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
  \[ x_i = P X_i, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n \]
- Problem: estimate \( m \) projection matrices \( P \), and \( n \) 3D points \( X \) from the \( mn \) correspondences \( x_{ij} \)

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

\[
x = P X = (P Q^{-1}) Q X
\]
Recap: Affine Structure from Motion
- Let’s create a $2m \times n$ data (measurement) matrix:

$$D = \begin{bmatrix}
\hat{x}_{11} & \hat{x}_{12} & \cdots & \hat{x}_{1n} \\
\hat{x}_{21} & \hat{x}_{22} & \cdots & \hat{x}_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
\hat{x}_{m1} & \hat{x}_{m2} & \cdots & \hat{x}_{mn}
\end{bmatrix}$$

Points $(3 \times n)$

Cameras $(2m \times 3)$

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


Slide credit: Svetlana Lazebnik

Recap: Affine Factorization
- Obtaining a factorization from SVD:

$$D = U_3 W_{22} S_3 W_{22}^T V_3^T$$

Possible decomposition:

$$M = U_3 W_{22}$$

S = $W_{22}^T V_3$

This decomposition minimizes $|| D - MS ||^2$

Slide credit: Martial Hebert

Projective Structure from Motion
- Given: $m$ images of $n$ fixed 3D points

$$z_i X_0 = P_i X_0, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n$$

- Problem: estimate $m$ projection matrices $P_i$ and $n$ 3D points $X_0$ from the $mn$ correspondences $x_{ij}$

- With no calibration info, cameras and points can only be recovered up to a $4 \times 4$ projective transformation $Q$:

$$X \rightarrow QX, \quad P \rightarrow PQ^{-1}$$

- We can solve for structure and motion when

$$2mn = \frac{11m + 3n - 15}{2}$$

- For two cameras, at least 7 points are needed.

Slide credit: Svetlana Lazebnik

Projective Factorization
- Known points $X_0$ that are also seen by this camera –

$$D = P_1 \cdots P_n \begin{bmatrix}
z_1 X_{11} & z_1 X_{12} & \cdots & z_1 X_{1n} \\
z_2 X_{21} & z_2 X_{22} & \cdots & z_2 X_{2n} \\
\vdots & \vdots & \ddots & \vdots \\
z_m X_{m1} & z_m X_{m2} & \cdots & z_m X_{mn}
\end{bmatrix}$$

Points $(4 \times n)$

Cameras $(3m \times 4)$

- 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).

Slide credit: Svetlana Lazebnik

Sequential Structure from Motion
- Initialize motion from two images using fundamental matrix

Slide credit: Svetlana Lazebnik

Sequential Structure from Motion
- 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

Slide credit: Svetlana Lazebnik

Slide credit: Martial Hebert
Sequential Structure from Motion

- 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

Bundle Adjustment

- Non-linear method for refining structure and motion
- Minimizing mean-square reprojection error
  \[ E(P, X) = \sum_{i=1}^{n} D(x_i, PX_i) \]

Practical Considerations (1)

1. Role of the baseline
   - Small baseline: large depth error
   - Large baseline: difficult search problem
2. Solution
   - Track features between frames until baseline is sufficient.

Practical Considerations (2)

2. There will still be many outliers
   - Incorrect feature matches
   - Moving objects
   ⇒ Apply RANSAC to get robust estimates based on the inlier points.
3. Estimation quality depends on the point configuration
   - Points that are close together in the image produce less stable solutions.
   ⇒ Subdivide image into a grid and try to extract about the same number of features per grid cell.

Some Commercial Software Packages

- boujou (http://www.2d3.com/)
- PFTTrack (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: Matchmoving

- Putting virtual objects into real-world videos
  
  **Original sequence**  |  **Tracked features**  |  **SfM results**  |  **Final video**

[Image: Matchmoving example]

Video

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

[Image: Video sequence]

Applications of Segmentation to Video

- Background subtraction
  - A static camera is observing a scene.
  - Goal: separate the static background from the moving foreground.

How to come up with background frame estimate without access to "empty" scene?

[Image: Background subtraction example]

Video

- Background subtraction
- Shot boundary detection
  - Commercial video is usually composed of shots or sequences showing the same objects or scene.
  - Goal: segment video into shots for summarization and browsing (each shot can be represented by a single keyframe in a user interface).
  - Difference from background subtraction: the camera is not necessarily stationary.
Applications of Segmentation to Video

- Background subtraction
- Shot boundary detection
  - For each frame, compute the distance between the current frame and the previous one:
    - Pixel-by-pixel differences
    - Differences of color histograms
    - Block comparison
  - If the distance is greater than some threshold, classify the frame as a shot boundary.

Motion and Perceptual Organization
- Sometimes, motion is the only cue...
  - Not grouped
  - Proximity
  - Similarity
  - Symmetry
  - Common Scene
  - Common Region
  - Continuity
  - Closure

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

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

\[
\begin{align*}
  v_x &= \frac{fV_x - V_y}{Z} \\
  v_y &= \frac{fV_y - V_x}{Z} \\
  v &= \sqrt{v_x^2 + v_y^2}
\end{align*}
\]

- Image motion is a function of both the 3D motion \( V \) and the depth of the 3D point \( Z \).
Motion Field and Parallax

• Pure translation: \( \mathbf{V} \) is constant everywhere

\[
\begin{align*}
    v_x &= \frac{fV_x - V_p}{Z} \\
    v_y &= \frac{fV_y - V_p}{Z} \\
    v_z &= (fV_x, fV_y)
\end{align*}
\]

• \( v_z \) is nonzero:
  - Every motion vector points toward (or away from) \( v_0 \), 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 \( Z \).

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.
**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**

$I_x \cdot u + I_y \cdot v + I_t = 0$

- How many equations and unknowns per pixel?
  - One equation, two unknowns
- Intuitively, what does this constraint mean?
  - 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**

**The Barber Pole Illusion**

[Link to Wikipedia article: Barberpole illusion](http://en.wikipedia.org/wiki/Barberpole_illusion)
The Barber Pole Illusion

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

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

Conditions for Solvability

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

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

- When is this solvable?
  - \(A^TA\) should be invertible.
  - \(A^TA\) entries should not be too small (noise).
  - \(A^TA\) should be well-conditioned.

Eigenvectors of \(A^TA\)

- Haven’t we seen an equation like this before?
- Recall the Harris corner detector: \(M = A^TA\) 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:

\[ \lambda_2 \text{ and } \lambda_3 \text{ are small} \]

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

\[ \lambda_1 \gg \lambda_2 \]

"Corner"

"Edge"

"Flat" region

Slide credit: Kristen Grauman

Edge

- Gradients very large or very small
- Large \( \lambda_1 \), small \( \lambda_2 \)

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

Slide credit: SvetaLana Lazebnik

Low-Texture Region

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

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

Slide credit: SvetaLana Lazebnik

High-Texture Region

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

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

Slide credit: SvetaLana Lazebnik

Per-Pixel Estimation Procedure

- Let \( M = \sum (\nabla I)(\nabla I)^T \) and \( b = -\sum I \cdot I \cdot I \).
- Algorithm: At each pixel compute \( U \) by solving \( MU = b \).
- If 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
  - I.e., only normal flow is available (aperture problem)
- Corners and textured areas are OK

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

Slide credit: Steve Seitz

Iterative Refinement

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

\[ \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_y I_x & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_1 \\ \sum I_y I_2 \end{bmatrix} \]

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

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

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

**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
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:
  \[ I_x \cdot u + I_y \cdot v + I_z = 0 \]

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

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.
Coarse-to-fine Optical Flow Estimation

- Gaussian pyramid of image 1
- Gaussian pyramid of image 2

- Image 2
- Image 1

- u=10 pixels
- u=5 pixels
- u=2.5 pixels
- u=1.25 pixels

Slide credit: Steve Seitz

Dense Optical Flow

- Dense measurements can be obtained by adding smoothness constraints.

- Color map

(c) Thomas Brox 2009

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