Local invariant feature detection

Thurs Feb 23
Kristen Grauman
UT Austin

Last time

• Fitting an arbitrary shape with “active” deformable contours
Deformable contours
a.k.a. active contours, snakes

**Given:** initial contour (model) near desired object

**Goal:** evolve the contour to fit exact object boundary

**Main idea:** elastic band is iteratively adjusted so as to
- be near image positions with high gradients, and
- satisfy shape “preferences” or contour priors

Deformable contours: intuition
Limitations

- May over-smooth the boundary

- Cannot follow topological changes of objects

Limitations

- External energy: snake does not really “see” object boundaries in the image unless it gets very close to it.

image gradients $\nabla I$ are large only directly on the boundary
Distance transform

• External image can instead be taken from the distance transform of the edge image.

Value at (x,y) tells how far that position is from the nearest edge point (or other binary image structure)

>> help bwdist

Aspects we need to consider

• Representation of the contours
• Defining the energy functions
  – External
  – Internal
• Minimizing the energy function
• Extensions:
  – Tracking
  – Interactive segmentation
Tracking via deformable contours

1. Use final contour/model extracted at frame $t$ as an initial solution for frame $t+1$
2. Evolve initial contour to fit exact object boundary at frame $t+1$
3. Repeat, initializing with most recent frame.

Applications:
- Traffic monitoring
- Human-computer interaction
- Animation
- Surveillance
- Computer assisted diagnosis in medical imaging

http://www.robots.ox.ac.uk/~vdg/
3D active contours

Interactive forces

How can we implement such an interactive force with deformable contours?

Slide credit: Kristen Grauman
Interactive forces

- An energy function can be altered online based on user input – use the cursor to push or pull the initial snake away from a point.
- Modify external energy term to include:

\[ E_{\text{push}} = \sum_{i=0}^{n-1} \frac{r^2}{|v_i - p|^2} \]

Nearby points get pushed hardest

Intelligent scissors

Another form of interactive segmentation:

Compute optimal paths from every point to the seed based on edge-related costs.

[Figure 2: Image demonstrating how the live-wire segment adapts and snaps to an object boundary as the free point moves (via cursor movement). The path of the free point is shown in white. Live-wire segments from previous free point positions (\(t_0, t_1,\) and \(t_2\)) are shown in green.]

[Mortensen & Barrett, SIGGRAPH 1995, CVPR 1999]
Intelligent scissors

Deformable contours: pros and cons

Pros:
- Useful to track and fit non-rigid shapes
- Contour remains connected
- Possible to fill in "subjective" contours
- Flexibility in how energy function is defined, weighted.

Cons:
- Must have decent initialization near true boundary, may get stuck in local minimum
- Parameters of energy function must be set well based on prior information

Slide credit: Kristen Grauman
Recap: Deformable contours

- Deformable shapes and active contours are useful for
  - Segmentation: fit or “snap” to boundary in image
  - Tracking: previous frame’s estimate serves to initialize the next

- Fitting active contours:
  - Define terms to encourage certain shapes, smoothness, low curvature, push/pulls, …
  - Use weights to control relative influence of each component cost
  - Can optimize 2d snakes with Viterbi algorithm.

- Image structure (esp. gradients) can act as attraction force for interactive segmentation methods.

Previously: Features and filters
Transforming and describing images; textures, colors, edges
Previously: Grouping & fitting

- Parallelism
- Symmetry
- Continuity
- Cloure

Clustering, segmentation, fitting; what parts belong together?

Now: Multiple views

Matching, invariant features, stereo vision, instance recognition

Slide credit: Kristen Grauman
Important tool for multiple views: Local features

Multi-view matching relies on local feature correspondences.

How to detect which local features to match?

Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

\[
x_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]
\]

3) Matching: Determine correspondence between descriptors in two views

\[
x_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]
\]

Slide credit: Kristen Grauman
Goal: interest operator repeatability

- We want to detect (at least some of) the same points in both images.
- Yet we have to be able to run the detection procedure independently per image.

No chance to find true matches!

Goal: descriptor distinctiveness

- We want to be able to reliably determine which point goes with which.
- Must provide some invariance to geometric and photometric differences between the two views.
Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

3) Matching: Determine correspondence between descriptors in two views

• What points would you choose?
Detecting corners

Compute “cornerness” response at every pixel.
Detecting corners

Detecting local invariant features

- Detection of interest points
  - Harris corner detection
  - Scale invariant blob detection: LoG
- (Next time: description of local patches)
Corners as distinctive interest points

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

Slide credit: Alyosha Efros, Darya Frolova, Denis Simakov

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

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

Notation:

\[
I_x \Leftrightarrow \frac{\partial I}{\partial x} \quad I_y \Leftrightarrow \frac{\partial I}{\partial y} \quad I_x I_y \Leftrightarrow \frac{\partial I}{\partial x} \frac{\partial I}{\partial y}
\]
What does this matrix reveal?

First, consider an axis-aligned corner:

\[
M = \sum \begin{bmatrix}
I_x^2 & I_x I_y \\
I_x I_y & I_y^2
\end{bmatrix} = \begin{bmatrix}
\lambda_1 & 0 \\
0 & \lambda_2
\end{bmatrix}
\]

This means dominant gradient directions align with \(x\) or \(y\) axis.

Look for locations where both \(\lambda\)'s are large.

If either \(\lambda\) is close to 0, then this is not corner-like.

What if we have a corner that is not aligned with the image axes?
What does this matrix reveal?

Since $M$ is symmetric, we have

$$M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T$$

$$Mx_i = \lambda_i x_i$$

The eigenvalues of $M$ reveal the amount of intensity change in the two principal orthogonal gradient directions in the window.

Corner response function

"edge":
$$\lambda_1 \gg \lambda_2$$
$$\lambda_2 \gg \lambda_1$$

"corner":
$$\lambda_1$$ and $$\lambda_2$$ are large,
$$\lambda_1 \sim \lambda_2;$$

"flat" region
$$\lambda_1$$ and $$\lambda_2$$ are small;

Cornerness score
(other variants possible)

$$f = \frac{\lambda_1 \lambda_2}{\lambda_1 + \lambda_2}$$
Harris corner detector

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

Harris Detector: Steps
### Harris Detector: Steps

1. **Compute corner response** $f$

   ![Image of Harris Detector response](image1)

2. **Find points with large corner response**: $f > \text{threshold}$

   ![Image of thresholded Harris Detector](image2)
Harris Detector: Steps

Take only the points of local maxima of $f$
Properties of the Harris corner detector

Rotation invariant?  Yes

\[ M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T \]

Scale invariant?  No

All points will be classified as edges

Corner !
Scale invariant interest points

How can we independently select interest points in each image, such that the detections are repeatable across different scales?

Automatic Scale Selection

How to find corresponding patch sizes, with only one image in hand?
Automatic scale selection

**Intuition:**
- Find scale that gives local maxima of some function $f$ in both position and scale.

**Automatic Scale Selection**
- Function responses for increasing scale (scale signature)
Automatic Scale Selection

- Function responses for increasing scale (scale signature)
Automatic Scale Selection

• Function responses for increasing scale (scale signature)

\[ f(I_{x,\sigma}(x, \sigma)) \quad f(I_{x',\sigma}(x', \sigma)) \]

K. Grauman, B. Leibe
Automatic Scale Selection

- Function responses for increasing scale (scale signature)

What can be the “signature” function?
Blob detection in 2D

Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D

\[ \nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \]

Blob detection in 2D: scale selection

Laplacian-of-Gaussian = “blob” detector

\[ \nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \]
Blob detection in 2D

We define the characteristic scale as the scale that produces peak of Laplacian response.

Example

Original image at 3/4 the size

Slide credit: Lana Lazebnik

Slide credit: Kristen Grauman
Scale invariant interest points

Interest points are local maxima in both position and scale.

\[ L_{xx}(\sigma) + L_{yy}(\sigma) \rightarrow \] 

\[ \Rightarrow \text{List of } (x, y, \sigma) \]

Scaled filter response maps

Slide credit: Kristen Grauman

Scale-space blob detector: Example

Original image | Scale-space maxima of \( (\nabla_{\text{norm}}^2 L)^2 \)

Scale-space blob detector: Example

We can approximate the Laplacian with a difference of Gaussians; more efficient to implement.

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

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

\[ I(k\sigma) - I(\sigma) = DoG \]
Summary

• Desirable properties for local features for correspondence
• Basic matching pipeline
• Interest point detection
  – Harris corner detector
  – Laplacian of Gaussian, automatic scale selection

Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

3) Matching: Determine correspondence between descriptors in two views

\[ x_1 = [x_1^{(1)}, \ldots, x_d^{(1)}] \]
\[ x_2 = [x_1^{(2)}, \ldots, x_d^{(2)}] \]