Basic Algorithms for Digital Image Analysis: a course

Dmitrij Csetverikov

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

http://visual.ipan.sztaki.hu
Lecture 6: Template Matching and Feature Detection

- Template matching
  - Similarity and dissimilarity measures
  - Interior matching versus contour matching
  - Invariance
  - Distortion-tolerant matching
  - Stable matching
  - Fast implementations

- Types of local image features
  - Edges
  - Lines
  - Corners
  - Blobs
Interior matching versus contour matching

<table>
<thead>
<tr>
<th>Template</th>
<th>Input Image</th>
<th>Output of CC</th>
<th>Output of NCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>0 0 0</td>
<td>1 2 3 2 1</td>
<td>1.0 1.4 1.7 1.4 1.0</td>
</tr>
<tr>
<td>1 1 1</td>
<td>1 1 1</td>
<td>1 2 3 2 1</td>
<td>1.0 1.2 1.0</td>
</tr>
<tr>
<td>0 0 0</td>
<td>0 0 0</td>
<td>1 2 3 2 1</td>
<td>1.0 1.2 1.0</td>
</tr>
<tr>
<td>0 1 0</td>
<td>0 1 0</td>
<td>1 1 1</td>
<td></td>
</tr>
<tr>
<td>0 1 0</td>
<td>0 1 0</td>
<td>1 1 1</td>
<td></td>
</tr>
<tr>
<td>0 1 0</td>
<td>0 1 0</td>
<td>1 1 1</td>
<td></td>
</tr>
<tr>
<td>1 1 1</td>
<td>1 1 1</td>
<td>1 2 3 2 1</td>
<td>1.2 1.3 1.2</td>
</tr>
<tr>
<td>1 0 1</td>
<td>1 1 2 1 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1 1</td>
<td>1 2 3 2 1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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:

<table>
<thead>
<tr>
<th>input image</th>
<th>template</th>
<th>output of NCC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 1 1 1</td>
<td>0 0 0 0 0</td>
<td>1.2</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>0 1 1 1 1 0</td>
<td>1.3 2.0 1.3</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>0 1 1 1 1 0</td>
<td>1.2 2.0 3.0 2.0 1.2</td>
</tr>
<tr>
<td>1 1 1 1</td>
<td>0 0 0 0 0</td>
<td>1.3 2.0 1.3</td>
</tr>
</tbody>
</table>

| 1 1 1 1     | 0 0 0 0 0 | 1.3           |
| 1 1 1 1     | 0 1 1 1 1 0| 1.3 1.4       |
| 1 0 1 1     | 0 1 0 1 1 0| 1.3 1.4 2.8 1.4 1.3 |
| 1 1 1 1     | 0 1 1 1 1 0| 1.4           |
| 1 1 1 1     | 0 0 0 0 0 | 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.


- Left: Small shift of template results in drastic decrease of contour overlap and negligible decrease of area overlap.
  ⇒ Contour matching is sharper.

- Right: Distortion of pattern has a similar effect.
  ⇒ 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
  - 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

- 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.

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

- Work with **local features** of images and templates rather than 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):

\[
    f \otimes w = \text{IFFT} \left[ \text{FFT} [f(x,y)]^* \cdot \text{FFT} [w(x,y)] \right],
\]

where $\text{IFFT}$ 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.
Types of local image features considered in this course:

- Edges – More detail
- Lines – Less detail
- Corners – Less detail
- Blobs – No details
Edges and blobs

An image edge is a drastic change of intensity across object contour.

- **Image edges** do not necessarily coincide with **physical edges**
  - Image edges are intensity discontinuities
  - Physical edges are surface discontinuities
  - Example: Edges of **shadows** are not surface discontinuities

- Importance of intensity edges: Human eye detects them ‘in hardware’, at the initial level of visual processing.

A **blob** is a compact image region of approximately constant intensity.

- Blobs are elementary patterns used to build more sophisticated patterns.
  - Example: Eyes in a face model

- Blob detectors exist, but they are less frequently used than edge and corner detectors.
A line is a narrow, elongated image region of approximately constant width and intensity.

- Formally, a line can be viewed as two parallel sequences of edges.
  - In practice, a thin line is rarely detected like that.

- Blurred lines and other linear objects may have different cross-sections.
  - Example: Roof-shaped.

- Two different operations are often called line detection:
  - Line filtering (enhancement), detecting pieces of lines. This is a local operation similar to edge detection.
  - Detecting lines (curves) of a given shape. This is a global operation, whose typical example is the Hough transform.
Corners

A corner is a sharp turn of a contour.

- Corners are used in shape analysis and motion analysis

- Corners and other points of high curvature are dominant in human perception of 2D shapes
  - Shapes can be approximately reconstructed from their dominant points

- Two different operations but related operations are called corner detection:
  - Detection of corners in digital curves
    ⇒ This assumes extracted contours
  - Detection of corners in greyscale images
    ⇒ This does not assume extracted contours
The aperture problem and the use of corners in motion analysis. The displacement vectors are ambiguous at an edge, but unambiguous at a corner.

The Attneave’s Cat. The original smooth shape has been restored based on a small number of high curvature points. The cat is easy to recognise.
Three image features and their intensity profiles along the indicated lines.