Motion & Optical Flow

·AIPKU·

人工智能引论实践课 计算机视觉小班

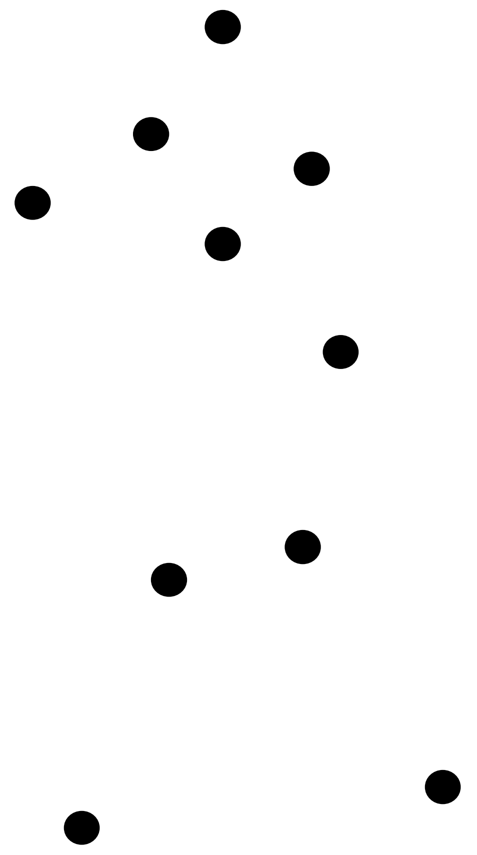
主讲人：刘家瑛

We live in a moving world

- Perceiving, understanding and predicting motion is an important part of our daily lives

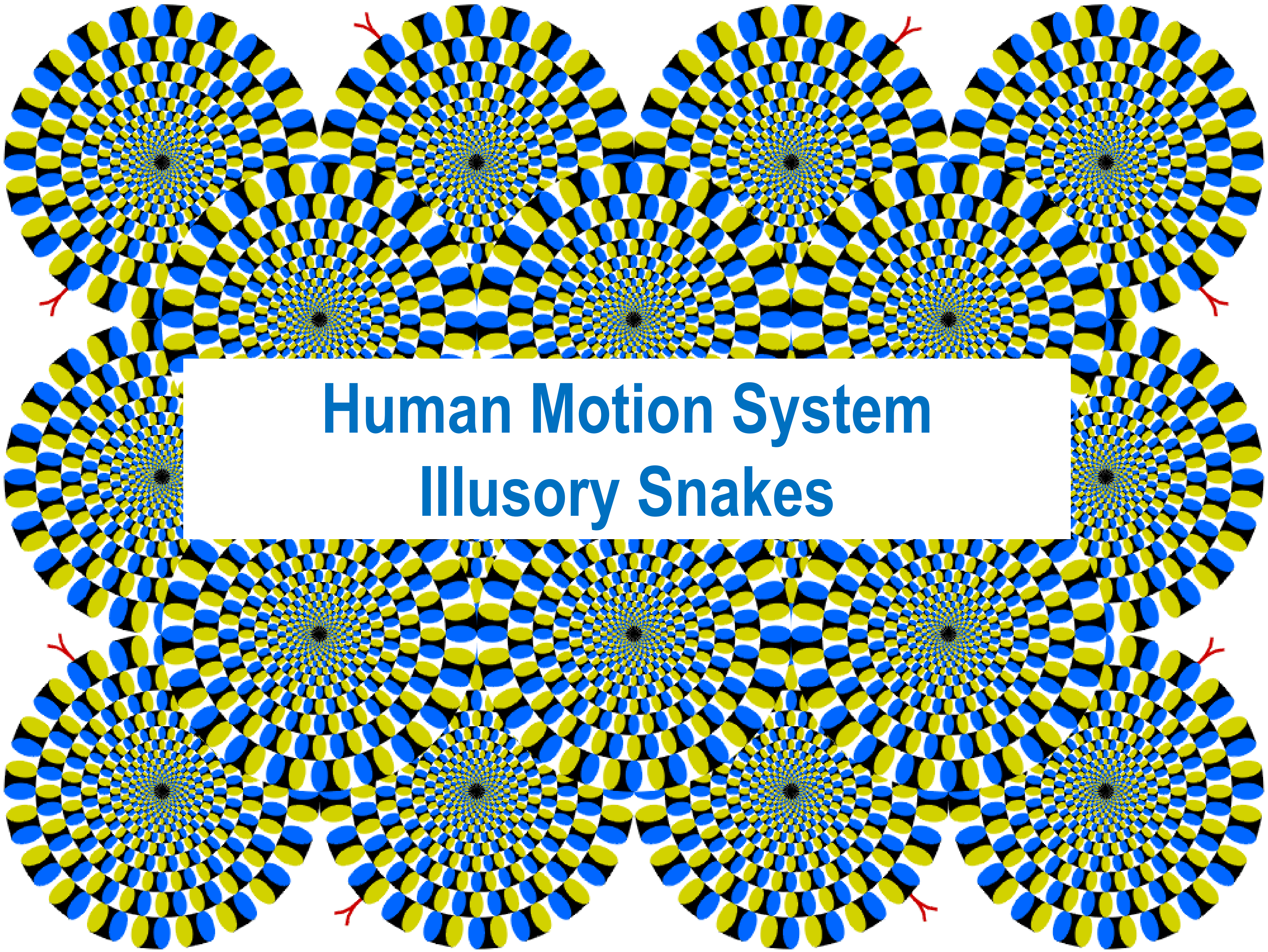
Motion and perceptual organization

- 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.

Seeing motion from a static picture



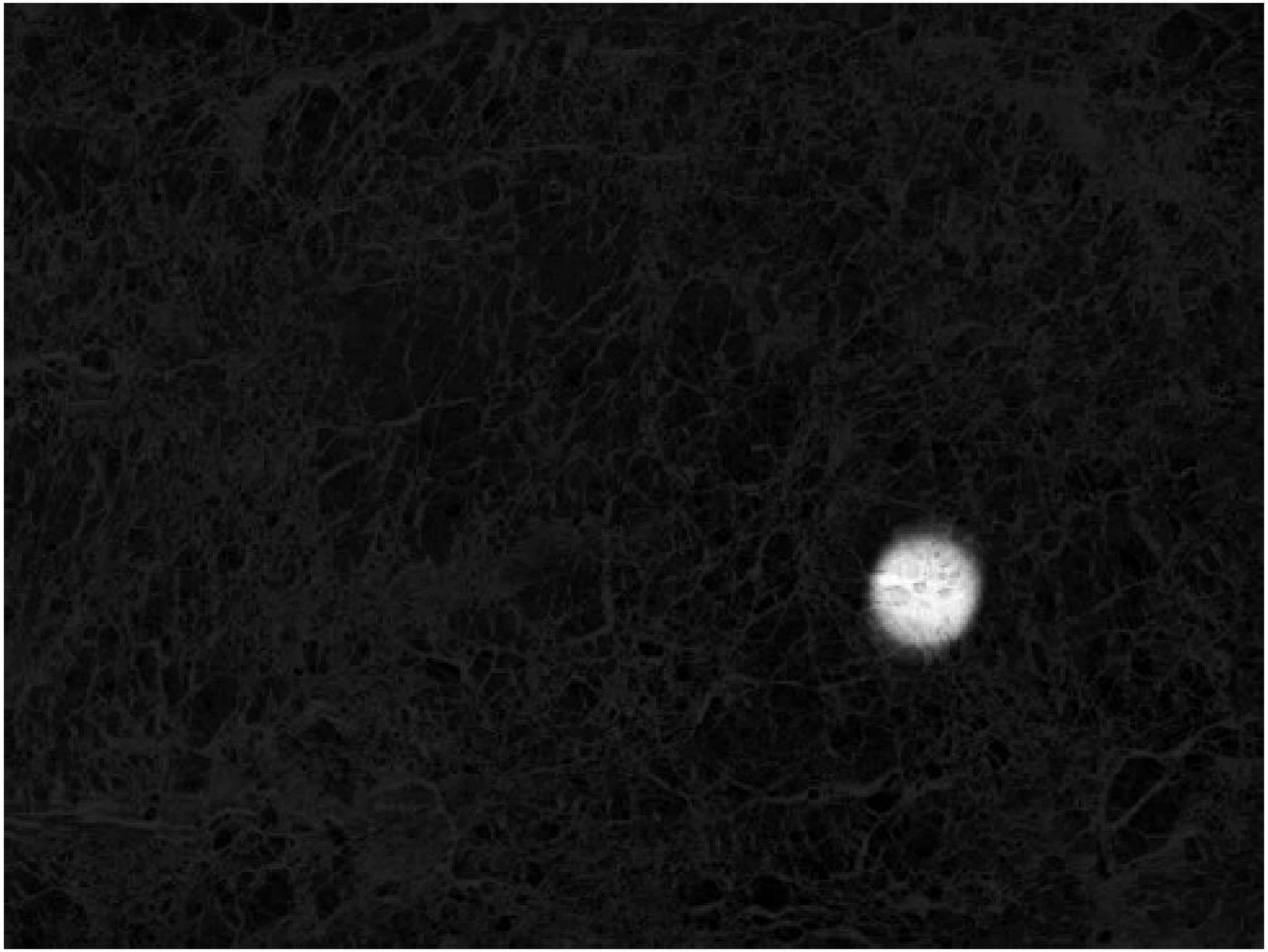
Human Motion System
Illusory Snakes

http://www.ritsumei.ac.jp/~akitaoka/index-e.html

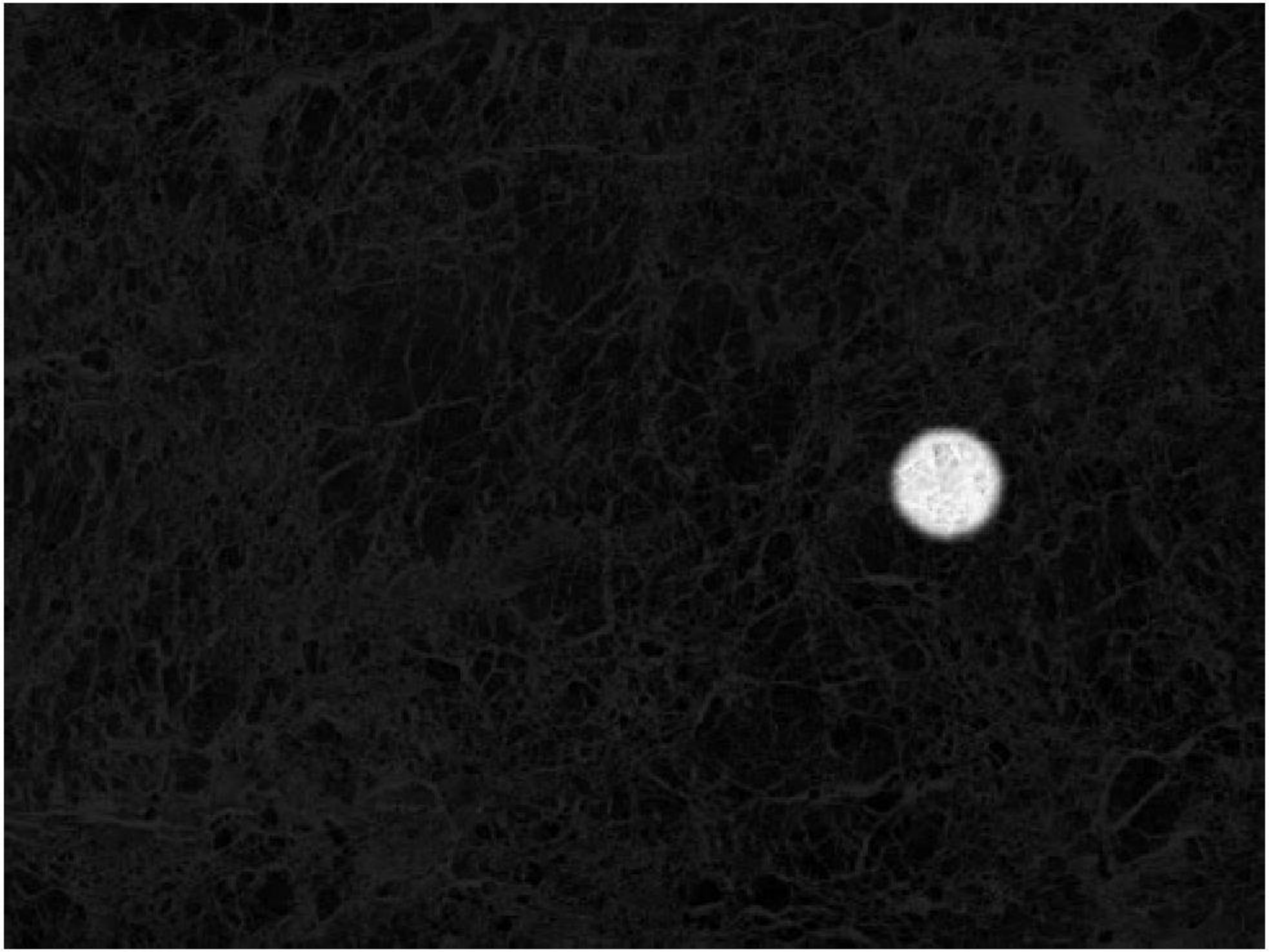
How is this possible?

- The true mechanism is to be revealed
- FMRI data suggest that illusion is related to some component of eye movements
- We don’t expect computer vision to “see” motion from these stimuli, yet

What do you see?



In fact, ...



The cause of motion

- Three factors in imaging process

  - Light

  - Object

  - Camera

- Varying either of them causes motion

  - Static camera, moving objects (surveillance)

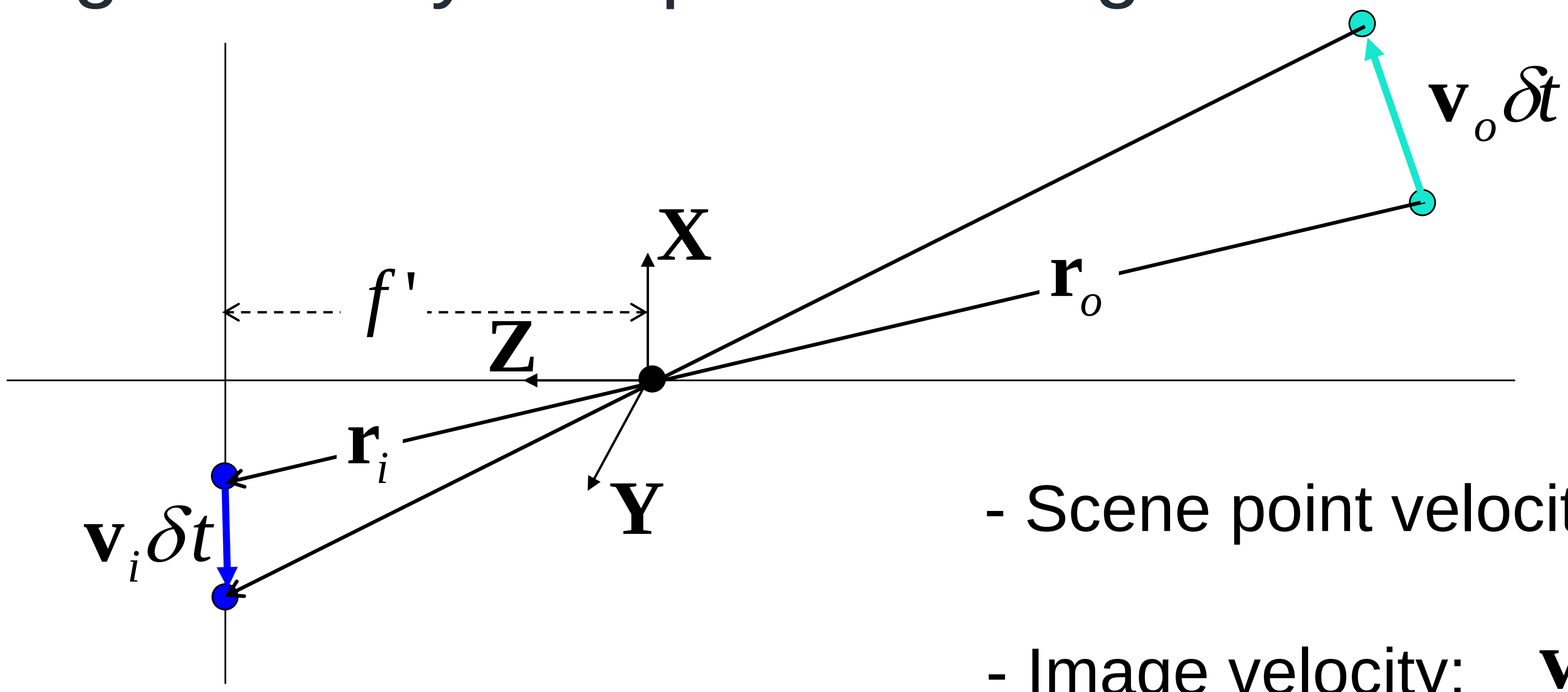
  - Moving camera, static scene (3D capture)

  - Moving camera, moving scene (sports, movie)

  - Static camera, moving objects, moving light (time lapse)

Motion Field

- Image velocity of a point moving in the scene



- Scene point velocity: $\mathbf{v}_o = \dfrac{d\mathbf{r}_o}{dt}$

- Image velocity: $\mathbf{v}_i = \dfrac{d\mathbf{r}_i}{dt}$

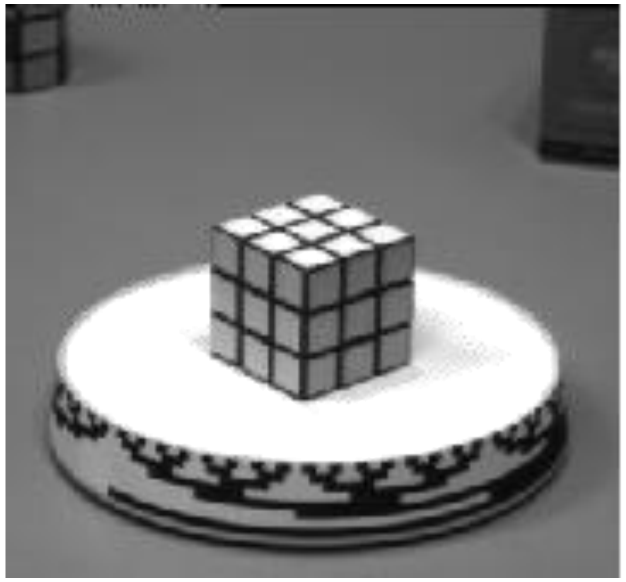
- Perspective projection: $\dfrac{1}{f\,'}\mathbf{r}_i = \dfrac{\mathbf{r}_o}{\mathbf{r}_o \cdot \mathbf{Z}}$

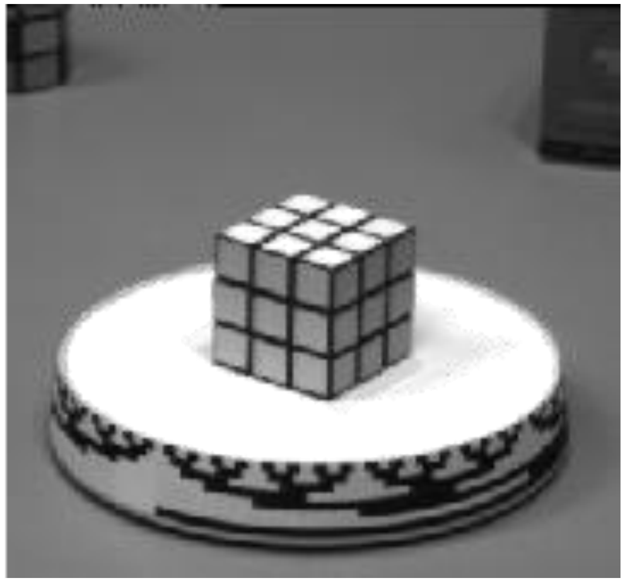
- Motion field

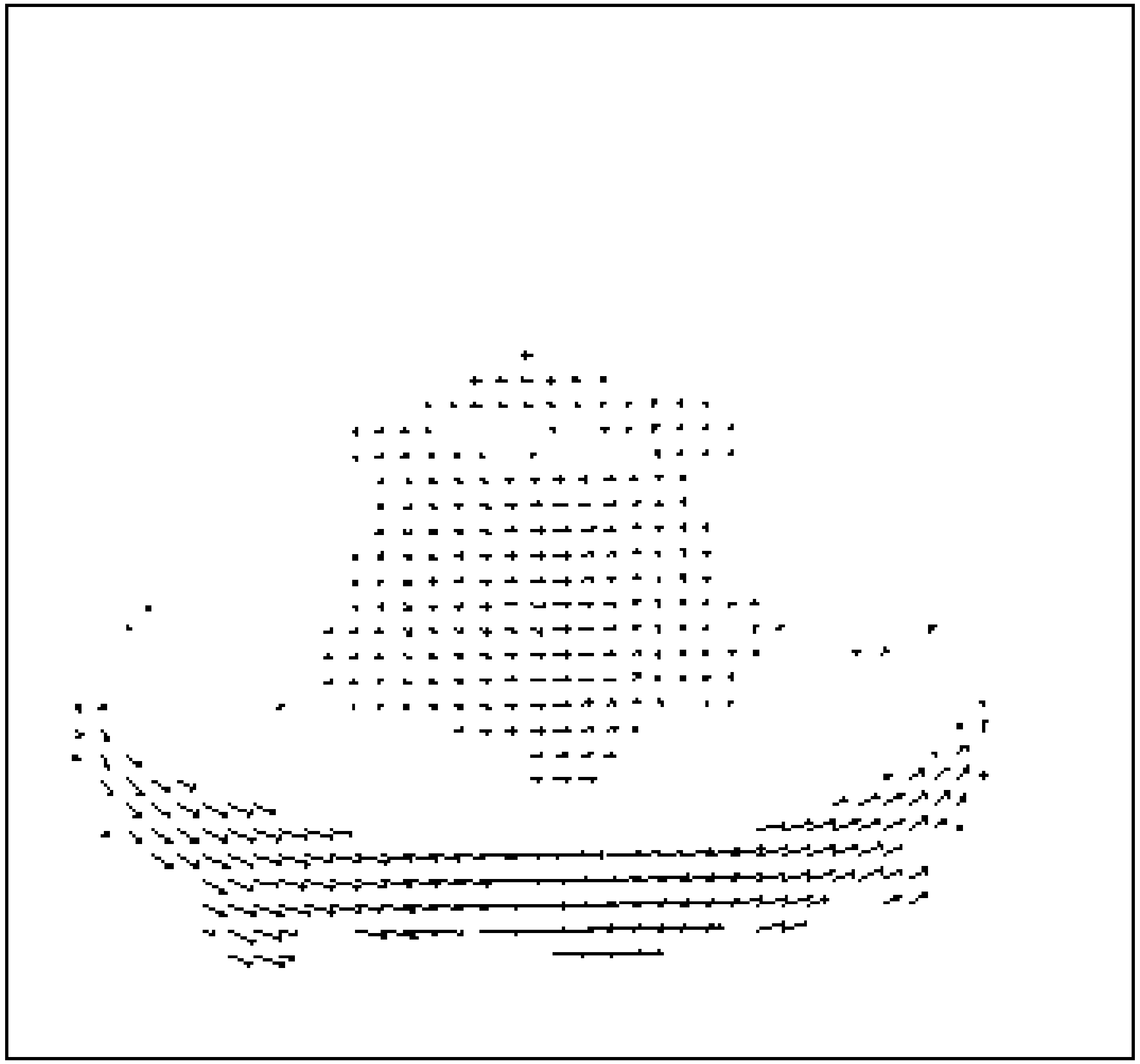
$$\boxed{\mathbf{v}_i = \frac{d\mathbf{r}_i}{dt} = f\,'\frac{(\mathbf{r}_o \cdot \mathbf{Z})\mathbf{v}_o - (\mathbf{v}_o \cdot \mathbf{Z})\mathbf{r}_o}{(\mathbf{r}_o \cdot \mathbf{Z})^2}}$$

Optical Flow

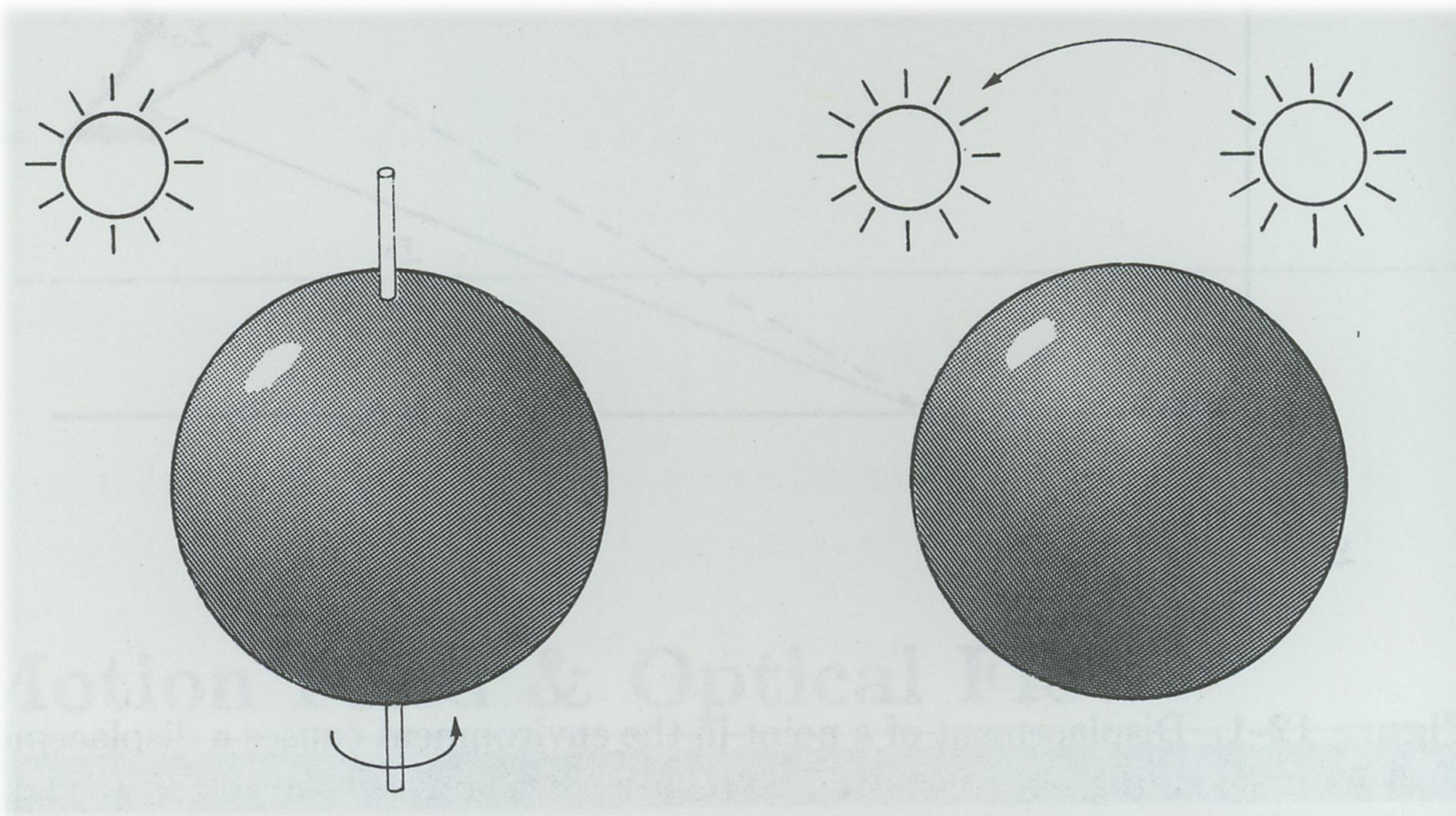
- Motion of brightness pattern in the image
- **Ideally** Optical flow = Motion field







Optical Flow $\neq$ Motion Field



(a)
Motion field exists
but no optical flow

(b)
No motion field
but shading changes

Why Estimate Motion?

- Lots of uses

  - Track object behavior

  - Correct for camera jitter (stabilization)

  - Align images (mosaics)

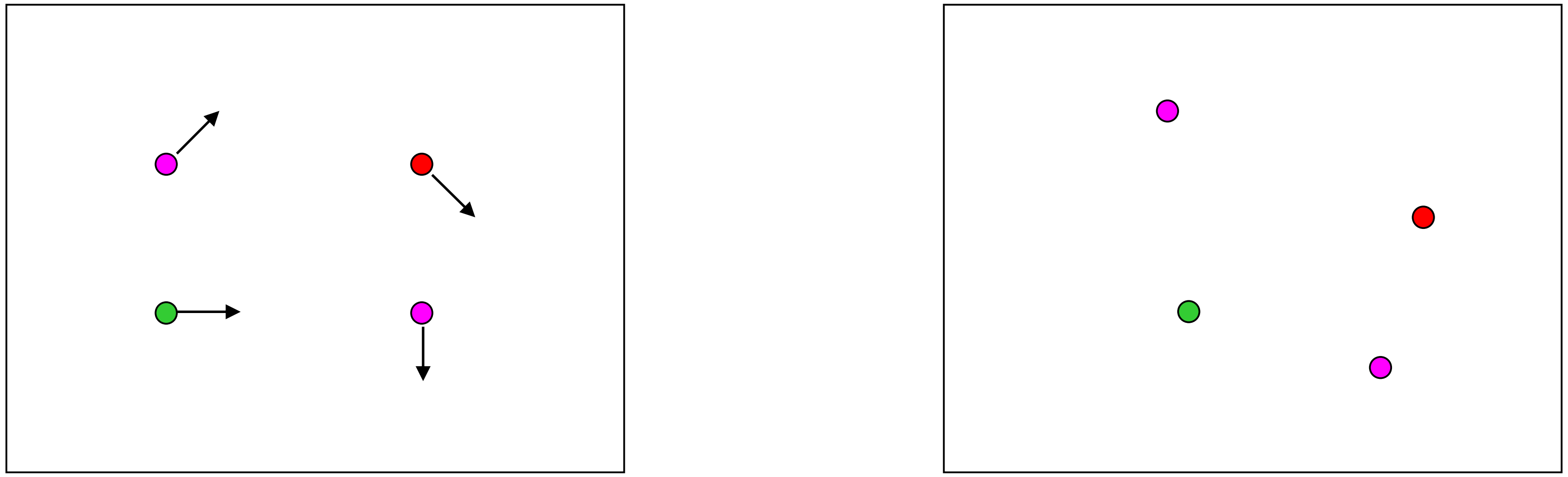
  - 3D shape reconstruction

  - Special effects

- Video Stabilization

Problem Definition: Optical Flow

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



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

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

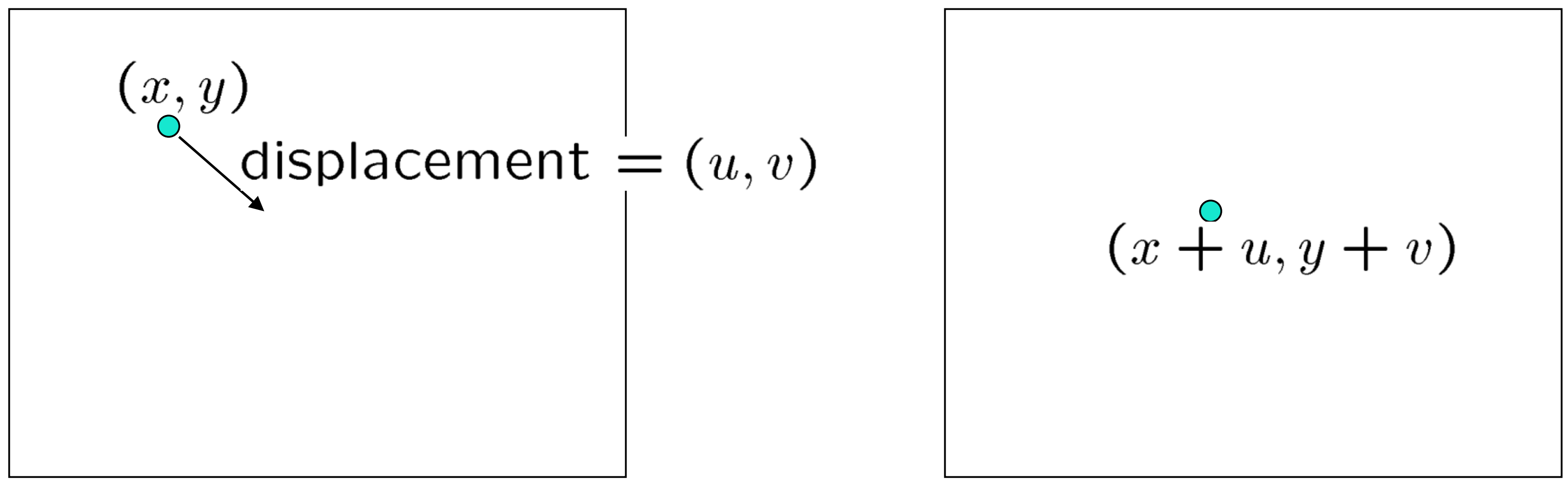
Key assumptions

- **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

- Let's look at these constraints more closely



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

- 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

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

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

Optical Flow Equation

- Q: how many unknowns and equations per pixel?

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

2 Unknowns, ONE 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

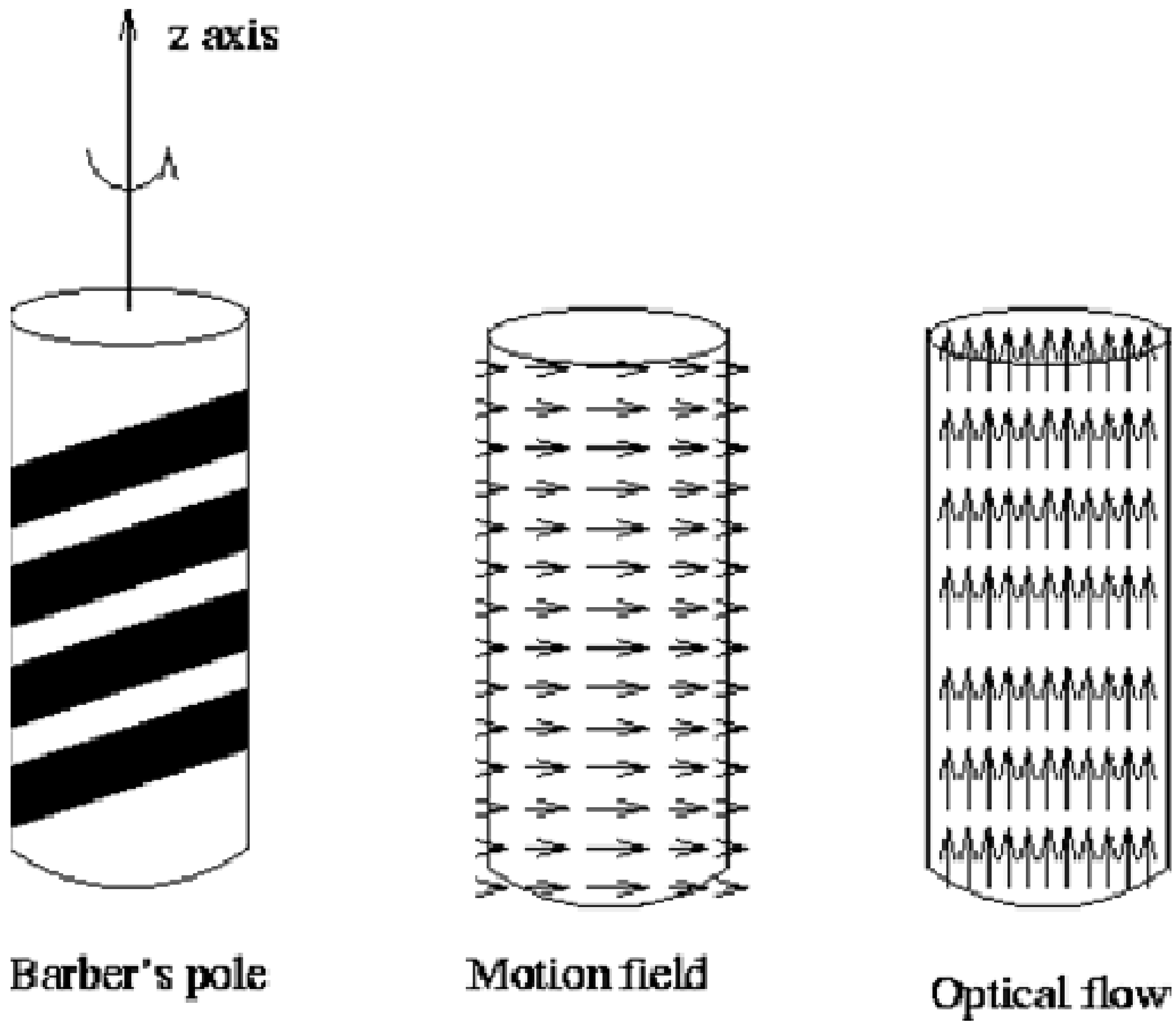
This explains the Barber Pole illusion

http://www.liv.ac.uk/~marcob/Trieste/barberpole.html

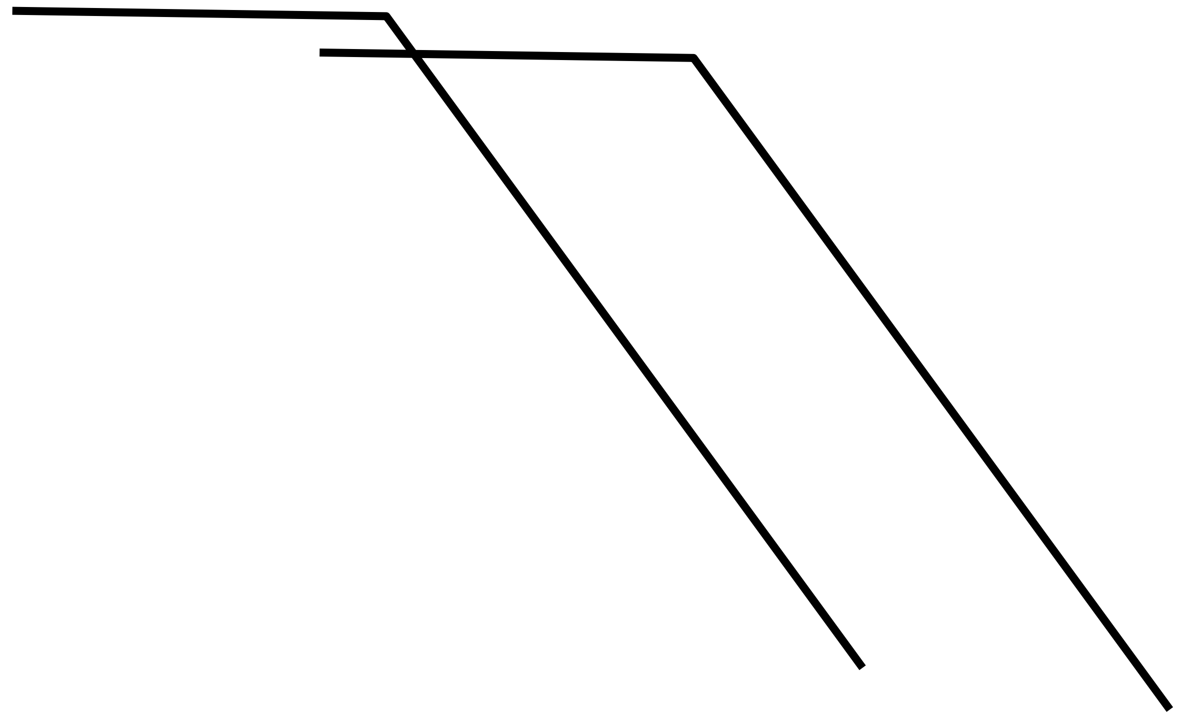
http://en.wikipedia.org/wiki/Barber's_pole

Optical Flow Constraint

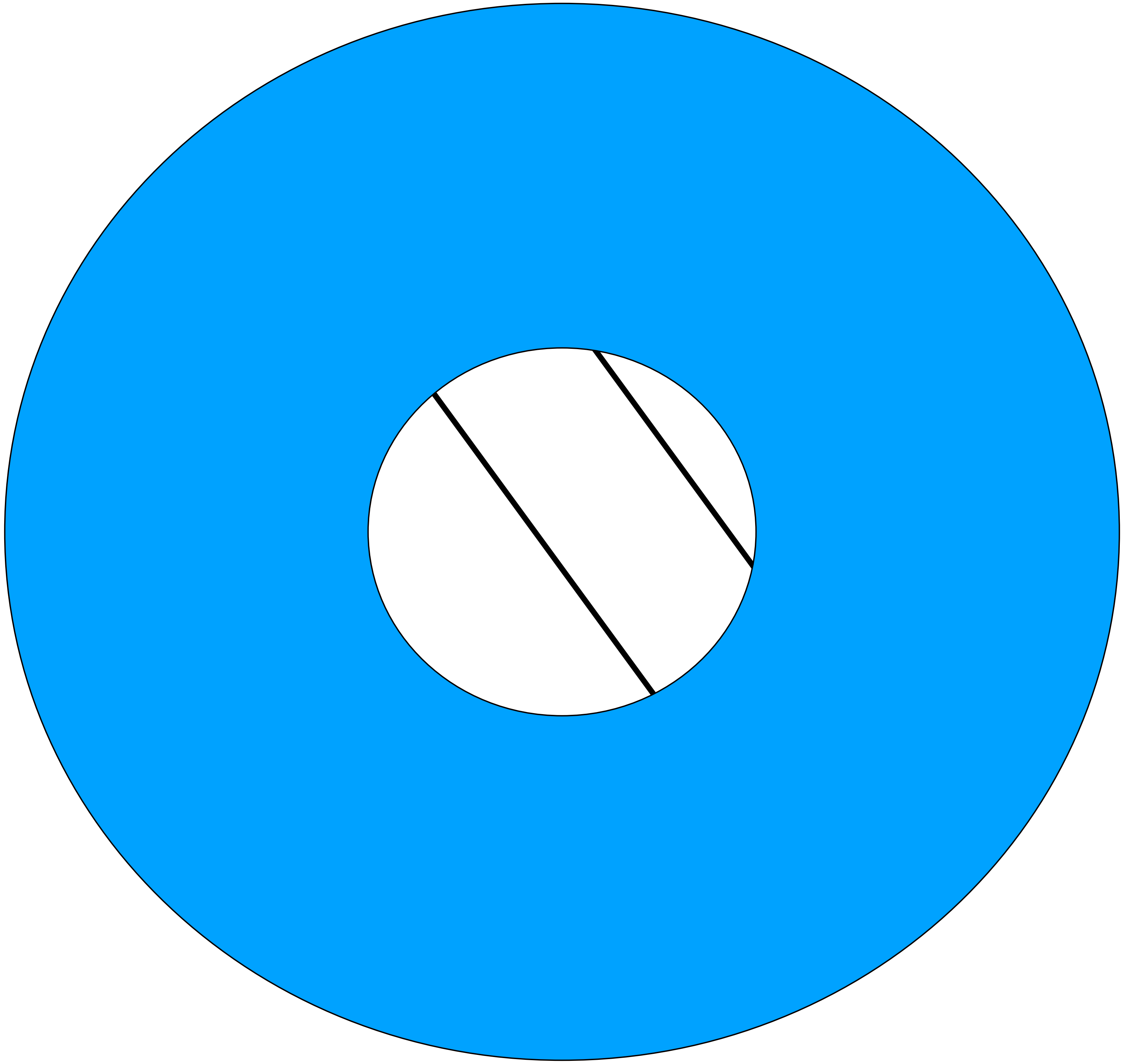
- Barber Pole Illustration



Aperture Problem



Aperture Problem



Solving the Aperture Problem

- 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 5×5 window, that gives us 25 equations per pixel!

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

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

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 5×5 window, that gives us $25*3$ equations per pixel!

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

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

Lukas-Kanade Flow

- Problem: 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:

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

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

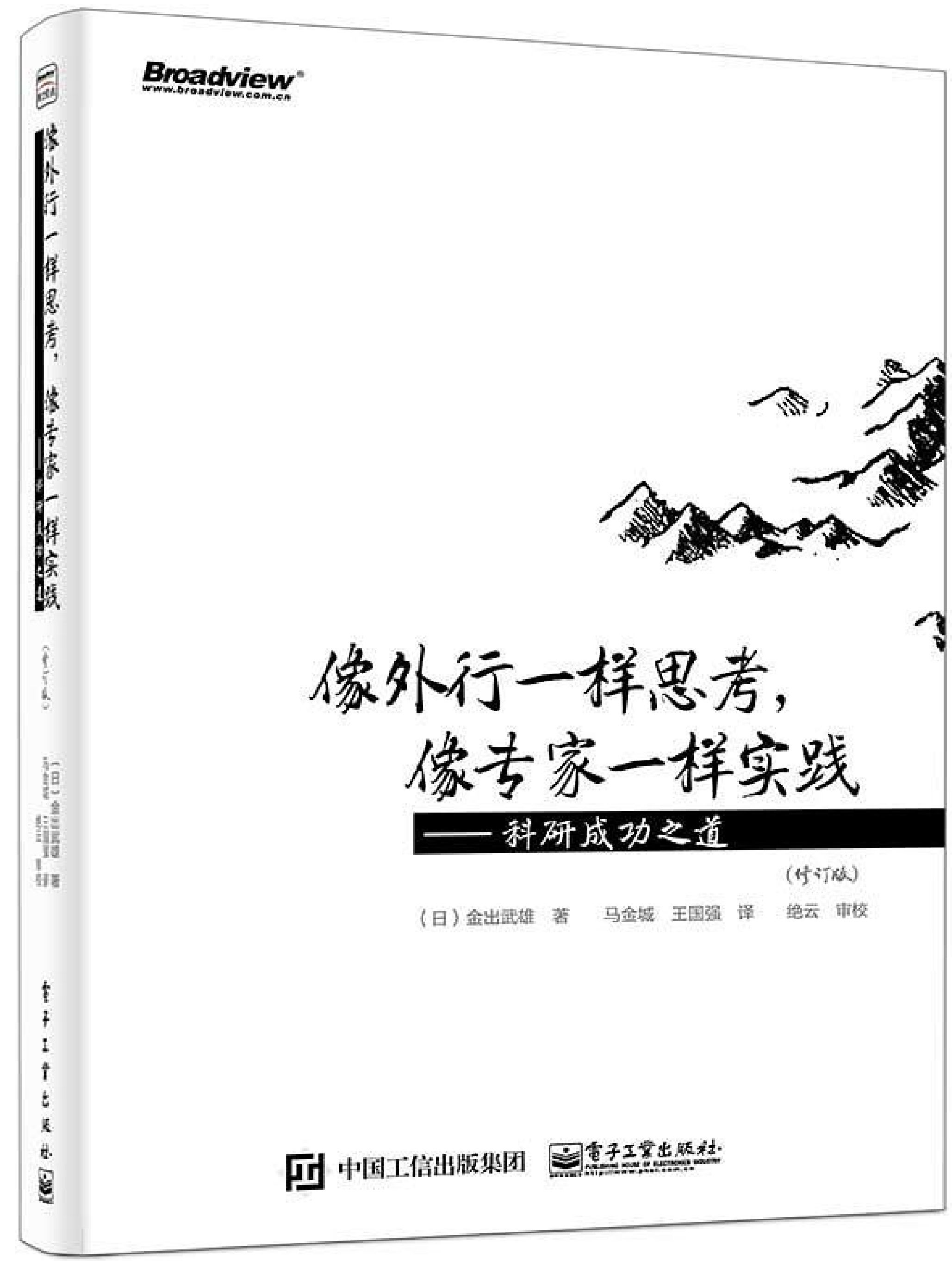
- The summations are over all pixels in the K x K window
- This technique was first proposed by Lukas & Kanade [1981]

Prof. Takeo Kanade

Carnegie Mellon University
The Robotics Institute

Takeo Kanade

U.A. and Helen Whitaker
University Professor, RI / CS
Phone: (412) 268-3016
Administrative Assistant: Yukiko Kano
Lab: Human Sensing
kanade@andrew.cmu.edu
Mailing Address



Conditions for Solvability

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

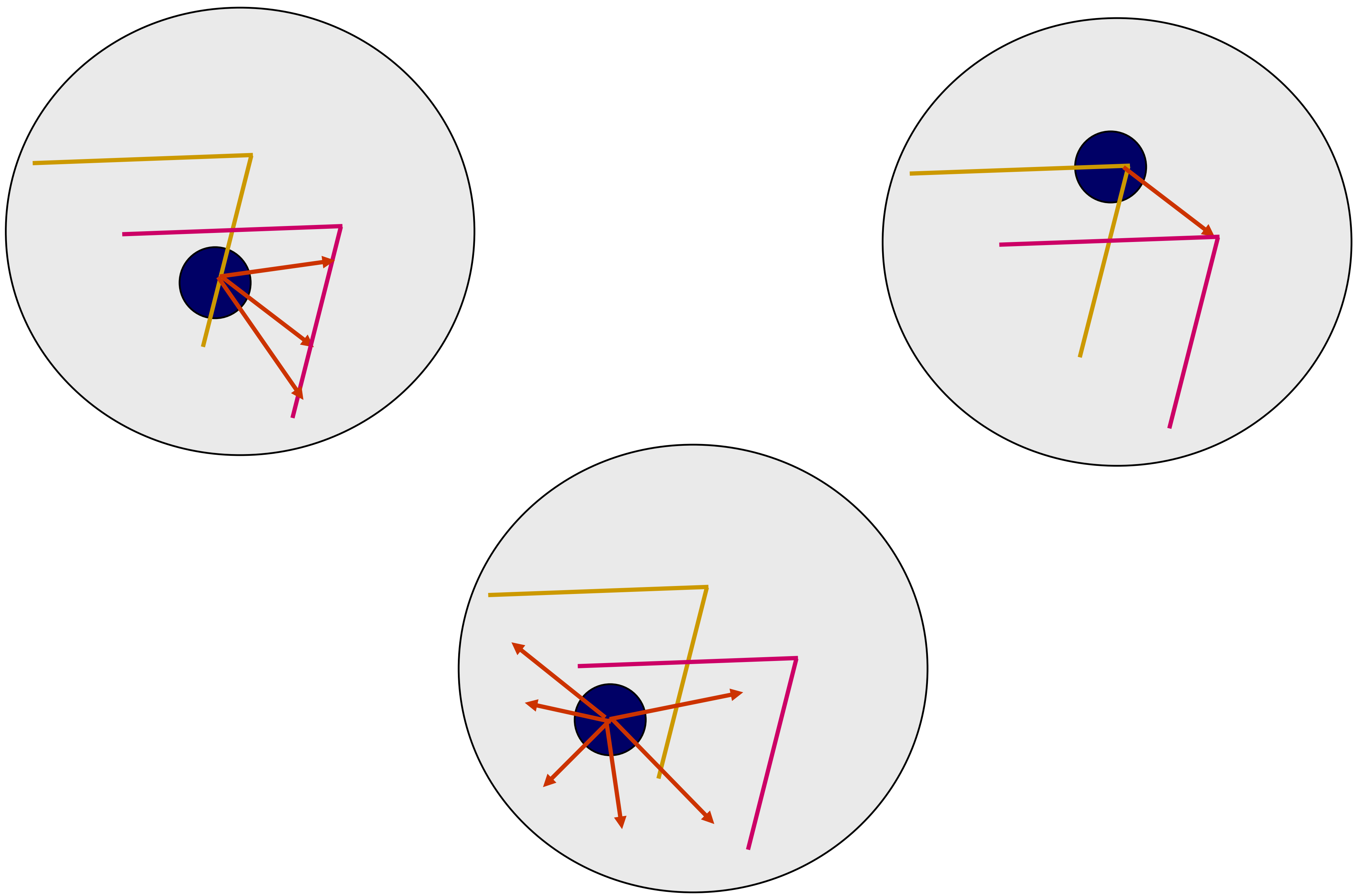
$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

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

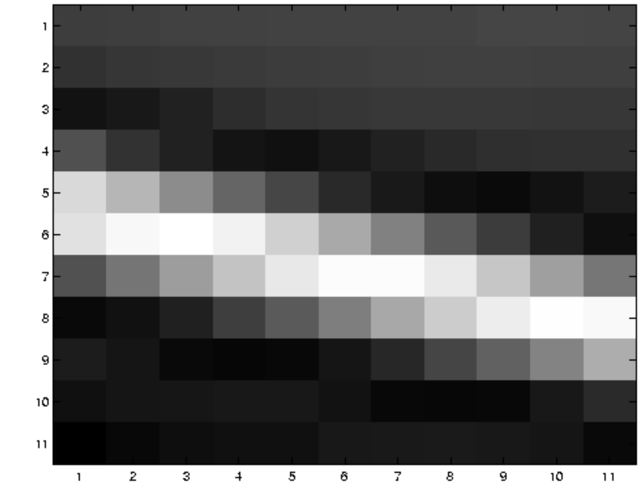
A^TA is solvable when there is no aperture problem

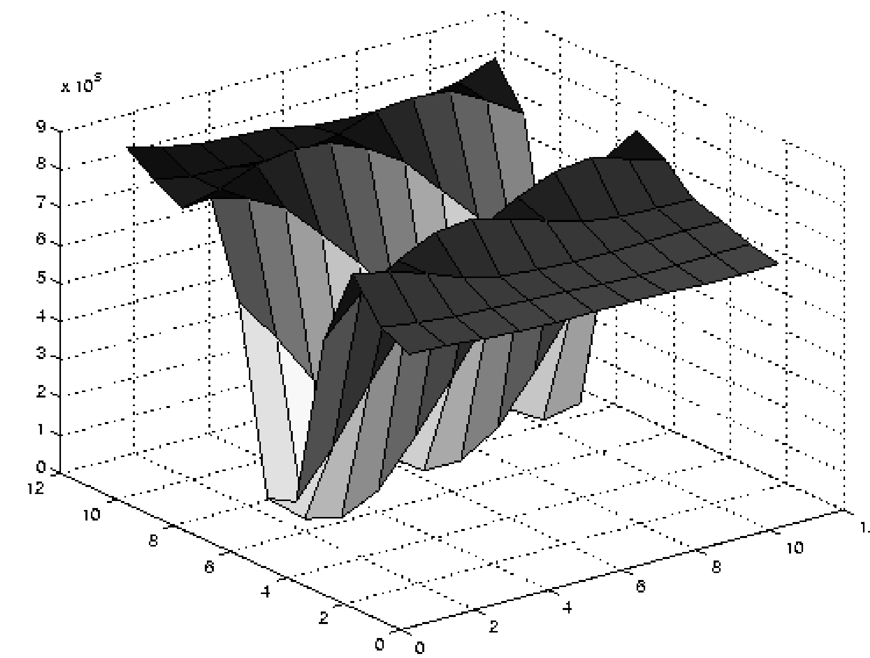
$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y] = \sum \nabla I (\nabla I)^T$$

Local Patch Analysis



Edge

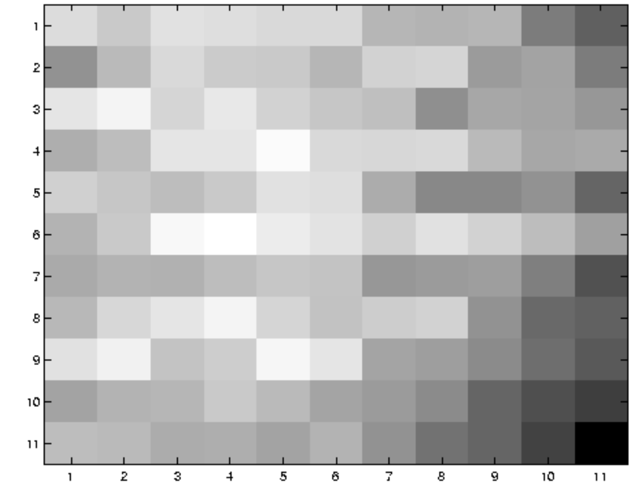


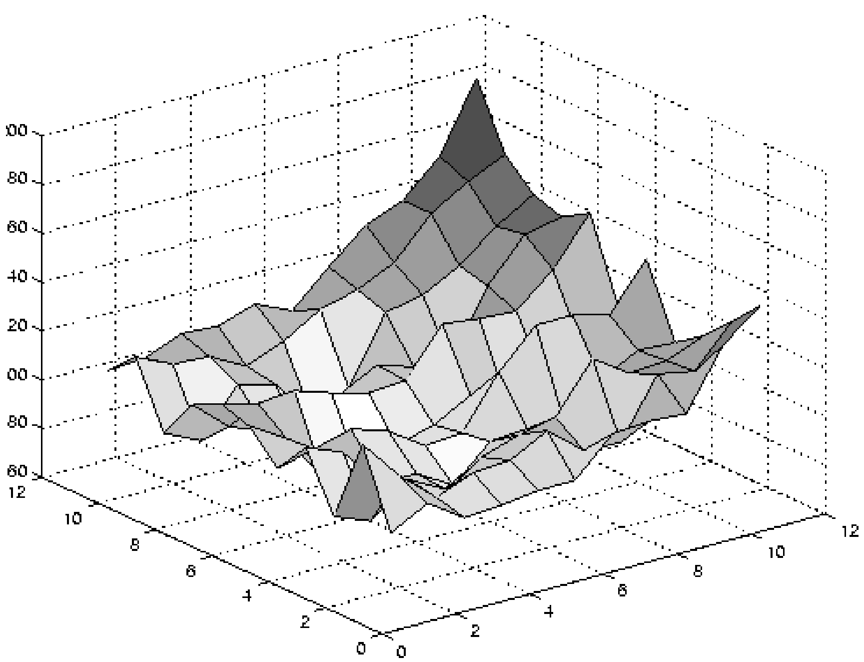


$$\sum \nabla I (\nabla I)^T$$

- large gradients, all the same
- large λ_1, small λ_2

Low Texture Region

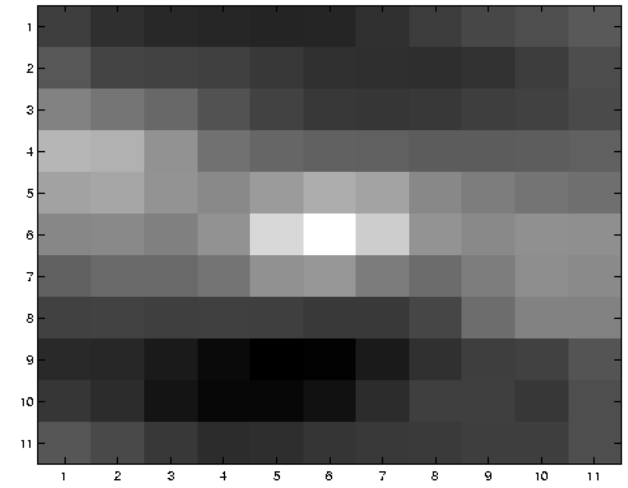


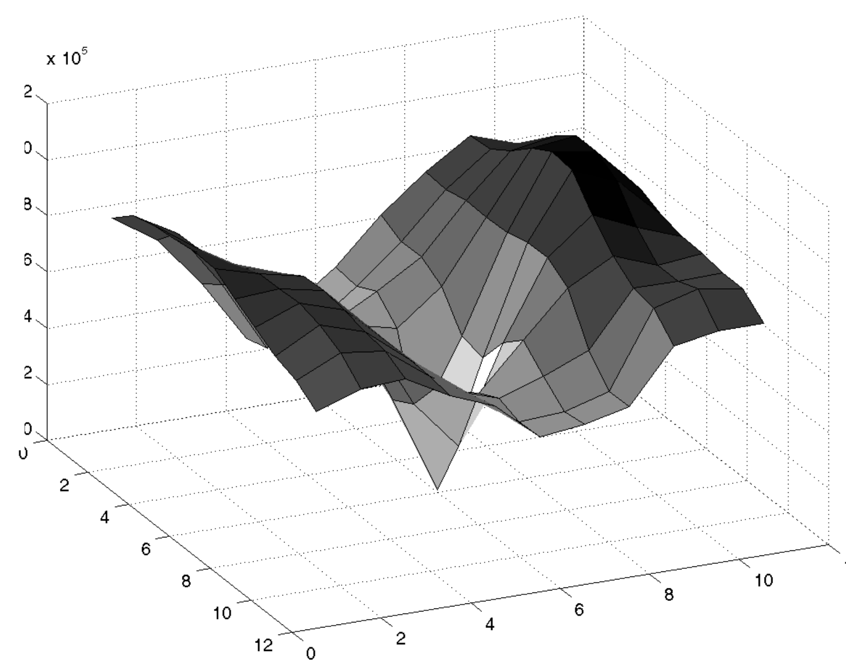


$$\sum \nabla I (\nabla I)^T$$

- gradients have small magnitude
- small 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

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 later on when we do feature tracking ...

Errors in Lukas-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?

Iterative Refinement

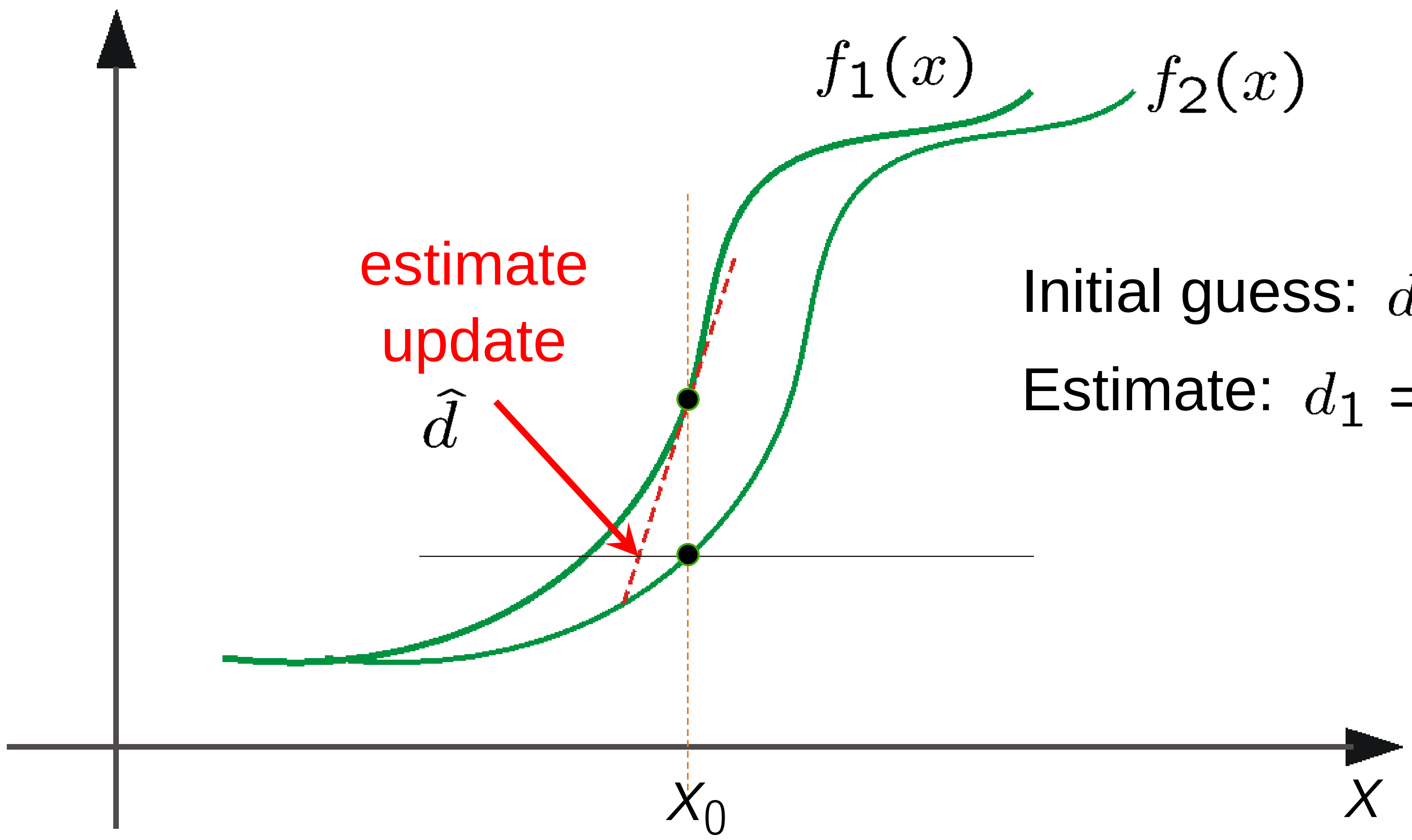
Iterative Lukas-Kanade Algorithm

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

Optical Flow: Iterative Estimation

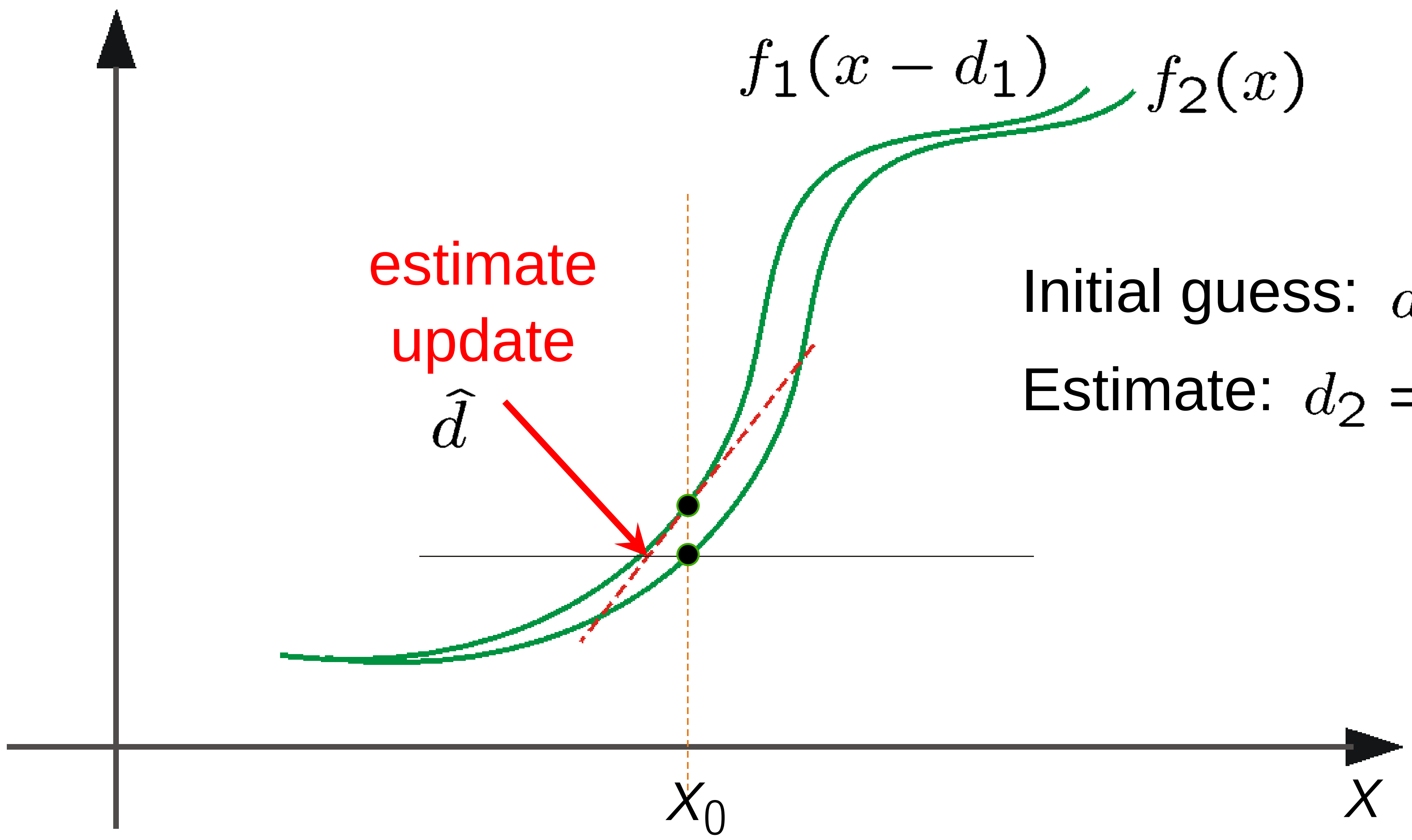


Initial guess: $d_0 = 0$

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

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

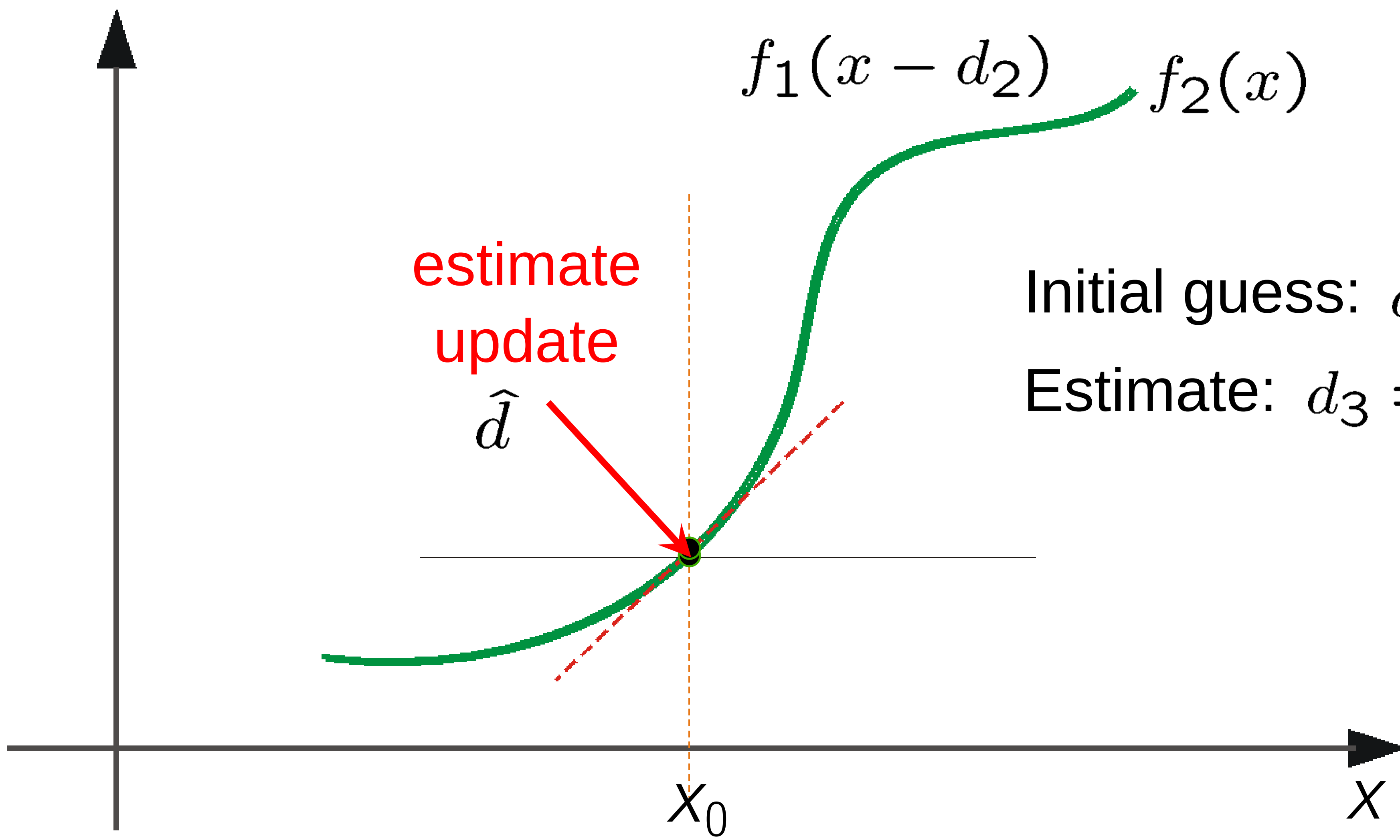
Optical Flow: Iterative Estimation



Initial guess: d_1

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

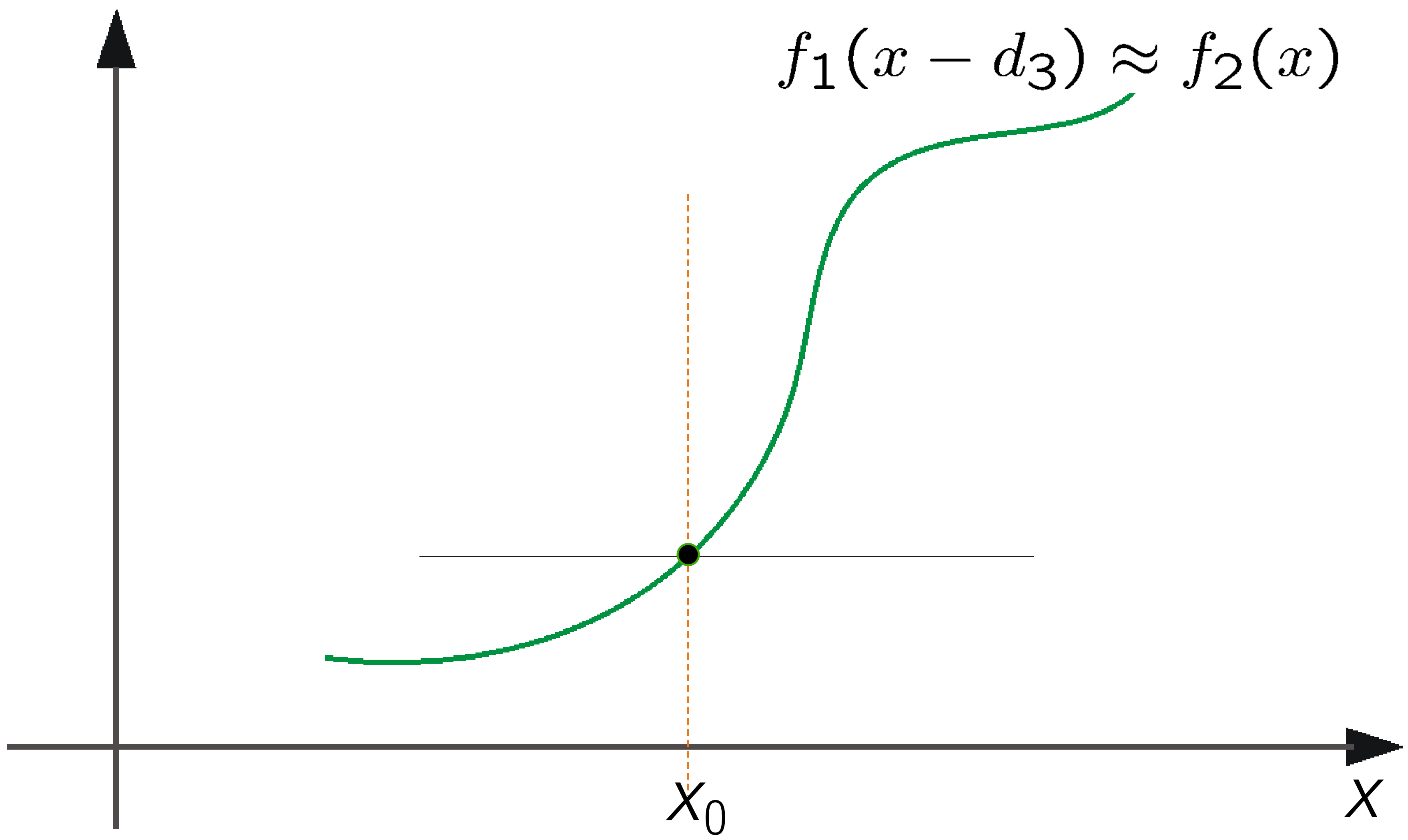
Optical Flow: Iterative Estimation



Initial guess: d_2

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

Optical Flow: Iterative Estimation



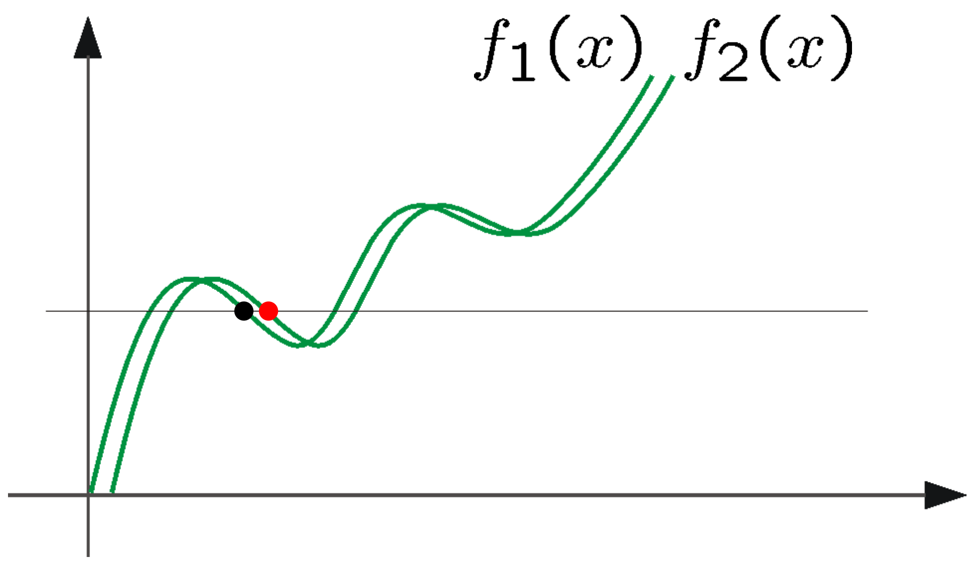
Revisiting Small Motion Assumption

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

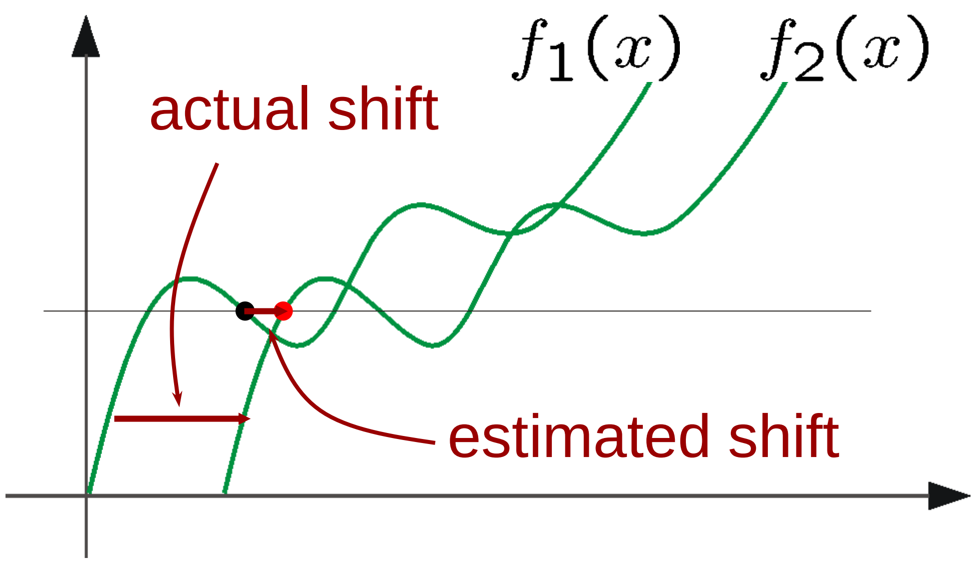
Optical Flow: Aliasing

Temporal aliasing causes ambiguities in optical flow because images can have many pixels with the same intensity.

i.e., how do we know which ‘correspondence’ is correct?



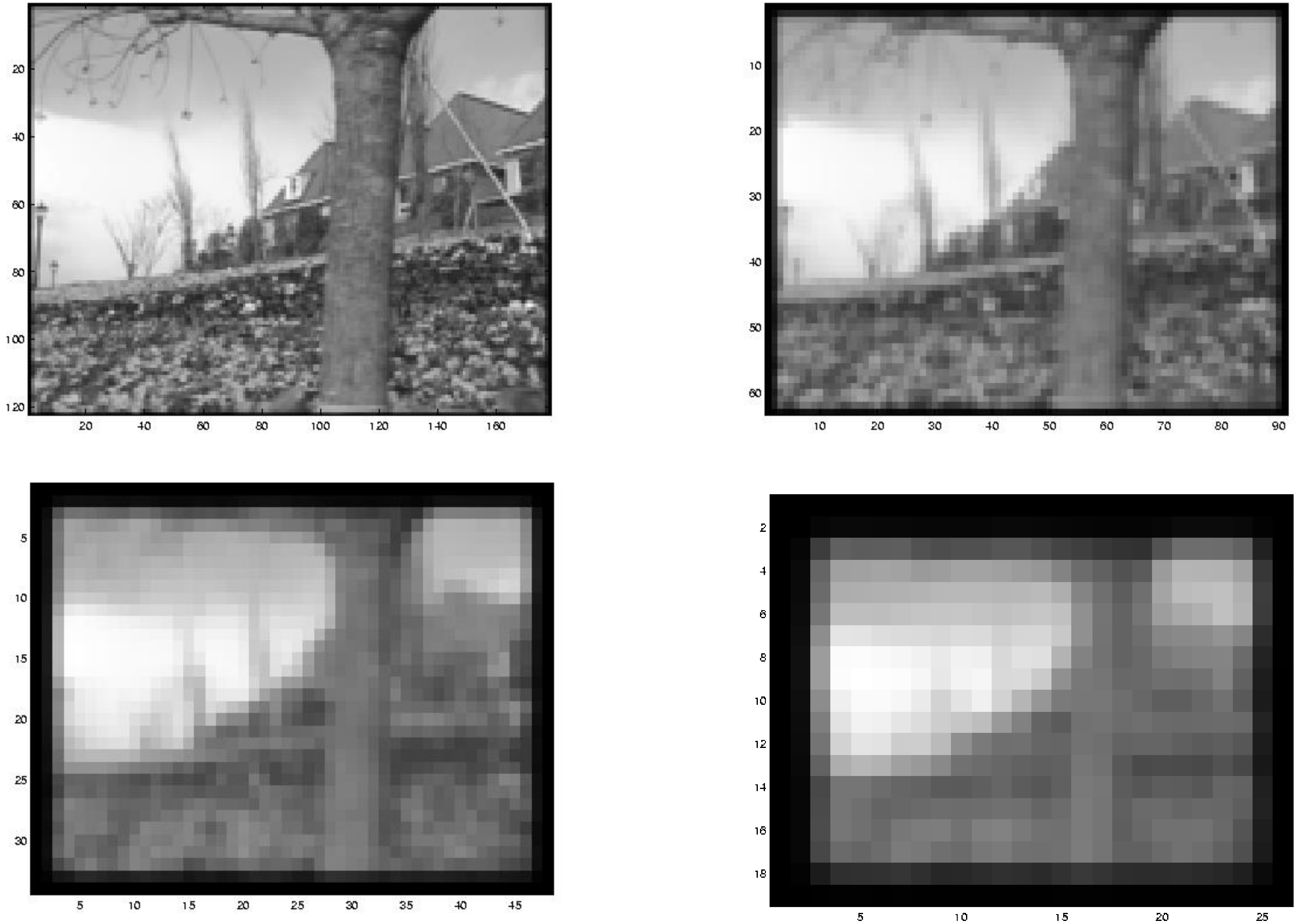
nearest match is correct (no aliasing)



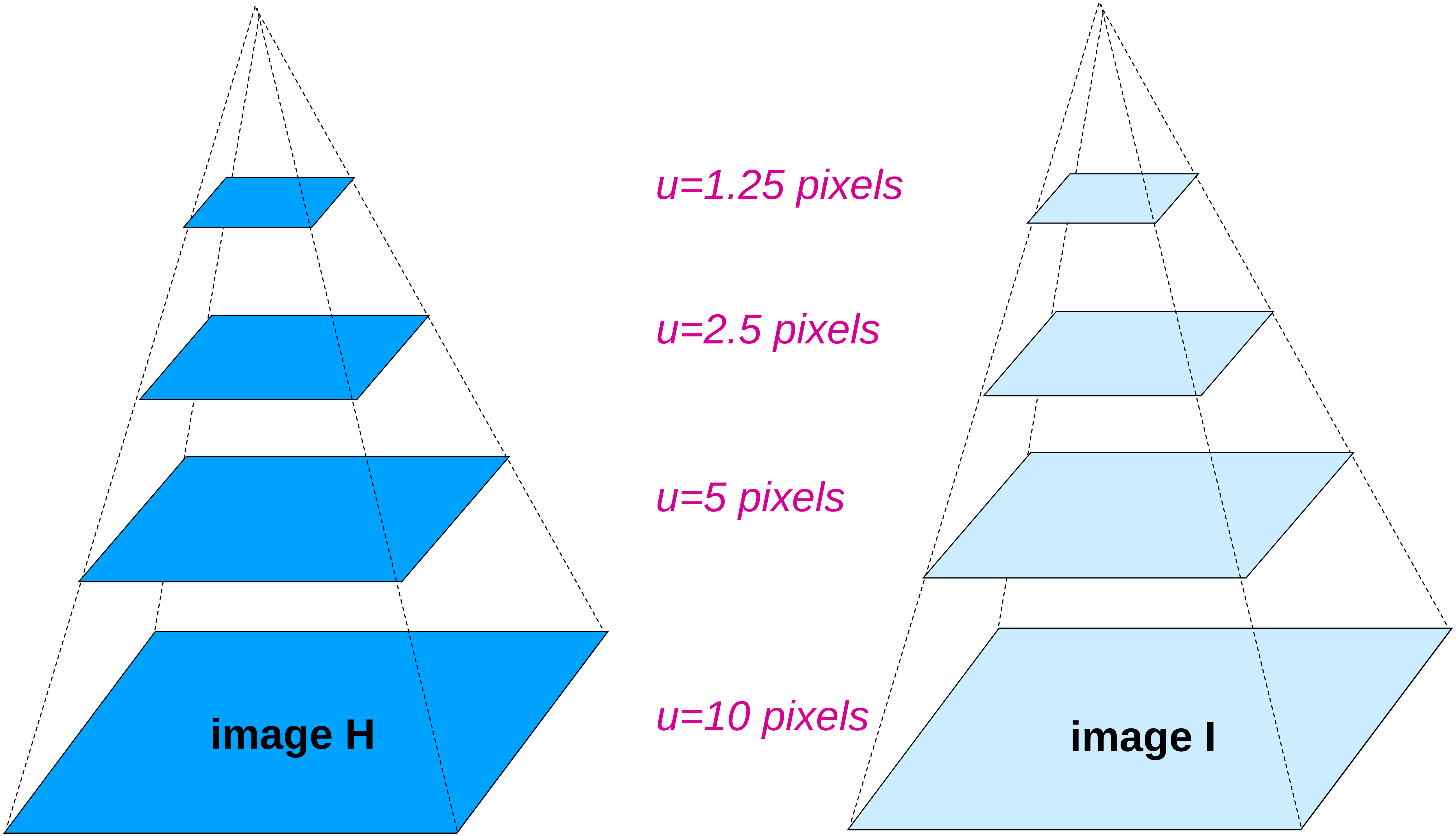
nearest match is incorrect (aliasing)

To overcome aliasing: coarse-to-fine estimation.

Reduce the Resolution!



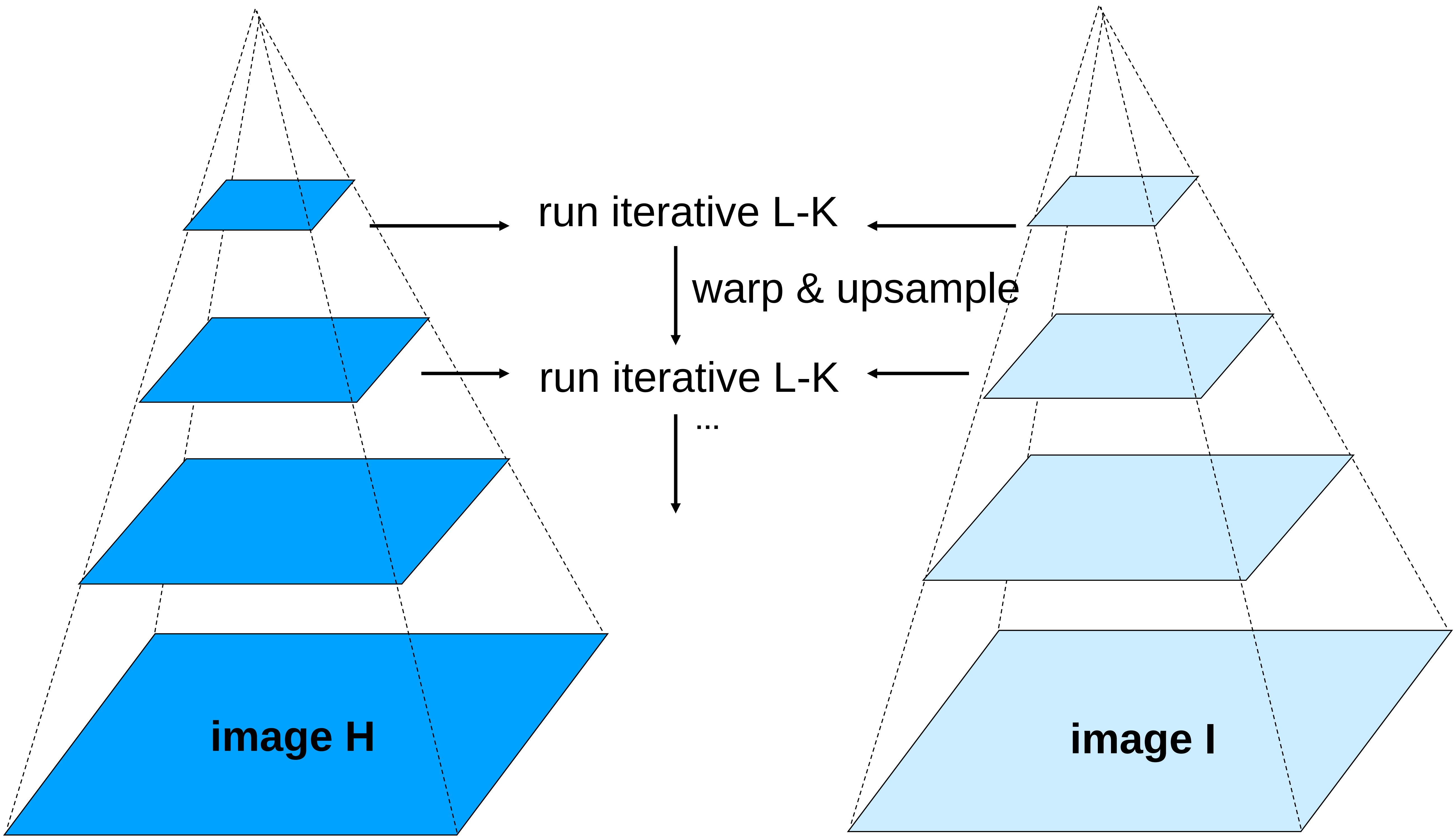
Coarse-to-Fine Optical Flow Estimation



Gaussian pyramid of image H

Gaussian pyramid of image I

Coarse-to-Fine Optical Flow Estimation



Gaussian pyramid of image H

Gaussian pyramid of image I

Block-Based Motion Prediction

- Break image up into square blocks
- Estimate translation for each block
- Use this to predict next frame, code difference (MPEG-2)