Feature detection and description

MVA/IMA – Vision 3D artificielle

Mathieu Aubry
http://imagine.enpc.fr/~aubrym/

10/30/2019
What do I work on?

Historical and non photo-realistic data

Deep 3D model generation/analysis.

-> let me know if you are interested in internships/PhDs on these topics
Goal/Key concepts

Goal: “Draw” correspondences between images

1. Detection: Harris/Laplacian/Hessian
2. Description: HOG/BRIEF
3. Case study: SIFT/SURF
4. Deep approaches
Motivation: panorama
Motivation: panorama

1. Which features?
2. How to describe them?
3. How to match them?
Motivation: panorama

1. Find distinctive features
2. Match similar features
3. Compute the Fundamental matrix (1\textsuperscript{st} and 2\textsuperscript{nd} lecture)
3D reconstruction (next time)

- External camera calibration
  = determination of pose (i.e., location and orientation) of each camera in a common coordinate system
  - requires enough corresponding points in several images
  → detection and matching of salient points

- Dense 3D reconstruction
  = by triangulation, given camera pose (!) not restricted to salient points only
  - requires matching image patches in several images
Motivation: tracking

J. Lezama, K. Alahari, J. Sivic, I. Laptev
Track to the Future: Spatio-temporal Video Segmentation with Long-range Motion Cues
CVPR 2011
Motivation: instance retrieval

1. Identify salient points
2. Look for similar salient points in other image
3. Check geometrical consistency (rigid or deformable)
Motivation: content-based image retrieval

Philbin, J., Chum, O., Isard, M., Sivic, J. and Zisserman, A.
Object retrieval with large vocabularies and fast spatial matching
CVPR 2007
Difficulty

Viewpoint changes
Difficulty

Illumination changes
Difficulty

Season changes
Difficulty

Clutter and occlusion
Difficulty

- change of scale (camera motion or change of focal length)
- change of orientation (rotation)
- change of viewpoint (affine, projective transformations)
- change of illumination (time of day, weather, flash...)

And also
- change of camera parameters (speed/aperture ...)
- non-rigid scene (objects in motion, deformable surface)
- surface reflectance (Lambertian or not, reflection, transpar.)
- repetitive patterns (windows, road marks...)
Today

• Feature detection: How to extract informative features consistently?

• Feature description: how to compare features?
Convolutions, correlations, derivatives
Convolution

- Continuous convolution (1D)
  - $f, g : \mathbb{R} \to \mathbb{R}$
  - $(f * g)(x) = \int_{-\infty}^{+\infty} f(u)g(x-u) \, du = \int_{-\infty}^{+\infty} f(x-u)g(u) \, du$

- Discrete convolution (1D)
  - $f, g : \mathbb{Z} \to \mathbb{R}$
  - $(f * g)(n) = \sum_{m=-\infty}^{+\infty} f(m)g(n-m) = \sum_{m=-\infty}^{+\infty} f(n-m)g(m)$

... if integral/sum exists
  - sufficient: $f$ compactly supported, $f$ integrable and $g$ bounded...
Convolution

• Extension to dimension \(d\)
  
  - \(f, g : \mathbb{IR}^d \text{ or } \mathbb{Z}^d \rightarrow \mathbb{IR}\) (or with values in \(\mathbb{C}\))

• Ex. 2D continuous convolution
  
  \[
  (f \ast g)(x, y) = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(u, v) g(x-u, y-v) \, du \, dv = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x-u, y-v) g(u, v) \, du \, dv
  \]

• Ex. 2D discrete convolution
  
  \[
  (f \ast g)(i, j) = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} f(i, j) g(i-k, j-l) = \sum_{k=-\infty}^{+\infty} \sum_{l=-\infty}^{+\infty} f(i-k, j-l) g(k, l)
  \]

  ... if integral/sum exists

  - sufficient: \(f\) compactly supported, \(f\) integrable and \(g\) bounded...

• Efficient convolution in Fourier
Blur-Convolution

• Blurred image: \[ O = K \ast I \]

  e.g. uniform motion blur

Convolution

\[(K \ast I)(i, j) = \sum_{k,l} K(-k, -l)I(i + k, j + l)\]
Convolution

$$(K * I)(i, j) = \sum_{k,l} K(-k, -l)I(i + k, j + l)$$
Convolution

\[(K \ast I)(i, j) = \sum_{k,l} K(-k, -l) I(i + k, j + l)\]
Convolution

\[(K \ast I)(i, j) = \sum_{k,l} K(-k, -l) I(i + k, j + l)\]
Convolution

\[(K \ast I)(i, j) = \sum_{k,l} K(-k, -l)I(i + k, j + l)\]
Filtering in frequency domain

- FFT
- Inverse FFT

Slide: Hoiem
Linear stationary operators

\[ T_u : f \rightarrow T_u[f] = f(\cdot - u) \]

- A linear operator \( L \) is stationary if for any \( u \):
  \[ T_u[L(f)] = L(T_u[f]) \]

- Every linear and stationary operator can be written as a convolution

Idea:

\[ f(x) = \int f(u)\delta_u(x)du \]
Practice with linear filters

Original

0 0 0
0 1 0
0 0 0

Source: D. Lowe
Practice with linear filters

Original

Source: D. Lowe
Practice with linear filters

Original

0 0 0
0 0 1
0 0 0

Source: D. Lowe
Practice with linear filters

Original

Shifted left
By 1 pixel

Source: D. Lowe
Practice with linear filters

Original

\[
\begin{bmatrix}
1 & 1 & 1 \\
1 & 1 & 1 \\
1 & 1 & 1 \\
\end{bmatrix}
\]

Source: D. Lowe
Practice with linear filters

Original

Blur (with a box filter)

Source: D. Lowe
Practice with linear filters

Original

(Note that filter sums to 1)

Source: D. Lowe
Practice with linear filters

Original

Sharpening filter
- Accentuates differences with local average

Source: D. Lowe
Sharpening

- What does blurring take away?

Let’s add it back:
Sharpening

before

after

Source: D. Lowe
Smoothing with box filter revisited

- What’s wrong with this picture?
- What’s the solution?

Source: D. Forsyth
Gaussian convolution

\[ g_\sigma(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}. \]
Gaussian convolution

\[ g_\sigma(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}. \]

extrait de Kandinsky

\[ \sigma = 1.0 \]
Gaussian convolution

\[ g_\sigma(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2 + y^2}{2\sigma^2}}. \]

extrait de Kandinsky

\( \sigma = 1.5 \)
Gaussian convolution

\[ g_\sigma(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2 + y^2}{2\sigma^2}}. \]

extrait de Kandinsky

\( \sigma = 2.0 \)
Gaussian convolution

\[ g_\sigma(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2 + y^2}{2\sigma^2}}. \]

extrait de Kandinsky

\[ \sigma = 2.5 \]
Gaussian convolution

\[ g_\sigma(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2 + y^2}{2\sigma^2}}. \]

extrait de Kandinsky

\[ \sigma = 3.0 \]
Separability

- Let $h : \{1, \ldots, M\} \times \{1, \ldots, N\} \rightarrow \mathbb{R}$ be a kernel/matrix
  - usual convolution of $h$ with image $f$ requires $M \times N$ multiplications and additions per pixel of $f$

- Separable kernel
  - if there exist vectors $h_1, h_2$ such that $h = h_1^T h_2$ then
    $$h = h_1^T h_2 = h_1^T * h_2 = h_2 * h_1^T$$

- Separate convolution
  - $f * h = f * (h_1^T * h_2) = (f * h_1^T) * h_2$ [or $(f * h_2) * h_1^T$]
  - perform two 1D-convolutions instead of one 2D-convolution
  - only $M + N$ multiplications and additions per pixel of $f$

-> The Gaussian Kernel is separable!
Convolution

-> Can be used for template matching (eg. object detection, blob detection)

-> Can be used on more than 1 channel (eg. Color image, Convolutional Neural Networks)
Convolution vs. correlation

- **Continuous convolution (1D):** \( f \ast g \)
  \[
  (f \ast g)(x) = \int_{-\infty}^{+\infty} f(u)g(x-u)\,du = \int_{-\infty}^{+\infty} f(x-u)g(u)\,du = (g \ast f)(x)
  \]

- **Continuous correlation (1D):** \( f \otimes g \) [see variant later]
  \[
  (f \otimes g)(x) = \int_{-\infty}^{+\infty} f(u)g(x+u)\,du = \int_{-\infty}^{+\infty} f(u-x)g(u)\,du \neq (g \otimes f)(x)
  \]

- If \( f \) symmetric, i.e., \( f(x) = f(-x) \), then \( f \otimes g = f \ast g \)
Convolution vs. correlation

Convolution

Cross-correlation

Autocorrelation
Convolution: properties

- **Linearity: distributivity + associativity with scalar mult.**
  - \( f \ast (h_1 + h_2) = f \ast h_1 + f \ast h_2 \)
  - \((f_1 + f_2) \ast h = f_1 \ast h + f_2 \ast h\)
  - \( c.(f \ast h) = (c.f) \ast h = f \ast (c.h) \) (also true of \( \otimes \))

- **Shift-invariance**
  - translation: \((T_u f)(x) = f(x-u)\)
  - \((T_u f) \ast h = T_u (f \ast h)\) (also true of \( \otimes \))
    - \( g(i,j) = f(i+k, j+l) \Leftrightarrow (g \ast h)(i,j) = (f \ast h)(i+k, j+l)\)

- **Associativity**
  - \((f \ast g) \ast h = f \ast (g \ast h)\) (also true of \( \otimes \))

- **Commutativity**
  - \( f \ast h = h \ast f \) (not true of \( \otimes \))
Image boundary effects

- **Padding strategies** (aka wrapping mode, texture addressing mode)
  - pad with 0 (or constant), wrap (loop around), clamp (replicate edge pixel), mirror (reflect pixels across edge)

Blurring examples:

- blurred: zero
- normalized zero
- clamp
- mirror

* or discard results close to boundary
Derivatives with convolution

For 2D function $f(x,y)$, the partial derivative is:

$$\frac{\partial f(x, y)}{\partial x} = \lim_{\varepsilon \to 0} \frac{f(x + \varepsilon, y) - f(x, y)}{\varepsilon}$$

For discrete data, we can approximate using finite differences:

$$\frac{\partial f(x, y)}{\partial x} \approx \frac{f(x + 1, y) - f(x, y)}{1}$$

To implement the above as convolution, what would be the associated filter?

Source: K. Grauman
Partial derivatives of an image

\[ \frac{\partial f(x, y)}{\partial x} \]

\[ \frac{\partial f(x, y)}{\partial y} \]

Which shows changes with respect to \(x\)?

Source: L. Lazebnik
Image gradient

- The gradient of an image:
  \[ \nabla f = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \]

- The gradient points in the direction of most rapid increase in intensity

- How does this direction relate to the direction of the edge?
  The gradient direction is given by
  \[ \theta = \tan^{-1} \left( \frac{\frac{\partial f}{\partial y}}{\frac{\partial f}{\partial x}} \right) \]

  The edge strength is given by the gradient magnitude
  \[ ||\nabla f|| = \sqrt{\left( \frac{\partial f}{\partial x} \right)^2 + \left( \frac{\partial f}{\partial y} \right)^2} \]

Source: Steve Seitz
Image gradient

• Directional derivative along $n = (n_x, n_y)$:

$$\frac{\partial u}{\partial n}(x, y) = \lim_{h \to 0} \frac{u(x + hn_x, y + hn_y) - u(x, y)}{h}$$

• Relation between directional derivative and gradient:

$$\frac{\partial u}{\partial n}(x, y) = \frac{\partial u}{\partial x}(x, y)n_x + \frac{\partial u}{\partial y}(x, y)n_y = \nabla u(x, y) \cdot n$$
Intensity profile
With a little Gaussian noise

Source: D. Hoiem
Effects of noise

- Consider a single row or column of the image

\[ f(x) \]

\[ \frac{d}{dx} f(x) \]

Where is the edge?  

Source: S. Seitz
Solution: smooth first

- To find edges, look for peaks in $\frac{d}{dx}(f \ast g)$

Source: S. Seitz
Derivative theorem of convolution

• Differentiation is convolution, and convolution is associative:

\[
\frac{d}{dx} (f * g) = f * \frac{d}{dx} g
\]

• This saves us one operation:

Source: S. Seitz
Derivative of Gaussian filters

- Which one finds horizontal/vertical edges?

Source: L. Lazebnik
Derivative of Gaussian filters

- Are these filters separable?

Source: L. Lazebnik
More generaly

• The convolution of a discrete image with a Gaussian define a continuous function, to which we can apply any differentiable operator.

• The operator can be applies to the Gaussian before the convolution
Outline

You should know: convolutions, correlations, derivatives

1. Classical feature detection
   Harris (Corner)
   Laplacian, Hessian (Blob)

2. Classical feature description:
   SSD, ZNCC, ShapeContext, HOG, BRIEF

3. Important and many details: SIFT/SURF

4. Learning description, detection and correspondences
Which features?

- Edges: eg. Canny edges

Canny, J. (1986). A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*
Which features?

- Regions: eg. MSER

Which features?

- Simple region, blobs: eg. Harris-affine
Which features?

- Points
Which features?

- distinctive/repeatable
Harris Corner
Harris Corner

Idea: compare a patch centered in (0,0) defined by the weights $w$ to a patch centered in (x,y) using pixel intensity

\[
S(x, y) = \sum_u \sum_v w(u, v) (I(u + x, v + y) - I(u, v))^2
\]

-> if the difference is large for all (x,y) the patch is distinctive.

Initial idea Moravec 1980
Auto-correlation for corner detection (Moravec 1980)

- Corner?

A. Interior Region
   Little intensity variation in any direction

B. Edge
   Little intensity variation along edge, large variation perpendicular to edge

C. Edge
   Large intensity variation in all directions

D. Edge
   Large intensity variation in all directions

Parks & Gravel © McGill U.
Harris Corner

Idea: compare a patch centered in $(0,0)$ defined by the weights $w$ to a patch centered in $(x,y)$ using pixel intensity

$$S(x, y) = \sum_{u} \sum_{v} w(u, v) \left( I(u + x, v + y) - I(u, v) \right)^2$$
Harris Corner

Idea: compare a patch centered in \((0,0)\) defined by the weights \(w\) to a patch centered in \((x,y)\) using pixel intensity

\[
S(x, y) = \sum_u \sum_v w(u, v) \left( I(u + x, v + y) - I(u, v) \right)^2
\]

Use Taylor extension of \(I\) (Harris and Stephen 1988):

\[
I(u + x, v + y) \approx I(u, v) + I_x(u, v)x + I_y(u, v)y
\]

\[
S(x, y) \approx \sum_u \sum_v w(u, v) \left( I_x(u, v)x + I_y(u, v)y \right)^2
\]
Harris Corner

Idea: compare a patch centered in (0,0) defined by the weights \( w \) to a patch centered in \((x,y)\) using pixel intensity

\[
S(x, y) = \sum_u \sum_v w(u, v) \left( I(u + x, v + y) - I(u, v) \right)^2
\]

Use Taylor extension of \( I \) (Harris and Stephen 1988):

\[
I(u + x, v + y) \approx I(u, v) + I_x(u, v)x + I_y(u, v)y
\]

\[
S(x, y) \approx (x \quad y) \sum_u \sum_v w(u, v) \begin{bmatrix} I_x^2 & I_xI_y \\ I_xI_y & I_y^2 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]
Harris Corner

- $S$ large in all direction $\leftrightarrow$ condition on

$$\sum_u \sum_v w(u, v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$
Spectral theorem

We can diagonalize the any symmetric positively defined matrix $M$ in an orthonormal basis $(e_1, \ldots, e_m)$, i.e. write

$$M = \sum_{i=1}^{m} \lambda_i e_i e_i^T$$

$$M e_i = \lambda_i e_i$$

$\lambda_1 \geq \ldots \geq \lambda_m \geq 0$ are the eigenvalues

Interpretation: $M$
Spectral theorem

We can diagonalize the any symmetric positively defined matrix $M$ in an orthonormal basis $(e_1, \ldots, e_m)$, i.e. write

$$M = \sum_{i=1}^{m} \lambda_i e_i e_i^T \quad \text{and} \quad Me_i = \lambda_i e_i$$

$$\lambda_1 \geq \ldots \geq \lambda_m \geq 0$$

are the eigenvalues

- Si $u = \sum_{i=1}^{m} u_i e_i$

$$\min_i \lambda_i \|u\|^2 \leq u^T M u = \sum_{i=1}^{m} \lambda_i u_i^2 \leq \max_i \lambda_i \|u\|^2$$
Harris Corner

• $S$ large in all direction $\iff$ the two eigenvalues of
  \[
  \sum_u \sum_v w(u, v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}
  \] are large
Harris Corner

• S large in all direction <-> the two eigenvalues of
  \[ \sum_u \sum_v w(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \]
  are large

• Harris and Stephens suggest to use

  \[ M_c = \lambda_1 \lambda_2 - \kappa (\lambda_1 + \lambda_2)^2 = \det(A) - \kappa \text{trace}^2(A) \]

with

  \[ A = \sum_u \sum_v w(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = w * \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \]

the auto-correlation or second moment matrix

w Gaussian -> invariant to in plane rotation
Harris Corner

\[ \lambda_2 \quad \text{“Edge”} \quad \lambda_2 \gg \lambda_1 \]

\[ \lambda_1 \text{ and } \lambda_2 \text{ are large, } \lambda_1 \sim \lambda_2; \]

\[ \text{“Corner”} \]

\[ \lambda_1 \gg \lambda_2 \]

“Flat” region

Figure from Cordelia Schmid
Harris Corner: algo

- Compute images derivatives $I_x(x_q)$ and $I_y(y_q)$ for each pixel $q$ of $I$
  - compute smooth derivation operators: e.g., convolve $d_x = [-\frac{1}{2} 0 \frac{1}{2}]$
    with 1D Gaussian $G$ (e.g., $\sigma_d = 1$) → “mask” $G_x$; then define $G_y = G_x^T$
  - compute “image derivatives” $I_x$ and $I_y$: convolve $I$ with masks $G_x$ and $G_y$
- Compute “product images” $I_x^2, I_x I_y, I_y^2$ (not matrix products!)
  - then add extra smoothing using an “integration” Gaussian (e.g., $\sigma_i = 2$)
    (again using two 1D-convolutions rather than one 2D-convolution)
- Consider auto-correlation matrix $A = \begin{bmatrix} I_x^2 & I_x I_y & I_y^2 \\ I_x I_y & I_x I_x & I_y I_y \end{bmatrix}$
  - compute corner response (or strength) for each $q$
  - response above threshold and local maximum (8 neighbors) → detection
  - possibly: only keep locally significant responses (see ANMS below)
Multi-scale

Gaussian pyramid

- Convolution with Gaussian of varying $\sigma$
  \[ G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \]

- Scale-space representation
  \[ L(x, y; \sigma) = G(x, y; \sigma) * I(x, y) \]

- Scale pyramid

Space: $x, y$ dimensions (location)
Scale-space: $\sigma$ dimension
Scale-space

\( \sigma = 0 \) (original image) \hspace{1cm} \sigma = 1 \hspace{1cm} \sigma = 4

\( \sigma = 16 \) \hspace{1cm} \sigma = 64 \hspace{1cm} \sigma = 256
Blob detection
Blob detection: idea

- Find maxima and minima of blob filter response in space and scale

Source: N. Snavely
Blob detection: LoG

- Idea: convolve image with Laplacian of Gaussian and look for extrema

- Laplacian of Gaussian (LoG)
  - \( L(x, y; \sigma) = G(x, y; \sigma) \ast I(x, y) \)
  - \( \nabla^2 L = \nabla^2 (G \ast I) = (\nabla^2 G) \ast I \)
  - \( \nabla^2 G(x, y; \sigma) = -\frac{1}{\pi \sigma^4} \left[ 1 - \frac{x^2 + y^2}{2 \sigma^2} \right]^2 \exp\left(-\frac{x^2 + y^2}{2 \sigma^2}\right) \)
  - strongest response for compact blobs of extent \( \sqrt{2\sigma} \)
Scale-normalized LoG

- Need for normalization (Lindeberg 1994)
Scale-normalized LoG

- Need for normalization (Lindeberg 1994)

original signal (radius=8)  increasing σ
Scale-normalized LoG

- Need for normalization (Lindeberg 1994)
  - scale-space smoothing $\sigma \Rightarrow$ spatial derivatives $\nabla$, factor $\sigma$
  - $\nabla^2 G$ second derivatives $\Rightarrow \nabla$ factor $\sigma^2$
  - scale-normalized LoG: $\nabla^2_{\text{norm}} G = \sigma^2 \nabla^2 G$

- or looking at the formula by homogeneity

- or writing the desired scale invariance

- or thinking of DoG (see later) (see later)
Scale-normalized LoG

Image S. Lazebnik
Blob detection: Hessian

Hessian: its eigen-values/vector give principal curvatures of the image

\[
H = \begin{pmatrix}
L_{xx} & L_{xy} \\
L_{xy} & L_{yy}
\end{pmatrix}
\]

\[
L(x + dx, y + dy) \simeq L(x, y) + \nabla(L)^T (dx, dy)^T + (dx, dy) H(x, y) (dx, dy)^T
\]

\[
Tr(H) = L_{xx} + L_{yy} = \lambda_0 + \lambda_1 \quad \text{also large around edges}
\]

-> also use \[
Det(H) = \lambda_0 \lambda_1
\]
Non-Max suppression
Non-Maximum Suppression

- **Problem:**
  
  maximality in 3x3 neighborhood
  
  → uneven distribution
  (dense where high contrast)
  
  → poor robustness
  (sensitive to noise)

- **Solution 1 (NMS):**
  
  - check in larger region around $p$ (e.g., disk of given radius $r$):
    
    check maximality w.r.t. all points $q$ such that $\|x_p - x_q\| \leq r$
  
  - check almost largest response (e.g., within 10%):
    
    $\forall q \quad \|x_p - x_q\| \leq r \Rightarrow c f(x_q) \leq f(x_p)$  [e.g., $c = 0.9$]
Adaptive non-maximal suppression (ANMS)

- Problem with NMS
  - Distribution still uneven
  - Need to tune $r$
- Solution 2: ANMS
  - Compute a radius for each point
  - Sort by radius
ANMS : algo

Sort $DetectedPoints$ by decreasing response

$p_1 \leftarrow$ point with highest response

$r_{p_1} \leftarrow +\infty$

$ProcessedPoints \leftarrow \{p_1\}$, and remove $p_1$ from $DetectedPoints$

For each detection $p \in DetectedPoints$, in decreasing strength order

$$r_p \leftarrow \min_{q \in ProcessedPoints} \| x_p - x_q \| \text{ such that } f(x_p) < c f(x_q)$$

[as $f(x_p) > c f(x_q)$ guaranteed for $q \notin ProcessedPoints$]

add $p$ to $ProcessedPoints$

Return $n$ first points $p$ with the highest suppression radius $r_p$

// Still quadratic in number of points. (But there are subquadratic algorithms.)

// Compute, store and compare $r^2$ rather than $r$ to avoid computing a square root for $r = \| x_p - x_q \|$
ANMS: example

4 strongest points:

4 strongest points with NMS:

4 strongest points with ANMS:
ANMS: example

4 strongest points: 110, 105, 97, 96
4 strongest points with NMS:
4 strongest points with ANMS:
ANMS: example

4 strongest points: 110, 105, 97, 96
4 strongest points with NMS: 110, 105, 94, 93
4 strongest points with ANMS:
ANMS: example

4 strongest points: 110, 105, 97, 96
4 strongest points with NMS: 110, 105, 94, 93
4 strongest points with ANMS: 110, 105, 94, 78
Blob detection
Covariance / Invariance
Covariance / Invariance

• We want the description to be *invariant*:
  \[ \text{features(\text{transform(image)})} = \text{features(image)} \]

• A solution is to have a *covariant* detection:
  \[ \text{det(\text{transform(image)})} = \text{transform(det(image))} \]
Rotation invariance/covariance

- example: use the dominant gradient direction to rotate the image
Affine invariance/covariance

- example: use the eigen-decomposition of the second moment matrix

\[
M = \begin{pmatrix}
  L_x^2 & L_x L_y \\
  L_x L_y & L_y^2
\end{pmatrix}
\]

- Give direction of maximum and minimum variation of the image and a characteristic scale

-> Normalize the image \( x' \rightarrow M^{\frac{1}{2}} x \)
Evaluation
Evaluation

• The main criteria for a detector is repeatability, i.e. to detect the same features in two different views of the same scene.

• Another criteria is the number of features detected/image

• ”Same” can mean different things depending on the feature type (location, scale, orientation...)
Evaluation

- **Setting** (Schmid et al. 2000, Mikolajczyk & Schmid 2001, 2002)
  - images of planar scenes
  - known homography and scale transformations

- **Location error**
  - detected points $x_a$ in $I$, $x_b$ in $I'$
  - $I$ and $I'$ related by homography $H$: $I = H(I')$
  - $\epsilon_{\text{pos}} = \| x_a - Hx_b \| < 1.5$ pixels means success (e.g.)

- **Scale error** (for detectors that compute a feature scale)
  - scale ratio within given factor, e.g. 1.2, means success
Evaluation

- Affinity error
  - $\hat{H}$ local affine approximation of $H$ at point $x_b$
  - $\mu_A$ and $\mu_B$ elliptical regions defined by $\mu_M = \{ x \mid x^T M x \leq 1 \}$ corresponding to Harris correlation matrices $A$ and $B$
  - Jaccard distance
    \[
    \epsilon_{\text{surf}} = 1 - \frac{\mu_A \cap (\hat{H}^T \mu_B \hat{H})}{\mu_A \cup (\hat{H}^T \mu_B \hat{H})}
    \]
  - $\epsilon_{\text{surf}} < 0.2$ means success (e.g.)
Homework: Harris detection

• Find or implement a Harris detector.

• Evaluate
  – Qualitatively, in the presence of rotation, noise, scale changes, viewpoint changes. Comment.
  – Quantitatively using synthetic images with rotation, noise, scale changes, viewpoint changes
    • Number of detection (in function of the threshold)
    • Repeatability of the detection (in function of the threshold for considering a point and to be considered a repeated detection)

• Implement and test NMS and ANMS. Evaluate again quantitatively, compare and comment.

• Send report + implemented code (C/C++ or Matlab)
Homework: advice

• Use images you deform yourself (scale, rotate, add noise, perform homography...) for your tests

• Think about what’s important for a keypoint detector in your evaluation. You can take inspiration from the graphs in «Mikolajczyk, et al. A comparison of affine region detectors. IJCV 2005» and «Lowe, D. G. Distinctive image features from scale-invariant keypoints. IJCV 2004»

• Analyze the effect of parameters, try very different scenes (conclusion on one scene are often biased)
Outline

You should know: convolutions, correlations, derivatives

1. Classical feature detection
   Harris (Corner)
   Laplacian, Hessian (Blob)

2. Classical feature description:
   SSD, ZNCC, ShapeContext, HOG, BRIEF

3. Important and many details: SIFT/SURF

4. Learning description, detection and correspondences
Feature descriptors

Many type of descriptors

• Pixel values
• Based on local image statistics (local derivatives, answer to filters...)
• Based on local histograms
• Binary comparisons
• CNN-based
• ...

Patches as descriptors

Idea: normalize/reshape the detected regions and compare the resulting patches

• How to compare patches?
  – Directly compare pixels
  – Look at a more meaningful embedding of images

• Which distance/similarity?
Comparing patches using pixel values

Two square patches $P_0$ and $P_1$ of size $w$

• L2 distance:

$$\sum_{i,j} (P_0(i,j) - P_1(i,j))^2 = \sum_{i,j} (P_0(i,j)^2 + P_1(i,j)^2) - 2 \sum_{i,j} P_0(i,j) P_1(i,j)$$

Sensitive to illumination changes – average luminosity of the patch
Comparing patches using pixel values

Two square patches $P_0$ and $P_1$ of size $w$

- Zero-mean Normalized Cross-correlation (ZNCC)

$$
\frac{1}{w} \sum_{i,j} \frac{P_0(i, j) - \mu_0}{\sigma_0} \cdot \frac{P_1(i, j) - \mu_1}{\sigma_1}
$$

$$
\mu_k = \frac{1}{w} \sum_{i,j} P_k(i, j)
$$

$$
\sigma_k^2 = \frac{1}{w} \sum_{i,j} (P_k(i, j) - \mu_k)^2
$$

Invariant to affine illumination changes, robust to noise.

-> Problem: still limited robustness
Using binary comparisons between random locations

\[
\tau(p; x, y) := \begin{cases} 
1 & \text{if } p(x) < p(y) \\
0 & \text{otherwise}
\end{cases}
\]

\[
f_{n_{d}}(p) := \sum_{1 \leq i \leq n_{d}} 2^{i-1} \tau(p; x_{i}, y_{i})
\]

Calonder, M., Lepetit, V., Strecha, C., & Fua, P. Brief: Binary robust independent elementary features. *ECCV 2010*

See also Local Binary Patterns (LBP)
Shape context

1. Detect edges ; 2. Sample points ; 3. Build histogram

Belongie, S., Malik, J., & Puzicha, J.
Shape matching and object recognition using shape contexts. *PAMI 2002*
Histograms of Oriented Gradients

Same idea as SIFT: histogram of gradients orientations

Evaluation for sparse matching

• Can only be evaluated together with a detector / for a specific dataset.

• The main criteria is the proportion of valid matches between the local features.
Measuring matching performance

- true positives (TP): correct matches
- false positives (FP): proposed matches that are incorrect
- true negatives (TN): non-matches correctly rejected
- false negatives (FN): matches considered not matching

Combinations
- positive predictive value (precision) : PPV = TP/(TP+TN)
- true positive rate (recall): TPR = TP/(TP+FN) = TP/P
- false positive rate: FPR = FP/(FP+TN) = FP/N
- accuracy: ACC = (TP+TN)/(P+N)
- F-measure: 2 (PPV TPR)/(PPV + TPR) [harmonic mean]
- Area under ROC curve
ROC curve

- Plot (FPR, TPR) for one varying parameter of the matching strategy
- Good performance:
  - close to the upper left corner
  - large area under the curve (AUC)
Outline

You should know: convolutions, correlations, derivatives

1. Classical feature detection
   Harris (Corner)
   Laplacian, Hessian (Blob)

2. Classical feature description:
   SSD, ZNCC, ShapeContext, HOG, BRIEF

3. Important and many details: SIFT/SURF

4. Learning description, detection and correspondences
SIFT

• Scale Invariant Feature Transform, David Lowe, ICCV 1999, IJCV 2004
• Detector + descriptor
• Optimized for speed and precision, designed using performance over synthetic transformations (rotation, scaling, affine stretch, change in brightness and contrast, and addition of image noise)
• Still the main baseline for sparse features, even if deep methods can lead to better descriptors
Approximating LoG with DoG

- Property of Gaussian:
  \[
  \frac{\partial G}{\partial \sigma} = \sigma \nabla^2 G
  \]
  \(\leftrightarrow\) solution to heat diffusion equation

- Finite difference approximation
  \[
  \frac{\partial G}{\partial \sigma} \approx (G(x,y;k\sigma) - G(x,y;\sigma)) / (k\sigma - \sigma) \text{ for } k \approx 1
  \]
  - hence \( G(x,y;k\sigma) - G(x,y;\sigma) \approx (k - 1)\sigma^2 \nabla^2 G \)

- Instead of searching extrema via \( \sigma^2 \nabla^2 G \) convolution, look for extrema via \( G(x,y;k\sigma) - G(x,y;\sigma) \)
  \(\rightarrow\) simpler, faster
Efficiency: Difference of Gaussian

\[ L = \sigma^2 \left( G_{xx}(x, y, \sigma) + G_{yy}(x, y, \sigma) \right) \]

(Laplacian)

\[ DoG = G(x, y, k\sigma) - G(x, y, \sigma) \]

(Difference of Gaussians)
Approximating LoG with DoG

- **Difference of Gaussian (DoG)**
  \[
  D(x,y; \sigma) = (G(x,y;k\sigma) - G(x,y;\sigma)) * I(x,y) \\
  = L(x,y;k\sigma) - L(x,y;\sigma)
  \]

- **$k$: constant for all scales**
  - approximation OK when $k \rightarrow 1$
  - in practice, even OK for $k = \sqrt{2}$

- **Basic idea:**
  - sample scales for scale-space exploration
  - subtract blurred images at successive scales
Efficient computation of scale-space: DoG

- Geometric progression of scales with ratio $k = 2^{1/s}$
- Successive convolutions
- At each octave (i.e., every sample $s \rightarrow$ scale factor of 2), resample image
  - every second pixel in each row and column
  - no accuracy loss
  - space & time efficient
Extremum in scale-space
Scale discretization parameter

- Highest repeatability: $s = 3$ samples per octave

- reason: more samples $\rightarrow$ more features, but less stable

- depends on use (and test images):
  - OK if $s$ greater for object recognition (quantity vs quality)
Accurate localization

- Subpixel-subsacle optimization (Brown and Lowe 2002)
  - fit a 3D quadratic function to local sample points
  - interpolate location of extremum
  - → substantial improvement to matching and stability
- Taylor expansion up to 2\textsuperscript{nd} order at sample point
  - \( \mathbf{x} = (x, y, \sigma)^T \): offset from sample point
  - \( D \) and its derivatives evaluated at sample point

\[
\tilde{D}(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}} \mathbf{x} + \frac{1}{2} \mathbf{x}^T \frac{\partial^2 D}{\partial \mathbf{x}^2} \mathbf{x}
\]
Accurate localization

- Extremum $\hat{x}$ such that $\frac{\partial D}{\partial \hat{x}}(\hat{x}) = 0$ i.e., $\hat{x} = -\frac{\partial^2 D^{-1}}{\partial x^2} \frac{\partial D}{\partial x}$
  - Hessian and derivative via finite difference at neighborhood
  - solve 3x3 linear system

- If any dimension of offset $\hat{x} > 0.5$
  - extremum is closer to a different sample point
  - change sample point for interpolation
Rejection of unstable keypoints with low contrast

• DoG value at extremum
  - using above equations

\[
\tilde{D}(\hat{x}) = D + \frac{1}{2} \frac{\partial D^T}{\partial x} \hat{x}
\]

• Discard extrema such that \(|\tilde{D}(\hat{x})| < 0.03\) (e.g.)
  (assuming pixels range in [0,1])
Eliminating edge responses

- Problem: strong response along edges
  - large principal curvature across edge
  - small one in the perpendicular direction
    - unstable, location poorly determined
- Solution: check curvatures (cf. Harris and Stephens 1988)
  - use trace and determinant rather than explicit eigenvalues

\[
H = \begin{pmatrix}
D_{xx} & D_{xy} \\
D_{xy} & D_{yy}
\end{pmatrix}
\]

\[
\text{trace}(H) = D_{xx} + D_{yy} = \lambda_0 + \lambda_1
\]

\[
\text{det}(H) = D_{xx} D_{yy} - D_{xy}^2 = \lambda_0 \lambda_1
\]
Eliminating edge responses

- Assume $\lambda_0 \leq \lambda_1$, let $r = \frac{\lambda_1}{\lambda_0}$, then
  $$\frac{\text{trace}(H)^2}{\det(H)} = \frac{(\lambda_0 + \lambda_1)^2}{\lambda_0 \lambda_1} = \frac{(r+1)^2}{r}$$

- Ratio $(r+1)^2/r$
  - is at minimum when eigenvalues are equal
  - increases with $r$

- Discard any extremum such that $\frac{\text{trace}(H)^2}{\det(H)} > \frac{(r_{\text{max}}+1)^2}{r_{\text{max}}}$
  - in practice (Lowe 2004): $r_{\text{max}} = 10$
Orientation assignment

- Principle:
  find stable orientation at keypoint
  normalize w.r.t. orientation for rotation invariance
Orientation assignment

- Orientation = dominant direction of local gradient

\[ I_x = \frac{1}{2} (L(x+1, y) - L(x-1, y)), \quad I_y = \frac{1}{2} (L(x, y+1) - L(x, y-1)) \]

\[ m(x, y) = \sqrt{I_x^2 + I_y^2}, \quad \theta(x, y) = \arctan \left( \frac{I_y}{I_x} \right) \]

- Solution 1: average gradient in region near keypoint
  - can be small, unreliable indicator of orientation

- Solution 2: histogram of orientations near keypoint
  - 36 bins covering 360°
  - weighted by magnitude \( m \)
  - weighted by Gaussian window with scale \( 1.5 \times \sigma_{\text{keypoint}} \)
Orientation assignment

- Dominant directions = histogram peaks
  - keep highest peak
  - also keep all directions within 80% of dominant one (→ +15%)
    ▪ contributes to matching stability
    ▪ in practice, extra directions implemented as additional features

- Accurate orientation
  - interpolate peak position, i.e., gradient orientation
    ▪ use 3 histogram values closest to peak, fit parabola
Orientation repeatability

- Repeatability measure:
  - images randomly rotated and scaled + random pixel noise
  - 2\textsuperscript{nd} line of graph: success if orientation difference $< 15^\circ$
  - standard deviation
    - $2.5^\circ$ when no noise
    - $3.9^\circ$ when 10\% noise
      - only 5\% loss in repeat.
  - major cause of error
    - imprecision in feature location and scale
SIFT descriptor: use local gradients

- Inspired by biological vision (Edelman et al. 1997)
  complex neurons in primary visual cortex respond to a
  gradient at a particular orientation and spatial frequency

- Sample gradient orientation around keypoint
  + some normalization
  + rotate relative to keypoint orientation
  at keypoint scale
SIFT descriptor

- Descriptor sampling region
  - gradients at keypoint scale $\sigma_{\text{keypoint}}$
  - grid size around keypoint (Lowe: 4x4)
  - nb of (scaled) pixels / grid cell (Lowe: 4x4, Vedaldi: 3x3)

- Gaussian weight on gradient magnitude
  - less emphasis far from center (affected by misregistration)
  - prevention of sudden changes
  - deviation related to grid width

[ 0.5 grid width, shown here on 2x2 grid of 4x4-pixel cells ]
SIFT: gradient orientation histogram

- orientation bins (Lowe: 8), weighted by magnitude
- for each pixel of a given grid cell (Lowe: 4x4)
  [shown: 2x2 grid of 4x4-pixel cells]
SIFT: gradient orientation histogram
SIFT: Histogram smoothing

Figure 5: The effect of interpolation when generating the SIFT descriptor. *A single pixel's gradient*

- Avoid bin boundary effects: histogram smoothing
  - distribute each gradients into adjacent bins
  - trilinear interpolation (for each dimension: x, y, orientation)
    - multiply by weight \(1 - d\) where \(d\) distance to the central bin value
SIFT: robustness to illumination changes

Reduce effects of illumination change

• affine contrast change:
  – normalize vector to unit length

• non linear illumination (camera saturation, highlights...): need to reduce the influence of large gradient magnitudes
  – max threshold value in normalized vector: not larger than 0.2
  – renormalize feature vector

-> greater emphasis on distribution of orientations
SIFT matching

• Measure of similarity between descriptors
  – Euclidian distance

• Descriptor matching test (to reduce nb of outliers)
  – > fixed threshold on distance
    • difficult to set (but can be learnt)
    • varies a lot depending on feature space regions
  – > comparison with neighbors
SIFT matching

• Matching with nearest neighbor (in 128D-space)
  - OK if 2\textsuperscript{nd} nearest neighbor at least 20\% (e.g.) more distant
    ▪ score of $D_A$: \(|D_A - D_B| / |D_A - D_C|\), OK if score < 0.8 (e.g.)

\begin{itemize}
  \item 2\textsuperscript{nd} nearest neighbor \sim rough estimation of false match density
  \item discards (some) ambiguous matches
  \item typically eliminates 90\% false matches and 5\% correct matches
\end{itemize}

- orange match blue?
- red match green?
SIFT

- Tons of parameters (sizes, thresholds, etc.)
  - “good” parameters found by experimentation
    - possible bias towards used image database

- Many tiny details, some unsaid at all

- Most likely no 2 implementations give the same result
SURF

• Inspired by SIFT
• Faster, using approximations and integral images

Bay, H., Tuytelaars, T., & Van Gool, Surf: Speeded up robust features, *ECCV 2006*
What should you remember?

• General ideas:
  – Invariance/covariance, derivative, scale space

• Detectors:
  – Harris, Laplacian (LoG/DoG), Hessian, NMS/ANMS

• Descriptors:
  – Histogram, NN

• Lots of small but very practical idea:
  – Evaluations, 1st to 2nd NN ratio, soft assignment...
Outline

You should know: convolutions, correlations, derivatives

1. Classical feature detection
   Harris (Corner)
   Laplacian, Hessian (Blob)

2. Classical feature description:
   SSD, ZNCC, ShapeContext, HOG, BRIEF

3. Important and many details: SIFT/SURF

4. Learning description, detection and correspondences
Learning features
CNNs/Deep features

Standard CNNs (eg. AlexNet):

- Succession of convolutions, non linearities (ReLU) and max-poolings
- Trained for image classification (1 million images from ImageNet)

Fischer, P., Dosovitskiy, A., & Brox, T., Descriptor matching with convolutional neural networks: a comparison to SIFT. *arXiv preprint 2014*

Conv 4 features seem generic and outperform SIFTs
Descriptor learning

Idea: create a large database of ground truth local feature matches using the 3D of reconstructed scenes and use it to learn the parameter of a descriptor.

One of the first: M. Brown, G. Hua, and S. Winder. Discriminative learning of local image descriptors. PAMI 2011

-> 0.5 million pairs
Going larger scale

- 0.5 billion correspondences from Google Street View

Zamir, A. R., Wekel, T., Agrawal, P., Wei, C., Malik, J., & Savarese, S. Generic 3D Representation via Pose Estimation and Matching, ECCV 2016
Going larger scale

• Large datasets have been curated for SFM

73 models

200 models
Li, Z., & Snavely, N.
Megadepth: Learning single-view depth prediction from internet photos. CVPR 2018

-> can be used for training, but accuracy not perfect
Deep feature descriptors

Idea: learning to compare features using a large database of ground truth correspondences

Zagoruyko, S., & Komodakis, N. Learning to Compare Image Patches via Convolutional Neural Networks. *CVPR 2015*
Questions

• Architecture e.g. decision network with early vs. late fusion

• In recent works, simply feature comparison with cosine/L2 distance

• Losses: e.g. classification, L1/L2, triplet loss, hard negative mining...
ContextDesc

Luo, Z., Shen, T., Zhou, L., Zhang, J., Yao, Y., Li, S., ... & Quan, L.  
ContextDesc: Local Descriptor Augmentation with Cross-Modality Context, CVPR 2019
ContextDesc

Luo, Z., Shen, T., Zhou, L., Zhang, J., Yao, Y., Li, S., ... & Quan, L.  
ContextDesc: Local Descriptor Augmentation with Cross-Modality Context, CVPR 2019
Deep feature detection and description

2 key elements:

- Soft-argmax: similar to softmax

- Spatial transformer network: cropping is differentiable

Jaderberg, M., Simonyan, K., & Zisserman, A. Spatial transformer networks. NIPS 2015
LIFT

- Stay close to SIFT pipeline

Note: every operation can be made differentiable (softargmax, spatial transformer networks)

Yi, K. M., Trulls, E., Lepetit, V., & Fua, P.
Lift: Learned invariant feature transform, ECCV 2016
D2-Net

Dusmanu, M., Rocco, I., Pajdla, T., Pollefeys, M., Sivic, J., Torii, A., & Sattler, T.
D2-Net: A Trainable CNN for Joint Detection and Description of Local Features. CVPR 2019
Learning

transformation/correspondences
Learning transformation

• By learning to predict transformation

I. Rocco, R. Arandjelović and J. Sivic, CVPR 2017
Convolutional neural network architecture for geometric matching
Learning transformation

• By learning to predict transformation

I. Rocco, R. Arandjelović and J. Sivic, CVPR 2017
Convolutional neural network architecture for geometric matching
Learning to match features

- Using consistency

\[ r_{ijkl}^A = \frac{c_{ijkl}}{\max_{ab} c_{abkl}} \quad \text{and} \quad r_{ijkl}^B = \frac{c_{ijkl}}{\max_{cd} c_{ijcd}} \]

\[ s_{ijkl}^A = \frac{\exp(c_{ijkl})}{\sum_{ab} \exp(c_{abkl})} \quad \text{and} \quad s_{ijkl}^B = \frac{\exp(c_{ijkl})}{\sum_{cd} \exp(c_{ijcd})} \]

Rocco, I., Cimpoi, M., Arandjelović, R., Torii, A., Pajdla, T., & Sivic, J
Neighbourhood consensus networks. NIPS 2018
Learning to match features

• Using weak supervision only
  (pairs of matching and non matching images)

$$\mathcal{L}(I^A, I^B) = -y \left( \bar{s}^A + \bar{s}^B \right)$$

y=1 for positive pairs, -1 for negative pairs

Rocco, I., Cimpoi, M., Arandjelović, R., Torii, A., Pajdla, T., & Sivic, J
Neighbourhood consensus networks. NIPS 2018
Learning to match features

\[
CN \left( o_i^l \right) = \frac{\left( o_i^l - \mu^l \right)}{\sigma^l}
\]

\[
\mu^l = \frac{1}{N} \sum_{i=1}^{N} o_i^l, \quad \sigma^l = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (o_i^l - \mu^l)^2}
\]

Moo Yi, K., Trulls, E., Ono, Y., Lepetit, V., Salzmann, M., & Fua, P.
Learning to find good correspondences. CVPR 2018
Evaluation

• Remains an important challenge, as datasets with good Ground Truth are rare and small.

• A solution can be to evaluate another task than 3D reconstruction, for which GT is easier to get, e.g. localization.
Conclusion

• Detection-description powerful idea, part of SFM success with classical detector-descriptors as SIFT

• Lots of classical approaches are being “deepified”, i.e. formulated as modular and end-to-end learnable framework, with important performance gains

• Tricks from classical approach often remain important in NN-architectures
Homework: Harris detection

• Find or implement a Harris detector.

• Evaluate
  – Qualitatively, in the presence of rotation, noise, scale changes, viewpoint changes. **Comment.**
  – **Quantitatively using synthetic images** with rotation, noise, scale changes, viewpoint changes
    • Number of detection (in function of the threshold)
    • Repeatability of the detection (in function of the threshold for considering a point and to be considered a repeated detection)

• Implement and test NMS and ANMS. **Evaluate again quantitatively, compare and comment.**

• Send report + implemented code (C/C++ or Matlab)
Homework: advice

• Use images you deform yourself (scale, rotate, add noise, perform homography...) for your tests

• Think about what’s important for a keypoint detector in your evaluation. You can take inspiration from the graphs in « Mikolajczyk, et al. A comparison of affine region detectors. IJCV 2005 »

• Analyze the effect of parameters, try very different scenes (conclusion on one scene are often biaised)