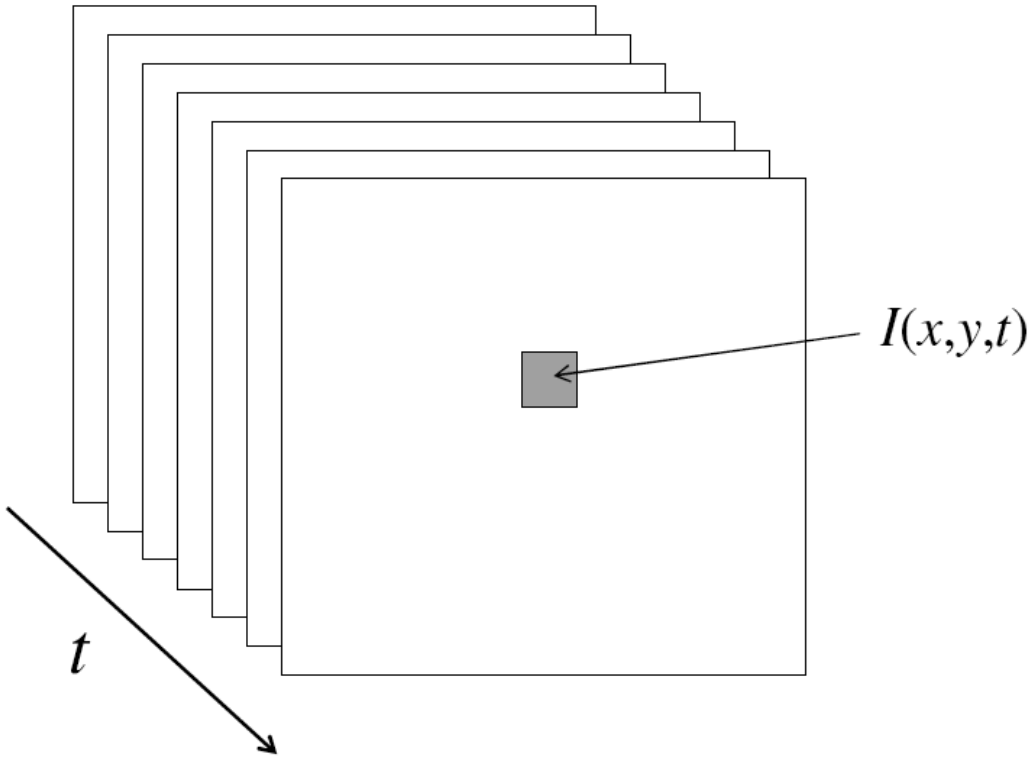
Optical Flow and Feature Tracking

Many slides adapted from S. Seitz, R. Szeliski, M. Pollefeys, S. Lazebnik

Video

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

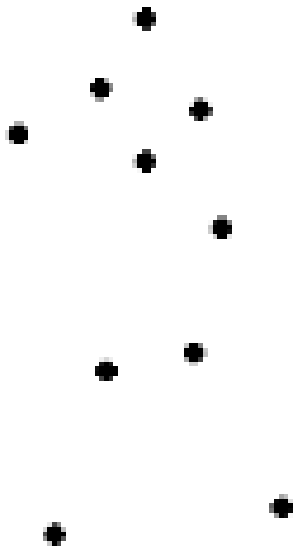


Motion and perceptual organization

- Sometimes, motion is foremost cue

Motion and perceptual organization

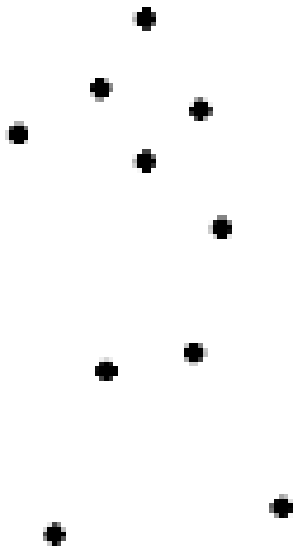
- Even “impoverished” motion data can evoke a strong percept



G. Johansson, “Visual Perception of Biological Motion and a Model For Its Analysis", *Perception and Psychophysics 14, 201-211, 1973.*

Motion and perceptual organization

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



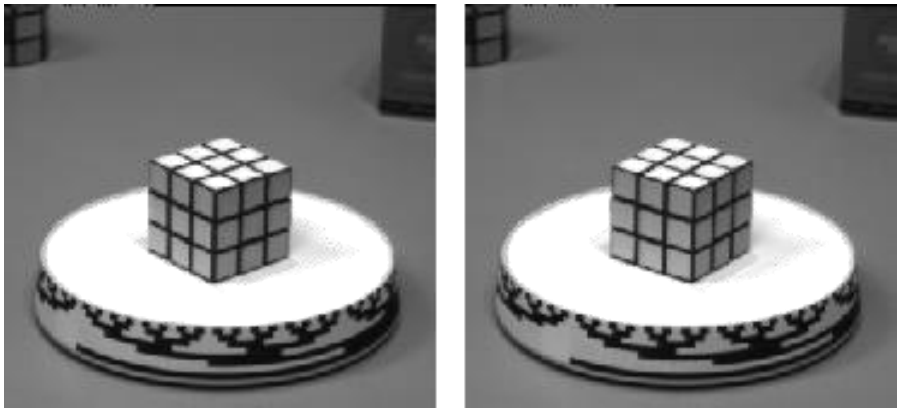
G. Johansson, “Visual Perception of Biological Motion and a Model For Its Analysis", *Perception and Psychophysics 14, 201-211, 1973.*

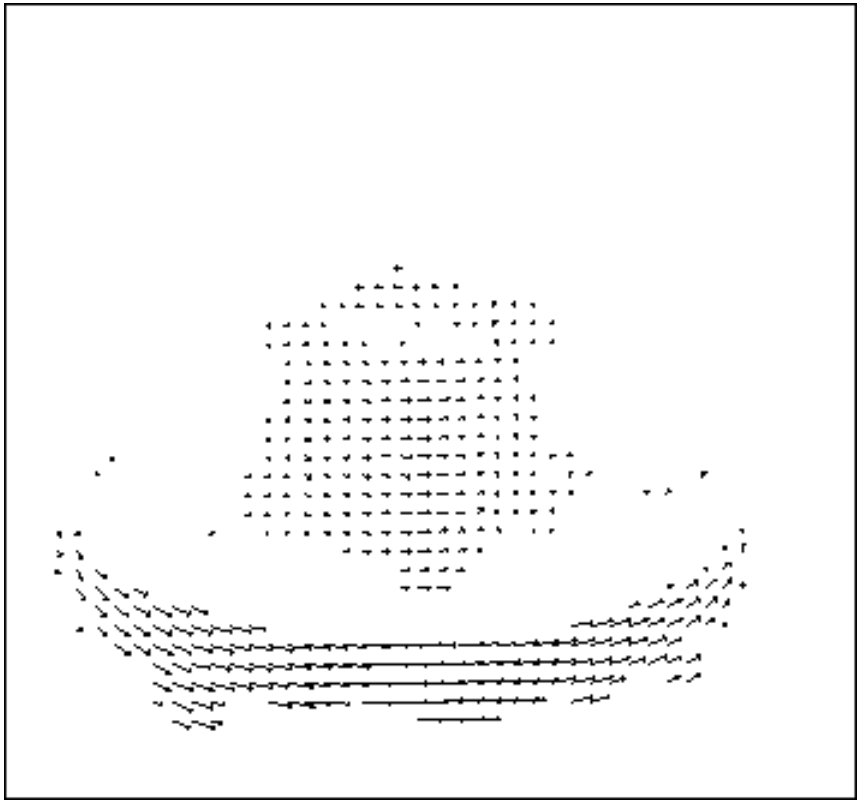
Uses of motion

- Estimating 3D structure
- Segmenting objects based on motion cues
- Learning dynamical models
- Recognizing events and activities
- Improving video quality (motion stabilization)

Motion field

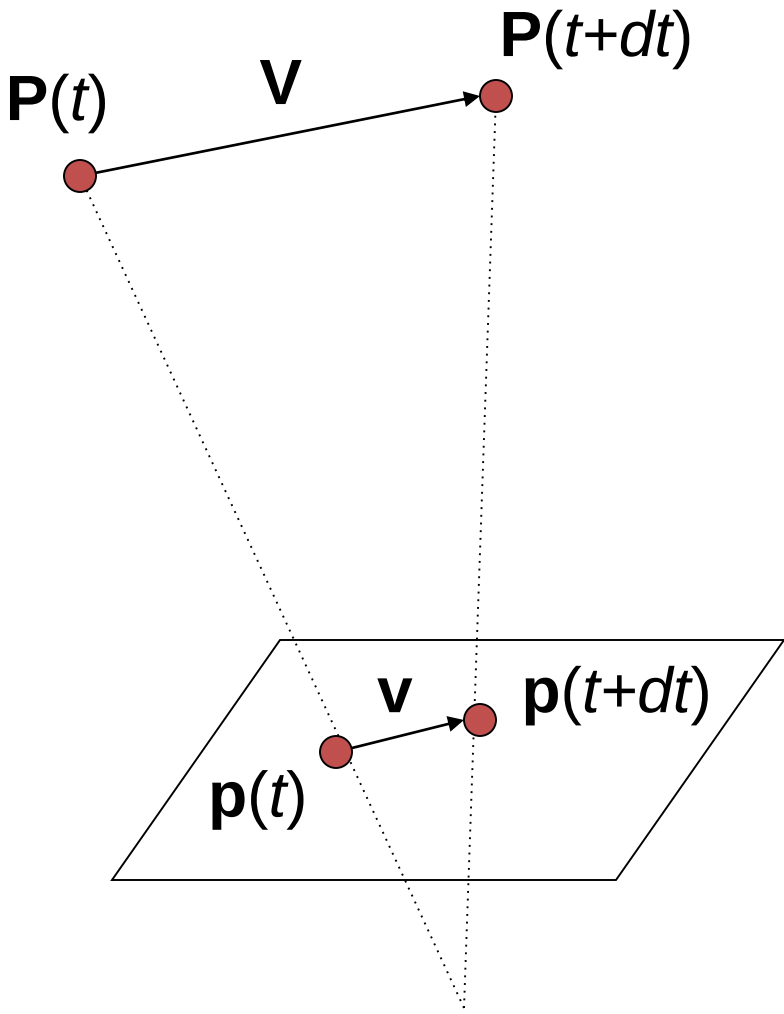
- The motion field is the projection of the 3D scene motion into the image





Motion field

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



Motion estimation techniques

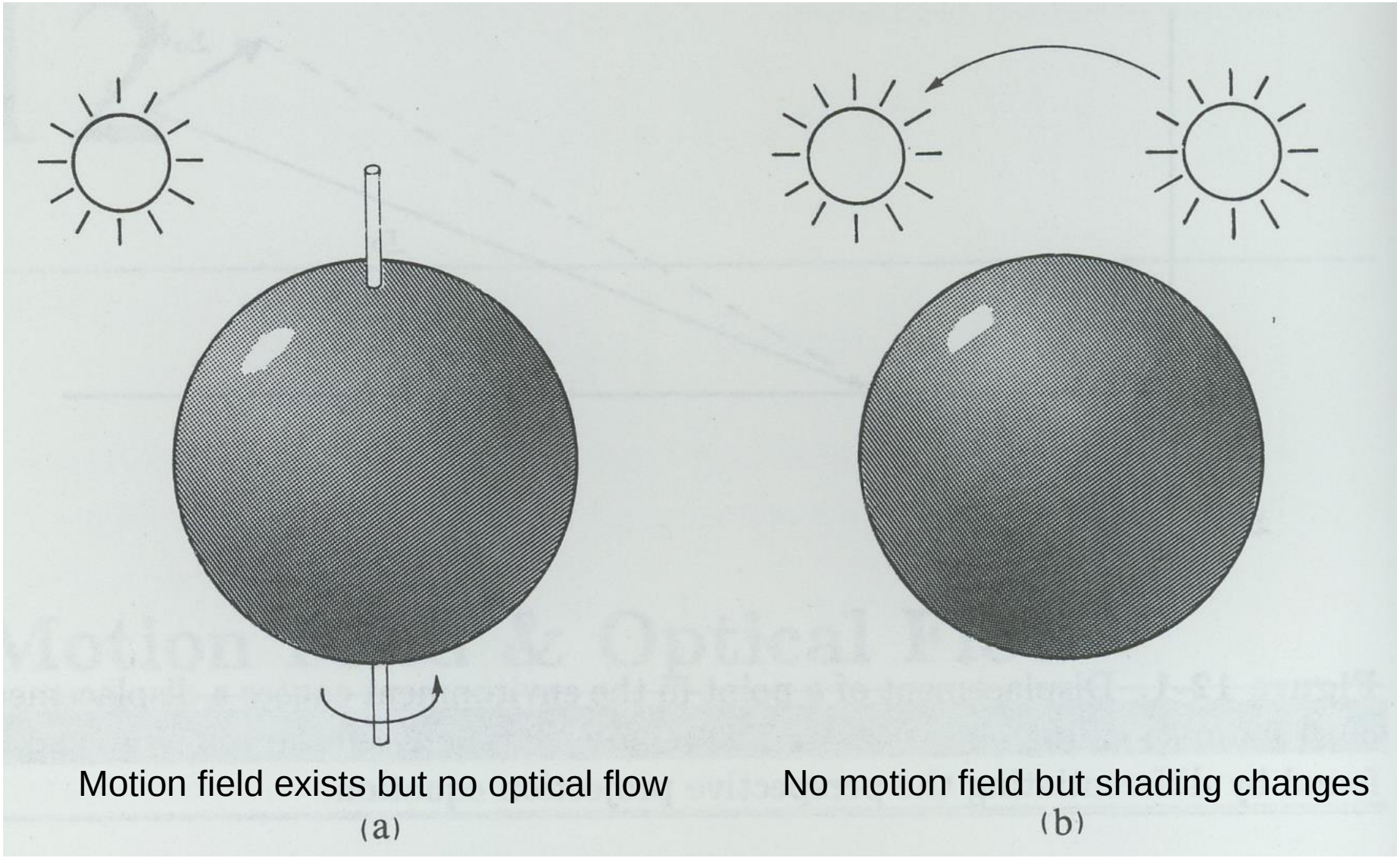
- Direct methods
  - Directly recover image motion at each pixel from spatio-temporal image brightness variations
  - Dense motion fields, but sensitive to appearance variations
  - Suitable for video and when image motion is small

- Feature-based methods
  - Extract visual features (corners, textured areas) and track them over multiple frames
  - Sparse motion fields, but more robust tracking
  - Suitable when image motion is large

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

Optical Flow ≠ Motion Field



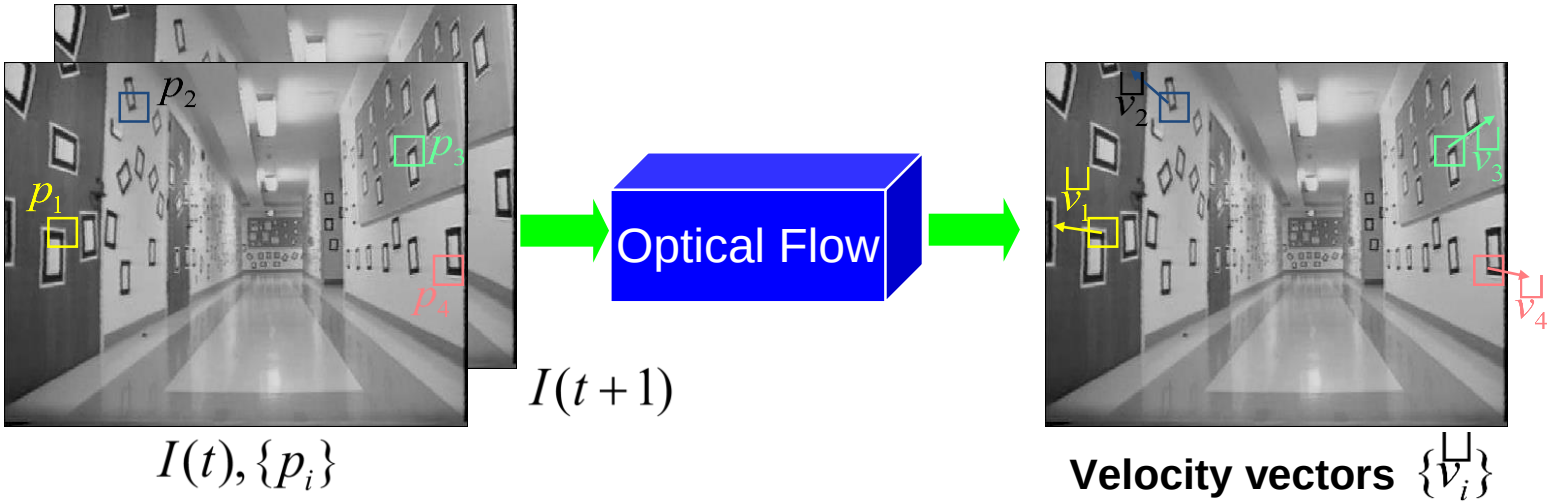
Motion field exists but no optical flow

(a)

No motion field but shading changes

(b)

What is Optical Flow?



Optical flow is the relation of the motion field

- *the 2D projection of the physical movement of points relative to the observer*

to 2D displacement of pixel patches on the image plane.

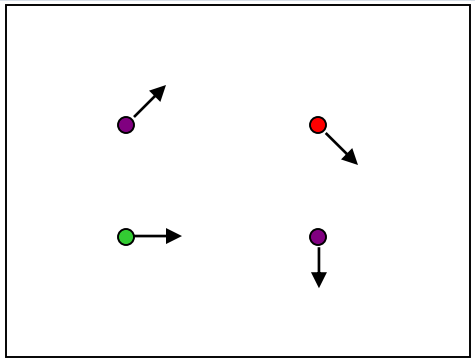
Common assumption:

The appearance of the image patches do not change (brightness constancy)

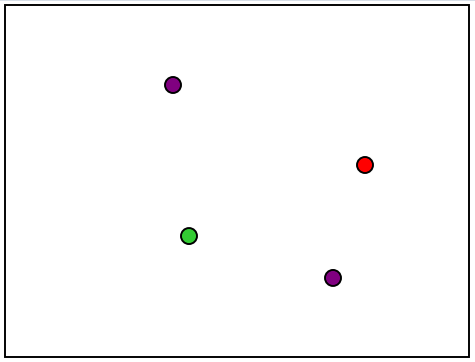
$$I(p_i, t) = I(p_i + v_i, t+1)$$

Note: more elaborate tracking models can be adopted if more frames are process all at once

Estimating optical flow



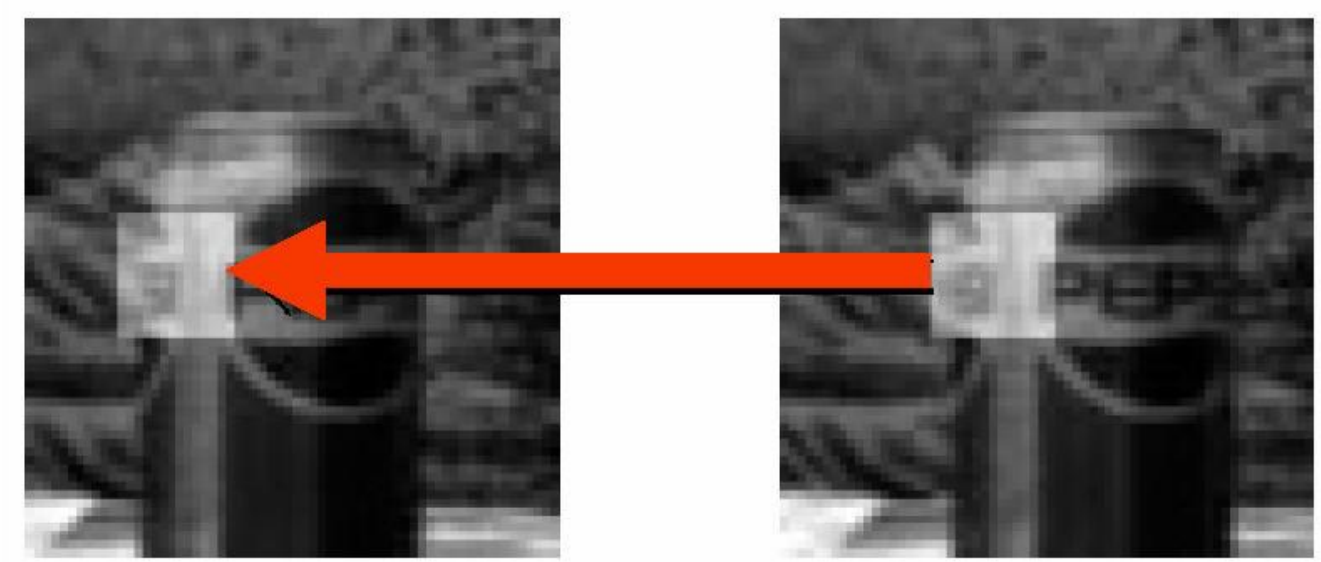
$I(x,y,t-1)$



$I(x,y,t)$

- Given two subsequent frames, estimate the apparent motion field 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

Optical Flow Assumptions: Brightness Constancy



Assumption

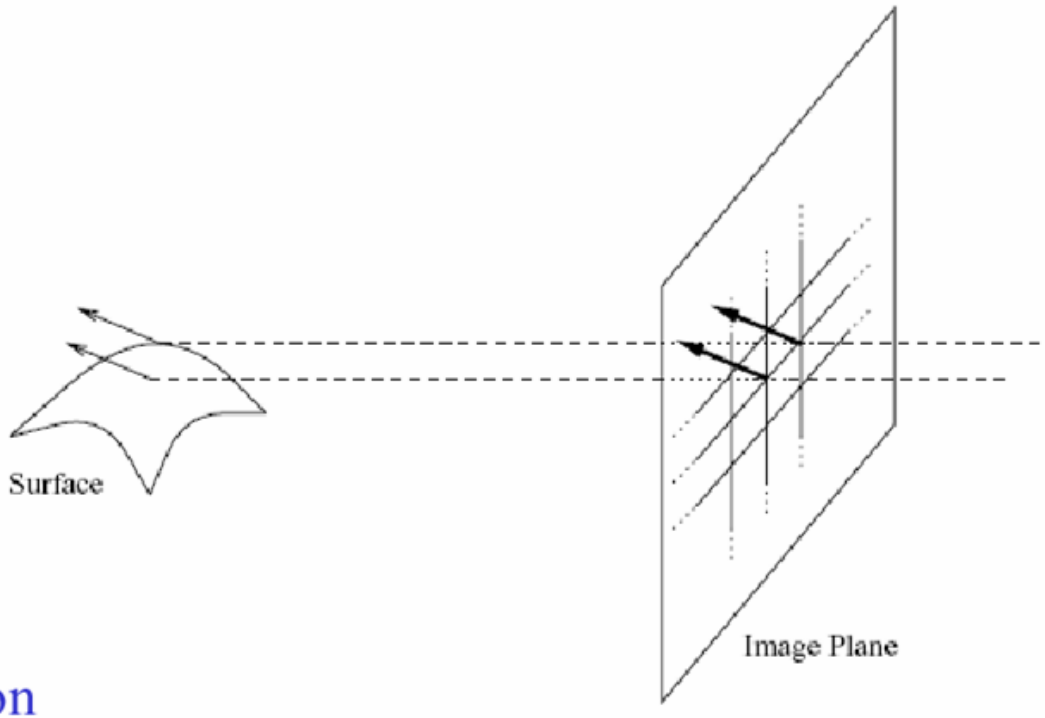
Image measurements (e.g. brightness) in a small region remain the same although their location may change.

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

(assumption)

* Slide from Michael Black, CS143 2003

Optical Flow Assumptions: Spatial Coherence

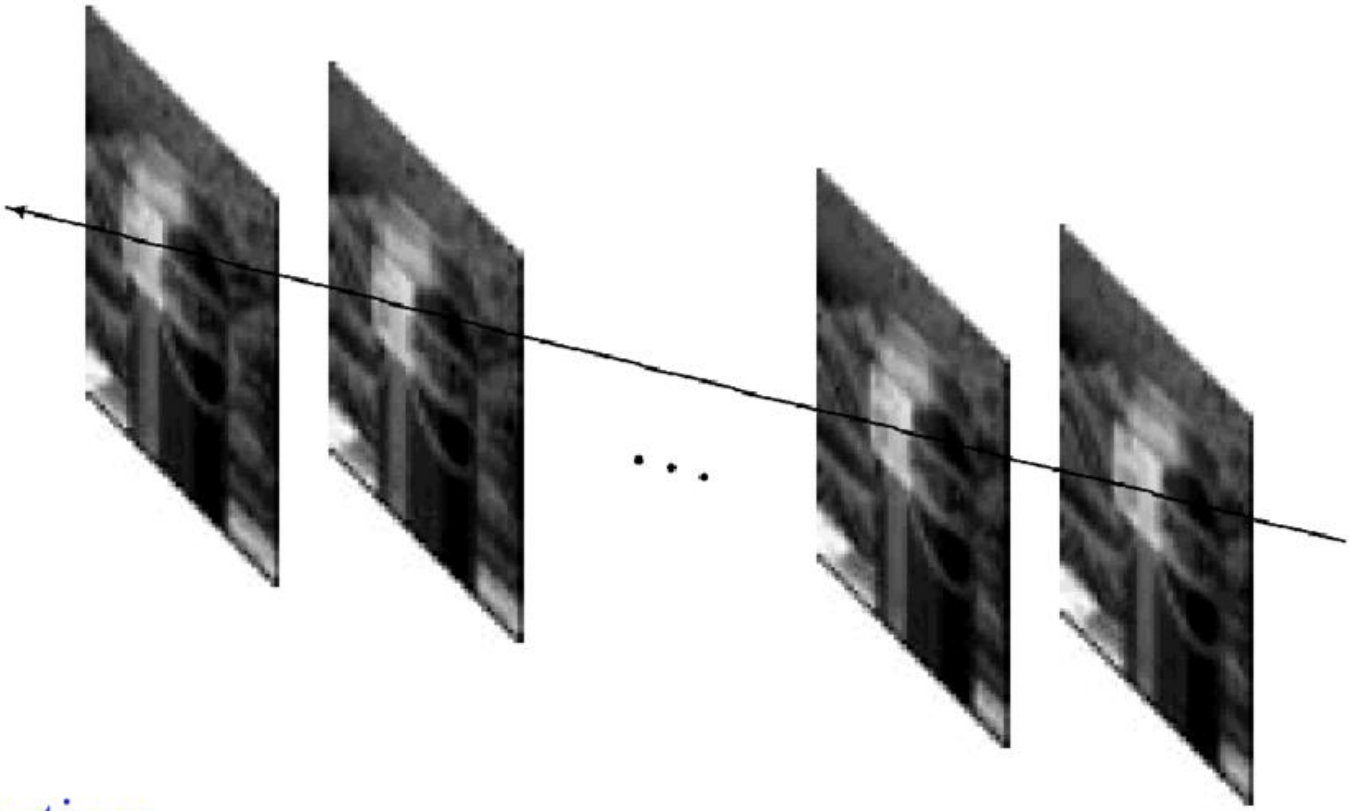


Assumption

* Neighboring points in the scene typically belong to the same surface and hence typically have similar motions.
* Since they also project to nearby points in the image, we expect spatial coherence in image flow.

* Slide from Michael Black, CS143 2003

Optical Flow Assumptions: Temporal Persistence



Assumption:

The image motion of a surface patch changes gradually over time.

* Slide from Michael Black, CS143 2003

Estimating Optical Flow

- Assume the image intensity I is constant

Time = t

Time = $t+dt$

$$I(x, y, t) = I(x+dx, y+dy, t+dt)$$

Brightness Constancy Equation

$$I(x,y,t) = I(x+dx, y+dy, t+dt)$$

First order Taylor Expansion

$$= I(x,y,t) + \frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy + \frac{\partial I}{\partial t}dt$$

Simplify notations:

$$I_x dx + I_y dy + I_t dt = 0$$

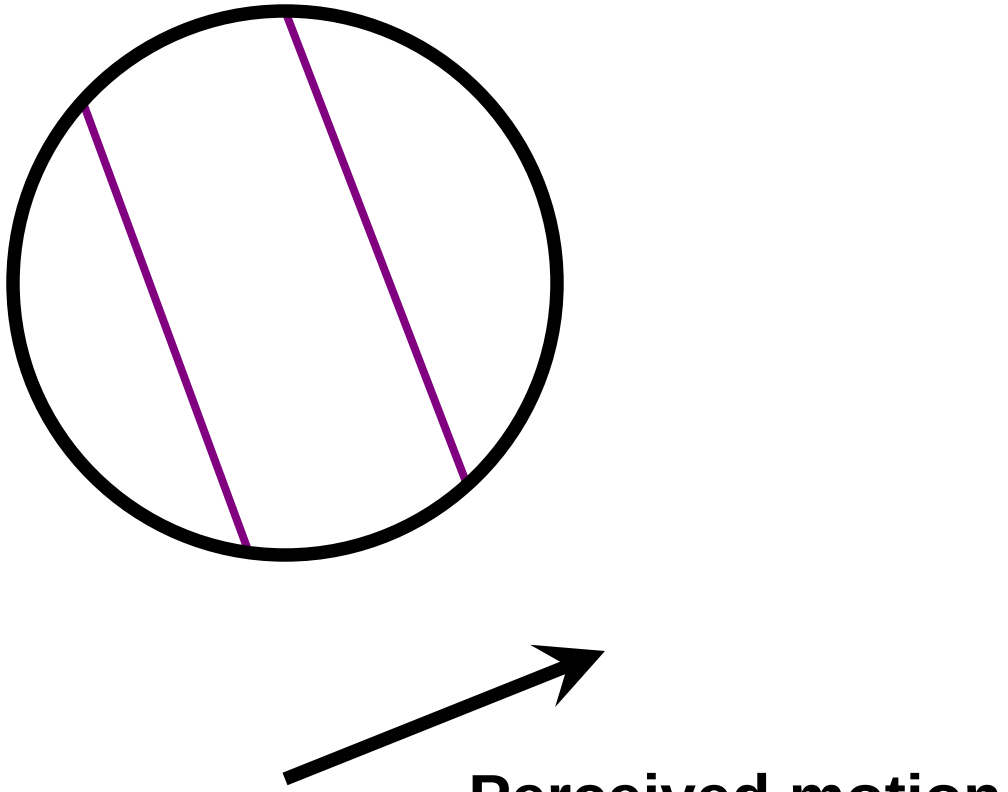
Divide by *dt* and denote:

$$u = \frac{dx}{dt} \qquad v = \frac{dy}{dt}$$

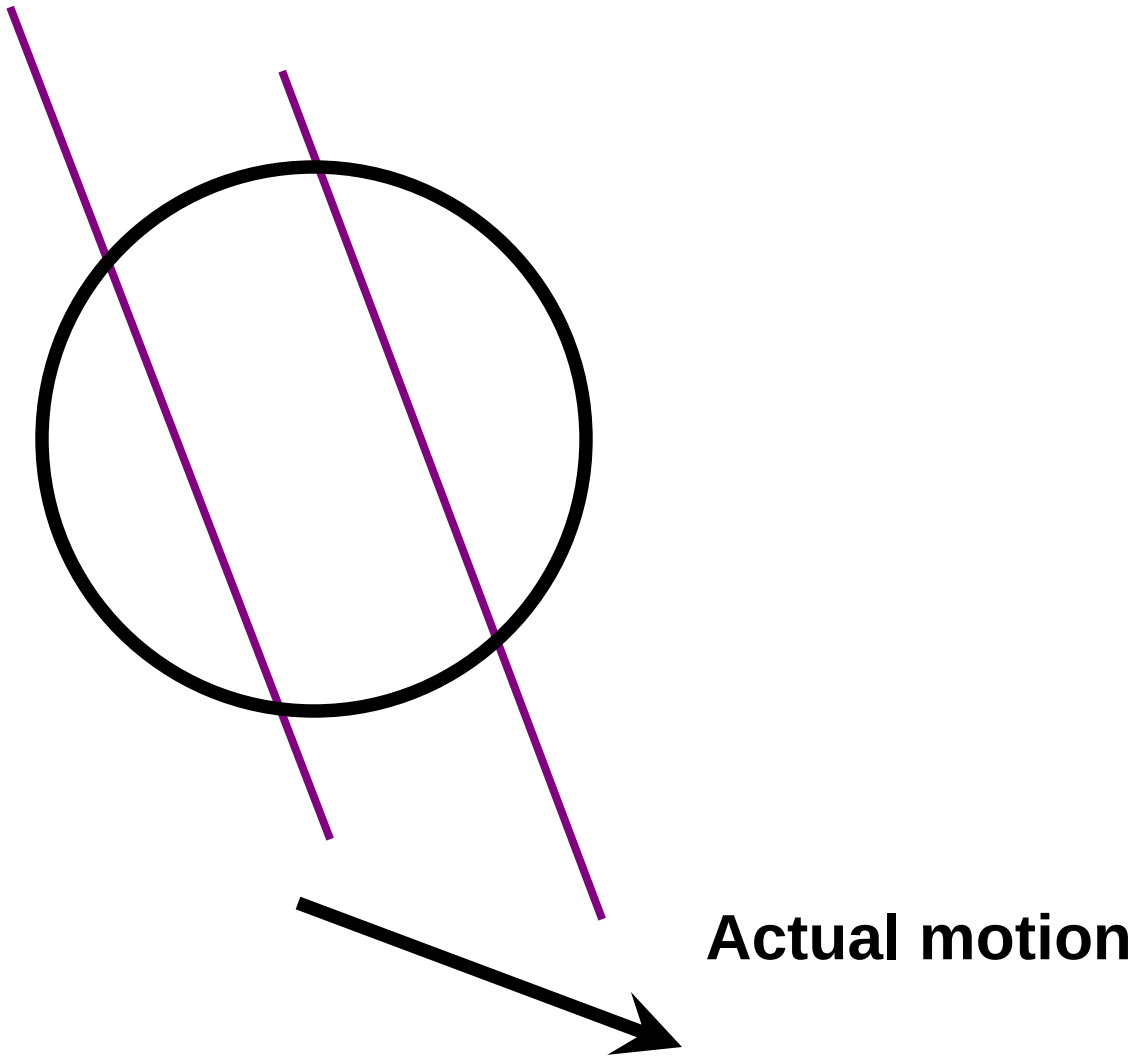
$$I_x u + I_y v = -I_t$$

Problem I: One equation, two unknowns

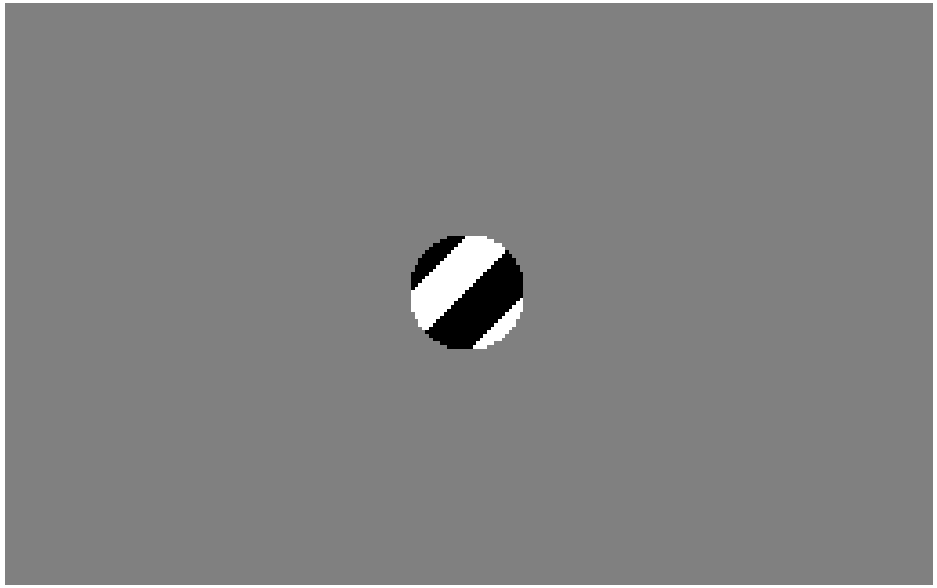
The aperture problem



The aperture problem

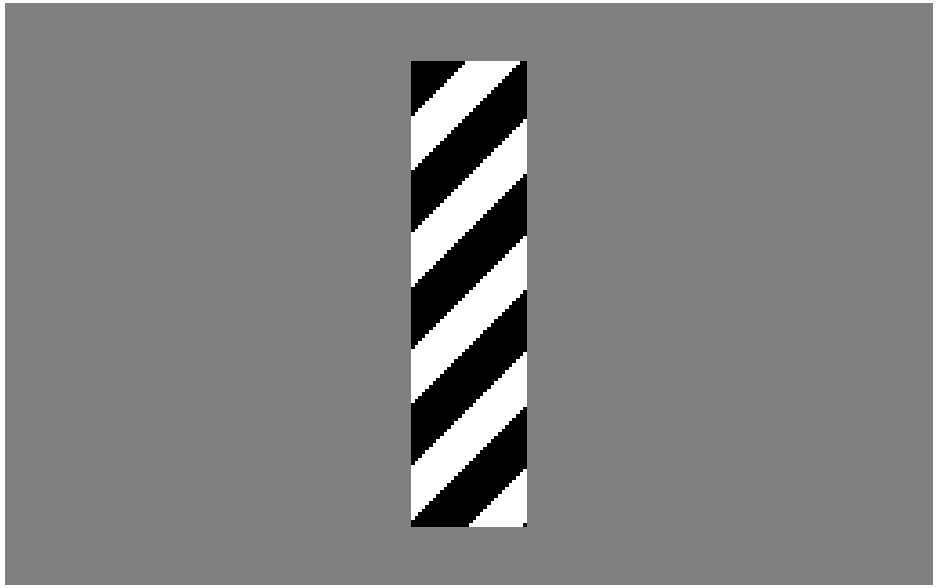


The barber pole illusion



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

The barber pole illusion



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

The barber pole illusion

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

Solving the aperture problem (grayscale image)

- How to get more equations for a pixel?
- **Spatial coherence constraint:** pretend the pixel's neighbors have the same (u,v)
  - If we use a 5x5 window, that gives us 25 equations per pixel

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

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

$$\underset{25\text{x}2}{A} \quad \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

Solving the aperture problem

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:

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

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

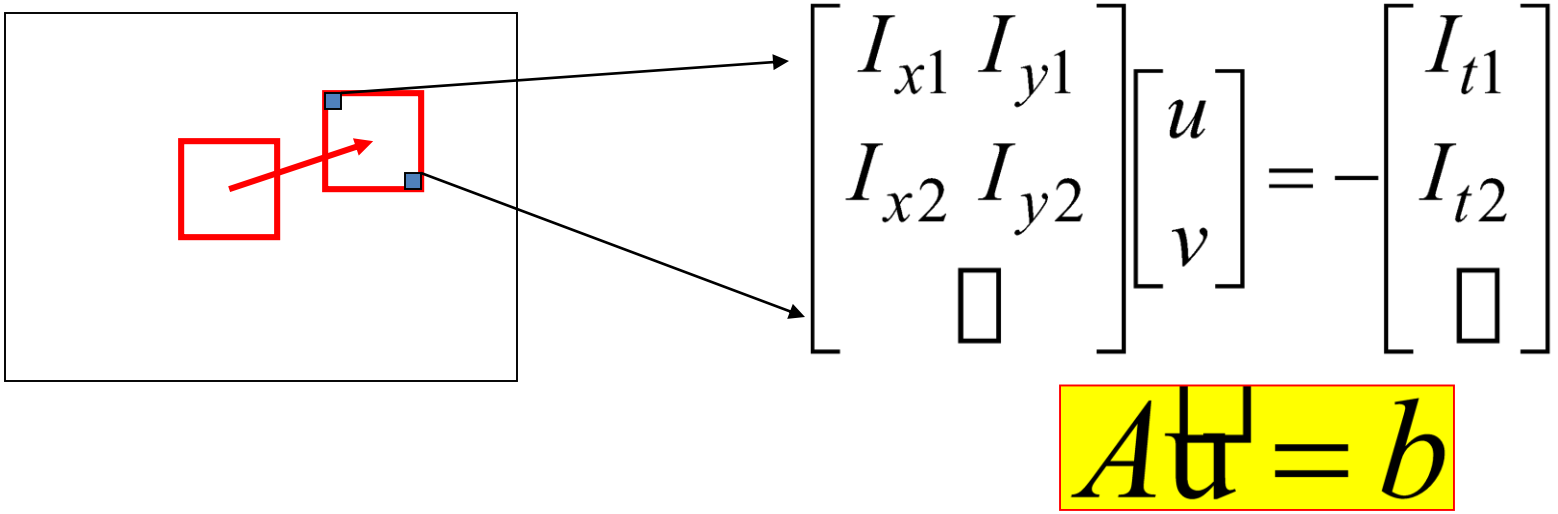
$A^T A$ $\qquad$ $A^T b$

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

Local smoothness
Lucas Kanade (1981)

$$I_x u + I_y v = -I_t \quad \Rightarrow \quad \begin{bmatrix} I_x & I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = -I_t$$

Assume constant (u,v) in small neighborhood



Lucas Kanade (1981)

Goal: Minimize $\|A\vec{u} - b\|^2$

Method: Least-Squares

$$A\vec{u} = b$$

$$\downarrow$$

$$\underbrace{A^T A}_{2\text{x}2}\ \underbrace{\vec{u}}_{2\text{x}1} = \underbrace{A^T b}_{2\text{x}1}$$

$$\downarrow$$

$$\vec{u} = \left(A^T A\right)^{-1} A^T b$$

Conditions for solvability

Optimal (u, v) satisfies Lucas-Kanade equation

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

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

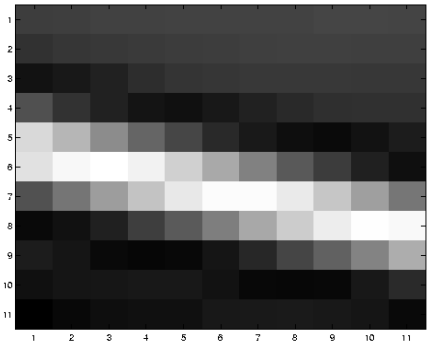
When is this solvable? I.e., what are good points to track?

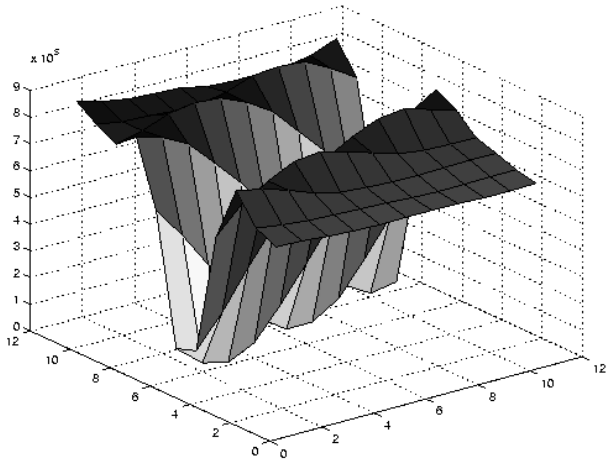
- $\mathbf{A^T A}$ should be invertible
- $\mathbf{A^T A}$ should not be too small due to noise
  - eigenvalues λ_1 and λ_2 of $\mathbf{A^T A}$ should not be too small
- $\mathbf{A^T A}$ should be well-conditioned
  - λ_1 / λ_2 should not be too large (λ_1 = larger eigenvalue)

Does this remind you of anything?

Criteria for Harris corner detector

Edge





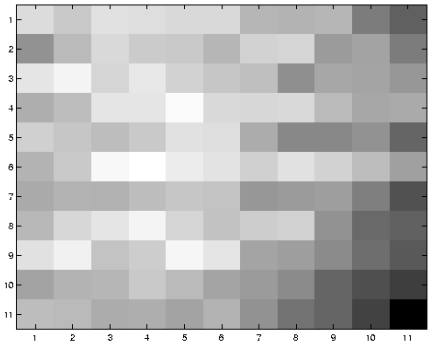
$$\sum \nabla I (\nabla I)^T$$

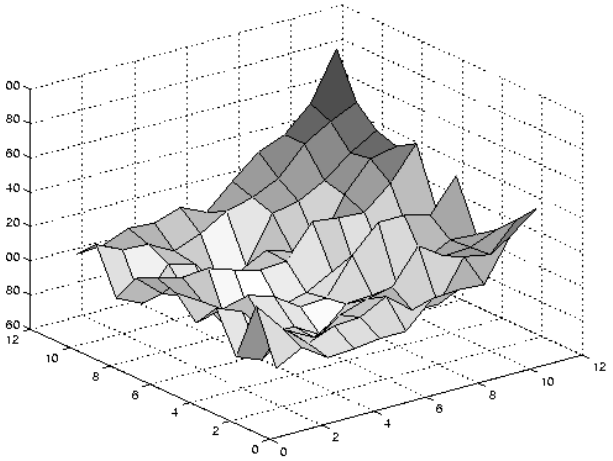
– large gradients, all the same

– large λ_1, small λ_2

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Low texture region





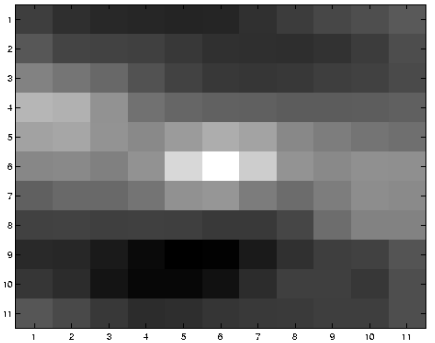
$$\sum \nabla I (\nabla I)^T$$

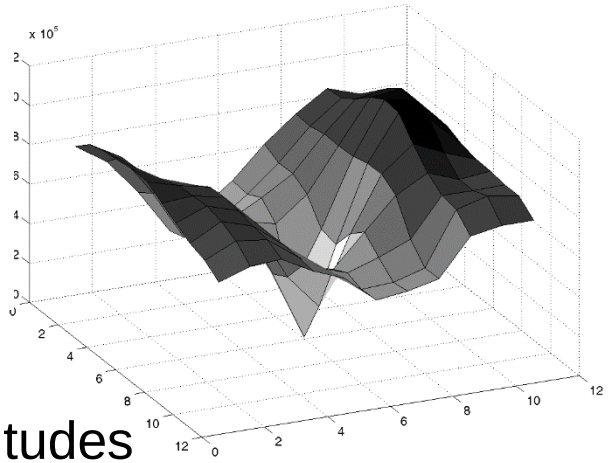
– gradients have small magnitude

– small λ_1, small λ_2

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

High textured region





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

– gradients are different, large magnitudes

– large λ_1, large λ_2

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Optical flow

www.societyofrobots.com/programming_computer_vision_tutorial_pt4.shtml

Errors in Lucas-Kanade

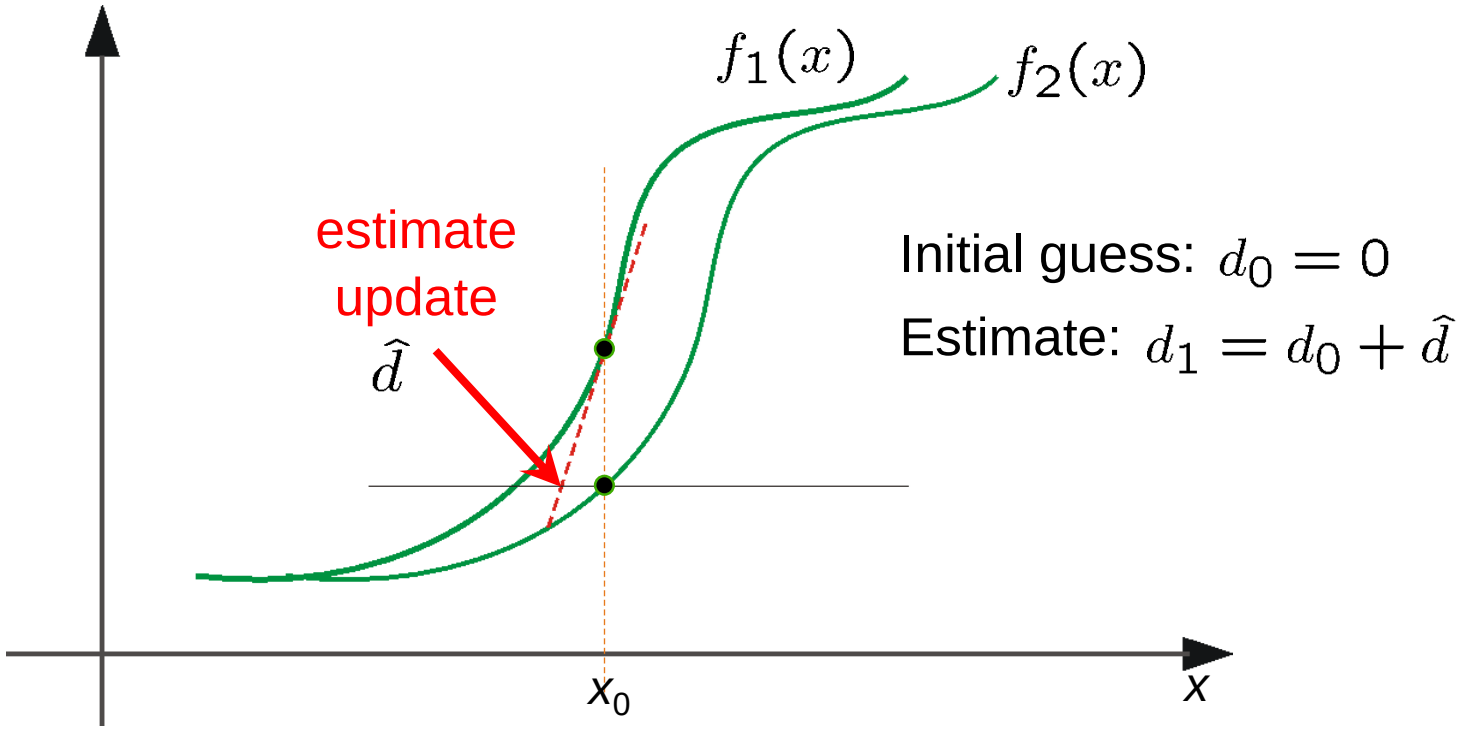
- The motion is large (larger than a pixel)
  - Iterative refinement
  - Coarse-to-fine estimation
  - Exhaustive neighborhood search (feature matching)
- A point does not move like its neighbors
  - Motion segmentation
- Brightness constancy does not hold
  - Exhaustive neighborhood search with normalized correlation

Iterative Refinement

- Estimate velocity at each pixel using one iteration of Lucas and Kanade estimation
- Warp one image toward the other using the estimated flow field

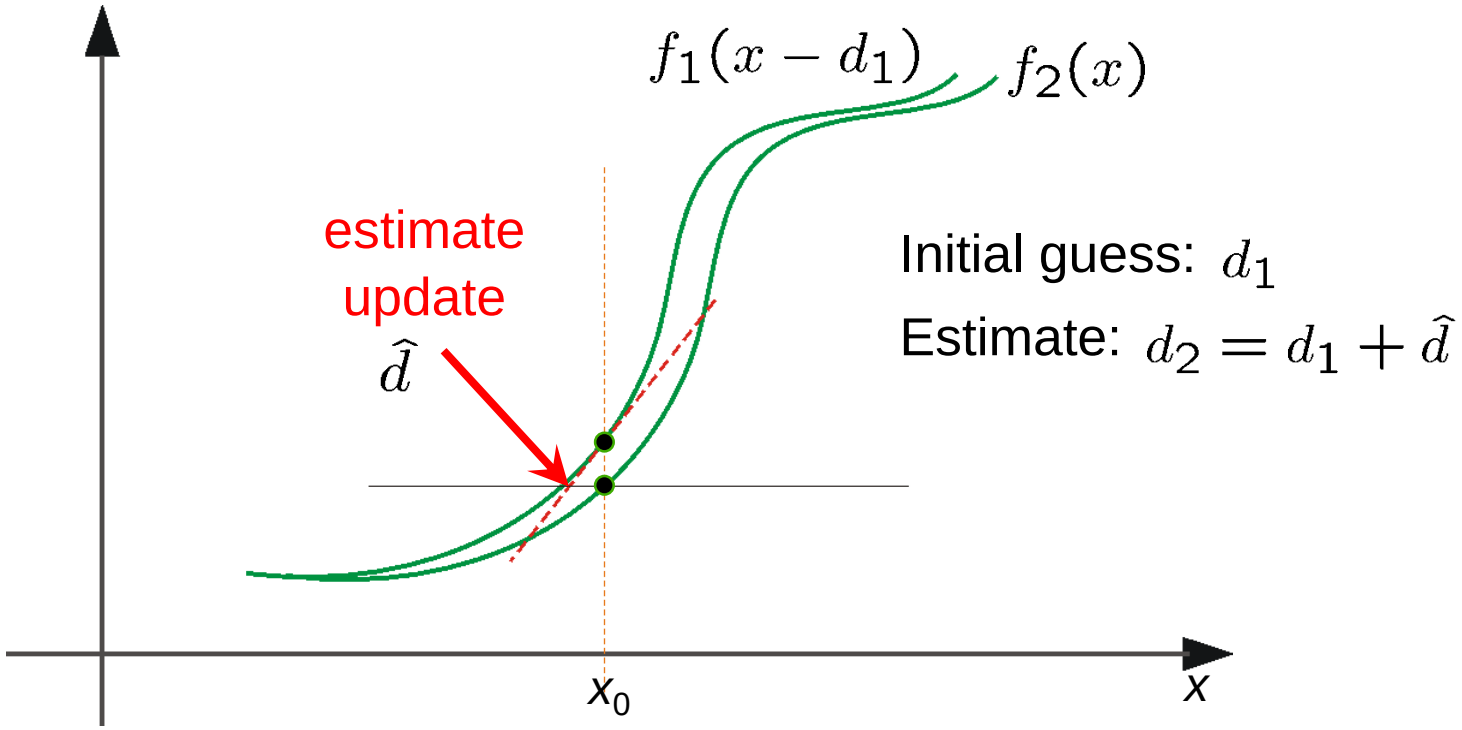
  (easier said than done)
- Refine estimate by repeating the process

Optical Flow: Iterative Estimation

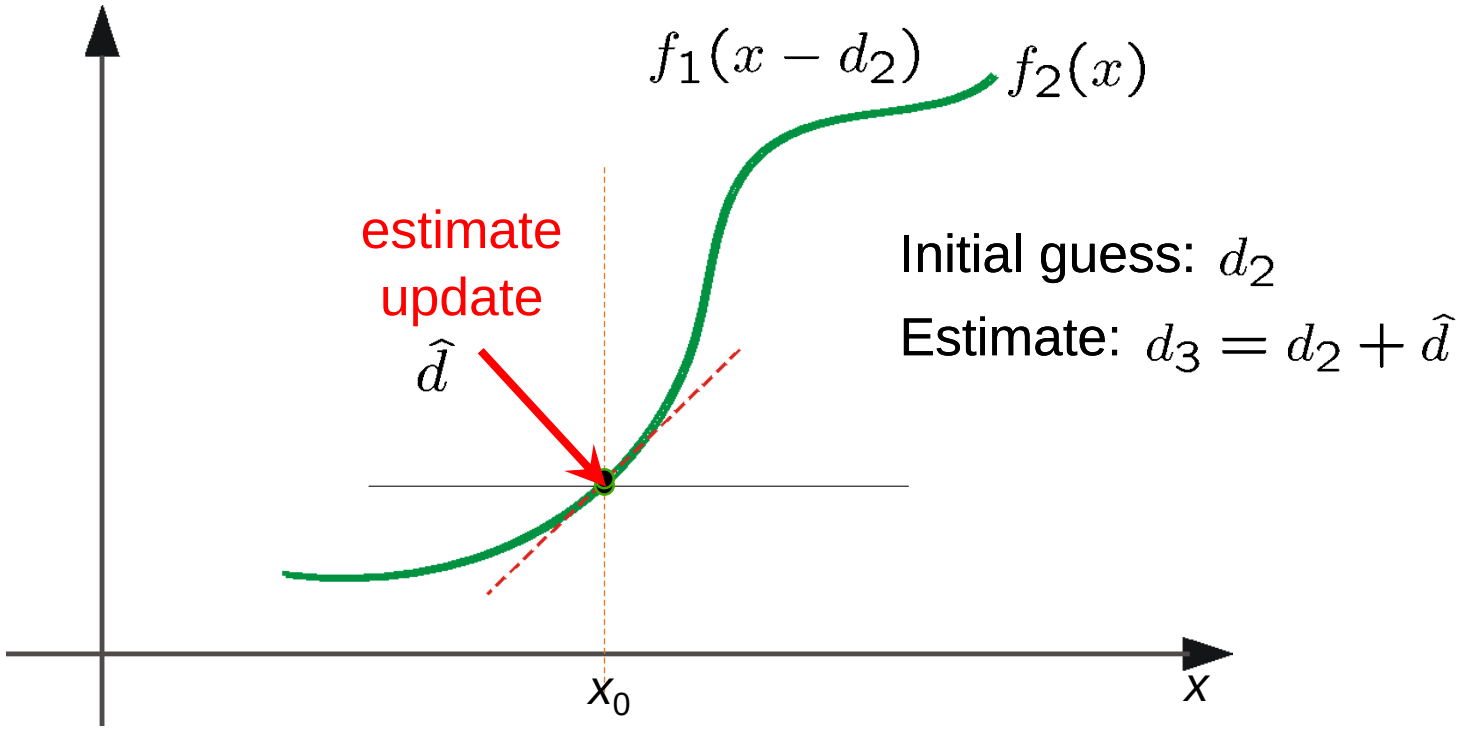


(using *d* for *displacement* here instead of *u*)

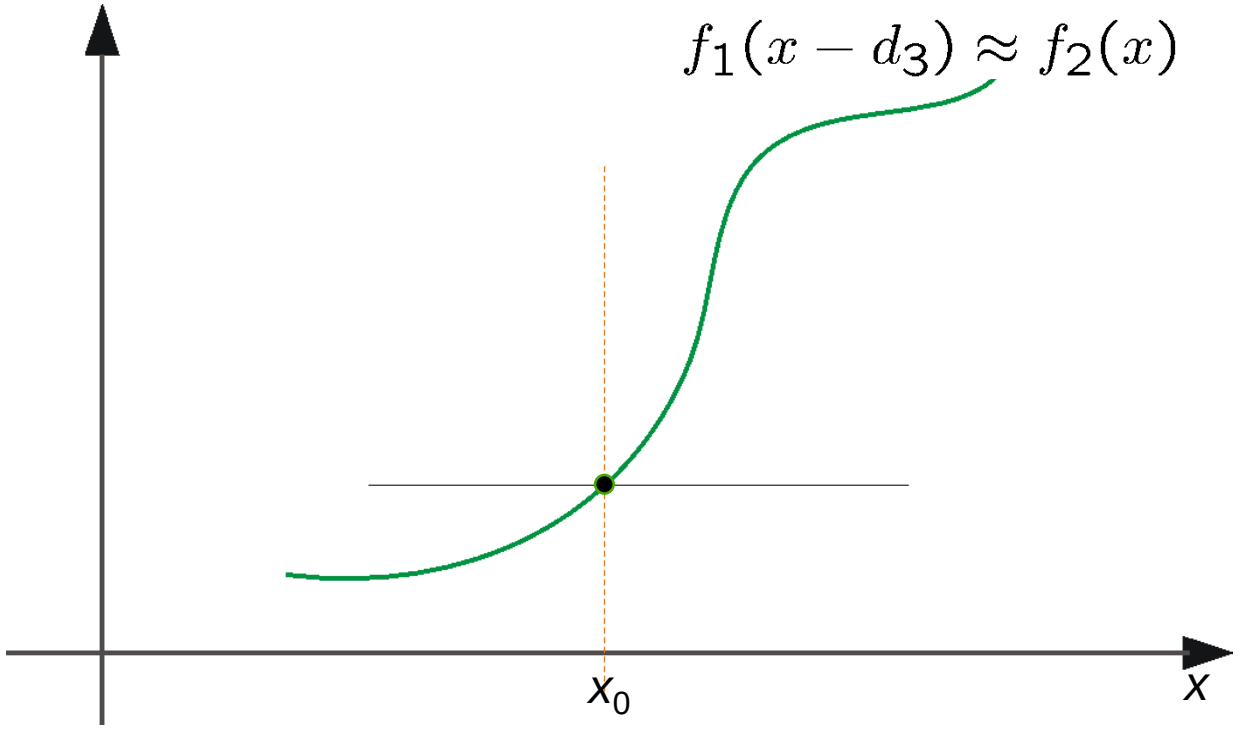
Optical Flow: Iterative Estimation



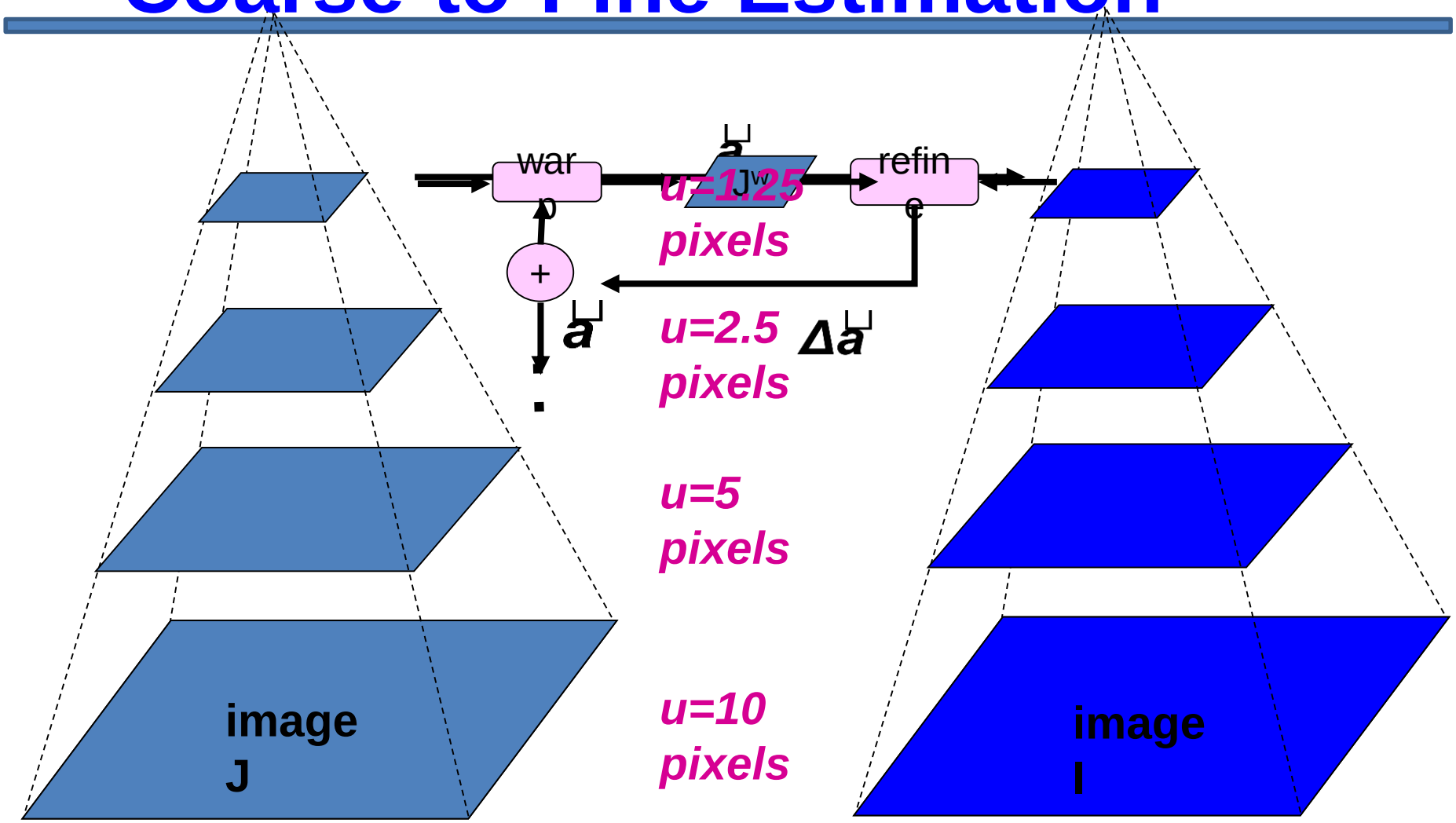
Optical Flow: Iterative Estimation



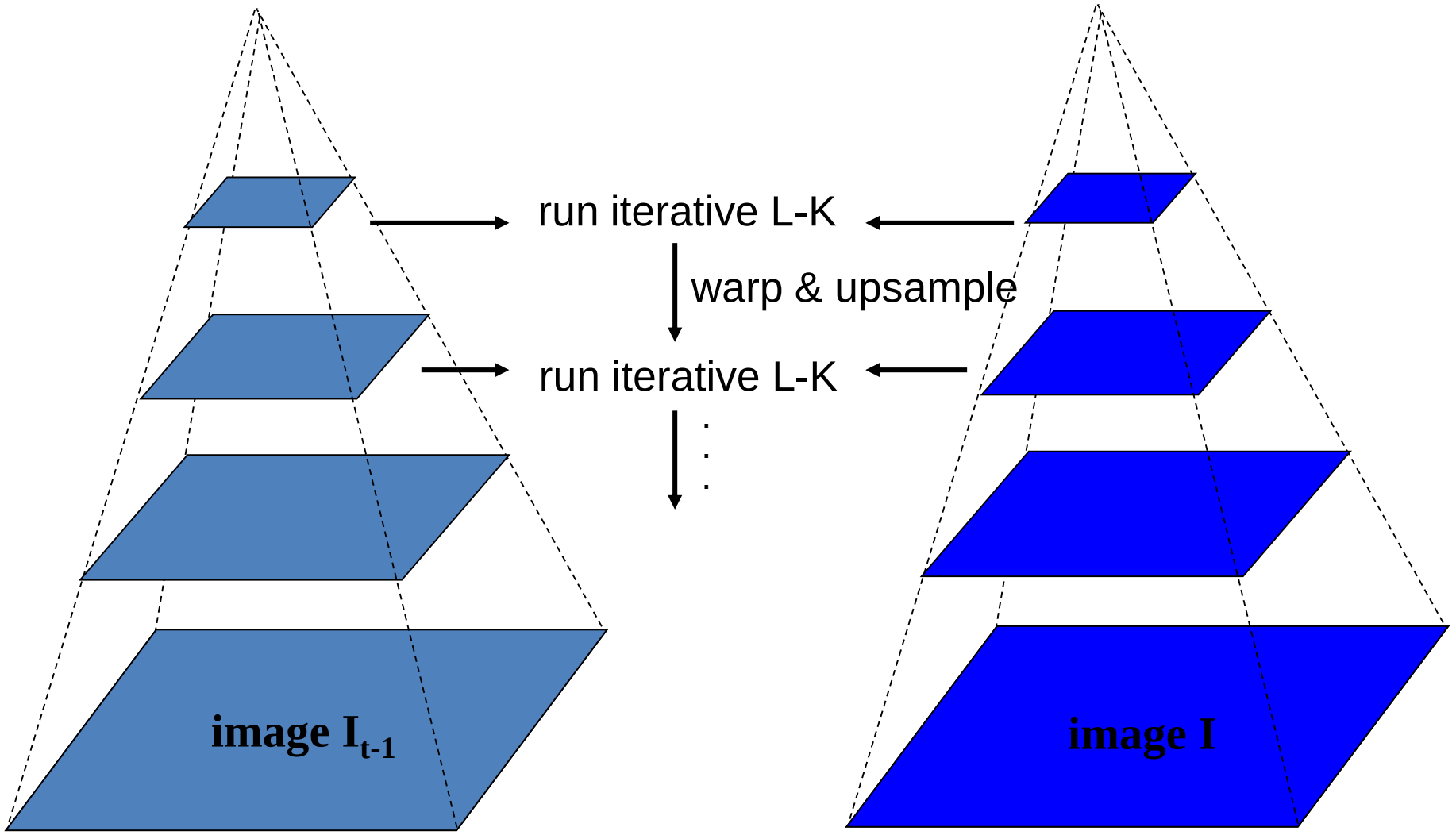
Optical Flow: Iterative Estimation



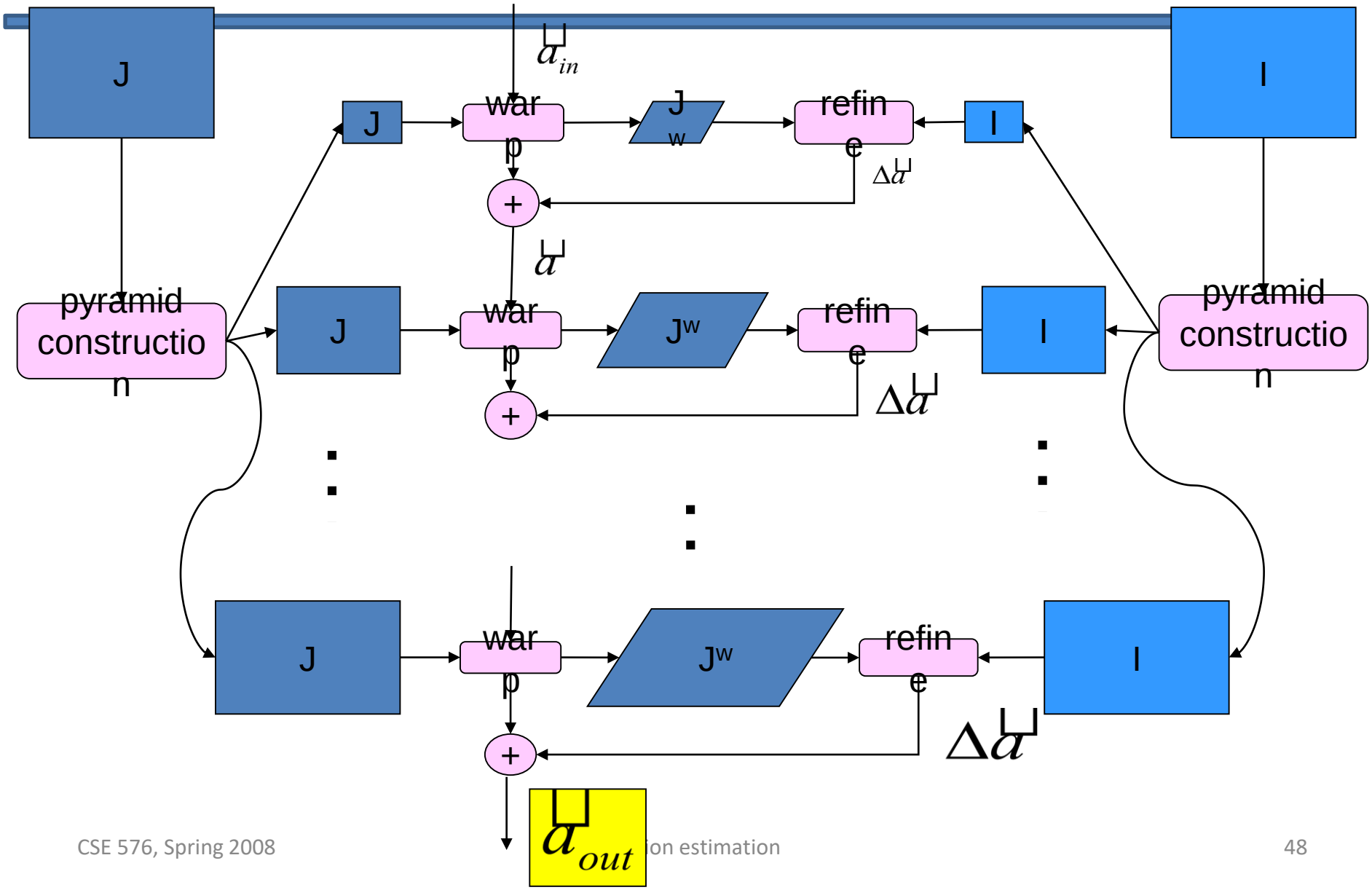
Coarse-to-Fine Estimation



Coarse-to-fine optical flow estimation



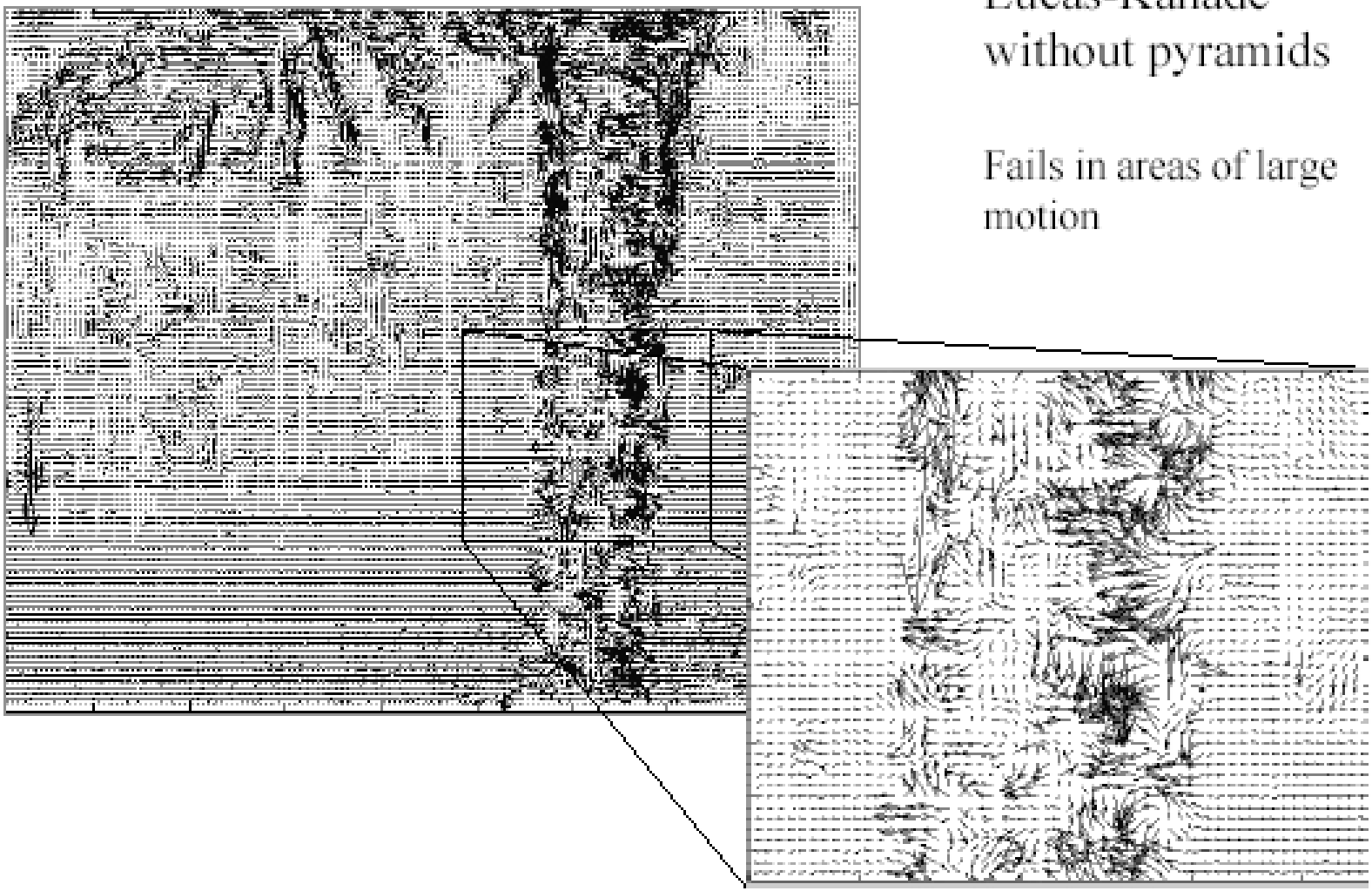
Coarse-to-Fine Estimation



Multi-resolution Lucas Kanade Algorithm

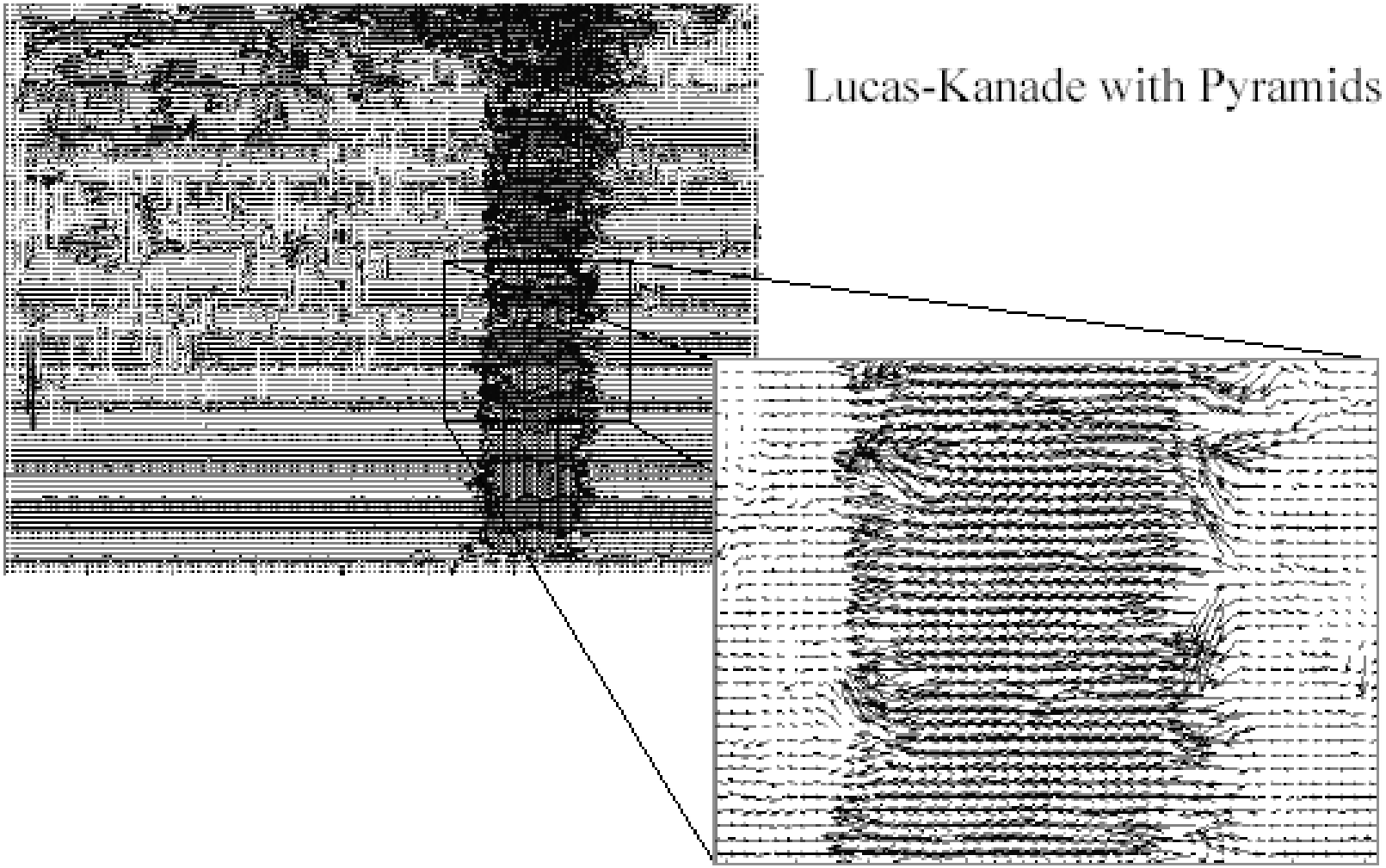
- Compute 'simple' LK at highest level
- At level i
  - Take flow u_{i-1}, v_{i-1} from level i-1
  - bilinear interpolate it to create u_i^*, v_i^* matrices of twice resolution for level i
  - multiply u_i^*, v_i^* by 2
  - compute f_t from a block displaced by $u_i^*(x,y)$, $v_i^*(x,y)$
  - Apply LK to get $u_i'(x, y)$, $v_i'(x, y)$ (the correction in flow)
  - Add corrections u_i' v_i', *i.e.* $u_i = u_i^* + u_i'$, $v_i = v_i^* + v_i'$.

Optical Flow Results



* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

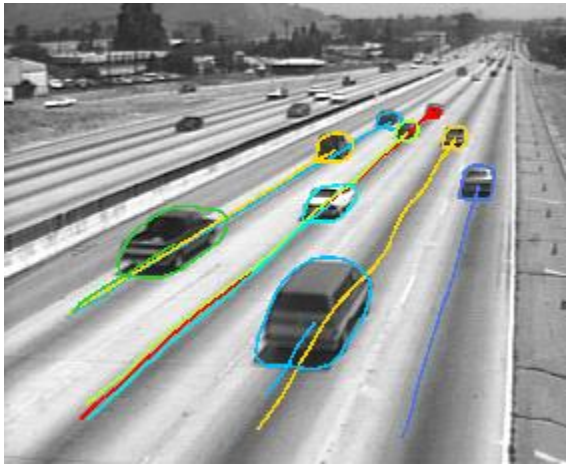
Optical Flow Results



* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Feature tracking

- So far, we have only considered optical flow estimation in a pair of images
- If we have more than two images, we can compute the optical flow from each frame to the next
- Given a point in the first image, we can in principle reconstruct its path by simply “following the arrows”



www.societyofrobots.com/programming_computer_vision_tutorial_pt4.shtml
http://docs.opencv.org/trunk/d7/d8b/tutorial_py_lucas_kanade.html

Tracking challenges

- Ambiguity of optical flow
  - Need to find good features to track
- Large motions, changes in appearance, occlusions, disocclusions
  - Need mechanism for deleting, adding new features
- Drift – errors may accumulate over time
  - Need to know when to terminate a track

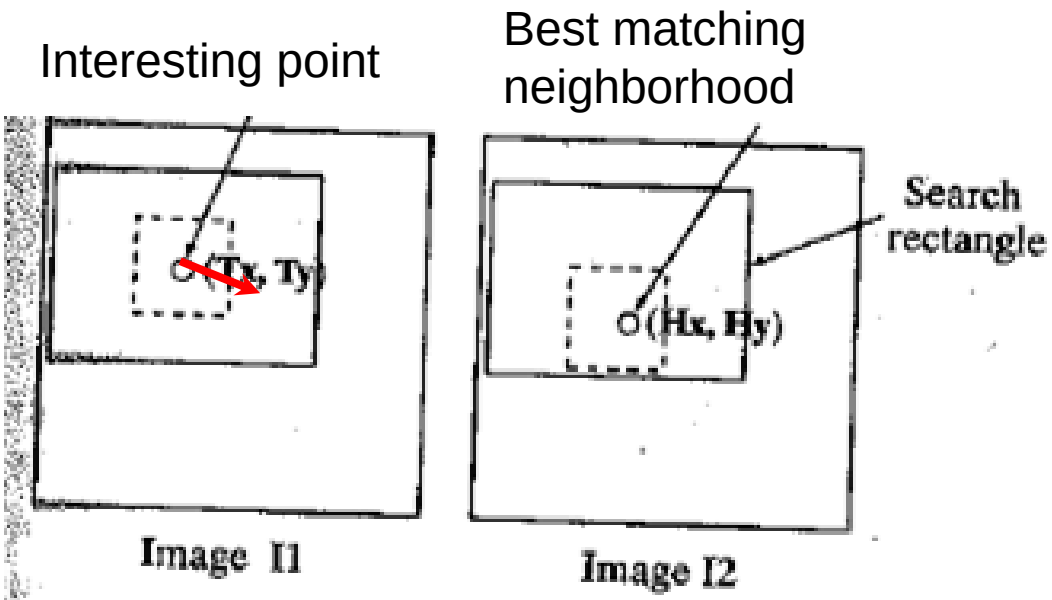
Tracking over many frames

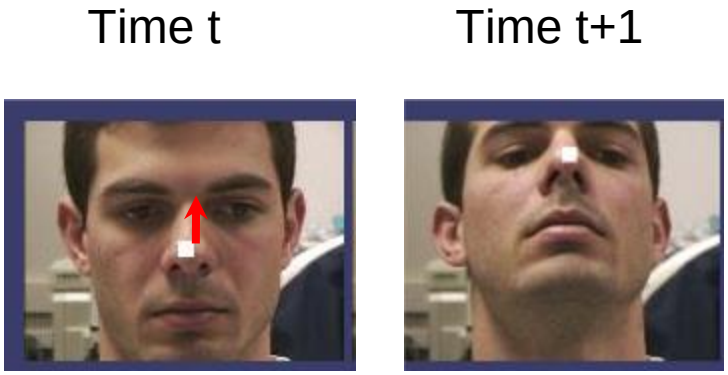
- Select features in first frame
- For each frame:
  - Update positions of tracked features
    - Discrete search or Lucas-Kanade (or a combination of the two)
  - Terminate inconsistent tracks
    - Compute similarity with corresponding feature in the previous frame or in the first frame where it's visible
  - Find more features to track

What are good features to track?

- Harris corner detector
  - Can measure quality of features from just a single image
  - Automatically select candidate “templates”

Feature-based matching for motion





Shi-Tomasi feature tracker

1. Find good features (min eigenvalue of 2×2 Hessian)
2. Use Lucas-Kanade to track with pure translation
3. Use affine registration with first feature patch
4. Terminate tracks whose dissimilarity gets too large
5. Start new tracks when needed

A Camera Mouse

- Video interface: use feature tracking as mouse replacement

- User clicks on the feature to be tracked
- Take the 15x15 pixel square of the feature
- In the next image do a search to find the 15x15 region with the highest correlation
- Move the mouse pointer accordingly
- Repeat in the background every 1/30th of a second

James Gips and Margrit Betke
http://www.bc.edu/schools/csom/eagleeyes/

Implementation issues

- Window size
  - Small window more sensitive to noise and may miss larger motions (without pyramid)
  - Large window more likely to cross an occlusion boundary (and it's slower)
  - 15x15 to 31x31 seems typical

- Weighting the window
  - Common to apply weights so that center matters more (e.g., with Gaussian)

Optical Flow: Iterative Estimation

- Some Implementation Issues:
  - Warping is not easy (ensure that errors in warping are smaller than the estimate refinement)
  - Warp one image, take derivatives of the other so you don't need to re-compute the gradient after each iteration.
  - Often useful to low-pass filter the images before motion estimation (for better derivative estimation, and linear approximations to image intensity)

Summary

- Motion field: 3d motions projected to 2d images; dependency on depth
- Solving for motion with
  - sparse feature matches
  - dense optical flow
- Optical flow – key ideas
  - Brightness constancy assumption
  - Aperture problem
  - Solution with spatial coherence assumption
  - Coarse-to-fine registration