EE795: Computer Vision and Intelligent Systems

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

Lecture 11
140311

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

Outline

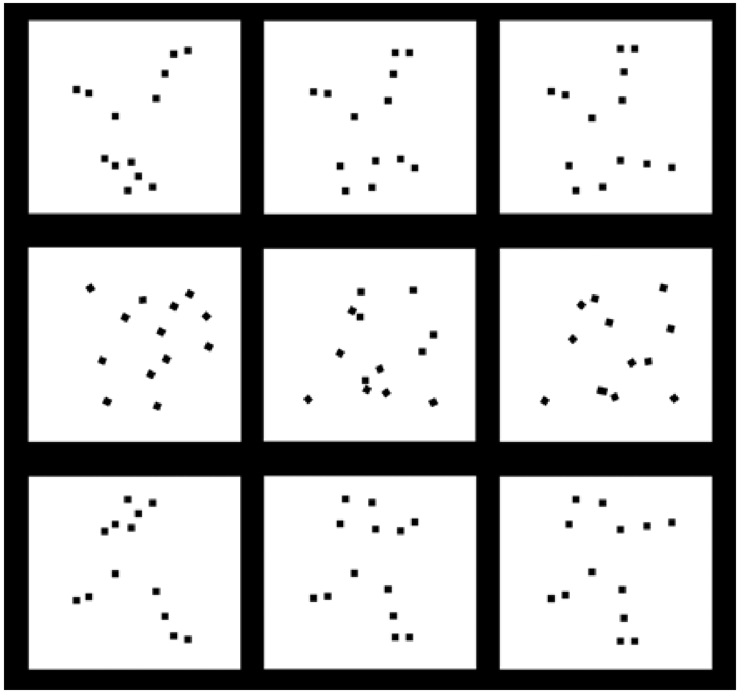
- Motion Analysis Motivation
- Differential Motion
- Optical Flow

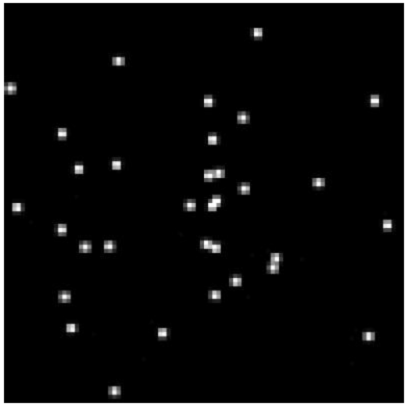
Dense Motion Estimation

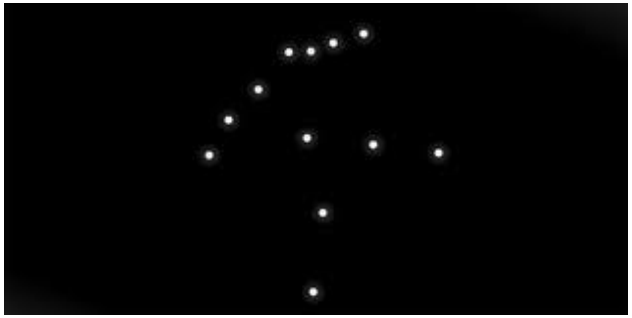
- Motion is extremely important in vision
- Biologically: motion indicates what is food and when to run away
  - We have evolved to be very sensitive to motion cues (peripheral vision)
- Alignment of images and motion estimation is widely used in computer vision
  - Optical flow
  - Motion compensation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

Biological Motion

- Even limited motion information is perceptually meaningful







- http://www.biomotionlab.ca/Demos/BMLwalker.html

Motion Estimation

- Input: sequence of images
- Output: point correspondence
- Prior knowledge: decrease problem complexity
  - E.g. camera motion (static or mobile), time interval between images, etc.

- Motion detection
  - Simple problem to recognize any motion (security)
- Moving object detection and location
  - Feature correspondence: "Feature Tracking"
    - We will see more of this when we examine SIFT
  - Pixel (dense) correspondence: "Optical Flow"

Dynamic Image Analysis

- Motion description
  - Motion/velocity field – velocity vector associated with corresponding keypoints
  - Optical flow – dense correspondence that requires small time distance between images
- Motion assumptions
  - Maximum velocity – object must be located in an circle defined by max velocity
  - Small acceleration – limited acceleration
  - Common motion – all object points move similarly
  - Mutual correspondence – rigid objects with stable points

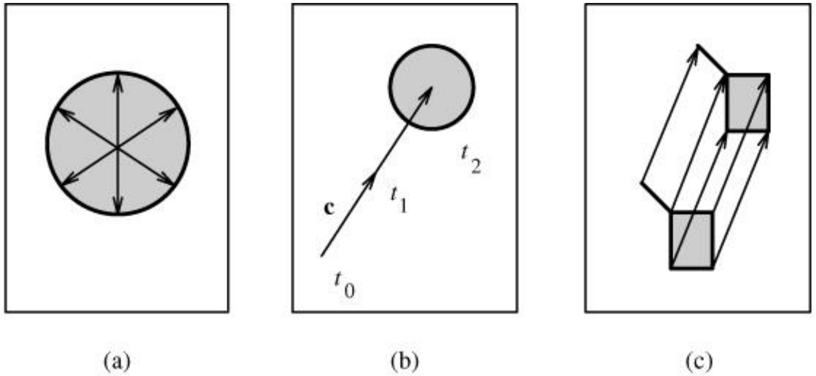


Figure 16.1: Object motion assumptions. (a) Maximum velocity (shaded circle represents area of possible object location). (b) Small acceleration (shaded circle represents area of possible object location at time t_2). (c) Common motion and mutual correspondence (rigid objects).

General Motion Analysis and Tracking

- Two interrelated components:

- Localization and representation of object of interest (target)
  - Bottom-up process: deal with appearance, orientation, illumination, scale, etc.
- Trajectory filtering and data association
  - Top-down process: consider object dynamics to infer motion (motion models)

Differential Motion Analysis

- Simple motion detection possible with image subtraction
  - Requires a stationary camera and constant illumination
  - Also known as change detection
- Difference image
  - $d(i,j) = \begin{cases} 1 & |f_1(i,j) - f_2(i,j)| < \epsilon \\ 0 & else \end{cases}$
  - Binary image that highlights moving pixels
- What are the various "detections" from this method?
  - See book

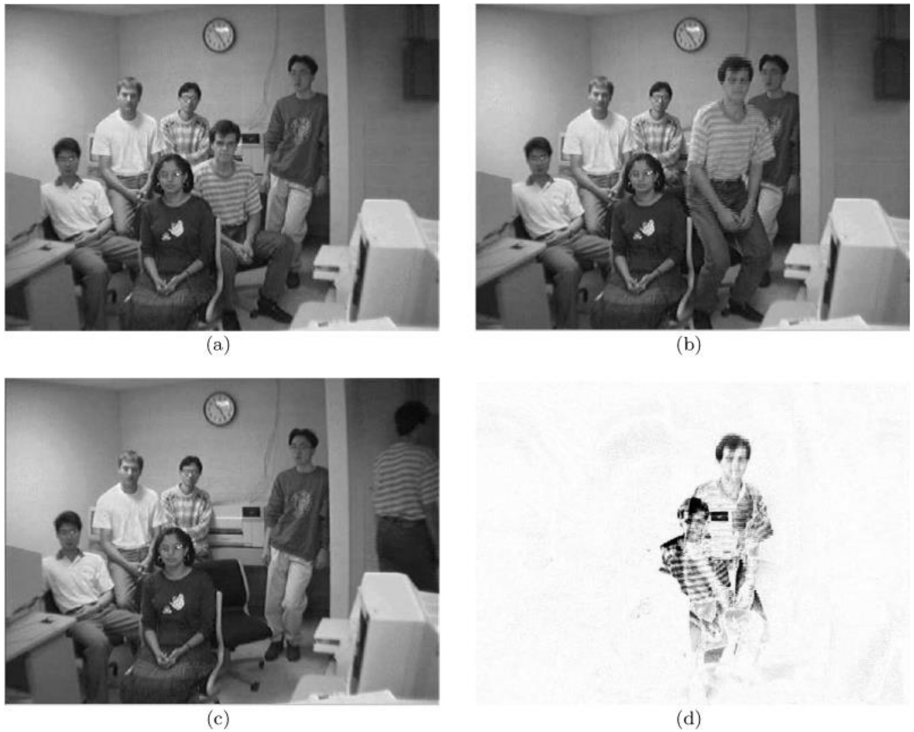


Figure 16.2: Motion detection. (a) First frame of the image sequence. (b) Frame 2 of the sequence. (c) Last frame (frame 5). (d) Differential motion image constructed from image frames 1 and 2 (inverted to improve visualization). © *M. Sonka 2015.*

Background Subtraction

- Motion is an important
  - Indicates an object of interest

- Background subtraction
  - Given an image (usually a video frame), identify the **foreground objects** in that image
    - Assume that foreground objects are moving
    - Typically, moving objects more interesting than the scene
    - Simplifies processing – less processing cost and less room for error

Background Subtraction Example

- Often used in traffic monitoring applications
  - Vehicles are objects of interest (counting vehicles)

⟹

- Human action recognition (run, walk, jump, …)
- Human-computer interaction ("human as interface")
- Object tracking

Requirements

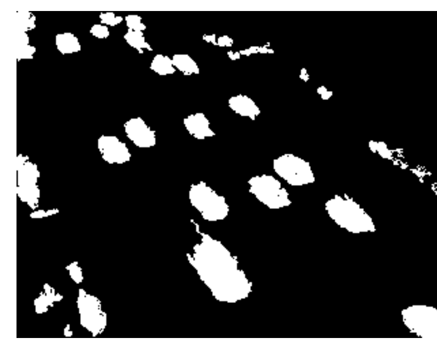
- A reliable and robust background subtraction algorithm should handle:
  - Sudden or gradual illumination changes
    - Light turning on/off, cast shadows through a day
  - High frequency, repetitive motion in the background
    - Tree leaves blowing in the wind, flag, etc.
  - Long-term scene changes
    - A car parks in a parking spot

Basic Approach

- Estimate the background at time t
- Subtract the estimated background from the current input frame
- Apply a threshold, Th, to the absolute difference to get the foreground mask.
  - $|I(x,y,t) - B(x,y,t)| > Th = F(x,y,t)$

$|$ 

$-$ 

$| > Th =$ 

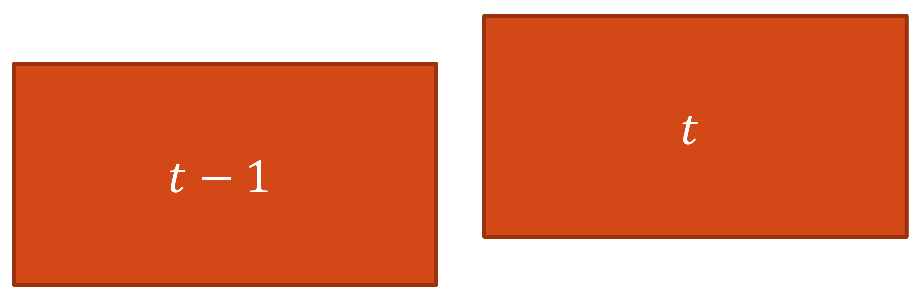
$I(x,y,t)$ $B(x,y,t)$ $F(x,y,t)$

How can we estimate the background?

Frame Differencing

- Background is estimated to be the previous frame
  - $B(x, y, t) = I(x, y, t - 1)$
- Depending on the object structure, speed, frame rate, and global threshold, may or may not be useful
  - Usually not useful – generates impartial objects and ghosts

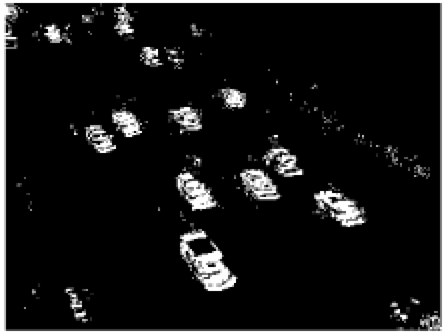
Incomplete object



ghosts

Frame Differencing Example

$Th = 25$



$Th = 50$

$Th = 100$

$Th = 200$

Mean Filter

- Background is the mean of the previous N frames
  - $B(x, y, t) = \frac{1}{N}\sum_{i=0}^{N-1} I(x, y, t - i)$
  - Produces a background that is a temporal smoothing or “blur”
- $N = 10$

Estimated Background

Foreground Mask

Mean Filter

- $N = 20$

Estimated Background

Foreground Mask

- $N = 50$

Estimated Background

Foreground Mask

Median Filter

- Assume the background is more likely to appear than foreground objects
  - $B(x, y, t) = median\big(I(x, y, t - i)\big),\ i \in \{0, N - 1\}$

- $N = 10$

Estimated Background

Foreground Mask

Median Filter

- $N = 20$

Estimated Background

Foreground Mask

- $N = 50$

Estimated Background

Foreground Mask

Frame Difference Advantages

- Extremely easy to implement and use
- All the described variants are pretty fast
- The background models are not constant
  - Background changes over time

Frame Differencing Shortcomings

- Accuracy depends on object speed/frame rate
- Mean and median require large memory
  - Can use a running average
  - $B(x, y, t) = (1 - \alpha)B(x, y, t - 1) + \alpha I(x, y, t)$
    - α – is the learning rate
- Use of a global threshold
  - Same for all pixels and does not change with time
  - Will give poor results when the:
    - Background is bimodal
    - Scene has many slow moving objects (mean, median)
    - Objects are fast and low frame rate (frame diff)
    - Lighting conditions change with time

Improving Background Subtraction

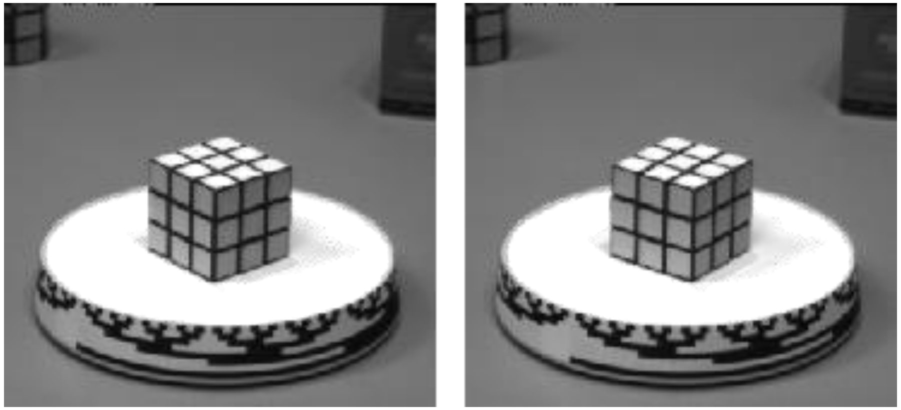
- Adaptive Background Mixture Models for Real-Time Tracking
  - Chris Stauffer and W.E.L. Grimson

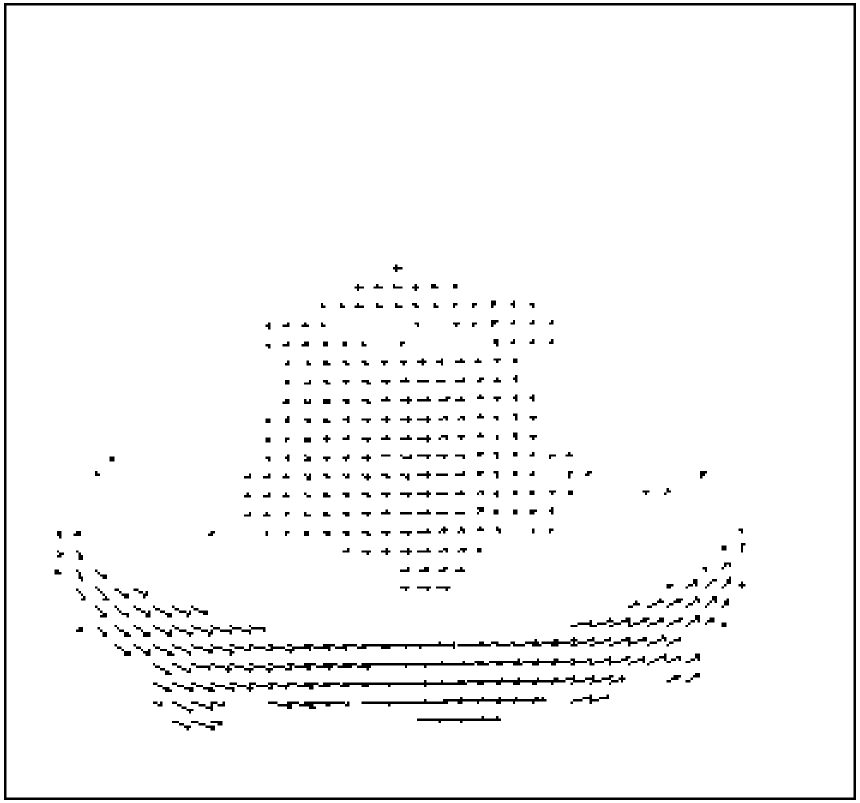
- “The” paper on background subtraction
  - Over 4000 citations since 1999

  - Will read this and see more next time

Optical flow

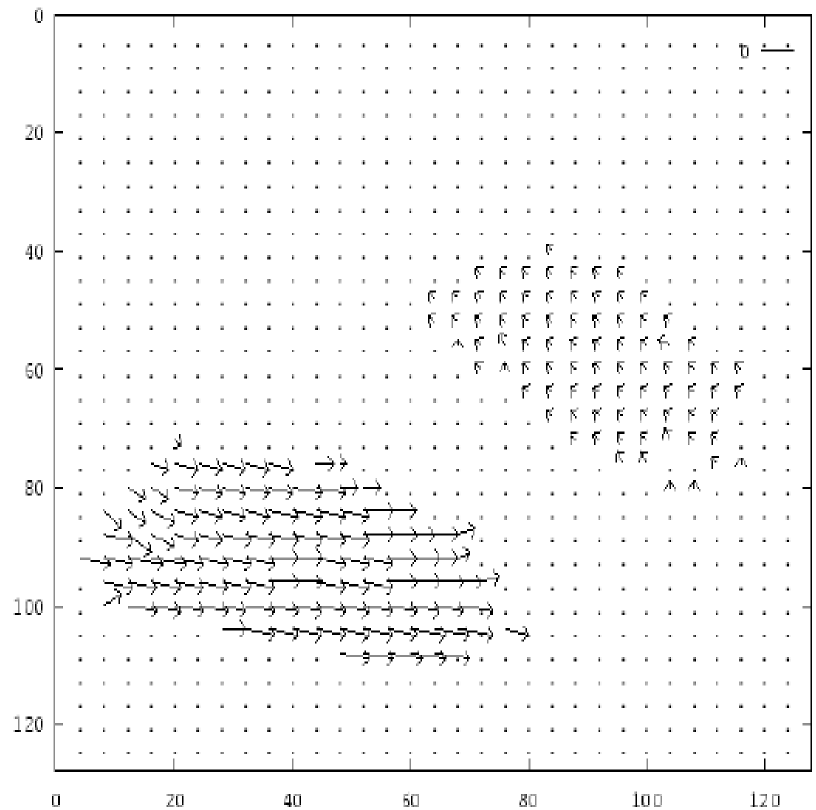
- Dense pixel correspondence





Optical Flow

- Dense pixel correspondence
  - Hamburg Taxi Sequence



Translational Alignment

- Motion estimation between images requires a 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 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, 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

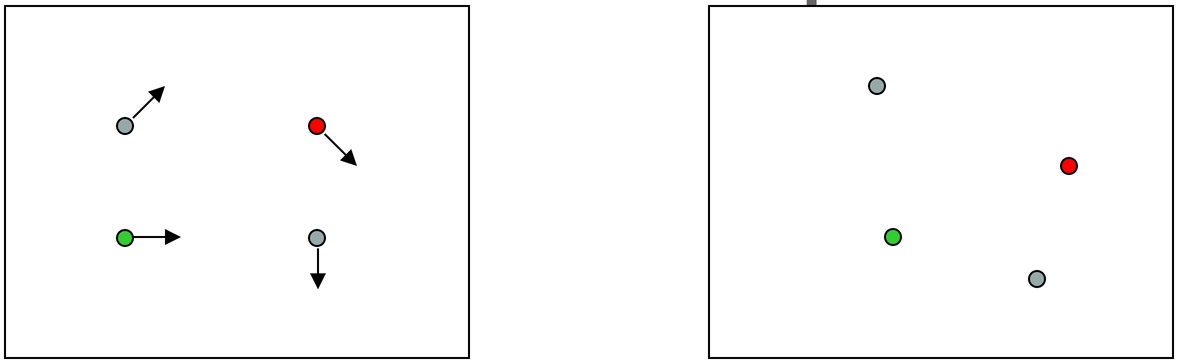
- Instead of minimizing pixel differences, maximize correlation
- Normalized cross-correlation

$$E_{\text{NCC}}(\boldsymbol{u}) = \frac{\sum_i [I_0(\boldsymbol{x}_i) - \overline{I_0}]\,[I_1(\boldsymbol{x}_i + \boldsymbol{u}) - \overline{I_1}]}{\sqrt{\sum_i [I_0(\boldsymbol{x}_i) - \overline{I_0}]^2}\sqrt{\sum_i [I_1(\boldsymbol{x}_i + \boldsymbol{u}) - \overline{I_1}]^2}},$$

$$\begin{aligned} \overline{I_0} &= \frac{1}{N}\sum_i I_0(\boldsymbol{x}_i) \quad \text{and} \\ \overline{I_1} &= \frac{1}{N}\sum_i I_1(\boldsymbol{x}_i + \boldsymbol{u}) \end{aligned}$$

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

Problem definition: optical flow



$H(x, y)$ $I(x, y)$

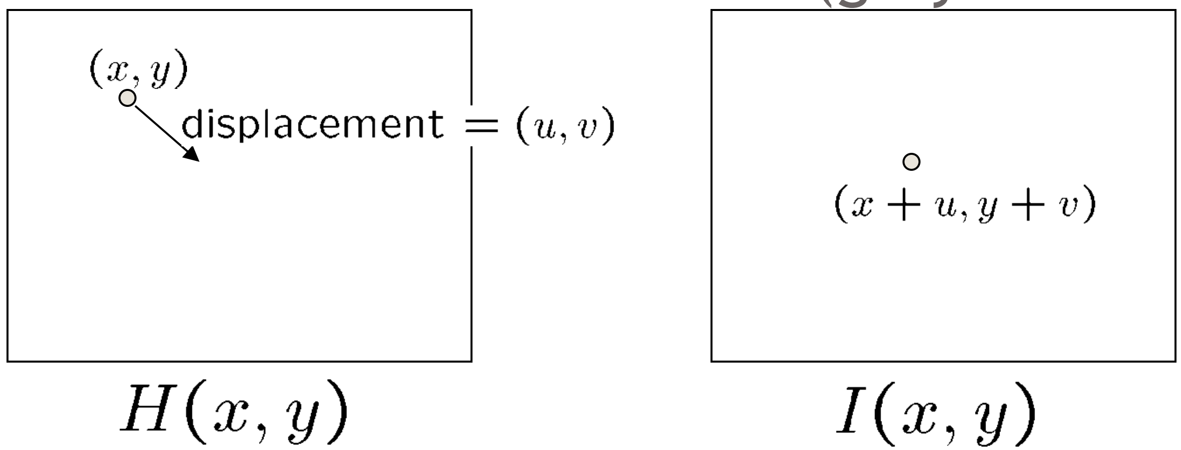
- How to estimate pixel motion from image H to image I?
  - Solve pixel correspondence problem
    - given a pixel in H, look for nearby pixels of the same color in I

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

shorthand: $I_x = \frac{\partial I}{\partial x}$

$$
\begin{aligned}
0 &= I(x+u, y+v) - H(x,y) \\
&\approx I(x,y) + I_x u + I_y v - H(x,y) \\
&\approx (I(x,y) - H(x,y)) + I_x u + I_y v \\
&\approx I_t + I_x u + I_y v \\
&\approx I_t + \nabla I \cdot [u \;\; v]
\end{aligned}
$$

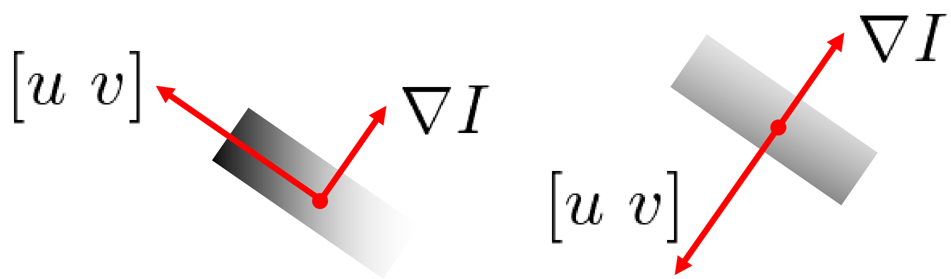
In the limit as u and v go to zero, this becomes exact

$$0 = I_t + \nabla I \cdot [\frac{\partial x}{\partial t} \;\; \frac{\partial y}{\partial t}]$$

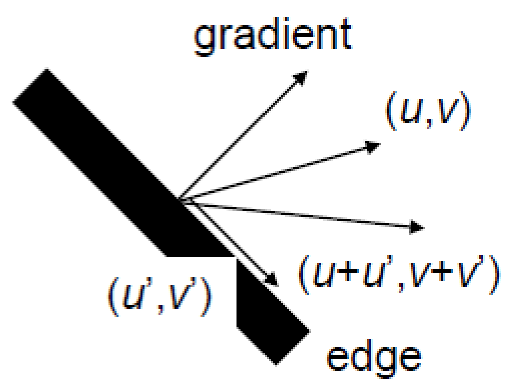
Optical flow equation

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

- Q: how many unknowns and equations per pixel?
  - u and v are unknown - 1 equation, 2 unknowns



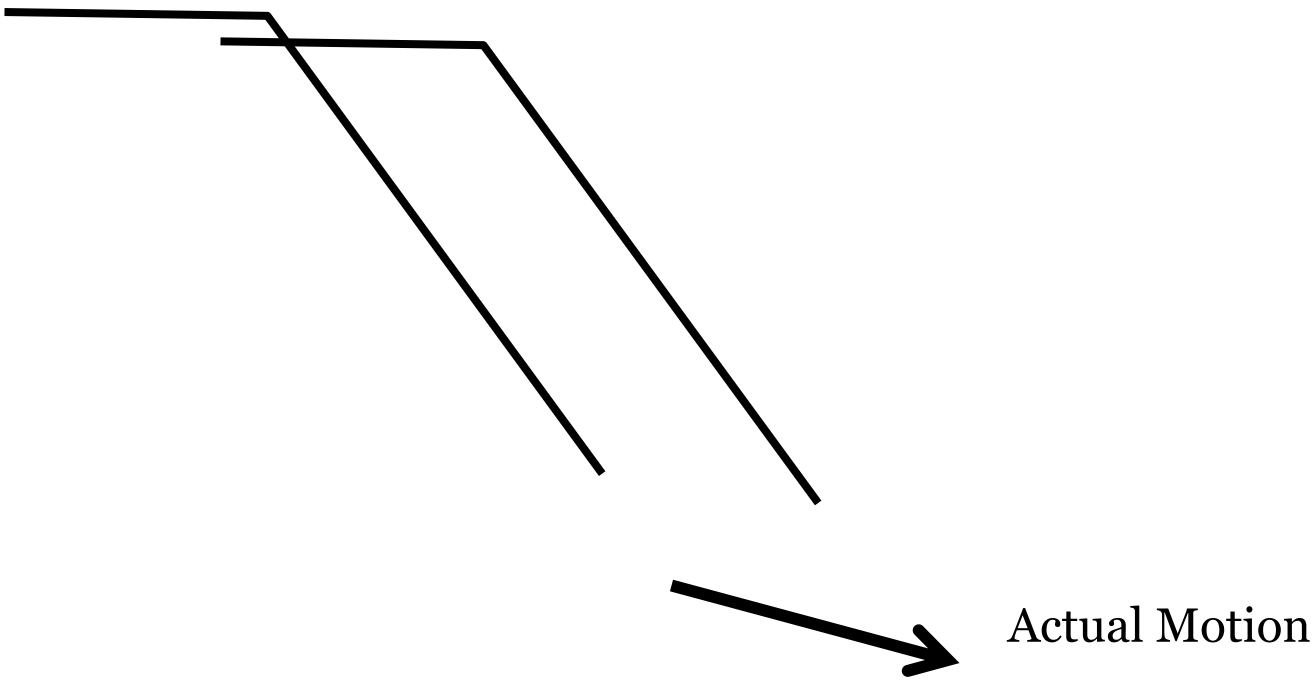
- Intuitively, what does this constraint mean?
  - The component of the flow in the gradient direction is determined
  - The component of the flow parallel to an edge is unknown
- This explains the Barber Pole illusion
  - http://www.sandlotscience.com/Ambiguous/Barberpole_Illusion.htm



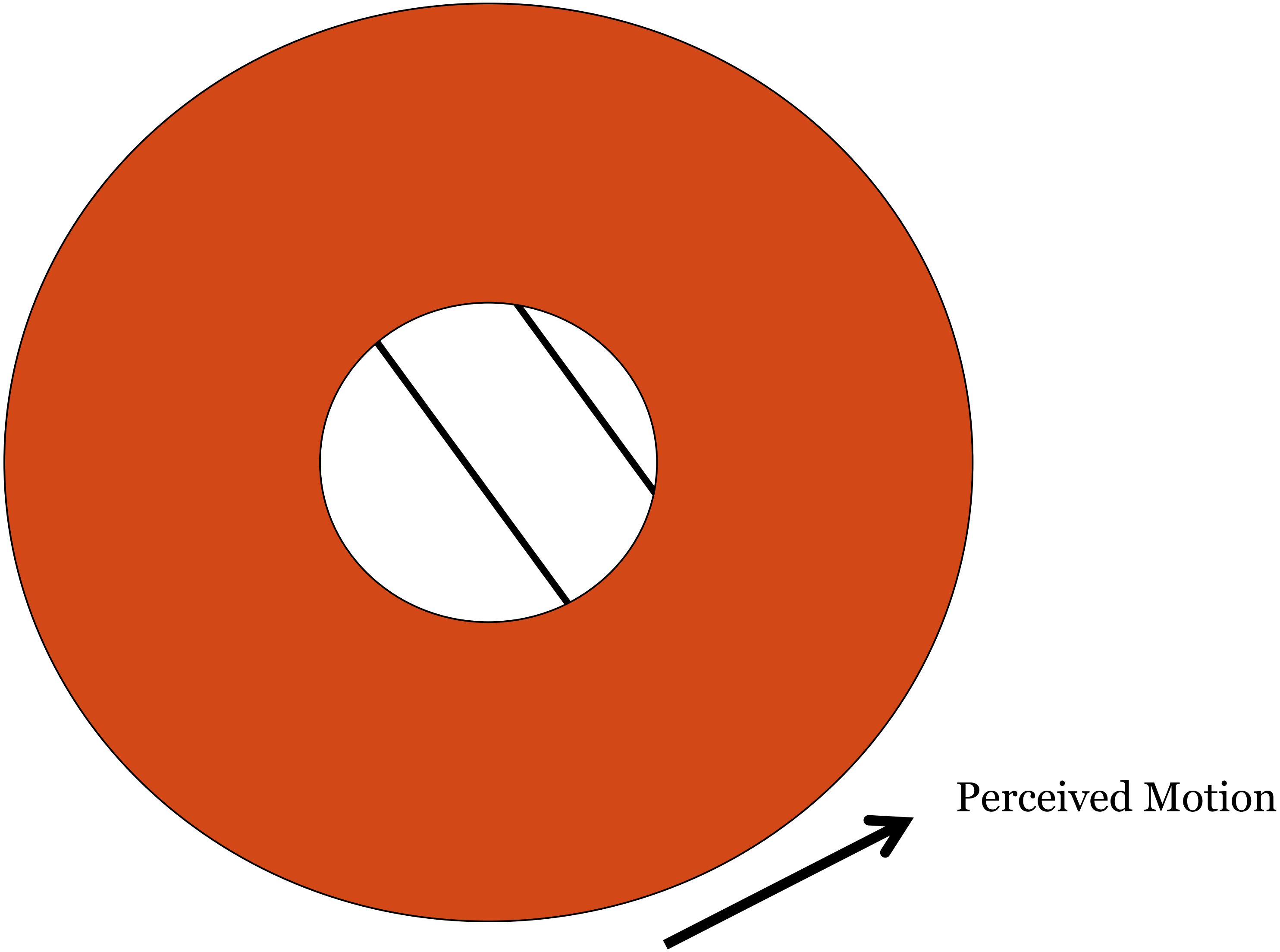
If (u, v) satisfies the equation, so does $(u + u', v + v')$ if

$$\nabla I \cdot [u' \; v'] = 0$$

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:
  - S. Baker, M. Black, J. P. Lewis, S. Roth, D. Scharstein, and R. Szeliski. *A database and evaluation methodology for optical flow*. In Proc. ICCV, 2007
  - 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!

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

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

RGB version

- 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!

$$0 = I_t(\mathbf{p_i})[0,1,2] + \nabla I(\mathbf{p_i})[0,1,2] \cdot [u \;\; v]$$

$$\underbrace{\begin{bmatrix} I_x(\mathbf{p_1})[0] & I_y(\mathbf{p_1})[0] \\ I_x(\mathbf{p_1})[1] & I_y(\mathbf{p_1})[1] \\ I_x(\mathbf{p_1})[2] & I_y(\mathbf{p_1})[2] \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}})[0] & I_y(\mathbf{p_{25}})[0] \\ I_x(\mathbf{p_{25}})[1] & I_y(\mathbf{p_{25}})[1] \\ I_x(\mathbf{p_{25}})[2] & I_y(\mathbf{p_{25}})[2] \end{bmatrix}}_{\substack{A \\ 75\text{x}2}} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\substack{d \\ 2\times1}} = - \underbrace{\begin{bmatrix} I_t(\mathbf{p_1})[0] \\ I_t(\mathbf{p_1})[1] \\ I_t(\mathbf{p_1})[2] \\ \vdots \\ I_t(\mathbf{p_{25}})[0] \\ I_t(\mathbf{p_{25}})[1] \\ I_t(\mathbf{p_{25}})[2] \end{bmatrix}}_{\substack{b \\ 75\text{x}1}}$$

Lucas-Kanade flow

Prob: we have more equations than unknowns

$$\underset{25\times2}{A}\ \underset{2\times1}{d} = \underset{25\times1}{b} \longrightarrow \text{minimize } \|Ad - b\|^2$$

Solution: solve least squares problem

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

$$\underset{2\times2}{(A^T A)}\ \underset{2\times1}{d} = \underset{2\times1}{A^T b}$$

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

- The summations are over all pixels in the K x K window
- This technique was first proposed by Lucas & Kanade (1981)

Conditions for solvability

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

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

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

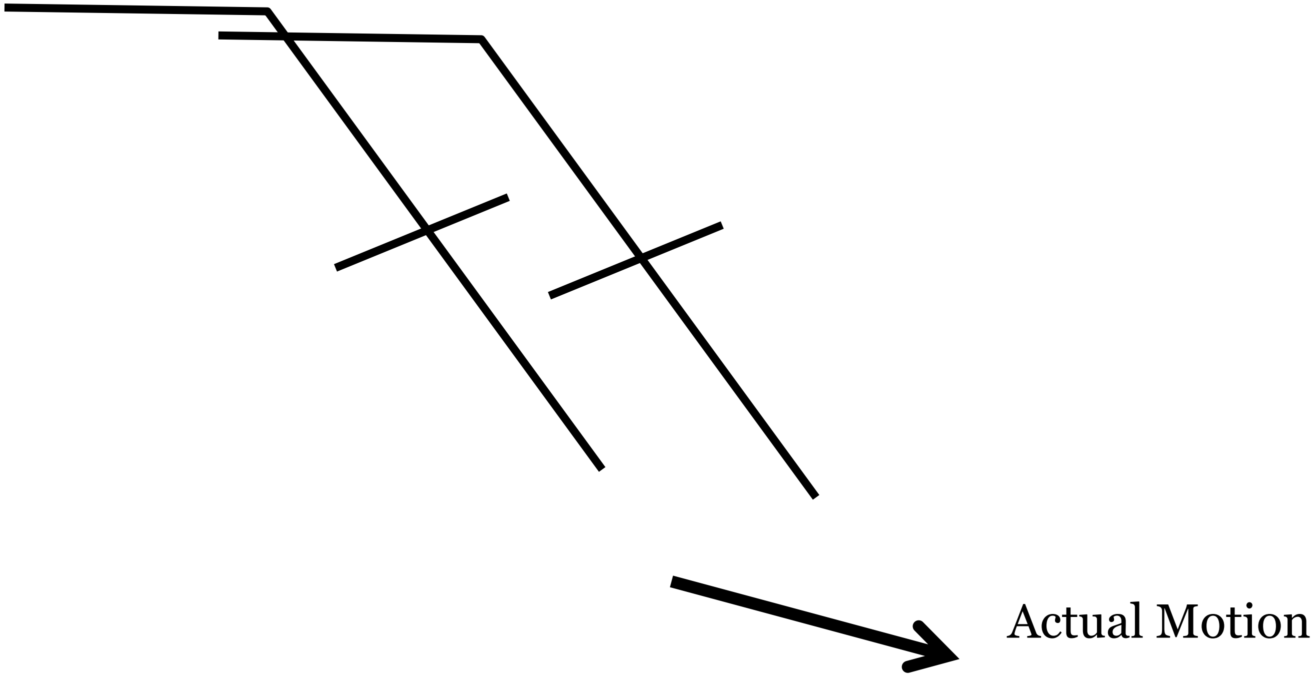
- When is This Solvable?
  - **$A^T A$** should be invertible
  - **$A^T A$** should not be too small due to noise
    - eigenvalues λ_1 and λ_2 of **$A^T A$** should not be too small
  - **$A^T A$** should be well-conditioned
    - λ_1 / λ_2 should not be too large (λ_1 = larger eigenvalue)

- Does this look familiar?
  - **$A^T A$** is the Harris matrix

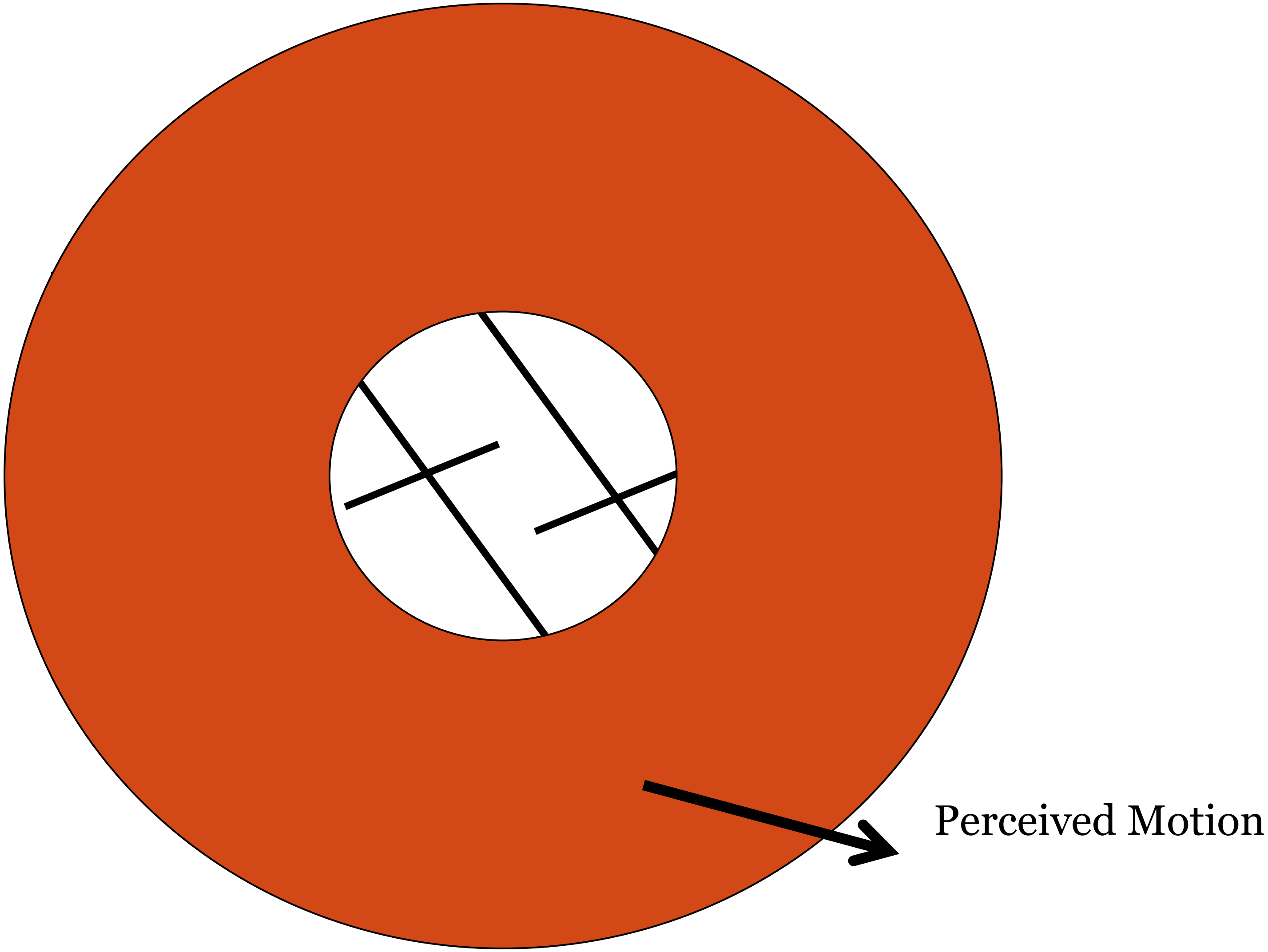
Observation

- 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...

Aperture problem



Aperture problem



Errors in Lucas-Kanade

- What are the potential causes of errors in this procedure?
  - Suppose A^TA 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

$$\begin{aligned} 0 &= I(x+u, y+v) - H(x,y) \\ &\approx I(x,y) + I_x u + I_y v - H(x,y) \end{aligned}$$

- 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

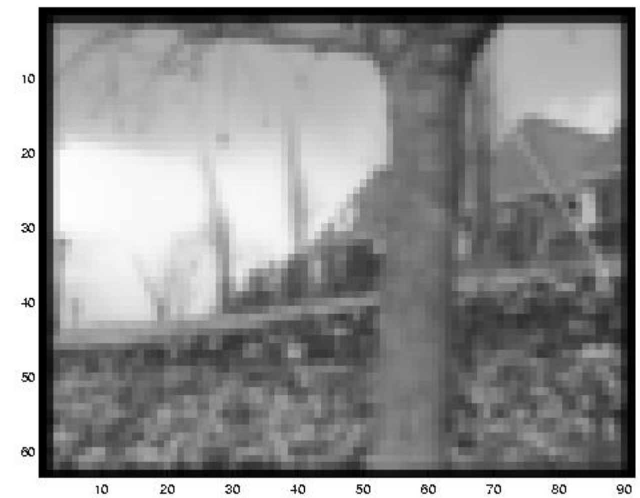
Iterative Refinement

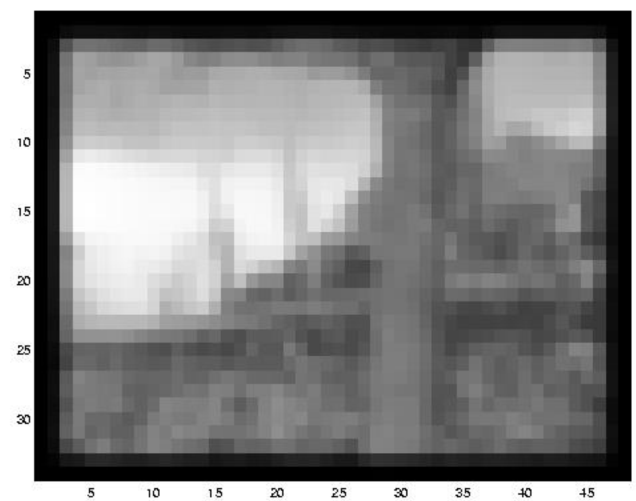
- Iterative Lucas-Kanade Algorithm
  1. Estimate velocity at each pixel by solving Lucas-Kanade equations
  2. Warp H towards I using the estimated flow field
     - *- use image warping techniques*
  3. Repeat until convergence

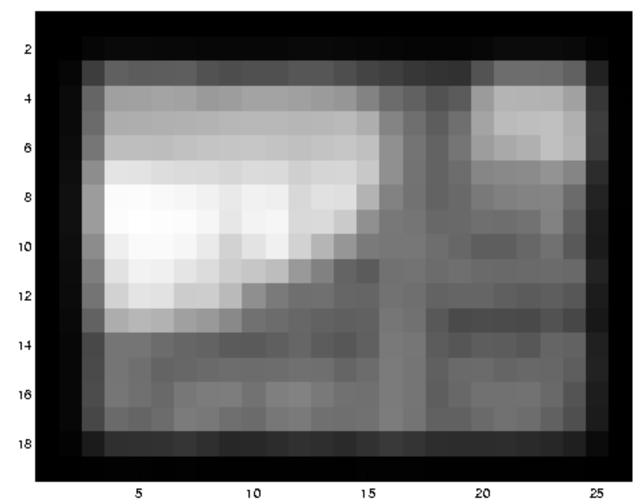
Revisiting the small motion assumption

- Is this motion small enough?
  - Probably not—it's much larger than one pixel (2nd order terms dominate)
  - How might we solve this problem?

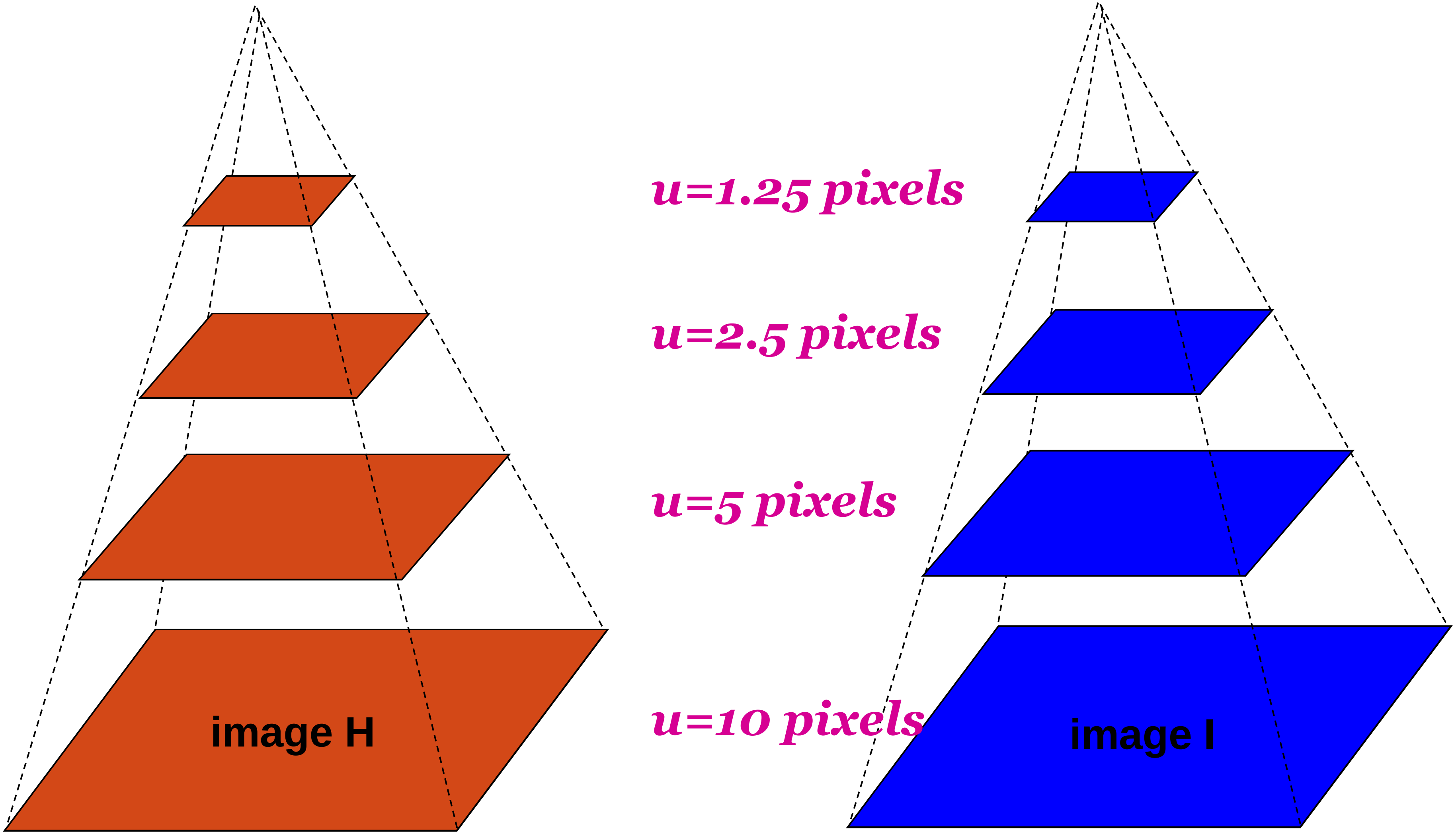
Reduce the resolution!





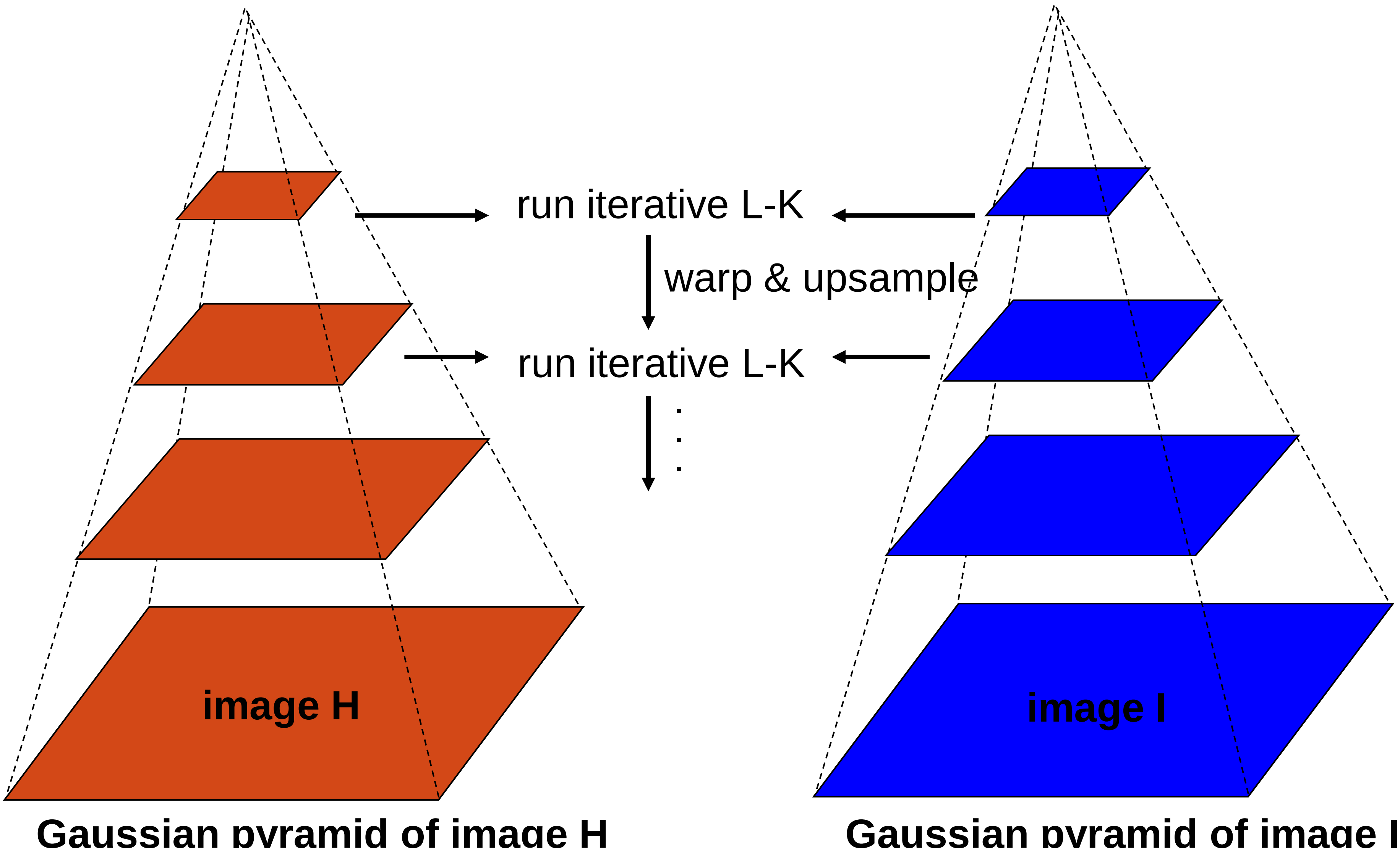


Coarse-to-fine optical flow estimation

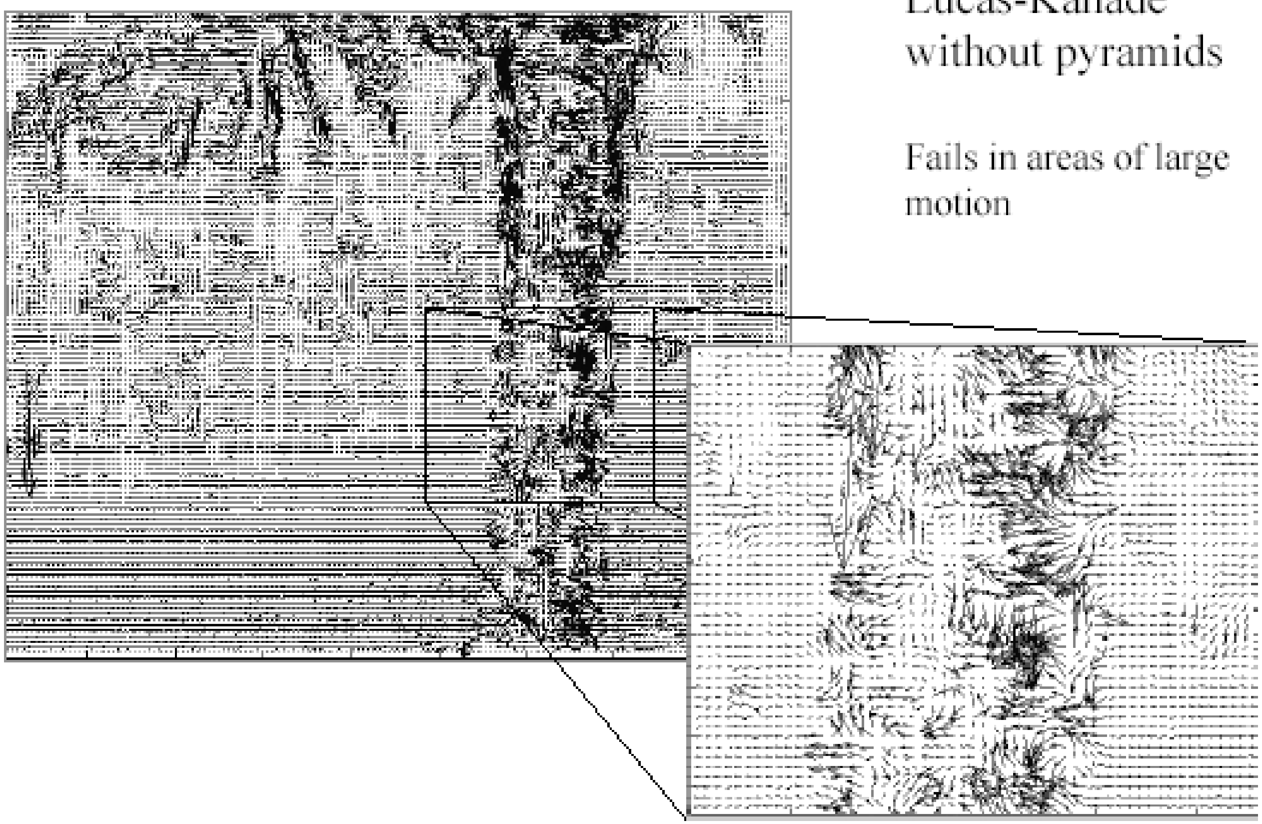


Gaussian pyramid of image H **Gaussian pyramid of image I**

Coarse-to-fine optical flow estimation



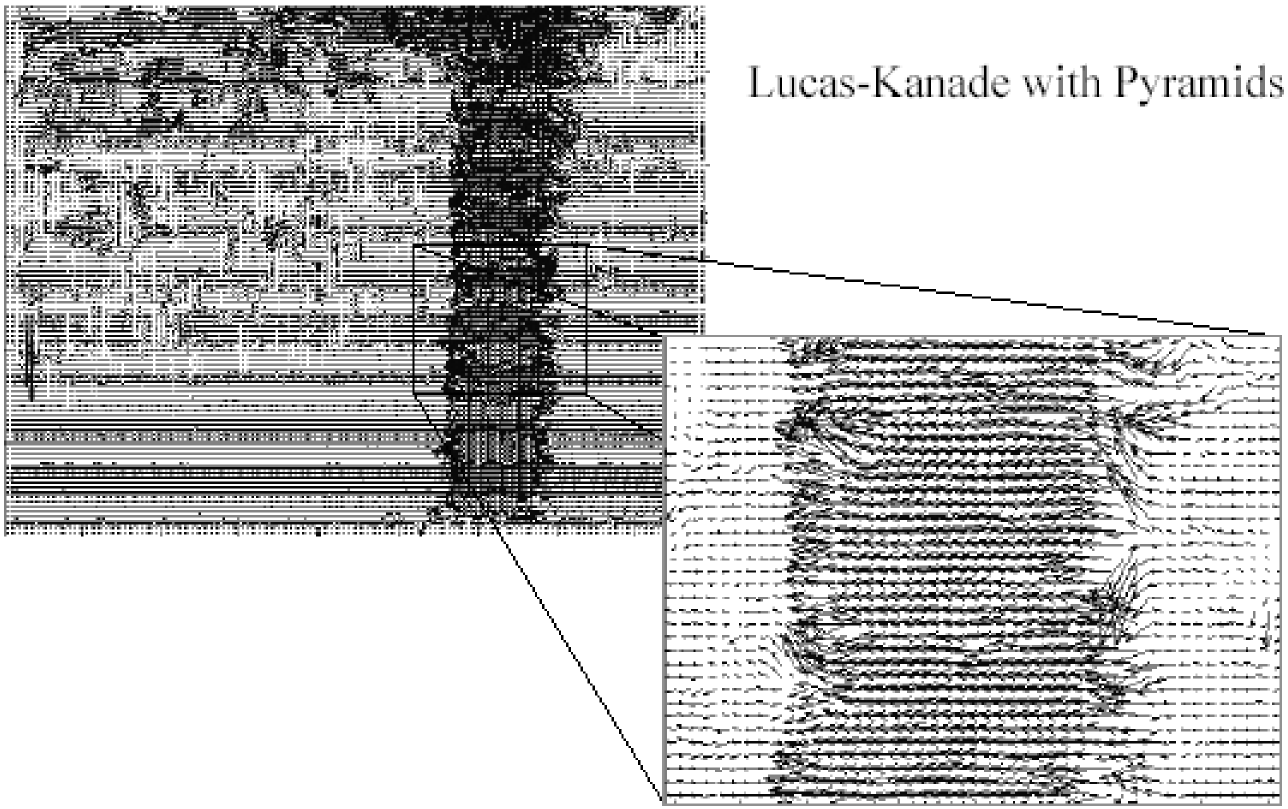
Optical Flow Results



Lucas-Kanade without pyramids

Fails in areas of large motion

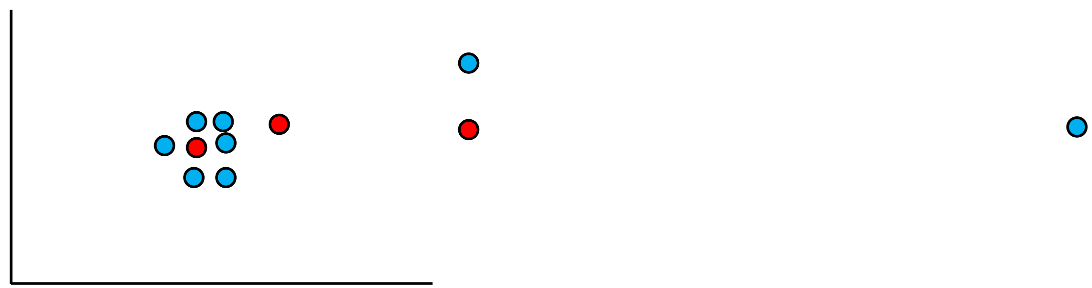
Optical Flow Results



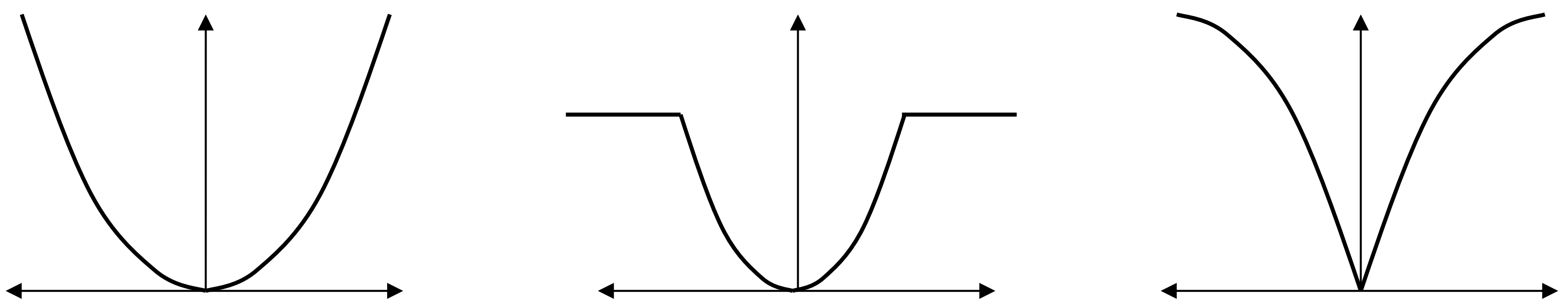
Lucas-Kanade with Pyramids

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

$$\rho_{\alpha,\lambda}(x) = \begin{cases} \lambda x^2 & \text{if } |x| < \frac{\sqrt{\alpha}}{\sqrt{\lambda}} \\ \alpha & \text{otherwise} \end{cases}$$

lorentzian

$$\rho_\sigma(x) = log\left(1 + \frac{1}{2}\left(\frac{x}{\sigma}\right)^2\right)$$

Robust optical flow

- Robust Horn & Schunk

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

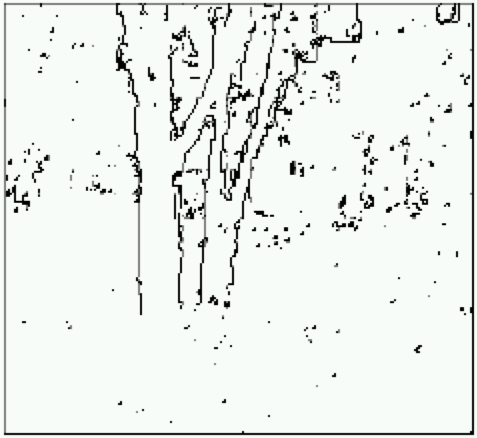
- Robust Lucas-Kanade

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

first image

quadratic flow

lorentzian flow



detected outliers

Black, M. J. and Anandan, P., A framework for the robust estimation of optical flow, *Fourth International Conf. on Computer Vision* (ICCV), 1993, pp. 231-236

http://www.cs.washington.edu/education/courses/576/03sp/readings/black93.pdf

Benchmarking optical flow algorithms

- Middlebury flow page
  - http://vision.middlebury.edu/flow/

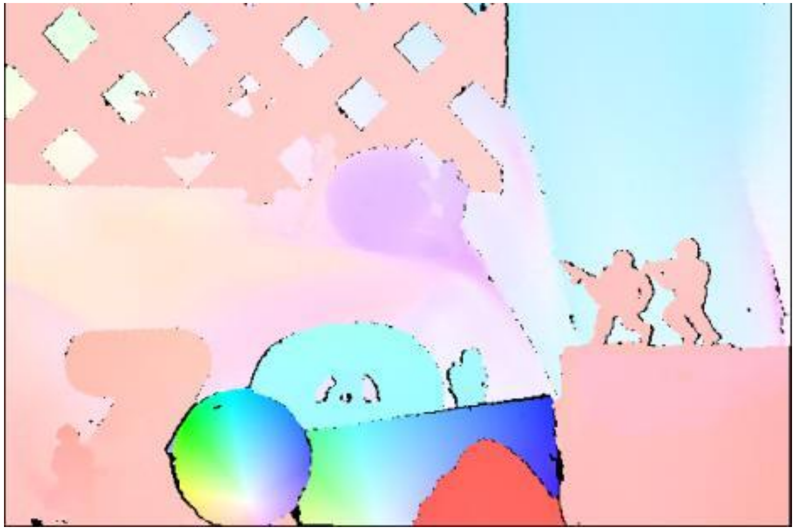
Flow quality evaluation

Flow quality evaluation

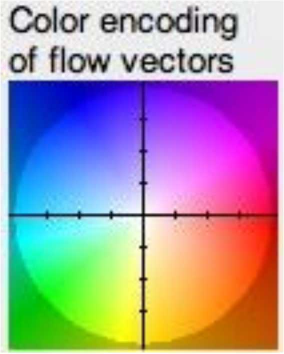
Flow quality evaluation

Middlebury flow page

- http://vision.middlebury.edu/flow/



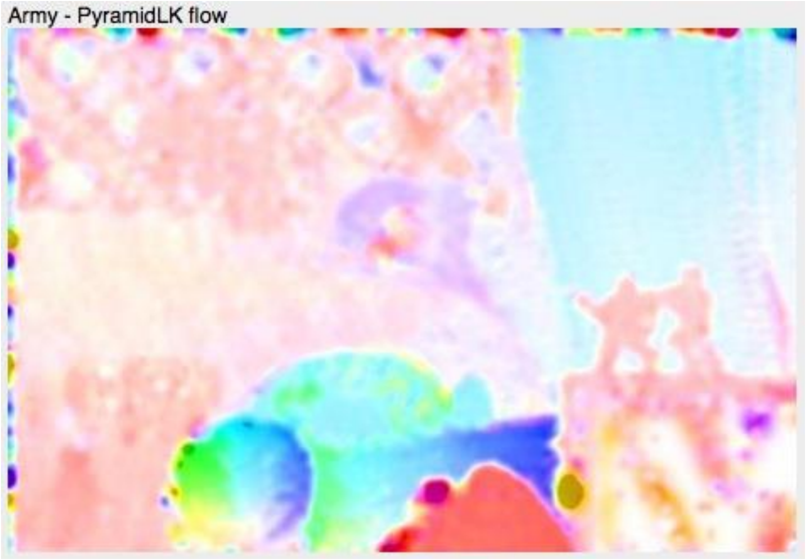
Ground Truth



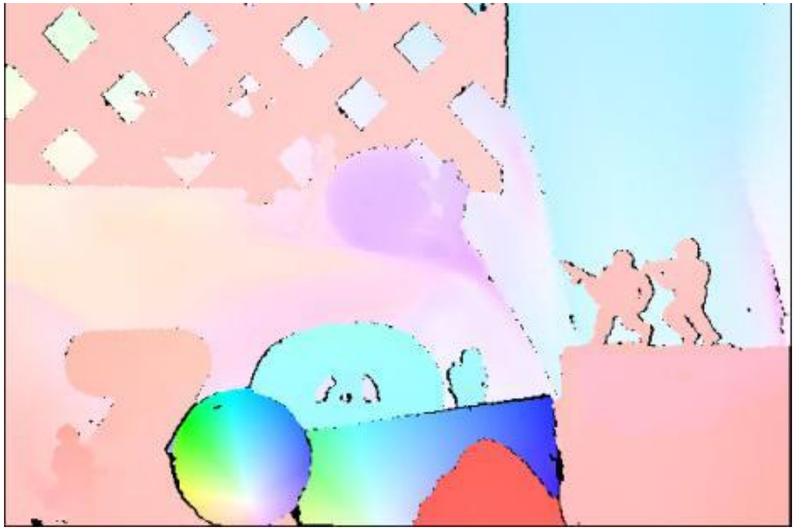
Flow quality evaluation

Middlebury flow page

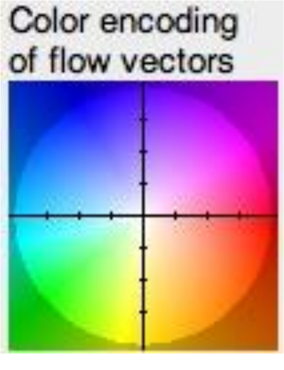
- http://vision.middlebury.edu/flow/



Lucas-Kanade flow



Ground Truth

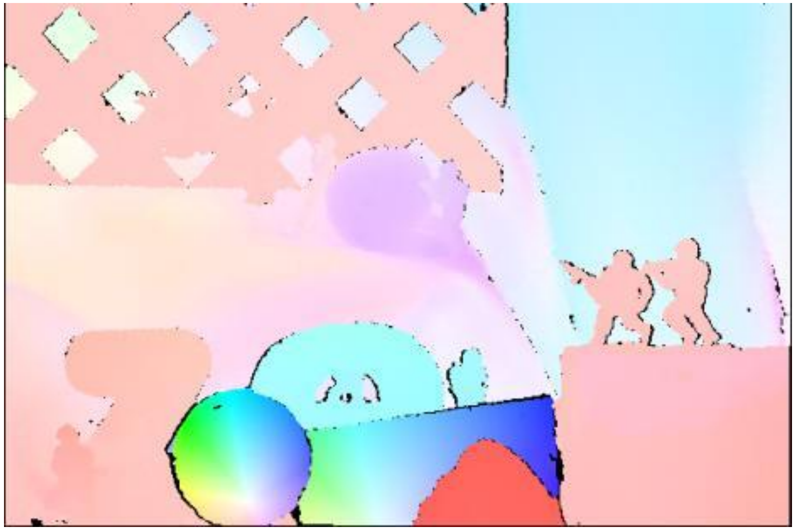


Flow quality evaluation

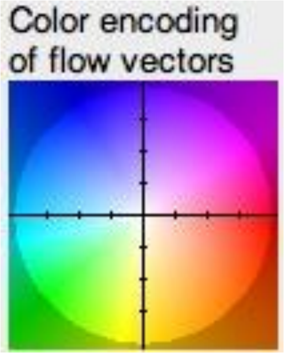
Middlebury flow page

- http://vision.middlebury.edu/flow/

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



Ground Truth



Discussion: features vs. flow?

- Features are better for:

- Flow is better for:

Advanced topics

- Particles: combining features and flow
  - Peter Sand et al.
  - http://rvsn.csail.mit.edu/pv/
- State-of-the-art feature tracking/SLAM
  - Georg Klein et al.
  - http://www.robots.ox.ac.uk/~gk/