Image Motion

Brightness Constancy Equation

Image intensity at Time = t, $I(x, y, t)$

Image intensity at Time = t + dt, $I(x + dx, y + dy, t + dt)$

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

Taylor Series expansion

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

Therefore,

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

$$\frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy + \frac{\partial I}{\partial t}dt = 0$$

Brightness Constancy Equation

$$\frac{\partial I}{\partial x}dx + \frac{\partial I}{\partial y}dy + \frac{\partial I}{\partial t}dt = 0$$

Taking derivative wrt time:

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0$$

Let

$$\nabla I = \begin{bmatrix} \frac{\partial I}{\partial x} \\ \frac{\partial I}{\partial y} \end{bmatrix}$$

(Frame spatial gradient)

$$d = \begin{bmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \end{bmatrix}$$

(optical flow)

$$I_t = \frac{\partial I}{\partial t}$$

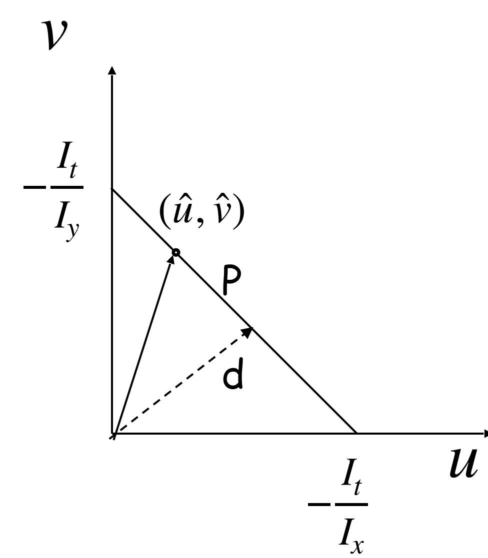
(derivative across frames)

Brightness Constancy Equation

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0$$

Becomes:

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



where, $u = \frac{dx}{dt}; v = \frac{dy}{dt}$

Line equation $v = -\frac{I_x}{I_y}u - \frac{I_t}{I_y}$

$$d = \frac{f_t}{\sqrt{f_x^2 + f_y^2}}$$

The OF is CONSTRAINED to be on a line !

Brightness Constancy Equation

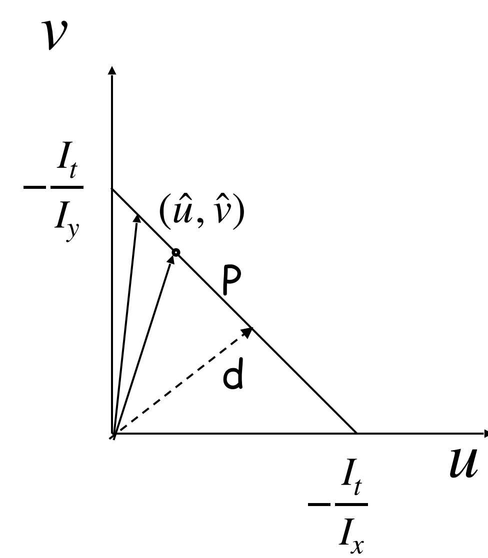
$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0$$

Becomes:

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

where, $u = \frac{dx}{dt}; v = \frac{dy}{dt}$

Line equation $v = -\frac{I_x}{I_y}u - \frac{I_t}{I_y}$



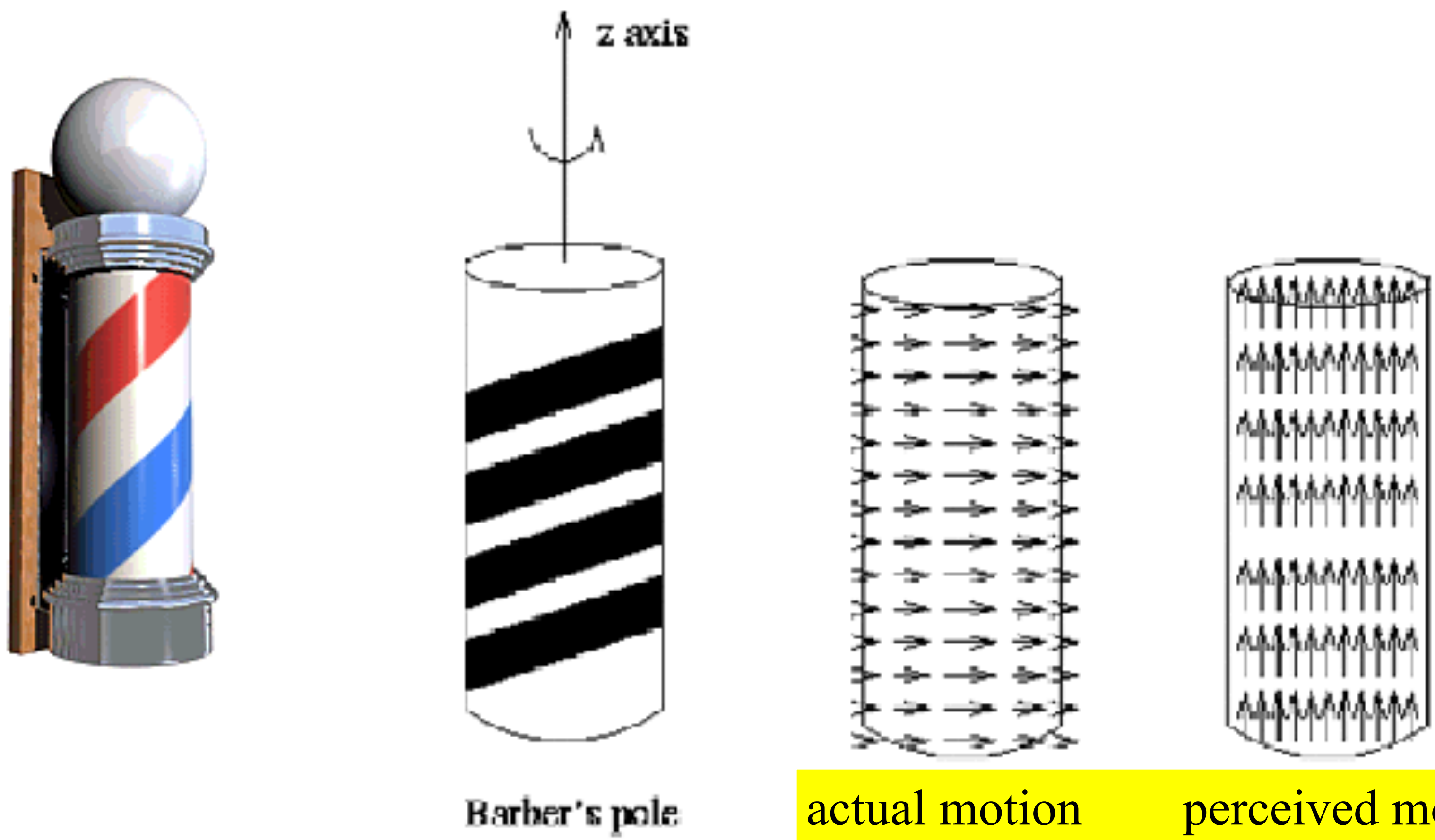
$$d = \frac{f_t}{\sqrt{f_x^2 + f_y^2}}$$

The OF is CONSTRAINED to be on a line !

Aperture Problem in Real Life

Aperture Problem

Barber pole illusion



actual motion perceived motion

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 5x5 window, that gives us 25 equations per pixel!

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

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

Lukas-Kanade flow

- Prob: we have more equations than unknowns

$$\underset{25\times 2}{A}\ \underset{2\times 1}{d} = \underset{25\times 1}{b} \longrightarrow \text{minimize } \|Ad - b\|^2$$

- Solution: solve least squares problem
  - minimum least squares solution given by solution (in d) of: $\underset{2\times 2}{(A^T A)}\ \underset{2\times 1}{d} = \underset{2\times 1}{A^T b}$

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

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

Taking a closer look at ($A^T A$)

$$A = \begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \cdots & \cdots \\ I_x(p_{N^2}) & I_y(p_{N^2}) \end{bmatrix}$$

$$A^T = \begin{bmatrix} I_x(p_1) & I_x(p_2) & \cdots & I_x(p_{N^2}) \\ I_y(p_1) & I_y(p_2) & \cdots & I_y(p_{N^2}) \end{bmatrix}$$

$$A^T A = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

This is the same matrix we used for corner detection!

Taking a closer look at (A^TA)

The matrix for corner detection:

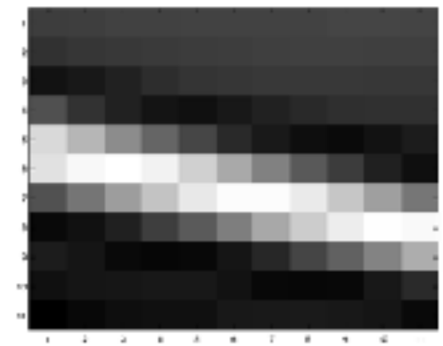
$$A^T A = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

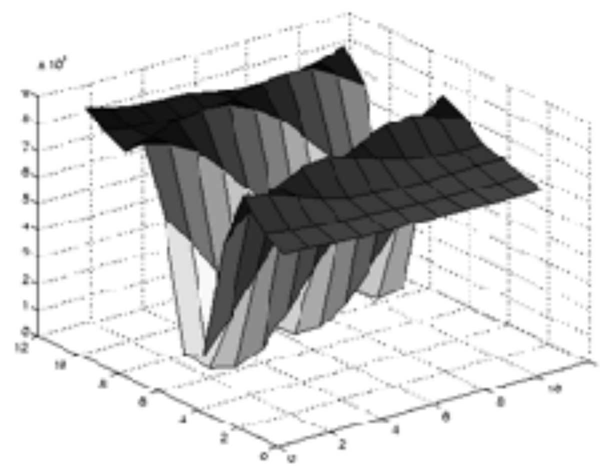
is singular (not invertible) when $\det(A^TA) = 0$

But $\det(A^TA) = \prod \lambda_i = 0$ -> one or both e.v. are 0

One e.v. = 0 -> no corner, just an edge
Two e.v. = 0 -> no corner, homogeneous region

} Aperture Problem !

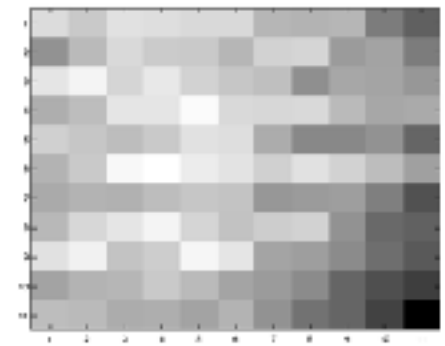


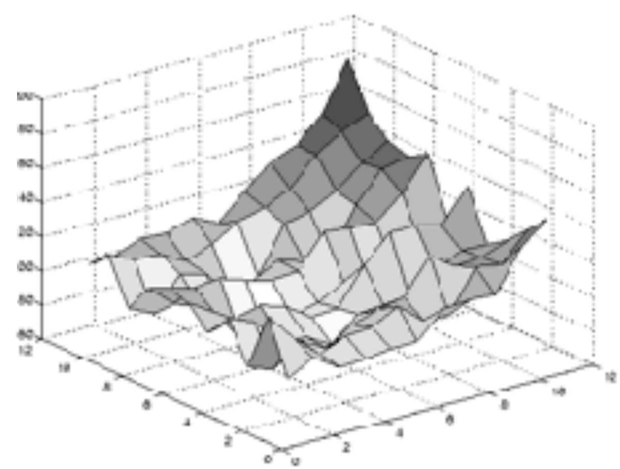


$$\sum \nabla I (\nabla I)^T$$

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

Low texture region

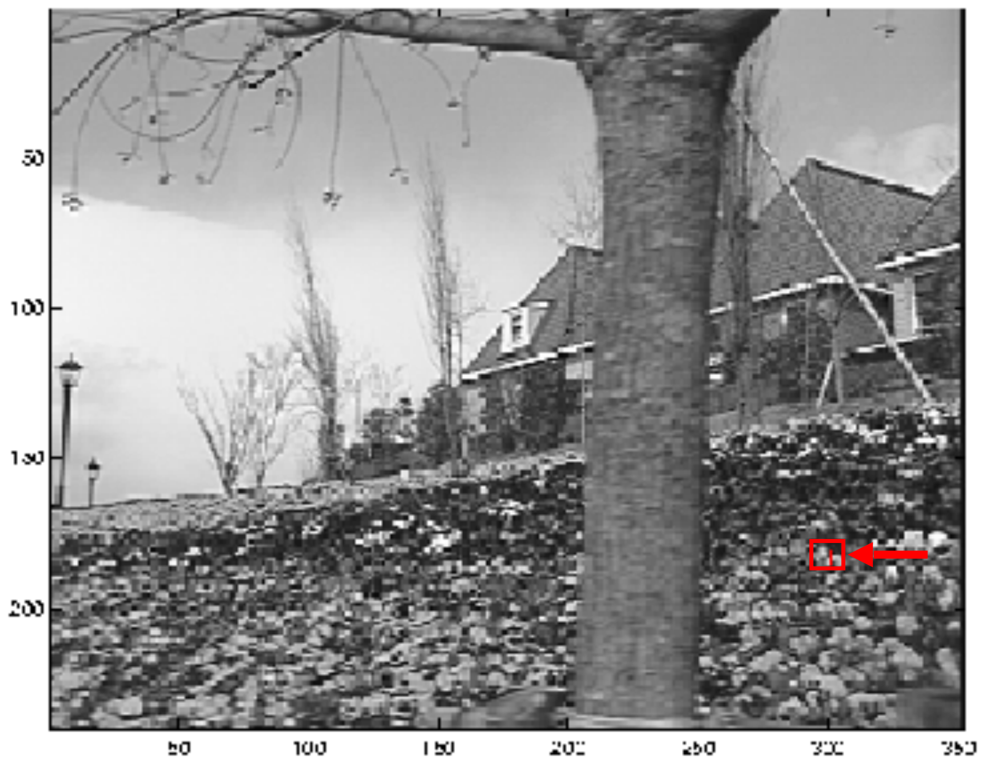


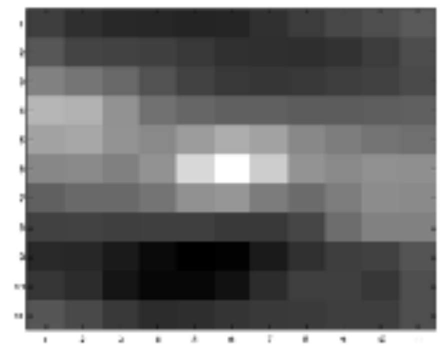


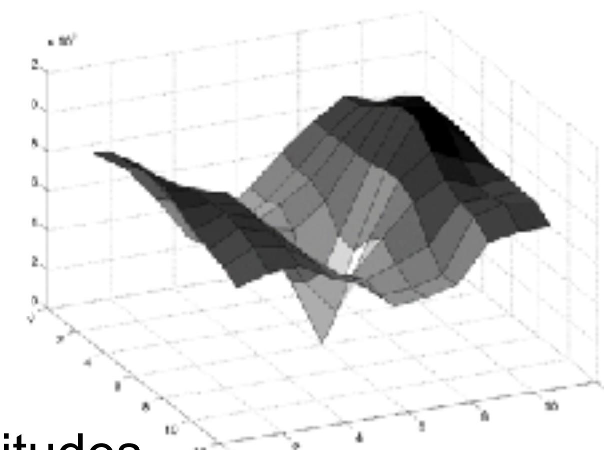
$$\sum \nabla I (\nabla I)^T$$

- gradients have small magnitude
- small λ_1, small λ_2

High textured region





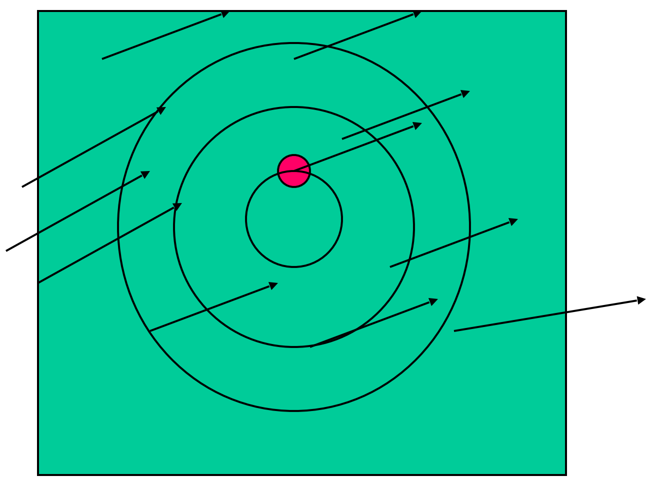


$$\sum \nabla I (\nabla I)^T$$

- gradients are different, large magnitudes
- large λ_1, large λ_2

An improvement …

- NOTE:
  - The assumption of constant OF is more likely to be wrong as we move away from the point of interest (the center point of Q)



Use weights to control the influence of the points: the farther from p, the less weight

Solving for v with weights:

- Let W be a diagonal matrix with weights
- Multiply both sides of $Av = b$ by W:

$$W A v = W b$$

- Multiply both sides of $WAv = Wb$ by $(WA)^T$:

$$A^T WWA v = A^T WWb$$

- $A^T W^2 A$ is square (2x2):
  - $(A^T W^2 A)^{-1}$ exists if $\det(A^T W^2 A) \neq 0$
- Assuming that $(A^T W^2 A)^{-1}$ does exists:

$$(A^T W^2 A)^{-1} (A^T W^2 A) v = (A^T W^2 A)^{-1} A^T W^2 b$$

$$v = (A^T W^2 A)^{-1} A^T W^2 b$$

Observation

- This is a problem involving two images 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...

Revisiting the small motion assumption

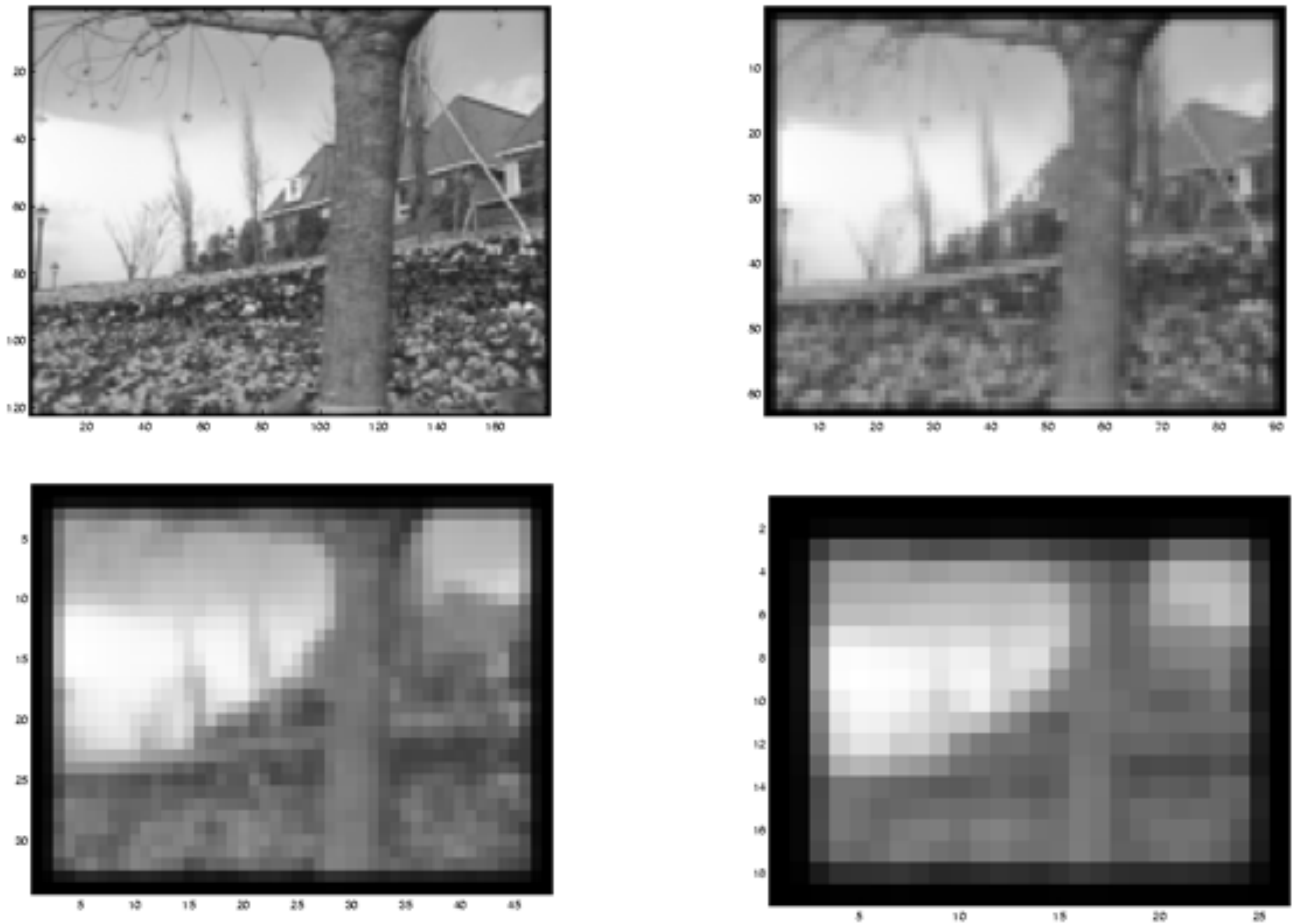
- Is this motion small enough?
  - Probably not—it’s much larger than one pixel (2nd order terms dominate)
  - How might we solve this problem?

Iterative Refinement

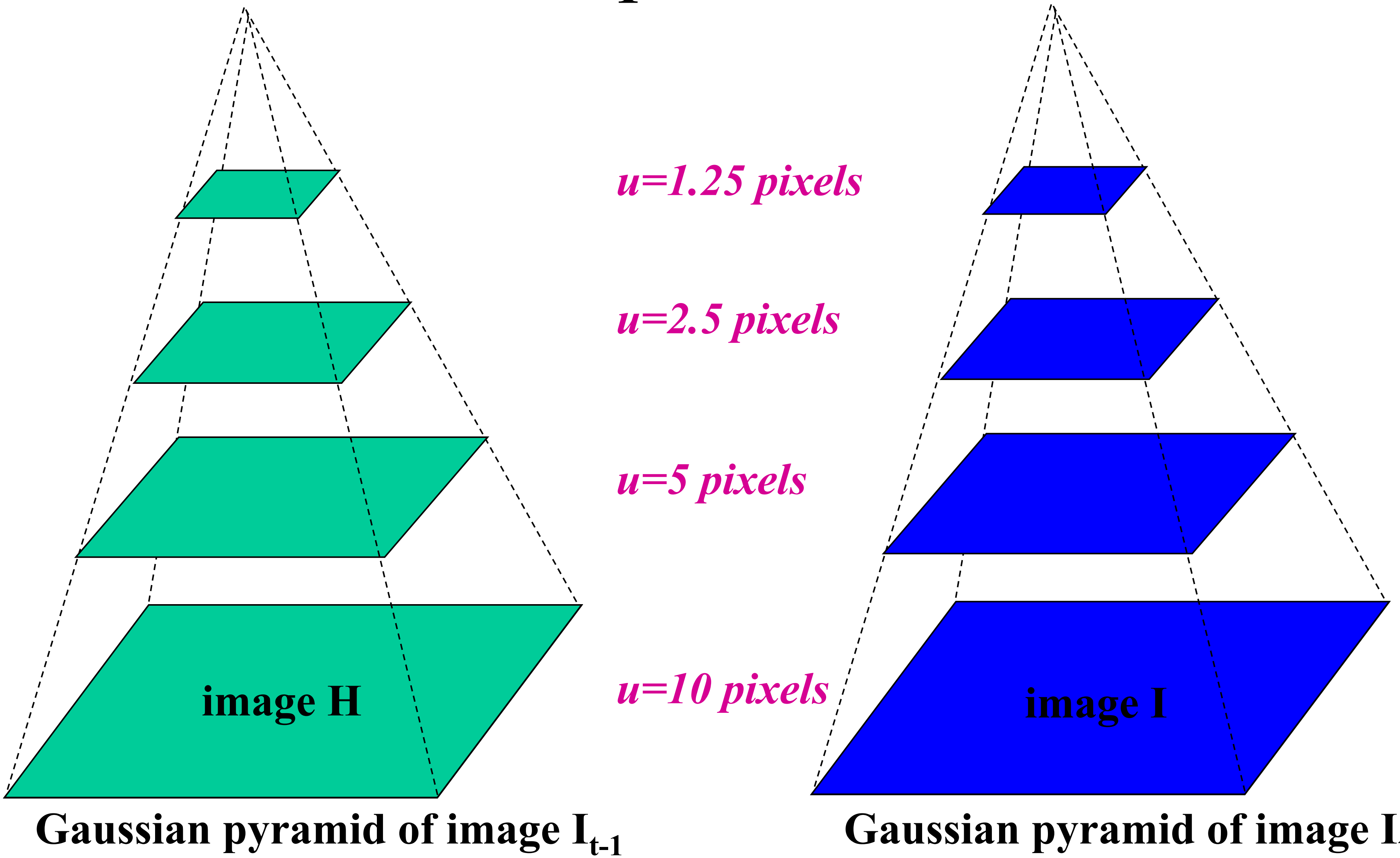
- Iterative Lukas-Kanade Algorithm
  1. Estimate velocity at each pixel by solving Lucas-Kanade equations
  2. Warp I(t-1) towards I(t) using the estimated flow field

     - use image warping techniques
  3. Repeat until convergence

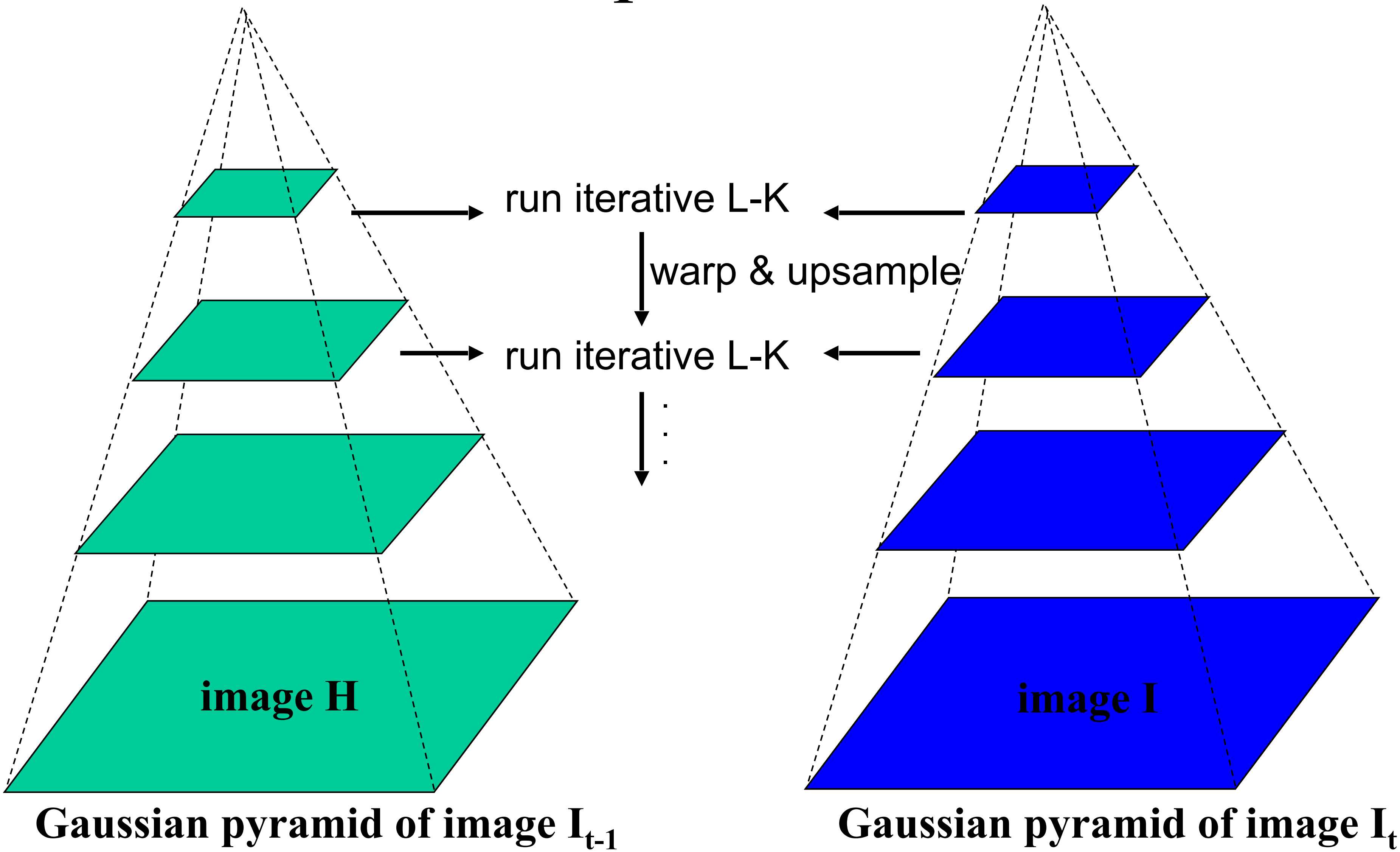
Reduce the resolution!



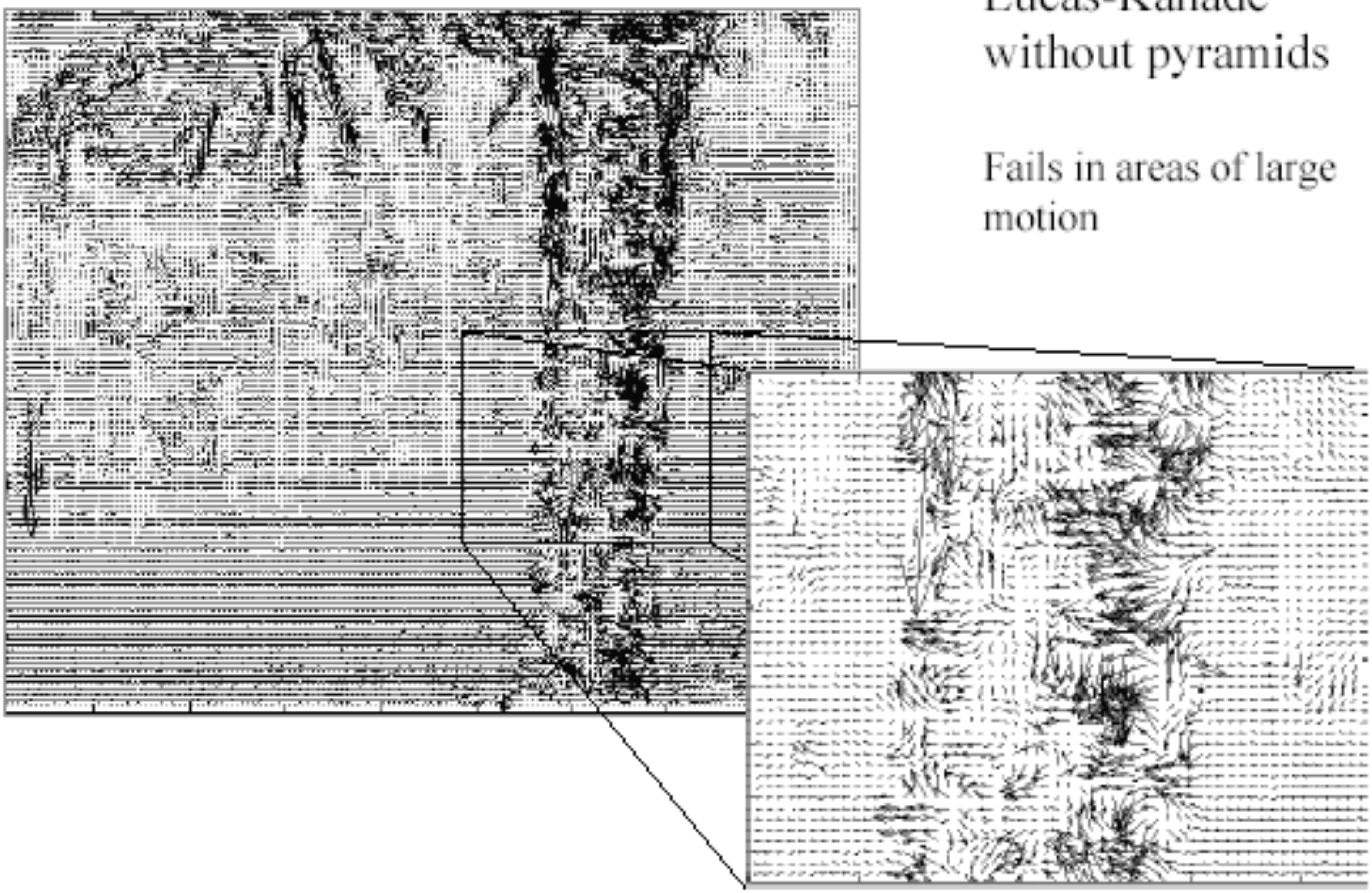
Coarse-to-fine optical flow estimation



Coarse-to-fine optical flow estimation



Optical Flow Results

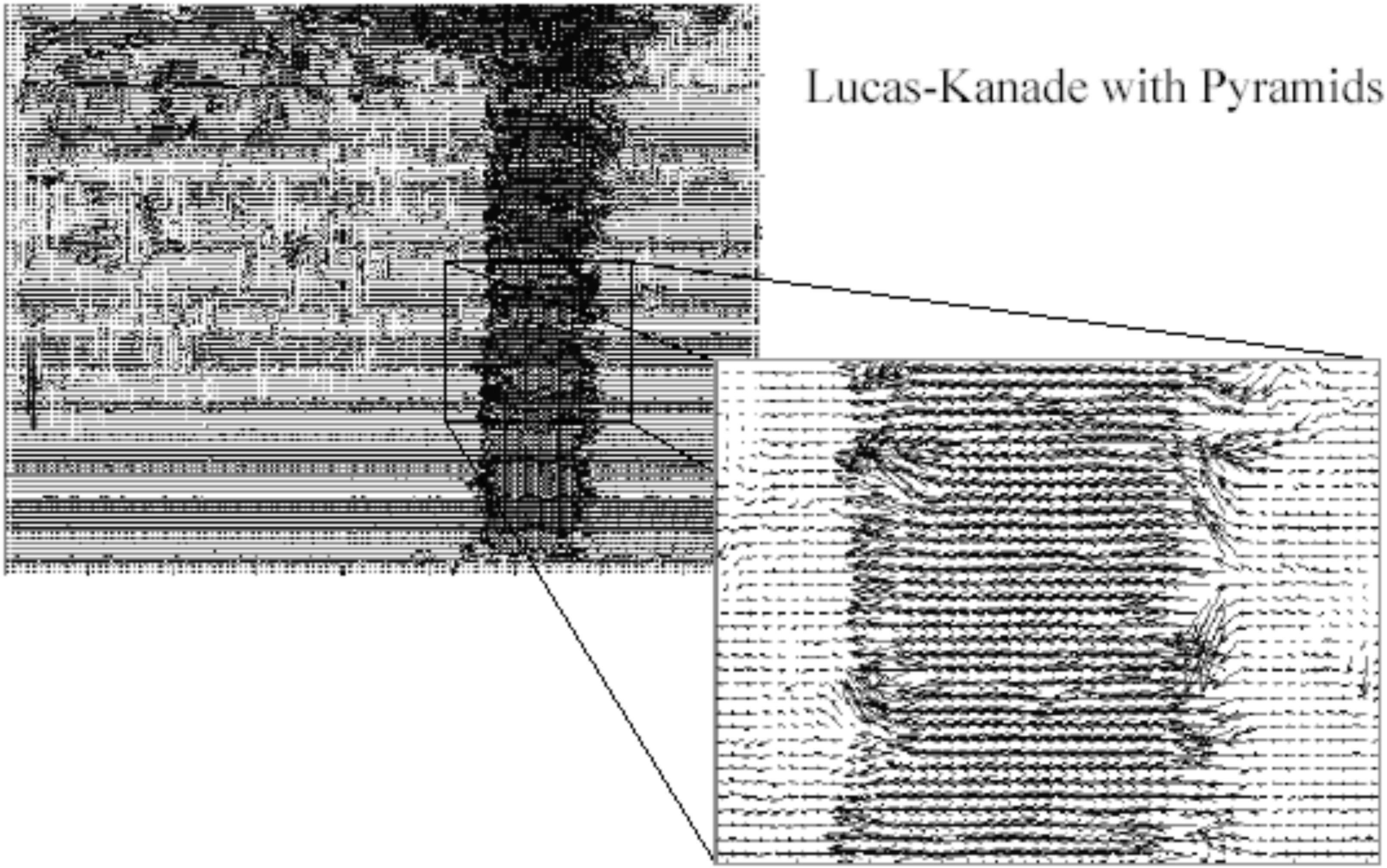


Lucas-Kanade without pyramids

Fails in areas of large motion

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Optical Flow Results



* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003