SAIR

Spatial AI & Robotics Lab

CSE 473/573-A

W11B: OPTICAL FLOW

Chen Wang

Spatial AI & Robotics Lab

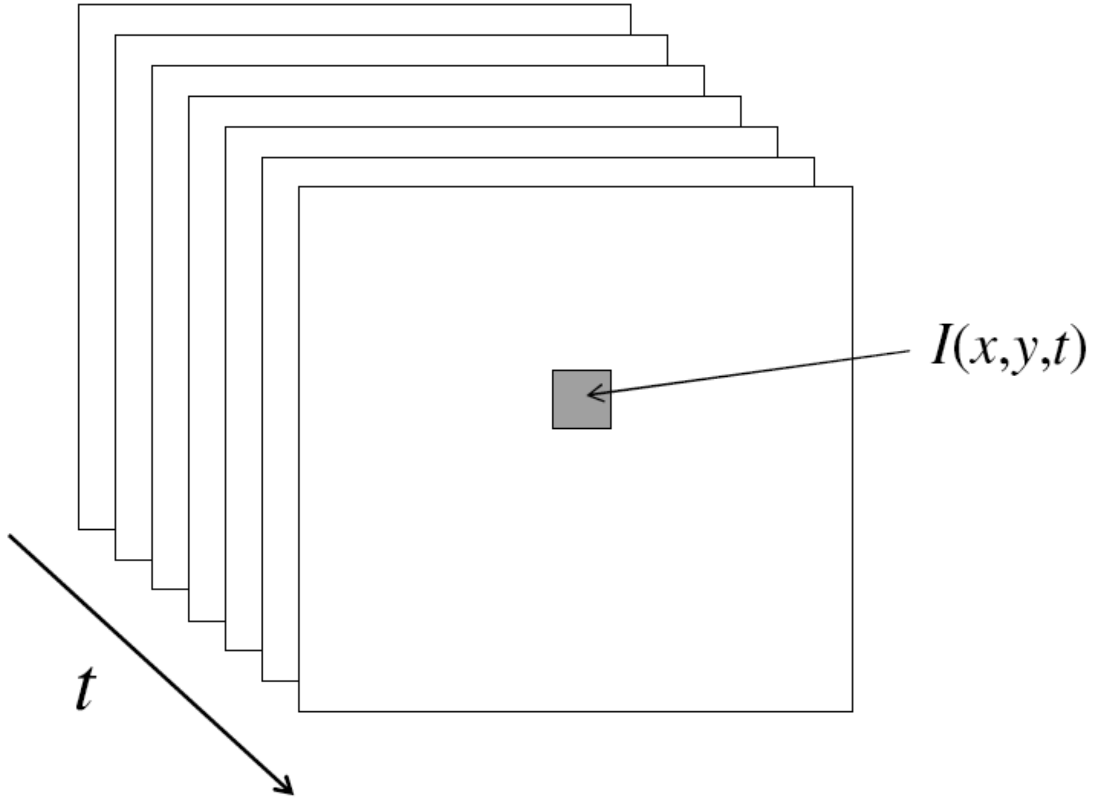
Department of Computer Science and Engineering

University at Buffalo The State University of New York

Many Slides from Lana Lazebnik

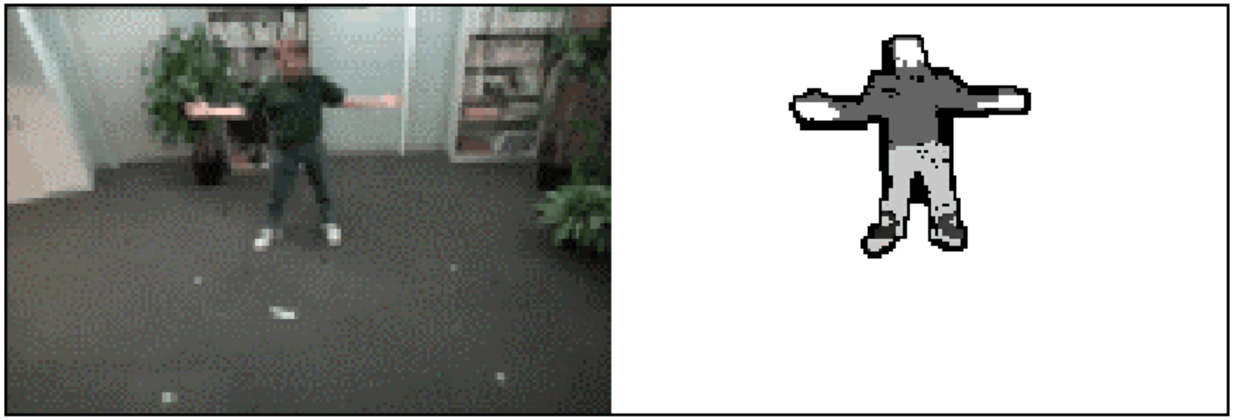
Video

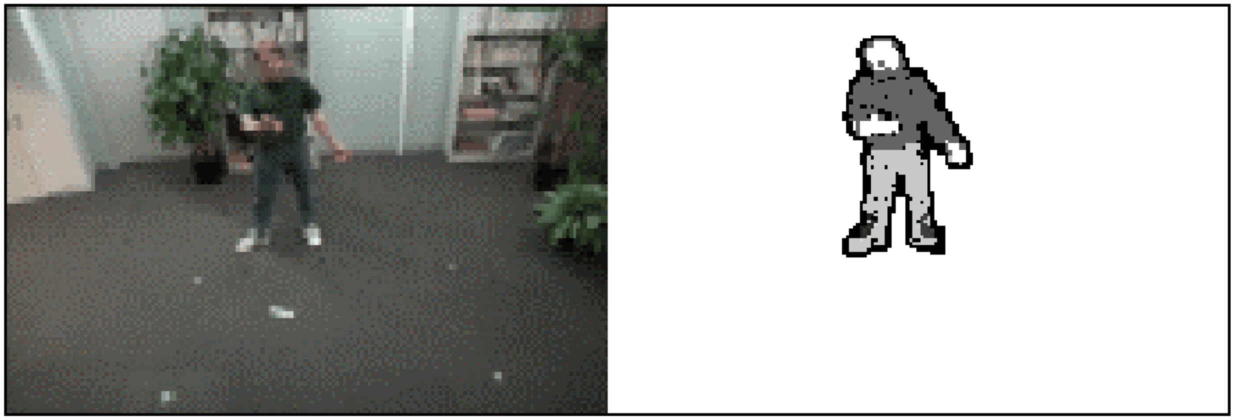
- A video is a sequence of frames captured over time
- Image data is a function of space (x, y) and time (t)



Motion: Background subtraction

- A static camera is observing a scene
- Separate the static *background* from the moving *foreground*



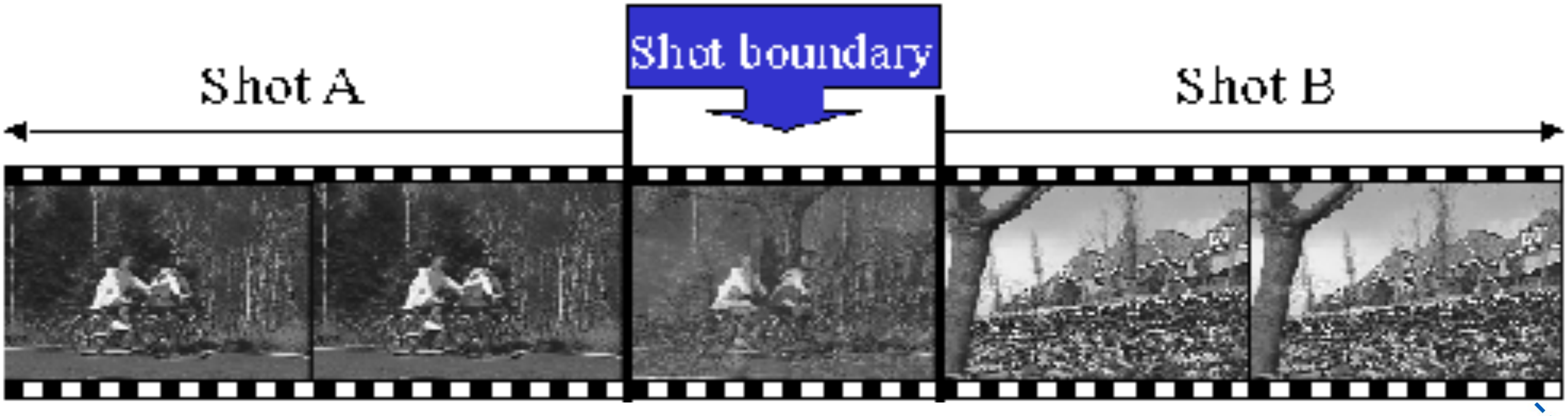


Motion: Background subtraction

- Form an **initial background estimate**
- For each frame:
  - Update estimate using a **moving average**
  - **Subtract** the **background** estimate from the frame
  - Label as foreground where the **magnitude of the difference** is greater than some threshold
  - Use **median filtering** to “clean up” the results
- Challenges?
  - Periodic Motion
  - Camera motion
  - Shadows

Motion: Shot Boundary Detection

- Commercial video is usually composed of *shots* or sequences showing the same objects or scene
- Goal: segment video into shots for summarization and browsing (each shot can be represented by a single key-frame in a user interface)
- Difference from background subtraction
  - The camera is not necessarily stationary



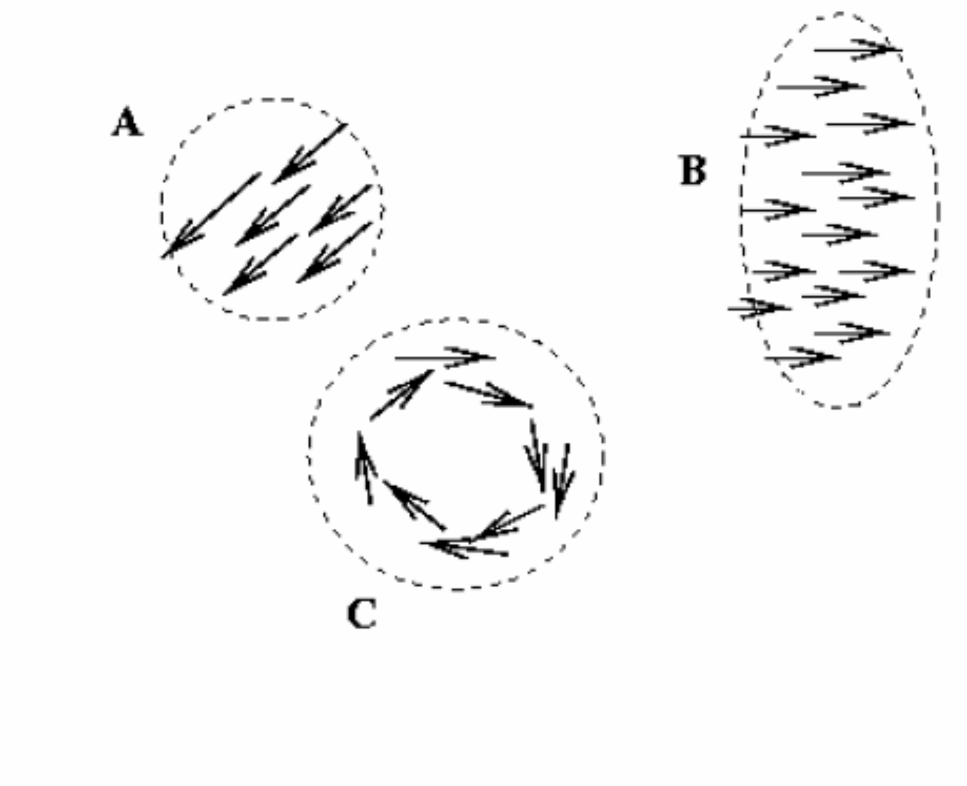
Motion: Shot Boundary Detection

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

- Challenges?
  - Content shift (slow or fast)

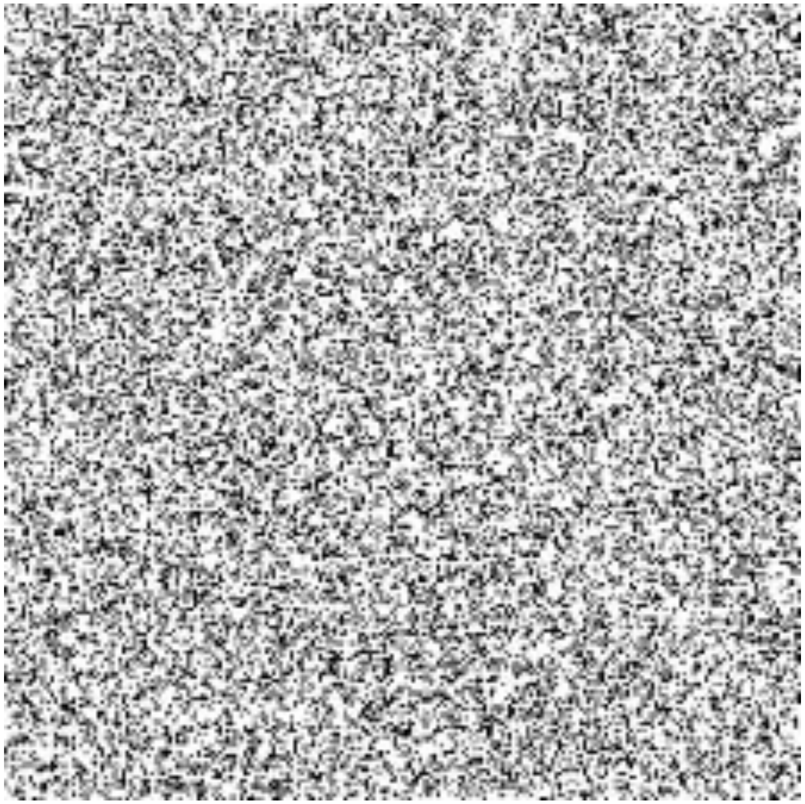
Motion: Motion Segmentation

- Segment video into multiple coherently moving objects



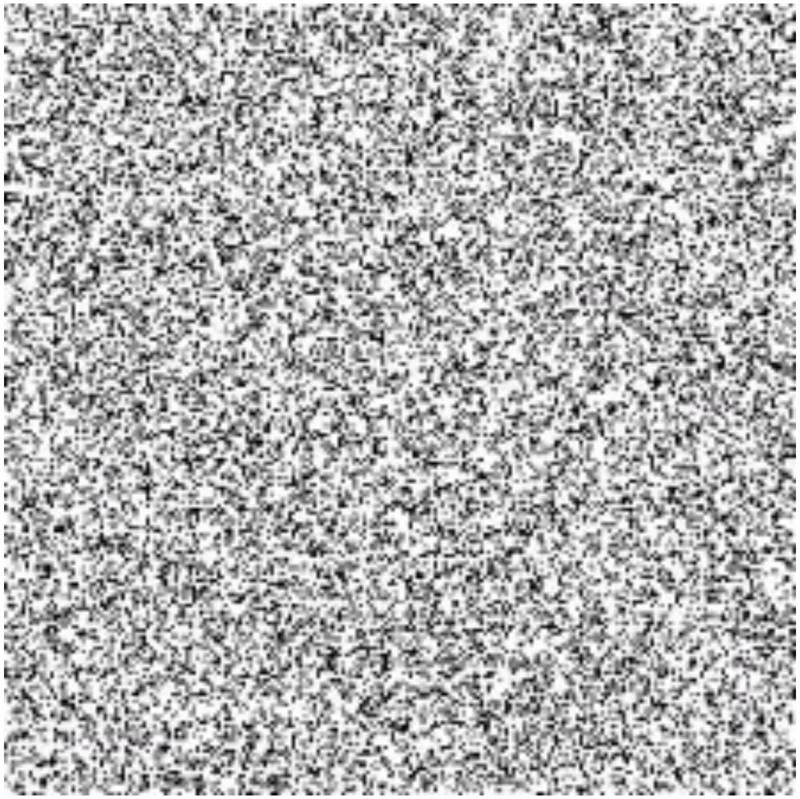
Motion and perceptual organization

- Sometimes, motion is the only cue



Motion and perceptual organization

- Sometimes, motion is the only cue



Motion and perceptual organization

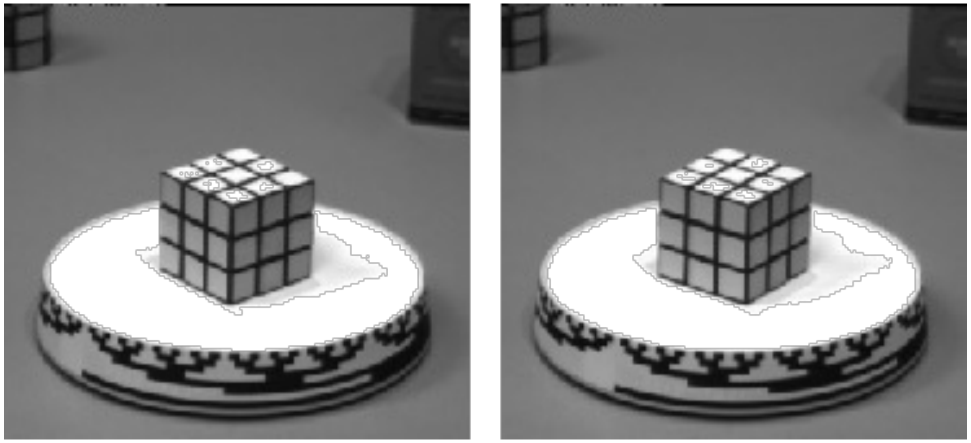
- Even “impoverished” motion data can evoke a strong percept

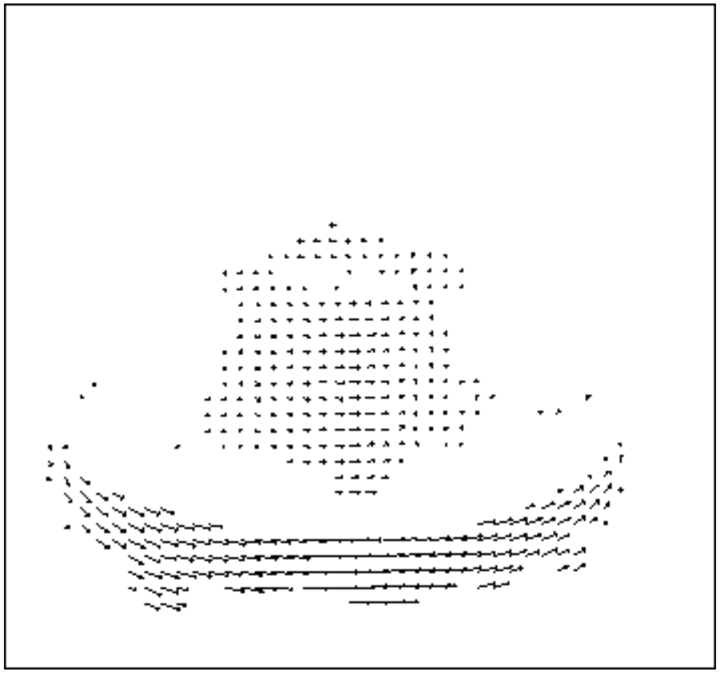
Motion and perceptual organization

- Even “impoverished” motion data can evoke a strong percept

Motion estimation: Optical flow

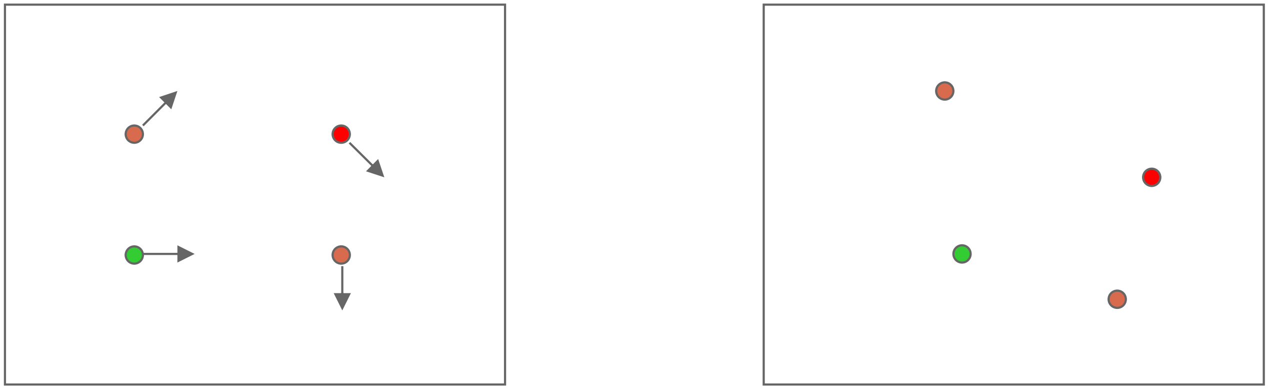
- *Optic flow* is the **apparent** motion of objects or surfaces





We will start by estimating motion of each pixel separately
Then will consider motion of entire image

Problem definition: optical flow



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

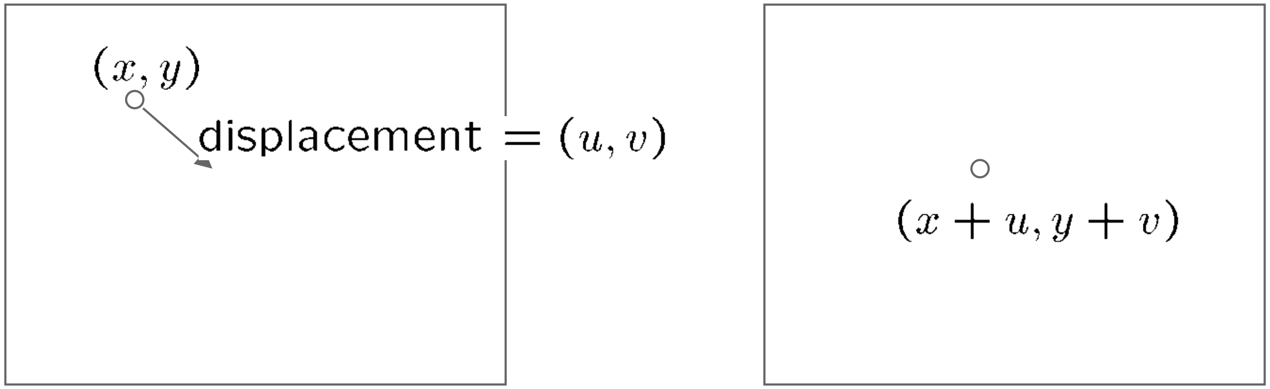
How to estimate pixel motion from $I(x, y, t)$ to $I(x, y, t+1)$

- Solve pixel correspondence problem
  - given a pixel in $I(x, y, t)$, look for nearby pixels of the same color in $I(x, y, t+1)$

Key assumptions

- Small motion: points do not move very far.
- Color constancy: a point in $I(x, y, t)$ looks the same in $I(x, y, t+1)$
  - For grayscale images, this is brightness constancy

Optical flow constraints (grayscale images)



$I(x, y, t)$ $\qquad$ $I(x, y, t+1)$

- Let's look at these constraints more closely
  - Brightness constancy constraint (equation)

$$I(x, y, t) = I(x+u, y+v, t+1)$$

  - Small motion: (u and v are less than 1 pixel, or smooth)
    - Taylor series expansion of I:

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

$$\approx I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

SAIR
Spatial AI & Robotics Lab

Optical flow equation

$$I(x,y,t) = I(x+u, y+v, t+1)$$

$$I(x+u, y+v) = I(x,y) + \frac{\partial I}{\partial x} u + \frac{\partial I}{\partial y} v$$

(Shorthand: $I_x = \frac{\partial I}{\partial x}$, for t **or** $t+1$)

- Combining these two equations

$$\begin{aligned} 0 &= I(x+u, y+v, t+1) - I(x,y,t) \\ &\approx I(x,y,t+1) + I_x u + I_y v - I(x,y,t) \\ &\approx \boxed{[I(x,y,t+1) - I(x,y,t)]} + I_x u + I_y v \\ &\approx \boxed{I_t} + I_x u + I_y v \\ &\approx I_t + \nabla I \cdot < u, v > \end{aligned}$$

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

$$0 = I_t + \nabla I \cdot < u, v >$$

Brightness constancy constraint equation

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

How does this make sense?

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

Brightness constancy constraint equation

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

The brightness constancy constraint

Can we use it to recover image motion (u, v) at each pixel?

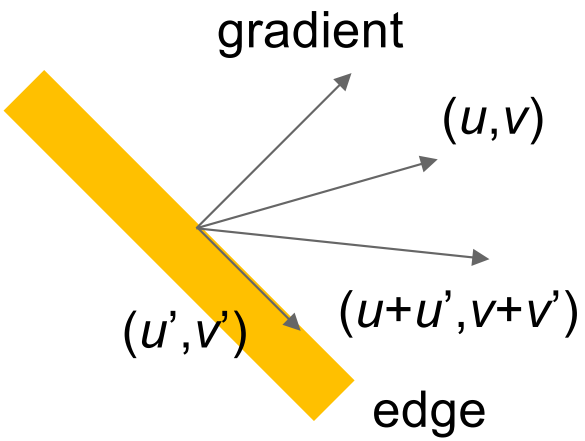
$$0 = I_t + \nabla I \cdot < u, v > \quad \text{or} \quad I_x u + I_y v + I_t = 0$$

- 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

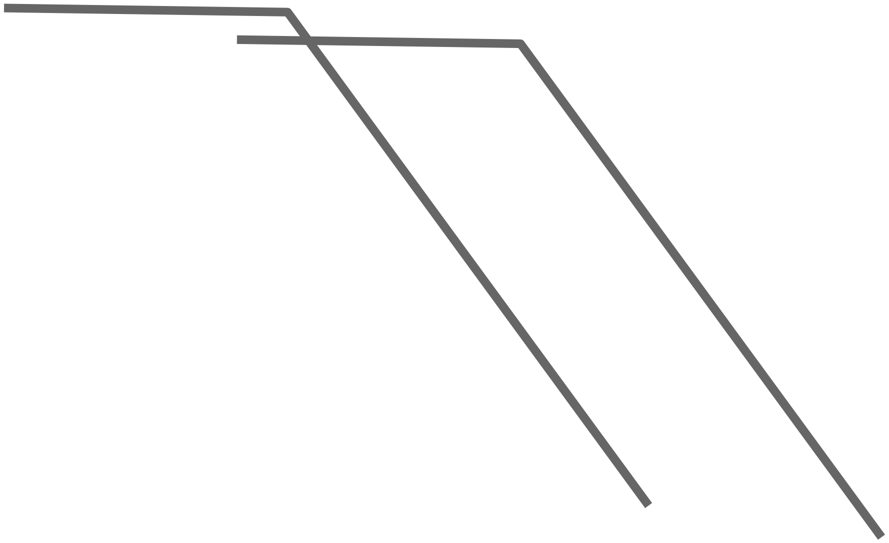
If (u, v) satisfies the equation, so does $(u + u', v + v')$ if

$$\nabla \mathrm{I} \cdot \left[\mathrm{u}' \;\; \mathrm{v}'\right]^{\mathrm{T}} = 0$$

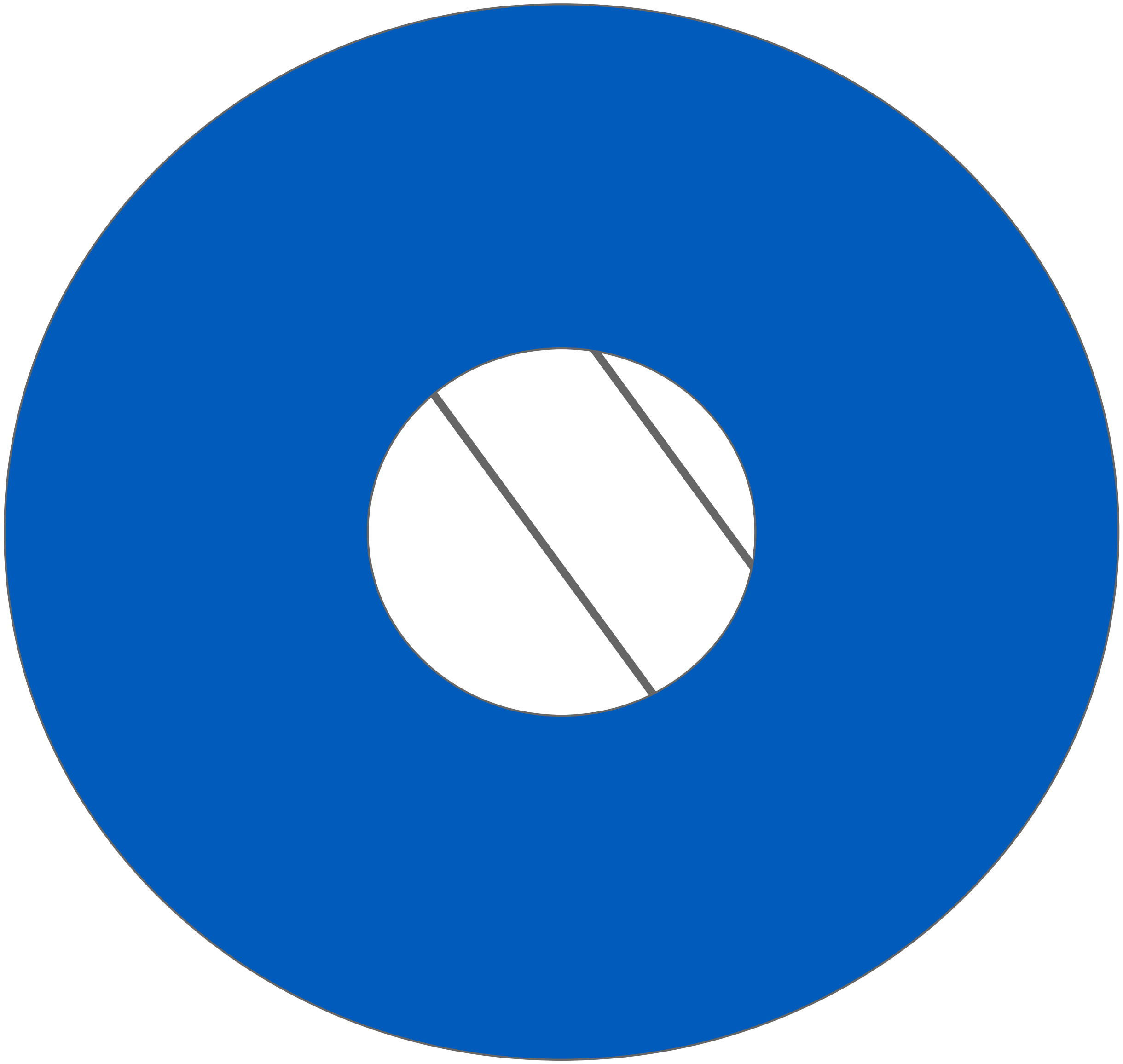


How can we solve this ambiguity?

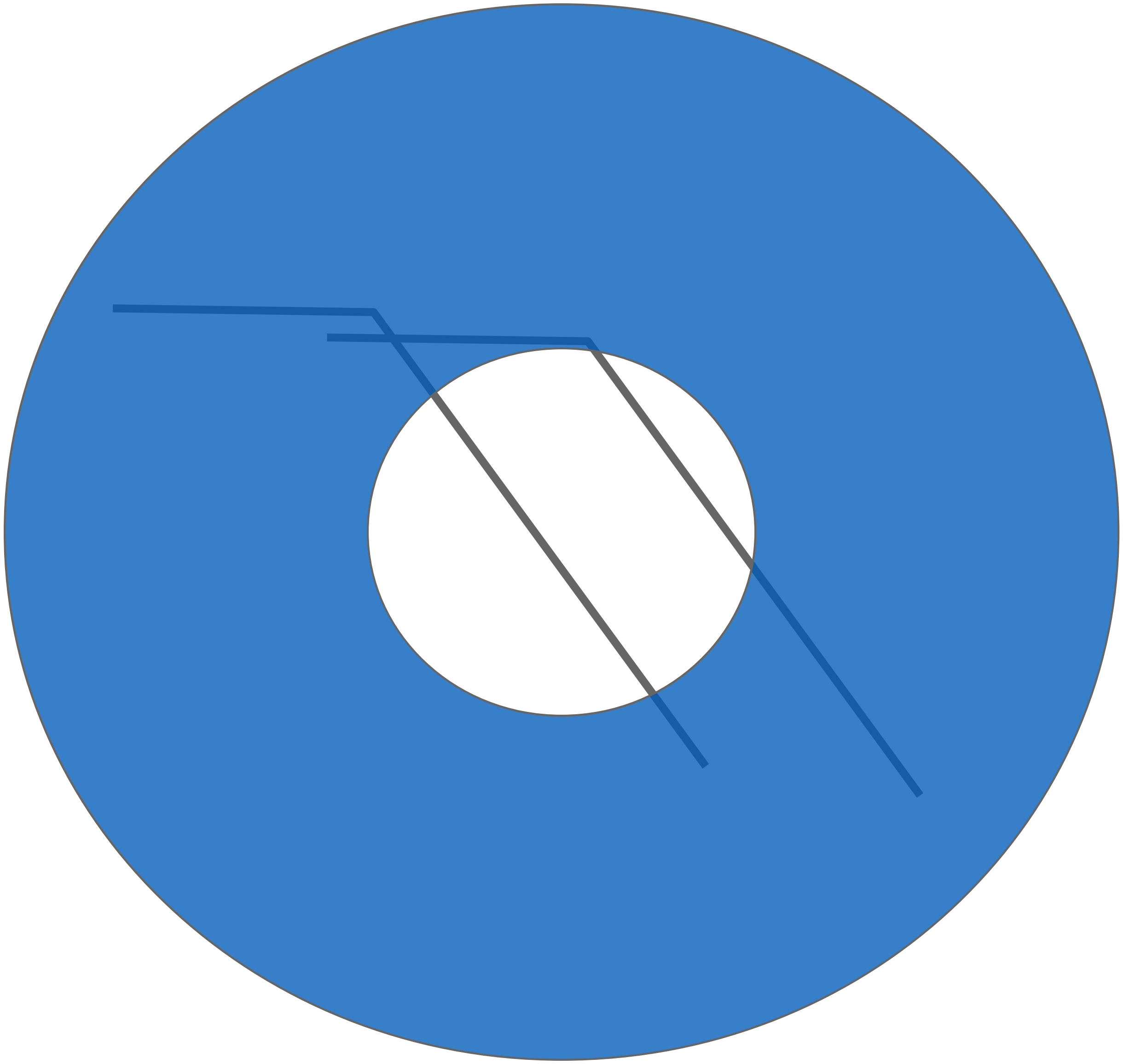
Aperture problem



Aperture problem



Aperture problem



The barber pole illusion

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

The barber pole illusion

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

Lucas-Kanade (LK) Algorithm

- 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 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.

Matching patches across images

- Least squares problem (Overconstrained):

$$\begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix} \qquad \underset{25\text{x}2}{A} \; \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

Least squares solution for d given by $(A^T A)\ d = A^T b$

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

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

Conditions for solvability

Optimal (u, v) satisfies Lucas-Kanade equation

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

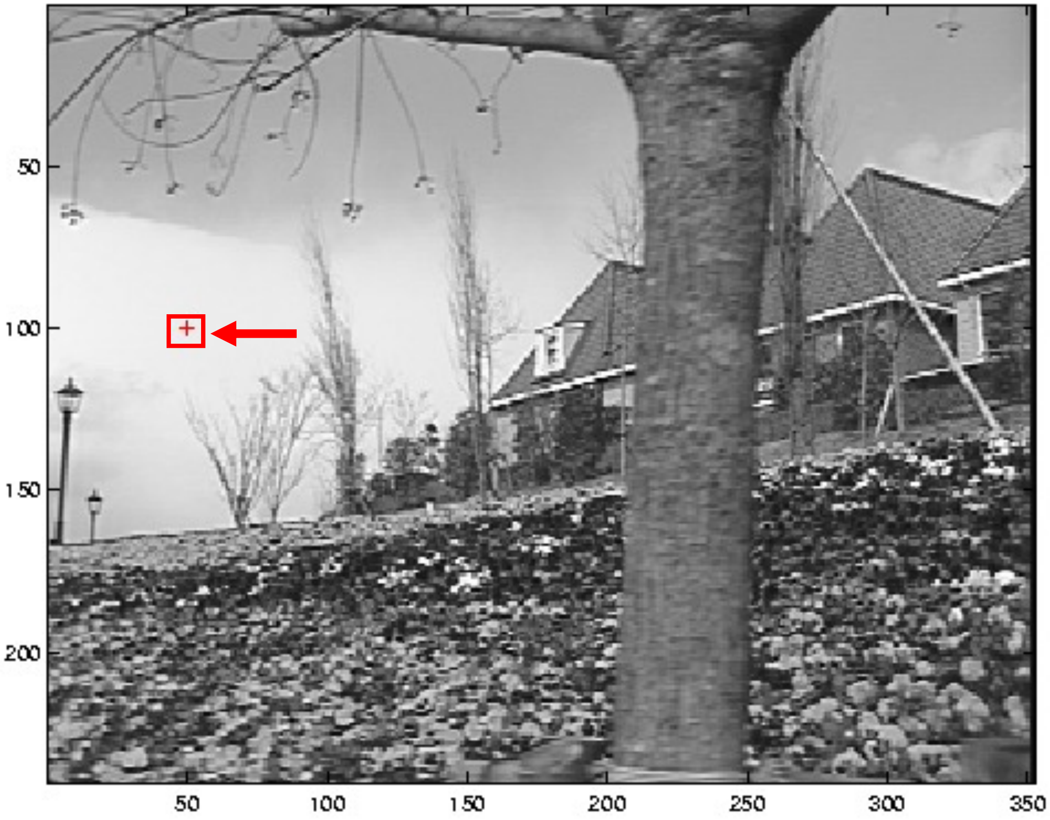
When is this solvable? What are good points to track?

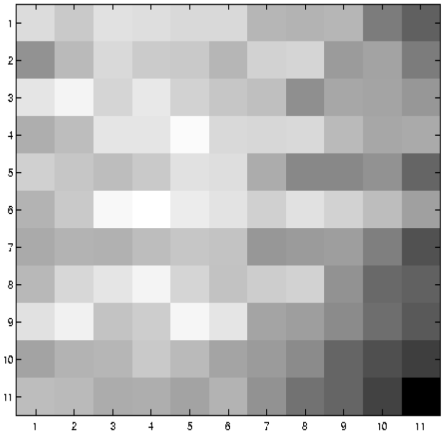
- $A^T A$ should be invertible
- $A^T A$ should not be too small due to noise
  - eigenvalues λ_1 and λ_2 of $A^T A$ should not be too small
- $A^T A$ should be well-conditioned
  - λ_1/λ_2 should not be too large (λ_1 = larger eigenvalue)

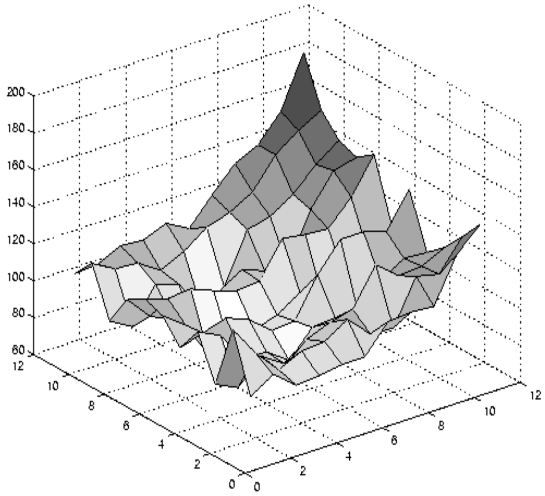
Does this remind you of anything?

Criteria for Harris corner detector

Low texture region





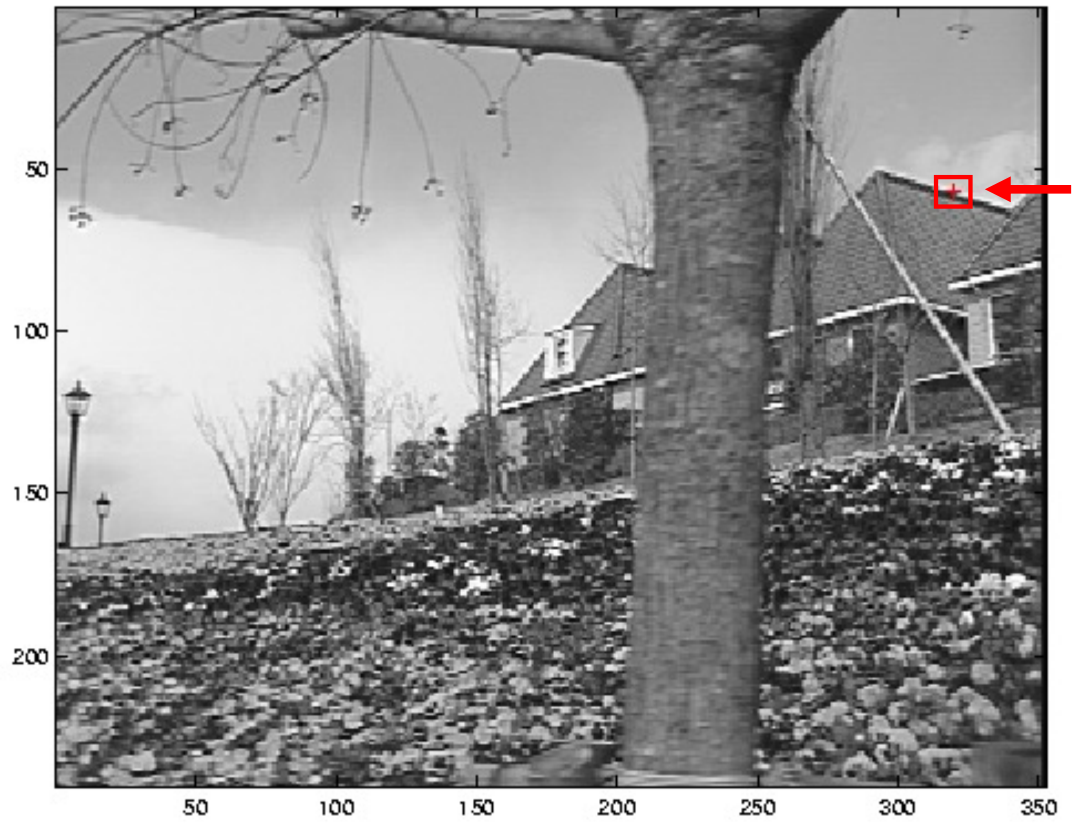


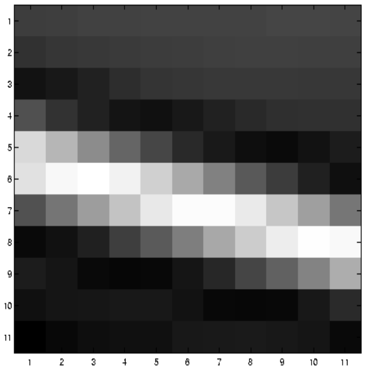
$$\sum \nabla I (\nabla I)^T$$

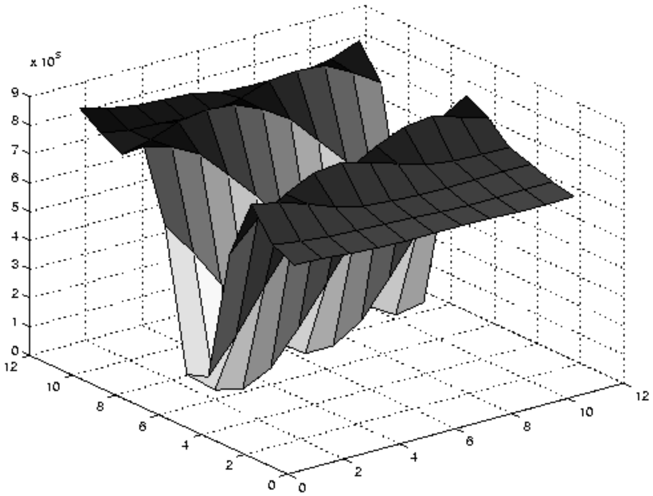
– gradients have small magnitude

– small λ_1, small λ_2

Edge





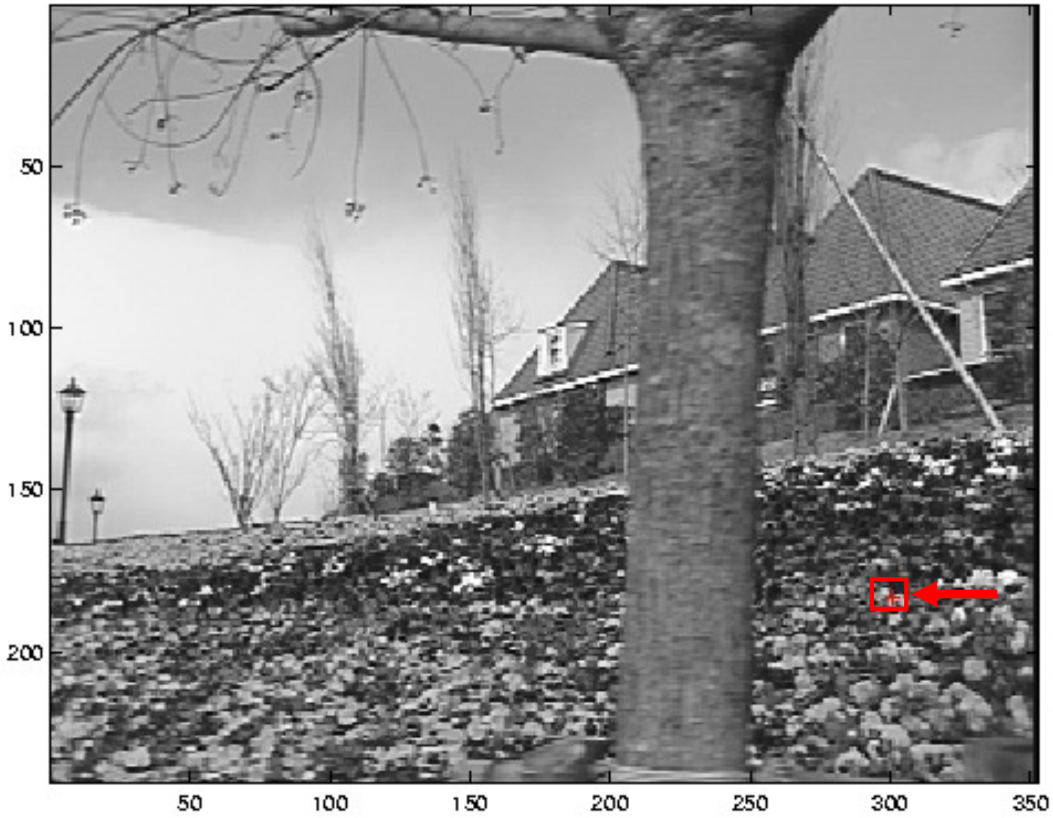


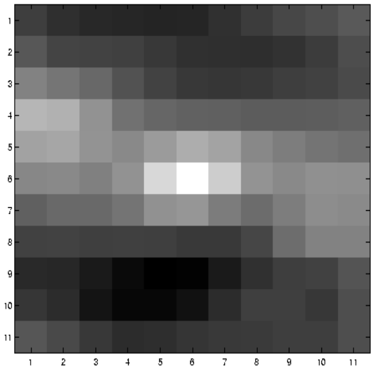
$$\sum \nabla I (\nabla I)^T$$

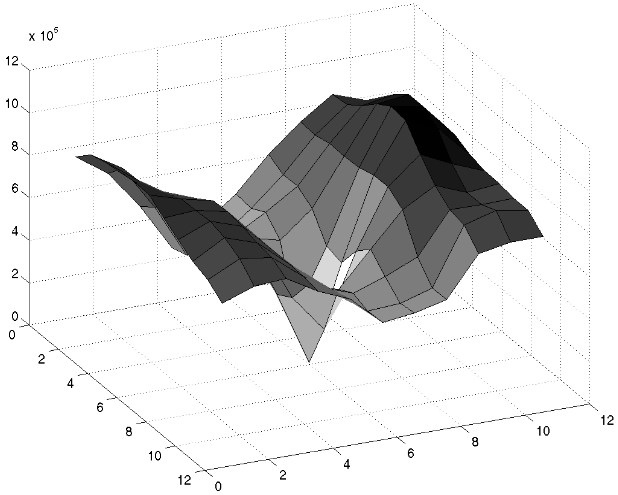
– large gradients, all the same

– large λ_1, small λ_2

High textured region





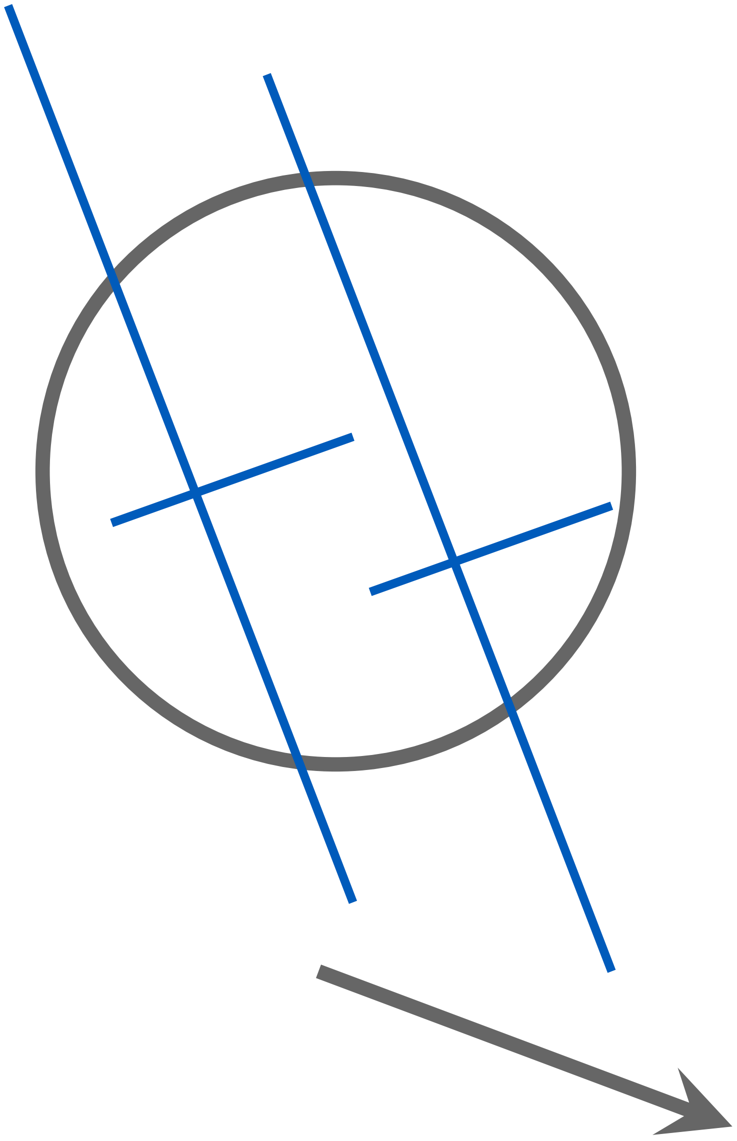


$$\sum \nabla I (\nabla I)^T$$

– gradients are different, large magnitudes

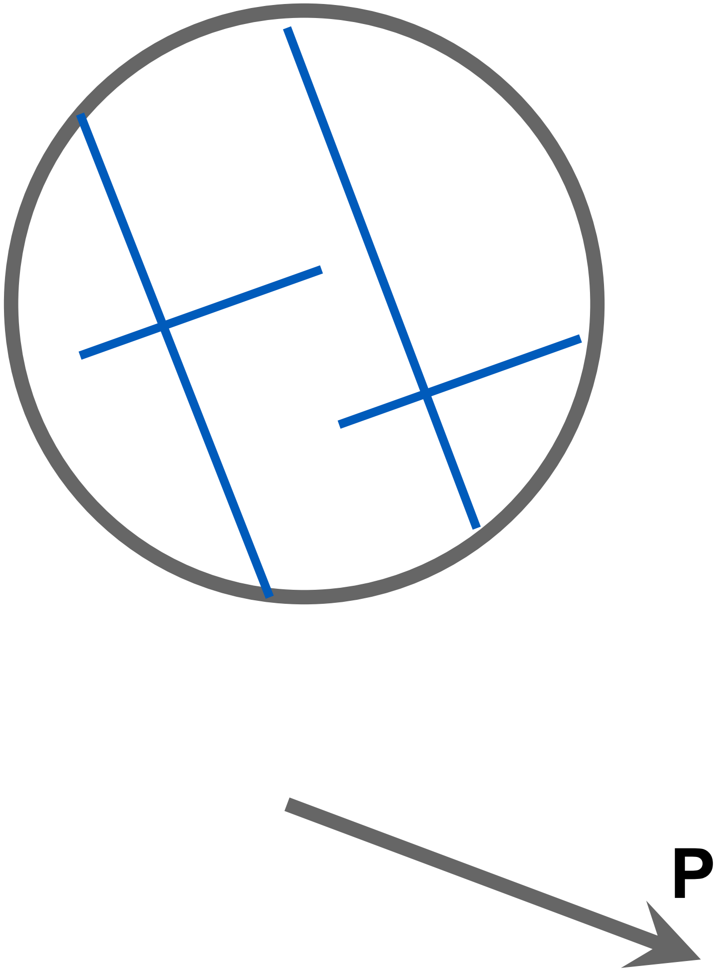
– large λ_1, large λ_2

The aperture problem resolved



Actual motion

The aperture problem resolved



Perceived motion

Errors in Lucas-Kanade

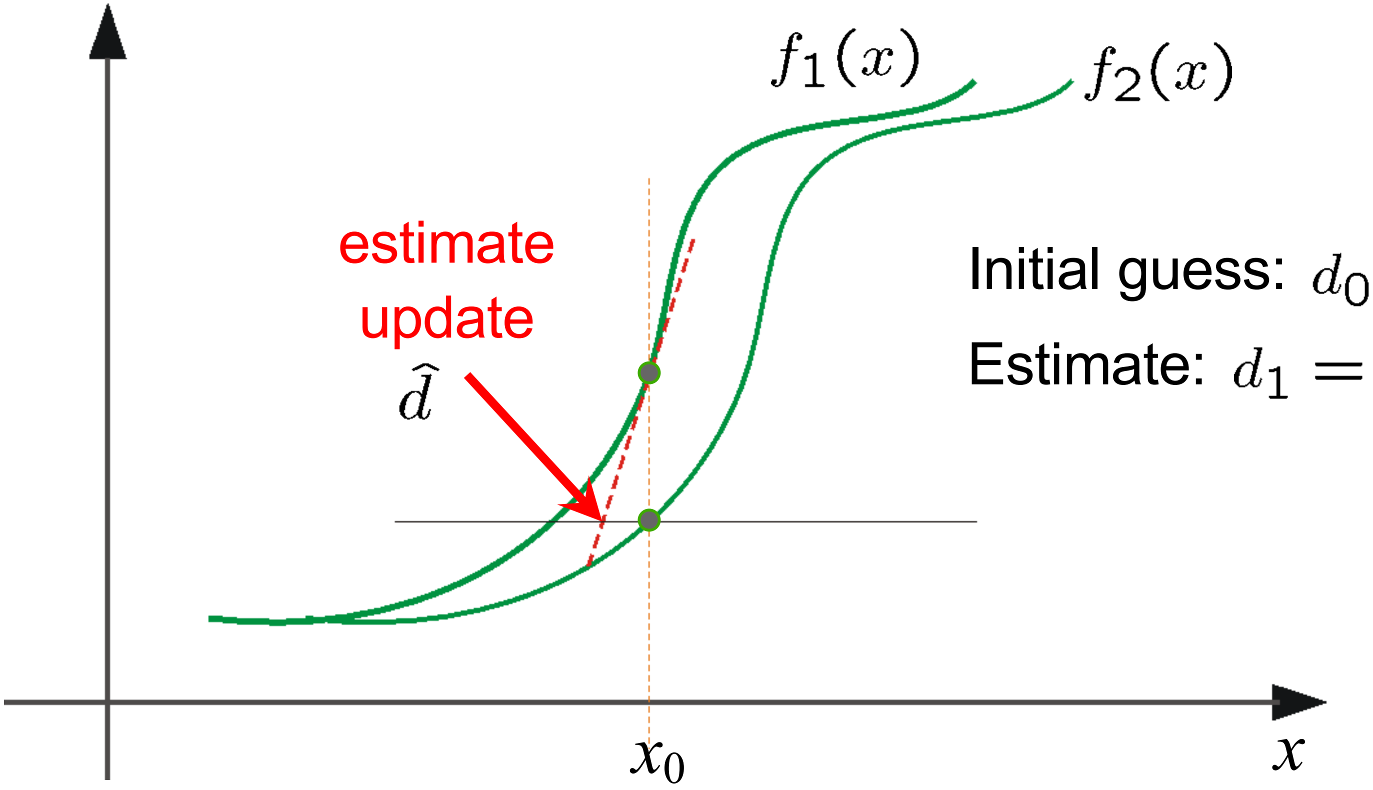
- A point does not move like its neighbors
  - Motion segmentation
- Brightness constancy does not hold
  - Do exhaustive neighborhood search with normalized correlation - tracking features – maybe SIFT
- The motion is large (larger than a pixel)
  1. Not-linear: Iterative refinement
  2. Local minima: coarse-to-fine estimation

Iterative Refinement

Iterative Lukas-Kanade Algorithm

1. Estimate velocity at each pixel by solving Lucas-Kanade equations
2. Warp I_t towards I_{t+1} with estimated flow.
   - *use image warping techniques*
3. Repeat until convergence

Optical Flow: Iterative Estimation



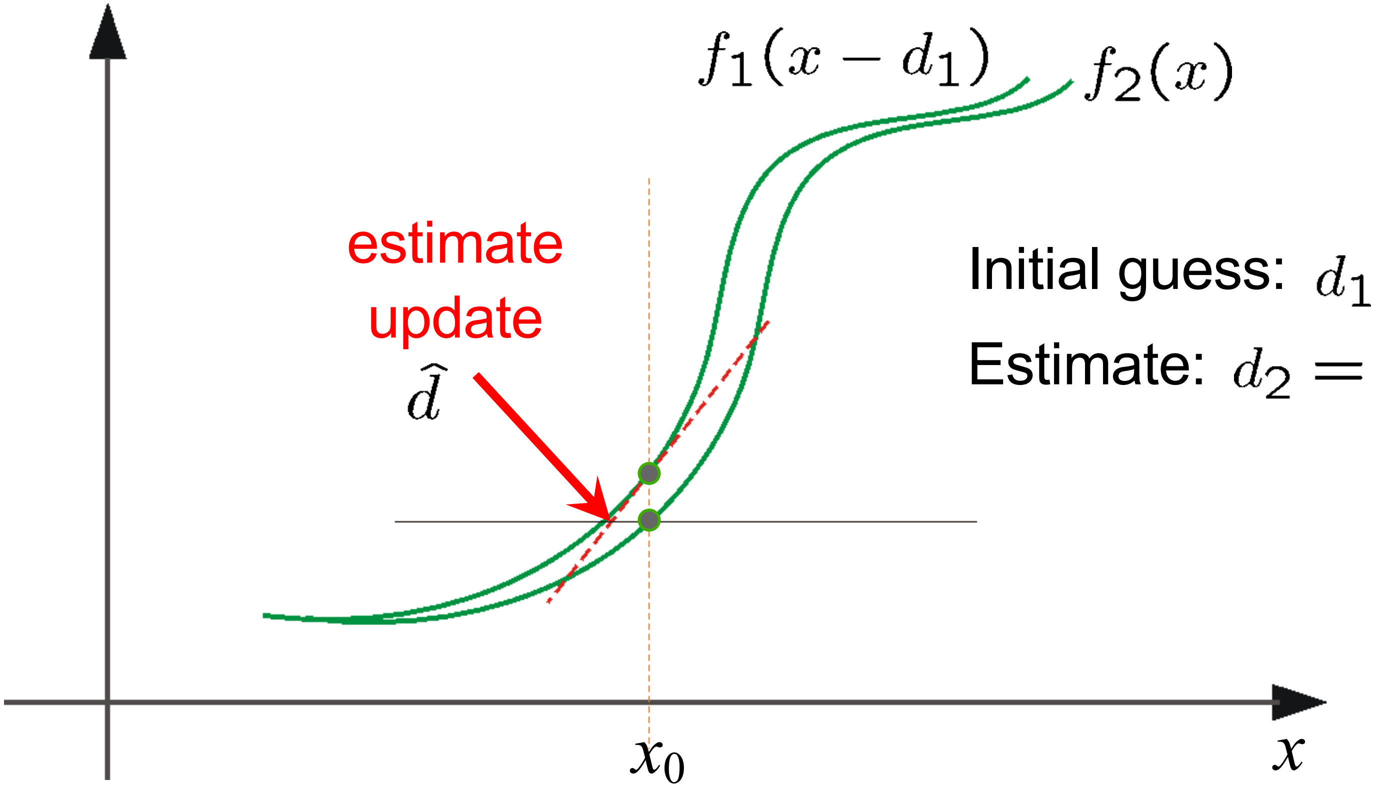
estimate update $\hat{d}$

Initial guess: $d_0 = 0$

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

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

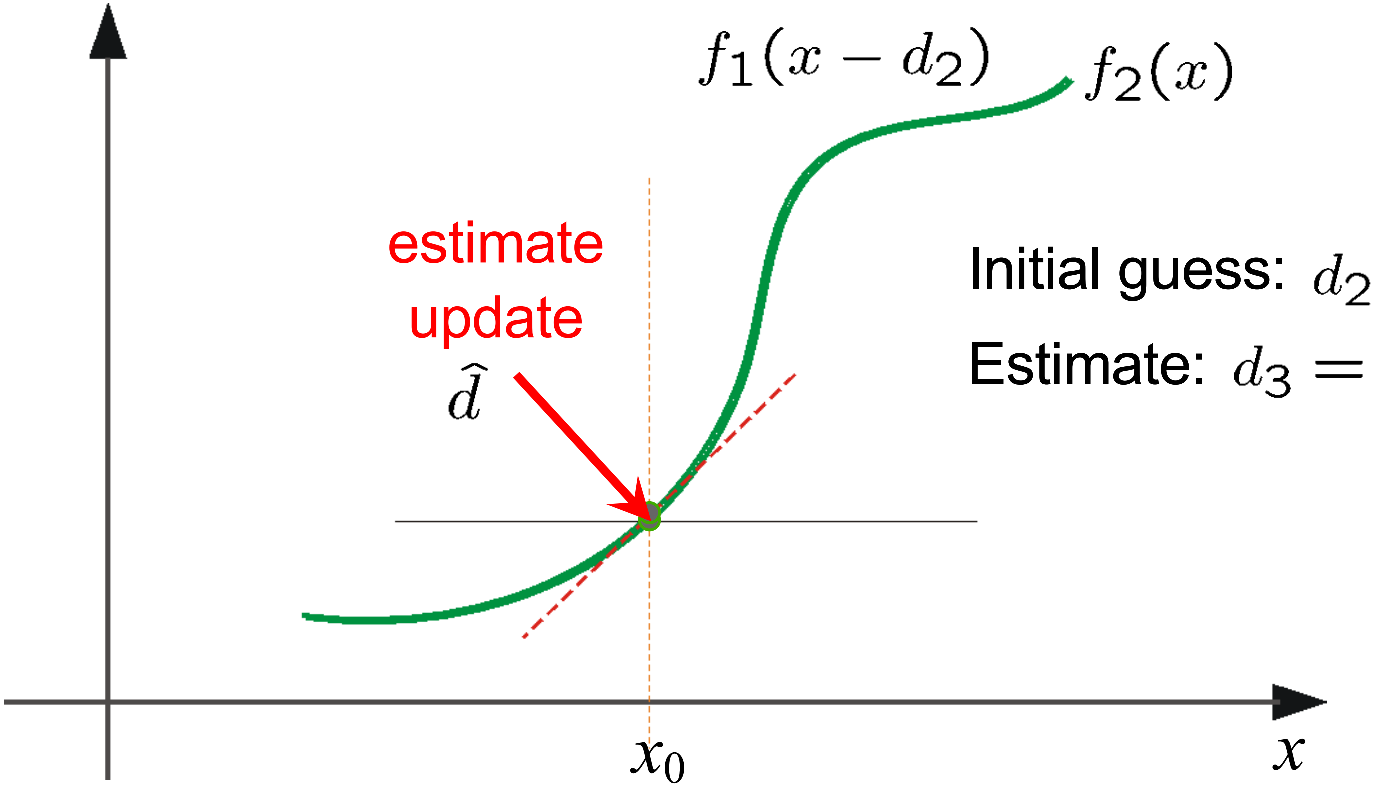
Optical Flow: Iterative Estimation



Initial guess: d_1

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

Optical Flow: Iterative Estimation

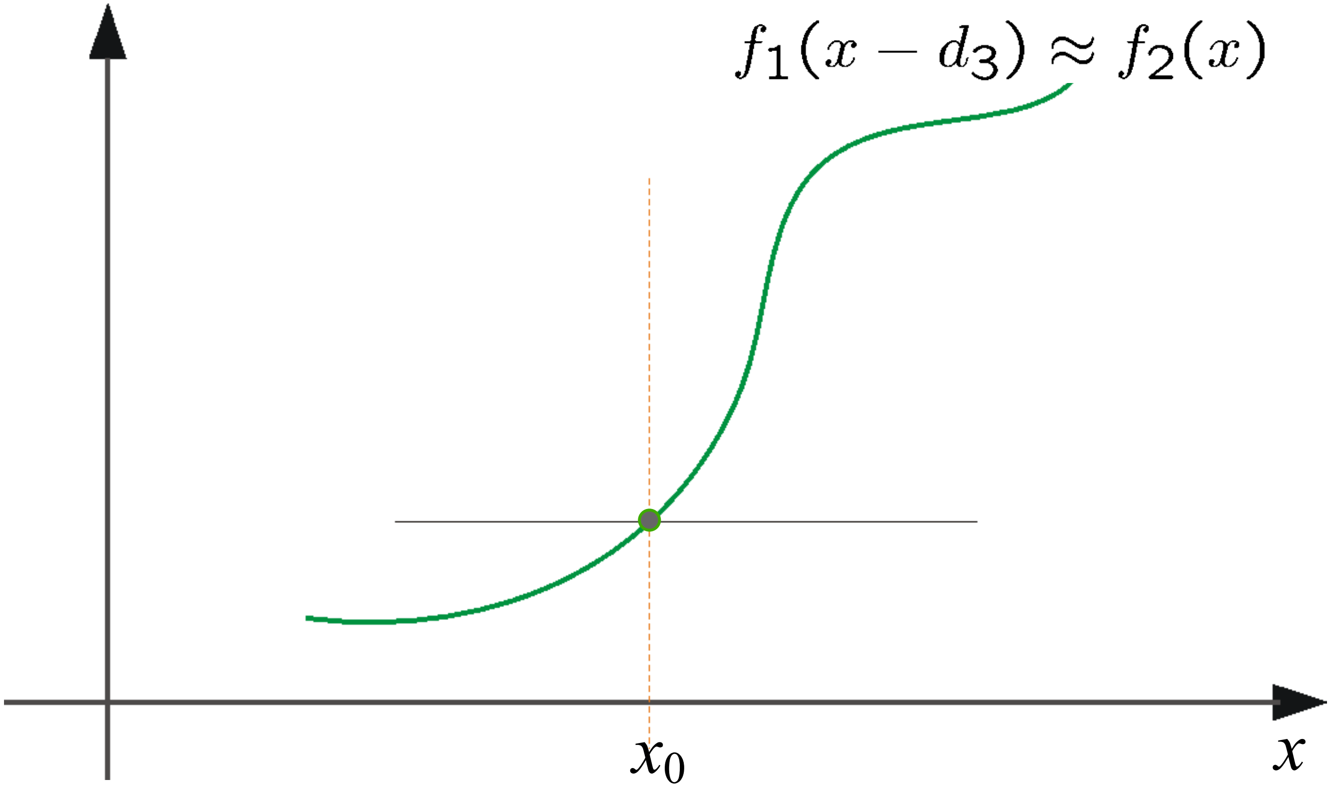


estimate update $\hat{d}$

Initial guess: d_2

Estimate: $d_3 = d_2 + \hat{d}$

Optical Flow: Iterative Estimation



Optical Flow: Iterative Estimation

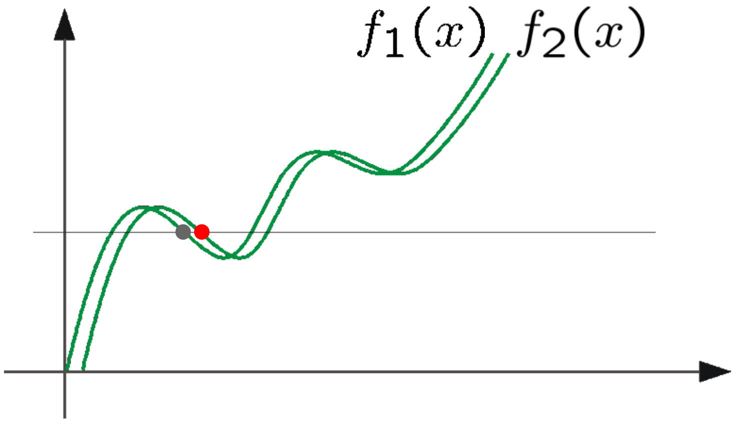
- Some Implementation Issues:
  - Warping is not easy (ensure that errors in warping are smaller than the estimate refinement).
  - Often useful to low-pass filter the images before motion estimation (for better derivative estimation, and linear approximations to image intensity)

Revisiting the small motion assumption

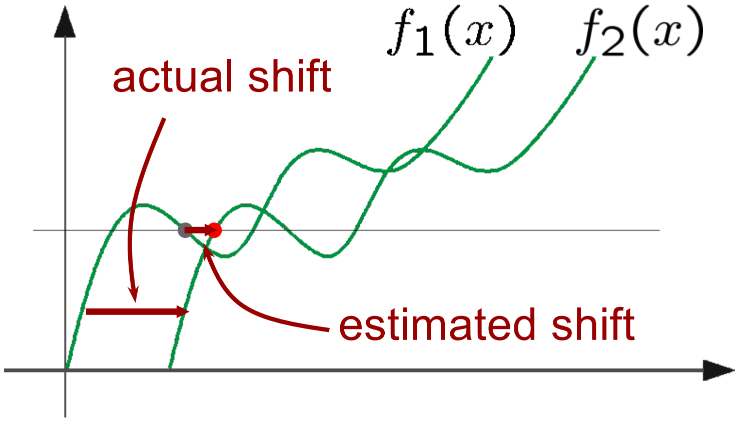
- Is this motion small enough?
  - Probably not—it’s much larger than one pixel
  - How might we solve this problem?

Optical Flow: Aliasing

- Temporal aliasing causes ambiguities, because we can have many pixels with the same intensity.
- How do we know which ‘correspondence’ is correct?



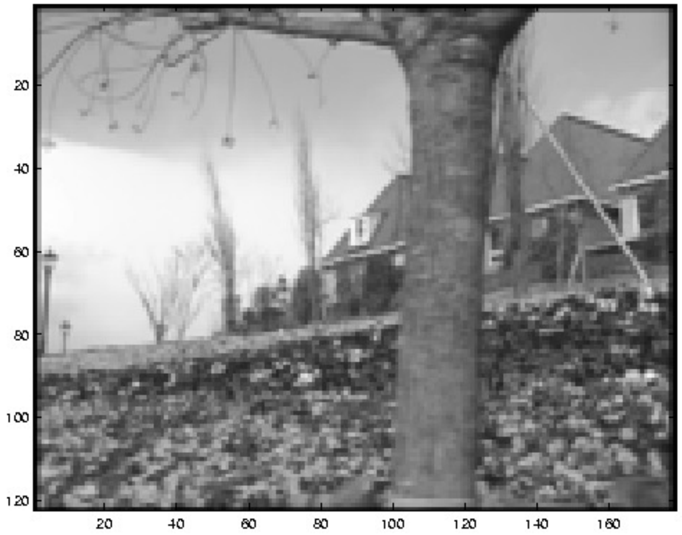
nearest match is correct
(no aliasing)

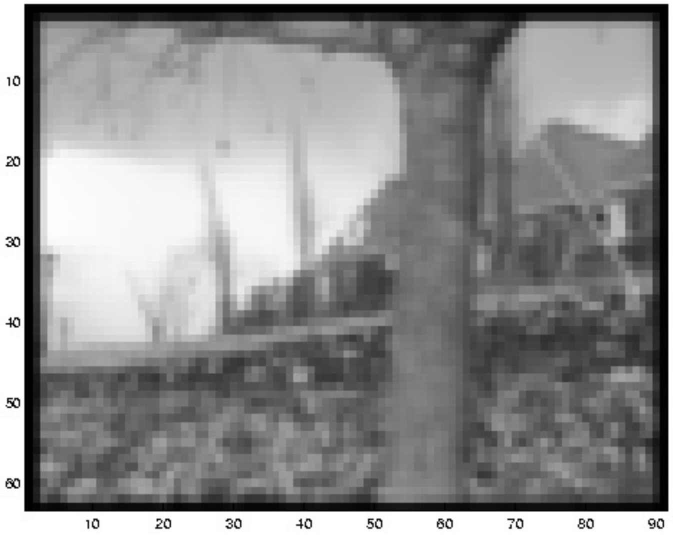


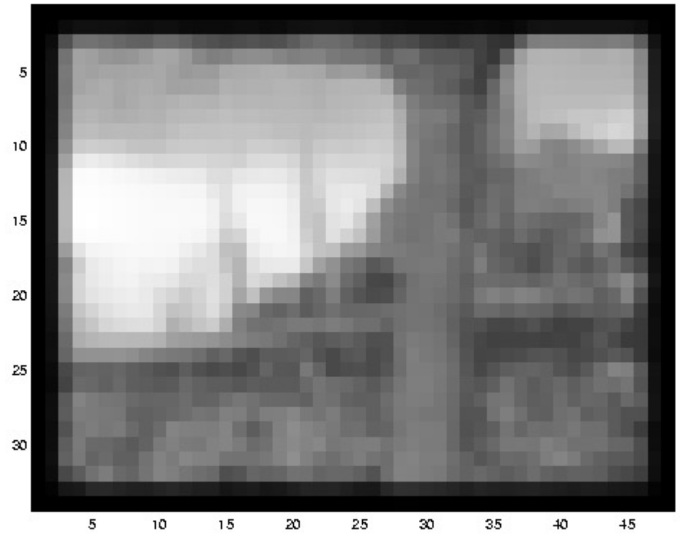
nearest match is incorrect
(aliasing)

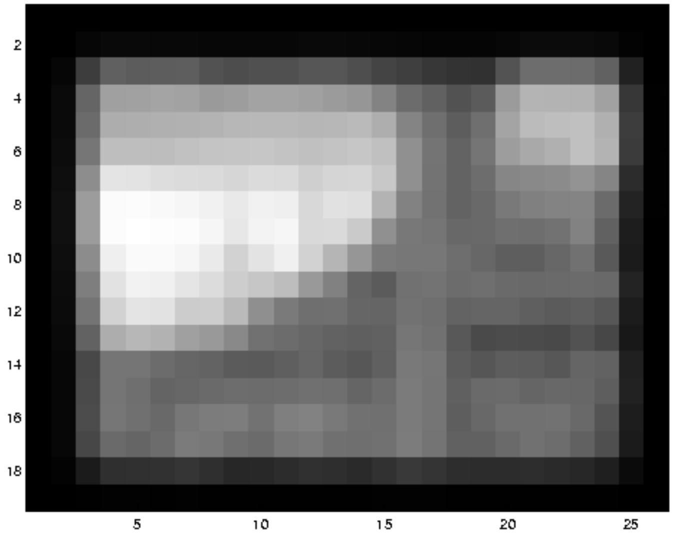
To overcome aliasing: coarse-to-fine estimation.

Reduce the resolution!

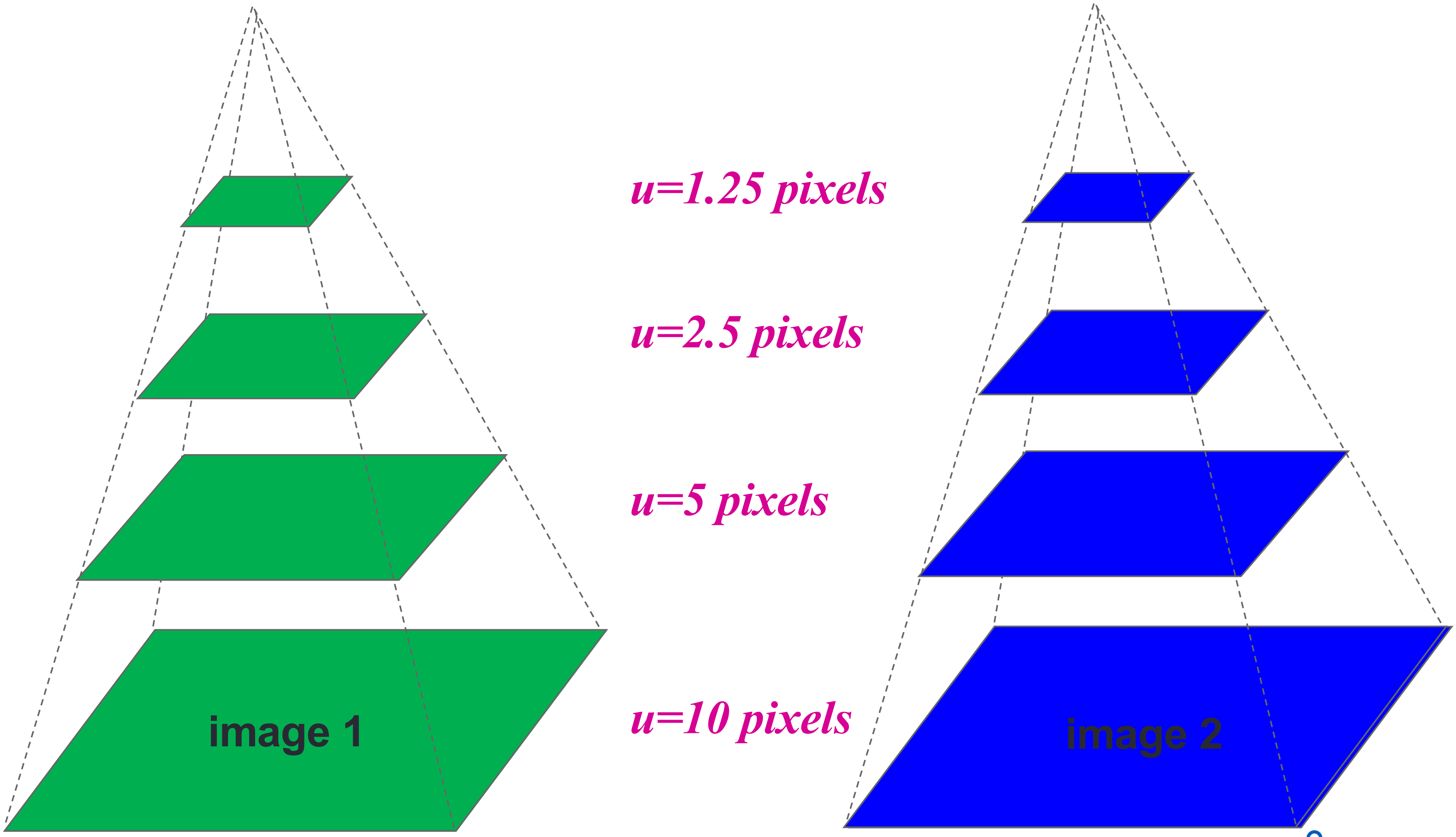








Coarse-to-fine optical flow estimation



u=1.25 pixels

u=2.5 pixels

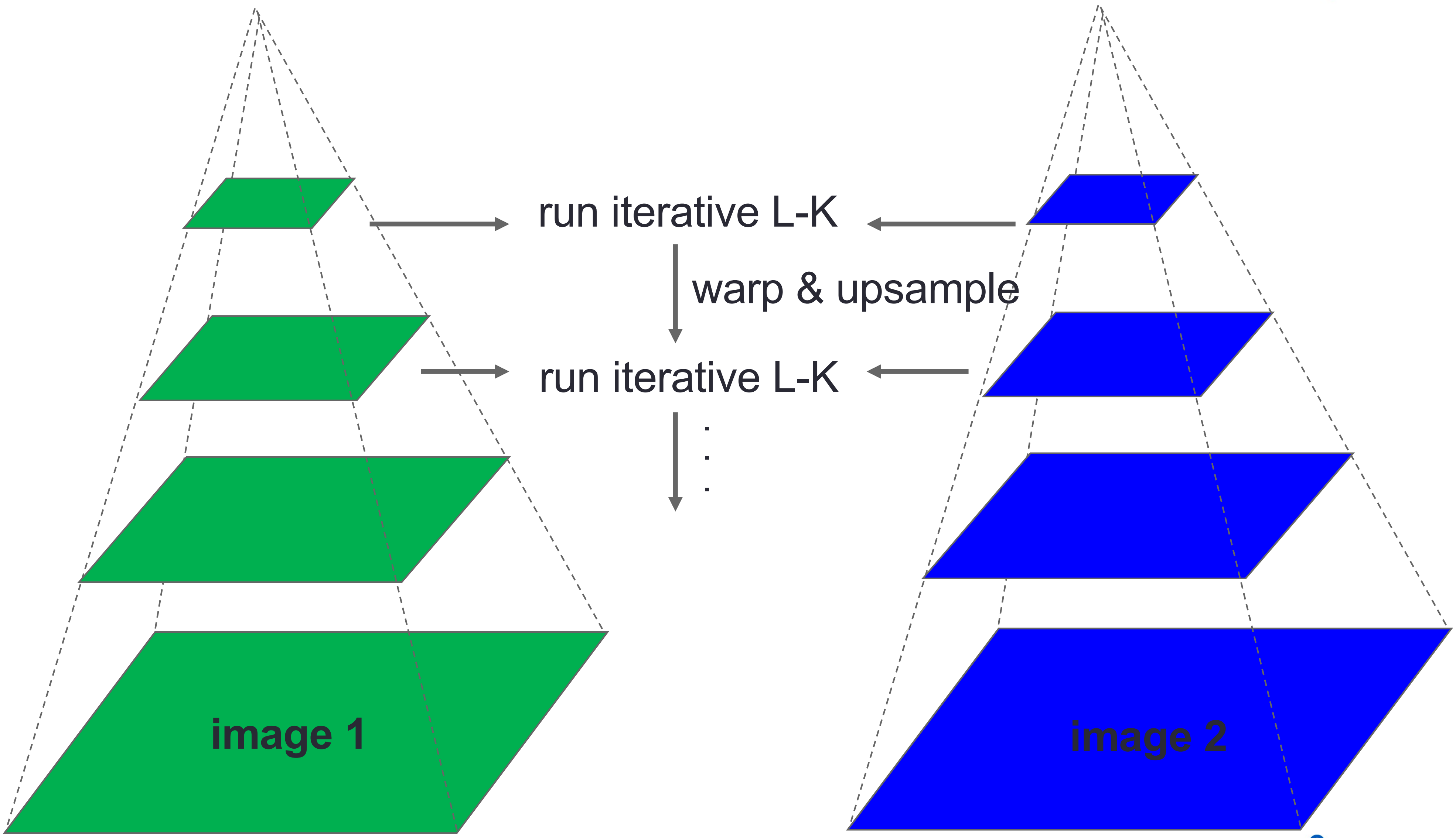
u=5 pixels

u=10 pixels

Gaussian pyramid of image 1

Gaussian pyramid of image 2

Coarse-to-fine optical flow estimation



run iterative L-K

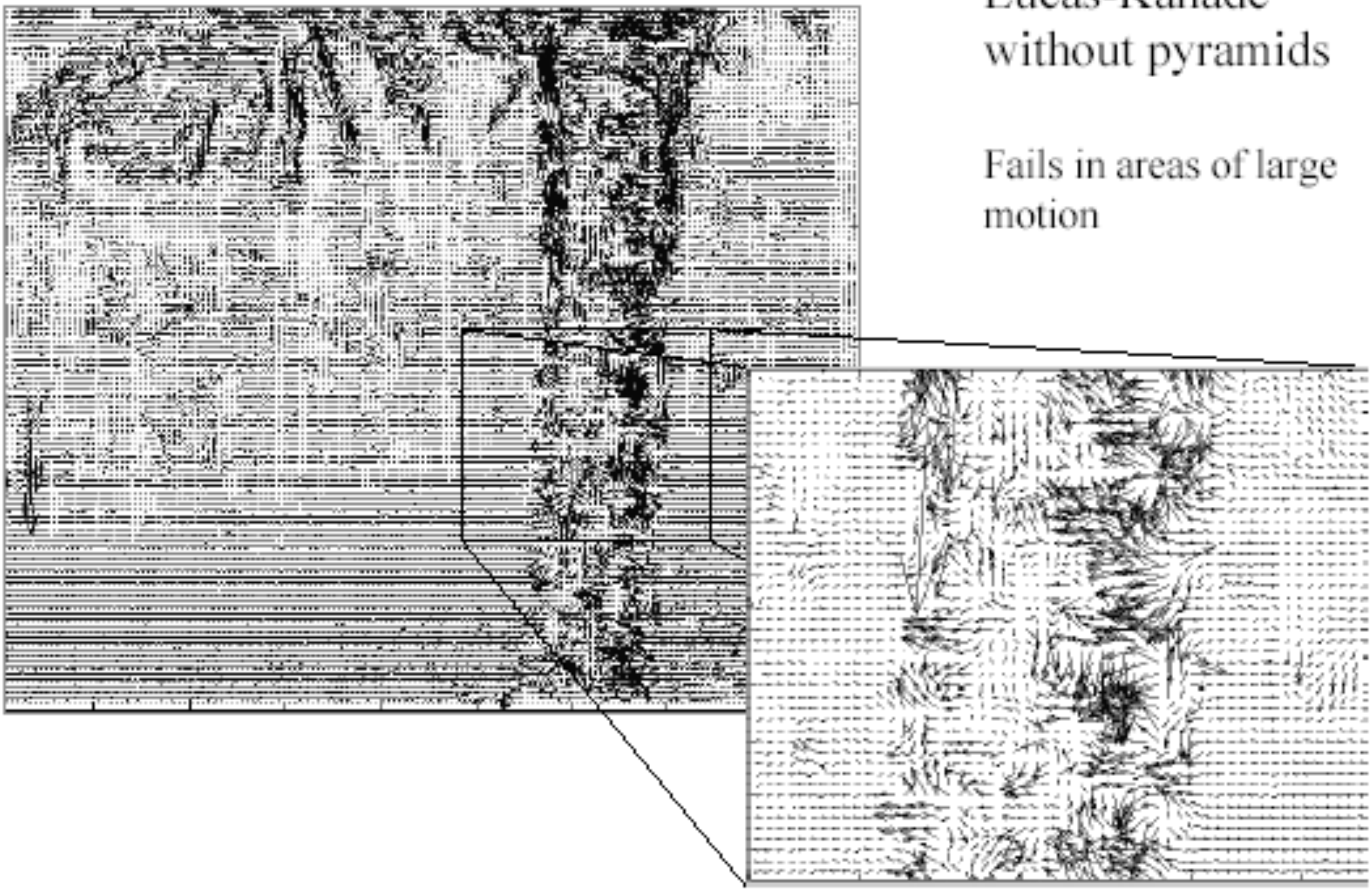
warp & upsample

run iterative L-K

Gaussian pyramid of image 1

Gaussian pyramid of image 2

Optical Flow Results

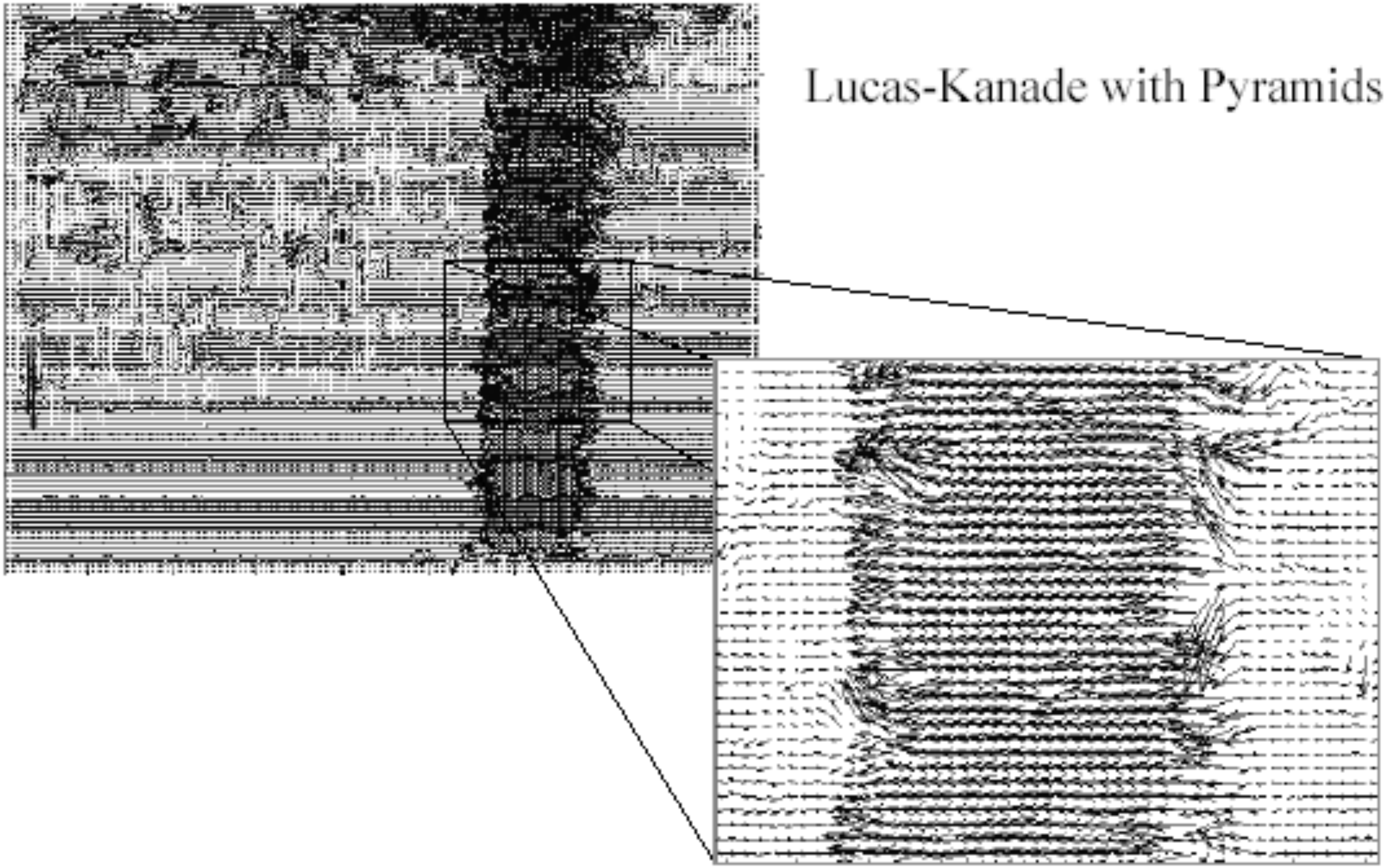


Lucas-Kanade without pyramids

Fails in areas of large motion

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Optical Flow Results

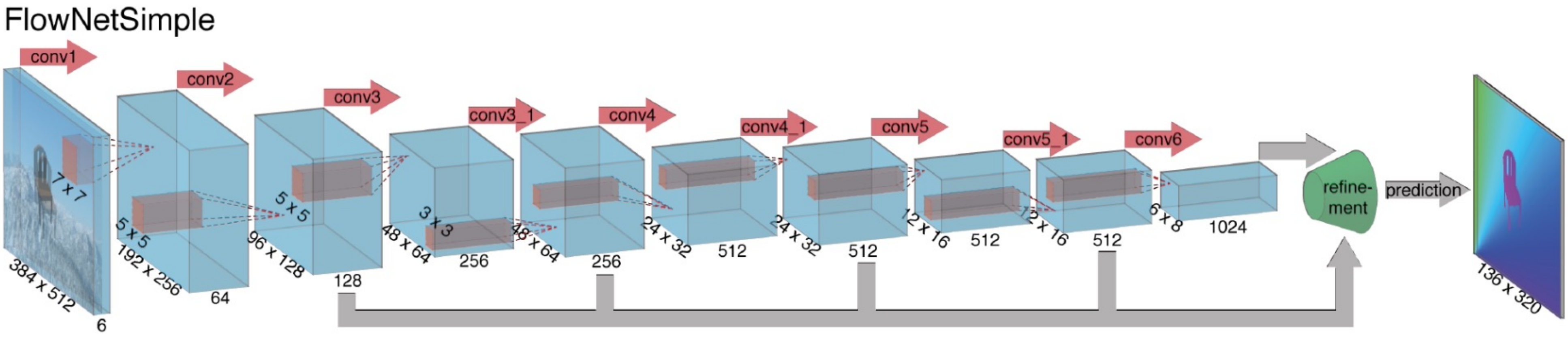


Lucas-Kanade with Pyramids

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Deep Optical Flow

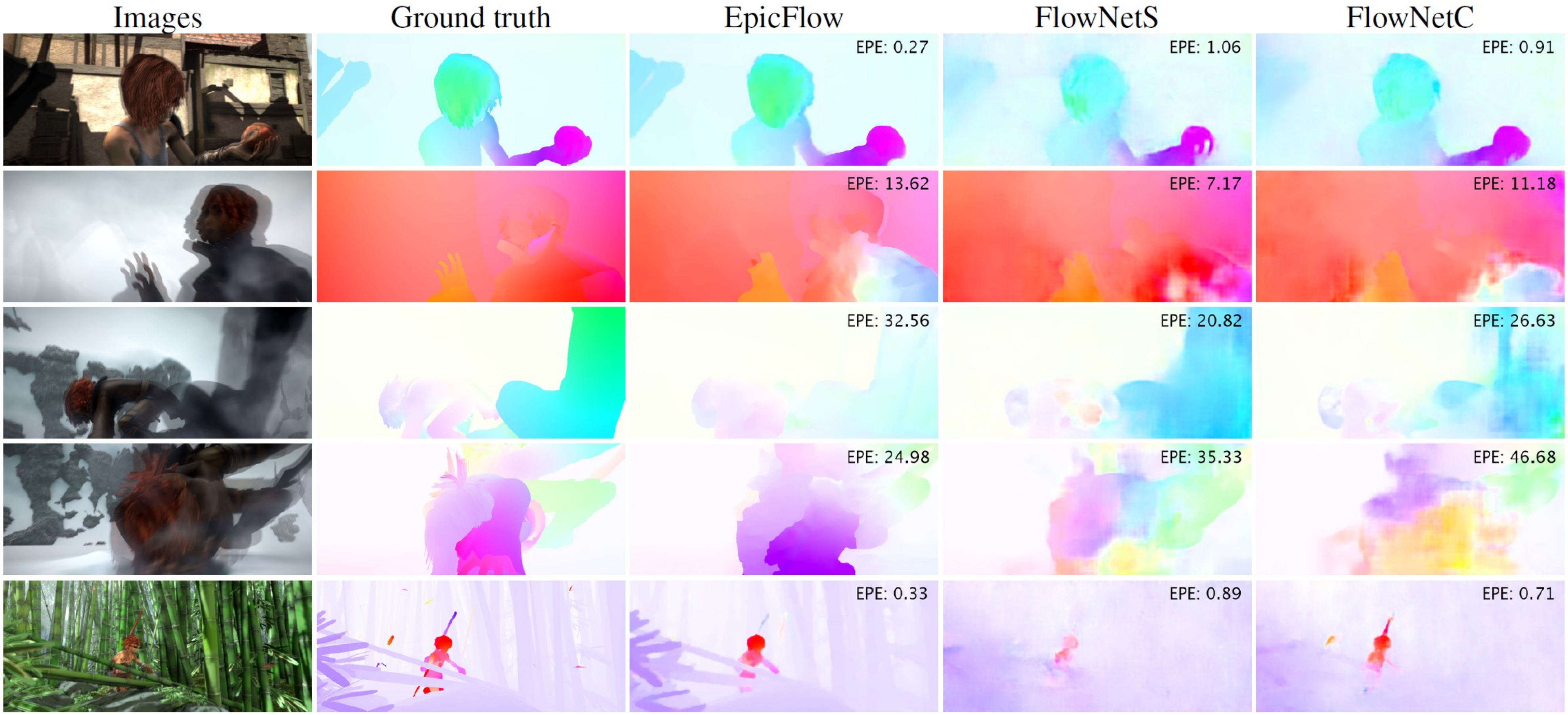
- Deep convolutional network, which accepts a pair of input frames and upsamples the estimated flow back to input resolution.



Fischer et al. 2015. https://arxiv.org/abs/1504.06852

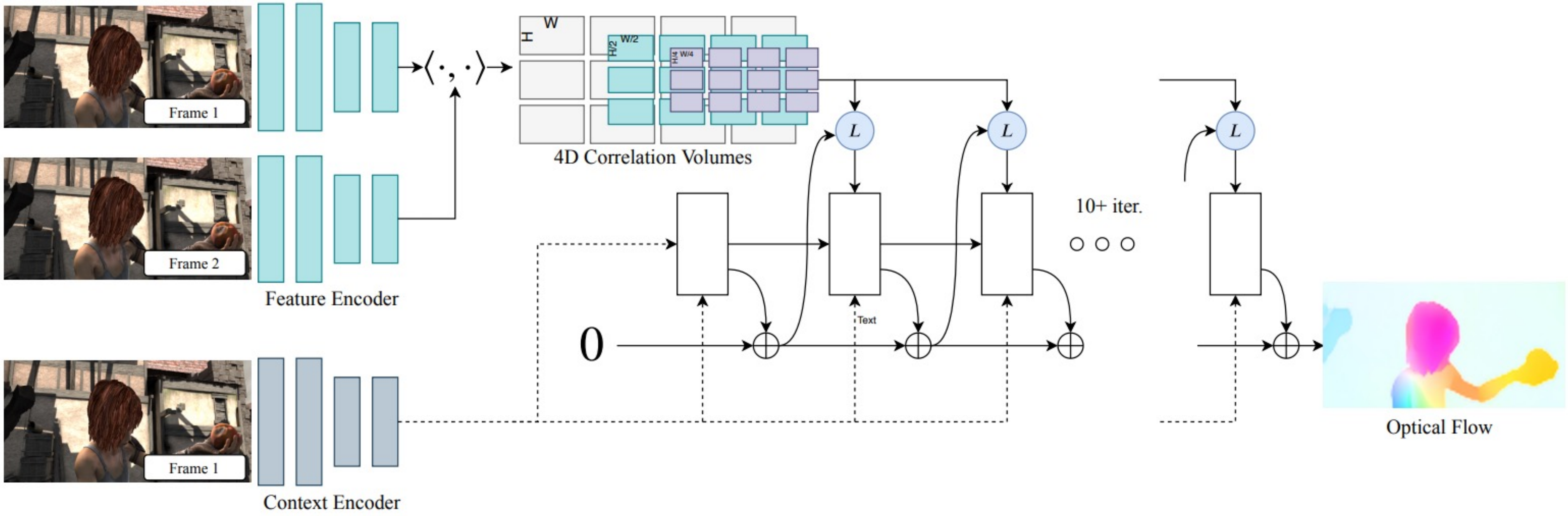
Deep optical flow, 2015

Results on Sintel



Fischer et al. 2015. https://arxiv.org/abs/1504.06852

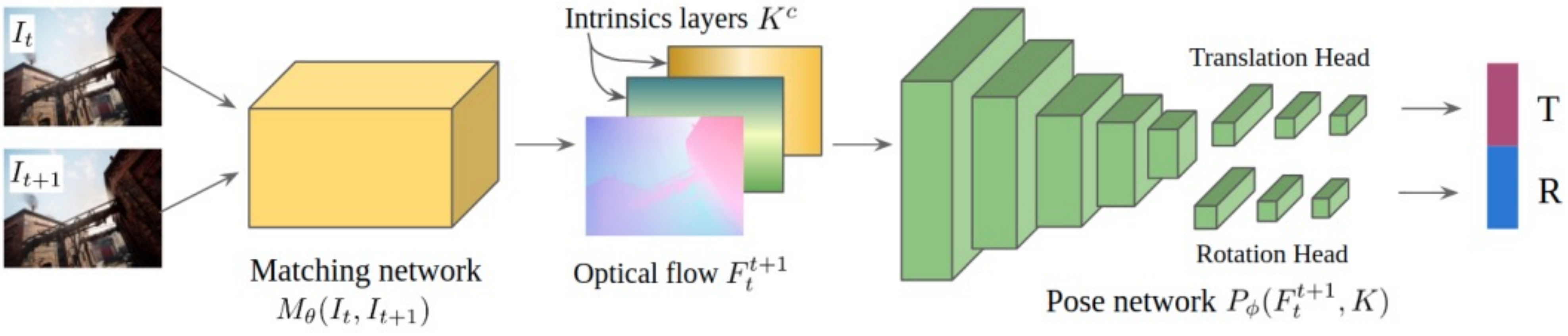
Deep Recurrent Optical Flow, 2020



- A feature encoder that extracts per-pixel features.
- A correlation layer by taking the inner product of all pairs of feature vectors.
- An update operator which recurrently updates optical flow by using the current estimate.

Teed, Zachary, and Jia Deng. "Raft: Recurrent all-pairs field transforms for optical flow." *ECCV 2020*

Learning-based Visual Odometry, 2021



- The two-stage network architecture.
  - A matching network, which estimates optical flow from two consecutive RGB images,
  - A pose network predicting camera motion from the optical flow.

Wang, W., Hu, Y., & Scherer, S. (2021, October). TartanVO: A generalizable learning-based VO. In *Conference on Robot Learning* (pp. 1761-1772). PMLR.