Short Master Machine Learning 2020 Prof. Costantino Grana

Template matching

Template matching

- Goal: find 💽 in image
- Main challenge: What is a good similarity or distance measure between two patches?
 - Correlation
 - Zero-mean correlation
 - Sum Square Difference
 - Normalized Cross Correlation



• Goal: find 💽 in image

Input

• Method 0: filter the image with eye patch

$$h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l]$$

$$f = image$$

$$g = filter$$
What went wrong?

Filtered Image

- Goal: find 💽 in image
- Method 1: filter the image with zero-mean eye

$$h[m,n] = \sum_{k,l} (f[k,l] - \bar{f}) \underbrace{(g[m+k,n+l])}_{\text{mean of f}}$$



Input



Filtered Image (scaled)



Thresholded Image

- Goal: find in image
- Method 2: SSD

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







Input

1- sqrt(SSD)

Thresholded Image

- Goal: find 💽 in image
- Method 2: SSD

Input

What's the potential downside of SSD?

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

1- sqrt(SSD)



- Goal: find 💽 in image
- Method 3: Normalized cross-correlation

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

- Goal: find
 in image
- Method 3: Normalized cross-correlation



Input

Normalized X-Correlation

Thresholded Image

- Goal: find
 in image
- Method 3: Normalized cross-correlation



Input

Normalized X-Correlation

Thresholded Image

Q: What is the best method to use?

A: Depends

- SSD: faster, sensitive to overall intensity
- Normalized cross-correlation: slower, invariant to local average intensity and contrast
- But really, neither of these baselines are representative of modern recognition.

Q: What if we want to find larger or smaller eyes?

A: Image Pyramid

Short Master Machine Learning 2020

Review of Sampling



Gaussian pyramid



Source: Forsyth

Template Matching with Image Pyramids

Input: Image, Template

- 1. Match template at current scale
- 2. Downsample image
- 3. Repeat 1-2 until image is very small
- 4. Take responses above some threshold, perhaps with non-maxima suppression

Coarse-to-fine Image Registration

- 1. Compute Gaussian pyramid
- 2. Align with coarse pyramid
- 3. Successively align with finer pyramids
 - Search smaller range



Why is this faster?

Are we guaranteed to get the same result?

2D edge detection filters



 ∇^2 is the **Laplacian** operator:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

Laplacian filter



Source: Lazebnik

Computing Gaussian/Laplacian Pyramid



http://sepwww.stanford.edu/~morgan/texturematch/paper_html/node3.html

Laplacian pyramid



Source: Forsyth

Image representation

- Pixels: great for spatial resolution, poor access to frequency
- Fourier transform: great for frequency, not for spatial info
- Pyramids/filter banks: balance between spatial and frequency information

Major uses of image pyramids

- Compression
- Object detection
 - Scale search
 - Features
- Detecting stable interest points

- Registration
 - Course-to-fine

Short Master Machine Learning 2020 Prof. Costantino Grana

Texture

Representing Texture



Texture and Material









http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

Texture and Orientation







http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

Texture and Scale



http://www-cvr.ai.uiuc.edu/ponce_grp/data/texture_database/samples/

What is texture?

Regular or stochastic patterns caused by bumps, grooves, and/or markings

How can we represent texture?

Compute responses of blobs and edges at various orientations and scales

Overcomplete representation: filter banks



Code for filter banks: www.robots.ox.ac.uk/~vgg/research/texclass/filters.html

Filter banks

Process image with each filter and keep responses (or squared/abs responses)



How can we represent texture?

- Measure responses of blobs and edges at various orientations and scales
- Idea 1: Record simple statistics (e.g., mean, std.) of absolute filter responses

Can you match the texture to the response?





Representing texture by mean abs response



Representing texture

 Idea 2: take vectors of filter responses at each pixel and cluster them, then take histograms



Short Master Machine Learning 2020 Prof. Costantino Grana

Interest points and corners

Interest points

- Note: "interest points" = "keypoints", also sometimes called "feature points"
- Many applications
 - tracking: which points are good to track?
 - recognition: find patches likely to tell us something about object category
 - 3D reconstruction: find correspondences across different views
Interest points

- Suppose you have to click on some point, go away and come back after I deform the image, and click on the same points again.
 - Which points would you choose?



Overview of Keypoint Matching



 $d(f_A, f_B) < T$

- 1. Find a set of distinctive keypoints
- 2. Define a region around each keypoint
- 3. Extract and normalize the region content
- 4. Compute a local descriptor from the normalized region
- 5. Match local descriptors

K. Grauman, B. Leibe

Goals for Keypoints



Detect points that are *repeatable* and *distinctive*



Key trade-offs



Detection of interest points

More Repeatable

Robust detection Precise localization

More Points

Robust to occlusion Works with less texture

B

Description of patches

More Distinctive Minimize wrong matches More Flexible

Robust to expected variations Maximize correct matches

Invariant Local Features

•Image content is transformed into local feature coordinates that are invariant to translation, rotation, scale, and other imaging parameters



Features Descriptors

Choosing interest points

Where would you tell your friend to meet you?



Choosing interest points

Where would you tell your friend to meet you?



Feature extraction: Corners



Slides from Rick Szeliski, Svetlana Lazebnik, and Kristin Grauman

Corner Detection: Basic Idea

- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



"flat" region: no change in all directions

"edge": no change along the edge direction

"corner": significant change in all directions

Change in appearance of window *w*(*x*,*y*) for the shift [*u*,*v*]:

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







Change in appearance of window *w*(*x*,*y*) for the shift [*u*,*v*]:

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







Change in appearance of window w(x,y) for the shift [*u*,*v*]:



Change in appearance of window *w*(*x*,*y*) for the shift [*u*,*v*]:

$$E(u,v) = \sum_{x,y} w(x,y) \left[I(x+u,y+v) - I(x,y) \right]^{2}$$

We want to find out how this function behaves for small shifts

Change in appearance of window w(x,y) for the shift [*u*,*v*]:

$$E(u,v) = \sum_{x,y} w(x,y) \left[I(x+u,y+v) - I(x,y) \right]^{2}$$

We want to find out how this function behaves for small shifts

Local quadratic approximation of E(u,v) in the neighborhood of (0,0) is given by the second-order *Taylor expansion*:

 $E(u,v) \approx E(0,0) + \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2} \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$

x, y

Corner Detection: Mathematics

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

Second-order Taylor expansion of E(u,v) about (0,0): $E(u,v) \approx E(0,0) + \begin{bmatrix} u & v \end{bmatrix} \begin{vmatrix} E_u(0,0) \\ E_u(0,0) \end{vmatrix} + \frac{1}{2} \begin{bmatrix} u & v \end{bmatrix} \begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_u(0,0) & E_{uv}(0,0) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$ $E_{u}(u,v) = \sum 2w(x,y) [I(x+u,y+v) - I(x,y)] I_{x}(x+u,y+v)$ $E_{uu}(u,v) = \sum 2w(x,y)I_{x}(x+u,y+v)I_{x}(x+u,y+v)$ + $\sum 2w(x, y) [I(x+u, y+v) - I(x, y)] I_{xx}(x+u, y+v)$ $E_{uv}(u,v) = \sum 2w(x,y)I_{v}(x+u,y+v)I_{x}(x+u,y+v)$ + $\sum 2w(x, y) [I(x+u, y+v) - I(x, y)] I_{xv}(x+u, y+v)$

$$E(u,v) = \sum_{x,y} w(x,y) \left[I(x+u,y+v) - I(x,y) \right]^2$$

Second-order Taylor expansion of E(u,v) about (0,0): $E(u,v) \approx E(0,0) + [u \ v] \begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2} [u \ v] \begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$

$$E(0,0) = 0$$

$$E_{u}(0,0) = 0$$

$$E_{v}(0,0) = 0$$

$$E_{uu}(0,0) = \sum_{x,y} 2w(x,y)I_{x}(x,y)I_{x}(x,y)$$

$$E_{vv}(0,0) = \sum_{x,y} 2w(x,y)I_{y}(x,y)I_{y}(x,y)$$

$$E_{uv}(0,0) = \sum_{x,y} 2w(x,y)I_{x}(x,y)I_{y}(x,y)$$

$$E(u,v) = \sum_{x,y} w(x,y) \left[I(x+u,y+v) - I(x,y) \right]^{2}$$

Second-order Taylor expansion of $E(u,v)$ about (0,0):
$$E(u,v) \approx [u \ v] \left[\sum_{x,y} w(x,y)I_{x}^{2}(x,y) - \sum_{x,y} w(x,y)I_{x}(x,y)I_{y}(x,y) - \sum_{x,y} w(x,y)I_{y}^{2}(x,y) - \sum_{x,y} w(x,y)I_{y}(x,y) - \sum_{x,y} w(x,y)I_{y}$$

M

Corner Detection: Mathematics

The quadratic approximation simplifies to

$$E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$$

where *M* is a second moment matrix computed from image derivatives:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$
$$= \begin{bmatrix} \sum_{x} I_x I_x & \sum_{x} I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x I_y] = \sum \nabla I (\nabla I)^T$$

Corners as distinctive interest points

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

2 x 2 matrix of image derivatives (averaged in neighborhood of a point).



Interpreting the second moment matrix

The surface E(u,v) is locally approximated by a quadratic form. Let's try to understand its shape.

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

Interpreting the second moment matrix

Consider a horizontal "slice" of E(u, v): $\begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$ This is the equation of an ellipse.



Interpreting the second moment matrix

Consider a horizontal "slice" of E(u, v): $\begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$

This is the equation of an ellipse.

Diagonalization of M:
$$M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

The axis lengths of the ellipse are determined by the eigenvalues and the orientation is determined by R



Visualization of second moment matrices



Visualization of second moment matrices



Interpreting the eigenvalues

Classification of image points using eigenvalues of *M*:



Corner response function

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

 α : constant (0.04 to 0.06)



Harris corner detector

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

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

Harris Detector [Harris88]

• Second moment matrix

$$\mu(\sigma_{I},\sigma_{D}) = g(\sigma_{I}) * \begin{bmatrix} I_{x}^{2}(\sigma_{D}) & I_{x}I_{y}(\sigma_{D}) \\ I_{x}I_{y}(\sigma_{D}) & I_{y}^{2}(\sigma_{D}) \end{bmatrix}$$
1. Image derivatives (optionally, blur first)
$$1 + \frac{1}{2} = \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2} = \frac{1}{2} + \frac{1}{2} + \frac{1}{2} + \frac{1}{2} = \frac{1}{2} + \frac{1$$

$$har = \det[\mu(\sigma_{I}, \sigma_{D})] - \alpha[\operatorname{trace}(\mu(\sigma_{I}, \sigma_{D}))^{2}] = g(I_{x}^{2})g(I_{y}^{2}) - [g(I_{x}I_{y})]^{2} - \alpha[g(I_{x}^{2}) + g(I_{y}^{2})]^{2}$$

5. Non-maxima suppression

Harris Detector: Steps



Harris Detector: Steps Compute corner response *R*



Harris Detector: Steps

Find points with large corner response: *R*>threshold



Harris Detector: Steps Take only the points of local maxima of *R*

Harris Detector: Steps



Invariance and covariance

- We want corner locations to be *invariant* to photometric transformations and *covariant* to geometric transformations
 - Invariance: image is transformed and corner locations do not change
 - Covariance: if we have two transformed versions of the same image, features should be detected in corresponding locations



Affine intensity change

- Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$
- Intensity scaling: $I \rightarrow a I$





x (image coordinate)

Partially invariant to affine intensity change

Image translation



• Derivatives and window function are shift-invariant

Corner location is covariant w.r.t. translation
Image rotation



Second moment ellipse rotates but its shape (i.e. eigenvalues) remains the same

Corner location is covariant w.r.t. rotation





All points will be classified as edges

Corner location is not covariant to scaling!

Short Master Machine Learning 2020 Prof. Costantino Grana

Local descriptors

Harris Detector [Harris88]

• Second moment matrix

$$\mu(\sigma_{I},\sigma_{D}) = g(\sigma_{I}) * \begin{bmatrix} I_{x}^{2}(\sigma_{D}) & I_{x}I_{y}(\sigma_{D}) \\ I_{x}I_{y}(\sigma_{D}) & I_{y}^{2}(\sigma_{D}) \end{bmatrix}$$
1. Image derivatives (optionally, blur first)
$$1 + M = \lambda_{I}\lambda_{2}$$

$$\frac{2. \text{ Square of derivatives}}{1 + \lambda_{2}}$$
3. Gaussian filter $g(\sigma_{I})$

$$\frac{1}{1 + \lambda_{2}} = \frac{1}{1 +$$

har

$$har = \det[\mu(\sigma_{I}, \sigma_{D})] - \alpha[\operatorname{trace}(\mu(\sigma_{I}, \sigma_{D}))^{2}] = g(I_{x}^{2})g(I_{y}^{2}) - [g(I_{x}I_{y})]^{2} - \alpha[g(I_{x}^{2}) + g(I_{y}^{2})]^{2}$$

5. Non-maxima suppression

So far: can localize in x-y, but not scale





How to find corresponding patch sizes?









• Function responses for increasing scale (scale signature)





 $f(I_{i_1...i_m}(x,\sigma))$





What Is A Useful Signature Function?

• Difference-of-Gaussian = "blob" detector



Difference-of-Gaussian (DoG)









DoG – Efficient Computation

• Computation in Gaussian scale pyramid



Find local maxima in position-scale space of Difference-of-Gaussian



Results: Difference-of-Gaussian



Orientation Normalization

- Compute orientation histogram
- Select dominant orientation
- Normalize: rotate to fixed orientation

[Lowe, SIFT, 1999]



Image representations

- Templates
 - Intensity, gradients, etc.



- Histograms
 - Color, texture, SIFT descriptors, etc.

Image Representations: Histograms



Global histogram

Represent distribution of features

- Color, texture, depth, ...

Image Representations: Histograms

Histogram: Probability or count of data in each bin





Marginal histogram

- Requires independent features
- More data/bin than joint histogram

Computing histogram distance

histint
$$(h_i, h_j) = 1 - \sum_{m=1}^{K} \min(h_i(m), h_j(m))$$

Histogram intersection (assuming normalized histograms)

$$\chi^{2}(h_{i},h_{j}) = \frac{1}{2} \sum_{m=1}^{K} \frac{[h_{i}(m) - h_{j}(m)]^{2}}{h_{i}(m) + h_{j}(m)}$$

Chi-squared Histogram matching distance



Cars found by color histogram matching using chi-squared

What kind of things do we compute histograms of?

Color



• Texture (filter banks or HOG over regions)

What kind of things do we compute histograms of?

• Histograms of oriented gradients



SIFT vector formation

- Computed on rotated and scaled version of window according to computed orientation & scale
 - resample the window
- Based on gradients weighted by a Gaussian of variance half the window (for smooth falloff)



SIFT vector formation

- 4x4 array of gradient orientation histogram weighted by magnitude
- 8 orientations x 4x4 array = 128 dimensions
- Motivation: some sensitivity to spatial layout, but not too much.



Ensure smoothness

- Gaussian weight
- Trilinear interpolation
 - a given gradient contributes to 8 bins:
 - 4 in space times 2 in orientation



Reduce effect of illumination

- 128-dim vector normalized to 1
- Threshold gradient magnitudes to avoid excessive influence of high gradients
 - after normalization, clamp gradients >0.2
 - renormalize



Local Descriptors

- Most features can be thought of as templates, histograms (counts), or combinations
- The ideal descriptor should be
 - Robust
 - Distinctive
 - Compact
 - Efficient
- Most available descriptors focus on edge/gradient information
 - Capture texture information
 - Color rarely used

Local Descriptors: SURF



Fast approximation of SIFT idea

Efficient computation by 2D box filters & integral images ⇒ 6 times faster than SIFT Equivalent quality for object identification

GPU implementation available

Feature extraction @ 200Hz (detector + descriptor, 640×480 img)

http://www.vision.ee.ethz.ch/~surf

[Bay, ECCV'06], [Cornelis, CVGPU'08]

Choosing a descriptor

- Again, need not stick to one
- For object instance recognition or stitching, SIFT or variant is a good choice

Things to remember

- Keypoint detection: repeatable and distinctive
 - Corners, blobs, stable regions
 - Harris, DoG



- Descriptors: robust and selective
 - spatial histograms of orientation
 - SIFT

