Computer Vision - Lecture 20

Motion and Optical Flow

25.01.2017

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

---

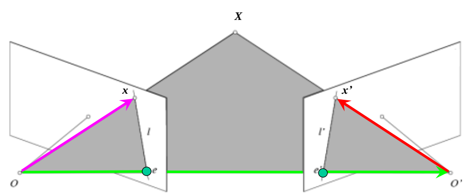
Announcements

- Lecture Evaluation
  - Please fill out the evaluation form...

Course Outline

- Image Processing Basics
- Segmentation & Grouping
- Object Recognition
- Local Features & Matching
- Object Categorization
- 3D Reconstruction
  - Epipolar Geometry and Stereo Basics
  - Camera calibration & Uncalibrated Reconstruction
  - Active Stereo
- Motion
  - Motion and Optical Flow
- 3D Reconstruction (Reprise)
  - Structure-from-Motion

Recap: Epipolar Geometry - Calibrated Case

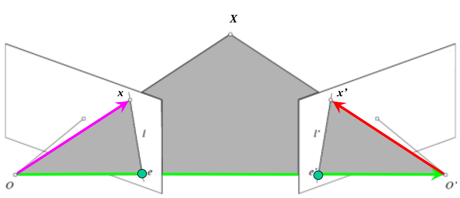


$$x \cdot [t \times (R x')] = 0 \quad \Rightarrow \quad x^T E x' = 0 \quad \text{with} \quad E = [t_\times]R$$

Essential Matrix (Longuet-Higgins, 1981)

Slide credit: Svetlana Lazebnik

Recap: Epipolar Geometry - Uncalibrated Case



$$\hat{x}^T E \hat{x}' = 0 \quad \Rightarrow \quad x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K'^{-1}$$

$$x = K \hat{x}$$

$$x' = K' \hat{x}'$$

Fundamental Matrix (Faugeras and Luong, 1992)

Slide credit: Svetlana Lazebnik

Recap: The Eight-Point Algorithm

$x = (u, v, 1)^T, \quad x' = (u', v', 1)^T$

$$(u, v, 1)\begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix}\begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = 0 \quad \Rightarrow \quad [u'u, u'v, u', uv', vv', v', u, v, 1]\begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = 0$$

$$\begin{bmatrix} u'_1u_1 & u'_1v_1 & u'_1 & u_1v'_1 & v_1v'_1 & v'_1 & u_1 & v_1 & 1 \\ u'_2u_2 & u'_2v_2 & u'_2 & u_2v'_2 & v_2v'_2 & v'_2 & u_2 & v_2 & 1 \\ u'_3u_3 & u'_3v_3 & u'_3 & u_3v'_3 & v_3v'_3 & v'_3 & u_3 & v_3 & 1 \\ u'_4u_4 & u'_4v_4 & u'_4 & u_4v'_4 & v_4v'_4 & v'_4 & u_4 & v_4 & 1 \\ u'_5u_5 & u'_5v_5 & u'_5 & u_5v'_5 & v_5v'_5 & v'_5 & u_5 & v_5 & 1 \\ u'_6u_6 & u'_6v_6 & u'_6 & u_6v'_6 & v_6v'_6 & v'_6 & u_6 & v_6 & 1 \\ u'_7u_7 & u'_7v_7 & u'_7 & u_7v'_7 & v_7v'_7 & v'_7 & u_7 & v_7 & 1 \\ u'_8u_8 & u'_8v_8 & u'_8 & u_8v'_8 & v_8v'_8 & v'_8 & u_8 & v_8 & 1 \end{bmatrix}\begin{bmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\mathbf{Af} = \mathbf{0}$$

Solve using... SVD!

This minimizes:

$$\sum_{i=1}^{N} (x_i^T F x_i')^2$$

Slide adapted from Svetlana Lazebnik

Recap: Normalized Eight-Point Algorithm

1. Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels.
2. Use the eight-point algorithm to compute F from the normalized points.
3. Enforce the rank-2 constraint using SVD.

$$F \overset{\text{SVD}}{=} UDV^T = U \begin{bmatrix} d_{11} & & \\ & d_{22} & \\ & & d_{33} \end{bmatrix} \begin{bmatrix} v_{11} & \cdots & v_{13} \\ \vdots & \ddots & \vdots \\ v_{31} & \cdots & v_{33} \end{bmatrix}^T$$

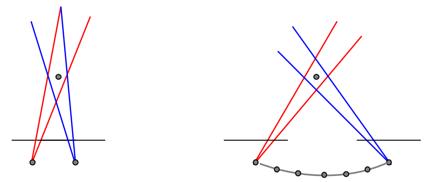
Set d_{33} to zero and reconstruct F

4. Transform fundamental matrix back to original units: if T and T' are the normalizing transformations in the two images, than the fundamental matrix in original coordinates is $T^T F T'$.

Slide credit: Svetlana Lazebnik B. Leibe [Hartley, 1995]

Computer Vision WS 16/17

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.

Slide adapted from Steve Seitz B. Leibe

Computer Vision WS 15/16

Topics of This Lecture

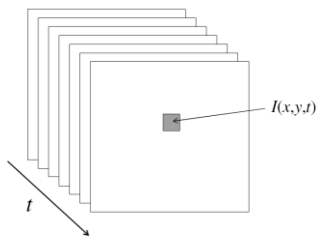
- Introduction to Motion
   - Applications, uses
- Motion Field
   - Derivation
- Optical Flow
   - Brightness constancy constraint
   - Aperture problem
   - Lucas-Kanade flow
   - Iterative refinement
   - Global parametric motion
   - Coarse-to-fine estimation
   - Motion segmentation
- KLT Feature Tracking

B. Leibe

Computer Vision WS 15/16

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)

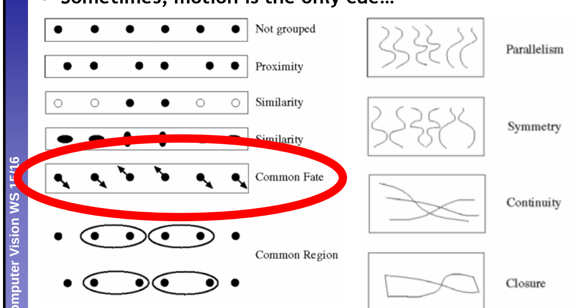


Slide credit: Svetlana Lazebnik B. Leibe

Computer Vision WS 15/16

Motion and Perceptual Organization

- Sometimes, motion is the only cue…



Slide credit: Svetlana Lazebnik B. Leibe

Computer Vision WS 15/16

Motion and Perceptual Organization

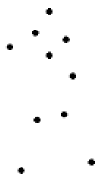
- Sometimes, motion is foremost cue

Slide credit: Kristen Grauman B. Leibe

Computer Vision WS 15/16

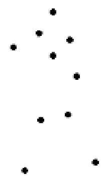
Motion and Perceptual Organization

- Even "impoverished" motion data can evoke a strong percept



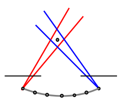
Motion and Perceptual Organization

- 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

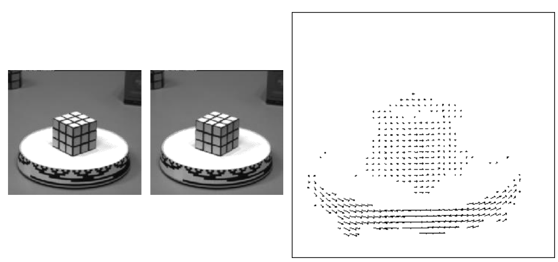
- Direct methods
  - Directly recover image motion at each pixel from spatio-temporal image brightness variations
  - Dense motion fields, but sensitive to appearance variations
  - Suitable for video and when image motion is small
- Feature-based methods
  - Extract visual features (corners, textured 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

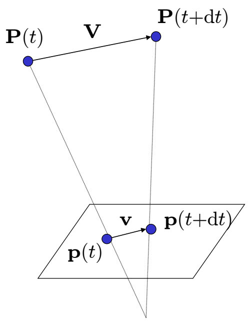
Motion Field

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



Motion Field and Parallax

- $\mathbf{P}(t)$ is a moving 3D point
- Velocity of 3D scene point: $\mathbf{V} = \mathrm{d}\mathbf{P}/\mathrm{d}t$
- $\mathbf{p}(t) = (x(t), y(t))$ is the projection of $\mathbf{P}$ in the image.
- Apparent velocity $\mathbf{v}$ in the image: given by components $v_x = \mathrm{d}x/\mathrm{d}t$ and $v_y = \mathrm{d}y/\mathrm{d}t$
- These components are known as the *motion field* of the image.



Motion Field and Parallax

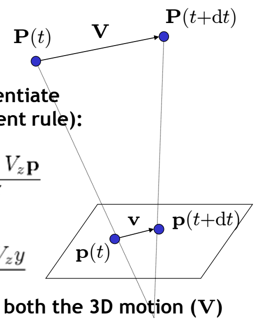
Quotient rule: $(f/g)' = (g\,f' - g'f)/g^2$

$$\mathbf{V} = [V_x, V_y, V_z] \qquad \mathbf{p} = f\frac{\mathbf{P}}{Z}$$

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

$$\mathbf{v} = f\frac{Z\mathbf{V} - V_z\mathbf{P}}{Z^2} = \frac{f\mathbf{V} - V_z\mathbf{p}}{Z}$$

$$v_x = \frac{fV_x - V_z x}{Z} \qquad v_y = \frac{fV_y - V_z y}{Z}$$



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

Motion Field and Parallax

- Pure translation: $\mathbf{V}$ is constant everywhere

$$v_x = \frac{fV_x - V_z x}{Z} \qquad \mathbf{v} = \frac{1}{Z}(\mathbf{v}_0 - V_z\mathbf{p}),$$

$$v_y = \frac{fV_y - V_z y}{Z} \qquad \mathbf{v}_0 = (fV_x, fV_y)$$

Motion Field and Parallax

- Pure translation: $\mathbf{V}$ 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.

Motion Field and Parallax

- Pure translation: $\mathbf{V}$ 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.

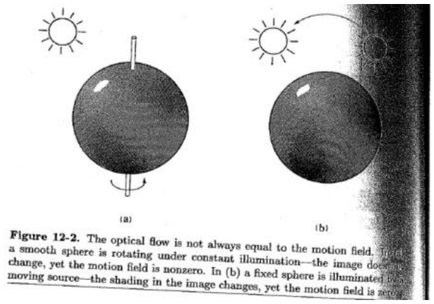
Topics of This Lecture

- 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

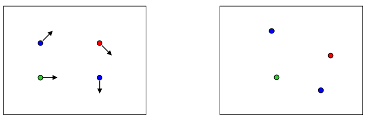
Optical Flow

- Definition
  - Optical flow is the *apparent* motion of brightness patterns in the image.
- Important difference
  - Ideally, optical flow would be the same as the motion field.
  - But we 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



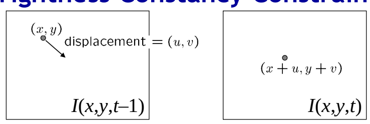
Estimating Optical Flow



$I(x,y,t-1)$ $I(x,y,t)$

- 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, $I_x \cdot u + I_y \cdot v + I_t \approx 0$

Spatial derivatives Temporal derivative

The Brightness Constancy Constraint

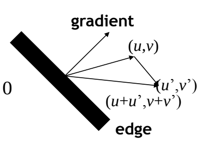
$$I_x \cdot u + I_y \cdot v + I_t = 0$$

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

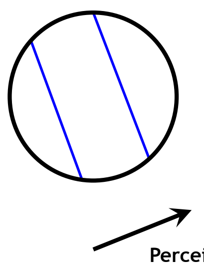
$$\nabla I \cdot (u,v) + I_t = 0$$

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

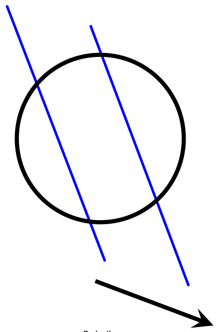
If (u,v) satisfies the equation, so does $(u+u', v+v')$ if $\nabla I \cdot (u',v') = 0$



The Aperture Problem



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

The Barber Pole Illusion

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

$$0 = I_t(\mathbf{p_i}) + \nabla I(\mathbf{p_i}) \cdot [u \; v]$$

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

B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 674-679, 1981.

Solving the Aperture Problem

- Least squares problem:

$$\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix} \qquad \underset{25\text{x}2}{A} \; \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

- Minimum least squares solution given by solution of

$$\underset{2\text{x}2}{(A^T A)} \; \underset{2\text{x}1}{d} = \underset{2\text{x}1}{A^T b}$$

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

(The summations are over all pixels in the $K \times K$ window)

Conditions for Solvability

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

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

$$A^T A \qquad\qquad\qquad A^T b$$

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

Computer Vision WS 15/16 | Slide credit: Svetlana Lazebnik | B. Leibe | 45

Eigenvectors of A^TA

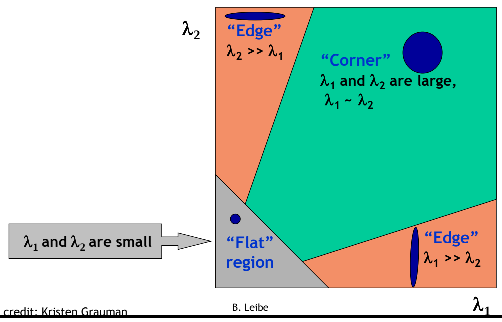
$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \; I_y] = \sum \nabla I (\nabla I)^T$$

- Haven't we seen an equation like this before?
- Recall the Harris corner detector
  - $M = A^TA$ 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.

Computer Vision WS 15/16 | Slide credit: Svetlana Lazebnik | B. Leibe | 46

Interpreting the Eigenvalues

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



Computer Vision WS 15/16 | Slide credit: Kristen Grauman | B. Leibe | 47

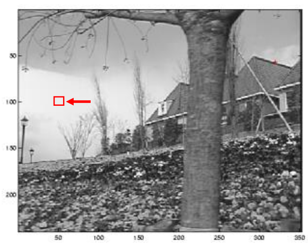
Edge

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

- Gradients very large or very small
- Large λ_1, small λ_2

Computer Vision WS 15/16 | Slide credit: Svetlana Lazebnik | B. Leibe | 48

Low-Texture Region

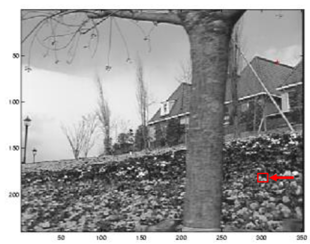


$\sum \nabla I (\nabla I)^T$

- Gradients have small magnitude
- Small λ_1, small λ_2

Computer Vision WS 15/16 | Slide credit: Svetlana Lazebnik | B. Leibe | 49

High-Texture Region



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

- Gradients are different, large magnitude
- Large λ_1, large λ_2

Computer Vision WS 15/16 | Slide credit: Svetlana Lazebnik | B. Leibe | 50

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

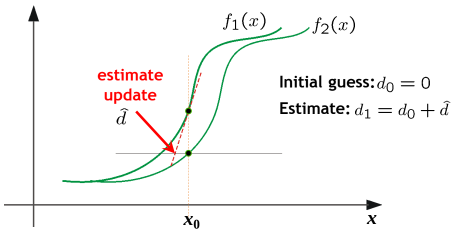
Iterative Refinement

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

$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

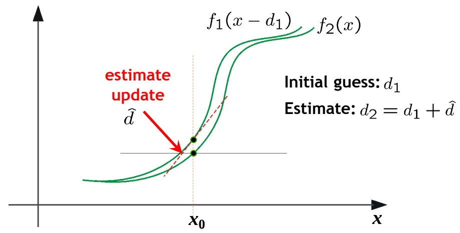
2. Warp one image toward the other using the estimated flow field.
   - *(Easier said than done)*
3. Refine estimate by repeating the process.

Optical Flow: Iterative Refinement



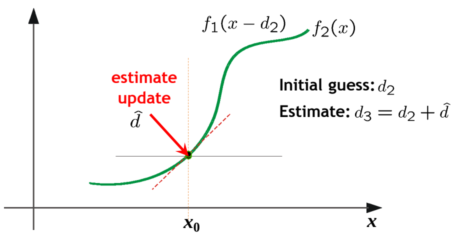
(using *d* for *displacement* here instead of *u*)

Optical Flow: Iterative Refinement



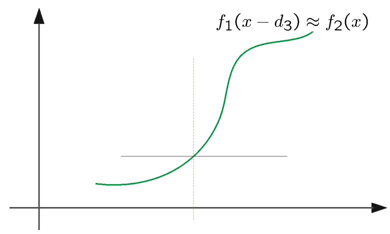
(using *d* for *displacement* here instead of *u*)

Optical Flow: Iterative Refinement



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

Optical Flow: Iterative Refinement



(using *d* for *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).

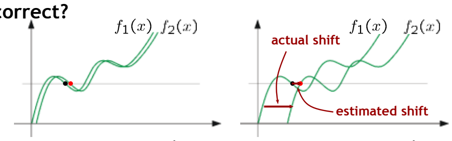
Problem Cases in Lucas-Kanade

- 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

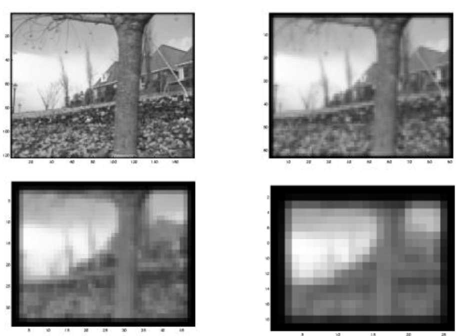
- Temporal aliasing causes ambiguities in optical flow because images can have many pixels with the same intensity.
- I.e., how do we know which 'correspondence' is correct?



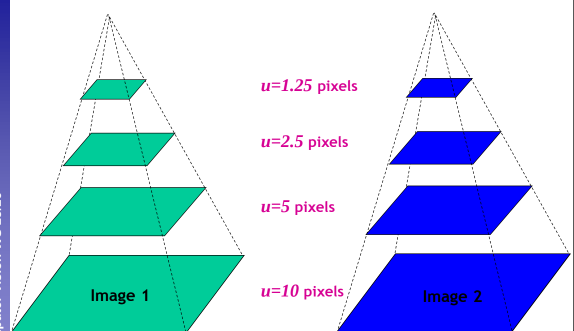
Nearest match is correct (no aliasing) *Nearest match is incorrect (aliasing)*

- To overcome aliasing: coarse-to-fine estimation.

Idea: Reduce the Resolution!

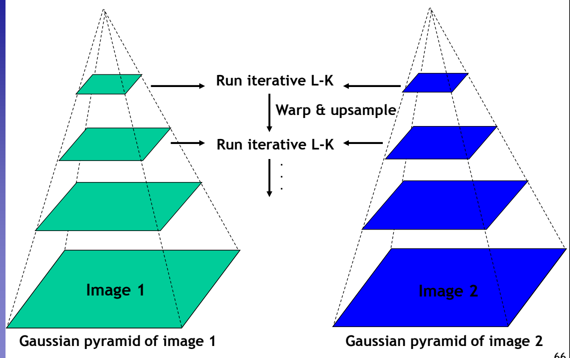


Coarse-to-fine Optical Flow Estimation



Gaussian pyramid of image 1 Gaussian pyramid of image 2

Coarse-to-fine Optical Flow Estimation



Slide credit: Steve Seitz

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**

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".

Slide credit: Svetlana Lazebnik

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, disocclusions
  - Need mechanism for deleting, adding new features
- Drift - errors may accumulate over time
  - Need to know when to terminate a track

Slide credit: Svetlana Lazebnik

Handling Large Displacements

- Define a small area around a pixel as the template.
- Match the template against each pixel within a search area in next image - just like stereo matching!
- Use a match measure such as SSD or correlation.
- After finding the best discrete location, can use Lucas-Kanade to get sub-pixel estimate.

Slide credit: Svetlana Lazebnik

Tracking Over Many Frames

- Select features in first frame
- For each frame:
  - Update positions of tracked features
    - Discrete search or Lucas-Kanade
  - Terminate inconsistent tracks
    - Compute similarity with corresponding feature in the previous frame or in the first frame where it's visible
  - Start new tracks if needed
    - Typically every ~10 frames, new features are added to "refill the ranks".

Slide credit: Svetlana Lazebnik

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.

J. Shi and C. Tomasi. Good Features to Track. CVPR 1994.

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).

J. Shi and C. Tomasi. Good Features to Track. CVPR 1994.

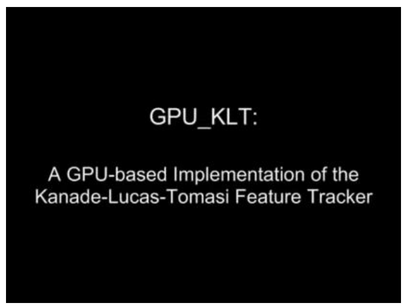
Real-Time GPU Implementations

- This basic feature tracking framework (Lucas-Kanade + Shi-Tomasi) is commonly referred to as “KLT tracking”.
  - Used as preprocessing step for many applications (recall the Structure-from-Motion pipeline)
  - Lends itself to easy parallelization
- Very fast GPU implementations available
  - C. Zach, D. Gallup, J.-M. Frahm, Fast Gain-Adaptive KLT tracking on the GPU. In CVGPU'08 Workshop, Anchorage, USA, 2008
  - 216 fps with automatic gain adaptation
  - 260 fps without gain adaptation

http://www.cs.unc.edu/~ssinha/Research/GPU_KLT/

http://cs.unc.edu/~cmzach/opensource.html

Real-Time Optical Flow Example



http://www.cs.unc.edu/~ssinha/Research/GPU_KLT/

http://cs.unc.edu/~cmzach/opensource.html

Dense Optical Flow

- Dense measurements can be obtained by adding smoothness constraints.

Color map

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
  - B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. IJCAI*, pp. 674–679, 1981.