EE795: Computer Vision and Intelligent Systems

Spring 2012
TTh 17:30-18:45 FDH 204

Lecture 15
130319

http://www.ee.unlv.edu/~b1morris/ecg795/
Outline

- Review
- Optical Flow
Motion estimation

- **Input:** sequence of images
- **Output:** point correspondence

- **Feature correspondence:** “Feature Tracking”
  - we’ve seen this already (e.g., SIFT)
  - can modify this to be more accurate/efficient if the images are in sequence (e.g., video)

- **Pixel (dense) correspondence:** “Optical Flow”
Dense Motion Estimation

- Motion is extremely important in vision
- Biologically: motion indicates what is food and when to run away
  - We have evolved to be very sensitive to motion cues (peripheral vision)
- Alignment of images and motion estimation is widely used in computer vision
  - Optical flow
  - Motion compensation for video compression
  - Image stabilization
  - Video summarization
Translational Alignment

- Motion estimation between images requires an error metric for comparison
- Sum of squared differences (SSD)
  - $E_{SSD}(u) = \sum_i [I_1(x_i + u) - I_0(x_i)]^2 = \sum_i e_i^2$
  - $u = (u, v)$ – is a displacement vector (can be subpixel)
  - $e_i$ - residual error
- Brightness constancy constraint
  - Assumption that corresponding pixels will retain the same value in two images
  - Objects tend to maintain the perceived brightness under varying illumination conditions [Horn 1974]
- Color images processed by channels and summed or converted to colorspace that considers only luminance
SSD Improvements

- As we have seen many times in class, SSD is the simplest approach and can be improved.

- Robust error metrics
  - $L_1$ norm (sum absolute differences)
    - Better outlier resilience

- Spatially varying weights
  - Weighted SSD to weight contribution of each pixel during matching
    - Ignore certain parts of the image (e.g. foreground), down-weight objects during images stabilization

- Bias and gain
  - Normalize exposure between images
    - Address brightness constancy
Correlation

- Instead of minimizing pixel differences, maximize correlation
- Normalized cross-correlation

\[ E_{NCC}(u) = \frac{\sum_i [I_0(x_i) - \bar{I}_0] [I_1(x_i + u) - \bar{I}_1]}{\sqrt{\sum_i [I_0(x_i) - \bar{I}_0]^2} \sqrt{\sum_i [I_1(x_i + u) - \bar{I}_1]^2}}, \]

\[ \bar{I}_0 = \frac{1}{N} \sum_i I_0(x_i) \quad \text{and} \]
\[ \bar{I}_1 = \frac{1}{N} \sum_i I_1(x_i + u) \]

- Normalize by the patch intensities
- Value is between \([-1, 1]\) which makes it easy to use results (e.g. threshold to find matching pixels)
Optical flow

- Dense pixel correspondence
Problem definition: optical flow

How to estimate pixel motion from image H to image I?

- Solve pixel correspondence problem
  - given a pixel in H, look for nearby pixels of the same color in I

Key assumptions

- **color constancy**: a point in H looks the same in I
  - For grayscale images, this is **brightness constancy**
- **small motion**: points do not move very far

This is called the **optical flow** problem
Optical flow constraints (grayscale images)

Let’s look at these constraints more closely

- brightness constancy: Q: what’s the equation?
  
  \[ H(x, y) = I(x + u, y + v) \]

- small motion: (u and v are less than 1 pixel)
  
  - suppose we take the Taylor series expansion of I:

  \[ I(x+u, y+v) = I(x, y) + \frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v + \text{higher order terms} \]

  \[ \approx I(x, y) + \frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v \]
Optical flow equation

• Combining these two equations

\[ 0 = I(x + u, y + v) - H(x, y) \]

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

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

\[ \approx I_t + I_x u + I_y v \]

\[ \approx I_t + \nabla I \cdot [u \ v] \]

In the limit as u and v go to zero, this becomes exact

\[ 0 = I_t + \nabla I \cdot \left[ \frac{\partial x}{\partial t} \frac{\partial y}{\partial t} \right] \]
Optical flow equation

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

- Q: how many unknowns and equations per pixel?
  - \( u \) and \( v \) are unknown, 1 equation

Intuitively, what does this constraint mean?

- The component of the flow in the gradient direction is determined
- The component of the flow parallel to an edge is unknown

This explains the Barber Pole illusion

- [link](http://www.sandlotscience.com/Ambiguous/Barberpole_Illusion.htm)
Aperture problem
Aperture problem
Solving the aperture problem

• **Basic idea:** assume motion field is smooth

• **Horn & Schunk:** add smoothness term

\[ \int \int (I_t + \nabla I \cdot [u \ v])^2 + \lambda^2 (||\nabla u||^2 + ||\nabla v||^2) \ dx \ dy \]

• **Lucas & Kanade:** assume locally constant motion
  ▫ pretend the pixel’s neighbors have the same \((u,v)\)

• **Many other methods exist. Here’s an overview:**
  ▫ [http://vision.middlebury.edu/flow/](http://vision.middlebury.edu/flow/)
Lucas-Kanade flow

- How to get more equations for a pixel?
  - Basic idea: impose additional constraints
    - most common is to assume that the flow field is smooth locally
    - one method: pretend the pixel’s neighbors have the same \((u,v)\)
      - If we use a 5x5 window, that gives us 25 equations per pixel!

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

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

\[
A \quad 25 \times 2 \\
d \quad 2 \times 1 \\
b \quad 25 \times 1
\]
How to get more equations for a pixel?

- Basic idea: impose additional constraints
  - most common is to assume that the flow field is smooth locally
  - one method: pretend the pixel’s neighbors have the same (u,v)
    - If we use a 5x5 window, that gives us 25*3 equations per pixel!

\[
0 = I_t(p_i)[0, 1, 2] + \nabla I(p_i)[0, 1, 2] \cdot [u \ v]
\]

\[
\begin{bmatrix}
I_x(p_1)[0] & I_y(p_1)[0] \\
I_x(p_1)[1] & I_y(p_1)[1] \\
I_x(p_1)[2] & I_y(p_1)[2] \\
\vdots & \vdots \\
I_x(p_{25})[0] & I_y(p_{25})[0] \\
I_x(p_{25})[1] & I_y(p_{25})[1] \\
I_x(p_{25})[2] & I_y(p_{25})[2]
\end{bmatrix} \begin{bmatrix}
u \\
v
\end{bmatrix} = - \begin{bmatrix}
I_t(p_1)[0] \\
I_t(p_1)[1] \\
I_t(p_1)[2] \\
\vdots \\
I_t(p_{25})[0] \\
I_t(p_{25})[1] \\
I_t(p_{25})[2]
\end{bmatrix}
\]

\[
A_{75\times2} \quad d_{2\times1} \quad b_{75\times1}
\]
Lucas-Kanade flow

Prob: we have more equations than unknowns

\[ A \begin{bmatrix} d \end{bmatrix}_{25x2} = b \begin{bmatrix} 1 \end{bmatrix}_{2x1} \quad \text{25x1} \quad \text{minimize} \quad \|Ad - b\|^2 \]

Solution: solve least squares problem

- minimum least squares solution given by solution (in \( d \)) of:

\[ (A^T A) \begin{bmatrix} d \end{bmatrix}_{2x2} = A^T b \begin{bmatrix} 1 \end{bmatrix}_{2x1} \]

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

- The summations are over all pixels in the \( K \times K \) window
- This technique was first proposed by Lucas & Kanade (1981)
Conditions for solvability

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

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

\[A^T A\]

\[A^T b\]

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

• Does this look familiar?
  • \(A^T A\) is the Harris matrix
Observation

- This is a two image problem BUT
  - Can measure sensitivity by just looking at one of the images!
  - This tells us which pixels are easy to track, which are hard
    - very useful for feature tracking...
Errors in Lucas-Kanade

- What are the potential causes of errors in this procedure?
  - Suppose $A^T A$ is easily invertible
  - Suppose there is not much noise in the image

- When our assumptions are violated
  - Brightness constancy is not satisfied
  - The motion is not small
  - A point does not move like its neighbors
    - window size is too large
    - what is the ideal window size?
Improving accuracy

- Recall our small motion assumption
  
  \[ 0 = I(x + u, y + v) - H(x, y) \]
  
  \[ \approx I(x, y) + I_x u + I_y v - H(x, y) \]

- Not exact, need higher order terms to do better
  
  \[ = I(x, y) + I_x u + I_y v + \text{higher order terms} - H(x, y) \]

- Results in polynomial root finding problem
  - Can be solved using Newton’s method
    - Also known as Newton-Raphson

- Lucas-Kanade method does a single iteration of Newton’s method
  - Better results are obtained with more iterations
Iterative Refinement

- Iterative Lucas-Kanade Algorithm
  1. Estimate velocity at each pixel by solving Lucas-Kanade equations
  2. Warp H towards I using the estimated flow field
     - use image warping techniques
  3. Repeat until convergence
Revisiting the small motion assumption

• Is this motion small enough?
  ▫ Probably not—it’s much larger than one pixel (2\textsuperscript{nd} order terms dominate)
  ▫ How might we solve this problem?
Reduce the resolution!
Coarse-to-fine optical flow estimation

Gaussian pyramid of image H

Gaussian pyramid of image I

\(u=10\) pixels

\(u=5\) pixels

\(u=2.5\) pixels

\(u=1.25\) pixels
Coarse-to-fine optical flow estimation

Gaussian pyramid of image H

Gaussian pyramid of image I

run iterative L-K

warp & upsample

run iterative L-K

run iterative L-K
Optical flow result

Dewey morph
Robust methods

- L-K minimizes a sum-of-squares error metric
  - least squares techniques overly sensitive to outliers

Error metrics

- quadratic: \( \rho(x) = x^2 \)
- truncated quadratic:
  \[
  \rho_{\alpha,\lambda}(x) = \begin{cases} 
  \lambda x^2 / \alpha & \text{if } |x| < \frac{\sqrt{\alpha}}{\sqrt{\lambda}} \\
  \alpha & \text{otherwise}
  \end{cases}
  \]
- lorentzian: \( \rho_\sigma(x) = \log \left( 1 + \frac{1}{2} \left( \frac{x}{\sigma} \right)^2 \right) \)
Robust optical flow

- Robust Horn & Schunk
  \[ \int \int \rho(I_t + \nabla I \cdot [u \ v]) + \lambda^2 \rho(||\nabla u||^2 + ||\nabla v||^2) \, dx \, dy \]

- Robust Lucas-Kanade
  \[ \sum_{(x,y) \in W} \rho(I_t + \nabla I \cdot [u \ v]) \]


Benchmarking optical flow algorithms

- Middlebury flow page
  - [http://vision.middlebury.edu/flow/](http://vision.middlebury.edu/flow/)
Flow quality evaluation
Flow quality evaluation
Flow quality evaluation

Middlebury flow page

- [http://vision.middlebury.edu/flow/](http://vision.middlebury.edu/flow/)

Ground Truth

Color encoding of flow vectors
Flow quality evaluation

Middlebury flow page

- [http://vision.middlebury.edu/flow/](http://vision.middlebury.edu/flow/)

![Lucas-Kanade flow](image1)

![Ground Truth](image2)

Color encoding of flow vectors
Flow quality evaluation

Middlebury flow page

- [http://vision.middlebury.edu/flow/](http://vision.middlebury.edu/flow/)

Best-in-class alg (as of 2/26/12)

Ground Truth

Color encoding of flow vectors
Discussion: features vs. flow?

- Features are better for:

- Flow is better for:
Advanced topics

- Particles: combining features and flow
  - Peter Sand et al.

- State-of-the-art feature tracking/SLAM
  - Georg Klein et al.
  - [http://www.robots.ox.ac.uk/~gk/](http://www.robots.ox.ac.uk/~gk/)