Lecture #9

Image features, optical flow, motion segmentation and tracking
Today

• Image Features
• Optical Flow
• Motion segmentation
• Tracking
Uses for feature point detectors and descriptors in computer vision and graphics.

- Image alignment and building panoramas
- 3D reconstruction
- Motion tracking
- Object recognition
- Indexing and database retrieval
- Robot navigation
- ... other

Slide credit: B. Freeman and A. Torralba
How do we build a panorama?

• We need to match (align) images

Slide credit: B. Freeman and A. Torralba
Matching with Features

- Detect feature points in both images
Matching with Features

- Detect feature points in both images
- Find corresponding pairs
Matching with Features

- Detect feature points in both images
- Find corresponding pairs
- Use these matching pairs to align images – the required mapping is called a homography.
Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

\[ x_1 = [x_1^{(1)}, \ldots, x_d^{(1)}] \]

\[ x_2 = [x_1^{(2)}, \ldots, x_d^{(2)}] \]

3) Matching: Determine correspondence between descriptors in two views
Local features: desired properties

• Repeatability
  – The same feature can be found in several images despite geometric and photometric transformations

• Saliency
  – Each feature has a distinctive description

• Compactness and efficiency
  – Many fewer features than image pixels

• Locality
  – A feature occupies a relatively small area of the image; robust to clutter and occlusion

Slide credit: K. Grauman
Goal: interest operator repeatability

- We want to detect (at least some of) the same points in both images.

No chance to find true matches!

- Yet we have to be able to run the detection procedure *independently* per image.
Goal: descriptor distinctiveness

- We want to be able to reliably determine which point goes with which.

- Must provide some invariance to geometric and photometric differences between the two views.
• What points would you choose?
Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

3) Matching: Determine correspondence between descriptors in two views

\[
x_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]
\]

\[
x_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]
\]

Slide credit: K. Grauman
Corners as distinctive interest points

- We should easily recognize the point by looking through a small window.
- Shifting a window in any direction should give a large change in intensity.

“flat” region: no change in all directions

“edge”: no change along the edge direction

“corner”: significant change in all directions

Slide credit: K. Grauman, Alyosha Efros, Darya Frolova, Denis Simakov
Harris Detector: Mathematics

Window-averaged squared change of intensity induced by shifting the image data by \([u,v]\):

\[
E(u, v) = \sum_{x, y} w(x, y) \left[ I(x + u, y + v) - I(x, y) \right]^2
\]

Window function \(w(x, y)\) =

1 in window, 0 outside  
or  
Gaussian

Slide credit: B. Freeman, A. Torralba, Darya Frolova, Denis Simakov
Harris Detector: Mathematics

- Taylor series approximation to shifted image gives quadratic form for error as function of image shifts.

\[ E(u, v) \approx \sum_{x, y} w(x, y)[I(x, y) + uI_x + vI_y - I(x, y)]^2 \]

\[ = \sum_{x, y} w(x, y)[uI_x + vI_y]^2 \]

\[ = (u \quad v) \sum_{x, y} w(x, y) \begin{bmatrix} I_xI_x & I_xI_y \\ I_xI_y & I_yI_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \]
Harris Detector: Mathematics

- Expanding $I(x,y)$ in a Taylor series expansion, we have, for small shifts $[u,v]$, a \textit{quadratic} approximation to the error surface between a patch and itself, shifted by $[u,v]$:

$$E(u, v) \cong \begin{bmatrix} u, v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$$

where $M$ is a $2\times2$ matrix computed from image derivatives:

$$M = \sum_{x,y} w(x, y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

$M$ is often called structure tensor

Slide credit: B. Freeman, A. Torralba, Darya Frolova, Denis Simakov
Harris Detector: Mathematics

\[ M = \sum w(x, y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_y I_x & I_y I_y \end{bmatrix} \]

2 x 2 matrix of image derivatives (averaged in neighborhood of a point).

Notation:

\[ I_x \leftrightarrow \frac{\partial I}{\partial x} \]
\[ I_y \leftrightarrow \frac{\partial I}{\partial y} \]
\[ I_x I_y \leftrightarrow \frac{\partial I}{\partial x} \frac{\partial I}{\partial y} \]
What does this matrix reveal?

First, consider an axis-aligned corner:
What does this matrix reveal?

First, consider an axis-aligned corner:

\[ M = \sum \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \]

This means dominant gradient directions align with x or y axis.

Look for locations where both \( \lambda \)'s are large.

If either \( \lambda \) is close to 0, then this is not corner-like.

What if we have a corner that is not aligned with the image axes?
What does this matrix reveal?

Since $M$ is symmetric, we have

$$M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T$$

$Mx_i = \lambda_i x_i$

The eigenvalues of $M$ reveal the amount of intensity change in the two principal orthogonal gradient directions in the window.
Harris Detector: Mathematics

Classification of image points using eigenvalues of $M$:

- $\lambda_1$ and $\lambda_2$ are small; $E$ is almost constant in all directions.
- $\lambda_2 \gg \lambda_1$; “Corner” $\lambda_1$ and $\lambda_2$ are large, $\lambda_1 \sim \lambda_2$; $E$ increases in all directions.
- $\lambda_1 \gg \lambda_2$; “Edge” region.
- “Flat” region.

Slide credit: B. Freeman, A. Torralba, Darya Frolova, Denis Simakov
Measure of corner response:

\[ R = \det M - k (\text{trace } M)^2 \]

\[ \det M = \lambda_1 \lambda_2 \]
\[ \text{trace } M = \lambda_1 + \lambda_2 \]

\( (k - \text{empirical constant, } k = 0.04-0.06) \)

(Shi-Tomasi variation: use \( \min(\lambda_1, \lambda_2) \) instead of \( R \))

Slide credit: B. Freeman, A. Torralba, Darya Frolova, Denis Simakov
• $R$ depends only on eigenvalues of $M$
• $R$ is large for a corner
• $R$ is negative with large magnitude for an edge
• $|R|$ is small for a flat region
Harris Detector

• The Algorithm:
  – Find points with large corner response function $R$ ($R > \text{threshold}$)
  – Take the points of local maxima of $R$
Harris corner detector algorithm

• Compute image gradients $I_x, I_y$ for all pixels
• For each pixel
  – Compute $M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$ by looping over neighbors $x,y$
  – Compute $R = \det M - k (\text{trace } M)^2$

• Find points with large corner response function $R$ ($R > \text{threshold}$)
• Take the points of locally maximum $R$ as the detected feature points (i.e., pixels where $R$ is bigger than for all the 4 or 8 neighbors).

Slide credit: B. Freeman, A. Torralba, Darya Frolova, Denis Simakov
Harris Detector: Workflow

Slide credit: B. Freeman, A. Torralba, Darya Frolova, Denis Simakov
Harris Detector: Workflow

Compute corner response $R$
Harris Detector: Workflow

Find points with large corner response: \( R > \text{threshold} \)

Slide credit: B. Freeman, A. Torralba, Darya Frolova, Denis Simakov
Harris Detector: Workflow

Take only the points of local maxima of $R$

Slide credit: B. Freeman, A. Torralba, Darya Frolova, Denis Simakov
Harris Detector: Workflow

Slide credit: B. Freeman, A. Torralba, Darya Frolova, Denis Simakov
Properties of the Harris corner detector

- Rotation invariant? Yes

\[ M = X \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} X^T \]

- Scale invariant?
Properties of the Harris corner detector

- Rotation invariant? Yes

- Scale invariant? No

All points will be classified as edges

Corner!
Scale invariant interest points

How can we independently select interest points in each image, such that the detections are repeatable across different scales?
Automatic scale selection

Intuition:
• Find scale that gives local maxima of some function $f$ in both position and scale.
• What can be the “signature” function?
Recall: Edge detection

Edge = maximum of derivative

Slide credit: K. Grauman, S. Seitz
Recall: Edge detection

Edge = zero crossing of second derivative

$\frac{d^2}{dx^2} g$

Second derivative of Gaussian (Laplacian)

$f \ast \frac{d^2}{dx^2} g$

Edge

$\Sigma = 50$

Source: S. Seitz

Slide credit: K. Grauman
From edges to blobs

- Edge = ripple
- Blob = superposition of two ripples

Spatial selection: the magnitude of the Laplacian response will achieve a maximum at the center of the blob, provided the scale of the Laplacian is “matched” to the scale of the blob

Slide credit: K. Grauman, L. Lazebnik
Blob detection in 2D

- Laplacian of Gaussian: Circularly symmetric operator for blob detection in 2D

\[ \nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \]
Blob detection in 2D: scale selection

- Laplacian-of-Gaussian = “blob” detector

\[ \nabla^2 g = \frac{\partial^2 g}{\partial x^2} + \frac{\partial^2 g}{\partial y^2} \]
Blob detection in 2D

- We define the *characteristic scale* as the scale that produces peak of Laplacian response.
Example

Original image at $\frac{3}{4}$ the size

Slide credit: K. Grauman
Original image at \(\frac{3}{4}\) the size
Scale invariant interest points

Interest points are local maxima in both position and scale.

\[ L_{xx}(\sigma) + L_{yy}(\sigma) \]

Squared filter response maps

⇒ List of \((x, y, \sigma)\)

Slide credit: K. Grauman
Scale-space blob detector: Example
Technical detail

• We can approximate the Laplacian with a difference of Gaussians; more efficient to implement.

\[ L = \sigma^2 \left( G_{xx}(x, y, \sigma) + G_{yy}(x, y, \sigma) \right) \]

(Laplacian)

\[ DoG = G(x, y, k\sigma) - G(x, y, \sigma) \]

(Difference of Gaussians)

\[
\begin{align*}
I(k\sigma) & \quad - \quad I(\sigma) \quad = \quad I(k\sigma) - I(\sigma)
\end{align*}
\]
Local features: main components

1) **Detection:** Identify the interest points

2) **Description:** Extract vector feature descriptor surrounding each interest point.

\[
x_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]
\]

\[
x_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]
\]

3) **Matching:** Determine correspondence between descriptors in two views

Slide credit: K. Grauman
Geometric transformations

e.g. scale, translation, rotation

Slide credit: K. Grauman
Photometric transformations

Figure from T. Tuytelaars ECCV 2006 tutorial

Slide credit: K. Grauman
Raw patches as local descriptors

The simplest way to describe the neighborhood around an interest point is to write down the list of intensities to form a feature vector.

But this is very sensitive to even small shifts, rotations.

Slide credit: K. Grauman
**SIFT descriptor [Lowe 2004]**

- Use histograms to bin pixels within sub-patches according to their orientation.

*Why subpatches?*

*Why does SIFT have some illumination invariance?*

Slide credit: K. Grauman
Making descriptor rotation invariant

- Rotate patch according to its dominant gradient orientation
- This puts the patches into a canonical orientation.
SIFT descriptor [Lowe 2004]

Extraordinarily robust matching technique

- Can handle changes in viewpoint
  - Up to about 60 degree out of plane rotation
- Can handle significant changes in illumination
  - Sometimes even day vs. night (below)

Fast and efficient—can run in real time

Lots of code available

Example

NASA Mars Rover images

Slide credit: K. Grauman
Example

NASA Mars Rover images with SIFT feature matches
Figure by Noah Snavely

Slide credit: K. Grauman
SIFT properties

• Invariant to
  – Scale
  – Rotation

• Partially invariant to
  – Illumination changes
  – Camera viewpoint
  – Occlusion, clutter
Local features: main components

1) Detection: Identify the interest points

2) Description: Extract vector feature descriptor surrounding each interest point.

3) Matching: Determine correspondence between descriptors in two views

\[
x_1 = [x_1^{(1)}, \ldots, x_d^{(1)}]
\]

\[
x_2 = [x_1^{(2)}, \ldots, x_d^{(2)}]
\]
Matching local features

Slide credit: K. Grauman
Matching local features

To generate **candidate matches**, find patches that have the most similar appearance (e.g., lowest SSD)

Simplest approach: compare them all, take the closest (or closest k, or within a thresholded distance)

Slide credit: K. Grauman
At what SSD value do we have a good match?

To add robustness to matching, can consider ratio: distance to best match / distance to second best match

If low, first match looks good.

If high, could be ambiguous match.

Slide credit: K. Grauman
Matching SIFT Descriptors

- Nearest neighbor (Euclidean distance)
- Threshold ratio of nearest to 2nd nearest descriptor
Recap: robust feature-based alignment

Slide credit: K. Grauman, L. Lazebnik
Recap: robust feature-based alignment

- Extract features
Recap: robust feature-based alignment

- Extract features
- Compute *putative matches*
Recap: robust feature-based alignment

- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation $T$ (small group of putative matches that are related by $T$)
Recap: robust feature-based alignment

- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation $T$ (small group of putative matches that are related by $T$)
  - *Verify* transformation (search for other matches consistent with $T$)

Slide credit: K. Grauman, L. Lazebnik
Recap: robust feature-based alignment

• Extract features
• Compute *putative matches*
• Loop:
  – *Hypothesize* transformation $T$ (small group of putative matches that are related by $T$)
  – *Verify* transformation (search for other matches consistent with $T$)

Slide credit: K. Grauman, L. Lazebnik
Applications of local invariant features

- Wide baseline stereo
- Motion tracking
- Panoramas
- Mobile robot navigation
- 3D reconstruction
- Recognition
- ...

Slide credit: K. Grauman
Automatic mosaicing

http://www.cs.ubc.ca/~mbrown/autostitch/autostitch.html

Slide credit: K. Grauman
Wide baseline stereo

[Image from T. Tuytelaars ECCV 2006 tutorial]

Slide credit: K. Grauman
Recognition of specific objects, scenes

Schmid and Mohr 1997

Sivic and Zisserman, 2003

Rothganger et al. 2003

Lowe 2002

Slide credit: K. Grauman
Summary

• Interest point detection
  – Harris corner detector
  – Laplacian of Gaussian, automatic scale selection

• Invariant descriptors
  – Rotation according to dominant gradient direction
  – Histograms for robustness to small shifts and translations (SIFT descriptor)
Reading Assignment

SURF: Speeded Up Robust Features

Herbert Bay¹, Tinne Tuytelaars², and Luc Van Gool¹²

¹ ETH Zurich
{bay, vangool}@vision.ee.ethz.ch
² Katholieke Universiteit Leuven
{Tinne.Tuytelaars, Luc.Vangool}@esat.kuleuven.be


http://www.vision.ee.ethz.ch/~surf
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 and perceptual organization

- Sometimes, motion is the only cue
Motion and perceptual organization

- Sometimes, motion is the only 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


Slide credit: J. Hays
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
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
Today

• Image Features
• Optical Flow
• Motion segmentation
• Tracking
Optical flow

Vector field function of the spatio-temporal image brightness variations

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

Slide credit: Fei-Fei Li
Motion field

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

- **$X(t)$** is a moving 3D point
- **Velocity of scene point:** $V = \frac{dX}{dt}$
- **$x(t) = (x(t), y(t))$** is the projection of $X$ 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 $x=(x,y)$ with respect to $t$ (using quotient rule):

$$
\begin{align*}
x &= f \frac{X}{Z} \\
v_x &= f \frac{ZV_x - V_z X}{Z^2} \\
    &= \frac{fV_x - V_z x}{Z}
\end{align*}
$$

$$
\begin{align*}
y &= f \frac{Y}{Z} \\
v_y &= \frac{fV_y - V_z y}{Z}
\end{align*}
$$

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: $V$ is constant everywhere

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

\[v = \frac{1}{Z} (v_0 - V_z x), \quad v_0 = (fV_x, fV_y)\]
Motion field and parallax

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

• The length of the motion vectors is inversely proportional to the depth $Z$

• $V_z$ is nonzero:
  
  – Every motion vector points toward (or away from) the vanishing point of the translation direction

Slide credit: A. Efros
Motion field and parallax

• Pure translation: $V$ is constant everywhere
  
  $$v = \frac{1}{Z} (v_0 - V_z x), \quad v_0 = (f V_x, f V_y)$$

• The length of the motion vectors is inversely proportional to the depth $Z$

• $V_z$ is nonzero:
  
  – Every motion vector points toward (or away from) the vanishing point of the translation direction

• $V_z$ is zero:
  
  – Motion is parallel to the image plane, all the motion vectors are parallel

Slide credit: A. Efros
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

Slide credit: J. Hays
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

Slide credit: S. 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:

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

Linearizing the right side using Taylor expansion:

Image derivative along \( x \) \( \quad \) Difference over frames

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

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

Hence,

\[
I_x \cdot u + I_y \cdot v + I_t \approx 0 \quad \rightarrow \nabla I \cdot [u \ v]^T + I_t = 0
\]
How does this make sense?

\[ \nabla I \cdot \begin{bmatrix} u & v \end{bmatrix}^T + I_t = 0 \]

- What do the static image gradients have to do with motion estimation?
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

Slide credit: J. Hays
The aperture problem

Perceived motion

Slide credit: J. Hays
The barber pole illusion

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

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

Slide credit: J. Hays
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:

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

\[A \ d = b\]

25x2 2x1 25x1

Slide credit: S. 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

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

Least squares solution for \(d\) given by

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

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

\(A^T A\)

\(A^T b\)

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

Slide credit: S. Savarese
Conditions for solvability

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

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

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

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

- \(A^T A\) 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 you of anything?

Criteria for Harris corner detector

Slide credit: S. Saverese
$M = A^T A$ is the *second moment matrix*!

(Harris corner detector...)

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

- 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

Slide credit: S. Savarese
Classification of image points using eigenvalues of $M$:

- **Corner**
  - $\lambda_1$ and $\lambda_2$ are large,
  - $\lambda_1 \sim \lambda_2$;
  - $E$ increases in all directions

- **Edge**
  - $\lambda_2 \gg \lambda_1$
- **Flat**
  - $\lambda_1$ and $\lambda_2$ are small;
  - $E$ is almost constant in all directions

Slide credit: S. Savarese, Darya Frolova, Denis Simakov
Low-texture region

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

- gradients have small magnitude
- small $l_1$, small $l_2$
\[ \sum \nabla I (\nabla I)^T \]

- gradients very large or very small
- large $l_1$, small $l_2$
High-texture region

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

- gradients are different, large magnitudes
- large $l_1$, large $l_2$
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!
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 (2nd order terms dominate)
  - How might we solve this problem?
Reduce the resolution!

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003
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 $f_i$ 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. $u_i = u_i^* + u'_i$,
    $v_i = v_i^* + v'_i$. 
Coarse-to-fine optical flow estimation

Gaussian pyramid of image 1

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

image 1

Gaussian pyramid of image 2

image 2

Slide credit: S. Savarese
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)

Slide credit: J. Hays
Optical Flow Results

Lucas-Kanade without pyramids
Fails in areas of large motion

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

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Slide credit: J. Hays
Reading Assignment

Large Displacement Optical Flow*

Thomas Brox¹  Christoph Bregler²  Jitendra Malik¹

¹University of California, Berkeley
Berkeley, CA, 94720, USA
{brox, malik}@eecs.berkeley.edu

²Courant Institute, New York University
New York, NY, 10003, USA
bregler@courant.nyu.edu

• T. Brox, C. Bregler, J. Malik, Large displacement optical flow, CVPR 2009
Today

• Image Features
• Optical Flow
• Motion segmentation
• Tracking
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


Slide credit: S. Savarese
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:

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

- Each pixel provides 1 linear constraint in 6 unknowns
- Least squares minimization:

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

Slide credit: S. Savarese
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
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


Slide credit: S. Savarese
Reading Assignment

Key-Segments for Video Object Segmentation

Yong Jae Lee, Jaechul Kim, and Kristen Grauman
University of Texas at Austin
yjlee0222@utexas.edu, jaechul@cs.utexas.edu, grauman@cs.utexas.edu

• Y.J. Lee, J. Kim and K. Grauman, Key-Segments for Video Object Segmentation, ICCV 2011
Today

- Image Features
- Optical Flow
- Feature Tracking
Tracking

Slide credit: Fei-Fei Li, K. Grauman, D. Ramanan
Optical flow for tracking?

If we have more than just a pair of frames, we could compute flow from one to the next:

But flow only reliable for small motions, and we may have occlusions, textureless regions that yield bad estimates anyway...

Slide credit: K. Grauman
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 overtime
  – Need to know when to terminate a track
Motion estimation techniques

• Optical flow
  – 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)
Feature-based matching for motion

Search window is centered at the point where we last saw the feature, in image $I_1$.

Best match = position where we have the highest normalized cross-correlation value.

Slide credit: K. Grauman
Feature-based matching for motion

- For a discrete matching search, what are the tradeoffs of the chosen search window size?
- Which patches to track?
  - Select interest points – e.g. corners
- Where should the search window be placed?
  - Near match at previous frame
  - More generally, taking into account the expected dynamics of the object

Slide credit: K. Grauman
Detection vs. tracking

Slide credit: K. Grauman
Detection vs. tracking

Detection: We detect the object independently in each frame and can record its position over time, e.g., based on blob’s centroid or detection window coordinates.
Tracking with *dynamics*: We use image measurements to estimate position of object, but also incorporate position predicted by dynamics, i.e., our expectation of object’s motion pattern.
Tracking with *dynamics*: We use image measurements to estimate position of object, but also incorporate position predicted by dynamics, i.e., our expectation of object’s motion pattern.
Tracking with dynamics

• Use model of expected motion to predict where objects will occur in next frame, even before seeing the image.

• Intent:
  – Do less work looking for the object, restrict the search.
  – Get improved estimates since measurement noise is tempered by smoothness, dynamics priors.

• Assumption: continuous motion patterns:
  – Camera is not moving instantly to new viewpoint
  – Objects do not disappear and reappear in different places in the scene
  – Gradual change in pose between camera and scene

Slide credit: K. Grauman
Tracking as inference

• The *hidden state* consists of the true parameters we care about, denoted $X$.

• The *measurement* is our noisy observation that results from the underlying state, denoted $Y$.

• At each time step, state changes (from $X_{t-1}$ to $X_t$) and we get a new observation $Y_t$. 
State vs. observation

Hidden state: parameters of interest
Measurement: what we get to directly observe

Slide credit: K. Grauman
Tracking as inference

• The *hidden state* consists of the true parameters we care about, denoted $X$.

• The *measurement* is our noisy observation that results from the underlying state, denoted $Y$.

• At each time step, state changes (from $X_{t-1}$ to $X_t$) and we get a new observation $Y_t$.

• **Our goal:** recover most likely state $X_t$ given
  – All observations seen so far.
  – Knowledge about dynamics of state transitions.
Tracking as inference: intuition

Time $t$

Belief

Measurement

Corrected prediction

Time $t+1$
Tracking as inference: intuition

Belief: prediction

Corrected prediction

measurement

old belief

Time t

Time t+1

Slide credit: K. Grauman
Independence assumptions

• Only immediate past state influences current state

\[ P(X_t|X_0,\ldots,X_{t-1}) = P(X_t|X_{t-1}) \]

dynamics model

• Measurement at time \( t \) depends on current state

\[ P(Y_t|X_0,Y_0\ldots,X_{t-1},Y_{t-1},X_t) = P(Y_t|X_t) \]

observation model

Slide credit: K. Grauman
• Prediction:
  – Given the measurements we have seen up to this point, what state should we predict?

\[ P(X_t | y_0, \ldots, y_{t-1}) \]

• Correction:
  – Now given the current measurement, what state should we predict?

\[ P(X_t | y_0, \ldots, y_t) \]
Questions

• How to represent the known dynamics that govern the changes in the states?

• How to represent relationship between state and measurements, plus our uncertainty in the measurements?

• How to compute each cycle of updates?

**Representation:** We’ll consider the class of *linear* dynamic models, with associated Gaussian pdfs.

**Updates:** via the Kalman filter.
Notation reminder

\[ x \sim N(\mu, \Sigma) \]

• Random variable with Gaussian probability distribution that has the mean vector \( \mu \) and covariance matrix \( \Sigma \).

• \( x \) and \( \mu \) are \( d \)-dimensional, \( \Sigma \) is \( d \times d \).

If \( x \) is 1-d, we just have one \( \Sigma \) parameter - the variance: \( \sigma^2 \)

Slide credit: K. Grauman
Linear dynamic model

- Describe the \textit{a priori} knowledge about
  - System dynamics model: represents evolution of state over time.
    \[ x_t \sim N(Dx_{t-1}, \Sigma_d) \]
    \[ n \times 1 \quad n \times n \quad n \times 1 \]
  - Measurement model: at every time step we get a noisy measurement of the state.
    \[ y_t \sim N(Mx_t, \Sigma_m) \]
    \[ m \times 1 \quad m \times n \quad n \times 1 \]
Example: randomly drifting points

\[ x_t \sim N(Dx_{t-1}; \Sigma_d) \]

- Consider a stationary object, with state as position
- Position is constant, only motion due to random noise term.
- State evolution is described by identity matrix $D=I$
Example: Constant velocity (1D points)
Example: Constant velocity (1D points)

- State vector: position $p$ and velocity $v$

\[
x_t = \begin{bmatrix} p_t \\ v_t \end{bmatrix}, \quad p_t =
\]

\[
x_t = D_t x_{t-1} + \text{noise} =
\]

- Measurement is position only

\[
y_t = Mx_t + \text{noise} =
\]
Questions

• How to represent the known dynamics that govern the changes in the states?

• How to represent relationship between state and measurements, plus our uncertainty in the measurements?

• How to compute each cycle of updates?

**Representation**: We’ll consider the class of *linear* dynamic models, with associated Gaussian pdfs.

**Updates**: via the Kalman filter.

Slide credit: K. Grauman
The Kalman filter

- Method for tracking linear dynamical models in Gaussian noise
- The predicted/corrected state distributions are Gaussian
  - Only need to maintain the mean and covariance
  - The calculations are easy
Kalman filter

Know corrected state from previous time step, and all measurements up to the current one →
Predict distribution over next state.

**Time update**

("Predict")

Receive measurement

Know prediction of state, and next measurement →
Update distribution over current state.

**Measurement update**

("Correct")

\[
P(X_t | y_0, \ldots, y_{t-1})
\]

Mean and std. dev. of predicted state:

\[
\mu_t^-, \sigma_t^-
\]

Time advances: \( t++ \)

\[
P(X_t | y_0, \ldots, y_t)
\]

Mean and std. dev. of corrected state:

\[
\mu_t^+, \sigma_t^+
\]

Slide credit: K. Grauman
1D Kalman filter: Prediction

- Have linear dynamic model defining predicted state evolution, with noise
  \[ X_t \sim N(dx_{t-1}, \sigma^2_d) \]
- Want to estimate predicted distribution for next state
  \[ P(X_t|y_0, \ldots, y_{t-1}) = N(\mu_t^-, (\sigma_t^-)^2) \]
- Update the mean:
  \[ \mu_t^- = d\mu_{t-1}^+ \]
- Update the variance:
  \[ (\sigma_t^-)^2 = \sigma^2_d + (d\sigma_{t-1}^+)^2 \]
1D Kalman filter: Correction

- Have linear model defining the mapping of state to measurements:
  \[ Y_t \sim N(mx_t, \sigma_m^2) \]

- Want to estimate corrected distribution given latest meas.:
  \[ P(X_t | y_0, \ldots, y_t) = N(\mu_t^+, (\sigma_t^+)^2) \]

- Update the mean:
  \[ \mu_t^+ = \frac{\mu_t^- \sigma_m^2 + my_t (\sigma^-)^2}{\sigma_m^2 + m^2 (\sigma^-)^2} \]

- Update the variance:
  \[ (\sigma_t^+)^2 = \frac{\sigma_m^2 (\sigma^-)^2}{\sigma_m^2 + m^2 (\sigma^-)^2} \]

Slide credit: L. Lazebnik
Prediction vs. correction

\[ \mu_t^+ = \frac{\mu_t^- \sigma_m^2 + m y_t (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2} \quad (\sigma_t^+)^2 = \frac{\sigma_m^2 (\sigma_t^-)^2}{\sigma_m^2 + m^2 (\sigma_t^-)^2} \]

• What if there is no prediction uncertainty \((\sigma_t^- = 0)\)?

\[ \mu_t^+ = \mu_t^- \quad (\sigma_t^+)^2 = 0 \]

The measurement is ignored!

• What if there is no measurement uncertainty \((\sigma_m = 0)\)?

\[ \mu_t^+ = \frac{y_t}{m} \quad (\sigma_t^+)^2 = 0 \]

The prediction is ignored!

Slide credit: L. Lazebnik
Constant velocity model

Kalman filter processing

- state
- x measurement
- * predicted mean estimate
- + corrected mean estimate
- bars: variance estimates before and after measurements

Slide credit: K. Grauman
Kalman filter processing

- State
- Measurement
- Predicted mean estimate
- Corrected mean estimate
- Bars: Variance estimates before and after measurements

Constant velocity model

Slide credit: K. Grauman
Constant velocity model

Kalman filter processing

- state
- x measurement
- * predicted mean estimate
- + corrected mean estimate
- bars: variance estimates before and after measurements

Slide credit: K. Grauman
Kalman filter processing

- state
- x measurement

* predicted mean estimate
+ corrected mean estimate

bars: variance estimates before and after measurements

Constant velocity model

Time $t$  

Time $t+1$  

Slide credit: K. Grauman
Detection, Tracking, and Censusing

Censusing natural populations of bats is important for understanding the ecological and economic impact of these animals on terrestrial ecosystems. Colonies of Brazilian free-tailed bats (*Tadarida brasiliensis*) are of particular interest because they represent some of the largest aggregations of mammals known to mankind. It is challenging to census these bats accurately, since they emerge in large numbers at night from their day-time roosting sites. We have used infrared thermal cameras to record Brazilian free-tailed bats in California, Massachusetts, New Mexico, and Texas. We have developed an automated image analysis system that detects, tracks, and counts the emerging bats.

Research Team

Faculty

- Margrit Betke
- Cutler Cleveland
- Thomas Kunz
- Stan Solaroff

- Thomas G. Hallam, University of Tennessee
- Nicholas C. Makris, Massachusetts Institute of Technology
- Gary F. McCraken, University of Tennessee
- John K. Westbrook, US Department of Agriculture

Students and Postdocs


---

Slide credit: K. Grauman

A bat census

http://www.cs.bu.edu/~betke/research/bats/

Slide credit: K. Grauman
Tracking: issues

• **Initialization**
  – Often done manually
  – Background subtraction, detection can also be used

• **Data association**, multiple tracked objects
  – Occlusions, clutter
  – Which measurements go with which tracks?
Tracking: issues

- **Initialization**
  - Often done manually
  - Background subtraction, detection can also be used

- **Data association**, multiple tracked objects
  - Occlusions, clutter

- **Deformable** and articulated objects

- **Constructing accurate models** of dynamics
  - E.g., Fitting parameters for a linear dynamics model

- **Drift**
  - Accumulation of errors over time
Tracking Summary

• Tracking as inference
  – Goal: estimate posterior of object position given measurement

• Linear models of dynamics
  – Represent state evolution and measurement models

• Kalman filters
  – Recursive prediction/correction updates to refine measurement

• General tracking challenges

Slide credit: K. Grauman
Reading Assignment

Visual Tracking with Histograms and Articulating Blocks

S.M.S. Nejhum, J. Ho and M-H Yang, Visual Tracking with Histograms and Articulating Blocks, CVPR 2008