Chapter 8

Shape representation and description

8.1 Matching

- we wish to match some model to the data
  - At simplest, known binary patterns representing characters of a font may be sought in properly aligned scans of text—this is template matching applied to OCR.
  - More generally, font-independent OCR demands recognition of characters of unknown font and size, perhaps with skew—this requires the matching of the *pattern* of characters.
  - More generally still, face recognition requires the matching of the *pattern of a face* into a picture of a 3D scene: pose, alignment, scale, beards, spectacles, color will all be unknowns.
  - At the most abstract, perhaps a pedestrian has been matched in a video sequence, and we seek to match the individual's *behavior* to some known model—is the pedestrian
    crossing a road?
    queuing?
    acting suspiciously?

- each of these requires matching a model pattern M to some observation from the image(s) X
- algorithm used may be elementary when the problem is straightforward or extremely complex (e.g., behavior matching)
- matching algorithms are usually based on some criterion of optimality
  - Hough transform
  - Shape invariants
  - Snakes
  - Graph matching
  - PDMs/AAMs
  - Correspondence
  - Hypothesize and verify
  - other general approaches – discussed below
- Note relationships with image registration

8.1.1 Template matching

- locating a known object in an image is to seek its pixel-perfect copy
- implies no variation in scale or rotation and is artificially simple
- goal to match a *template*—the known image
- given a template T of dimension $r_T \times c_T$ and an image I, we will hold it at offsets $\mathbf{x} = (x_a, x_b)$
- if the template fits perfectly

$$E(\mathbf{x}) = \sum_{i=1}^{r_T} \sum_{j=1}^{c_T} (T_{i,j} - I_{x_a+i, x_b+j})^2 = 0\,, \tag{8.1}$$

E measures the error of the fit

- local minima of $E(\mathbf{x})$ will give indication of quality of template fit

$$\begin{aligned} E(\mathbf{x}) &= \sum_{i=1}^{r_T}\sum_{j=1}^{c_T}(T_{i,j} - I_{x_a+i,x_b+j})^2 \\ &= \sum_{i=1}^{r_T}\sum_{j=1}^{c_T}(T_{i,j})^2 - 2\sum_{i=1}^{r_T}\sum_{j=1}^{c_T}(T_{i,j}I_{x_a+i,x_b+j}) + \sum_{i=1}^{r_T}\sum_{j=1}^{c_T}(I_{x_a+i,x_b+j})^2 \end{aligned} \quad (8.2)$$

  the first term is constant
  the third is in most circumstances slowly varying with $\mathbf{x}$

- Template matching may thus be performed by maximizing the correlation expression

$$Corr_T(\mathbf{x}) = \sum_{i=1}^{r_T}\sum_{j=1}^{c_T}(T_{i,j}I_{x_a+i,x_b+j})\,. \quad (8.3)$$

  the summation is sensitive both to intensity range and size of the region T — use of spatial and/or intensity scaling parameters may be in order

- partial pattern positions, crossing the image borders, and similar special cases may have to be considered

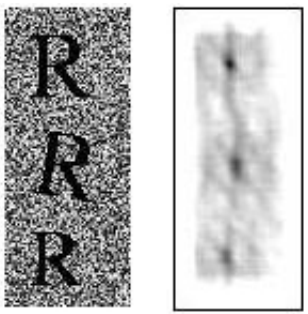


Figure 8.1: Template matching: A template of the letter **R** is sought in an image that has itself, a slightly rotated version, and a smaller version. The correlation response (contrast stretched for display) illustrates the diffuse response seen for even small adjustments to the original.

- this is severely limited—very small rotations of the template or changes in scale can cause radical jumps in the 'error' measure E.

- An alternative criterion for the same idea to minimize E in Equation 8.1 might be to maximize

$$C(\mathbf{x}) = \frac{1}{1 + E(\mathbf{x})} \,. \tag{8.4}$$

- 2 examples show the use of this criterion:

$$\begin{vmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 8 \end{vmatrix}$$

(a)

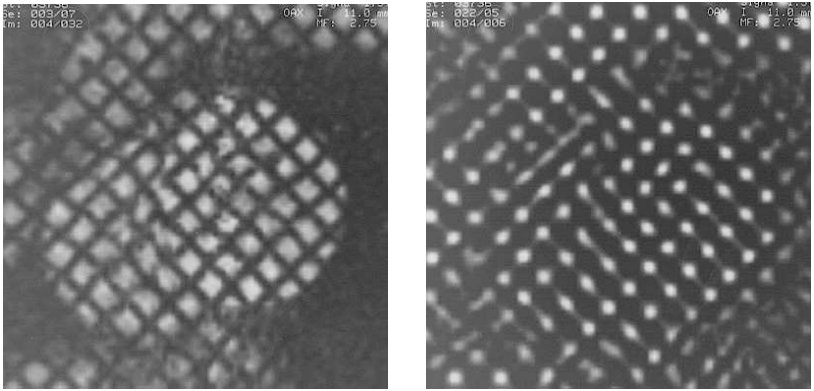
$$\begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}$$

(b)

$$\begin{vmatrix} \underline{\mathbf{1/3}} & 1/6 & 1/8 & \times & \times \\ \underline{1/5} & 1/7 & 1/8 & \times & \times \\ 1/8 & 1/9 & 1/57 & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{vmatrix}$$

(c)

Figure 8.2: Optimality matching criterion evaluation. (a) Image data. (b) Matched pattern. (c) Values of the optimality criterion C (the best match underlined).



(a) (b)

Figure 8.3: X-shaped mask matching. (a) Original image. (b) Correlation image; the better the local correlation with the X-shaped mask, the brighter the correlation image. *Courtesy of D. Fisher, S. Collins, The University of Iowa.*

- Fourier convolution theorem provides an efficient way of computing the correlation of a template and an image—to compute the product of two Fourier transforms, they must be of the same size
- — a template may have zero-valued lines and columns added to inflate it to the appropriate size
- — it may be better to add non-zero numbers, for example, the average gray-level of processed images.

- A related approach uses the chamfer image (which computes distances from image subsets – a distance transform image)
- — locates features such as known boundaries in edge maps
- — construct a chamfer (distance transform) image from an edge detection of the image under inspection
  — then any position of a required boundary can be judged for fit by summing the corresponding pixel values under each of its component edges in a positioning over the image
  —low values will be good and high poor
- chamfering will permit gradual changes in this measure with changes in position, so standard optimization techniques can be applied to its movement in search of a best match.

8.1.2 Control strategies of templating

- it is unusual for a known object to appear 'pixel perfect' in an image
- however, *components* of it—which may be quite small—may appear so
- If larger pattern is composed of these components connected by elastic links, the match of the larger pattern will require stretching or contraction of these links to accord with identification of the smaller components
- a good strategy is to look for the best partial matches first, followed by a heuristic graph construction of the best combination of these partial matches in which graph nodes represent pattern parts.

- Template-based segmentation is time consuming even in the simplest cases — process can often be accelerated
- the sequence of match tests must be data driven
- fast testing of image locations with a high probability of match may be the first step; then it is not necessary to test all possible pattern locations
- another speed improvement can be derived if a mismatch can be detected before all the corresponding pixels have been tested
- if a pattern is highly correlated with image data in some specific image location, then typically the correlation of the pattern with image data in some neighborhood of this location is also good
- $\longrightarrow$ correlation changes slowly around the best matching location (Figure 8.1)
- ... matching can be tested at lower resolution first, looking for an exact match in the neighborhood of good low-resolution matches only

- Mismatches should be detected as early as possible since they are found much more often than matches
- in Equation 8.4, testing in a specified position must stop when the value in the denominator (measure of mismatch) exceeds some preset threshold
- — this implies that it is better to begin the correlation test in pixels with a high probability of mismatch in order to get a steep growth in the mismatch criterion
- this criterion growth will be faster than that produced by an arbitrary pixel order computation.

8.1.3 SIFT

- Template matching approaches do not work in real-world problems.
- ... objects are subject to scale, pose and illumination variation, partial occlusion

- **SIFT**—the Scale Invariant Feature Transform [Lowe, 2004]— extracts *stable* points from images and attaches to them robust features
- a small subset of these, with geometric coherence, suffice to confirm a re-identification of objects in other images

- SIFT proceeds in three phases:
  - key location detection to identify 'interest points'
  - feature extraction to characterize them
  - matching of feature vectors between models and images

Key location detection

- 'Key locations' of an image are points within it that we might reasonably expect to appear in further images of the same object or scene
- —corners are an obvious example.

- In image I_0 – determined as maxima or minima of a DoG filter applied at all pixels of an image pyramid.
- the bottom of the pyramid is the original image, to which Gaussian filters with $\sigma = \sqrt{2}$ and $\sigma = 2$ are applied to give images A_0 and B_0 respectively
- $A_0 - B_0$ is then a DoG filter with ratio $\sqrt{2}$

- next layer of the pyramid is formed by re-sampling B_0 with a pixel spacing of 1.5.
  (These operations are efficient: the Gaussians can be separated into 1D convolutions, and the 1.5 reduction is simple to implement)

- Local extrema determined in 3×3 windows at levels of this pyramid
- if such an extremum is also greater/smaller than elements of the 3×3 windows at the corresponding positions above and below, then the pixel is maximal/mimimal in three dimensions and is tagged as a key location
- —note that the central pyramid layer of the extremum captures the *scale* at which the pixel is 'key'
- it delivers very stable points repeatedly

Feature extraction

- Given the key locations, we seek to derive a reliable feature vector to describe its immediate locality
- —this needs to take account of local edge directions and strengths
  - canonical direction is associated with each one
  - — a very simple edge detector determines an edge direction R_i and magnitude M_i at each pixel of the images A_i
  - — small magnitudes are neglected.
  - Gaussian weighed window with σ three times that of the current scale is created
  - — with the weights multiplying the thresholded magnitudes
  - 36-bin histogram of directions R relative to the key location edge direction is accumulated with respect to these weights
  - — and the canonical direction is then defined by the dominant peak in this histogram
  - if the histogram has competing peaks, they are all accepted by duplicating the keypoint with multiple orientations

- this approach delivers stable keys and directions in the presence of noise, contrast and brightness distortion, and affine projection
- 500×500 image typically generates over 1000 such points

- An 8×8 window around the point has its edge magnitude and orientation values blurred
- — a 4×4 array of 8-wide edge orientation histograms is compiled

- Histogram contributions are the edge magnitudes weighted by a Gaussian centered at the key location

- We now have a 128-D vector: this is normalized to compensate for contrast variation; very large elements are neglected (and the vector renormalized) to compensate for a variety of common lighting change effects.

Matching

- Suppose we have a set of sample, or model, images each represented by some number of 128-D vectors as described above.
- We may seek their appearance, or partial appearance, in a test image which has also generated a set of vectors.
- For each test vector, we locate its nearest neighbor in the union of sample vectors
- — it may represent noise or some feature not in the training set
- — matches are rejected if the ratio between the distances to nearest and next-nearest neighbors is greater than some threshold (0.8 is reported as good), and this successfully rejects a high proportion of spurious matches.

- Nearest-neighbor location is obviously a computational load, and much effort has been devoted to efficient approaches to this general problem.
- Any assumed match gives a candidate location, scale and orientation of the model.

- A Hough-like voting procedure with wide bins—the original paper uses 30^o for orientation, 2 for scale and 0.25 of model dimension for location—then collects multiple identifications of candidates.
- These Hough bins are sorted on occupancy and each candidate subjected to a verification.
- Each match provides a model point (x, y) and an image point (u, v).
- In many real-world circumstances it is reasonable to assume the image gives an *approximately* affine transform of the model, and so

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} . \tag{8.5}$$

- Each match provides 2 equations for 6 unknowns, so provided the bin has a population of 3 or more this can be solved;
- — with more than 3 it is overdetermined and a least-squared error solution can be found.
- The solution can be checked against observed matches; outliers are rejected and the transform recomputed—the match is rejected if this reduces the number of candidate matches to less than 3.

Algorithm 8.1: Scale Invariant Feature Transform—SIFT

1. For an image I_0, derive A_0, B_0 by convolving with Gaussians of $\sigma = \sqrt{2}, 2$ respectively.
2. Keep a DoG filter of I_0 as $A_0 - B_0$.
3. Build an image pyramid by letting I_{i+1} be a 1.5 re-sampling of B_i.
4. Locate pyramid positions at which the DoG is extremal horizontally, and at layers above and below. These are *key locations*.
5. For each key location (at level i) determine a canonical direction as the maximum of a binned direction histogram, with respect to a suitably weighted window of edge magnitudes of A_i.
6. Describe each key location by a 128-D vector which characterizes intensity magnitude and direction in a 8×8 neighborhood.
7. **Matching:** Determine plausible 128-D matches between model and image as efficiently as possible. Accumulate candidate instances of the model in the image in a Hough-like manner. Test candidates and reject outliers. Retain populous candidates as matches.

- SIFT proves extraordinarily robust
- — the requirement for no more than three matches to define a usable transform permits very significant occlusion since many more than 3 key points are customarily available.
- Matches can also usually be found through very significant distortion due to perspective projection and illumination changes.

- The following examples were generated with publicly available software from http://www.cs.ubc.ca/~lowe/keypoints.
- the model image of a book (235×173) is shown top left, and the 241 SIFT key locations at bottom left, where the arrow lengths indicate scale and orientation.

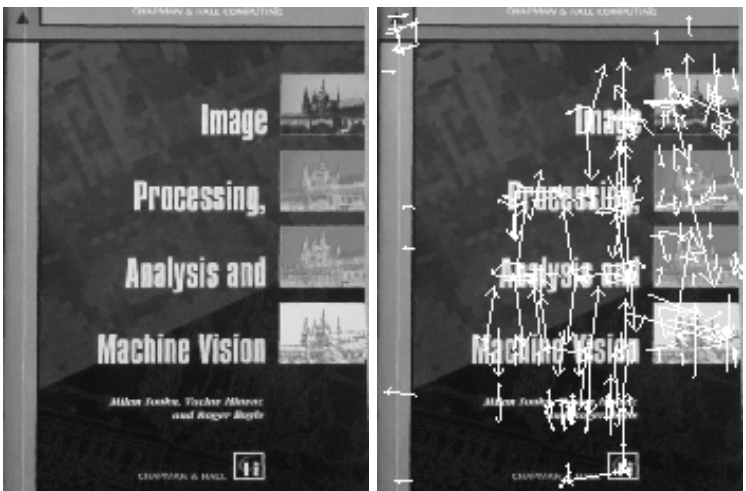


Figure 8.4: A model (left), and its SIFT key locations (right); the arrows indicate orientation (in their direction) and scale (in their length).

- A 473×455 scene including an occluded occurrence of the model is shown with 6 point matches—5 of these are correct while 1 (from the 'I' to the 'A') is wrong:
- — the Hough phase would reconcile these to one correct match.
- This is a challenging example as the occlusion masks most of the keypoints, the viewpoint has changed, the lighting on the glossy cover is significantly different and the matching criterion is strong.
- Less challenging examples generate many more, mostly correct, point matches.

Figure 8.5: Location by SIFT of 6 point matches in a challenging image, 5 of which are correct.

- SIFT is an early example of a class of detectors that derive their power from successful spatial descriptions local to interest points that are robust to deformations of many kinds
- A widely used alternative to SIFT is SURF—Speeded Up Robust Features [Bay et al., 2006]; this uses a similar strategy but exhibits far better performance as it exploits integral images which obviate the need for building a pyramid; further, the feature vector derived is shorter, and faster to build.

8.2 References

Bay H., Ess A., Tuytelaars T., and Van Gool L. Speeded-Up Robust Features (SURF). *Computer Vision Image Understanding*, 110(3):346–359, 2006.

Beis J. and Lowe D. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *Proceedings of the Conference on Computer Vision and Pattern Recognition CVPR*, pages 1000–1006, Puerto Rico, 1997.

Fisher D. J., Ehrhardt J. C., and Collins S. M. Automated detection of noninvasive magnetic resonance markers. In *Computers in Cardiology,* Chicago, IL, pages 493–496, Los Alamitos, CA, 1991. IEEE.

Goshtasby A. A. *2-D and 3-D Image Registration: for Medical, Remote Sensing, and Industrial Applications*. Wiley-Interscience, 2005.

Hajnal J. and Hill D. *Medical Image Registration*. Biomedical Engineering. Taylor & Francis, 2010. ISBN 9781420042474. URL http://books.google.co.uk/books?id=2dtQNsk-qBQC.

Ke Y. and Sukthankar R. PCA-SIFT: A more distinctive representation for local image descriptors. In *Proceedings of CVPR*, pages 506–513, 2004.

Lowe D. G. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

Tuytelaars T. and Mikolajczyk K. Local invariant descriptors: a survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2007.

Valera M. and Velastin S. A. Intelligent distributed surveillance systems: A review. In *IEE Proceedings - Vision, Image and Signal Processing*, 2005.

Yoo T. S. *Insight into Images: Principles and Practice for Segmentation, Registration, and Image Analysis*. AK Peters Ltd, 2004. ISBN 1568812175.