Computer Vision - Lecture 17

Motion and Optical Flow

22.01.2013

Bastian Leibe
RWTH Aachen
http://www.vision.rwth-aachen.de

leibe@vision.rwth-aachen.de

Many slides adapted from K. Grauman, S. Seitz, R. Szeliski, M. Pollefeys, S. Lazebnik
Course Outline

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
- Motion and Tracking
  - Motion and Optical Flow
  - Tracking with Linear Dynamic Models
- Repetition
Recap: Structure from Motion

- Given: $m$ images of $n$ fixed 3D points

$$x_{ij} = P_i X_j, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n$$

- Problem: estimate $m$ projection matrices $P_i$ and $n$ 3D points $X_j$ from the $mn$ correspondences $x_{ij}$

Slide credit: Svetlana Lazebnik
Recap: Structure from Motion Ambiguity

- If we scale the entire scene by some factor $k$ and, at the same time, scale the camera matrices by the factor of $1/k$, the projections of the scene points in the image remain exactly the same.

- More generally: if we transform the scene using a transformation $Q$ and apply the inverse transformation to the camera matrices, then the images do not change.

\[ x = PX = (PQ^{-1})QX \]
Recap: Hierarchy of 3D Transformations

- **Projective**
  - 15dof
  - \[
    \begin{bmatrix}
      A & t \\
      v^T & v
    \end{bmatrix}
  \]
  - Preserves intersection and tangency

- **Affine**
  - 12dof
  - \[
    \begin{bmatrix}
      A & t \\
      0^T & 1
    \end{bmatrix}
  \]
  - Preserves parallelism, volume ratios

- **Similarity**
  - 7dof
  - \[
    \begin{bmatrix}
      sR & t \\
      0^T & 1
    \end{bmatrix}
  \]
  - Preserves angles, ratios of length

- **Euclidean**
  - 6dof
  - \[
    \begin{bmatrix}
      R & t \\
      0^T & 1
    \end{bmatrix}
  \]
  - Preserves angles, lengths

- With no constraints on the camera calibration matrix or on the scene, we get a *projective* reconstruction.
- Need additional information to *upgrade* the reconstruction to affine, similarity, or Euclidean.
Recap: Affine Structure from Motion

- Let’s create a $2m \times n$ data (measurement) matrix:

\[
D = \begin{bmatrix}
\hat{X}_{11} & \hat{X}_{12} & \cdots & \hat{X}_{1n} \\
\hat{X}_{21} & \hat{X}_{22} & \cdots & \hat{X}_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
\hat{X}_{m1} & \hat{X}_{m2} & \cdots & \hat{X}_{mn}
\end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \\ \vdots \\ A_m \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix}
\]

Points (3 × n)

Cameras (2m × 3)

- The measurement matrix $D = MS$ must have rank 3!


Slide credit: Svetlana Lazebnik
Recap: Affine Factorization

- Obtaining a factorization from SVD:

\[ D = U_3 \times 3 W_3 \times 3 V_3^T \]

Possible decomposition:

\[ M = U_3 W_3^{1/2} \quad S = W_3^{1/2} V_3^T \]

This decomposition minimizes \[ |D-MS|^2 \]
Recap: Projective Factorization

\[
D = \begin{bmatrix}
z_{11}x_{11} & z_{12}x_{12} & \cdots & z_{1n}x_{1n} \\
z_{21}x_{21} & z_{22}x_{22} & \cdots & z_{2n}x_{2n} \\
z_{m1}x_{m1} & z_{m2}x_{m2} & \cdots & z_{mn}x_{mn}
\end{bmatrix} = \begin{bmatrix}
P_1 \\
P_2 \\
\vdots \\
P_m
\end{bmatrix} \begin{bmatrix}
x_1 \\
x_2 \\
\cdots \\
x_n
\end{bmatrix}
\]

Points (4 \times n)
Cameras (3m \times 4)

\[
D = MS \text{ has rank 4}
\]

- If we knew the depths \( z \), we could factorize \( D \) to estimate \( M \) and \( S \).
- If we knew \( M \) and \( S \), we could solve for \( z \).
- Solution: iterative approach (alternate between above two steps).
Recap: Sequential Projective SfM

- Initialize motion from two images using fundamental matrix
- Initialize structure
- For each additional view:
  - Determine projection matrix of new camera using all the known 3D points that are visible in its image - *calibration*
  - Refine and extend structure: compute new 3D points, re-optimize existing points that are also seen by this camera - *triangulation*
- Refine structure and motion: *bundle adjustment*

Slide credit: Svetlana Lazebnik
Recap: Bundle Adjustment

- Non-linear method for refining structure and motion
- Minimizing mean-square reprojection error

\[ E(P, X) = \sum_{i=1}^{m} \sum_{j=1}^{n} D(x_{ij}, P_i X_j)^2 \]
Practical Considerations

1. Role of the baseline
   - Small baseline: large depth error
   - Large baseline: difficult search problem

• Solution
   - Track features between frames until baseline is sufficient.
Some Commercial Software Packages

- boujou
  (http://www.2d3.com/)
- PFTrack
  (http://www.thepixelfarm.co.uk/)
- MatchMover
  (http://www.realviz.com/)
- SynthEyes
  (http://www.ssontech.com/)
- Icarus
  (http://aig.cs.man.ac.uk/research/reveal/icarus/)
- Voodoo Camera Tracker
  (http://www.digilab.uni-hannover.de/)
Applications: Large-Scale SfM from Flickr

Topics of This Lecture

• **Introduction to Motion**
  - Applications, uses

• **Motion Field**
  - Derivation

• **Optical Flow**
  - Brightness constancy constraint
  - Aperture problem
  - Lucas-Kanade flow
  - Iterative refinement
  - Global parametric motion
  - Coarse-to-fine estimation
  - Motion segmentation

• **KLT Feature Tracking**
Video

- 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 and Perceptual Organization

- Sometimes, motion is the only cue...

Slide credit: Svetlana Lazebnik
Motion and Perceptual Organization

- Sometimes, motion is foremost cue
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
  - Directly from optic flow
  - Indirectly to create correspondences for SfM
- Segmenting objects based on motion cues
- Learning dynamical models
- Recognizing events and activities
- Improving video quality (motion stabilization)
Motion Estimation Techniques

- **Direct methods**
  - Directly recover image motion at each pixel from spatio-temporal image brightness variations
  - Dense motion fields, but sensitive to appearance variations
  - Suitable for video and when image motion is small

- **Feature-based methods**
  - Extract visual features (corners, textured areas) and track them over multiple frames
  - Sparse motion fields, but more robust tracking
  - Suitable when image motion is large (10s of pixels)
Topics of This Lecture

• Introduction to Motion
  ➢ Applications, uses

• **Motion Field**
  ➢ Derivation

• Optical Flow
  ➢ Brightness constancy constraint
  ➢ Aperture problem
  ➢ Lucas-Kanade flow
  ➢ Iterative refinement
  ➢ Global parametric motion
  ➢ Coarse-to-fine estimation
  ➢ Motion segmentation

• KLT Feature Tracking
Motion Field

- The motion field is the projection of the 3D scene motion into the image
Motion Field and Parallax

- \( P(t) \) is a moving 3D point
- Velocity of scene point: \( V = \frac{dP}{dt} \)
- \( p(t) = (x(t), y(t)) \) is the projection of \( P \) in the image.
- Apparent velocity \( v \) in the image: given by components \( v_x = \frac{dx}{dt} \) and \( v_y = \frac{dy}{dt} \)
- These components are known as the motion field of the image.
Motion Field and Parallax

\[ \mathbf{V} = (V_x, V_y, V_Z) \]  \[ p = f \frac{P}{Z} \]  \[ P(t) \]

To find image velocity \( \mathbf{v} \), differentiate \( p \) with respect to \( t \) (using quotient rule):

\[ \mathbf{v} = f \frac{Z \mathbf{V} - \mathbf{V}_z \mathbf{P}}{Z^2} \]

\[ \mathbf{v}_x = \frac{f V_x - V_z x}{Z} \]  \[ \mathbf{v}_y = \frac{f V_y - V_z y}{Z} \]

- Image motion is a function of both the 3D motion (\( \mathbf{V} \)) and the depth of the 3D point (\( Z \)).

Slide credit: Svetlana Lazebnik
Motion Field and Parallax

- Pure translation: $V$ is constant everywhere

\[
\begin{align*}
    v_x &= \frac{fV_x - V_z x}{Z} \\
    v_y &= \frac{fV_y - V_z y}{Z}
\end{align*}
\]

\[
v = \frac{1}{Z} (v_0 - V_z p),
\]

\[
v_0 = (fV_x, fV_y)
\]
Motion Field and Parallax

- **Pure translation:** $V$ is constant everywhere
  \[
  v = \frac{1}{Z} (v_0 - V_z p),
  \]
  \[
  v_0 = (fV_x, fV_y)
  \]

- **$V_z$ is nonzero:**
  - Every motion vector points toward (or away from) $v_0$, the vanishing point of the translation direction.
Motion Field and Parallax

- Pure translation: \( V \) is constant everywhere
  \[
  \mathbf{v} = \frac{1}{Z} (\mathbf{v}_0 - V_z \mathbf{p}),
  \]
  \[
  \mathbf{v}_0 = (fV_x, fV_y)
  \]

- \( V_z \) is nonzero:
  - Every motion vector points toward (or away from) \( \mathbf{v}_0 \), the vanishing point of the translation direction.

- \( V_z \) is zero:
  - Motion is parallel to the image plane, all the motion vectors are parallel.

- The length of the motion vectors is inversely proportional to the depth \( Z \).
Topics of This Lecture

• Introduction to Motion
  ➢ Applications, uses

• Motion Field
  ➢ Derivation

• Optical Flow
  ➢ Brightness constancy constraint
  ➢ Aperture problem
  ➢ Lucas-Kanade flow
  ➢ Iterative refinement
  ➢ Global parametric motion
  ➢ Coarse-to-fine estimation
  ➢ Motion segmentation

• KLT Feature Tracking
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.
Apparent Motion ≠ Motion Field

Figure 12-2. The optical flow is not always equal to the motion field. In (a) a smooth sphere is rotating under constant illumination—the image does not change, yet the motion field is nonzero. In (b) a fixed sphere is illuminated by a moving source—the shading in the image changes, yet the motion field is zero.
Estimating Optical Flow

- Given two subsequent frames, estimate the apparent motion field $u(x,y)$ and $v(x,y)$ 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.
The Brightness Constancy Constraint

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

- Linearizing the right hand side using Taylor expansion:
  \[ I(x, y, t-1) \approx I(x, y, t) + I_x \cdot u(x, y) + I_y \cdot v(x, y) \]

- Hence, \( I_x \cdot u + I_y \cdot v + I_t \approx 0 \)
The Brightness Constancy Constraint

\[ I_x \cdot u + I_y \cdot v + I_t = 0 \]

- How many equations and unknowns per pixel?
  - One equation, two unknowns

- Intuitively, what does this constraint mean?
  \[ \nabla I \cdot (u, v) + I_t = 0 \]

- The component of the flow perpendicular to the gradient (i.e., parallel to the edge) is unknown

If \((u, v)\) satisfies the equation, so does \((u + u', v + v')\) if \(\nabla I \cdot (u', v') = 0\)
The Aperture Problem

Perceived motion
The Aperture Problem

Actual motion
The Barber Pole Illusion

http://en.wikipedia.org/wiki/Barberpole_illusion
The Barber Pole Illusion

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

Slide credit: Svetlana Lazebnik
The Barber Pole Illusion

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

Slide credit: Svetlana Lazebnik
Solving the Aperture Problem

- How to get more equations for a pixel?
- **Spatial coherence constraint**: pretend 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 Aperture Problem

• Least squares problem:

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

\[A \quad d = b\]

25x2 2x1 25x1

• Minimum least squares solution given by solution of

\[
(A^T A) d = A^T b
\]

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

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

(The summations are over all pixels in the K 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.
  - \(A^T A\) entries should not be too small (noise).
  - \(A^T A\) should be well-conditioned.
Eigenvectors of $A^T A$

$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y] = \sum \nabla I (\nabla I)^T$$

- Haven’t we seen an equation like this before?
- Recall the Harris corner detector: $M = A^T A$ is the second moment matrix.
- The eigenvectors and eigenvalues of $M$ 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:

\[ \lambda_1 \text{ and } \lambda_2 \text{ are small} \]

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

\[ \lambda_1 >> \lambda_2 \]

“Corner”

“Edge”

“Flat” region

Slide credit: Kristen Grauman
Edge

$$\sum \nabla I (\nabla I)^T$$

- Gradients very large or very small
- Large $\lambda_1$, small $\lambda_2$

Slide credit: Svetlana Lazebnik
Low-Texture Region

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

- Gradients have small magnitude
- Small \( \lambda_1 \), small \( \lambda_2 \)

Slide credit: Svetlana Lazebnik
High-Texture Region

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

- Gradients are different, large magnitude
- Large \( \lambda_1 \), large \( \lambda_2 \)

Slide credit: Svetlana Lazebnik
Per-Pixel Estimation Procedure

- Let \( M = \sum (\nabla I)(\nabla I)^T \) and \( b = \begin{bmatrix} -\sum I_x I_t \\ -\sum I_y I_t \end{bmatrix} \)

- Algorithm: At each pixel compute \( U \) by solving \( MU = b \)

- \( M \) is singular if all gradient vectors point in the same direction
  - E.g., along an edge
  - Trivially singular if the summation is over a single pixel or if there is no texture
  - I.e., only normal flow is available (aperture problem)

- Corners and textured areas are OK
Iterative Refinement

1. Estimate velocity at each pixel using one iteration of Lucas and Kanade estimation.

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

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

2. Warp one image toward the other using the estimated flow field.

-Easier said than done-

3. Refine estimate by repeating the process.
Optical Flow: Iterative Refinement

Initial guess: $d_0 = 0$

Estimate: $d_1 = d_0 + \hat{d}$

(using $d$ for displacement here instead of $u$)

Slide credit: Steve Seitz
Optical Flow: Iterative Refinement

Initial guess: $d_1$

Estimate: $d_2 = d_1 + \hat{d}$

(Using $d$ for displacement here instead of $u$)

Slide credit: Steve Seitz
Optical Flow: Iterative Refinement

Initial guess: \( d_2 \)
Estimate: \( d_3 = d_2 + \hat{d} \)

(\textit{using} \( d \) \textit{for} displacement \textit{here instead of} \( u \))
Optical Flow: Iterative Refinement

\[ f_1(x - d_3) \approx f_2(x) \]

(using \(d\) for \textit{displacement} here instead of \(u\))
Optic Flow: Iterative Refinement

• Some Implementation Issues:
  - Warping is not easy (ensure that errors in warping are smaller than the estimate refinement).
  - Warp one image, take derivatives of the other so you don’t need to re-compute the gradient after each iteration.
  - Often useful to low-pass filter the images before motion estimation (for better derivative estimation, and linear approximations to image intensity).
Extension: Global Parametric Motion Models

- Translation: 2 unknowns
- Affine: 6 unknowns
- Perspective: 8 unknowns
- 3D rotation: 3 unknowns

Slide credit: Steve Seitz
Example: 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:

\[
I_x \cdot u + I_y \cdot v + I_t \approx 0
\]
Example: 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:

\[ Err(\vec{a}) = \sum \left[ I_x (a_1 + a_2 x + a_3 y) + I_y (a_4 + a_5 x + a_6 y) + I_t \right]^2 \]
Problem Cases in Lucas-Kanade

- The motion is large (larger than a pixel)
  - Iterative refinement, coarse-to-fine estimation
- A point does not move like its neighbors
  - Motion segmentation
- Brightness constancy does not hold
  - Do exhaustive neighborhood search with normalized correlation.
Dealing with Large Motions

Slide credit: Svetlana Lazebnik
Temporal Aliasing

- Temporal aliasing causes ambiguities in optical flow because images can have many pixels with the same intensity.
- I.e., how do we know which ‘correspondence’ is correct?

- To overcome aliasing: coarse-to-fine estimation.
Idea: Reduce the Resolution!
Coarse-to-fine Optical Flow Estimation

Image 1

Gaussian pyramid of image 1

$u=10$ pixels

$u=5$ pixels

$u=2.5$ pixels

$u=1.25$ pixels

Image 2

Gaussian pyramid of image 2

Slide credit: Steve Seitz
Coarse-to-fine Optical Flow Estimation

Image 1

Gaussian pyramid of image 1

Run iterative L-K

Warp & upsample

Run iterative L-K

Image 2

Gaussian pyramid of image 2

Slide credit: Steve Seitz
Dense Optical Flow

- Dense measurements can be obtained by adding smoothness constraints.

T. Brox, C. Bregler, J. Malik, Large displacement optical flow, CVPR‘09, Miami, USA, June 2009.
Summary

- **Motion field**: 3D motions projected to 2D images; dependency on depth.

- **Solving for motion with**
  - Sparse feature matches
  - Dense optical flow

- **Optical flow**
  - Brightness constancy assumption
  - Aperture problem
  - Solution with spatial coherence assumption
References and Further Reading

• Here is the original paper by Lucas & Kanade

• And the original paper by Shi & Tomasi

• Read the story how optical flow was used for special effects in a number of recent movies
  - [http://www.fxguide.com/article333.html](http://www.fxguide.com/article333.html)