Faculty of Informatics
Eötvös Loránd University
Budapest, Hungary

Basic Algorithms for Digital Image Analysis: a course

Dmitrij Csetverikov

with help of Attila Lerch, Judit Verestóy, Zoltán Megyesi, Zsolt Jankó and Levente Hajder

http://visual.ipan.sztaki.hu

Lecture 5: Finding Patterns in Images

- Matching and correspondence in computer vision
- Template matching
  - Similarity and dissimilarity measures
  - Interior matching versus contour matching
  - Invariance
  - Distortion-tolerant matching
  - Stable matching
  - Fast implementations

Matching and correspondence in computer vision

Image matching: Finding **correspondences** between two or more images.

Basic tasks of computer vision related to matching:

1. Given images of a scene taken by **different sensors**, bring them into registration.
   - This is called **image registration**.
   - Typical example: Medical imaging
     - Images obtained by **sensors of different types** are called **modalities**.
2. Given images of a scene taken at **different times**, find correspondences, displacements, or changes.
   - This is **motion analysis**.
   - Typical example: Motion tracking.

3. Given images of a scene taken from **different positions**, find correspondent points to obtain 3D information about the scene.
   - This is stereopsis, or simply **stereo**.
     - Matching provides **disparity**: the shift of a point between the two views
     - By triangulation, disparity and baseline (distance between eyes) provide **depth**: the 3D distance to the point.
   - Generalised stereo is called **3D scene reconstruction** from multiple views.
4. Find places in an image or on a contour where it **matches a given pattern**.
   - **Template matching**: Pattern specified by template.
   - **Feature detection**: Feature specified by description.
5. Match **two contours** for object recognition, measurement, or positioning.
   - This is **contour matching**.

Key issues of matching:

- **Invariance** under imaging transformations
  - spatial
  - photometric (intensity)
- **Sensitivity** to noise and distortions

Considered in this course are:

- Tasks
  - Task 4: **Template matching and feature detection**
  - Task 5: **Contour matching**
- Transformations
  - Spatial: 2D shift and rotation
  - Photometric: Shift and scaling of intensity (linear)

Template matching

Compare a subimage (**template**) $w(r', c')$ with an image $f(r', c')$ for all possible displacements (r, c).
In other words: **Match** $w(r', c')$ **against** $f(r + r', c + c')$ for all (r, c).

Measures of dissimilarity between image f and template w in (r, c):

D1 Sum of Square Differences (**SSD**):

$$D(r,c) = \sum_{\substack{(r',c') \in W \\ (r+r',c+c') \in F}} \Big\{ f(r+r', c+c') - w(r', c') \Big\}^2$$

- W: set of pixel positions in template w (template coordinates)
- F: set of pixel positions in image f (image coordinates)

$D(r, c)$ is **not invariant** under the following **transformations**

- 2D rotation $\Rightarrow$ Cannot find significantly rotated pattern
- Shift or scaling of intensity $\Rightarrow$ Cannot cope with any varying illumination

D2 Intensity **shift-corrected SSD**:

$$\delta(r,c) = \sum_{\substack{(r',c') \in W \\ (r+r',c+c') \in F}} \Big\{ \Big[f(r+r', c+c') - \overline{f}(r,c) \Big] - \Big[w(r', c') - \overline{w} \Big] \Big\}^2$$

- $\overline{f}(r, c)$: Average value of image in region covered by template
  - computed in **each position** (r, c)
- $\overline{w}$: Average value of template
  - computed **only once**

$\delta(r, c)$ is a bit more sophisticated measure used to **compensate for intensity shift** due to uneven illumination.

- Handles **changes in average level** of signal
- Cannot handle **changes in amplitude** of signal

Measures of similarity between image f and template w in position (r, c):

S1 Unnormalised cross-correlation (CC) of image f with template w:

$$C_{un}(r,c) = \sum_{\substack{(r',c') \in W \\ (r+r',c+c') \in F}} f(r+r', c+c') \cdot w(r', c')$$

- We have already studied the properties of cross-correlation and convolution.
- $C_{un}(r, c)$ is formally the same as **filtering** image f with mask w.
  $\Rightarrow$ Our knowledge of filters **is applicable**, including normalisation, separability, fast implementation, etc.
- $C_{un}(r, c)$ is **not invariant** under intensity shift and scaling. When $w > 0$ and f is large, $C_{un}(r, c)$ is large, independently from (dis)similarity between w and f.
  $\Rightarrow$ To compensate for this, a normalised version is used.

S2 Normalised cross-correlation (NCC), or correlation coefficient:

$$C_{nr}(r,c) = \frac{1}{\sqrt{S_f(r,c) \cdot S_w}} \sum \Big[f(r+r', c+c') - \overline{f}(r,c) \Big] \cdot \Big[w(r',c') - \overline{w} \Big]$$

where

$$S_f(r,c) = \sum \Big[f(r+r', c+c') - \overline{f}(r,c) \Big]^2, \qquad S_w = \sum \Big[w(r',c') - \overline{w} \Big]^2$$

and for simplicity

$$\sum \text{ denotes } \sum_{\substack{(r',c') \in W \\ (r+r', c+c') \in F}}$$

- $S_f(r,c)$ is computed in **each position** (r,c), S_W is computed in **only once**.
- $C_{nr}(r,c)$ is **invariant to any linear intensity transformation**.
- If the average values are **not** subtracted, $C_{nr}(r,c)$ is only intensity scale-invariant (scale-corrected).

S3 Modified normalised cross-correlation (MNCC):

$$C_{mnr}(r,c) = \frac{1}{S_f(r,c) + S_w} \sum \Big[f(r+r', c+c') - \overline{f}(r,c) \Big] \cdot \Big[w(r',c') - \overline{w} \Big]$$

where as before

$$S_f(r,c) = \sum \Big[f(r+r', c+c') - \overline{f}(r,c) \Big]^2, \qquad S_w = \sum \Big[w(r',c') - \overline{w} \Big]^2$$

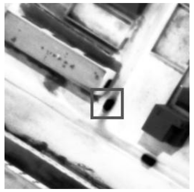
- C_{mnr} is **another normalisation**:

  C_{nr} is divided by $\sqrt{S_f(r,c) \cdot S_w}$
  C_{mnr} is divided by $S_f(r,c) + S_w$

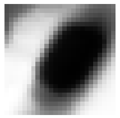
- C_{mnr} is used instead of the standard C_{nr} to avoid the numerically unstable division by a small number when $S_f(r,c)$ is small. (Small image variation.)

- Formally, C_{mnr} is only shift-corrected. In practice, C_{mnr} is reasonably insensitive to intensity scaling as well, since S_w is constant and $S_f(r,c) + S_w$ is roughly proportional to $S_f(r,c)$.

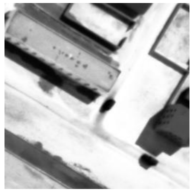
Template matching: Varying r and c, search for locations of **high similarity** $C_{un}(r,c)$, $C_{nr}(r,c)$, $C_{mnr}(r,c)$, or **low dissimilarity** $D(r,c)$, $\delta(r,c)$.



Left image

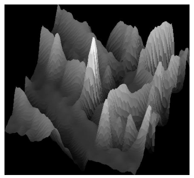


Template, zoomed



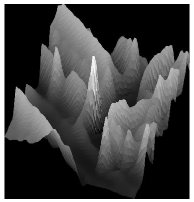
Right image

NCC image



NCC surface

SDD image



SDD surface

Examples of matching in stereo pair. Pattern from left image is searched in right image. NCC is Normalised Cross-Correlation, SSD is Sum of Square Differences.

Interior matching versus contour matching

template

0	0	0
1	1	1
0	0	0

input image

0	0	0
1	1	1
0	0	0

output of CC

1	2	3	2	1

output of NCC

1.0	1.4	1.7	1.4	1.0

1	1	1
1	1	1
1	1	1

1	2	3	2	1
1	2	3	2	1
1	2	3	2	1

1.0	1.2	1.0
	1.0	
1.0	1.2	1.0

0	1	0
0	1	0
0	1	0

1	1	1
1	1	1
1	1	1

1	1	1
1	0	1
1	1	1

1	2	3	2	1
1	1	2	1	1
1	2	3	2	1

1.2	1.3	1.2
1.2	1.3	1.2

Numerical examples of matching by unnormalised (CC) and normalised (NCC) cross-correlations. In output, values below 1 are set to 0 and not shown.

Observation in the numerical example: The perfect match value (1.7) is not much better than the near misses in position and shape.

- **The match is not sharp.**

Matching of the **outlines** yields **sharper** matches:

input image

1	1	1
1	1	1
1	1	1

template

0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

output of NCC

		1.2		
	1.3	2.0	1.3	
1.2	2.0	3.0	2.0	1.2
	1.3	2.0	1.3	
		1.2		

input image

1	1	1
1	0	1
1	1	1

template

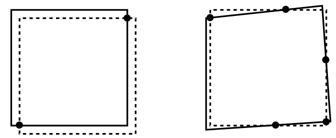
0	0	0	0	0
0	1	1	1	0
0	1	0	1	0
0	1	1	1	0
0	0	0	0	0

output of NCC

		1.3		
		1.4		
1.3	1.4	2.8	1.4	1.3
		1.4		
		1.3		

Trade-off between **localisation accuracy** and **reliability** of matching

- Matching the contours: faster, yields sharp matches, but sensitive to distortions;
- Matching the interior: slower and less sharp, but more robust.



Contours matching versus interior matching. Template: Dashed rectangle. Object: Solid line. Circles: Overlapping points of contours.

- Left: Small shift of template results in drastic decrease of contour overlap and negligible descrease of area overlap.

  $\Rightarrow$ Contour matching is sharper.

- Right: Distortion of pattern has a similar effect.

  $\Rightarrow$ Contour matching is less robust.

Critical issues in template matching

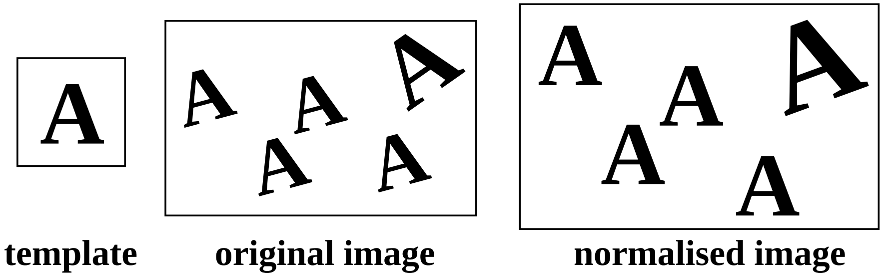
- Sensitivity to **changes in size and rotation**
- Sensitivity to **pattern distortion**
  - For example, because of varying viewing angle
- **'Noisy' matches**: Unexpected configurations may occur that produce high matching values
- Heavy **computational load**

Handling variations in size and orientation

Options:

- Normalisation: Transform image to **standard size and orientation**
  - Works only if there is no size or orientation variation **within the image**
  - Requires **definition of orientation**
- Adaptivity: Spatially **scale and rotate** the template in each position, select the best matching scale and rotation
  - Very slow if number of scales and rotations is large

  $\Rightarrow$ Used only for small number of scales and rotations
- Alternative solution: Use scale and rotation **invariant description**
  - Compare descriptions instead of patterns



Normalising an image for size and orientation.

- The letter **A** in the top right corner differs in size and orientation.

⇒ This letter **will not match**.

- The other four letters will match.

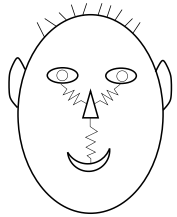
- How to define **image orientation**?

Distortion-tolerant matching

Use **flexible templates** composed of spatially connected subtemplates with flexible links ('springs').

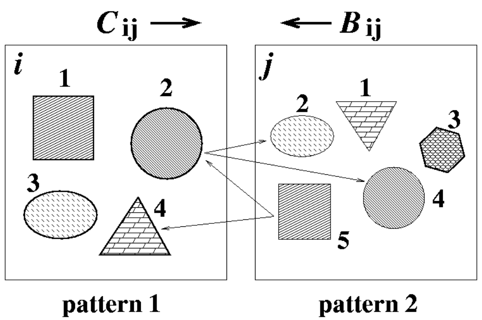
- The springs allow for a moderate spatial variation of the template.
  - A **cost function** is introduced to penalise large variations
  ⇒ The larger the variation the larger the penalty

- Works well when the subtemplates are characteristic enough for reliable matching.



Representing a face template as a set of flexibly connected subtemplates.

Matching segmented patterns



Matching two patterns by segmenting them into regions.

- Segment patterns into regions and find correspondences by **comparing region properties**.
  - A **distance measure between properties** of regions should be defined.

- This solution works well when the segmentation is reliable.

Algorithm 1: Stable Matching of Two Images

1. Compute **distance matrix** D_{ij}; i: i^{th} region of image 1, j: j^{th} region of image 2.

2. Calculate **forward matching matrix** C_{ij}: $C_{ij} = 1$ if $D_{ij} < D_{ik}$ for all $k \neq j$; otherwise, $C_{ij} = 0$.

3. Calculate **backward matching matrix** B_{ij}: $B_{ij} = 1$ if $D_{ij} < D_{kj}$ for all $k \neq i$; otherwise, $B_{ij} = 0$.

4. Match regions i and j if $C_{ij}B_{ij} = 1$.

5. Remove established correspondences from D_{ij}.

6. Iterate until no further matching is possible.

Comments to the Stable Matching algorithm:

- The **backward matching** (steps 2–4) is a **consistency check**.

  ⇒ This is a standard way to discard noisy (unreliable or erroneous) matches

- The **iterative procedure** is based on an algorithm for the Stable Marriage Problem.

left image | right image | original ME | consistent ME

Matching a stereo pair in presence of occlusion. ME is the matching error. The consistency check removes wrong matches due to occlusion.

Fast impementations of matching

- Work with **local features** of images and templates rather that the patterns themselves
  - For example: Edges, contours
  - Useful for sparse and reliable features
  - Caution: Remember sensitivity to distortions!

- For large templates ($> 13 \times 13$ pixels), use implementation of cross-correlation via **Fast Fourier Transform** (FFT):

$$f \otimes w = IFFT\Big[FFT\big[f(x,y)\big]^* \cdot FFT\big[w(x,y)\big]\Big],$$

where $IFFT$ is the inverse FFT and X^* is the complex conjugate of X.

  - Needs $O(N^2 \log N)$ operations for $N \times N$ images
  - Straightforward implementation needs $O(N^4)$ operations

Another solution: Use a fast procedure to

1. **Select** match **candidates** and **reject mismatches** rapidly, then
2. Test the selected candidates

Options for fast selection and rejection:

- Use a coarsely spaced **grid of template positions**, then rectify the candidates.
  - This is a **coarse-to-fine** sampling method for the cross-correlation function
  - It works if peaks of cross-correlation are **smooth and broad** (no spikes).

- Compute **simple properties** of template and image region. Reject region if its properties differ from properties of the template.

- Use **subtemplates to reject a mismatch** rapidly when a subtemplate does not match.

- If a cumulative measure of mismatch is used, reject a candidate when the mismatch **exceeds a preset threshold**.