

BLG453E COMPUTER VISION
Fall 2017 Term
Week 7



İstanbul Technical University
Computer Engineering Department

Instructor: Prof. Gözde ÜNAL

Teaching Assistant: Enes ALBAY

David Marr, a pioneer of “vision” in 70’s: “*Vision is the process of discovering from images what is present in the world, and where it is*”. **Indeed, Vision is MUCH MORE THAN THAT!**

Learning Outcomes of the Course

Students will be able to:

1. Discuss the main problems of computer (artificial) vision, its uses and applications
2. Design and implement various image transforms: point-wise transforms, neighborhood operation-based spatial filters, and geometric transforms over images
3. Define and construct segmentation, feature extraction, and visual motion estimation algorithms to extract relevant information from images
4. Construct least squares solutions to problems in computer vision
5. Describe the idea behind dimensionality reduction and how it is used in data processing
6. Apply object and shape recognition approaches to problems in computer vision

Week 7: Template Matching and Feature Extraction: Hough Method, Detection of Lines, circles etc, and Corner Detection

At the end of Week 7-8: Students will be able to:

3. Define and construct segmentation, **feature extraction**, and visual motion estimation algorithms **to extract relevant information from images**

Template Matching

- Simple filtering method of detecting a particular feature in the image
- Provided that the appearance of this feature in the image is known accurately, we can detect it with a TEMPLATE.
- Template: a subimage that looks like the image of the object

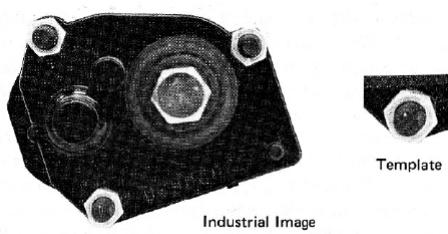
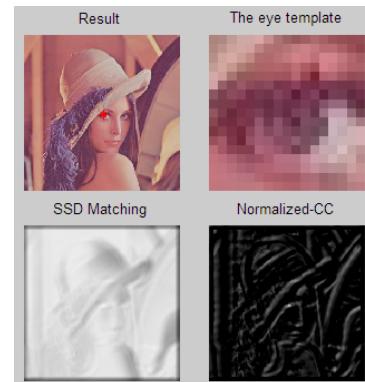


Fig. 3.3 An industrial image and template for a hexagonal nut.

Template Matching

- A similarity measure is computed to measure how well the template matches the image data for each possible template location
- Select the point of maximal match as possible template location



Fast/Robust Template Matching
by Dirk-Jan Kroon
Matlab central exchange

[https://www.mathworks.com/
matlabcentral/fileexchange/24925-fast-
robust-template-matching](https://www.mathworks.com/matlabcentral/fileexchange/24925-fast-robust-template-matching)

Template Matching

- A widely used **similarity measure** between a function f and a template t :

Sum of Squared Differences (SSD):

$$d(y)^2 = \sum_x [f(x) - t(x-y)]^2$$

- Here \sum_x means $\sum_{x=-M}^M \sum_{y=-N}^N$, M and N: size of the template t

- If the image at point y is an exact match: $d(y) \approx 0$

Template Matching

- Expand the $d(\mathbf{y})^2$ expression

$$d(\mathbf{y})^2 = \sum_{\mathbf{x}} [f^2(\mathbf{x}) - 2f(\mathbf{x}) t(\mathbf{x} - \mathbf{y}) + t^2(\mathbf{x} - \mathbf{y})]$$

- t^2 is constant, and assuming f^2 is constant, what is left is

Cross Correlation between f and t :

$$CC_{ft}(\mathbf{y}) = \sum_{\mathbf{x}} f(\mathbf{x}) t(\mathbf{x} - \mathbf{y})$$

Maximized when portion of image f under t is identical to t !

- **Like convolution:** Template-matching calculations are visualized as template being shifted across the image to different offsets, multiplied, and products are added

Template Matching

Cross Correlation (CC):

$$CC_{ft}(\mathbf{y}) = \sum_{\mathbf{x}} f(\mathbf{x}) t(\mathbf{x} - \mathbf{y})$$

Note that this is the same as convolution of $f(x)$ by $t(-x)$

→ template matching is a kind of Filtering for Object Detection

Template	Image	Correlation
1 1 1	1 1 0 0 0	7 4 2 x x
1 1 1	1 1 1 0 0	5 3 2 x x
1 1 1	1 0 1 0 0	2 1 9 x x
	0 0 0 0 0	x x x x x
	0 0 0 0 8	x x x x x
		x = undefined

Fig. 3.4 (a) A simple template. (b) An image with noise. (c) The aperiodic correlation array of the template and image. Ideally peaks in the correlation indicate positions of good match. Here the correlation is only calculated for offsets that leave the template entirely within the image. The correct peak is the upper left one at 0, 0 offset. The “false alarm” at offset 2, 2 is caused by the bright “noise point” in the lower right of the image.

Template Matching

e.g. A bright spot in the image can badly influence the correlation matching!

CC may work if the average image intensity f varies slowly compared to the template size

$$CC_{fi}(u, v) = \sum_{x,y} f(x, y) t(x - u, y - v)$$

Normalized Cross Correlation:

$$NCC_{fi}(u, v) = \frac{\sum_{x,y} [f(x, y) - \bar{f}] [t(x - u, y - v) - \bar{t}]}{\left\{ \sum_{x,y} [f(x, y) - \bar{f}]^2 \sum_{x,y} [t(x - u, y - v) - \bar{t}]^2 \right\}^{0.5}}$$

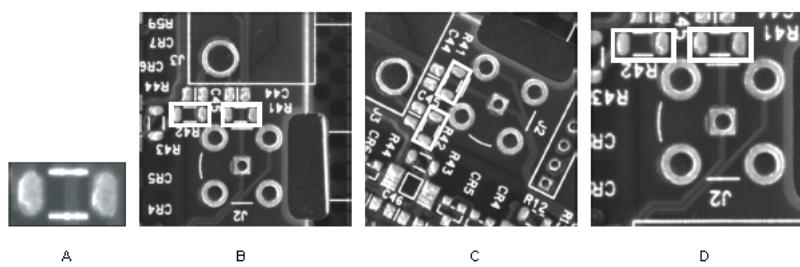
where the means are subtracted and divided by their standard deviations,
NCC less dependent on the local properties of the template and the input
image than CC

Normalized Cross Correlation

NCC is typically used in template matching problems of computer vision

$$NCC_{fi}(u, v) = \frac{\sum_{x,y} [f(x, y) - \bar{f}] [t(x - u, y - v) - \bar{t}]}{\left\{ \sum_{x,y} [f(x, y) - \bar{f}]^2 \sum_{x,y} [t(x - u, y - v) - \bar{t}]^2 \right\}^{0.5}}$$

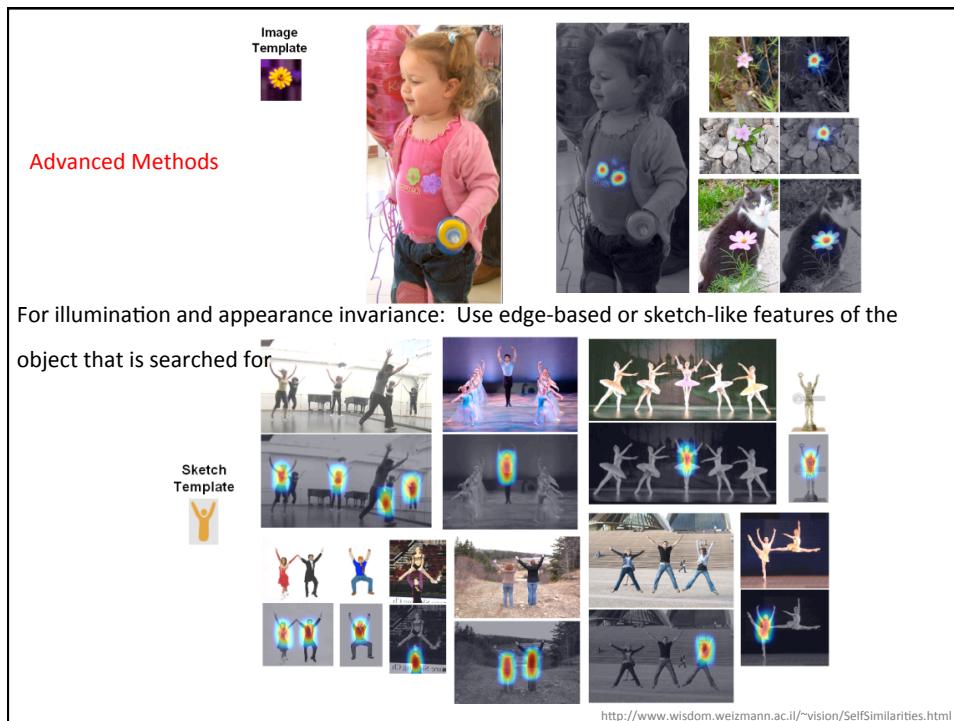
Could the form of NCC above help with matching the parts below?



When the pattern is rotated, scaled, or is imaged in a different lighting condition?

Different rotations and scales should be taken into account

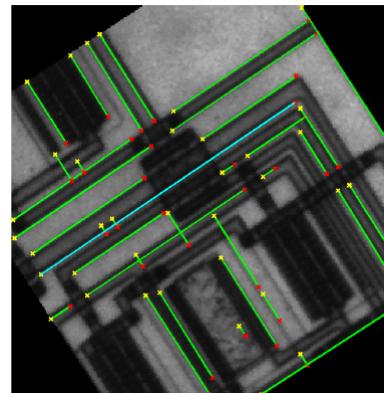
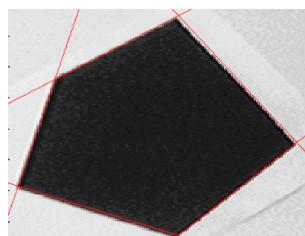
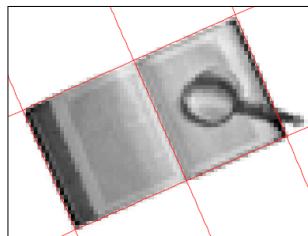
Figure from National Instruments website: what to expect from a pattern matching tool?



Next: Feature Extraction from images

Lines
Quadratic forms
Corners
...

Line Detection and Parameterization: Hough Transform



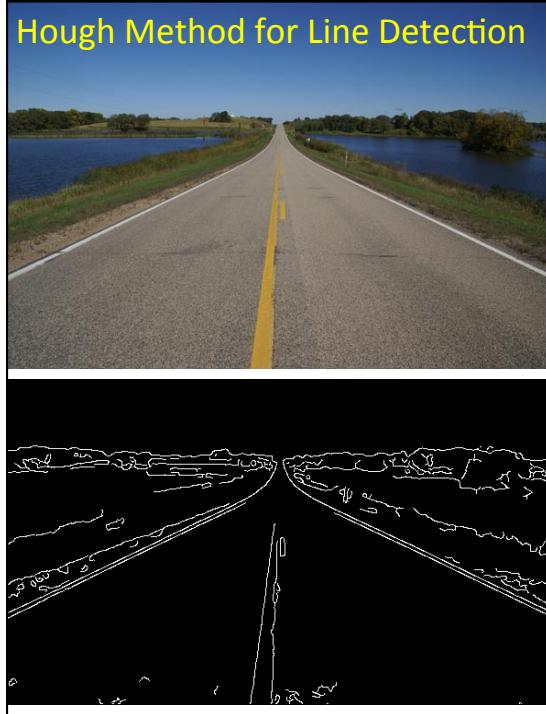
Hough Method for Line Detection



Goal: Detecting Straight Lines in Images

Assume we have detected some points likely to be on lines in an image

Hough technique will organize these points into straight lines by considering all possible straight lines at once and rating each one on how well it explains the data



Hough Method for Line Detection

Equation for a line: $y = mx + c$

Q: What are the lines that could pass through point x' below?

A: All the lines that satisfy

$$y' = mx' + c$$

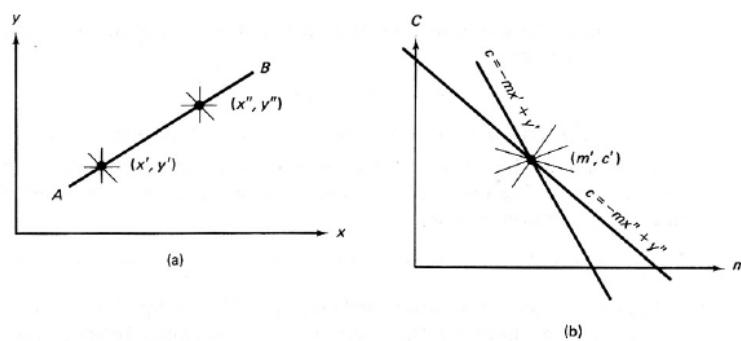
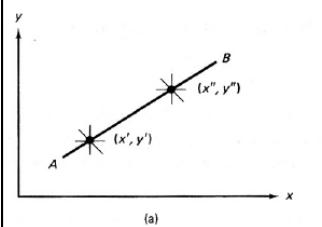


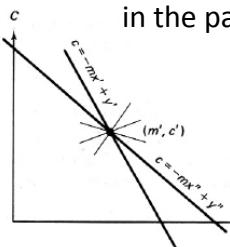
Fig. 4.5 A line (a) in image space; (b) in parameter space.

Hough Method for Line Detection



- Fixing (x', y') , the line equation is now in the parameter space (m, c) :

$$c = -mx + y$$



- A second point (x'', y'') will also have a line in the parameter space

→ All such lines will intersect at (m', c') → this point in the parameter space corresponds to the line AB in the image space with coord (x, y)

Hough Method for Line Detection

Algorithm 4.1: Line Detection with the Hough Algorithm

1. Quantize parameter space between appropriate maximum and minimum values for c and m .
2. Form an accumulator array $A(c, m)$ whose elements are initially zero.
3. For each point (x, y) in a gradient image such that the strength of the gradient exceeds some threshold, increment all points in the accumulator array along the appropriate line, i.e.,

$$A(c, m) := A(c, m) + 1$$

for m and c satisfying $c = -mx + y$ within the limits of the digitization.

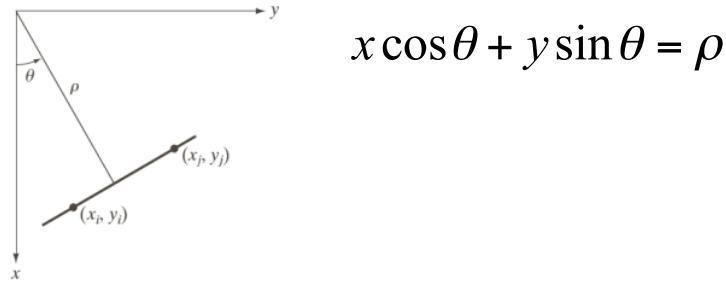
4. Local maxima in the accumulator array now correspond to collinear points in the image array. The values of the accumulator array provide a measure of the number of points on the line.

Q: Do you see where we might run into problem with this algorithm?

Hough Method for Line Detection

In the previous algorithm, when there is a vertical line in the image, m: the slope of the line is ∞

Recommended Rather than (c, m) parameters, use the line polar parameterization with : (ρ, θ)



Same idea as before, but with different parameterization of the line: now the lines become sinusoidal curves in the parameter space \rightarrow

Hough Method for Line Detection

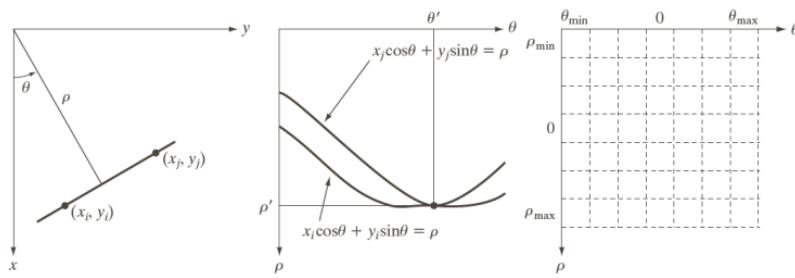


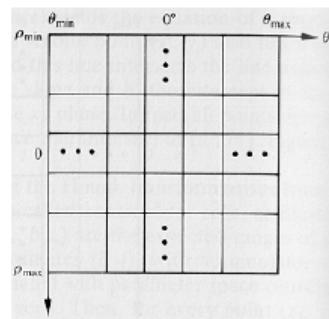
FIGURE 10.32 (a) (ρ, θ) parameterization of line in the xy -plane. (b) Sinusoidal curves in the $\rho\theta$ -plane; the point of intersection (ρ', θ') corresponds to the line passing through points (x_i, y_i) and (x_j, y_j) in the xy -plane.
(c) Division of the $\rho\theta$ -plane into accumulator cells.

The lines with the parameterization: $x \cos \theta + y \sin \theta = \rho$

become sinusoidal curves in the parameter space

Hough Algorithm

- Quantize the parameter space
 $P[\rho_{\min}, \dots, \rho_{\max}][\theta_{\min}, \dots, \theta_{\max}]$ (accumulator array)



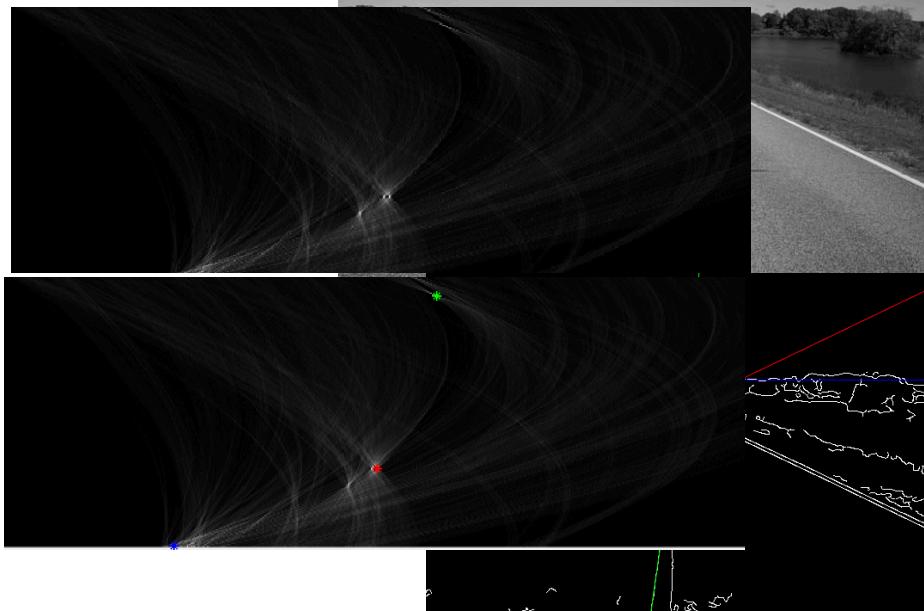
- For each edge point (x, y)

```

For( $\theta = \theta_{\min}; \theta \leq \theta_{\max}; \theta++$ ) {
     $\rho = x \cos \theta + y \sin \theta$ ; /* round off if needed */
     $(P[\rho][\theta])++$ ; /* voting */
}

```
- Find local maxima in $P[\rho][\theta]$

Line Detection: Hough Transform



Line Detection: Hough Transform

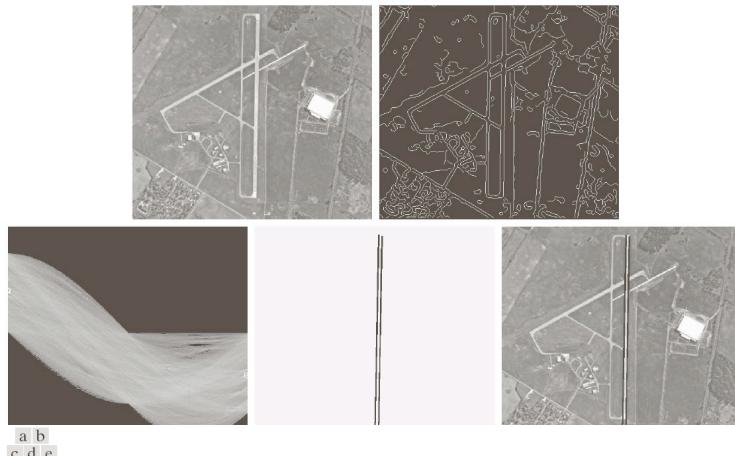


FIGURE 10.34 (a) A 502×564 aerial image of an airport. (b) Edge image obtained using Canny's algorithm. (c) Hough parameter space (the boxes highlight the points associated with long vertical lines). (d) Lines in the image plane corresponding to the points highlighted by the boxes. (e) Lines superimposed on the original image.

Digital Filtering of Images, Gonzalez and Woods Book

General Hough Method

- Can be extended to other functions
where \mathbf{a} is a parameter vector:

$$f(\mathbf{x}, \mathbf{a}) = 0$$



For example, Circle :

$$(x - a)^2 + (y - b)^2 = r^2$$

$$f(\mathbf{x}, \mathbf{a}) = (x - a)^2 + (y - b)^2 - r^2$$

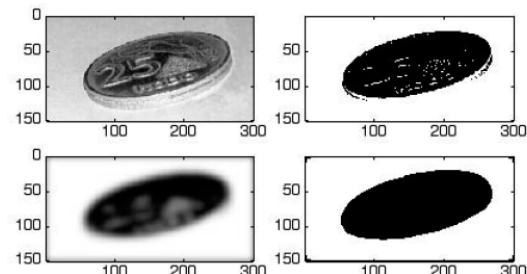
$\mathbf{x} = (x, y)$ $\mathbf{a} = (a, b, r)$
 known feature coordinates,
 e.g. Edge point coordinates Parameter vector
 to be estimated

General Hough Method



Try to obtain these yourself

Ellipse Detection



General Ellipse equation (including rotation) is a quadratic form:

$$ax^2 + by^2 + cxy + dx + ey + f = 0$$

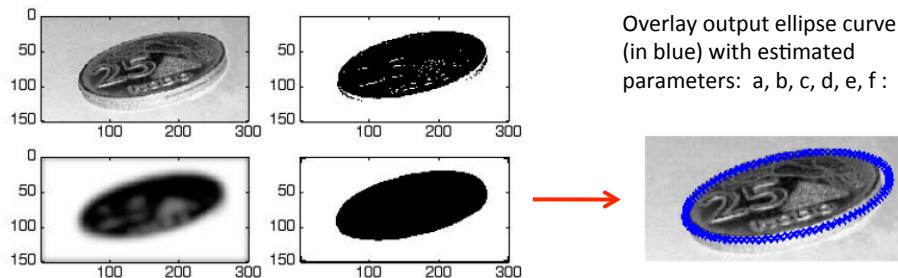
We can use the LEAST SQUARES technique, to solve for the parameters of the ellipse.

Recall: Ellipse not centered but with no rotation: $((x - x_0) / a)^2 + ((y - y_0) / b)^2 = 1$

Ellipse Detection

ON BOARD: Construct the LEAST SQUARES problem using this equation, and solve for the parameters of the ellipse

E.g. Take your ellipse equation: $ax^2 + by^2 + cxy + dx + ey - 1 = 0$



Self Reading (Not included)

RANSAC : Random sample consensus used to eliminate outlier points in model fitting

https://en.wikipedia.org/wiki/Random_sample_consensus

Last time we studied Edge Detection
We'll cover corner detection
and mention general features



Image Feature Extraction: Why we need features?

Example: Build a Panorama



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

M. Brown and D. G. Lowe. Recognising Panoramas. ICCV 2003
45/59

© R. Siegwart, I. Nourbakhsh

Image Feature Extraction: Why we need features?

How do we build panorama?

- We need to match (align) images

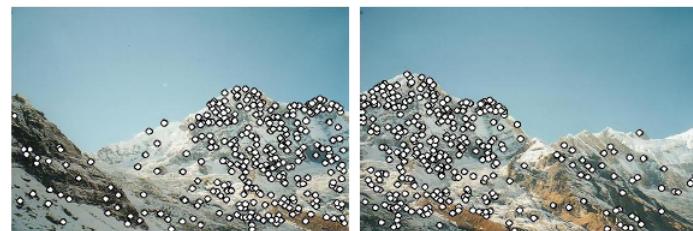


46/59

© R. Siegwart, I. Nourbakhsh

Matching with Features

- Detect feature points in both images

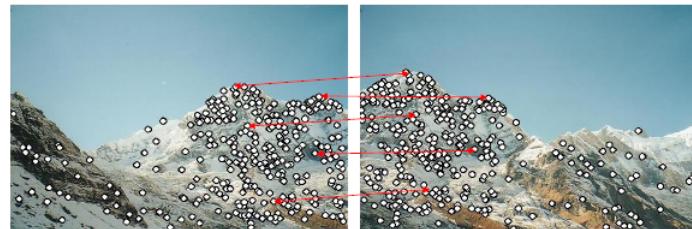


47/59

© R. Siegwart, I. Nourbakhsh

Matching with Features

- Detect feature points in both images
- Find corresponding pairs



48/59

© R. Siegwart, I. Nourbakhsh

Matching with Features

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



49/59

© R. Siegwart, I. Nourbakhsh

More motivation...

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

52/59

© R. Siegwart, I. Nourbakhsh

Matching with Features

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



no chance to match!

We need a repeatable detector

50/59

© R. Siegwart, I. Nourbakhsh

Matching with Features

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



We need a reliable and distinctive descriptor

51/59

© R. Siegwart, I. Nourbakhsh

Most Popular Feature Detectors

Hough method for line and other parametric structure detection (1959, 1972)

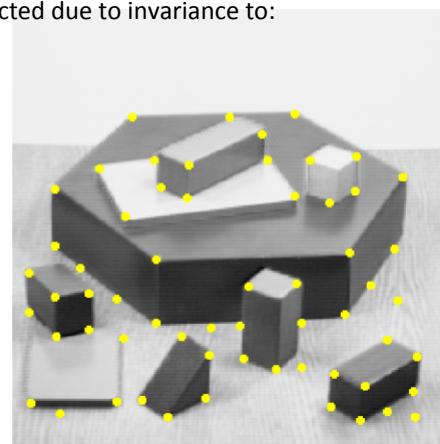
Harris Corner Detector (1988)

SIFT Feature Detector (2004) (you are not responsible in this class, because time does not permit to study it thoroughly, but please study it on your own, as it was one of the most popular feature extractors of early 2000's.)

Popular Feature Detectors

Popular interest points in images are selected due to invariance to:

- scale, rotation
- illumination variation
- image noise



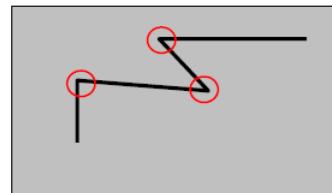
E.g. Corners

Harris Detector (1988)

Corner Detection: Harris Operator

Harris corner detector

An introductory example:

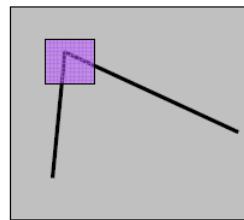


54/59 C.Harris, M.Stephens. "A Combined Corner and Edge Detector". 1988

© R. Siegwart, I. Nourbakhsh

The Basic Idea

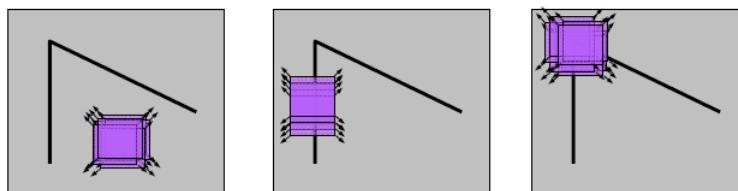
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



55/59

© R. Siegwart, I. Nourbakhsh

Harris Detector: Basic Idea



“flat” region:
no change in all
directions

“edge”:
no change along the
edge direction

“corner”:
significant change
in all directions

56/59

© R. Siegwart, I. Nourbakhsh

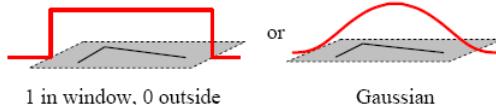
Corner (Harris) Detector (Mathematics)

Let's analyze the local intensity changes :

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

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

Window function $w(x,y) =$



57/59

© R. Siegwart, I. Nourbakhsh

Corner Detector (Mathematics)

For small shifts (u,v) , we have the following approximation (after Taylor series expansion on $I(x+u, y+v)$ and expanding the quadratic term):

$$E(u,v) \cong \begin{bmatrix} u & v \end{bmatrix} G \begin{bmatrix} u \\ v \end{bmatrix}$$

where G is a 2×2 matrix computed from image derivatives in the given window:

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

G is called the **Image Structure Tensor**

Image Structure Tensor

$$G(x, y) = \begin{bmatrix} \sum_w (I_x(x_i, y_i))^2 & \sum_w (I_x(x_i, y_i) I_y(x_i, y_i)) \\ \sum_w (I_x(x_i, y_i) I_y(x_i, y_i)) & \sum_w (I_y(x_i, y_i))^2 \end{bmatrix}$$

- W is the window of a fixed size in your image, (x_i, y_i) pixel coordinates in that window.
- I_x and I_y are the local approximations to the first order partial derivatives of the image J, which is the filtered image I with a Gaussian filter (as we've seen in previous week) :

$$I_x = \frac{\partial I}{\partial x} = \frac{I(x+1, y) - I(x-1, y)}{2} \quad I_y = \frac{\partial I}{\partial y} = \frac{I(x, y+1) - I(x, y-1)}{2}$$

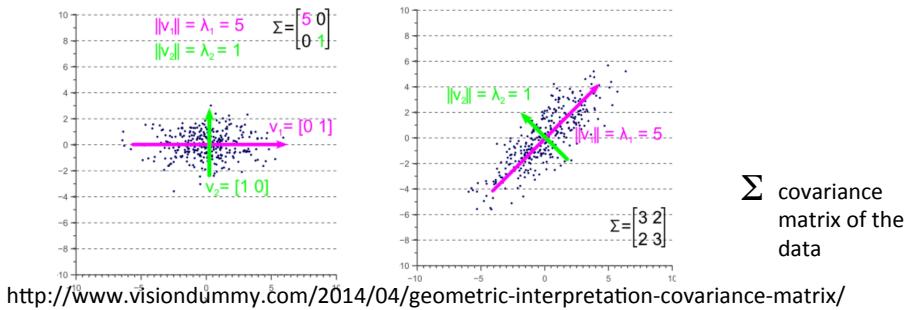
Corner Detector (Mathematics)

$$G(x, y) = \begin{bmatrix} \sum_w (I_x(x_i, y_i))^2 & \sum_w (I_x(x_i, y_i) I_y(x_i, y_i)) \\ \sum_w (I_x(x_i, y_i) I_y(x_i, y_i)) & \sum_w (I_y(x_i, y_i))^2 \end{bmatrix}$$

Intensity change in shifting window: perform eigenvalue analysis on G

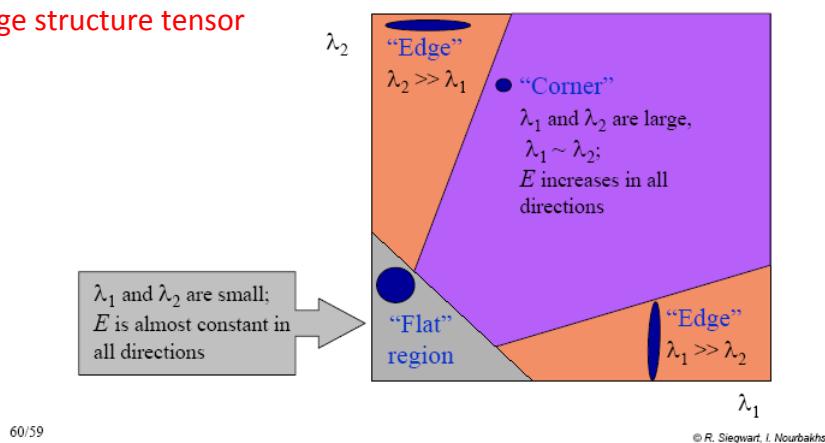
$$G = V \begin{bmatrix} \lambda_2 & 0 \\ 0 & \lambda_1 \end{bmatrix} V^{-1} \quad \lambda_2 \geq \lambda_1 \quad : \text{eigenvalues of } G$$

V : Eigenvector matrix of G



Classification of image points using eigenvalues of G

G: image structure tensor



60/59

© R. Siegwart, I. Nourbakhsh

- 1) If both eigenvalues of G are small, local autocorrelation function ($E(u,v)$) is FLAT, little change in any direction
- 2) If one eigen value is high, the other low, then only there is high shift in one direction → EDGE
- 3) If both eigen values are high, the local autocorr fn. is sharply peaked → CORNER

Corner Detection Algorithm

G: Image Structure Tensor

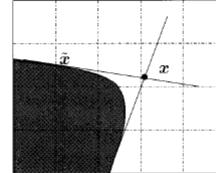


Figure 4.5. A corner feature x is the virtual intersection of local edges (within a window).

Algorithm 4.2 (Corner detector).

Given an image $I(x,y)$, follow the steps to detect whether a given pixel (x,y) is a corner feature:

- set a threshold $\tau \in \mathbb{R}$ and a window W of fixed size, and compute the image gradient (I_x, I_y)
- at all pixels in the window W around (x,y) compute the matrix

$$G = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}; \quad (4.30)$$

- if the smallest singular value $\sigma_2(G)$ is bigger than the prefixed threshold τ , then mark the pixel as a feature (or corner) point.

or the smallest eigenvalue of G

Ma, Soatto et al. "An Invitation to 3D Vision" Book

Harris Corner Detector (Mathematics)

A Measure of Corner Response:

$$R = C(G) = \det(G) - k \operatorname{trace}^2(G)$$

$$\det(G) = \lambda_1 \lambda_2$$

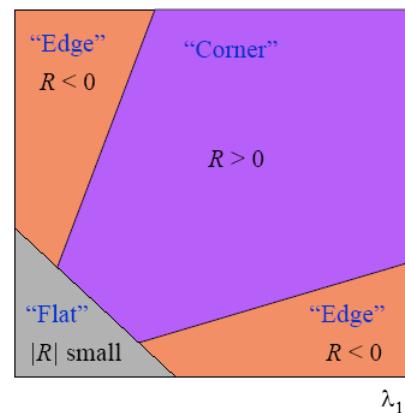
$$\operatorname{trace}(G) = \lambda_1 + \lambda_2$$

k – empirical constant, a small value $k=0.04 – 0.06$

Using the 2×2 local structure tensor matrix G , the Harris corner detector THRESHOLDS the quantity R .

Harris Detector: Mathematics

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



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

© R. Siegwart, I. Nourbakhsh

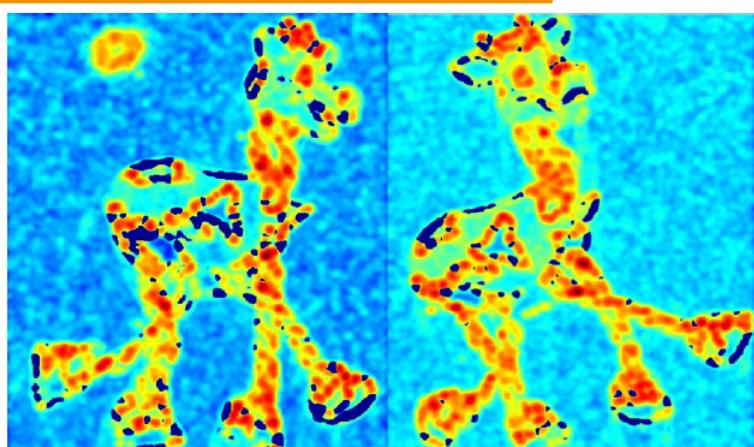
Harris Detector: Workflow



64/59

© R. Siegwart, I. Nourbakhsh

Harris Detector: Workflow

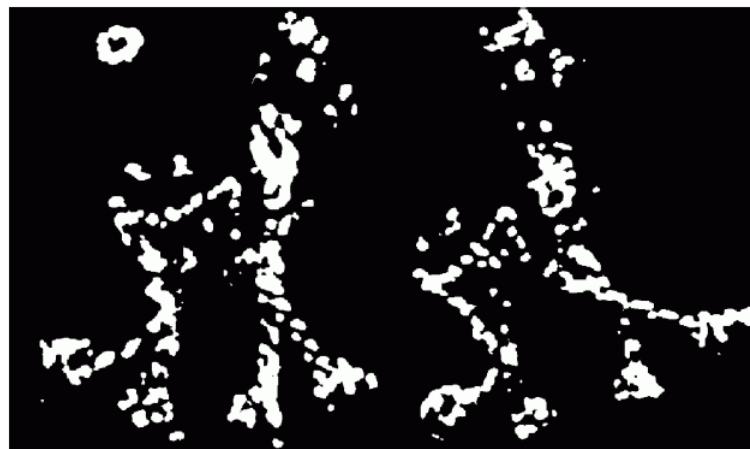
Compute corner response R 

65/59

© R. Siegwart, I. Nourbakhsh

Harris Detector: Workflow

Find points with large corner response: $R > \text{threshold}$

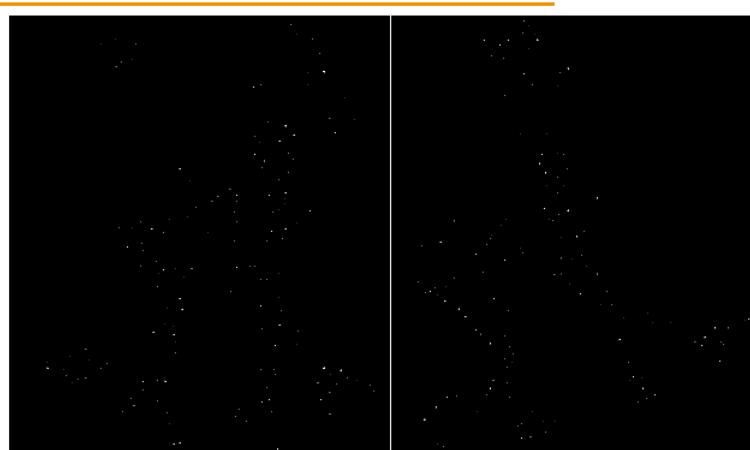


66/59

© R. Siegwart, I. Nourbakhsh

Harris Detector: Workflow

Take only the points of local maxima of R



67/59

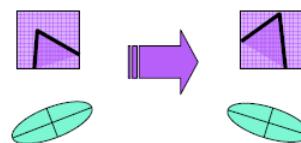
© R. Siegwart, I. Nourbakhsh

Harris Detector: Workflow



Harris Detector: Some Properties

- Rotation invariance

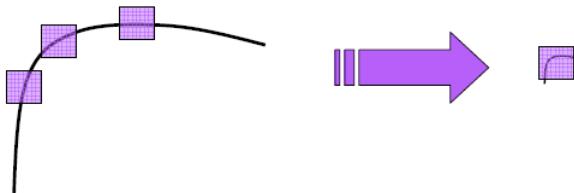


Ellipse rotates but its shape (i.e. eigenvalues)
remains the same

Corner response R is invariant to image rotation

Harris Detector: Some Properties

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



All points will be
classified as [edges](#)

Corner !

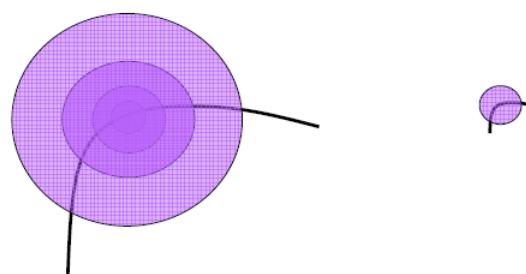
70/59

© R. Siegwart, I. Nourbakhsh

Feature Detection

Scale Invariant Detection

- Consider regions (e.g. circles) of different sizes around a point
- Regions of corresponding sizes will look the same in both images

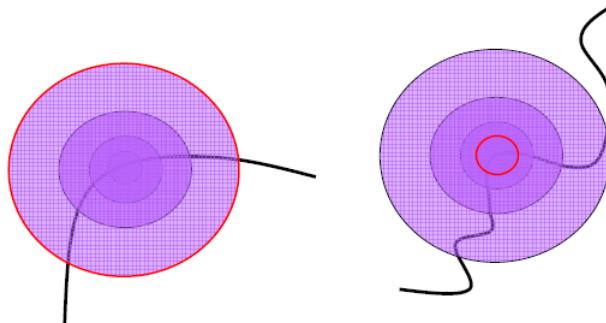


72/59

© R. Siegwart, I. Nourbakhsh

Scale Invariant Detection

- The problem: how do we choose corresponding circles *independently* in each image?



73/59

© R. Siegwart, I. Nourbakhsh

Scale Invariant Detection

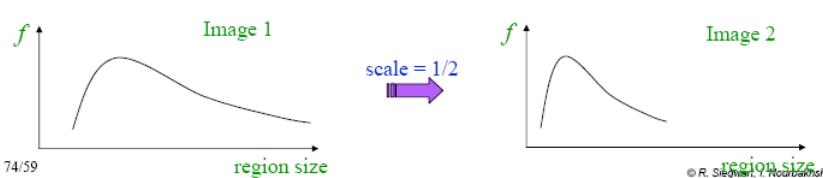
- Solution:

➤ Design a function on the region (circle), which is “scale invariant” (the same for corresponding regions, even if they are at different scales)

Example: average intensity. For corresponding regions (even of different sizes) it will be the same.

Typically, image gradient based functions are used

- For a point in one image, we can consider it as a function of region size (circle radius)



74/59

© R. Siegwart, I. Nourbakhsh

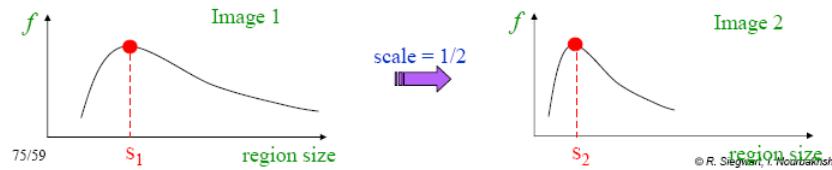
Scale Invariant Detection

- Common approach:

Take a local maximum of this function

Observation: region size, for which the maximum is achieved, should be *invariant* to image scale.

Important: this scale invariant region size is found in each image **independently!**



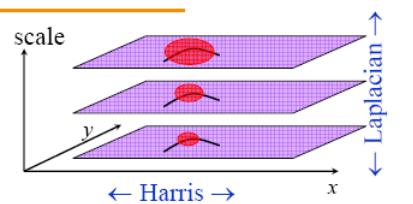
© R. Siegwart, I. Nourbakhsh

Scale Invariant Detectors

- Harris-Laplacian¹**

Find local maximum of:

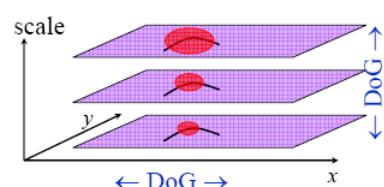
- Harris corner detector in space (image coordinates)
- Laplacian in scale



- SIFT (Lowe)²**

Find local maximum of:

- Difference of Gaussians in space and scale



¹K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001

²D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". IJCV 2004

© R. Siegwart, I. Nourbakhsh

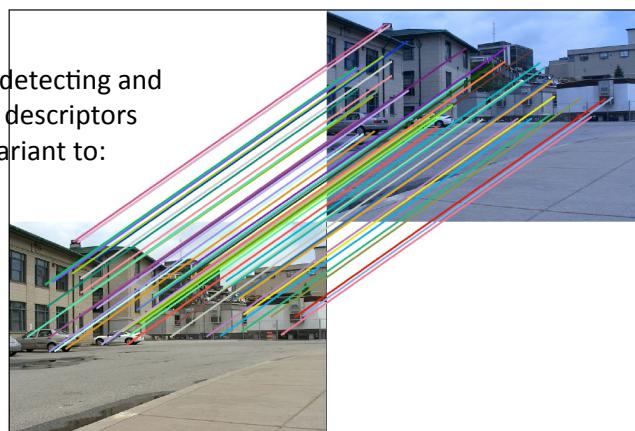
Feature Matching Problem

- After you detect features in 2 images, how to match them?
- You have to define a descriptor for a feature: The simplest solution is the intensities of its spatial neighbors. This might not be robust to brightness change or small shift/rotation.
- Therefore, typically, features are based on gradients of images

e.g. SIFT detector:

SIFT is an approach for detecting and extracting local feature descriptors that are reasonably invariant to:

- Rotation
- Scaling
- Small changes in viewpoint
- illumination



END OF LECTURE

Recall Learning objectives of Week 8: Students are able to:

3. Define and construct segmentation, **feature extraction**, and visual motion estimation algorithms **to extract relevant information from images**

Next assignment: Work on feature extraction and template matching

Reading Assignments:

[Klette Book] 1.1, 1.3, 2.1.1, 2.1.2

[Gonzalez and Woods]: Chap 3.1 – 3.3

Those interested: Advanced Feature Extractors like **SIFT**: David Lowe, Distinctive Image Features from Scale-Invariant Key Points, International Journal of Computer Vision 2004.

Extra Slides

Scale Invariant Detection

- A “good” function for scale detection:
has one stable sharp peak



- For usual images: a good function would be a one which responds to contrast (sharp local intensity change)

76/59

© R. Siegwart, I. Nourbakhsh

Scale Invariant Detection

- Functions for determining scale

$$f = \text{Kernel} * \text{Image}$$

Kernels:

$$L = \sigma^2 (G_{xx}(x, y, \sigma) + G_{yy}(x, y, \sigma))$$

(Laplacian)

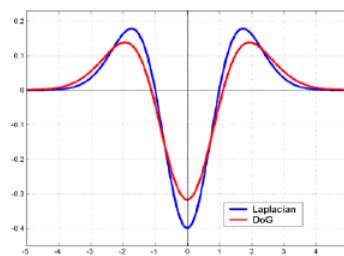
$$DoG = G(x, y, k\sigma) - G(x, y, \sigma)$$

(Difference of Gaussians)

where Gaussian

$$G(x, y, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

Note: both kernels are invariant to
scale and rotation



77/59

© R. Siegwart, I. Nourbakhsh

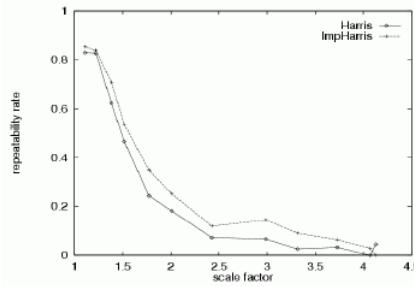
EXTRA MATERIAL

Harris Detector: Some Properties

- Quality of Harris detector for different scale changes

points repeated btw two images wrt total # detected points in each image (min of the two) **Repeatability rate:**

$$\frac{\# \text{ correspondences}}{\# \text{ possible correspondences}}$$



71/59 C.Schmid et.al. "Evaluation of Interest Point Detectors". IJCV 2000

© R. Siegwart, I. Nourbakhsh

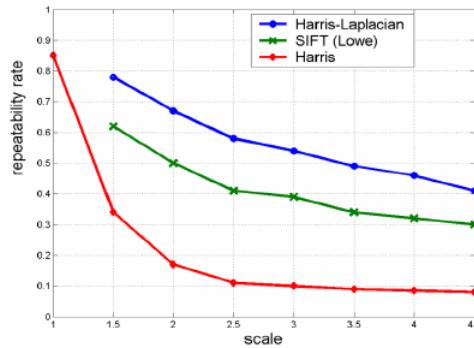
Interest points which can not be observed in both images corrupt the repeatability measure, therefore only points which lie in the common scene part used to compute repeatability

Scale Invariant Detectors

- Experimental evaluation of detectors w.r.t. scale change

Repeatability rate:

$$\frac{\# \text{ correspondences}}{\# \text{ possible correspondences}}$$



79/59 K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001

© R. Siegwart, I. Nourbakhsh

points repeated btw two images wrt total # detected points in each image (min of the two)

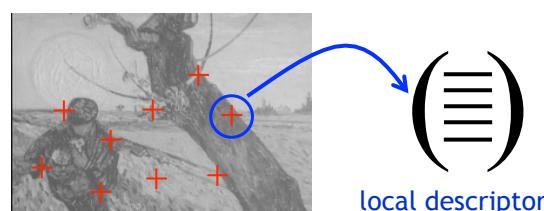
Interest points which can not be observed in both images corrupt the repeatability measure, therefore only points which lie in the common scene part used to compute repeatability

SIFT (Scale Invariant Feature Transform)

76

SIFT stages:

- | | |
|---|------------|
| <input type="checkbox"/> Scale-space extrema detection | detector |
| <input type="checkbox"/> Keypoint localization | |
| <input type="checkbox"/> Orientation assignment | |
| <input type="checkbox"/> Keypoint descriptor generation | descriptor |



matching

local descriptor

A 500x500 image gives about 2000 features

77

We want invariance!!!

- Good features should be robust to all sorts of nastiness that can occur between images.

78

Types of invariance

- Illumination



79

Types of invariance

- Illumination
- Scale



80

Types of invariance

- Illumination
- Scale
- Rotation



81

Types of invariance

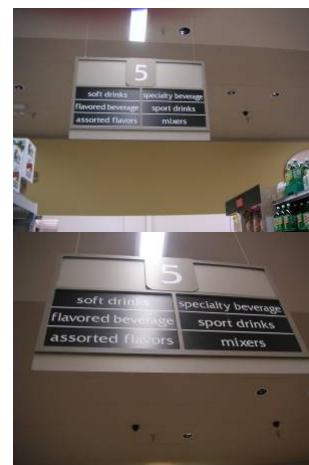
- Illumination**
- Scale**
- Rotation**
- Affine**



82

Types of invariance

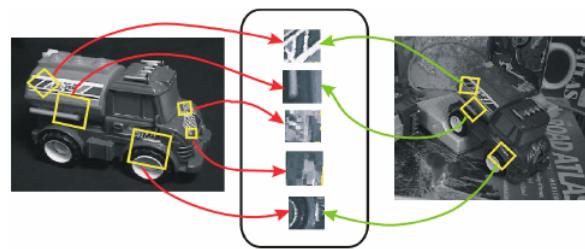
- Illumination**
- Scale**
- Rotation**
- Affine**
- Full Perspective**



83

Invariant Local Features

- Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters



81/59

© R. Siegwart, I. Nourbakhsh

Detection stages for SIFT features

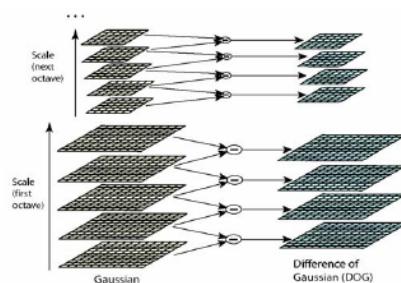
- Scale-space extrema detection
- Keypoint localization
- Orientation assignment
- Generation of keypoint descriptors.

82/59

© R. Siegwart, I. Nourbakhsh

Scale-space extrema detection

- The first step toward the detection of interest points is the convolution of the image with Gaussian filters at different scales, and the generation of Difference-of-Gaussian images (DoG) from the difference of adjacent blurred images

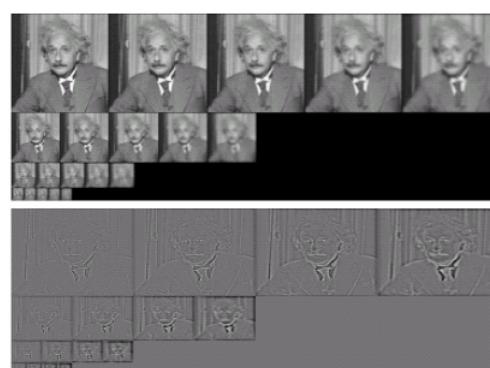


83/59

© R. Siegwart, I. Nourbakhsh

Scale-space extrema detection

- Gaussian blurred images grouped by octave
- Difference of Gaussian images (DoG) grouped by octave

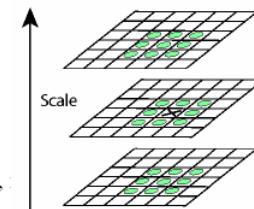


84/59

© R. Siegwart, I. Nourbakhsh

Scale-space extrema detection

- SIFT keypoints are identified as local maxima or minima of the DoG images across scales.
- Each pixel in the DoG images is compared to its 8 neighbors at the same scale, plus the 9 corresponding neighbors at neighboring scales.
- If the pixel is a local maximum or minimum, candidate keypoint.

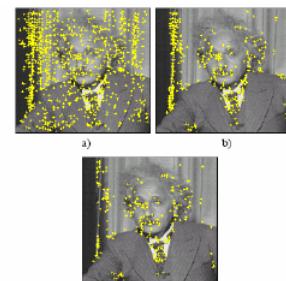


85/59

© R. Siegwart, I. Nourbakhsh

Key point localization

- For each candidate keypoint:
 - *Interpolation of nearby data is used to accurately determine its position.*
 - *Keypoints with low contrast are removed (b).*
 - *Responses along edges are eliminated (c).*

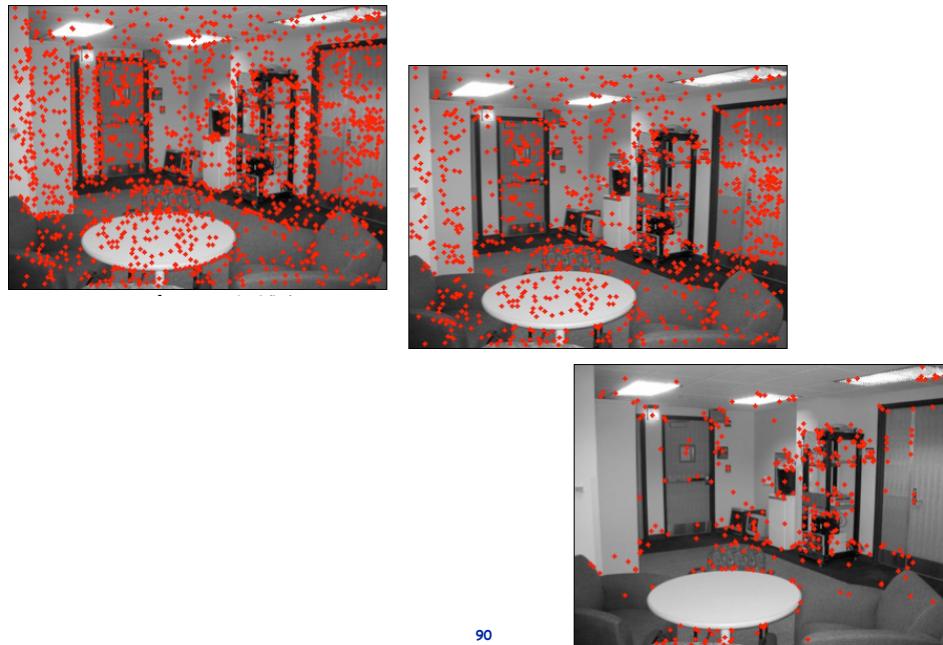


- a) Maxima of DoG across scales
 b) Remaining keypoints after removal of low contrast points
 c) Remaining keypoints after removal of edge responses

86/59

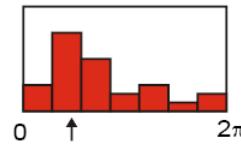
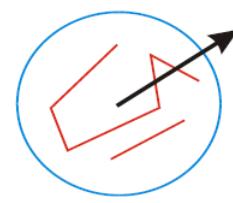
© R. Siegwart, I. Nourbakhsh

Remove low contrast and edges



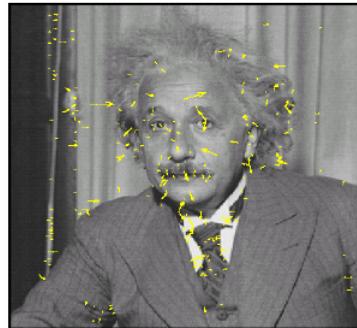
Orientation assignment

- To determine the keypoint orientation, a gradient orientation histogram is computed in the neighborhood of the keypoint.
- Peaks in the histogram correspond to dominant orientations. If more than one peak is found, a separated feature is assigned to the same point location.
- All the properties of the keypoint are measured relative to the keypoint orientation, this provides invariance to rotation.



Key point localization

- Final keypoints with selected orientation and scale

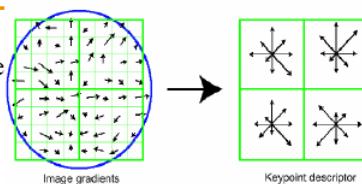


88/59

© R. Siegwart, I. Nourbakhsh

Generation of keypoint descriptor

- A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location, as shown on the left.
- These samples are then accumulated into orientation histograms summarizing the contents over 4x4 sub regions, as shown on the right, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region.
- The descriptor is formed from a vector containing the values of all the orientation histogram entries, corresponding to the arrows of the right side.
- This vector is then normalized to enhance invariance to changes in illumination.



89/59

© R. Siegwart, I. Nourbakhsh

Advantages of SIFT features

- Locality: features are local, so robust to occlusion and clutter (no prior segmentation)
- Distinctiveness: individual features can be matched to a large database of objects
- Quantity: many features can be generated for even small objects
- Efficiency: close to real-time performance

90/59

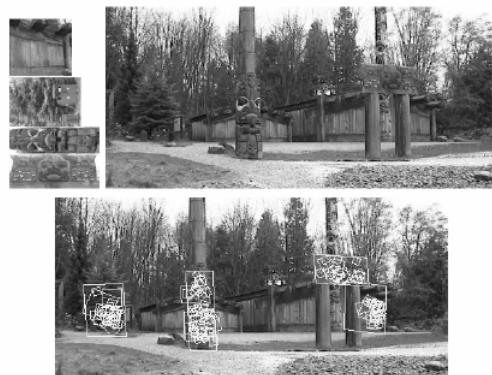
© R. Siegwart, I. Nourbakhsh

SIFT output: Feature Matching example



SIFT Detector Application Examples

Place recognition



97/59

© R. Siegwart, I. Nourbakhsh

SIFT Detector Application Examples

Planar recognition

- Planar surfaces can be reliably recognized at a rotation of 60° away from the camera
- Affine fit approximates perspective projection
- Only 3 points are needed for recognition



94/59

© R. Siegwart, I. Nourbakhsh

SIFT Detector Application Examples

Multiple panoramas from an unordered image set



Input images



Input images



© R. Siegwart, I. Nourbakhsh

99/59