Mechatronic Engineering program

Computer Vision
Image Objects and Features

Krzysztof Holak
AGH University of Krakow

1

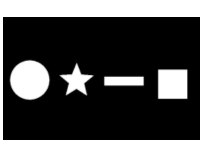
Schedule

- Lecture 1: An Introduction
- Lecture 2: Image Segmentation
- **Lecture 3: Image Features**
  - Geometric Features
  - Moments and Invariant Moments
  - Template Matching and Image Correlation
  - Point Features and Feature Descriptors
- Lecture 4: Video Processing

2

Image Analysis

To classify the objects in the image, first make a mathematical representation of them – compute **objects' features**



Area = ?
Centroid = ?
Euler Number = ?
Eccentricity = ?
Orientation = ?
…

→

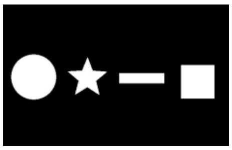
Is it a disk?
Is it a star?
Is it rectangle?
Is is a square?
…

3

Labelling

First you have to find how many there are on the image

Perform **labelling** – assign a natural number to each **disjoint object** in the image



4

Geometric features computation

In Matlab function regionprops provides a set of geometric features

Examples:

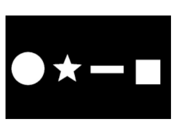
Area – How many pixels the object has (how big it is)
Centroid – it's centroid (it's position in the image)
Bounding Box – circumscribed rectangle (how much of space it takes)

Euler number – how many holes the object has
Circularity – how close the object is to a disk, how round it is
Eccentricity – how strech the object is – how far from being round

5

Geometric features computation

Examples



4x1 struct with 24 fields

Fields	Area	Centroid	BoundingBox	MajorAxisLengtl	MinorAxisLengtl	Eccentricity	Orientatio	ConvexAre	Circularity	FilledArea	EulerNumbe	EquivDiamete
1	12676	[87.5000,205.5000]	[23.5000,141.5000,128,128]	127.0473	127.0473	0	0	12776	1.0112	12676	1	127.0417
2	4389	[239.5095,208.5803]	[185.5000,151.5000,109,104]	83.8138	83.6805	0.0564	80.3601	7779	0.3614	4389	1	74.7546
3	3712	[393.5000,209]	[329.5000,194.5000,128,29]	147.8017	33.4863	0.9740	0	3712	0.5066	3712	1	68.7478
4	9025	[552,219]	[504.5000,171.5000,95,95]	109.6966	109.6966	0	0	9025	0.8369	9025	1	107.1960

4x1 struct with 22 fields

Fields	Solidity	Extent	Perimete	PerimeterOb	MaxFeretDiamete	MaxFeretAngle	MaxFeretCoordinates	MinFeretDiamete	MinFeretAngle	MinFeretCoordinates
1	0.9922	0.7737	396.8880	410.9605	128.4056	127.4054	[126.5000,154.5000,48.5000,256.5000]	127.2792	-45	[126.5000,154.5000,36.5000,244.5000]
2	0.5642	0.3872	390.6770	411.8894	109.1100	-107.6046	[272.5000,255.5000,239.5000,151.5000]	103.5184	-89.1319	[240.5000,151.5000,238.9317,255.0065]
3	1	1	303.4360	310	131.2440	167.2344	[457.5000,194.5000,329.5000,223.5000]	29	-90	[457.5000,194.5000,457.5000,223.5000]
4	1	1	368.1160	376	134.3503	135	[599.5000,171.5000,504.5000,266.5000]	95	-90	[599.5000,171.5000,599.5000,266.5000]

6

Moments

Image moments
Computed as in the case of section moments in structural mechanics

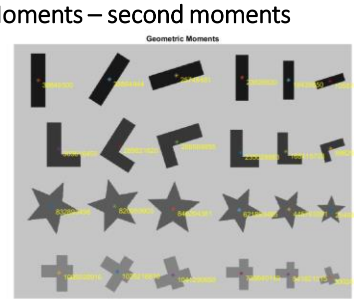
$$M_{pq} = \sum_x \sum_y x^p y^q I(x, y)$$

p, q – order of moment with respect to y and x axes

if p = 0 and q = 0 – object's area
If (p = 0 and q = 1) or (p = 1 and q = 0) – static moment – to compute centroid
If (p = 0 and q = 2) or (p = 2 and q = 0) - second moment, moment of inertia

7

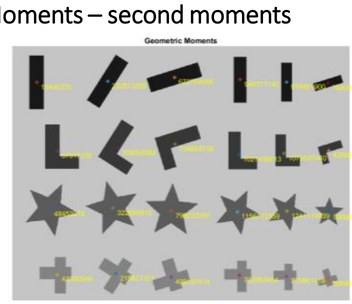
Raw Moments – second moments



p = 0
q = 2

8

Raw Moments – second moments



p = 2
q = 0

9

Central Moments

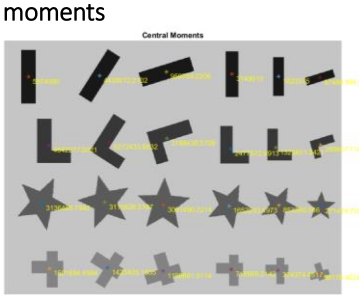
Raw (ordinary) moments **are not** shift, rotation and scale invariant!
To make tchem shift invariant – compute them about the object's **centroids**!

$$\mu_{pq} = \sum_x \sum_y (x-\bar{x})^p (y-\bar{y})^q I(x, y)$$

Central moment **do not** change as you shift objects in the image

10

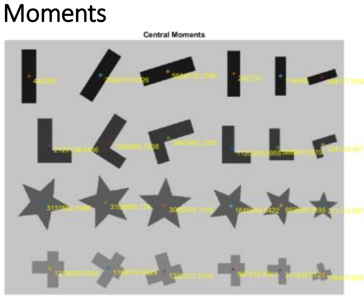
Central moments



p = 0
q = 2

11

Central Moments



p = 2
q = 0

12

Normalized Central Moments

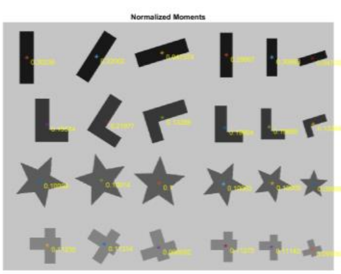
Normalize central moments to make them invariant to **scaling**

$$\varphi_{pq} = \frac{\mu_{pq}}{(\mu_{00})^{1+\frac{p+q}{2}}}$$

Moments are invariant to the **scale and shift** transformation of objects but **not** invariant to **rotations!**

13

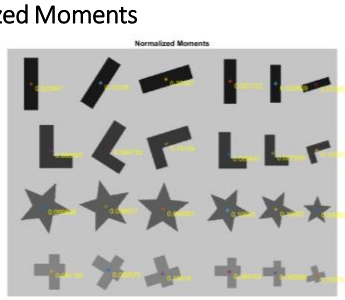
Normalized Moments



p = 0
q = 2

14

Normalized Moments



p = 2
q = 0

15

Hu Moments

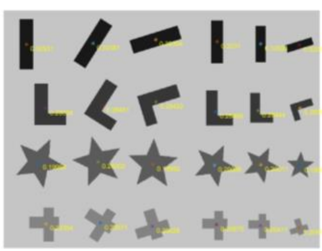
Seven moments that are invariant to **shift, scaling and rotation**

$$I_1 = \eta_{20} + \eta_{02}$$
$$I_2 = (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2$$
$$I_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$$
$$I_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$$
$$I_5 = (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$
$$I_6 = (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})$$
$$I_7 = (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]$$

16

Hu Moments

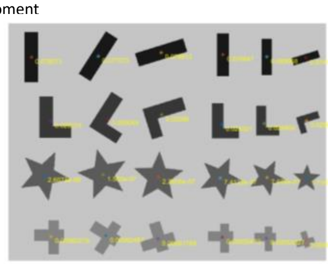
First Hu Moment



17

Hu Moments

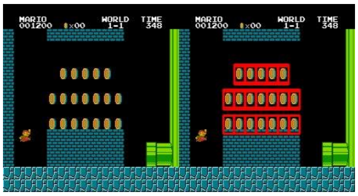
Second Hu Moment



18

Template macthing

There is a known pattern – an image path
Find the position of the pattern (patterns) in the image



Source: OpenCV

19

Template matching

Move template across the image – compute a similarity measure between template and the image patch under it

Summed Square Distance (SSD)

$$h[m,n]=\sum_{k,l}(g[k,l]-f[m+k,n+l])^2$$

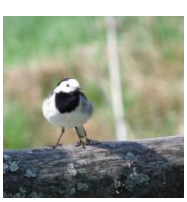
Normalized Cross Correlation (NCC)

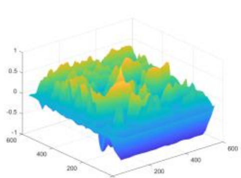
$$h[m,n]=\frac{\sum_{k,l}(g[k,l]-\bar{g})(f[m+k,n+l]-\bar{f}_{m,n})}{\left(\sum_{k,l}(g[k,l]-\bar{g})^2\sum_{k,l}(f[m+k,n+l]-\bar{f}_{m,n})^2\right)^{0.5}}$$

20

Image Correlation

Look at the correlation coefficient response

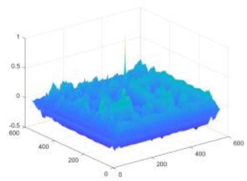




21

Image Correlation

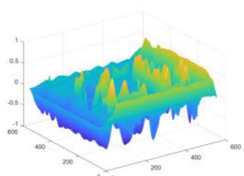
Is it better/worse to find?



22

Image Correlation

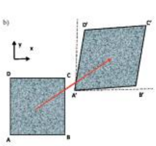
What about this one?

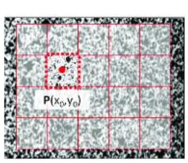


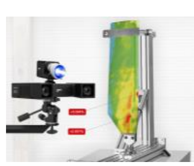
23

DIC – mechanical engineeering application

Digital Image Correlation (DIC) is a pratical application of image correlation idea







GOM Aramis System, Source: GOM

24

Local features

Geometric features are good for flat objects – in binary image processing
What about objects that appear in more general images?
cars, people, animals, buildings etc.

We need another approach to deal with such objects

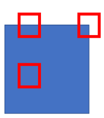
One possible solution – use **local features** describing the neighborhood of characteristic points – e.g. corners

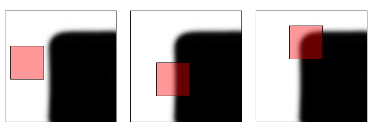
Then we can build objects' representation out of these **local features**

25

Corners in the image

Corners as base for **local feature descriptors**





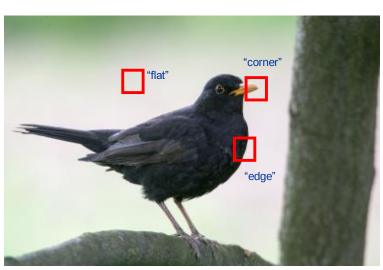
"flat" region: no change of brightnesss in all directions

"edge": no change of brightness along the edge

"corner": big change of brightness in all directions

26

Corners in the image



27

Harris Corner Detector

Idea: For each pixel, make an image pattern centered at that pixel

Move it a little bit in two directions and observe how the brightness of the pattern changes

In terms of math – for all pixels belonging to an image pattern, compute the measure:

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

Where dx and dy are shifts in x and y direction, I(x,y) is pixel intensity function and w(x,y) – is a windowing function (e.g. Gaussian blur filter mask)

28

Harris Corner Detector

We want to see how the function behaves for small shifts

Compute its local **quadratic** approximation

Use **Taylor series** expansion for I(x+dx,y+dy)

$$I(x + dx, y + fy) \approx I(x, y) + I_x(x, y)dx + I_y(x, y)dy$$

And after substituting into our equation we obtain

$$E(dx, dy) = \sum_{x,y} w(x, y) \, [I_x(x, y)dx + I_y(x, y)dy]^2$$

29

Harris Corner Detector

Now we write this in matrix form:

$$E(dx, dy) \approx [dx \quad dy] M \begin{bmatrix} dy \\ dy \end{bmatrix}$$

Where matrix M is called structural tensor

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

We can see that the elements of the matrix are computed from **image gradient** (derivatives in x and y directions) and w(x,y) is **Gaussian mask**

30

Harris Corner Detector

This is a quadratic form, an equation of an elipsoid

$$E(dx, dy) \approx [dx \quad dy] M \begin{bmatrix} dy \\ dy \end{bmatrix}$$

Since M is symmetric matrix, it can be diagonalized

$$M = R \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R^T$$

The eigenvalus of M reveal the intensity change in two principal orthogonal gradient directions in the window

31

Harris Corner Detector

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

Structural tensor matrix

Sum over all pixels in the neighbourhood

Kernel of smoothing mask – recall our discussion on **Gaussian mask**

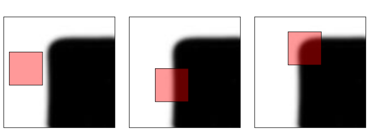
Matrix of squares of local image derivatives Computed by our **high-pass filter masks**

Matrix is diagonalized, there are two eigenvalues

32

Harris Corner Detector

Let's see values of eignevalue of M for our points of interest



"flat" region: λ_1 and λ_2 are small, $\lambda_1 \sim \lambda_2$;

"edge": $\lambda_1 >> \lambda_2$ or $\lambda_2 >> \lambda_1$

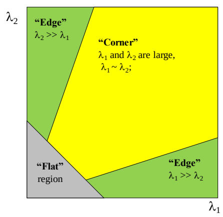
"corner": λ_1 and λ_2 are large, $\lambda_1 \sim \lambda_2$;

33

Harris Corner Detector

Harris gave us a simple cornerness score to assess both eigenvalues at the same time

$$R = \det(M) - \alpha \operatorname{trace}(M)^2 = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$



α: constant (0.04 to 0.06)

34

Harris Corner Detector

The Harris algorithm

1) Compute *M* matrix for each point of interest to get their *cornerness* scores.
2) Find points of interest whose surrounding window gave large corner response (*R*> threshold)
3) Take the points of local maxima, i.e., perform non-maximum suppression

C.Harris and M.Stephens. "A Combined Corner and Edge Detector." *Proceedings of the 4th Alvey Vision Conference*: pages 147—151, 1988.

35

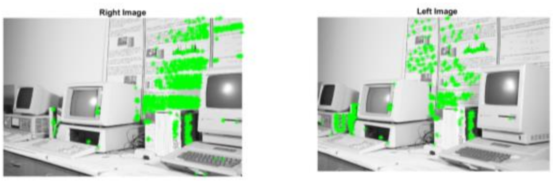
Applications of local features

Use automatic point matching algorithm (e.g. RANSAC method)

36

Applications of local features

Use local features for point matching between two or more images



37

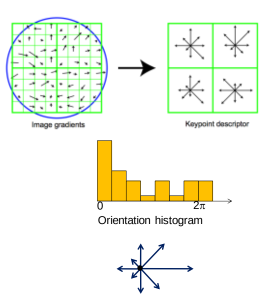
Applications of local features

Image Stitching – making panoramas
Object tracking in video sequence
Camera calibration
Stereovision – disparity and depth computation
Multiview tracking and 3D structure and pose reconstruction
Object recognition, detection and matching
Robot mapping and navigation
…

38

SIFT feature descriptor

Compute image gradient – magnitude and orientation
Quantize orientations
Divide image patch into cells
Compute orientation histogram in each cel
Make interest point descriptor

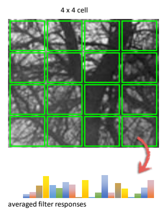


Distinctive Image Features from Scale-Invariant Keypoints. Lowe. In IJCV 2004

39

GIST feature descriptor

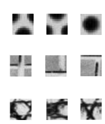
Compute response images to Gabor filter bank (N filters)
Divide image patch into 4x4 cells
Compute filter response average for each cel
Make descriptor for each image patch
Its size is 4x4xN

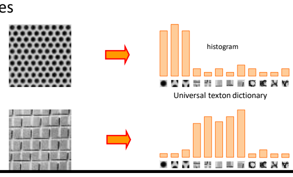


40

Texton feature descriptor

Prepare a database of texture elements – texton library
Find amount of each texture element in a given image patch – using template matching
Make histogram of textons based on template matching responses





41

Questions for Review

List three geometric features using in binary image analysis
What is a difference between raw (ordinary) and central moment?
What is a difference between central and normalized moment?
Which moments are invaraint to scale/rotation/shift?
What is the main application of image correlation?
What are local features?
What kind of image features are detected by Harris algorithm?
What are the applications of local features?
Which linear filters are used in Harris detector algorithm?

42

Thank you for attention

43