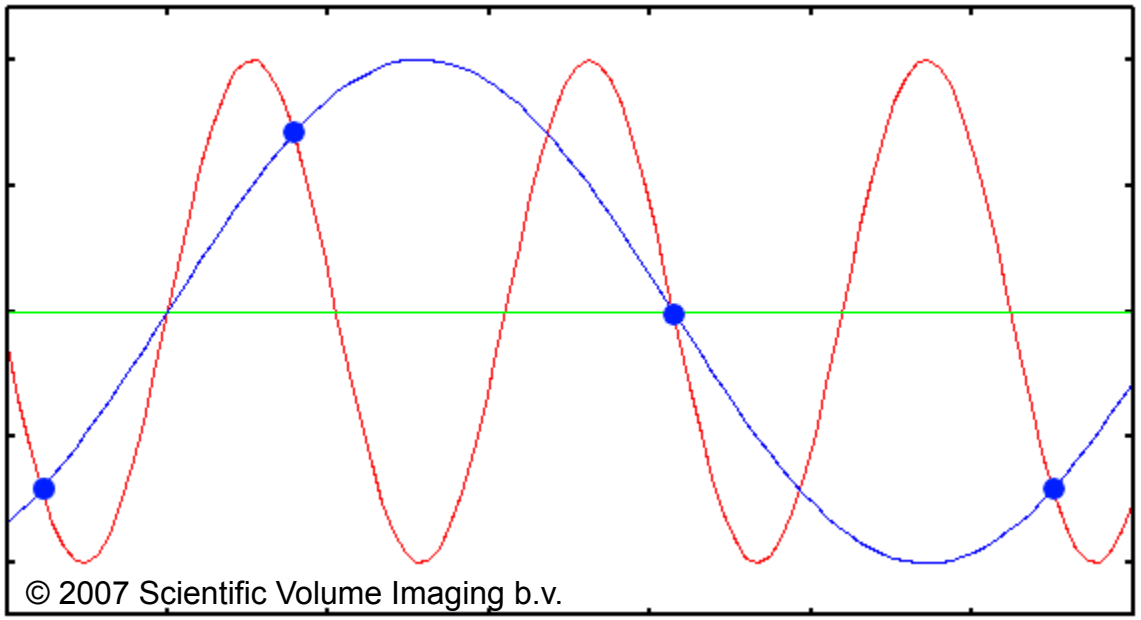
- Representation of a digital image



Digital Image Fundamentals

- Aliasing Effect

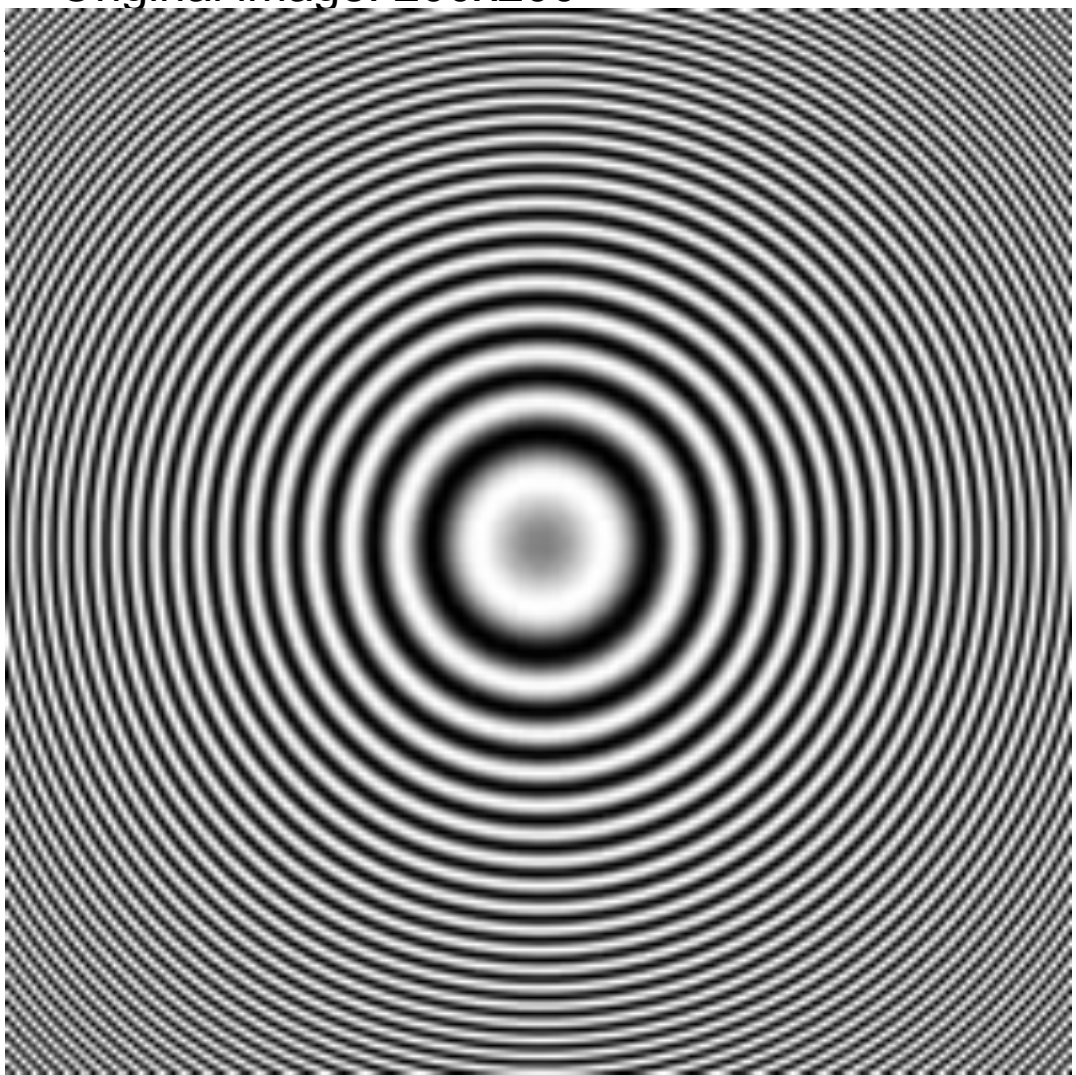
Example in 1 dimension

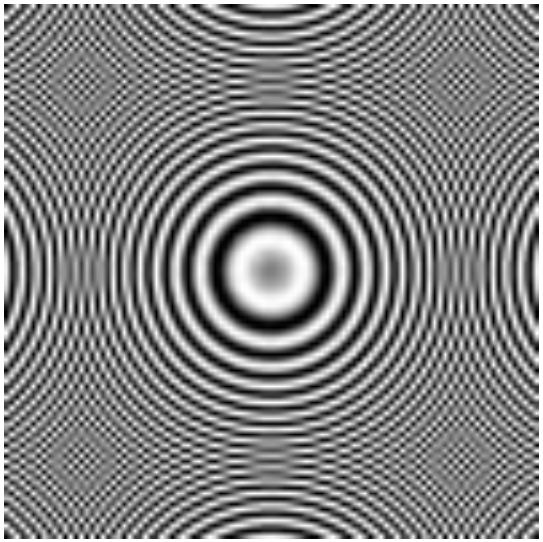


Digital Image Fundamentals

- Aliasing Effect

Original image: 200x200





Sampled image:
100x100 pixels

Digital Image Fundamentals

- Aliasing Effect

Aliasing Effect

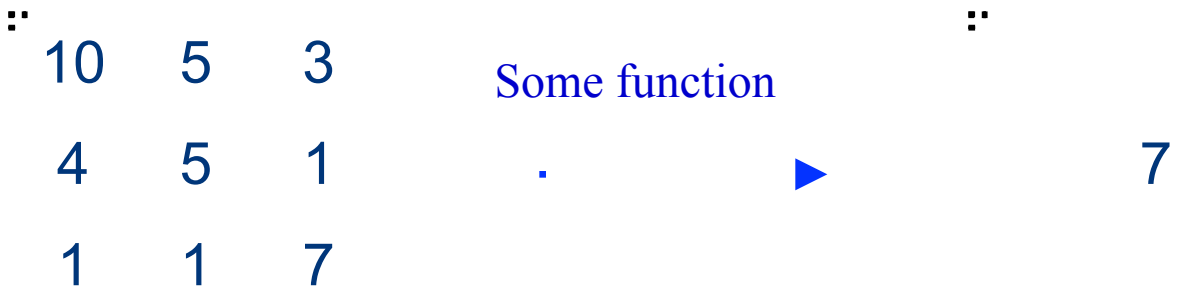
Original image: 622x756 pixels

205x250 pixels

“Moiré pattern”

Basic Image Filtering

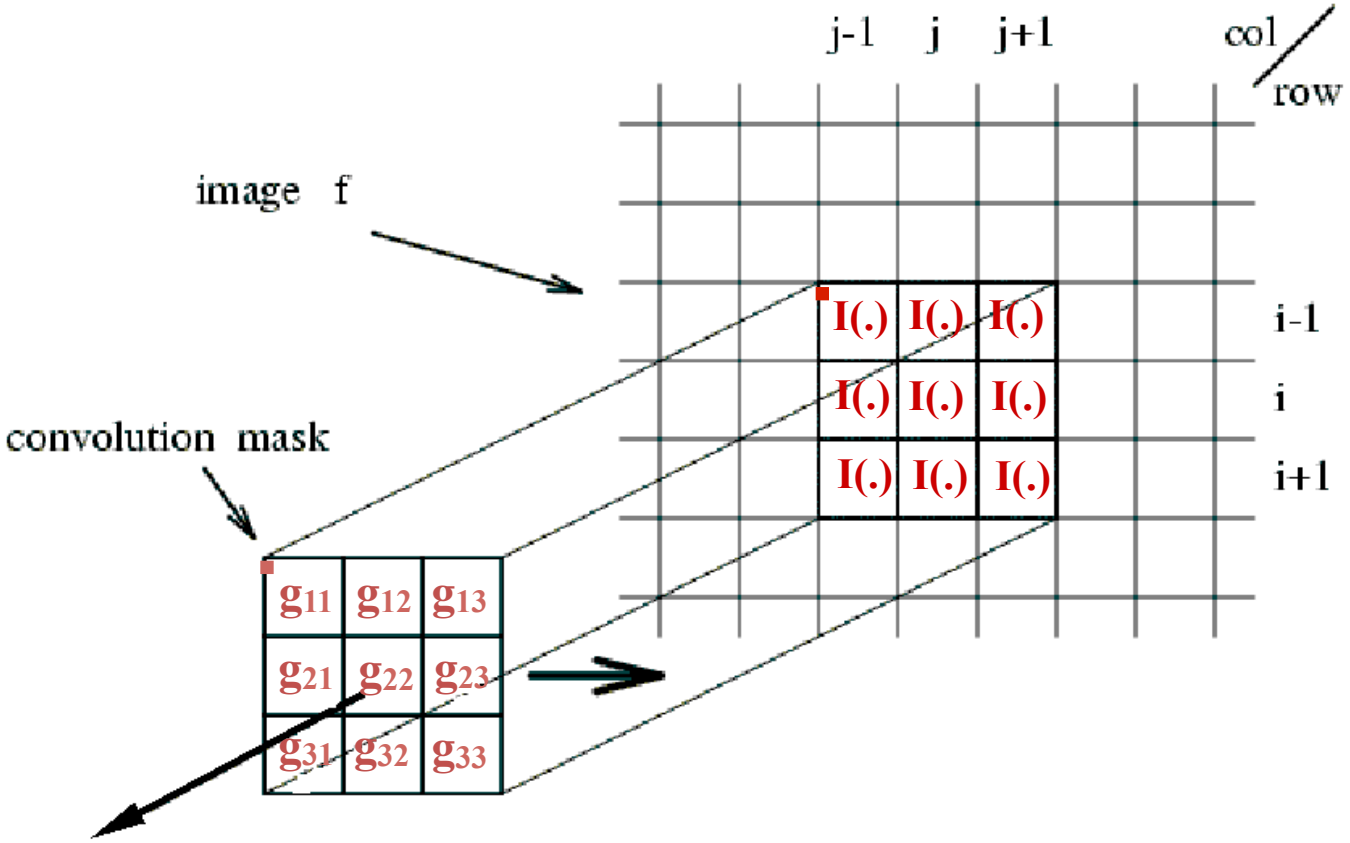
- Modify the pixels in an image based on some function of a local neighborhood of the pixels



Linear Filtering

- Linear case is simplest and most useful
  - Replace each pixel with a linear combination of its neighbors.
- The prescription for the linear combination is called the convolution kernel.

Linear Filter = Convolution



$$f(i,j) = g_{11}\, I(i-1,j-1) + g_{12}\, I(i-1,j) + g_{13}\, I(i-1,j+1) + g_{21}\, I(i,j-1) + g_{22}\, I(i,j) + g_{23}\, I(i,j+1) + g_{31}\, I(i+1,j-1) + g_{32}\, I(i+1,j) + g_{33}\, I(i+1,j+1)$$

Linear Filter = Convolution

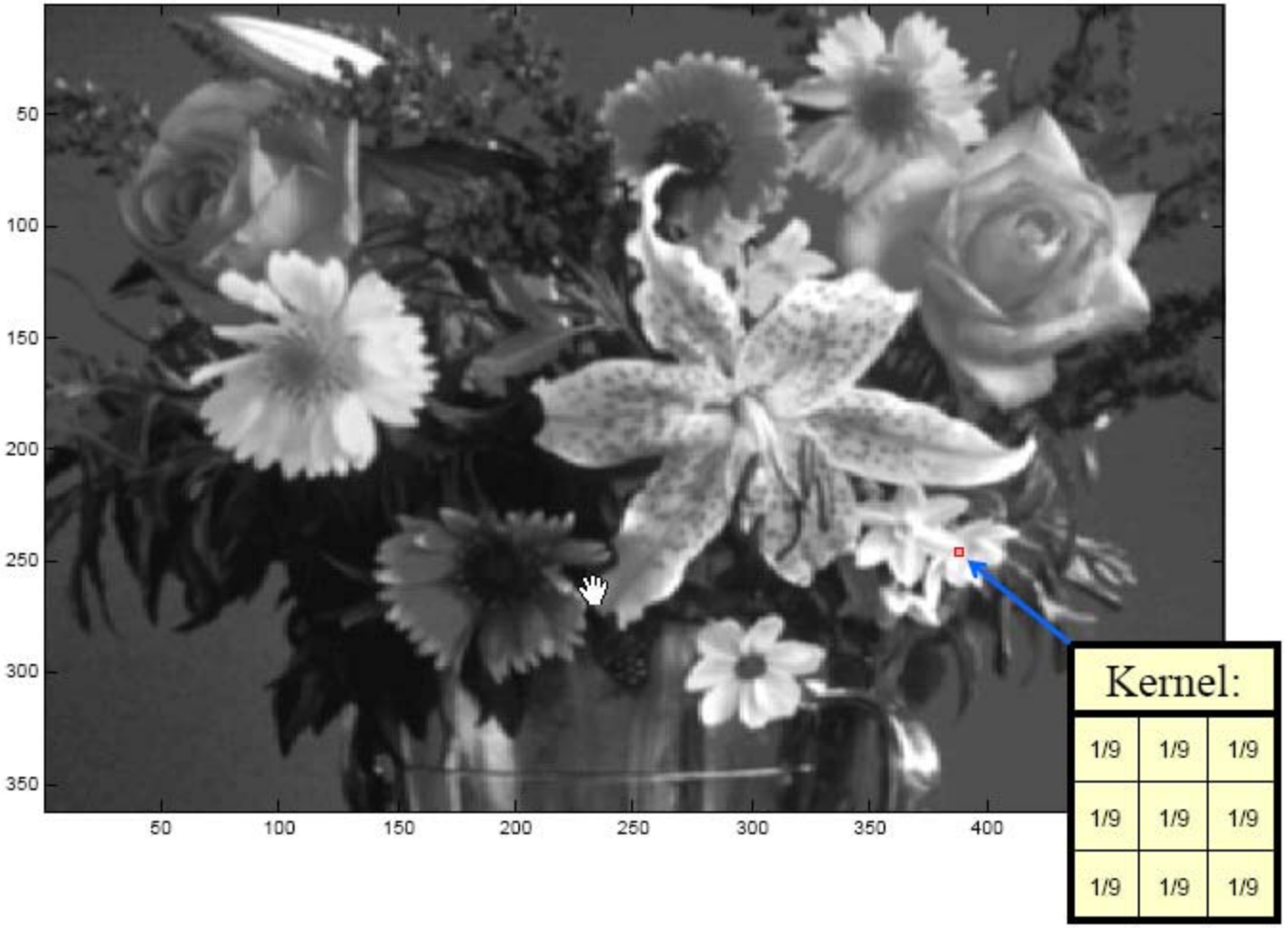
$$f[m,n] = I \otimes g = \sum_{k,l} I[m-k, n-l]\, g[k,l]$$

$$\text{with} \quad \sum_{k,l} g[k,l] = 1$$

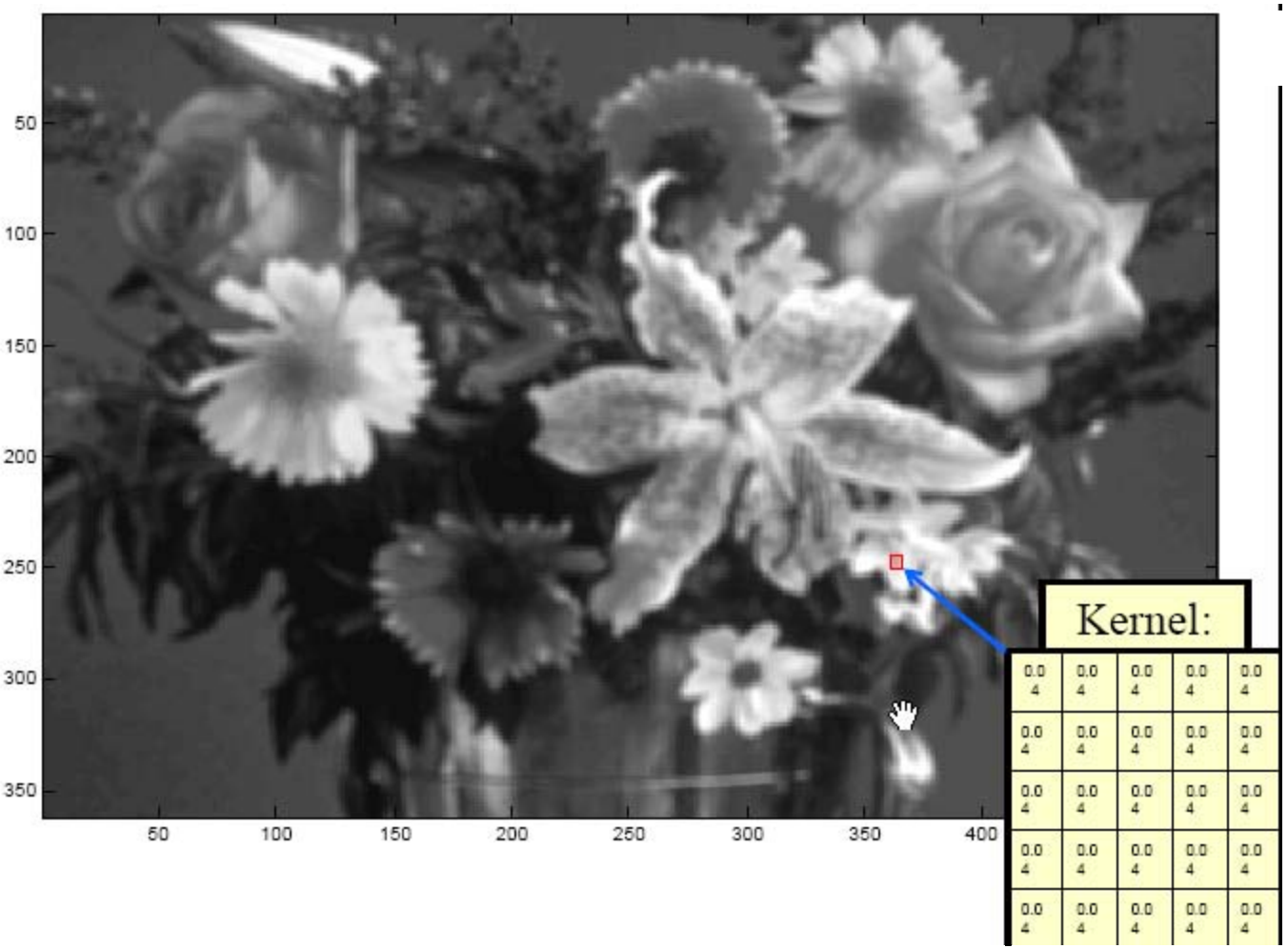
- Example on the web: www.jhu.edu/~signals/convolve
- Matlab function: conv(1D) or conv2(2D)

Original Image

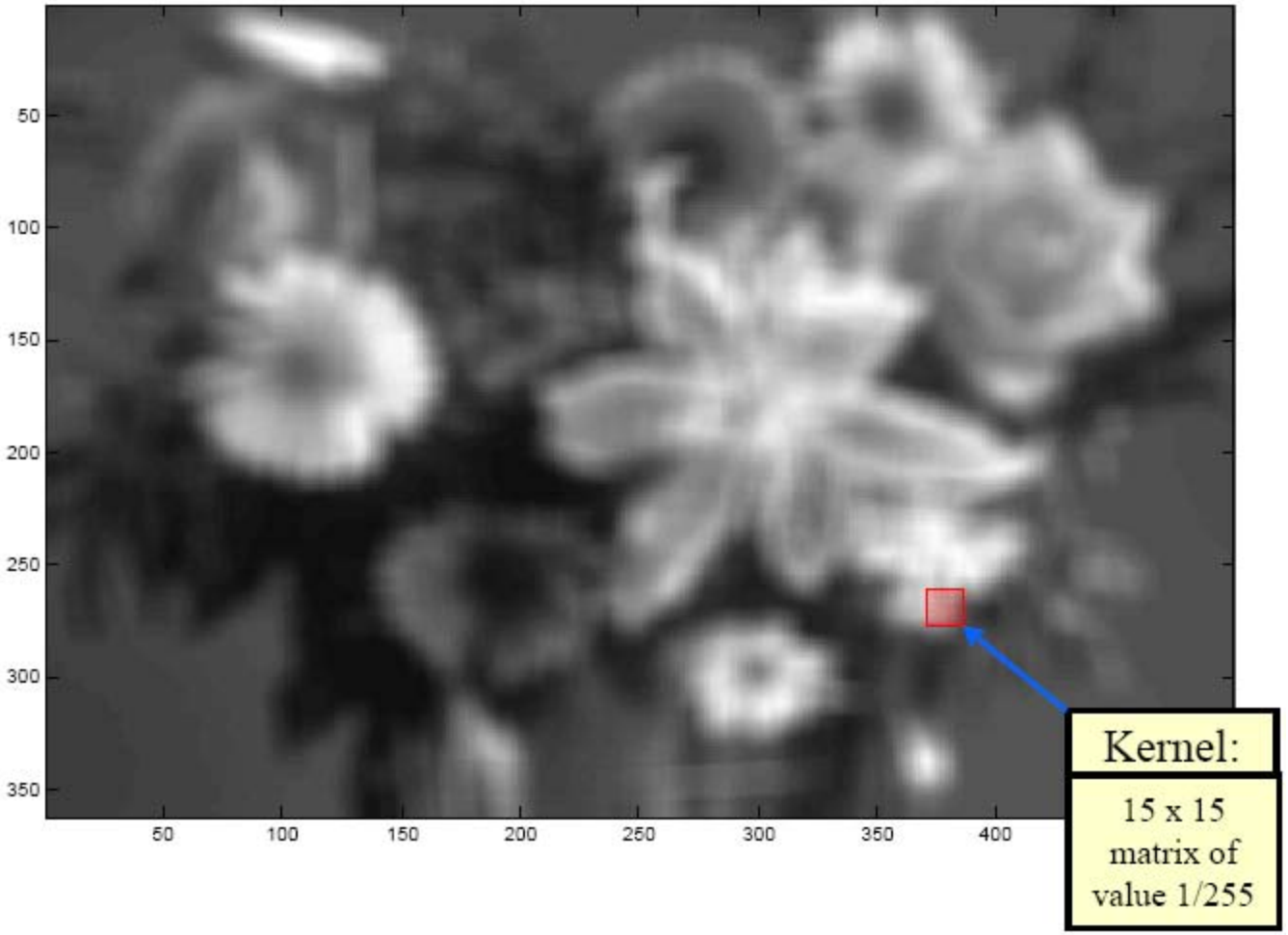
Slight Blurring



More Blurring



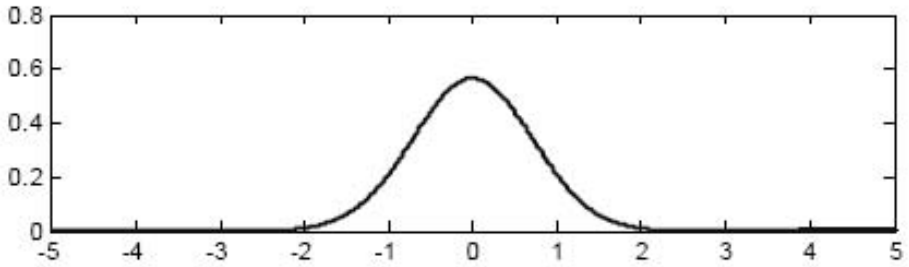
Lots of Blurring



Gaussian

1-D:

$$g(x) = e^{-\frac{x^2}{2\sigma^2}}$$

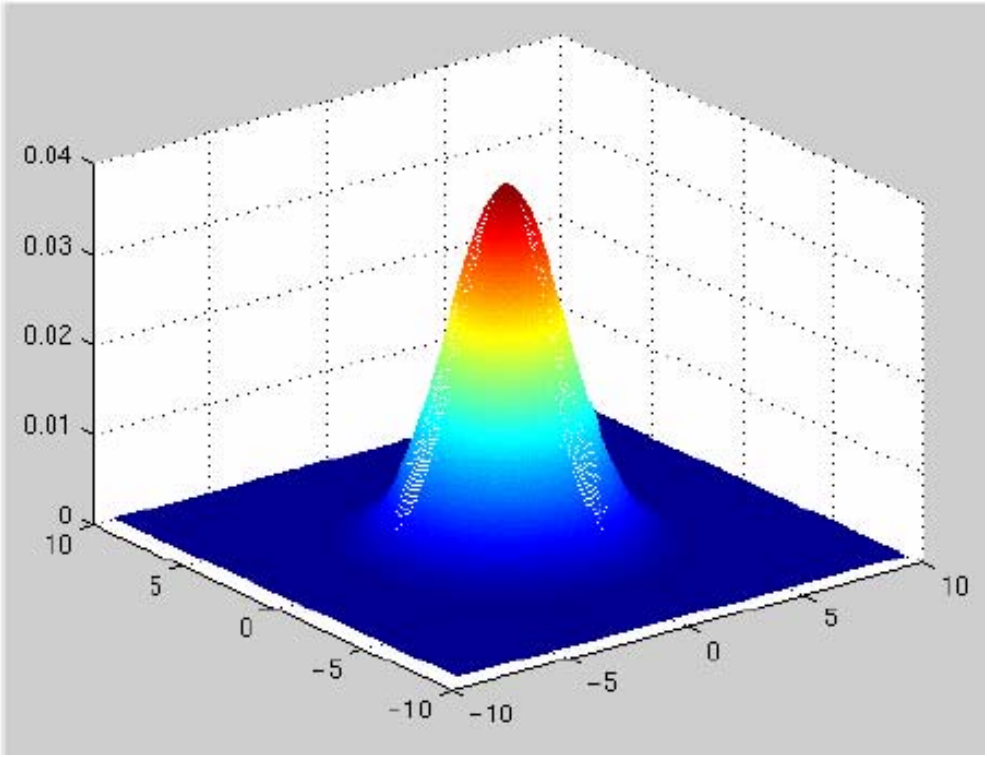


2-D:

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

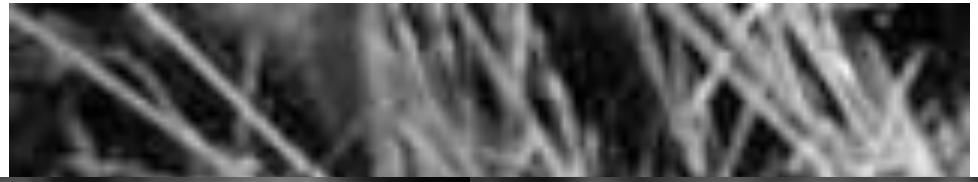
Slight abuse of notations:
We ignore the normalization constant such that

$$\int g(x)dx = 1$$

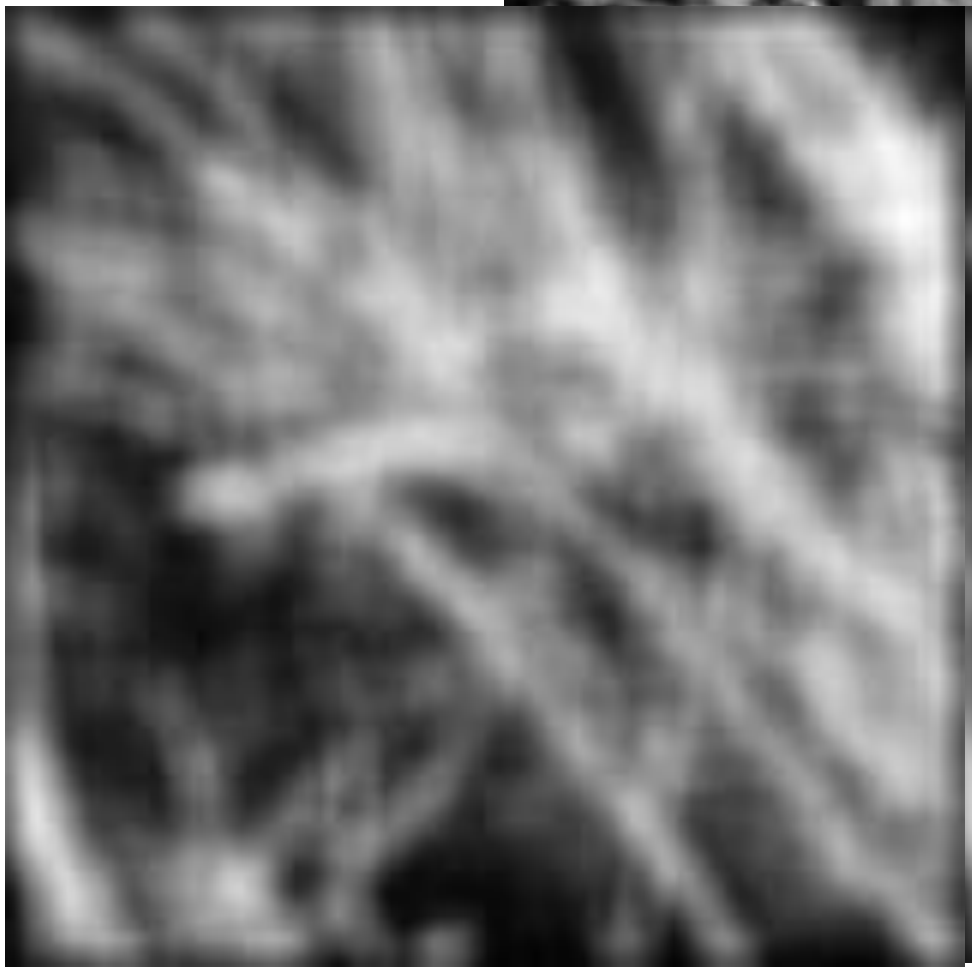


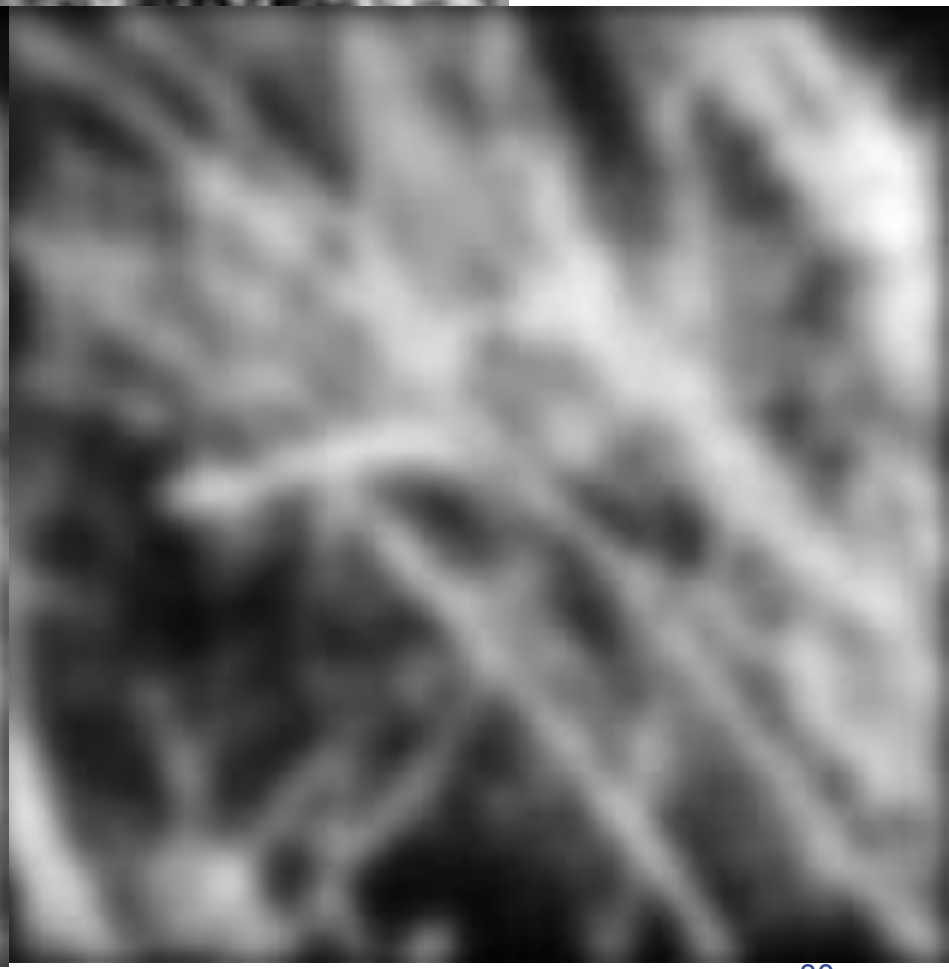
Smoothing with Gaussian

Averaging



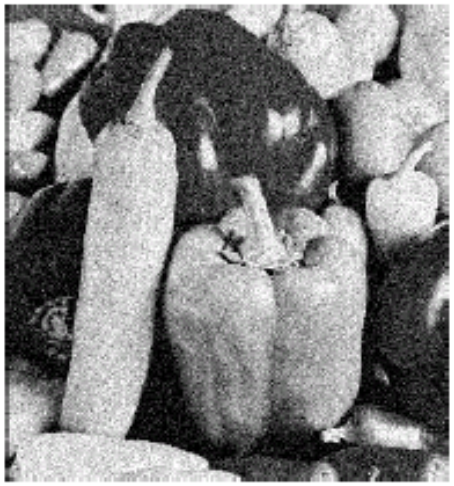
Gaussian

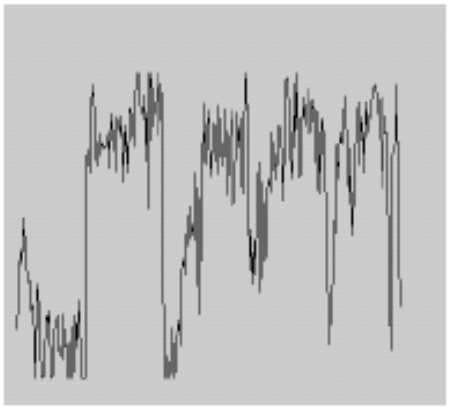


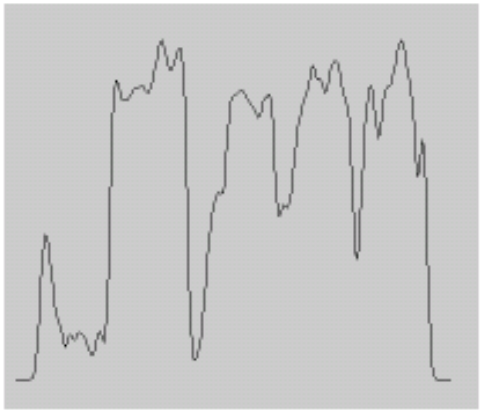


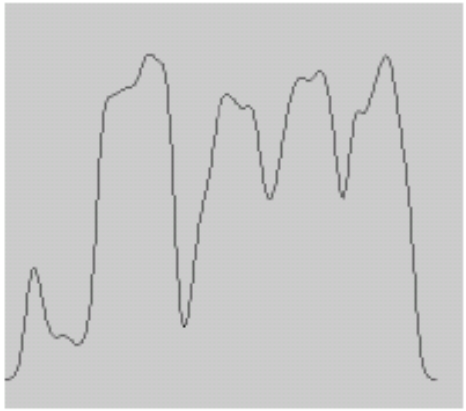
Slide credit: Marc Pollefeys

Gaussian Smoothing to Remove Noise







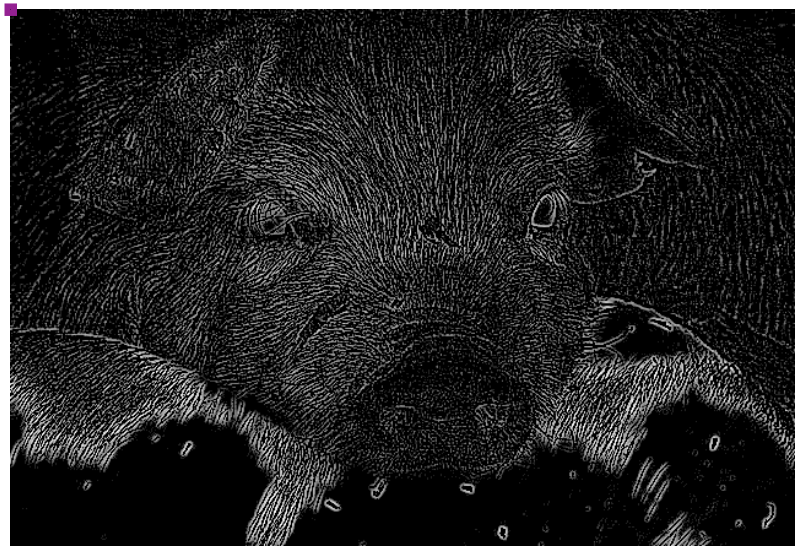


No smoothing $\sigma = 2$ $\sigma = 4$

Edge Detection with Smoothed Images

Image

Blurred Image



Smoothing to Reduce Aliasing

Scaling: 50%

Blurring prior to Scaling: 50%

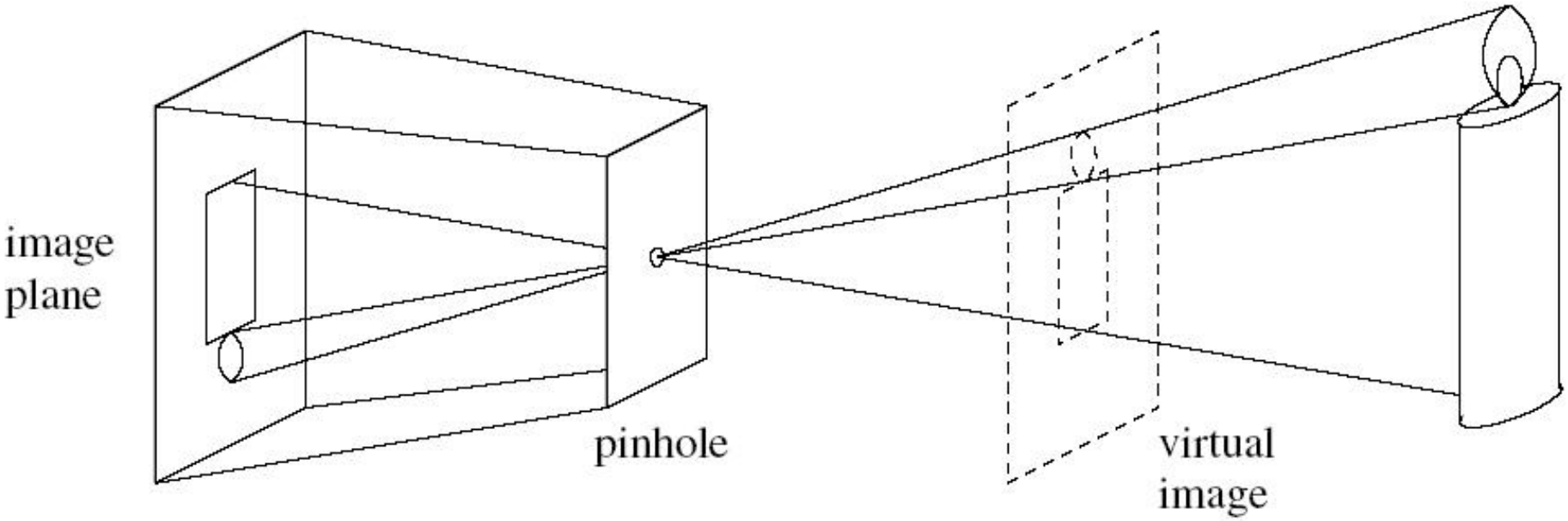
a b c

FIGURE 4.17 Illustration of aliasing on resampled images. (a) A digital image with negligible visual aliasing. (b) Result of resizing the image to 50% of its original size by pixel deletion. Aliasing is clearly visible. (c) Result of blurring the image in (a) with a 3×3 averaging filter prior to resizing. The image is slightly more blurred than (b), but aliasing is not longer objectionable. (Original image courtesy of the Signal Compression Laboratory, University of California, Santa Barbara.)

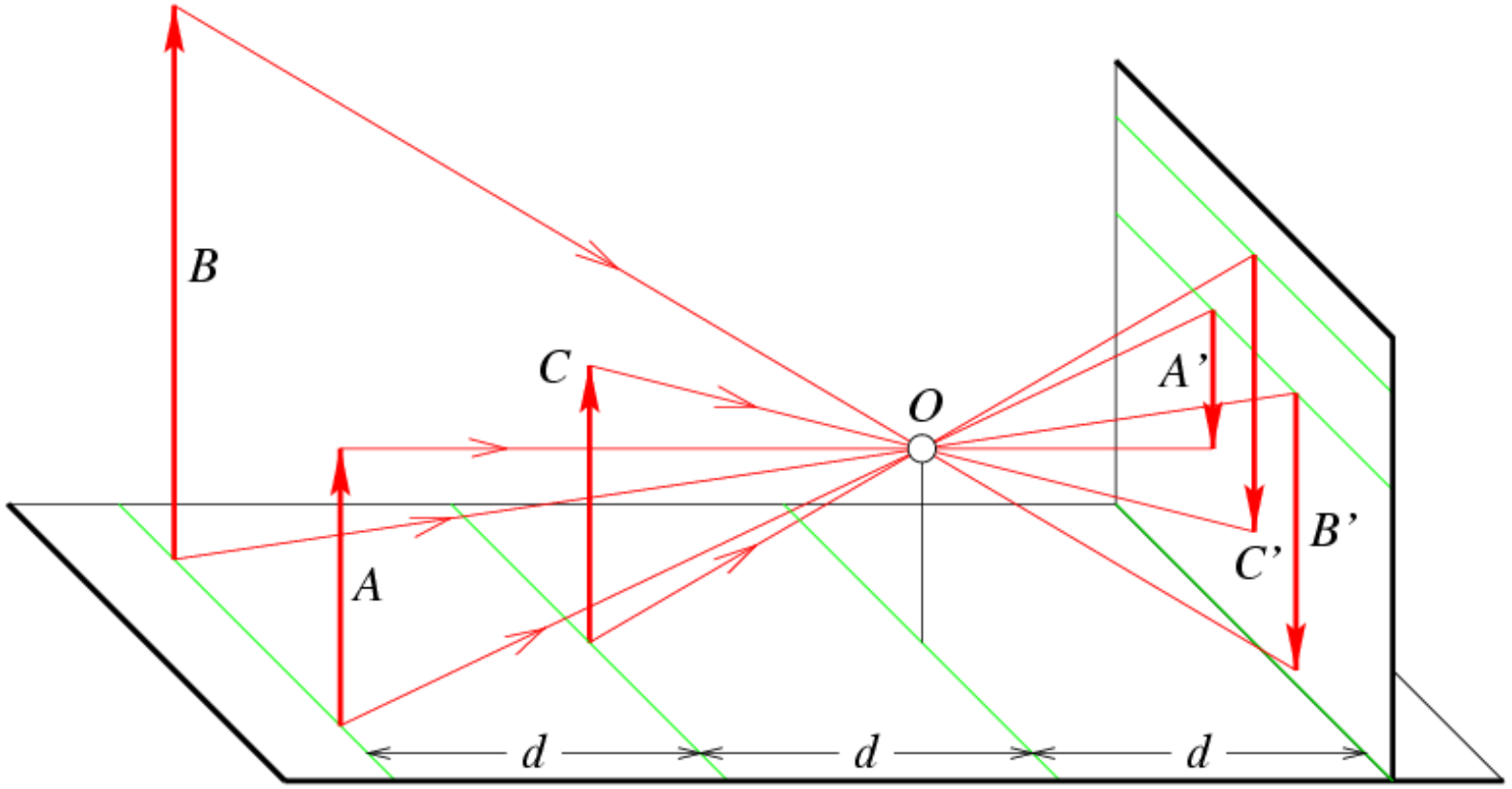
Camera Model & Geometry

- Why needed
- What is needed

Pinhole Imaging Model

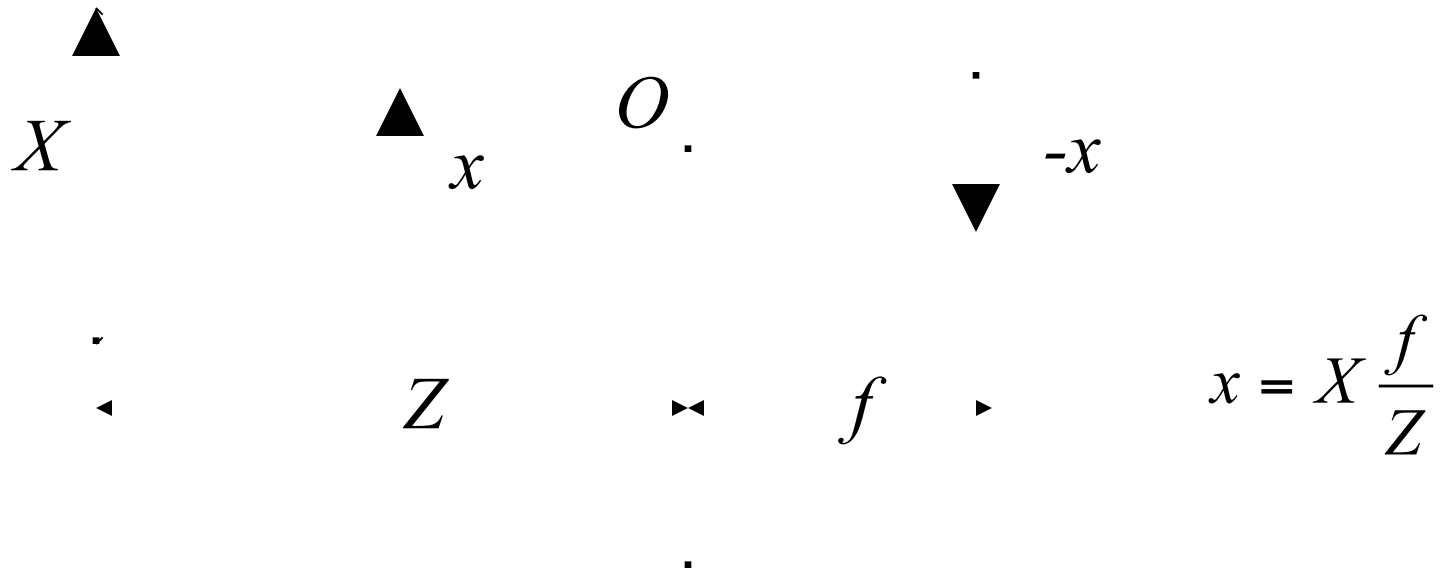


Perspective Projection



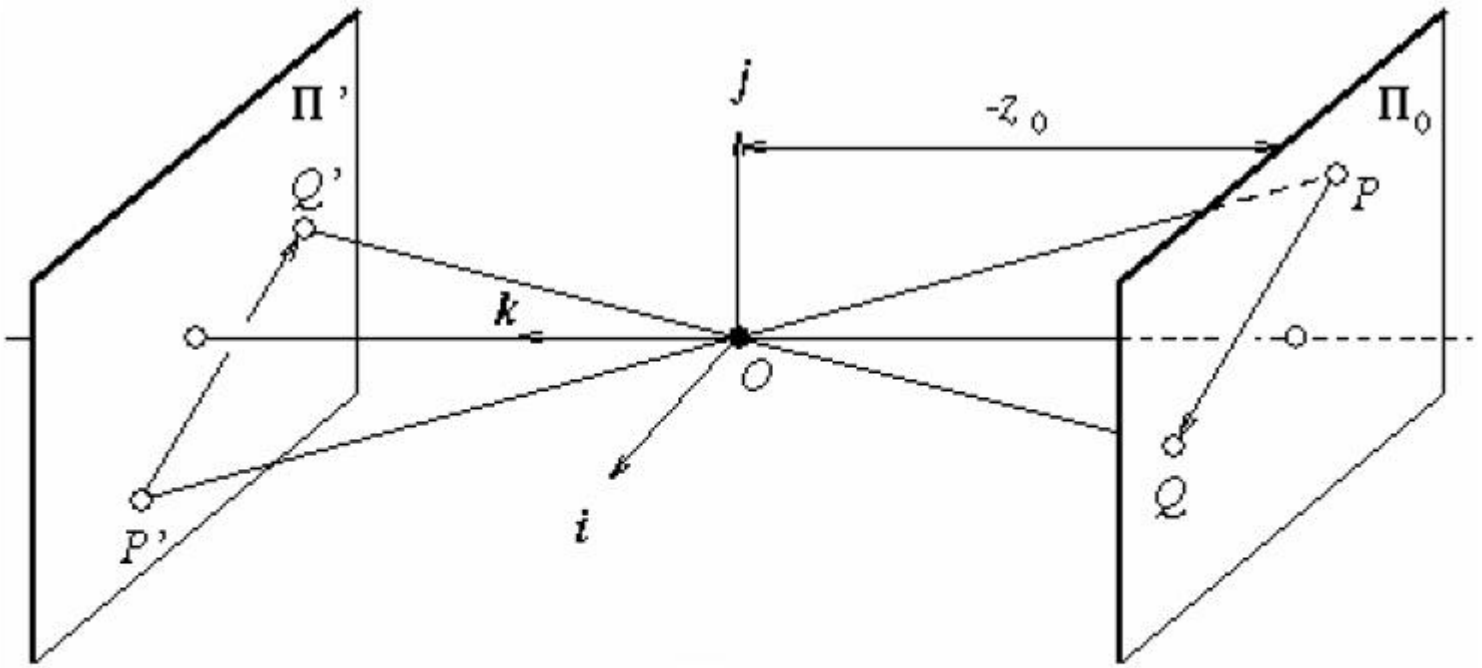
Apparent sizes of objects depend on distance

Perspective Projection



Apparent sizes of objects depend on distance (Z)

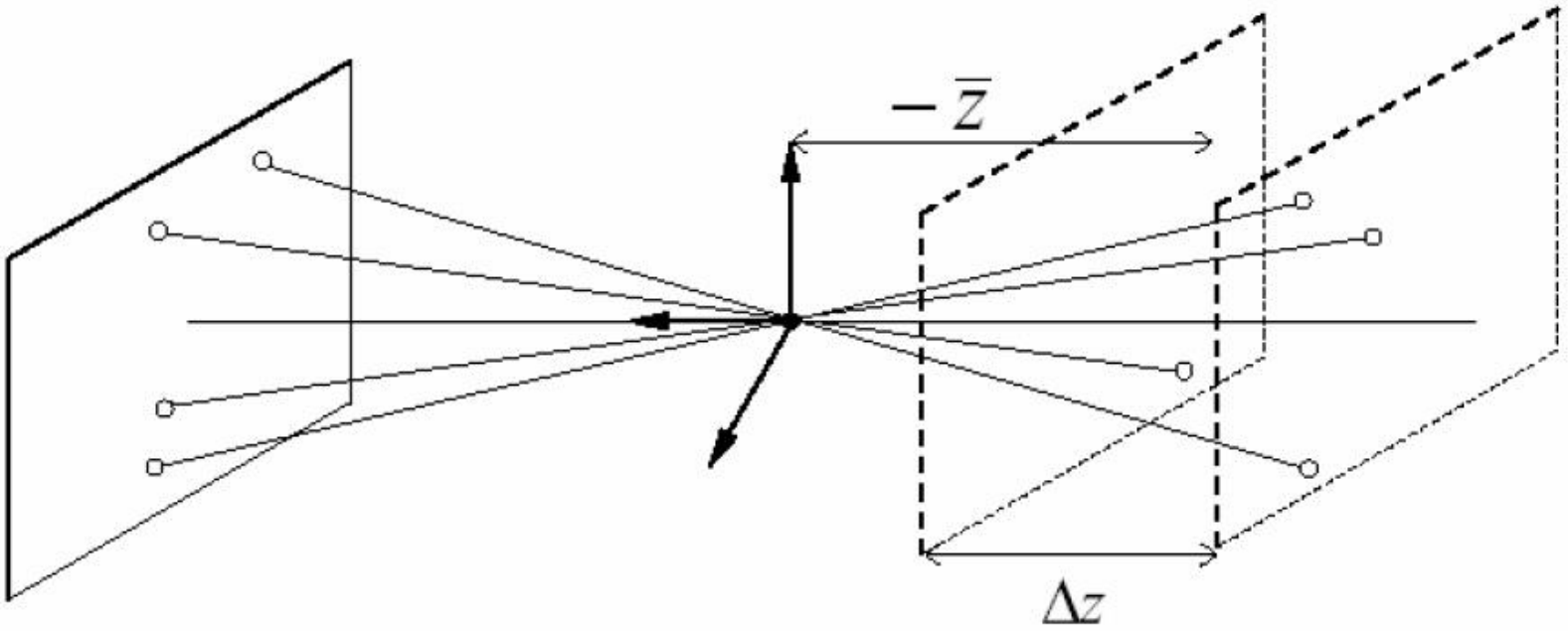
Special Case: Weak Perspective (Affine Projection)



$$x' \approx -mx$$
$$y' \approx -my$$
$$m = -\frac{f}{z_o}$$

If scene points are in a plane, projections are simply magnified by m

Special Case: Weak Perspective (Affine Projection)



If $\Delta z << -\bar{z}$: $x' \approx -mx$, $y' \approx -my$ $\quad m = -\frac{f}{\bar{z}}$

Justified if scene depth is small relative to average distance from camera

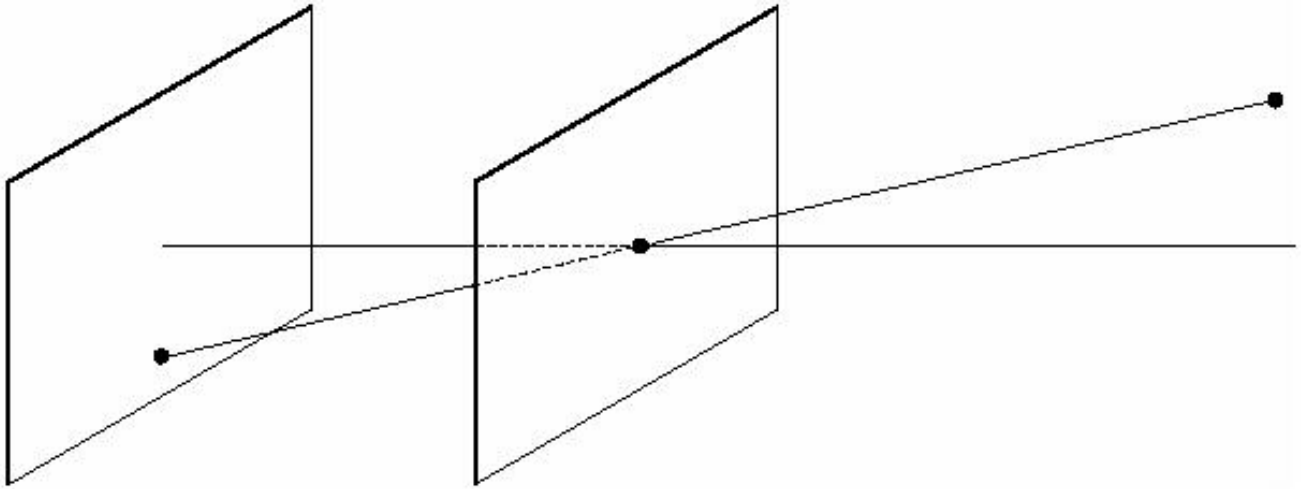
Pictorial Comparison

Weak perspective

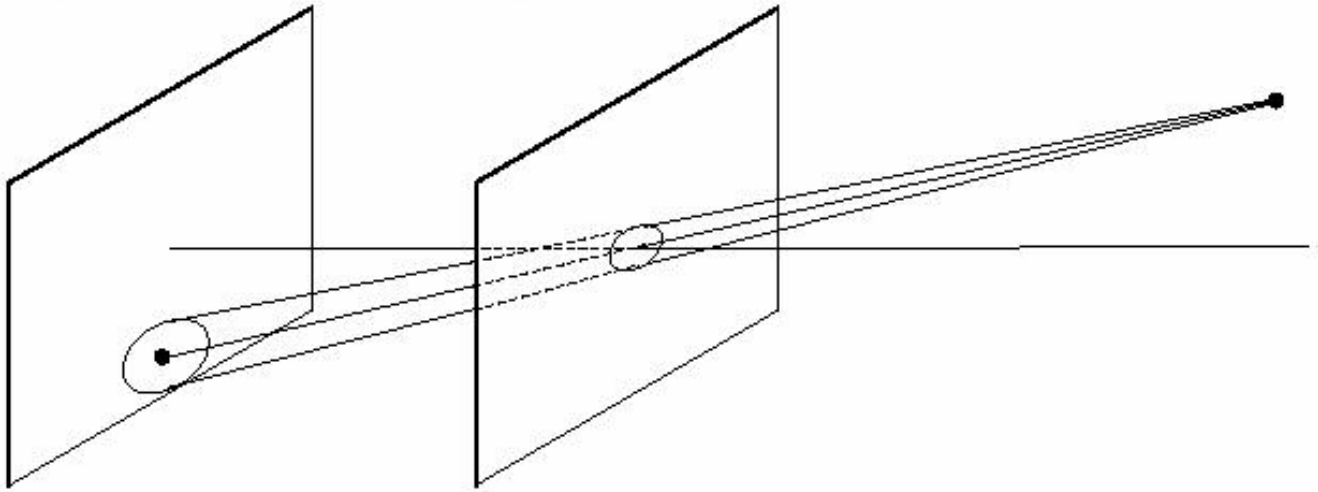
Perspective

Limitations of the pinhole model

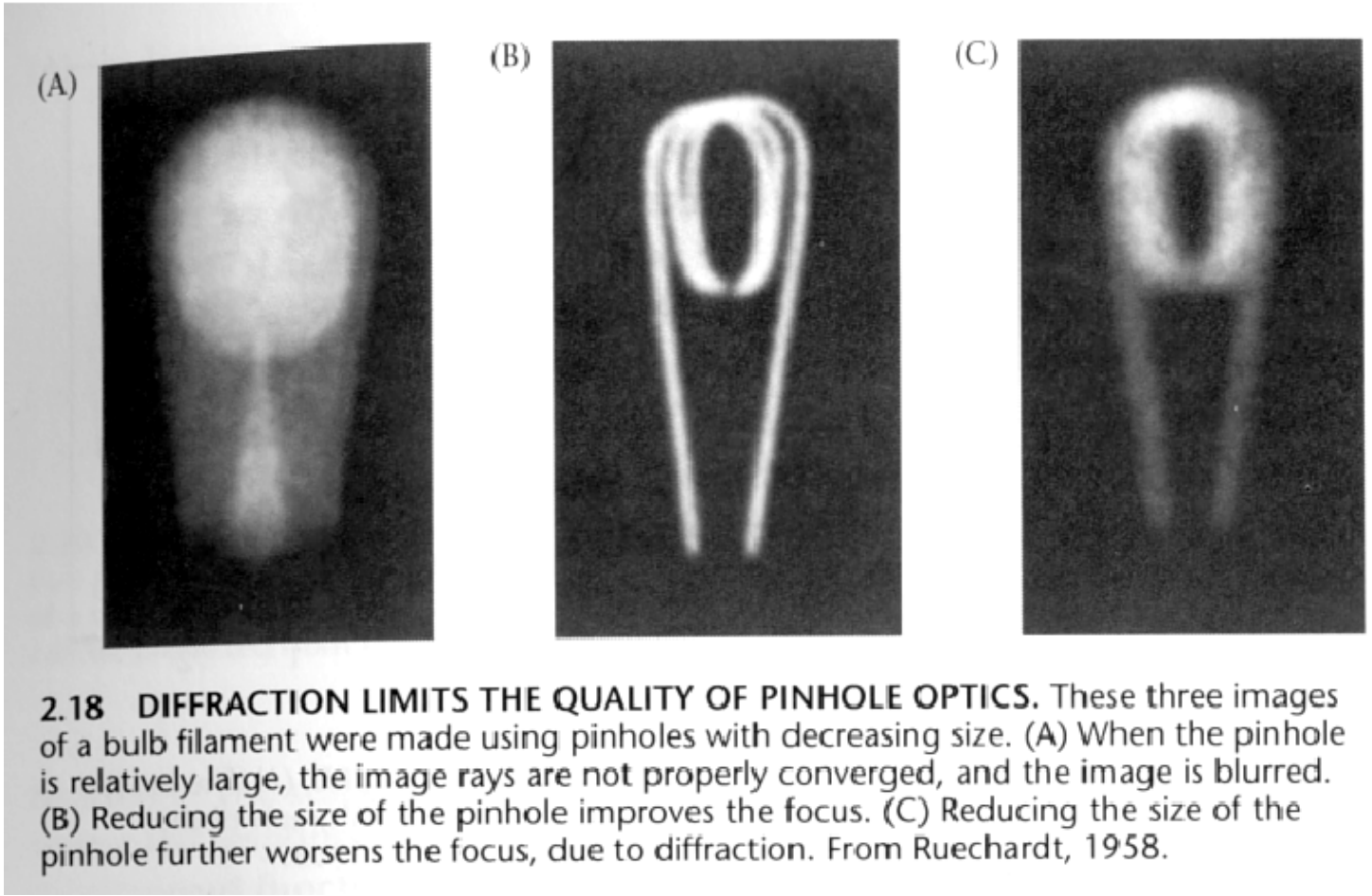
Ideal pinhole:
Single scene point generates single image
but:
Diffraction
Low light level



Finite-size pinhole:
Single scene point generates extended image.
Resulting image is blurry

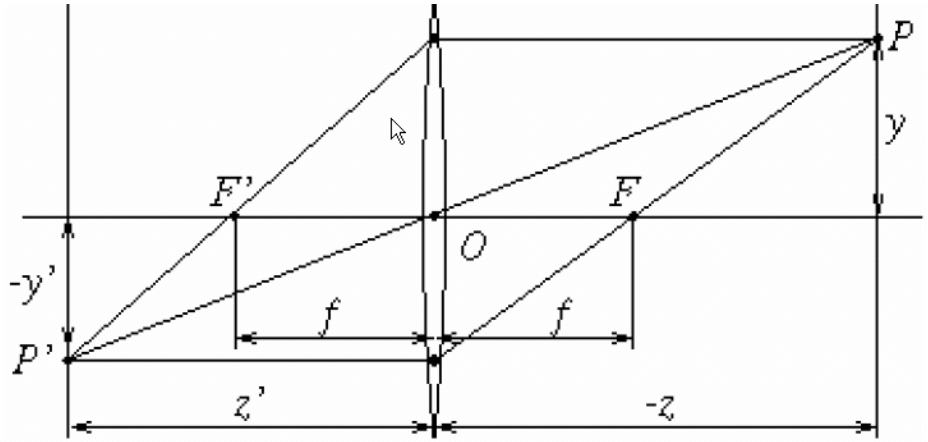


Limitations of the pinhole model



2.18 DIFFRACTION LIMITS THE QUALITY OF PINHOLE OPTICS. These three images of a bulb filament were made using pinholes with decreasing size. (A) When the pinhole is relatively large, the image rays are not properly converged, and the image is blurred. (B) Reducing the size of the pinhole improves the focus. (C) Reducing the size of the pinhole further worsens the focus, due to diffraction. From Ruechardt, 1958.

Thin Lens Model

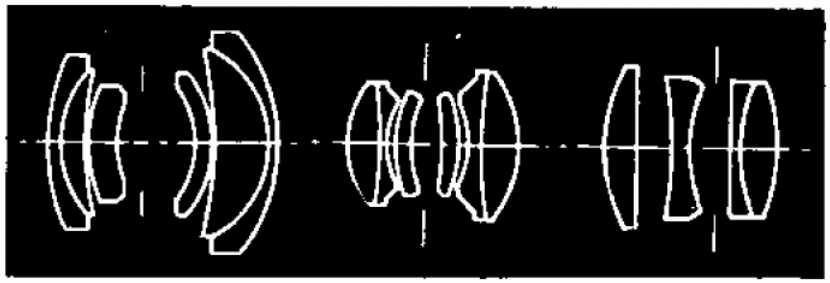


All rays emanating from ***P*** converge to a single point ***P'***

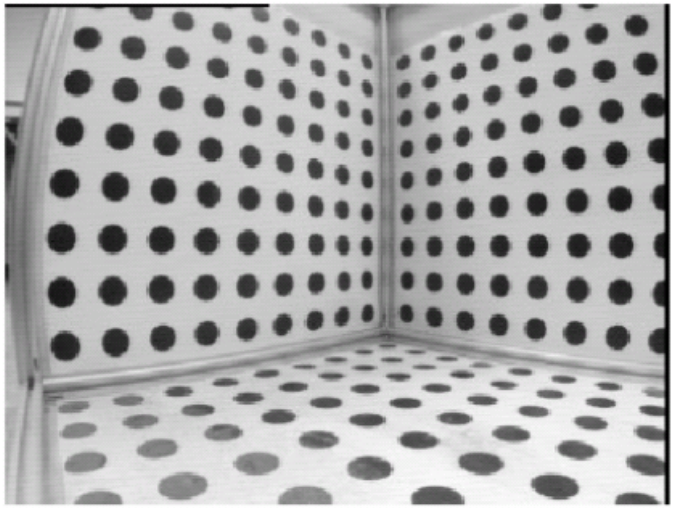
$$\frac{1}{z'} - \frac{1}{z} = \frac{1}{f}$$

Points at infinity are focused on plane $z' = f$

Ideal because: infinite aperture
infinite field of view
infinitely small distance between surfaces



Real Lenses

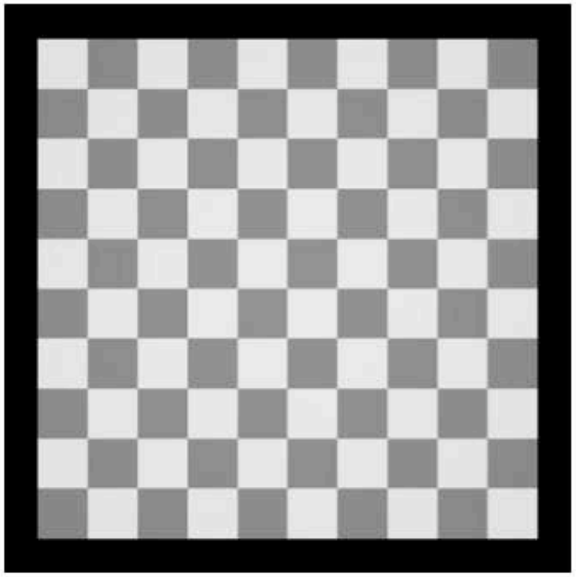


Geometric Distortion

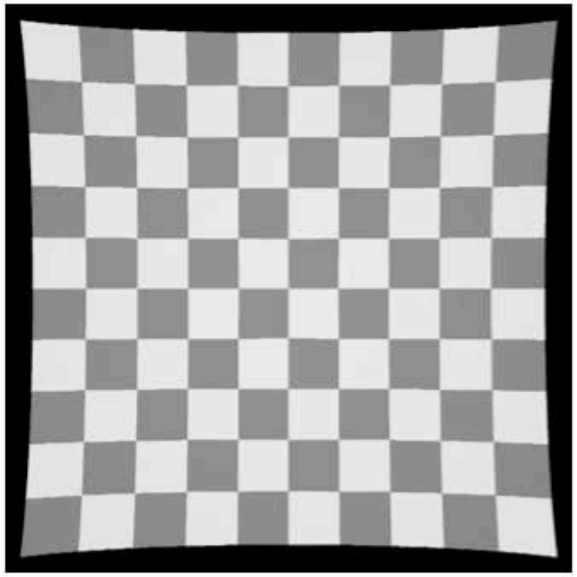
Thin Lens Model

Geometric Distortion

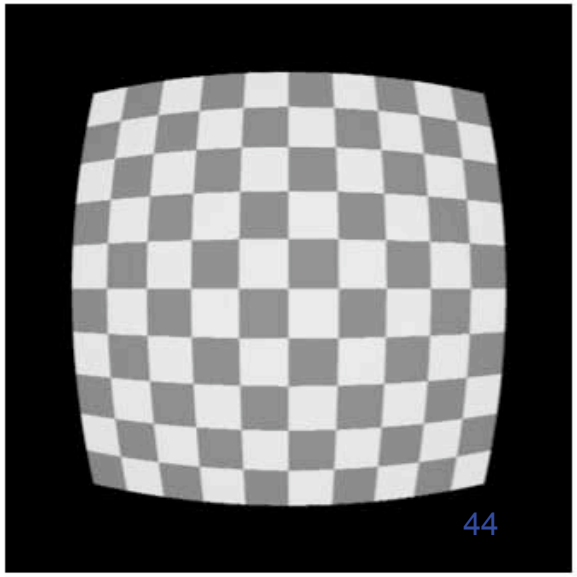
No distortion



Pincushion Distortions



Barrel Distortions



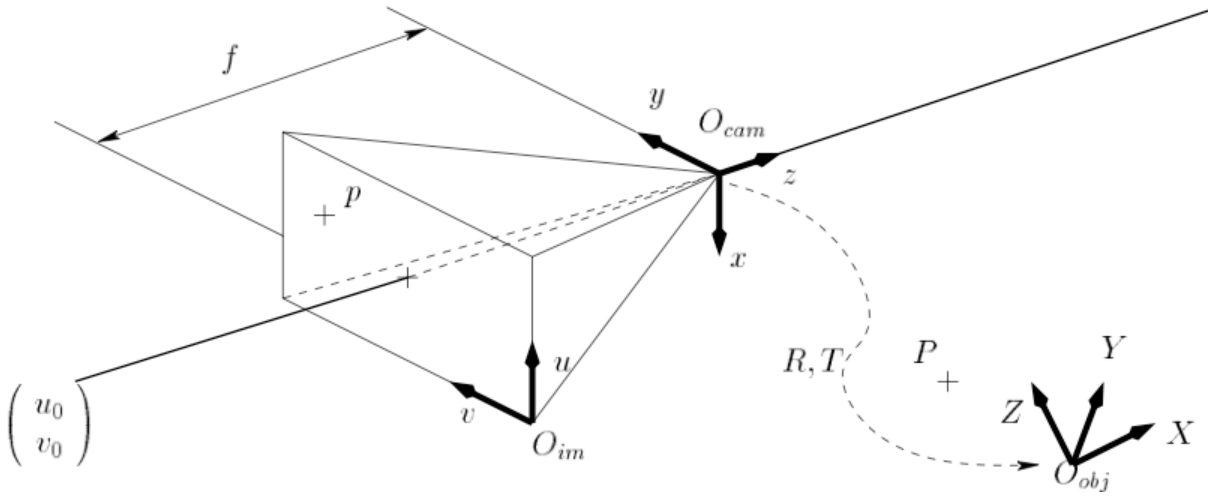
Camera Calibration

- Determine the intrinsic parameters of a camera (with lens)

- What are *Intrinsic Parameters*?
  - Focal Length f
  - Pixel size s_x, s_y (k_u, k_v)
  - Distortion coefficients $k_1, k_2 \ldots$
  - Image center u_0, v_0

- It is important because …

Camera Model



$$\alpha_u = k_u f$$

$$\alpha_v = k_v f$$

Coord. of Pixel in the image

$$\begin{pmatrix} su \\ sv \\ s \end{pmatrix} = \underbrace{\begin{pmatrix} \alpha_u & 0 & u_o & 0 \\ 0 & \alpha_v & v_o & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{\text{I}} \underbrace{\begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix}}_{\text{E}} \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}$$

Coord. Of Point in the "World" frame

Transf. between Camera frame and "World" frame

s = depth of 3D point in the scene (*scale factor*)

[Devy 2003]

Camera Model

$$\begin{pmatrix} su \\ sv \\ s \end{pmatrix} = M \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix} = I\ E \begin{pmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{pmatrix}$$

- M = Matrix of Perspective Projection
- I = Matrix of Intrinsic Parameters
- E = Matrix of Extrinsic Parameters (Rotation + Translation)

[Devy 2003]

Camera Calibration

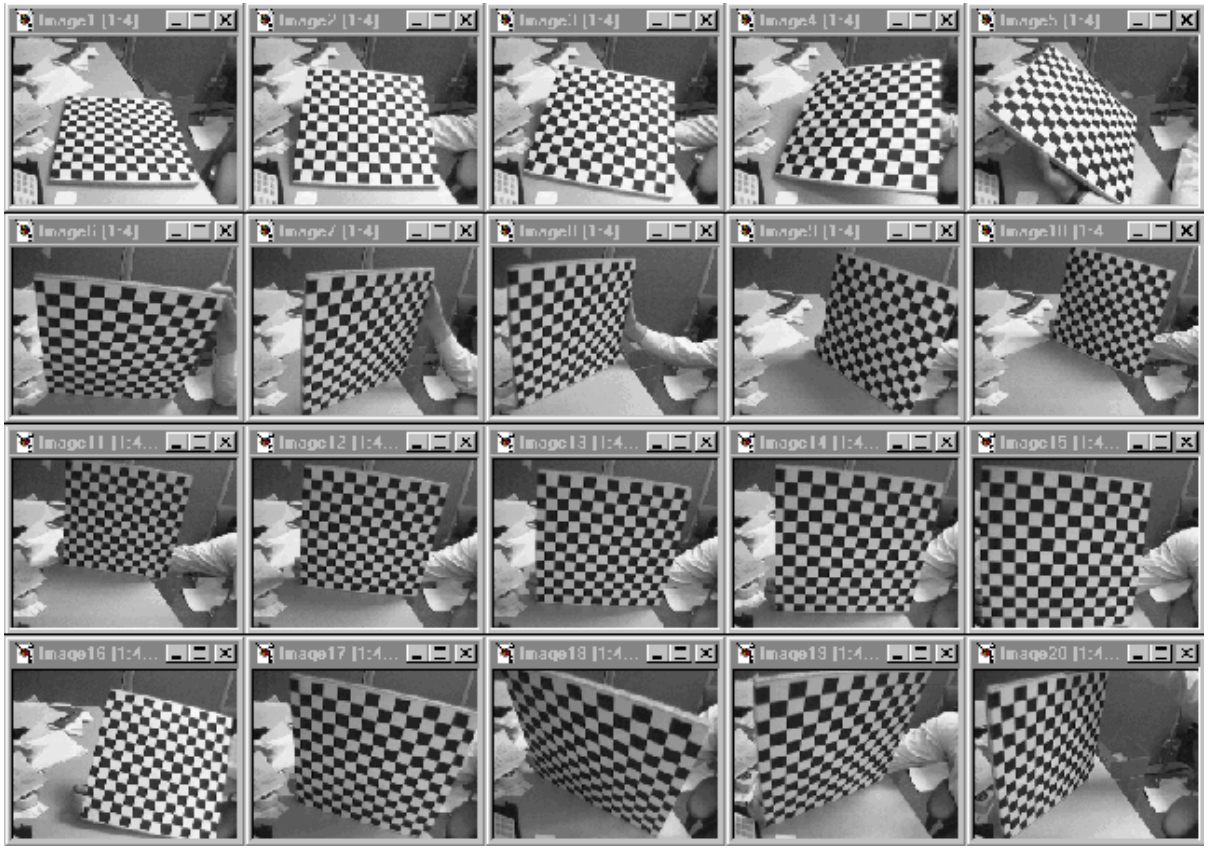
Locations in the world

Camera Model and Parameters

Locations in the image

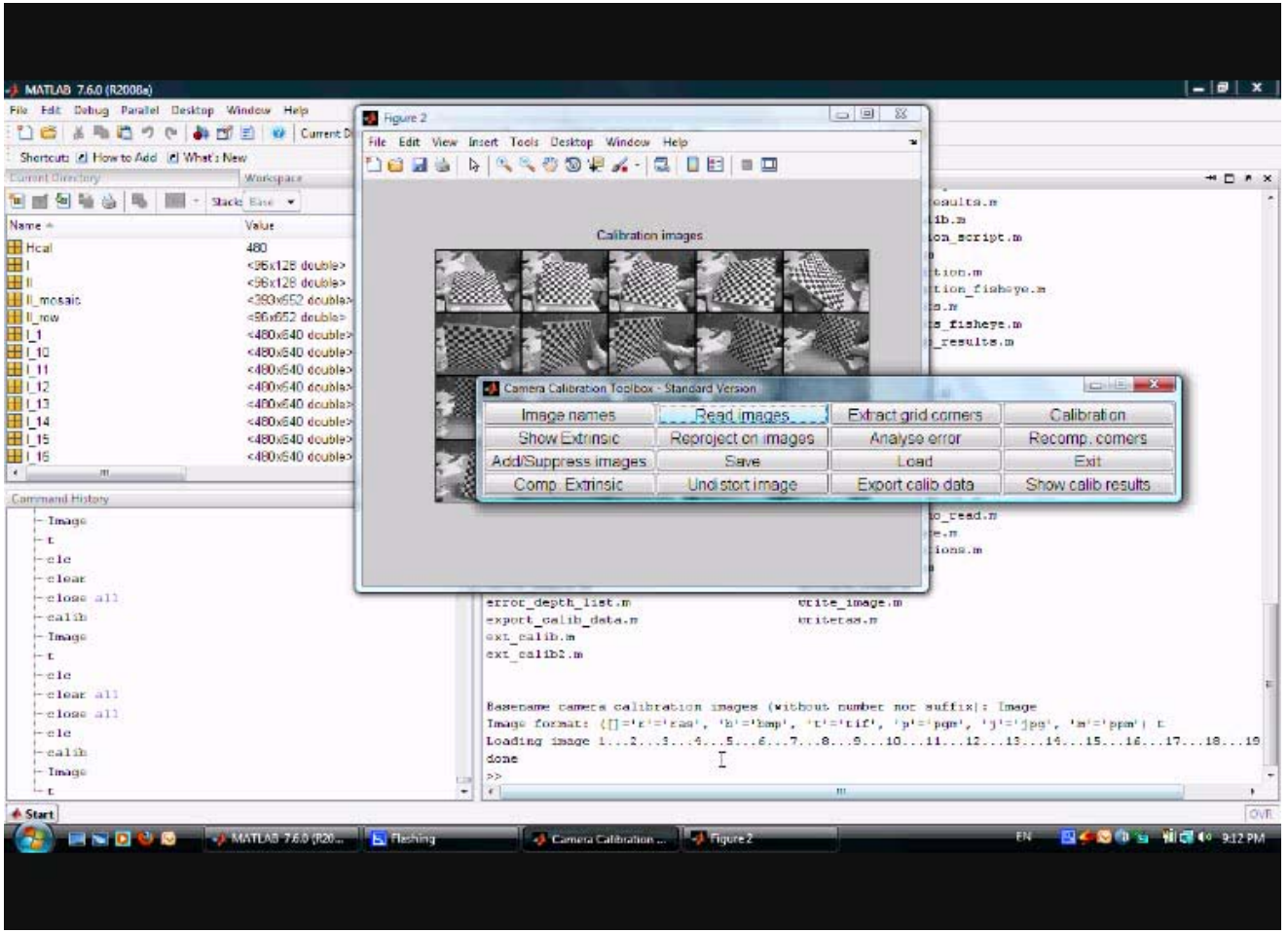
A Designed Object

Example of Calibration Procedure

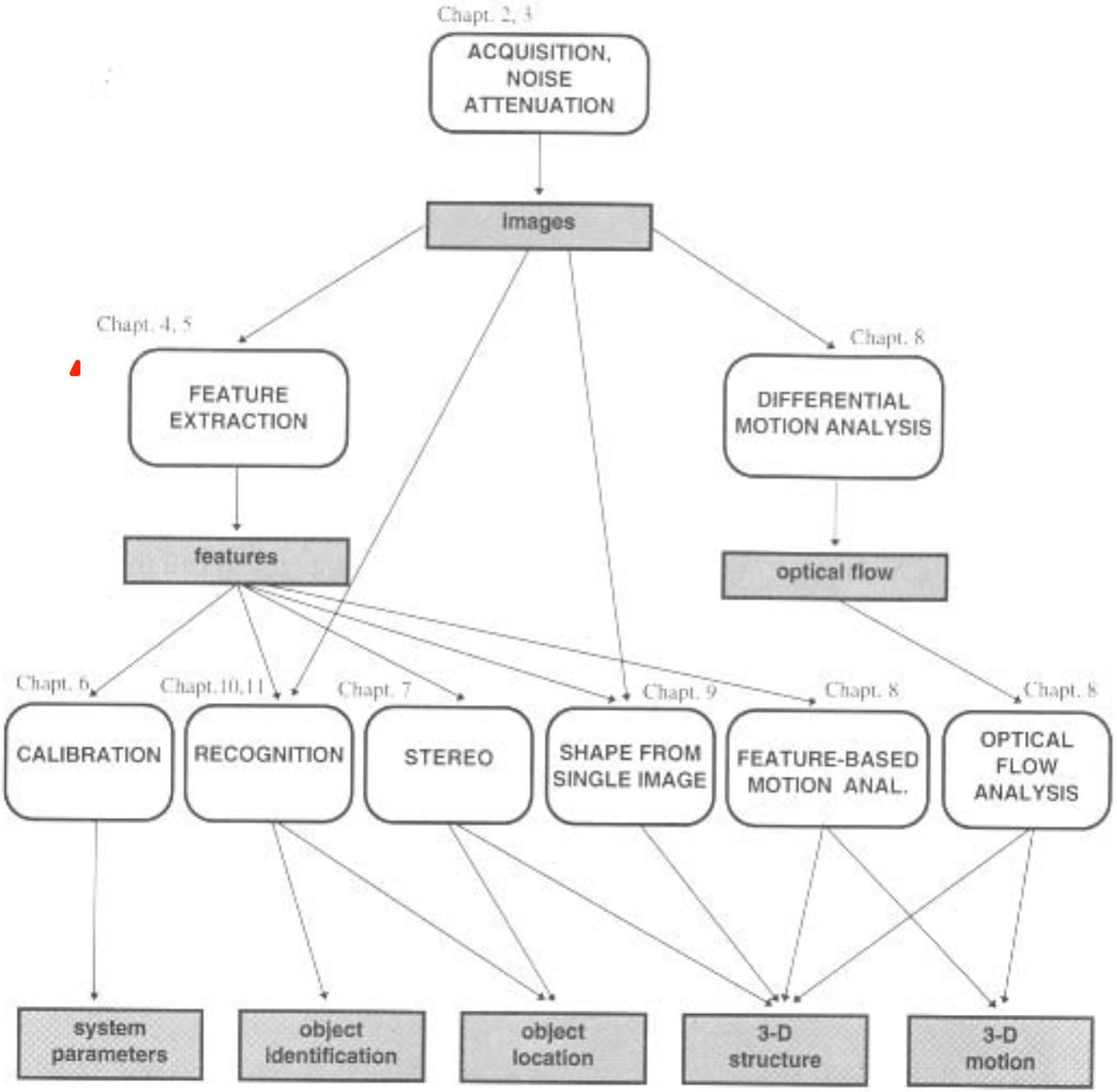


- Camera Calibration Toolbox for Matlab
  - http://www.vision.caltech.edu/bouguetj/calib_doc/index.html

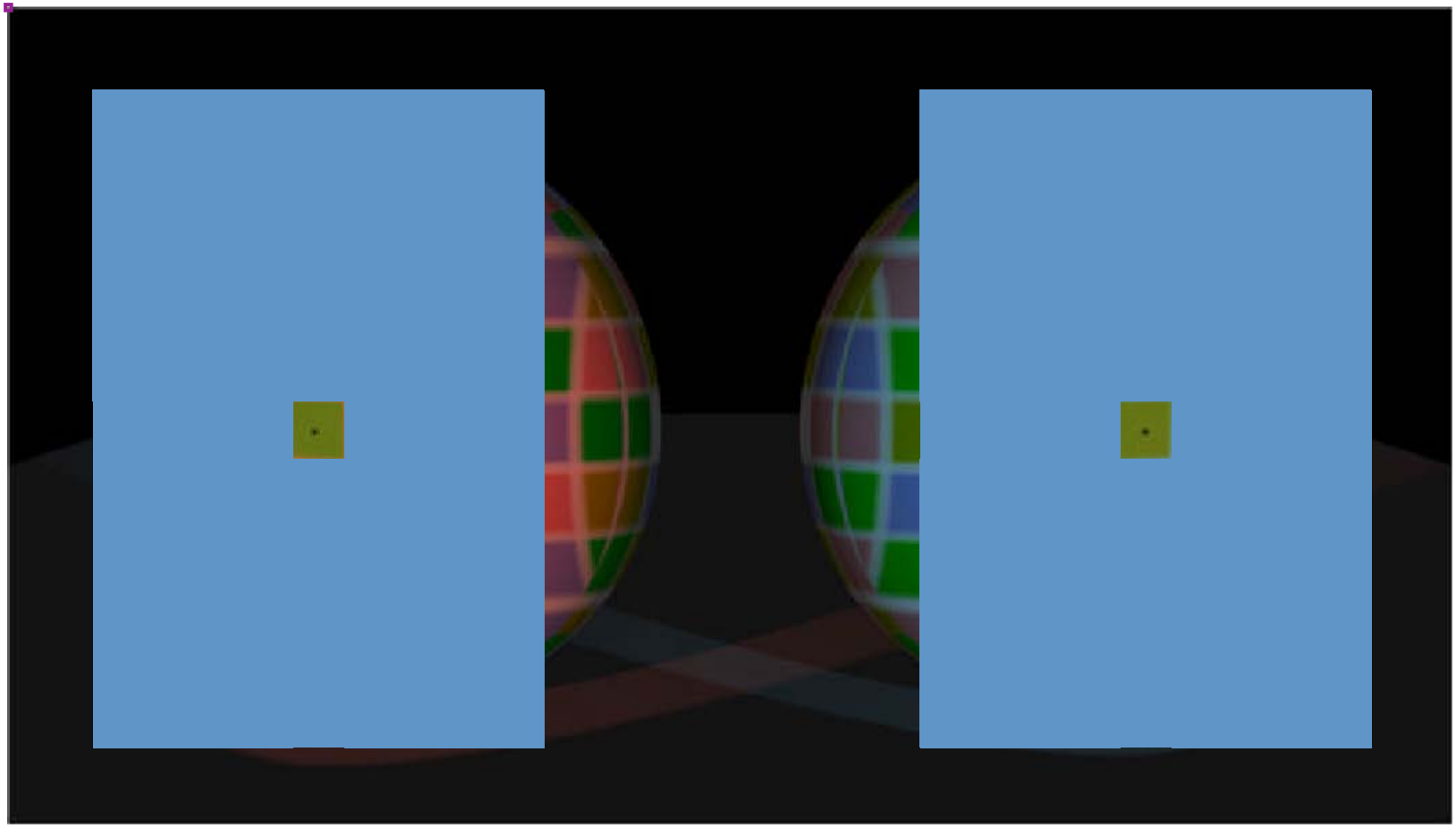
Example of Calibration Procedure



Output: Calib_Results.mat


Chapt. 2, 3
ACQUISITION, NOISE ATTENUATION
images
Chapt. 4, 5
FEATURE EXTRACTION
Chapt. 8
DIFFERENTIAL MOTION ANALYSIS
features
optical flow
Chapt. 6
Chapt. 10,11
Chapt. 7
Chapt. 9
Chapt. 8
Chapt. 8
CALIBRATION
RECOGNITION
STEREO
SHAPE FROM SINGLE IMAGE
FEATURE-BASED MOTION ANAL.
OPTICAL FLOW ANALYSIS
system parameters
object identification
object location
3-D structure
3-D motion

What is a Feature?



- Local, meaningful, detectable parts of the image

Features in Computer Vision

- What is a feature?
  - Location of sudden change
- Why use features?
  - Information content high
  - Invariant to change of view point, illumination
  - Reduces computational burden

Features in Computer Vision

Image 1

Image 2

Feature 1
Feature 2
:
Feature N

Feature 1
Feature 2
:
Feature N

Computer Vision Algorithm

What makes for GOOD features?

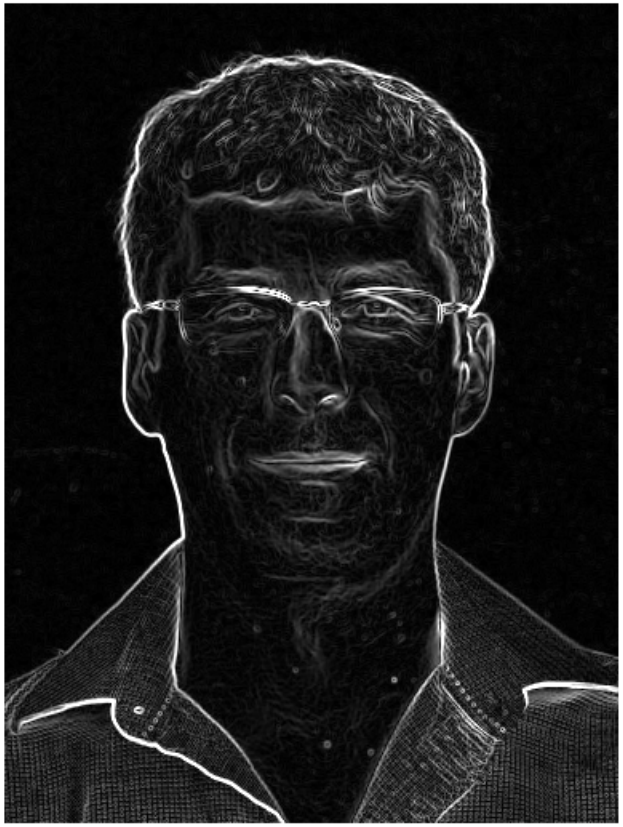
- Invariance
  - View point (scale, orientation, translation)
  - Lighting conditions
  - Object deformations
  - Partial occlusion
- Other Characteristics
  - Uniqueness
  - Sufficiently many
  - Tuned to the task

First Feature: Edge

- Depth discontinuity
- Surface orientation discontinuity
- Reflectance discontinuity (i.e., change in surface material properties)
- Illumination discontinuity (e.g., shadow)

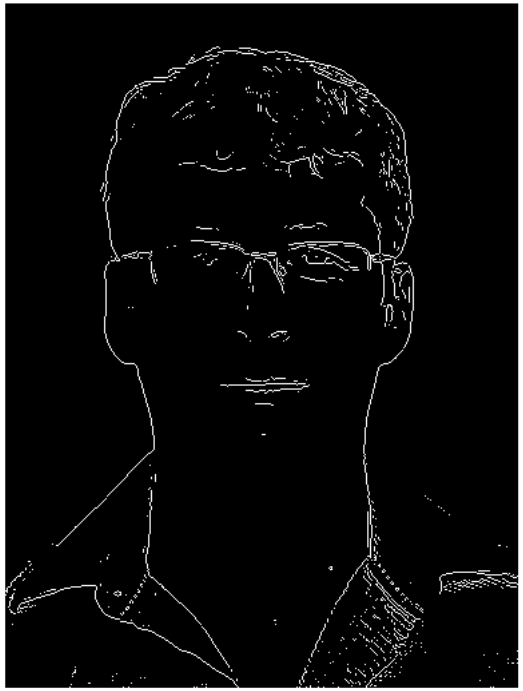
Slide credit: Christopher Rasmussen

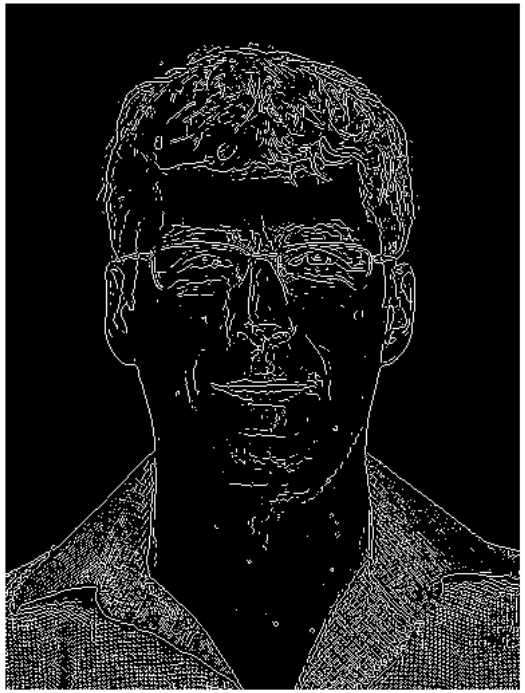
How to find Edges?



Basic Filtering → Edge Detectors → Corner Detectors → SIFT

Sobel Edge Detector





Sobel Operator/Kernel

Sobel operators :

S1 = $\frac{\partial f}{\partial x}$

−1	−2	−1
0	0	0
1	2	1

Horizontal edges

−1	0	1
−2	0	2
−1	0	1

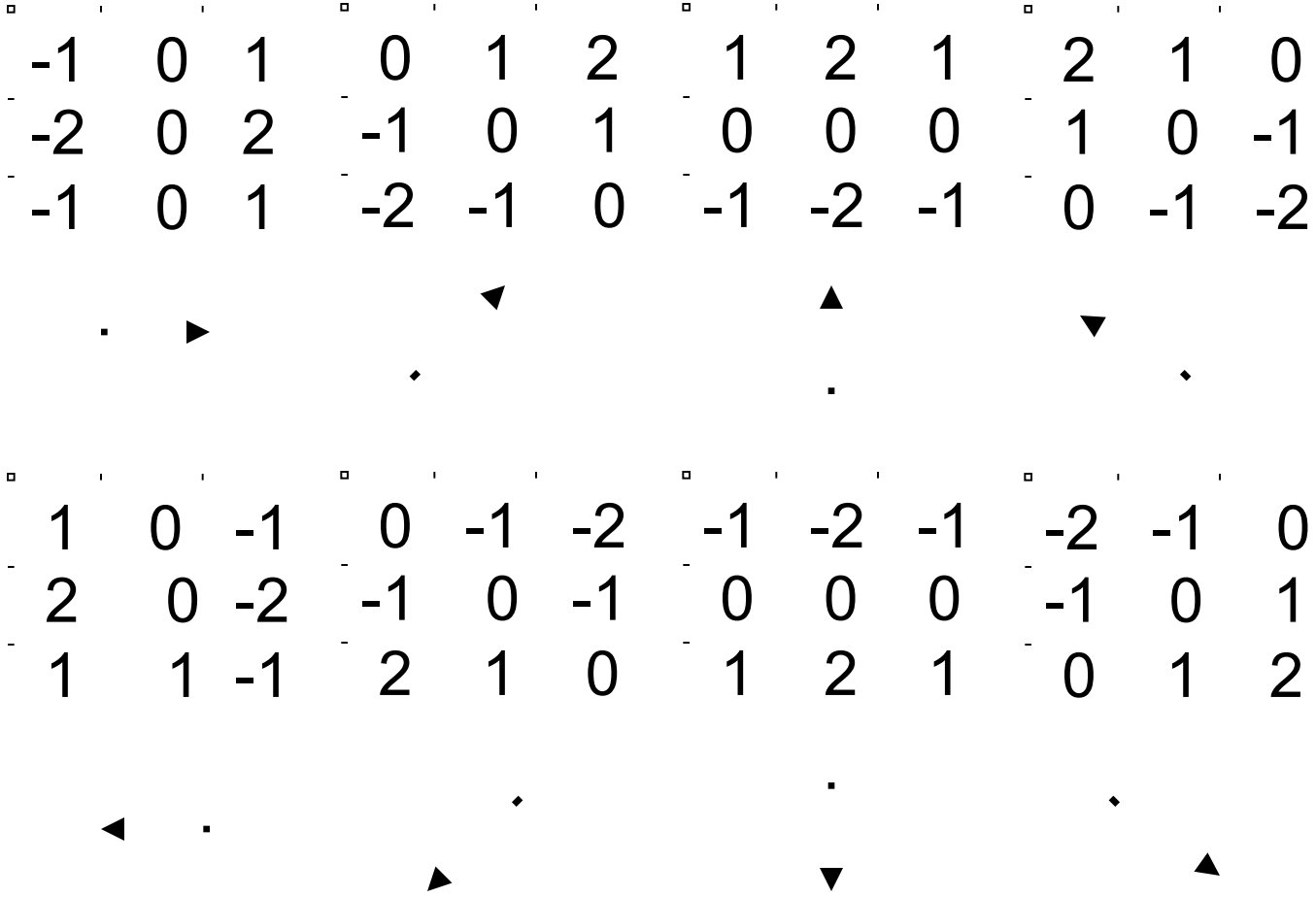
Vertical edges

= S2 $\frac{\partial f}{\partial y}$

Edge Magnitude = $\sqrt{S_1^2 + S_1^2}$

Edge Direction = $\tan^{-1}\left(\frac{S_1}{S_2}\right)$

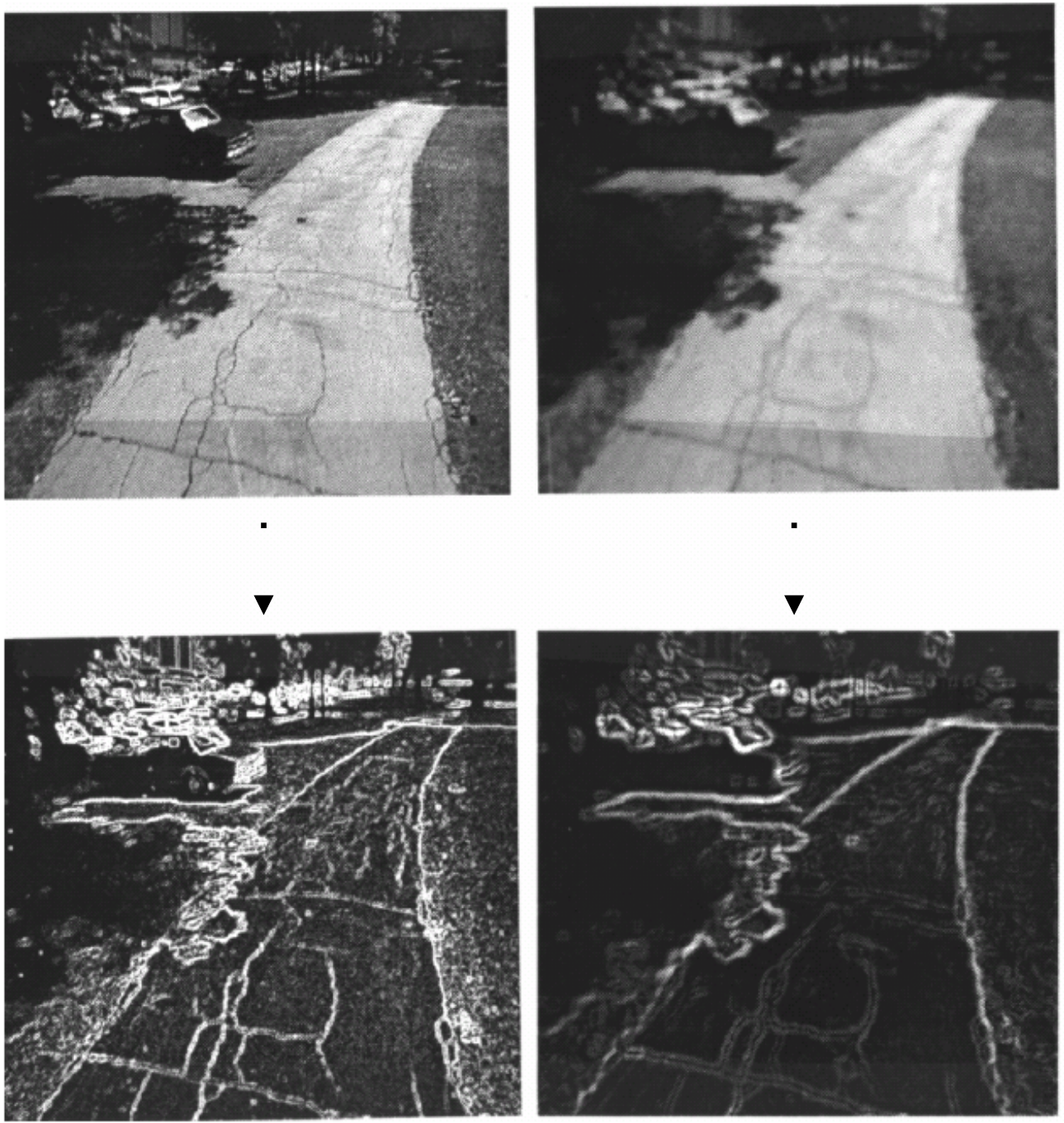
Robinson Compass Masks



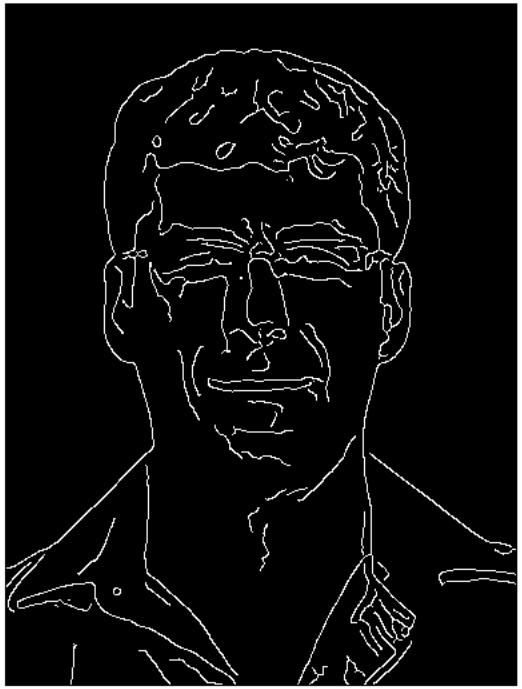
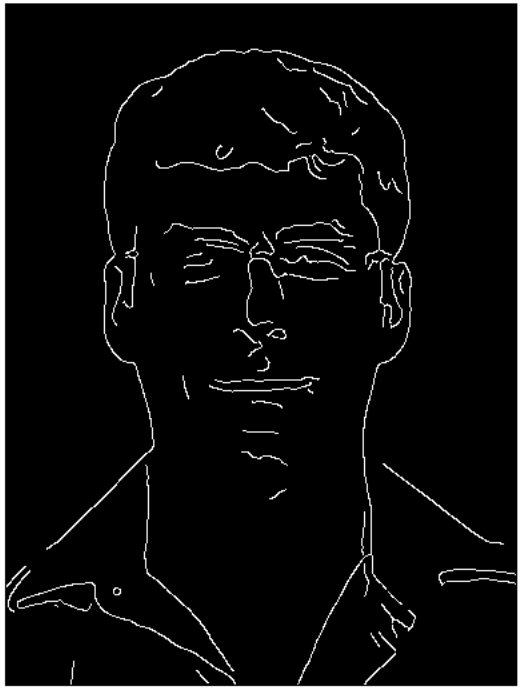
These kernels are Gradient operators

- Edges are discontinuities of intensity in images
- Correspond to local maxima of image gradient
- Gradient computed by convolution

- General principle applies:
  - Slight smoothing: Good localization, poor detection
  - More smoothing: Poor localization, good detection



Canny Edge Detector

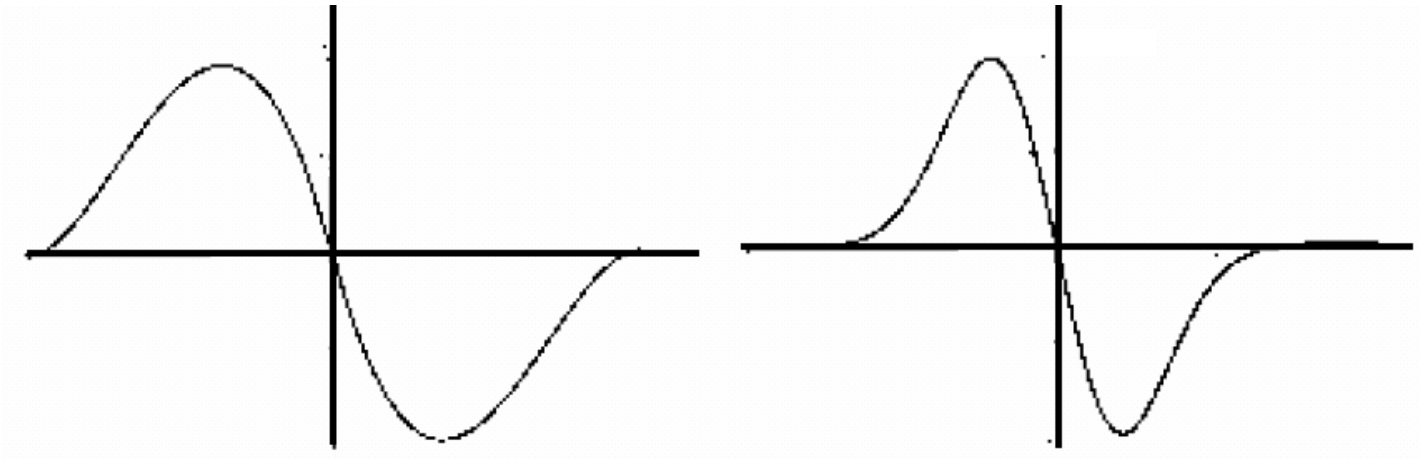
- Canny edge detector shows that Gaussian derivatives yield good compromise between localization and detection.

Canny's Result

- Given a filter *f*, define the two objective functions:
  - *A(f)* large if *f* produces good localization
  - *B(f)* large if *f* produces good detection
- Problem: Find a family of *f* that maximizes the compromise criterion

  $$A(f)B(f)$$

  under the constraint that a single peak is generated by a step edge.
- Solution: Unique solution, a close approximation is the Gaussian derivative.



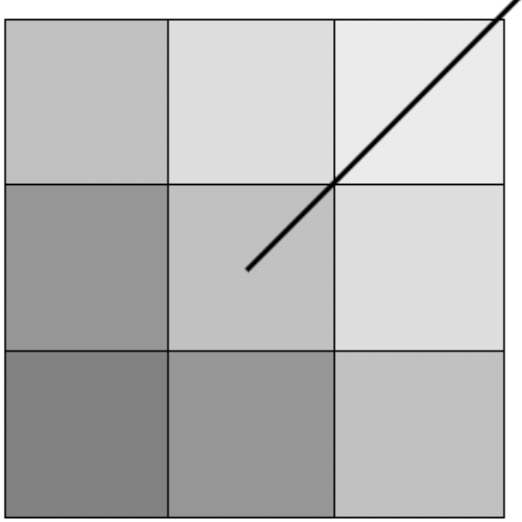
Canny

Derivative of Gaussian

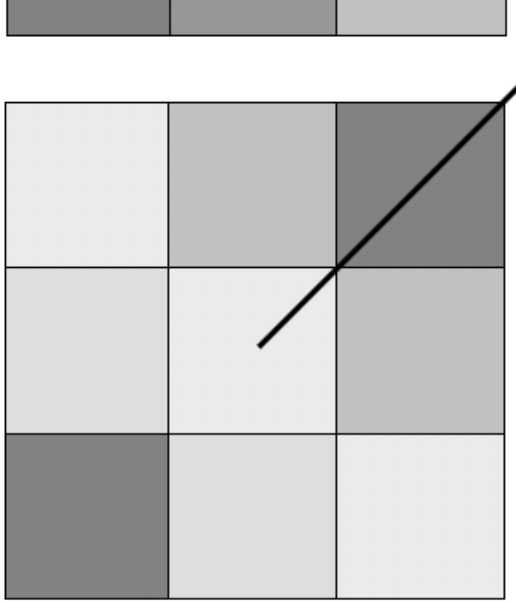
Next Steps

- The gradient magnitude enhances the edges but 2 problems remain:
  - What threshold should we use to retain only the “real” edges?
  - Even if we had a perfect threshold, we would still have poorly localized edges. How to optimally localize contours?
- Solution:
  - Non-local maxima suppression
  - Hysteresis thresholding

Non-Local Maxima Suppression

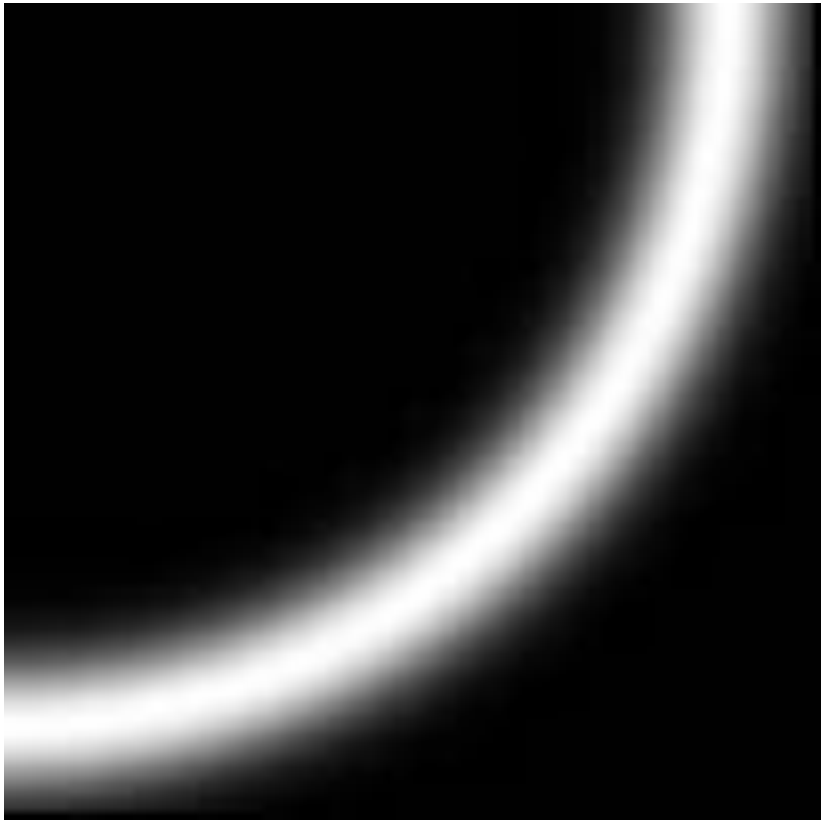


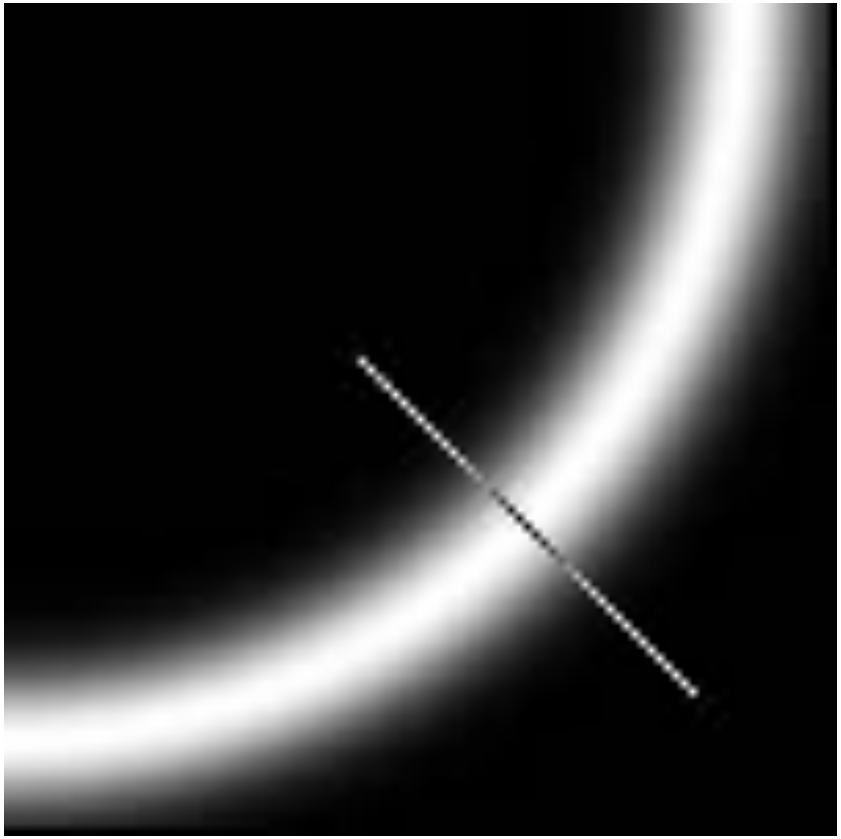
Gradient magnitude at center pixel
is lower than the gradient magnitude
of a neighbor *in the direction of the gradient*
→ Discard center pixel (set magnitude to 0)



Gradient magnitude at center pixel
is greater than gradient magnitude
of all the neighbors *in the direction of the gradient*
→ Keep center pixel unchanged

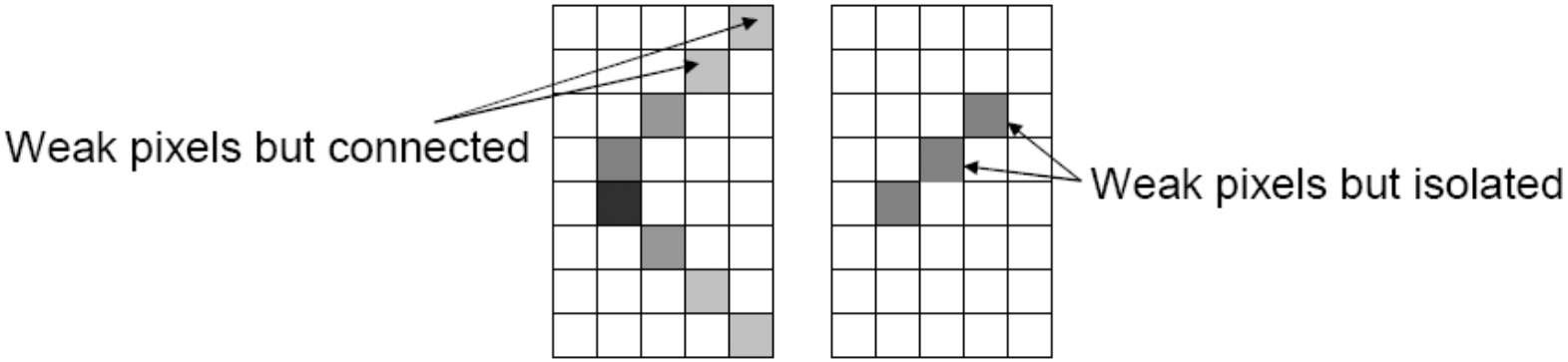
Non-Local Maxima Suppression

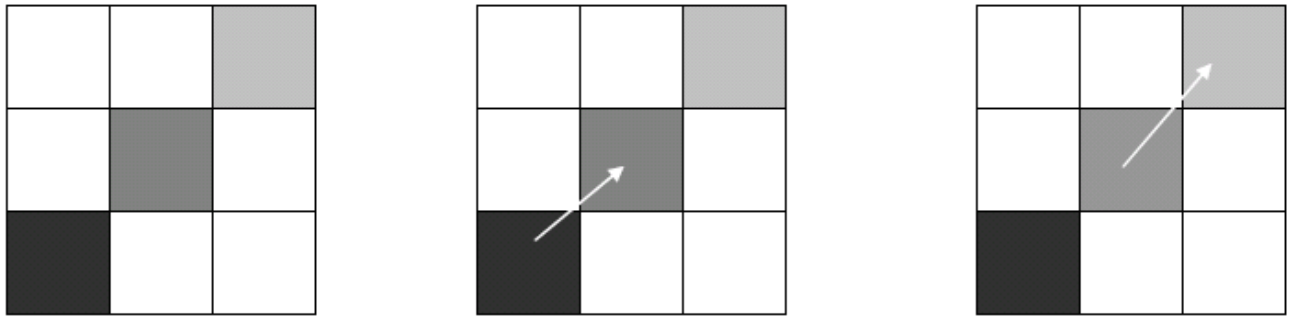




Select the single maximum point across the width of an edge.

Hysteresis Thresholding



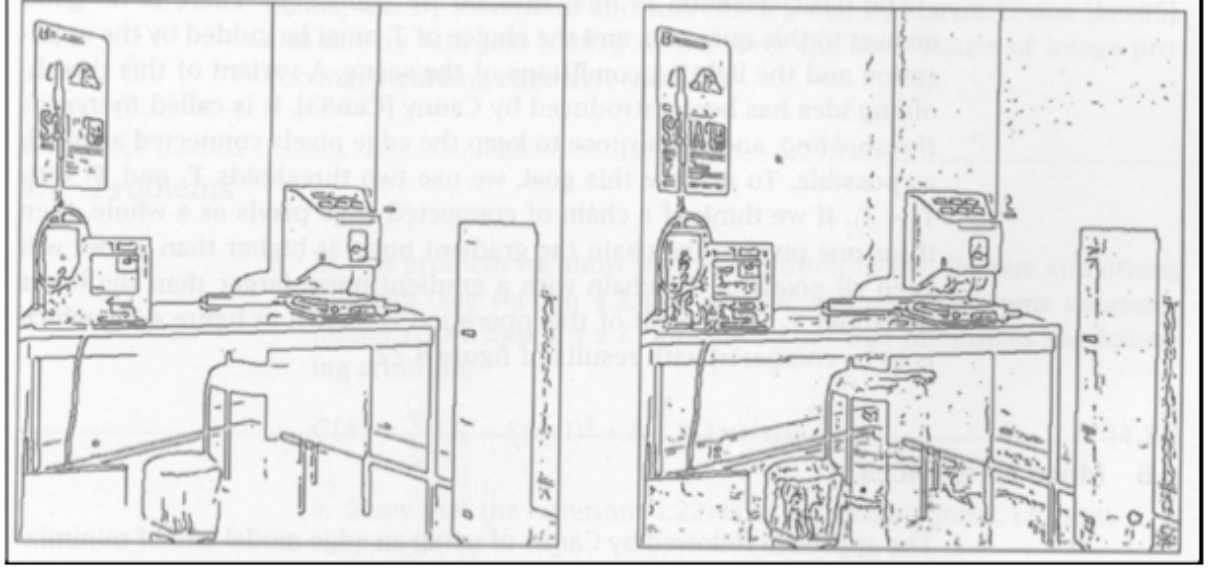


Very strong edge response. Let's start here

Weaker response but it is connected to a confirmed edge point. Let's keep it.

Continue….

Note: Darker squares illustrate stronger edge response (larger *M*)



Hysteresis
T_h=15 T_l = 5

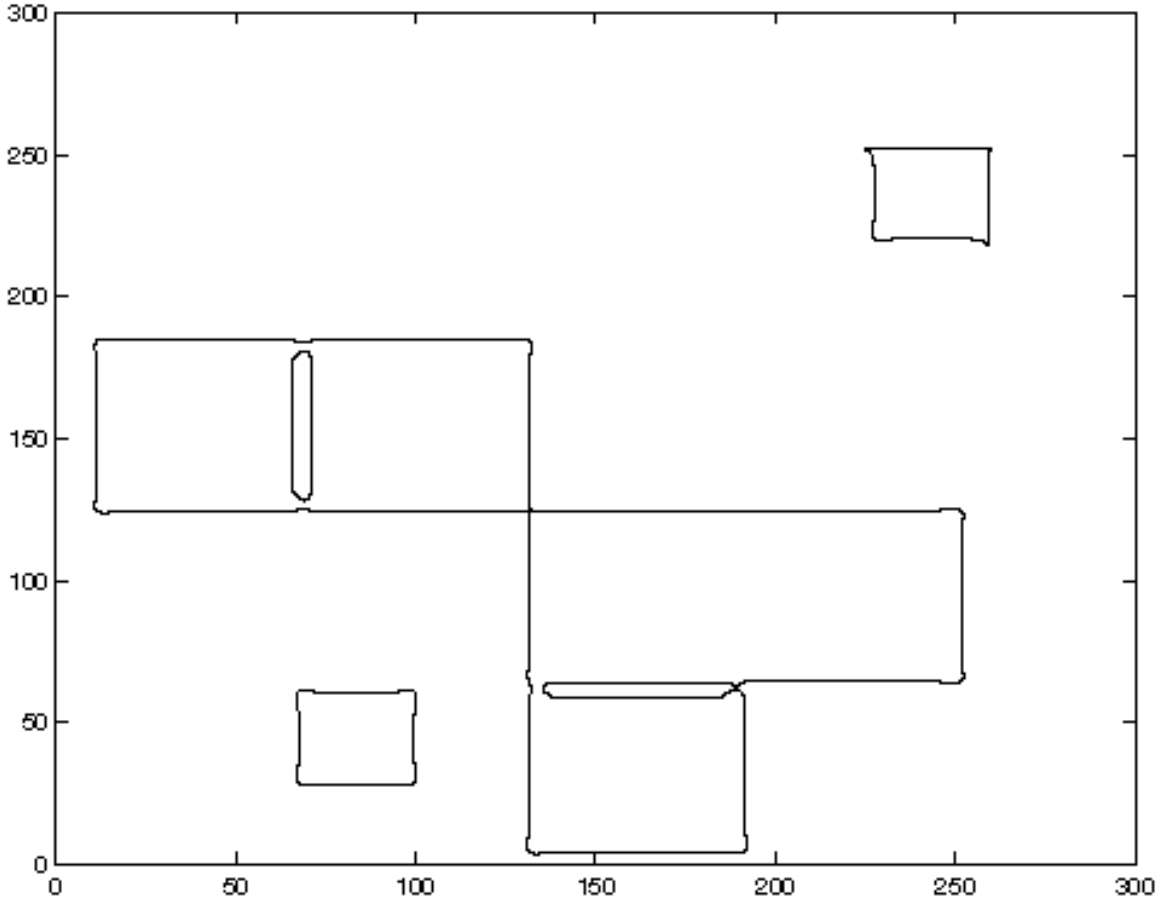
Hysteresis thresholding

Canny Edge Detector Algorithm

Steps:

1. Apply derivative of Gaussian
   - Smooth image using Gaussian filter
   - Compute gradient magnitude and angle images
2. Non-maximum suppression
   - Thin multi-pixel wide “ridges” down to single pixel width
3. Double thresholding and Linking
   - Low, high edge-strength thresholds
   - Accept all edges over low threshold that are connected to edge over high threshold

Corner Effect



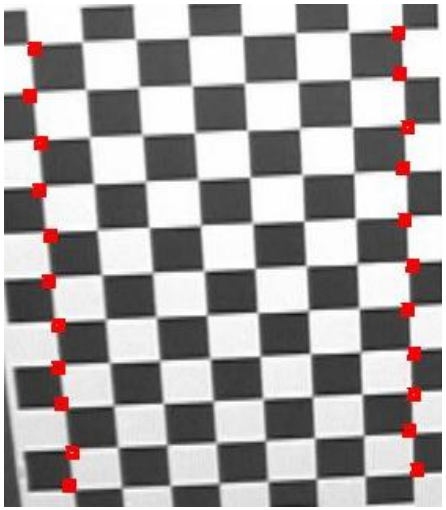
- We need a way to detect corners.
- Harris Corner Detector

Finding Corners

- Edge detectors perform poorly at corners.
- Corners provide repeatable points for matching, so are worth detecting.

Idea:

- Exactly at a corner, gradient is ill defined.
- However, in the region around a corner, gradient has two or more different values.



The Harris Corner Detector

Form the second-moment matrix:

Sum over a small region around the hypothetical corner

Gradient with respect to x, times gradient with respect to y

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

Matrix is symmetric

Simple Case

First, consider case where:

$$C = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

This means dominant gradient directions align with x or y axis

If either λ is close to 0, then this is not a corner, so look for locations where both are large.

General Case

It can be shown that since C is symmetric:

$$C = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

So every case is like a rotated version of the one on last slide.

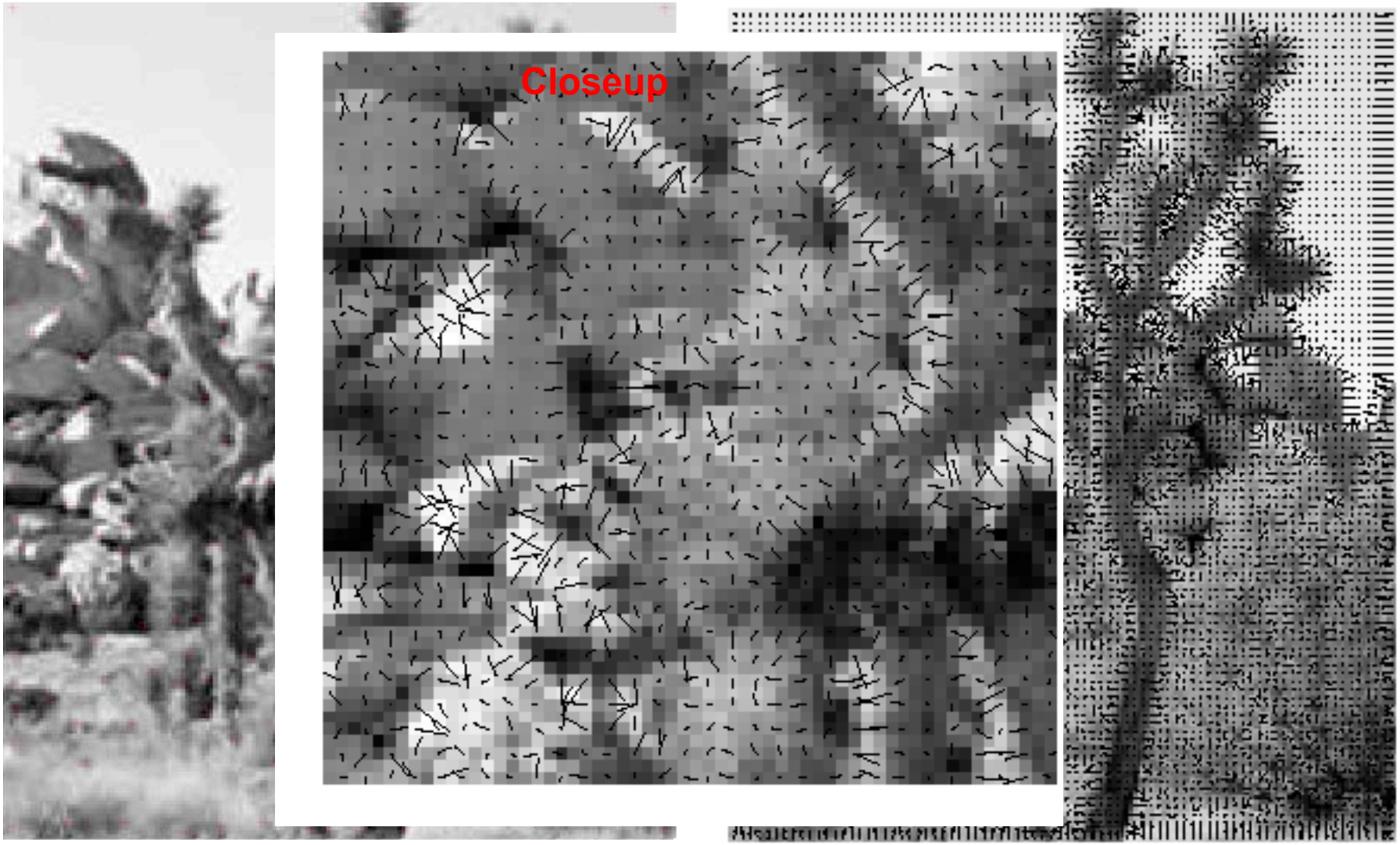
Slide credit: David Jacob

ACFR
AUSTRALIAN CENTRE FOR FIELD ROBOTICS

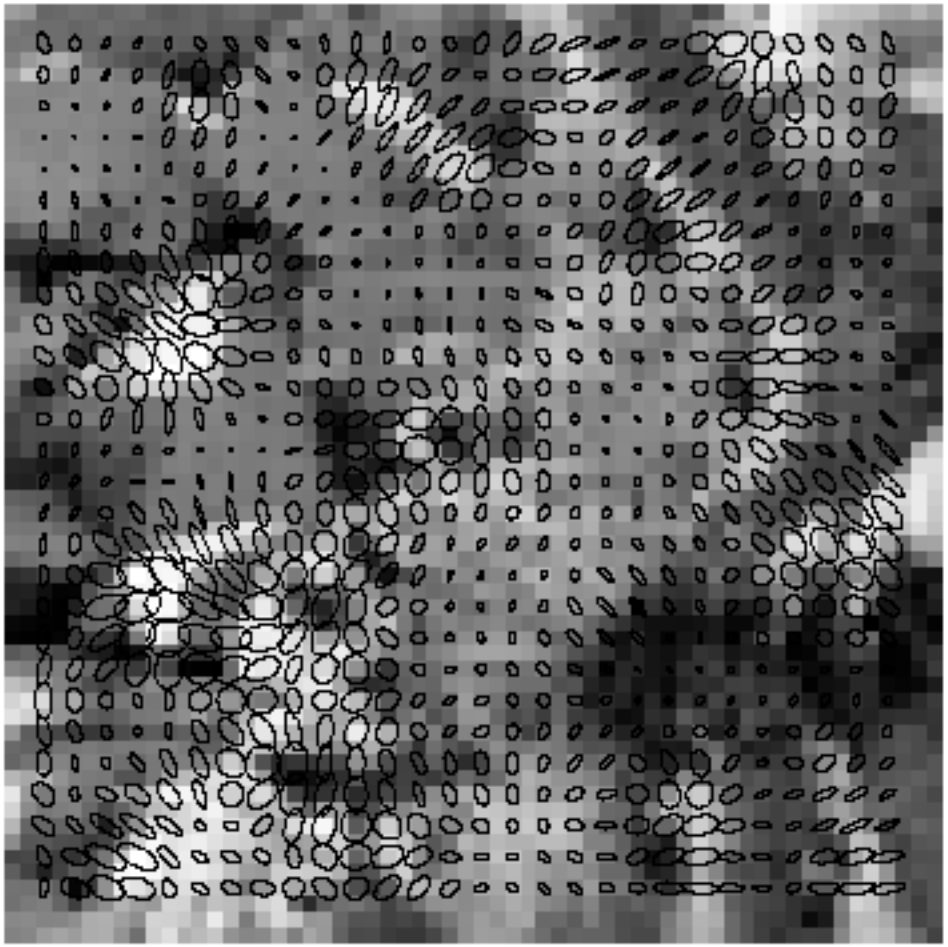
So, to detect corners

- Filter image with Gaussian to reduce noise
- Compute magnitude of the x and y gradients at each pixel
- Construct C in a window around each pixel (Harris uses a Gaussian window – just blur)
- Solve for product of λs (determinant of C)
- If λs are both big (product reaches local maximum and is above threshold), we have a corner (Harris also checks that ratio of λs is not too high)

Gradient Orientation

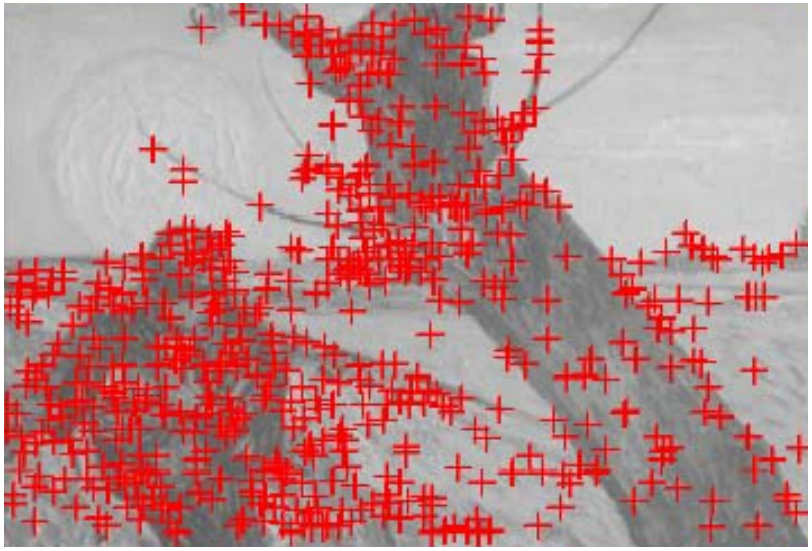


Corner Detection



Corners are detected where the product of the ellipse axis lengths reaches a local maximum.

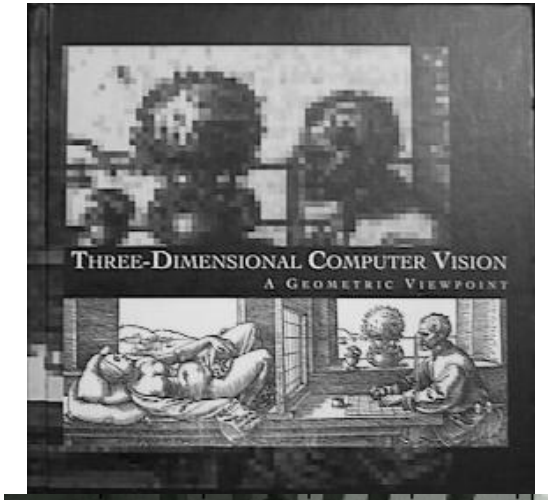
Harris Corners



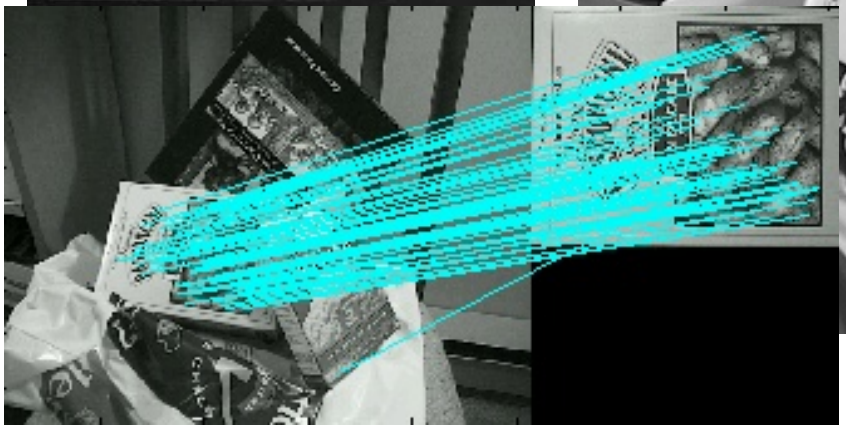
- Originally developed as features for motion tracking
- Greatly reduces amount of computation compared to tracking every pixel
- Translation and rotation invariant (but not scale invariant)

Scaling, Rotation, Illustration, Deformation

Want to find



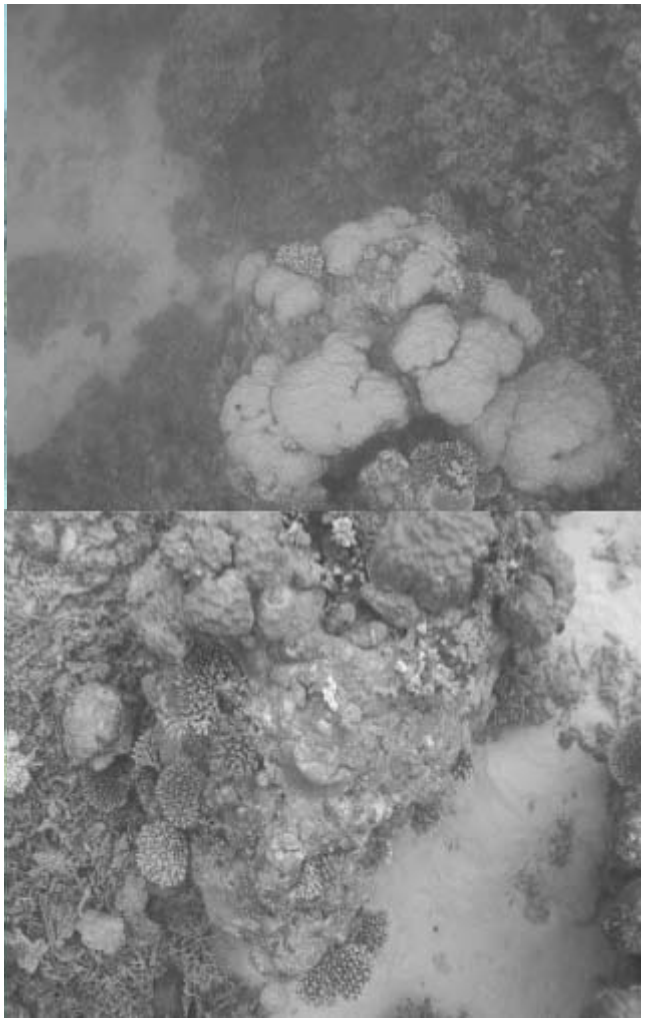
… in here



ACFR
AUSTRALIAN CENTRE FOR FIELD ROBOTICS

Scale Invariant Feature Transform (SIFT)

- Scale Invariant Feature Tracking (SIFT) features proposed by Lowe
- Successful matching over large change in range, lighting and orientation
- False positive rejection also verified
- Each keypoint: 128 element feature vector

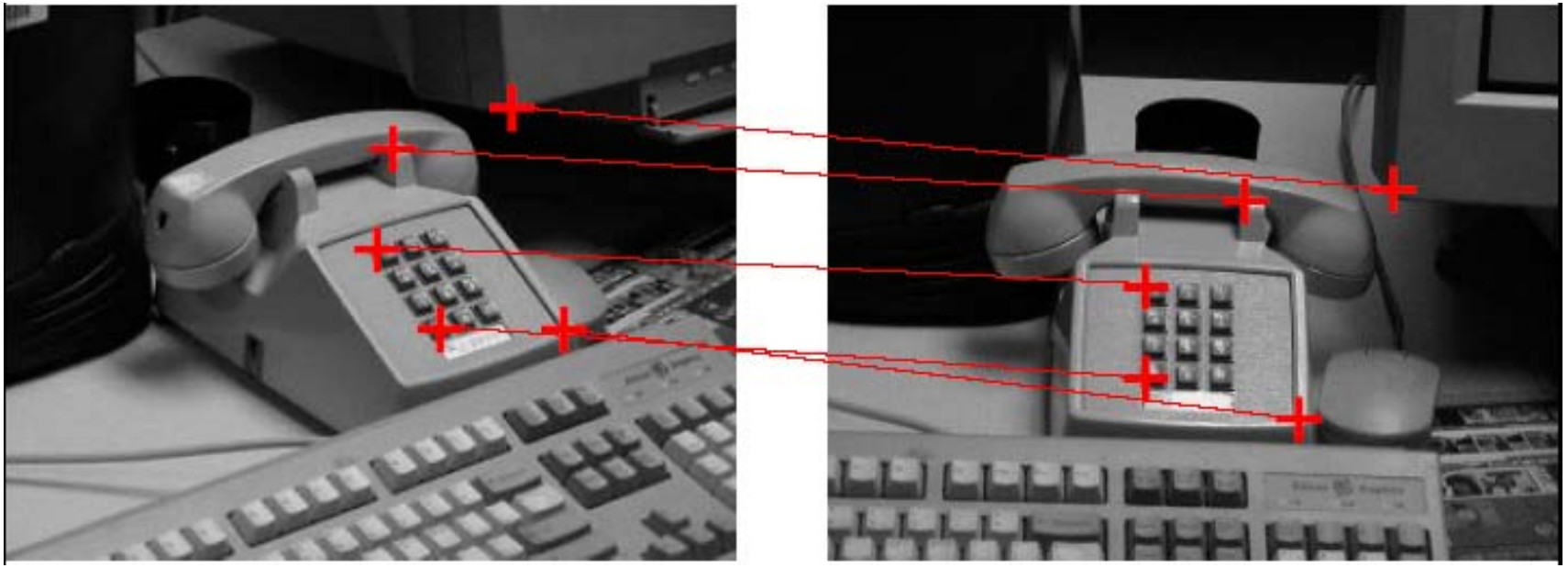


State of the art: SIFT = Scale Invariant Feature Transform

- David G. Lowe, Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision, 60, 2 (2004), pp. 91-110.

- Invariances:
  - Scaling — Yes
  - Rotation — Yes
  - Illumination — Yes
  - Deformation — Maybe
- Provides
  - Good localization — Yes

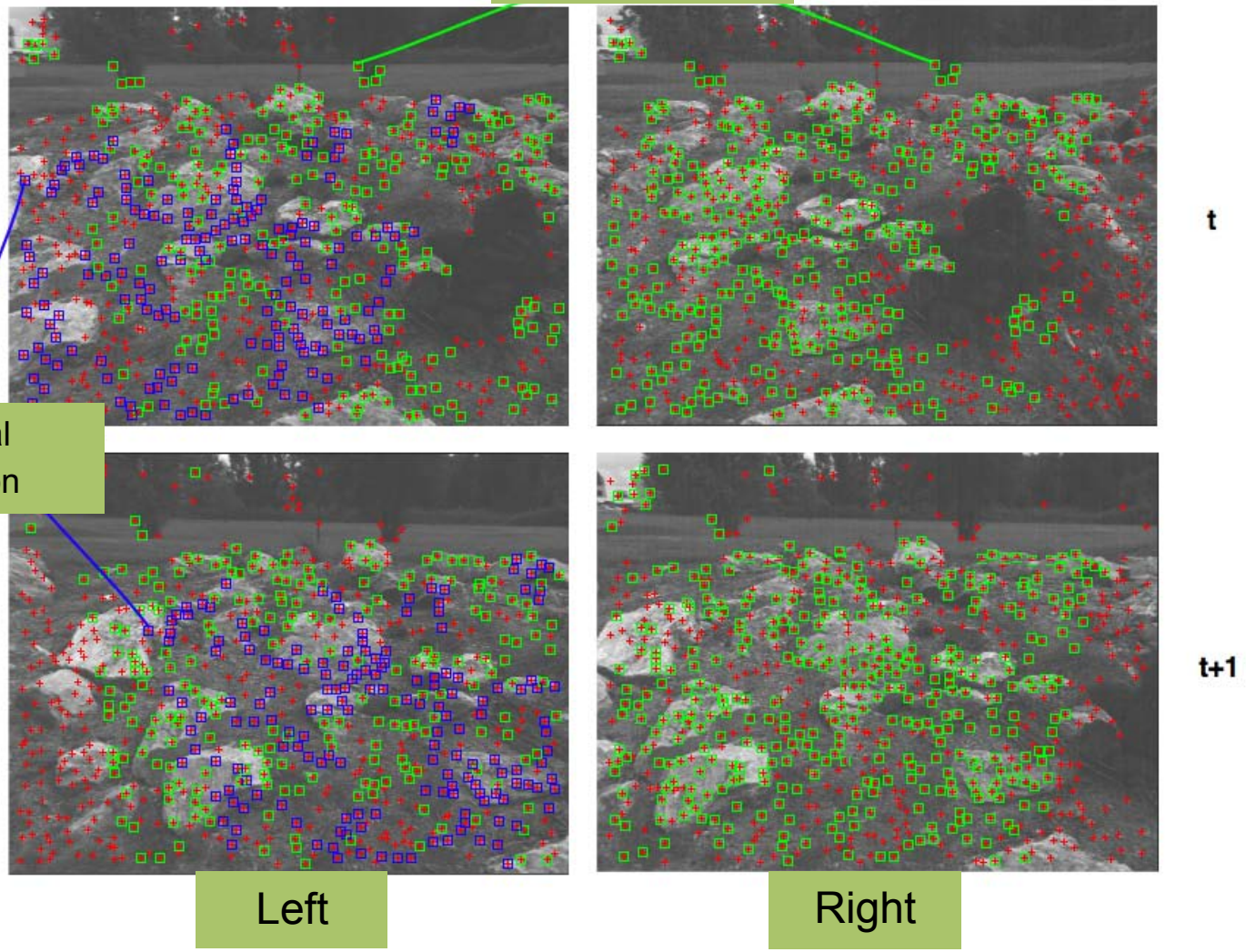
Finding Correspondences Between Images



- First step toward 3-D reconstruction
- First step toward tracking
- Object Recognition: finding correspondences between feature points in “training” and “test” images.

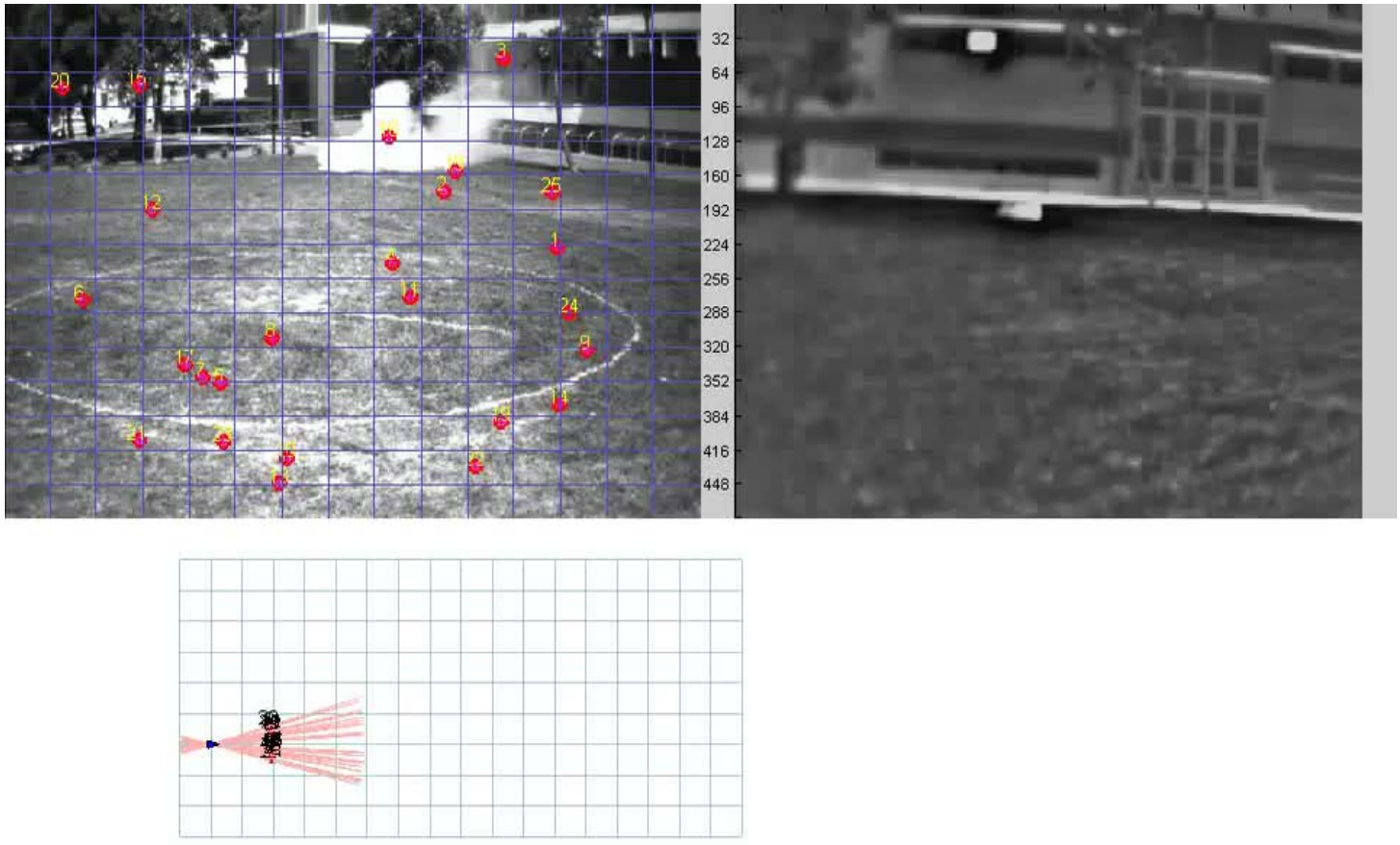
Application to Visual Odometry

Localisation
(visual odometry)



[Jung 2004]

Application to Visual SLAM



[Brunner, Peynot and Vidal-Calleja, 2011]

ACFR
AUSTRALIAN CENTRE FOR FIELD ROBOTICS

SIFT

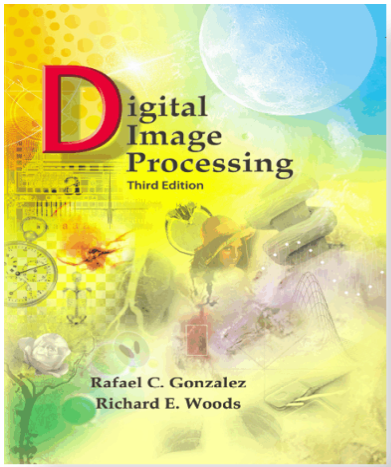
- Extensively used, for:
  - Camera localisation (Visual Odometry, SLAM…)
  - Structure from Motion
  - Tracking
  - Recognition
  - Etc…
- Computationally expensive
  - But: SURF (Bay, CVIU 2007) etc.

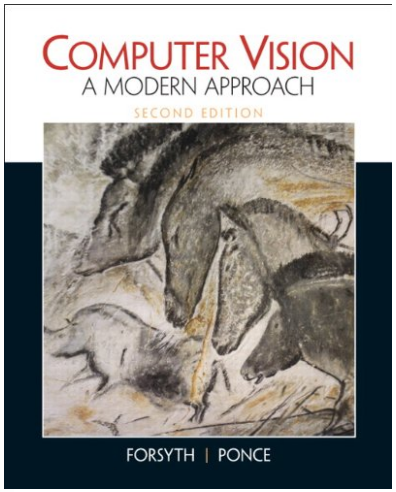
(Some) References

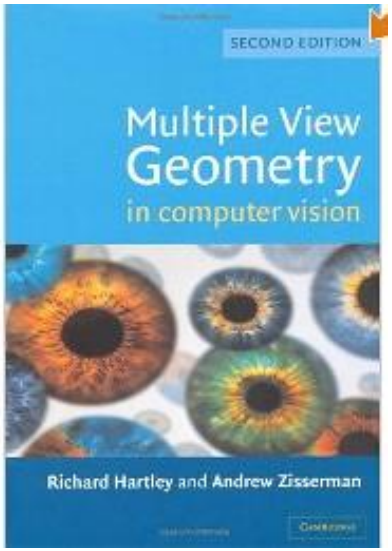
- Ballard and Brown, *Computer Vision*, Prentice Hall, 1982
- D. A. Forsyth and J. Ponce, *Computer Vision - A Modern Approach*, Prentice Hall, 2002
- R.C. Gonzalez and R.E. Woods, *Digital Image Processing*, 3rd Edition, Prentice Hall, 2008
- R. I. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, *Cambridge University Press*, 2000
- Countless Conference and Journal papers

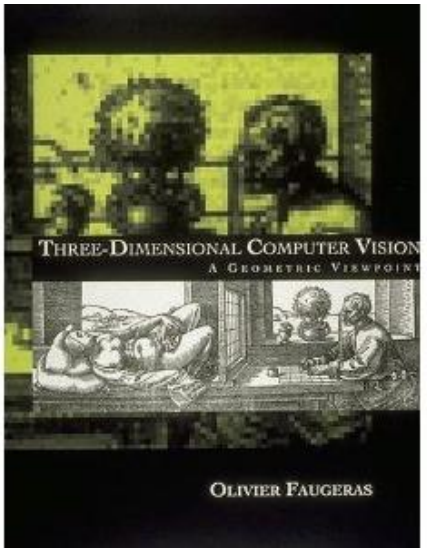
References

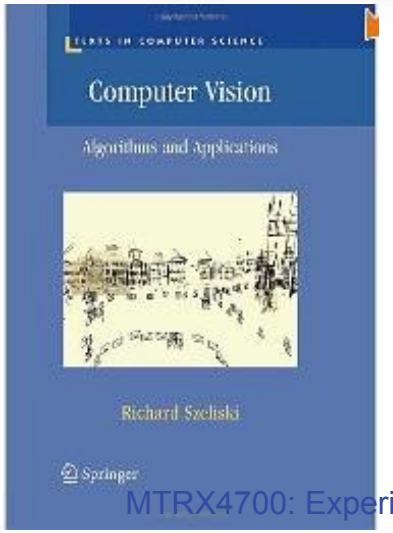
- Lots of books & papers, including:

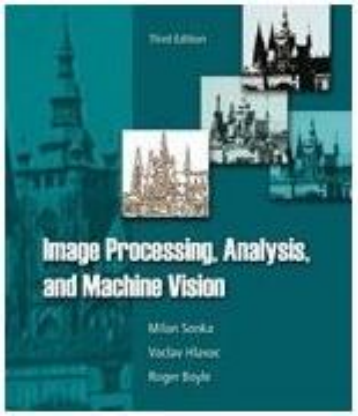












ACFR
AUSTRALIAN CENTRE FOR FIELD ROBOTICS

If you want to know more….

AMME4710:

Image Processing and Computer Vision

Coming to a theatre near you…

in Semester 2!