Lecture 13:
Tracking motion features
– optical flow

Professor Fei-Fei Li
Stanford Vision Lab
What we will learn today?

- Introduction
- Optical flow
- Feature tracking
- Applications
  - (Problem Set 3 (Q1))
From images to videos

- A video is a sequence of frames captured over time.
- Now our image data is a function of space \((x, y)\) and time \((t)\).
Motion estimation techniques

• Optical flow
  – Recover image motion at each pixel from spatio-temporal image brightness variations (optical flow)

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

Vector field function of the spatio-temporal image brightness variations

Picture courtesy of Selim Temizer - Learning and Intelligent Systems (LIS) Group, MIT
Feature-tracking

Courtesy of Jean-Yves Bouguet – Vision Lab, California Institute of Technology
Feature-tracking

Courtesy of Jean-Yves Bouguet – Vision Lab, California Institute of Technology

Fei-Fei Li

Lecture 13 - 7
9-Nov-11
Optical flow

• Definition: optical flow is the apparent motion of brightness patterns in the image.

• Note: 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.

**GOAL:** Recover image motion at each pixel from optical flow.
Estimating optical flow

- Given two subsequent frames, estimate the apparent motion field $u(x,y), v(x,y)$ between them.

- Key assumptions
  - **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.

Source: Silvio Savarese
The brightness constancy constraint

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

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

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

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

- **Brightness Constancy Equation:**

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

  Linearizing the right side using Taylor expansion:

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

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

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

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

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

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

The component of the flow 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

\[
\nabla I \cdot [u' \quad v']^T = 0
\]

Source: Silvio Savarese
The aperture problem

Actual motion
The aperture problem

Perceived motion
The barber pole illusion

http://en.wikipedia.org/wiki/Barberpole_illusion

Source: Silvio Savarese
The barber pole illusion

http://en.wikipedia.org/wiki/Barberpole_illusion

Source: Silvio Savarese
Aperture problem cont’d
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

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

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

- Overconstrained linear system:

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

\[
A \quad d = b
\]

Source: Silvio Savarese
Conditions for solvability

- When is this system solvable?
  - What if the window contains just a single straight edge?
Lucas-Kanade flow

- Overconstrained linear system

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

\[A \ d = b\]

Least squares solution for \(d\) given by

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

\[A^T A \ 2 \times 2 \ 2 \times 25 \quad A^T b\]

The summations are over all pixels in the K x K window.
Conditions for solvability

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

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

\(A^T A\)

\(A^T b\)

When is This Solvable?

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

Does this remind anything to you?
$M = A^T A$ is the second moment matrix!

(Harris corner detector...)

$$A^T A = \left[ \begin{array}{cc} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{array} \right] = \sum \left[ \begin{array}{c} I_x \\ I_y \end{array} \right] [I_x I_y] = \sum \nabla I (\nabla I)^T$$

- Eigenvectors and eigenvalues of $A^T A$ relate to edge direction and magnitude
  - The eigenvector associated with the larger eigenvalue points in the direction of fastest intensity change
  - The other eigenvector is orthogonal to it
Interpreting the eigenvalues

Classification of image points using eigenvalues of the second moment matrix:

- \( \lambda_1 \) and \( \lambda_2 \) are small
- \( \lambda_1 \) and \( \lambda_2 \) are large, \( \lambda_1 \sim \lambda_2 \)
- \( \lambda_1 \) >> \( \lambda_2 \)
- \( \lambda_2 \) >> \( \lambda_1 \)

Source: Silvio Savarese
Edge

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

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

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

- gradients have small magnitude
- small \( \lambda_1 \), small \( \lambda_2 \)
High-texture region

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

- gradients are different, large magnitudes
- large \( \lambda_1 \), large \( \lambda_2 \)
What are good features to track?

• Can measure “quality” of features from just a single image

• Hence: tracking Harris corners (or equivalent) guarantees small error sensitivity!

→ Implemented in Open CV

Source: Silvio Savarese
Recap

• **Key assumptions** (Errors in Lucas-Kanade)

  • **Small motion**: points do not move very far
  
  • **Brightness constancy**: projection of the same point looks the same in every frame
  
  • **Spatial coherence**: points move like their neighbors
Revisiting the small motion assumption

• Is this motion small enough?
  – Probably not—it’s much larger than one pixel (2\textsuperscript{nd} order terms dominate)
  – How might we solve this problem?
Reduce the resolution!
Multi-resolution Lucas Kanade Algorithm

- Compute ‘simple’ LK at highest level
- At level $i$
  - Take flow $u_{i-1}, v_{i-1}$ from level $i-1$
  - bilinear interpolate it to create $u^*_i, v^*_i$
  - matrices of twice resolution for level $i$
  - multiply $u^*_i, v^*_i$ by 2
  - compute $f_t$ from a block displaced by $u^*_i(x,y), v^*_i(x,y)$
  - Apply LK to get $u'_i(x,y), v'_i(x,y)$ (the correction in flow)
  - Add corrections $u'_i, v'_i$, i.e. $u_i = u^*_i + u'_i$, $v_i = v^*_i + v'_i$. 

Source: Silvio Savarese
Coarse-to-fine optical flow estimation

Gaussian pyramid of image 1

Gaussian pyramid of image 2

$u=10$ pixels

$u=5$ pixels

$u=2.5$ pixels

$u=1.25$ pixels

Source: Silvio Savarese

Fei-Fei Li
Iterative Refinement

- Iterative Lukas-Kanade Algorithm
  1. Estimate velocity at each pixel by solving Lucas-Kanade equations
  2. Warp $I(t-1)$ towards $I(t)$ using the estimated flow field
     - use image warping techniques
  3. Repeat until convergence
Coarse-to-fine optical flow estimation

- Gaussian pyramid of image 1 (t)
- Run iterative L-K
- Warp & upsample
- Gaussian pyramid of image 2 (t+1)
- Run iterative L-K
-...

Source: Silvio Savarese
Optical Flow Results

Lucas-Kanade without pyramids

Fails in areas of large motion
Optical Flow Results

- http://www.ces.clemson.edu/~stb/klt/
- OpenCV
Recap

- **Key assumptions** (Errors in Lucas-Kanade)
  - **Small motion**: points do not move very far
  - **Brightness constancy**: projection of the same point looks the same in every frame
  - **Spatial coherence**: points move like their neighbors
Motion segmentation

• How do we represent the motion in this scene?
Motion segmentation


• Break image sequence into “layers” each of which has a coherent (affine) motion
What are layers?

• Each layer is defined by an alpha mask and an affine motion model

Affine motion

\[ u(x, y) = a_1 + a_2 x + a_3 y \]
\[ v(x, y) = a_4 + a_5 x + a_6 y \]

- Substituting into the brightness constancy equation:

\[ I_x \cdot u + I_y \cdot v + I_t \approx 0 \]
Affine motion

\[ u(x, y) = a_1 + a_2 x + a_3 y \]
\[ v(x, y) = a_4 + a_5 x + a_6 y \]

- Substituting into the brightness constancy equation:

\[
I_x (a_1 + a_2 x + a_3 y) + I_y (a_4 + a_5 x + a_6 y) + I_t \approx 0
\]

- Each pixel provides 1 linear constraint in 6 unknowns

- Least squares minimization:

\[
Err(\tilde{a}) = \sum \left[ I_x (a_1 + a_2 x + a_3 y) + I_y (a_4 + a_5 x + a_6 y) + I_t \right]^2
\]
How do we estimate the layers?

1. Obtain a set of initial affine motion hypotheses
   - Divide the image into blocks and estimate affine motion parameters in each block by least squares
   - Eliminate hypotheses with high residual error

2. Map into motion parameter space
3. Perform k-means clustering on affine motion parameters
   - Merge clusters that are close and retain the largest clusters to obtain a smaller set of hypotheses to describe all the motions in the scene
How do we estimate the layers?

1. Obtain a set of initial affine motion hypotheses
   - Divide the image into blocks and estimate affine motion parameters in each block by least squares
     - Eliminate hypotheses with high residual error
   - Map into motion parameter space
   - Perform k-means clustering on affine motion parameters
     - Merge clusters that are close and retain the largest clusters to obtain a smaller set of hypotheses to describe all the motions in the scene

2. Iterate until convergence:
   - Assign each pixel to best hypothesis
     - Pixels with high residual error remain unassigned
How do we estimate the layers?

• 1. Obtain a set of initial affine motion hypotheses
   – Divide the image into blocks and estimate affine motion parameters in each block by least squares
   • Eliminate hypotheses with high residual error

• Map into motion parameter space
• Perform k-means clustering on affine motion parameters
   – Merge clusters that are close and retain the largest clusters to obtain a smaller set of hypotheses to describe all the motions in the scene
How do we estimate the layers?

1. Obtain a set of initial affine motion hypotheses
   - Divide the image into blocks and estimate affine motion parameters in each block by least squares
     - Eliminate hypotheses with high residual error
   - Map into motion parameter space
   - Perform k-means clustering on affine motion parameters
     - Merge clusters that are close and retain the largest clusters to obtain a smaller set of hypotheses to describe all the motions in the scene

2. Iterate until convergence:
   - Assign each pixel to best hypothesis
     - Pixels with high residual error remain unassigned
   - Perform region filtering to enforce spatial constraints
   - Re-estimate affine motions in each region
Example result


Source: Silvio Savarese
Tracking
What we will learn today?

• Introduction
• Optical flow
• Feature tracking
• Applications
Motion estimation techniques

- Optical flow
  - Recover image motion at each pixel from spatio-temporal image brightness variations (optical flow)

- Feature-tracking
  - Extract visual features (corners, textured areas) and "track" them over multiple frames
    - Shi-Tomasi feature tracker
    - Tracking with dynamics
Feature tracking

• So far, we have only considered optical flow estimation in a pair of images

• If we have more than two images, we can compute the optical flow from each frame to the next

• Given a point in the first image, we can in principle reconstruct its path by simply “following the arrows”
Tracking challenges

• Ambiguity of optical flow
  – Find good features to track

• Large motions
  – Discrete search instead of Lucas-Kanade

• Changes in shape, orientation, color
  – Allow some matching flexibility

• Occlusions, dis-occlusions
  – Need mechanism for deleting, adding new features

• Drift – errors may accumulate over time
  – Need to know when to terminate a track
Shi-Tomasi feature tracker


• Find good features using eigenvalues of second-moment matrix
  – Key idea: “good” features to track are the ones that can be tracked reliably

• From frame to frame, track with Lucas-Kanade and a pure translation model
  – More robust for small displacements, can be estimated from smaller neighborhoods

• 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).

Source: Silvio Savarese
Tracking with dynamics

• Key idea: Given a model of expected motion, predict where objects will occur in next frame, even before seeing the image
  – Restrict search for the object
  – Improved estimates since measurement noise is reduced by trajectory smoothness
Tracking with dynamics

**The Kalman filter:**

- Method for tracking linear dynamical models in Gaussian noise
- The predicted/corrected state distributions are Gaussian
  - Need to maintain the mean and covariance
  - Calculations are easy (all the integrals can be done in closed form)
2D Target tracking using Kalman filter in MATLAB
by AliReza KashaniPour

http://www.mathworks.com/matlabcentral/fileexchange/14243
What we will learn today?

- Introduction
- Optical flow
- Feature tracking
- Applications
Uses of motion

- Tracking features
- Segmenting objects based on motion cues
- Learning dynamical models
- Improving video quality
  - Motion stabilization
  - Super resolution
- Tracking objects
- Recognizing events and activities
Estimating 3D structure
Segmenting objects based on motion cues

- Background subtraction
  - A static camera is observing a scene
  - Goal: separate the static background from the moving foreground
Segmenting objects based on motion cues

- Motion segmentation
  - Segment the video into multiple *coherently* moving objects

Tracking objects

Synthesizing dynamic textures
Super-resolution

Example: A set of low quality images
Super-resolution

Each of these images looks like this:
Super-resolution

The recovery result:

Most of the test data of a couple of exceptions. The low-temperature solder investigated (or some of the manufacturing technologies) on wetting of 40In40Sn, microstructural coarsening, and cycling of 58Bi42Sn.
Recognizing events and activities

Recognizing events and activities

Recognizing events and activities

Crossing – Talking – Queuing – Dancing – jogging

W. Choi & K. Shahid & S. Savarese WMC 2010
W. Choi, K. Shahid, S. Savarese, "What are they doing? : Collective Activity Classification Using Spatio-Temporal Relationship Among People", 9th International Workshop on Visual Surveillance (VSWS09) in conjunction with ICCV 09
Optical flow without motion!
What we have learned today?

• Introduction
• Optical flow
• Feature tracking
• Applications
  • (Problem Set 3 (Q1))