Feature Tracking and Optical Flow

Computer Vision
Jia-Bin Huang, Virginia Tech

Many slides from D. Hoiem
Administrative Stuffs

• HW 1 due 11:59 PM Sept 25
  • Submission through Canvas

• HW 1 Competition: Edge Detection
  • Submission link
HW 1: frequently asked questions

• Hybrid image

• Image pyramid
  • Cannot use `impyramid`
  • Make sure that Laplacian pyramid is displayed properly

• Edge detection
  • Can use `edge(im, 'canny')` to get edge pixels
  • Need to compute scores
Feature matching

Given a feature in $I_1$, how to find the best match in $I_2$?

1. Define distance function that compares two descriptors
2. Test all the features in $I_2$, find the one with min distance
Feature distance

How to define the difference between two features $f_1, f_2$?

- Simple approach: $L_2$ distance, $||f_1 - f_2||$
- can give good scores to ambiguous (incorrect) matches
How to define the difference between two features $f_1, f_2$?

- Better approach: ratio distance $= \| f_1 - f_2 \| / \| f_1 - f_2' \|$
- $f_2$ is best SSD match to $f_1$ in $I_2$
- $f_2'$ is 2nd best SSD match to $f_1$ in $I_2$
- gives large values for ambiguous matches
Feature matching example

51 matches
Feature matching example

58 matches
Evaluating the results

How can we measure the performance of a feature matcher?
True/false positives

How can we measure the performance of a feature matcher?

The distance threshold affects performance

• True positives = # of detected matches that are correct
  • Suppose we want to maximize these—how to choose threshold?
• False positives = # of detected matches that are incorrect
  • Suppose we want to minimize these—how to choose threshold?
Matching SIFT Descriptors

• Nearest neighbor (Euclidean distance)
• Threshold ratio of nearest to 2\textsuperscript{nd} nearest descriptor

\begin{figure}
\centering
\includegraphics[width=0.8\textwidth]{sift_matching.pdf}
\caption{PDF for correct matches (dashed line) and incorrect matches (solid line).}
\end{figure}

Lowe IJCV 2004
SIFT Repeatability

![Graph showing the repeatability of SIFT features vs. image noise. The graph plots the percentage of repeatable keypoints against the percentage of image noise. Keylines represent different matching criteria: matching location and scale, matching location, scale, and orientation, and nearest descriptor in database. The graph shows a decrease in repeatability as image noise increases.](image-url)
SIFT Repeatability

Matching location, scale, and orientation
Nearest descriptor in database

Lowe IJCV 2004
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

Many other efficient descriptors are also available

[Bay, ECCV’06], [Cornelis, CVGPU’08]
Local Descriptors: Shape Context

Count the number of points inside each bin, e.g.:

Count = 4

Count = 10

Log-polar binning: more precision for nearby points, more flexibility for farther points.

Belongie & Malik, ICCV 2001
Local Descriptors: Geometric Blur

Example descriptor

Compute edges at four orientations
Extract a patch in each channel

Apply spatially varying blur and sub-sample

(Idealized signal)

Berg & Malik, CVPR 2001

K. Grauman, B. Leibe
Choosing a detector

• What do you want it for?
  – Precise localization in x-y: Harris
  – Good localization in scale: Difference of Gaussian
  – Flexible region shape: MSER

• Best choice often application dependent
  – Harris-/Hessian-Laplace/DoG work well for many natural categories
  – MSER works well for buildings and printed things

• Why choose?
  – Get more points with more detectors

• There have been extensive evaluations/comparisons
  – [Mikolajczyk et al., IJCV’05, PAMI’05]
  – All detectors/descriptors shown here work well
Comparison of Keypoint Detectors

Table 7.1 Overview of feature detectors.

<table>
<thead>
<tr>
<th>Feature Detector</th>
<th>Corner</th>
<th>Blob</th>
<th>Region</th>
<th>Rotation invariant</th>
<th>Scale invariant</th>
<th>Affine invariant</th>
<th>Repeatability</th>
<th>Localization accuracy</th>
<th>Robustness</th>
<th>Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Harris</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>+++</td>
<td>++</td>
</tr>
<tr>
<td>Hessian</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>SUSAN</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Harris-Laplace</td>
<td>✓</td>
<td>(✓)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Hessian-Laplace</td>
<td>(✓)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>+++</td>
<td>+</td>
</tr>
<tr>
<td>DoG</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>SURF</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>++</td>
<td>++</td>
<td>++</td>
<td>+++</td>
</tr>
<tr>
<td>Harris-Affine</td>
<td>✓</td>
<td>(✓)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Hessian-Affine</td>
<td>(✓)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>+++</td>
<td>++</td>
</tr>
<tr>
<td>Salient Regions</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Edge-based</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>MSER</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+++</td>
<td>+++</td>
<td>++</td>
<td>+++</td>
</tr>
<tr>
<td>Intensity-based</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>++</td>
<td>+</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>Superpixels</td>
<td>✓</td>
<td>✓</td>
<td>(✓)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

Tuytelaars Mikolajczyk 2008
Choosing a descriptor

• Again, need not stick to one

• For object instance recognition or stitching, SIFT or variant is a good choice
Recent advances in interest points

Binary feature descriptors

- **BRIEF**: Binary Robust Independent Elementary Features, ECCV 10
- **ORB (Oriented FAST and Rotated BRIEF)**, CVPR 11
- **BRISK**: Binary robust invariant scalable keypoints, ICCV 11
- **Freak**: Fast retina keypoint, CVPR 12
- **LIFT**: Learned Invariant Feature Transform, ECCV 16
Previous class

• Interest point/keypoint/feature detectors
  • Harris: detects corners
  • DoG: detects peaks/troughs

• Interest point/keypoint/feature descriptors
  • SIFT (do read the paper)

*Distinctive image features from scale-invariant keypoints*

DG Lowe - International journal of computer vision, 2004 - Springer

Abstract This paper presents a method for extracting distinctive invariant features from images that can be used to perform reliable matching between different views of an object or scene. The features are invariant to image scale and rotation, and are shown to provide ...

Cited by 36572  Related articles  All 214 versions  Web of Science: 13301  Import into BibTeX

• Feature matching
  • Ratio distance = \[ \left| \frac{||f_1 - f_2||}{||f_1 - f_2'||} \right| \]
  • Remove 90% false matches, 5% of true matches in Lowe’s study
This class: recovering motion

• Feature tracking
  • Extract visual features (corners, textured areas) and “track” them over multiple frames

• Optical flow
  • Recover image motion at each pixel from spatio-temporal image brightness variations

Two problems, one registration method

Feature tracking

• Many problems, such as structure from motion require matching points
• If motion is small, tracking is an easy way to get them
Feature tracking - Challenges

• Figure out which features can be tracked
• Efficiently track across frames
• Some points may change appearance over time (e.g., due to rotation, moving into shadows, etc.)
• Drift: small errors can accumulate as appearance model is updated
• Points may appear or disappear: need to be able to add/delete tracked points
Feature tracking

• Given two subsequent frames, estimate the point translation

• Key assumptions of Lucas-Kanade Tracker
  • **Brightness constancy:** projection of the same point looks the same in every frame
  • **Small motion:** points do not move very far
  • **Spatial coherence:** points move like their neighbors
The brightness constancy constraint

\[
(x, y) \quad \text{displacement} = (u, v)
\]

\[
I(x, y, t)
\]

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

\[
I(x, y, t+1)
\]

• Brightness Constancy Equation:

\[
I(x, y, t) = I(x + u, y + v, t + 1)
\]

Take Taylor expansion of \(I(x+u, y+v, t+1)\) at \((x,y,t)\) to linearize the right side:

Image derivative along \(x\)  

\[
I(x + u, y + v, t + 1) \approx I(x, y, t) + I_x \cdot u + I_y \cdot v + I_t
\]

Difference over frames

\[
I(x + u, y + v, t + 1) - I(x, y, t) = I_x \cdot u + I_y \cdot v + I_t
\]

So:

\[
I_x \cdot u + I_y \cdot v + I_t \approx 0 \quad \rightarrow \nabla I \cdot [u \ v]^T + I_t = 0
\]
The brightness constancy constraint

Can we use this equation to recover image motion \((u,v)\) at each pixel?

\[
\nabla I \cdot [u \ v]^T + I_t = 0
\]

• How many equations and unknowns per pixel?
  • One equation (this is a scalar equation!), two unknowns \((u,v)\)

The component of the motion perpendicular to the gradient (i.e., parallel to the edge) cannot be measured

If \((u, v)\) satisfies the equation, so does \((u+u', v+v')\) if

\[
\nabla I \cdot [u' \ v']^T = 0
\]
The aperture problem

Actual motion
The aperture problem
The barber pole illusion

http://en.wikipedia.org/wiki/Barberpole_illusion
The barber pole illusion

http://en.wikipedia.org/wiki/Barberpole_illusion
Solving the ambiguity...


- How to get more equations for a pixel?

- **Spatial coherence constraint**
  - Assume the pixel’s neighbors have the same \((u,v)\)
  - If we use a 5x5 window, that gives us 25 equations per pixel

\[
0 = I_t(p_i) + \nabla I(p_i) \cdot [u \ v]
\]

\[
\begin{bmatrix}
I_x(p_1) & I_y(p_1) \\
I_x(p_2) & I_y(p_2) \\
\vdots & \vdots \\
I_x(p_{25}) & I_y(p_{25})
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix} = -
\begin{bmatrix}
I_t(p_1) \\
I_t(p_2) \\
\vdots \\
I_t(p_{25})
\end{bmatrix}
\]
Solving the ambiguity...

- Least squares problem:

\[
\begin{bmatrix}
I_x(p_1) & I_y(p_1) \\
I_x(p_2) & I_y(p_2) \\
\vdots & \vdots \\
I_x(p_{25}) & I_y(p_{25})
\end{bmatrix}
\begin{bmatrix}
 u \\
v
\end{bmatrix}
= -
\begin{bmatrix}
 I_t(p_1) \\
 I_t(p_2) \\
\vdots \\
 I_t(p_{25})
\end{bmatrix}
\]

\[
A \ d = b
\]

25x2 2x1 25x1
Matching patches across images

- Overconstrained linear system

\[
\begin{bmatrix}
I_x(p_1) & I_y(p_1) \\
I_x(p_2) & I_y(p_2) \\
\vdots & \vdots \\
I_x(p_{25}) & I_y(p_{25})
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix}
= -
\begin{bmatrix}
I_t(p_1) \\
I_t(p_2) \\
\vdots \\
I_t(p_{25})
\end{bmatrix}
\]

\[
A \quad d = b
\]

Least squares solution for \( d \) given by

\[
\begin{bmatrix}
\sum I_x I_x & \sum I_x I_y \\
\sum I_x I_y & \sum I_y I_y
\end{bmatrix}
\begin{bmatrix}
u \\
v
\end{bmatrix}
= -
\begin{bmatrix}
\sum I_x I_t \\
\sum I_y I_t
\end{bmatrix}
\]

\[
A^T A \quad A^T b
\]

The summations are over all pixels in the \( K \times K \) window
Conditions for solvability

Optimal \((u, v)\) satisfies Lucas-Kanade equation

\[
\begin{bmatrix}
\sum I_x I_x & \sum I_x I_y \\
\sum I_x I_y & \sum I_y I_y \\
\end{bmatrix}
\begin{bmatrix}
u \\
v \\
\end{bmatrix}
= -
\begin{bmatrix}
\sum I_x I_t \\
\sum I_y I_t \\
\end{bmatrix}
\]

\[AT A \]

\[AT b\]

When is this solvable? I.e., what are good points to track?

- \(AT A\) should be invertible
- \(AT A\) should not be too small due to noise
  - eigenvalues \(\lambda_1\) and \(\lambda_2\) of \(AT A\) should not be too small
- \(AT A\) should be well-conditioned
  - \(\lambda_1 / \lambda_2\) should not be too large (\(\lambda_1\) = larger eigenvalue)

Does this remind you of anything?

Criteria for Harris corner detector
Low-texture region

\[ \sum \nabla I (\nabla I)^T \]
- gradients have small magnitude
- small \( \lambda_1 \), small \( \lambda_2 \)
Edge

\[ \sum \nabla I (\nabla I)^T \]

- gradients very large or very small
- large \( \lambda_1 \), small \( \lambda_2 \)
High-texture region

\[ \sum \nabla I (\nabla I)^T \]

- gradients are different, large magnitudes
- large \( \lambda_1 \), large \( \lambda_2 \)
The aperture problem resolved
The aperture problem resolved

Perceived motion
Dealing with larger movements: Iterative refinement

1. Initialize \((x', y') = (x, y)\)
2. Compute \((u, v)\) by
   \[
   \begin{bmatrix}
   \sum I_x I_x & \sum I_x I_y \\
   \sum I_x I_y & \sum I_y I_y
   \end{bmatrix}
   \begin{bmatrix}
   u \\
   v
   \end{bmatrix}
   = - \begin{bmatrix}
   \sum I_x I_t \\
   \sum I_y I_t
   \end{bmatrix}
   \]
   2\text{nd} moment matrix for feature patch in first image
   \(I_t = I(x', y', t+1) - I(x, y, t)\)
3. Shift window by \((u, v)\): \(x' = x' + u; \ y' = y' + v\)
4. Recalculate \(I_t\)
5. Repeat steps 2-4 until small change
   • Use interpolation for subpixel values
Dealing with larger movements: coarse-to-fine registration

Gaussian pyramid of image 1 (t)  \( \rightarrow \) run iterative L-K  \( \rightarrow \) run iterative L-K  \( \rightarrow \) upsample  \( \rightarrow \) run iterative L-K  \( \rightarrow \) Gaussian pyramid of image 2 (t+1)
Shi-Tomasi feature tracker

• Find good features using eigenvalues of second-moment matrix (e.g., Harris detector or threshold on the smallest eigenvalue)
  • Key idea: “good” features to track are the ones whose motion can be estimated reliably

• Track from frame to frame with Lucas-Kanade
  • This amounts to assuming a translation model for frame-to-frame feature movement

• Check consistency of tracks by affine registration to the first observed instance of the feature
  • Affine model is more accurate for larger displacements
  • Comparing to the first frame helps to minimize drift

Tracking example

Figure 1: Three frame details from Woody Allen’s *Manhattan*. The details are from the 1st, 11th, and 21st frames of a subsequence from the movie.

Figure 2: The traffic sign windows from frames 1, 6, 11, 16, 21 as tracked (top), and warped by the computed deformation matrices (bottom).

Summary of KLT tracking

• Find a good point to track (harris corner)

• Use intensity second moment matrix and difference across frames to find displacement

• Iterate and use coarse-to-fine search to deal with larger movements

• When creating long tracks, check appearance of registered patch against appearance of initial patch to find points that have drifted
Implementation issues

• Window size
  • Small window more sensitive to noise and may miss larger motions (without pyramid)
  • Large window more likely to cross an occlusion boundary (and it’s slower)
  • 15x15 to 31x31 seems typical

• Weighting the window
  • Common to apply weights so that center matters more (e.g., with Gaussian)
Why not just do local template matching?

- Slow (need to check more locations)
- Does not give subpixel alignment (or becomes much slower)
  - Even pixel alignment may not be good enough to prevent drift
- May be useful as a step in tracking if there are large movements
Optical flow

Vector field function of the spatio-temporal image brightness variations

Picture courtesy of Selim Temizer - Learning and Intelligent Systems (LIS) Group, MIT
Motion and perceptual organization

- Even “impoverished” motion data can evoke a strong percept

Motion and perceptual organization

- Even “impoverished” motion data can evoke a strong percept

Uses of motion

• Estimating 3D structure
• Segmenting objects based on motion cues
• Learning and tracking dynamical models
• Recognizing events and activities
• Improving video quality (motion stabilization)
Motion field

- The motion field is the projection of the 3D scene motion into the image

What would the motion field of a non-rotating ball moving towards the camera look like?
Optical flow

• Definition: optical flow is the *apparent* motion of brightness patterns in the image
• Ideally, optical flow would be the same as the motion field
• Have to be careful: apparent motion can be caused by lighting changes without any actual motion
  • Think of a uniform rotating sphere under fixed lighting vs. a stationary sphere under moving illumination
Lucas-Kanade Optical Flow

• Same as Lucas-Kanade feature tracking, but for each pixel
  • As we saw, works better for textured pixels
• Operations can be done one frame at a time, rather than pixel by pixel
  • Efficient
Iterative Refinement

• Iterative Lukas-Kanade Algorithm

1. Estimate displacement at each pixel by solving Lucas-Kanade equations
2. Warp \( I(t) \) towards \( I(t+1) \) using the estimated flow field
   - Basically, just interpolation
3. Repeat until convergence

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003
Coarse-to-fine optical flow estimation

Gaussian pyramid of image 1 (t)  ➔ run iterative L-K  ➔ warp & upsample  ➔ run iterative L-K

Gaussian pyramid of image 2 (t+1)

image 1

image 2
Example
Multi-resolution registration

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003
Optical Flow Results

Lucas-Kanade without pyramids

Fails in areas of large motion

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003
Optical Flow Results

Lucas-Kanade with Pyramids

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003
Errors in Lucas-Kanade

• The motion is large
  • Possible Fix: Keypoint matching

• A point does not move like its neighbors
  • Possible Fix: Region-based matching

• Brightness constancy does not hold
  • Possible Fix: Gradient constancy
State-of-the-art optical flow

Start with something similar to Lucas-Kanade
+ gradient constancy
+ energy minimization with smoothing term
+ region matching
+ keypoint matching (long-range)

Large displacement optical flow, Brox et al., CVPR 2009
Things to remember

• Major contributions from Lucas, Tomasi, Kanade
  • Tracking feature points
  • Optical flow
  • Stereo (later)
  • Structure from motion (later)

• Key ideas
  • By assuming brightness constancy, truncated Taylor expansion leads to simple and fast patch matching across frames
  • Coarse-to-fine registration
Next week

• HW 1 due Monday

• Object/image alignment