COPERNICUS MASTER
IN DIGITAL EARTH

Computer Vision

Lecture 7: Image Features

October 26, 2021

Prof. Sébastien Lefèvre
sebastien.lefevre@univ-ubs.fr

How to extract features from EO data?

So far we know:

- Spectral indices, NDVI, NDWI, BI, etc
- Morphological and Attribute Profiles
- Image edges

All figures today are from: Digital image processing; Gonzalez, Woods; Pearson

How to extract features from EO data?

So far we know:

- Spectral indices, NDVI, NDWI, BI, etc
- Morphological and Attribute Profiles
- Image edges

2 different problems (caution: vocabulary sometimes confusing)

- Feature extraction: image $\mapsto$ feature
- Feature description: image OR feature $\mapsto$ descriptor

All figures today are from: Digital image processing; Gonzalez, Woods; Pearson

How to extract features from EO data?

So far we know:

- Spectral indices, NDVI, NDWI, BI, etc
- Morphological and Attribute Profiles
- Image edges

2 different problems (caution: vocabulary sometimes confusing)

- Feature extraction: image $\mapsto$ feature
- Feature description: image OR feature $\mapsto$ descriptor

Descriptors can be:

- global, i.e. for the whole image
- local, i.e. for a part of the image (global by aggregating local ones)

All figures today are from: Digital image processing; Gonzalez, Woods; Pearson

Feature extraction

- edges
- corners/points
- lines
- keypoints
- blobs

Edge detection

1. High gradient areas
2. (significant) Laplacian zero crossings

Corner/point detection

1. Convolve with a Laplacian kernel and select points with magnitude higher than a threshold

2. Harris corner:

- compute in each point $\partial f/\partial x$ and $\partial f/\partial y$
- create matrix $M = \begin{pmatrix} A & C \\ C & B \end{pmatrix}$ with
  $A = \partial^2 f/\partial x^2$,
  $B = \partial^2 f/\partial y^2$,
  $C = (\partial f/\partial x)(\partial f/\partial y)$
- smooth with Gaussian or mean filter
- compute $H = det(M) - \alpha(tr(M))^2$ with $det(M) = AB - C^2$, $tr(M) = A + B, \alpha \in [0, 0.25]$
- compare with a threshold $H \geq T$

Line detection

Hough transform: a popular global approach to detect lines in an image.

- Represents lines in a cartesian space $y = ax + b$
  or better in a polar space $x \cos \theta + y \sin \theta = \rho$
- Accumulate the evidence for each line (i.e. pixels belonging to the line)
- Algorithm:

```
Detect edge points
For each edge point (x,y)
  For each theta
    Compute the corresponding rho
    Increment the cell (theta,rho)
Look for maxima in the accumulator
Display corresponding lines
```

Maximally Stable Extremal Regions (MSER)

Extremal regions are connected components from the stack of thresholdings

Maximally Stable are those that do not change size significantly over a range of thresholds

This descriptor can be straightforwardly implemented with morphological trees!

For a given node, stability is defined as the difference in area between ascending and descending nodes (with a level $\pm\Delta T$) normalized by the area of the node.

Keep stable nodes (difference lower than a threshold).

Extracted MSERs are represented as ellipses and called blobs.

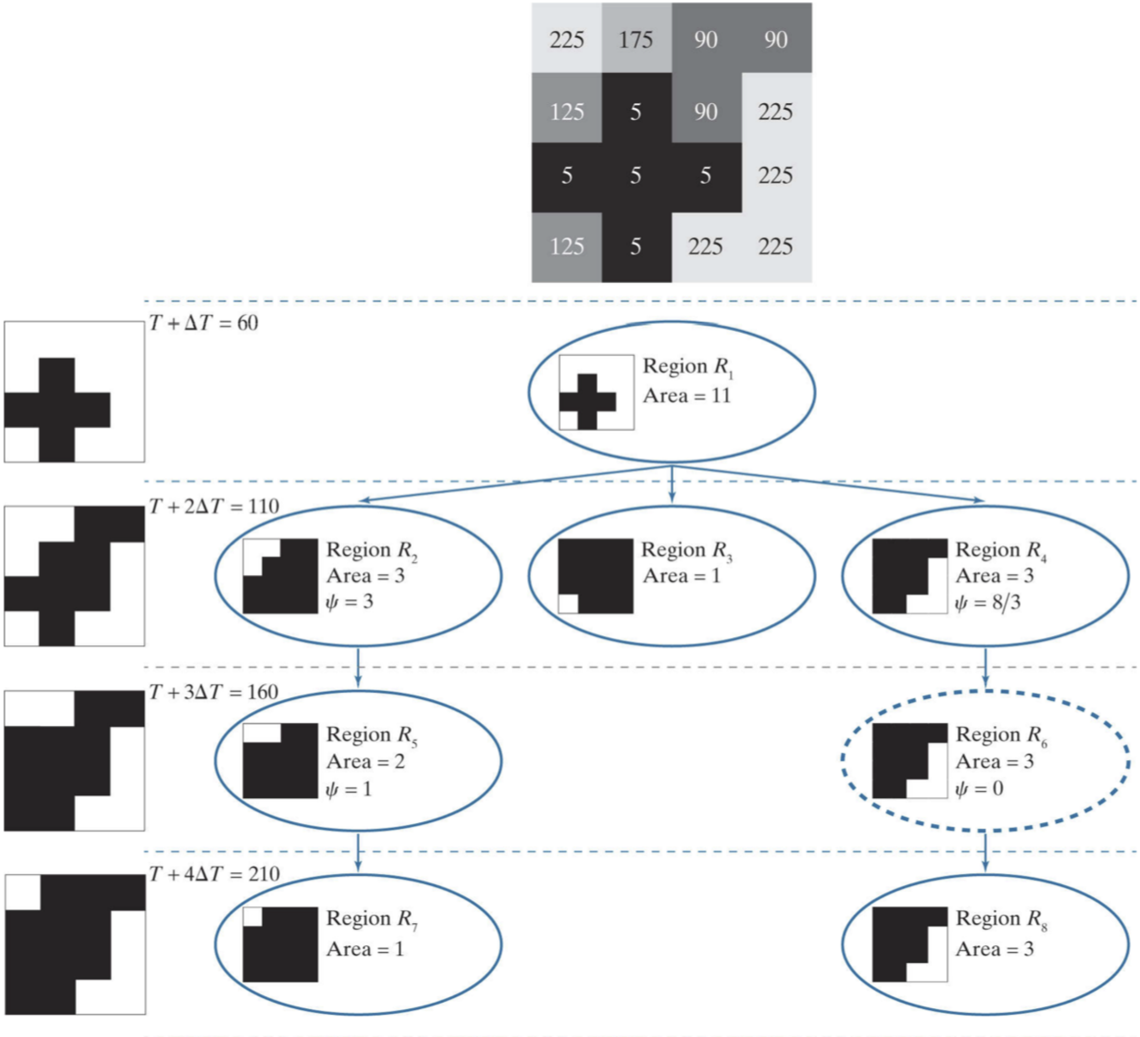


FIGURE 12.51

Detecting MSERs. Top: Grayscale image. Left: Thresholded images using $T = 10$ and $\Delta T = 50$. Right: Component tree, showing the individual regions. Only one MSER was detected (see dashed tree node on the rightmost branch of the tree). Each level of the tree is formed from the thresholded image on the left, at that same level. Each node of the tree contains one extremal region (connected component) shown in *white,* and denoted by a subscripted *R*.

Scale-Invariant Feature Transform (SIFT)

SIFT is a very popular local feature scheme.
It has led to numerous variants: SURF, ORB, etc.

It is actually made of 2 steps: keypoint extraction + keypoint description

Keypoint extraction

1. Gaussian scale-space (convolutions with Gaussian kernels)
2. Difference of Gaussians
3. Remove low contrast/poorly localized points
4. Remove edge responses (i.e. Harris detector)

Keypoint description to be presented later

Feature description

Various kinds of information:

- contour
- shape
- color/spectrum
- texture
- ...

Can be computed locally or globally.

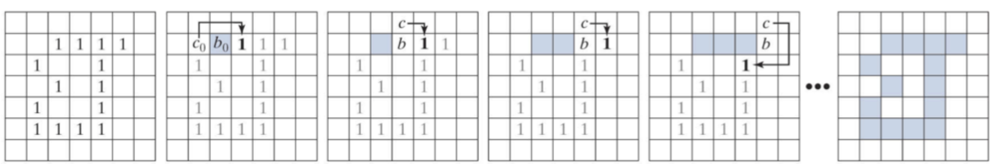
See also the attributes used with morphological trees.

Contour description

Geometric measures:

- contour length estimated by number of contour pixels, $L(C) \approx card(C)$
- diameter, $D(C) = \max_{i,j}(D(p_i, p_j))$ with $p_i, p_j \in C$
- length and orientation of major and minor axes of C
- elongation: ratio between major and minor axis

Contour tracing



a b c d e f

FIGURE 12.1
Illustration of the first few steps in the boundary-following algorithm. The point to be processed next is labeled in bold, black; the points yet to be processed are gray; and the points found by the algorithm are shaded. Squares without labels are considered background (0) values.

Freeman chain code

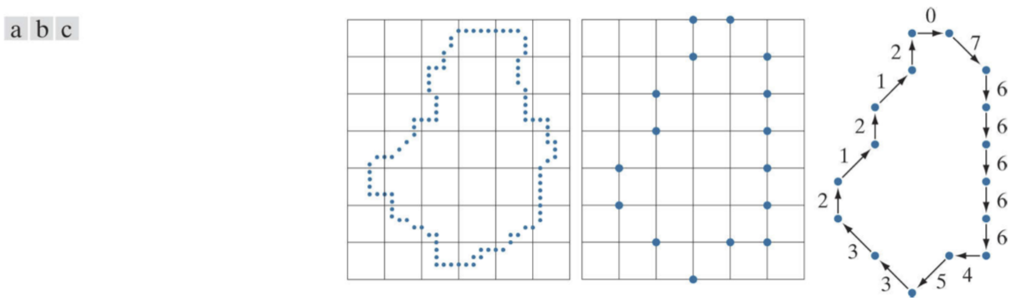


FIGURE 12.4
(a) Digital boundary with resampling grid superimposed. (b) Result of resampling. (c) 8-directional chain-coded boundary.

Statistical moments:

- C is modeled as a 1-D function $g(r)$
- compute moments (mean, variance, etc) over g

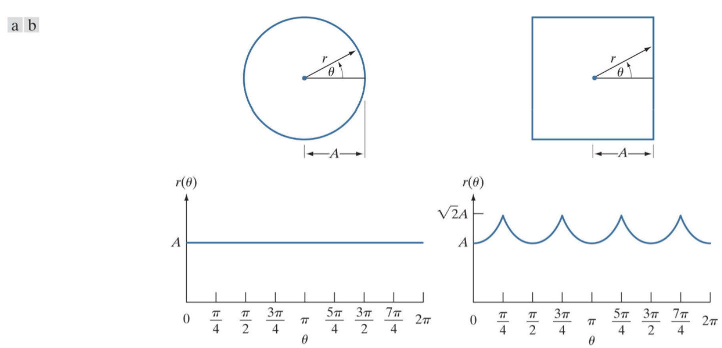


FIGURE 12.10
Distance-versus-angle signatures. In (a), $r(\theta)$ is constant. In (b), the signature consists of repetitions of the pattern $r(\theta) = A \sec\theta$ for $0 \le \theta \le \pi/4$, and $r(\theta) = A \csc\theta$ for $\pi/4 < \theta \le \pi/2$.

Fourier descriptors:

- Represent points as complexes: $s(k) = x(k) + iy(k)$
- Apply DFT and IDFT: $a(u) = \sum_k s(k)e^{-i2\pi uk/K}$
- Approximate the contour: $\hat{s}(k) = \sum_u^P a_u e^{-i2\pi uk/K}$

Quasi-invariance to transformations:

- rotation $s_r(k) = s(k)e^{i\theta}$
- translation $s_t(k) = s(k) + \delta_{xy}$
- scaling $s_h(k) = \alpha s(k)$
- origin change $s_o(k) = s(k - k_0)$

Shape description

Geometric measures:

- perimeter: $P(R) = L(F)$
- area: $A(R) = card(R)$
- compacity: $c = \frac{4\pi A(R)}{P^2(R)}$
- gravity center: $\begin{pmatrix} x_g \\ y_g \end{pmatrix} = \frac{1}{card(R)} \begin{pmatrix} \sum_i x_i \\ \sum_j x_j \end{pmatrix}$

Topologic measures:

- Euler number: $E = C - H$ with C and H the number of connected components and holes.
- in 4-connectivity, we have $E = \Omega_s - \Omega_a + \Omega_q$ with Ω_s, Ω_a, and Ω_q the number of pixels, pairs, and squares in the figure.

Spectral description

Histograms can be computed globally or locally (in each patch).

Mapping of RGB to another color space might be mandatory (e.g. HSV).

Quantization is also required, e.g. (4,4,4) or (7,3,3).

Histograms lack of spatial information:

- compute local histograms
- combine with morphological hierarchies (see e.g. HAP, Morphological Description of Color Images for Content-Based Image Retrieval, etc)

Texture description

Fourier spectrum:

- peaks provide the direction of textural patterns
- peaks location informs about the frequency of the patterns

Statistical approaches (using normalized histograms)

- moments and centered moments (e.g. average, standard deviation)
- homogeneity measure: $R = 1 - \frac{1}{1+\sigma^2(r)}$
- uniformity: $U = \sum_i p(r_i)^2$
- entropy: $e = -\sum_i p(r_i) \log_2 p(r_i)$

Cooccurrence matrices (a.k.a. GLCM)

Previous statistical measures do not take into account spatial information

$$A = (a)_{ij} = card((s, s+t) : I(s) = i, I(s+t) = j)$$

After normalization, $C = (c)_{ij} = A/n$

Main measures:

- maximal probability: $\max_{i,j}(c_{ij})$
- moment (of order n) of differences: $\sum_i \sum_j (i-j)^n c_{ij}$
- moment (of order n) of inverse differences: $\sum_i \sum_j c_{ij}/(i-j)^n$
- uniformity: $\sum_i \sum_j c_{ij}^2$
- entropy: $-\sum_i \sum_j c_{ij} \log_2 c_{ij}$
- contrast: $\sum_k k^2 \sum_{i,j:|i-j|=k} c_{ij}$

These and others form the so-called Haralick features

Moment invariants

2D moment: $m_{pq} = \sum_x \sum_y x^p y^q f(x, y)$

Central moment: $\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q f(x, y)$

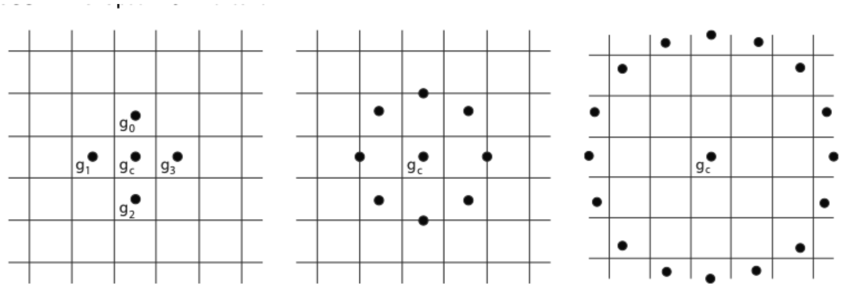
with $\bar{x} = \frac{m_{10}}{m_{00}}$ and $\bar{y} = \frac{m_{01}}{m_{00}}$

Normalized central moment: $\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}$ with $\gamma = \frac{p+q}{2} + 1$

Hu moment invariants:

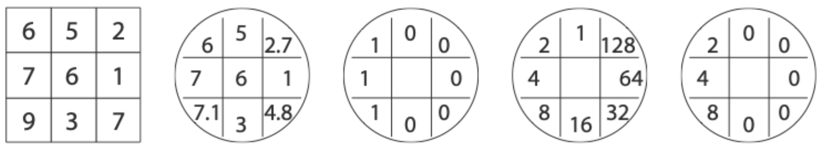
- $\phi_1 = \eta_{20} + \eta_{02}$
- $\phi_2 = (\eta_{20} + \eta_{02})^2 + 4\eta_{11}^2$
- $\phi_3 = (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2$
- $\phi_4 = (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2$
- $\ldots$

Local Binary Patterns



(a) P=4, R=1.0 (b) P=8, R=1.5 (c) P=16, R=3.0

Figure 15.6: Circularly symmetric neighborhoods for different values of P and R [Ojala et al., 2002b]. *© Cengage Learning 2015.*



(a) (b) (c) (d) (e)

Figure 15.7: Binary texture description operator $LBP_{8,1}$. (a) Original gray values of a 3×3 image. (b) Gray-level interpolation achieves symmetric circular behavior. Linear interpolation was used for simplicity (Section 5.2.2). (c) Circular operator values after binarization, equations (15.23–15.24). (d) Directional weights. (e) Directional values associated with $LBP_{8,1}$—the resulting value of $LBP_{8,1} = 14$. If rotationally normalized, the weighting mask would rotate by one position counterclockwise, yielding $LBP_{8,1}^{ri} = 7$. *© Cengage Learning 2015.*

Translation, contrast but also rotation, scale invariance

Histograms of Oriented gradients

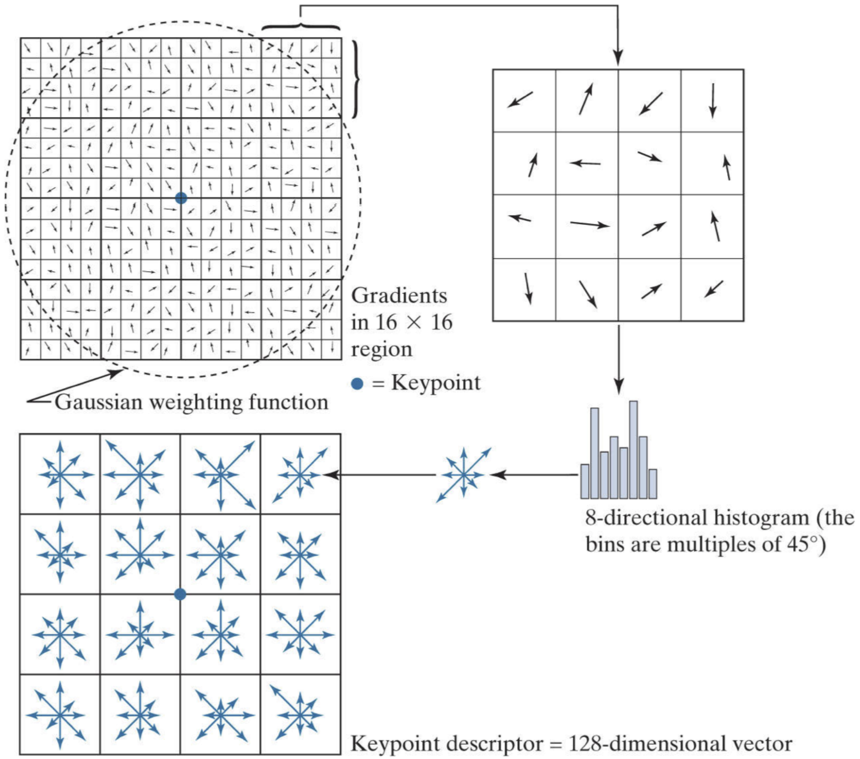


FIGURE 12.62
Approach used to compute a keypoint descriptor.
A 16 by 16 region is centered on the key point. Vectors in the squares extend in different directions, representing different gradients, and the key point forms the center of a circle representing the Gaussian weighting function. The gradients in a 4 by 4 corner section of the region can be used to produce an 8 directional histogram. In the histogram, the bins are multiples of 45 degrees, and the bin heights correspond to the magnitude of the direction vectors extending from the center of the selected section. This process is repeated until the original region is divided into a 4 by 4 array, with an 8-dimensional vector in each square. The final key descriptor is the 128 dimensional vector

HoG is the descriptor used for each SIFT keypoint.

And then comes deep learning

Deep features = features (or rather descriptors) that are learnt by the network

or simply using a pretrained network (without retraining), e.g. VGG16 features

Labs

Exercise 1

1. Load an urban remote sensing image (US-like)
2. Extract the road junctions with Harris detector
3. Compute the main road orientations with Hough transform

Exercise 2

1. Download the UC Merced Land Use dataset
2. Select randomly 10 images per class
3. Compute the Haralick features for these images
4. Apply a basic retrieval scheme through computing features on an input image and looking for its nearest neighbors among the reference images