VC 10/11 – T11
Optical Flow

Mestrado em Ciência de Computadores

Mestrado Integrado em Engenharia de Redes e Sistemas Informáticos

Miguel Tavares Coimbra

Outline

- Optical Flow Constraint Equation
- Aperture problem.
- The Lucas & Kanade Algorithm

Acknowledgements: Most of this course is based on the excellent courses offered by Prof. Shree Nayar at Columbia University, USA and by Prof. Srinivasa Narasimhan at CMU, USA. Please acknowledge the original source when reusing these slides for academic purposes.

Topic: Optical Flow Constraint Equation

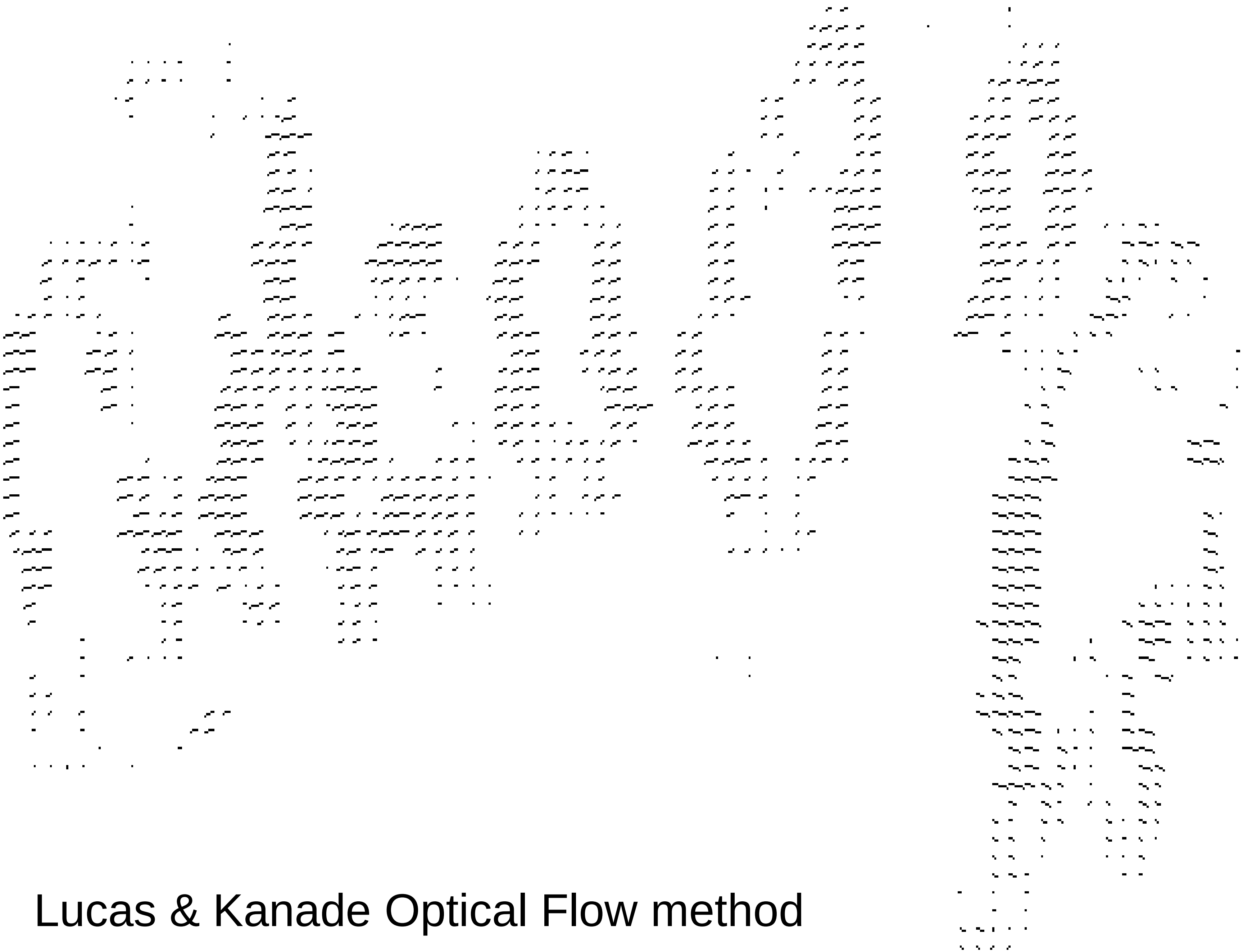
- Optical Flow Constraint Equation
- Aperture problem.
- The Lucas & Kanade Algorithm

Optical Flow and Motion

- We are interested in finding the movement of scene objects from time-varying images (videos).

- Lots of uses
  - Track object behavior
  - Correct for camera jitter (stabilization)
  - Align images (mosaics)
  - 3D shape reconstruction
  - Special effects

Where can i find motion?

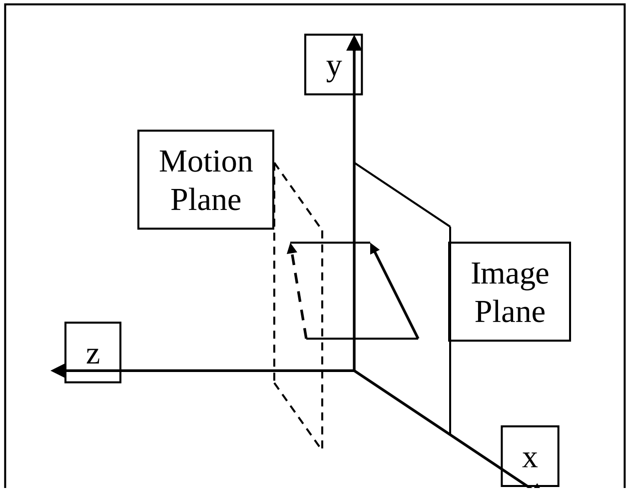


Lucas & Kanade Optical Flow method

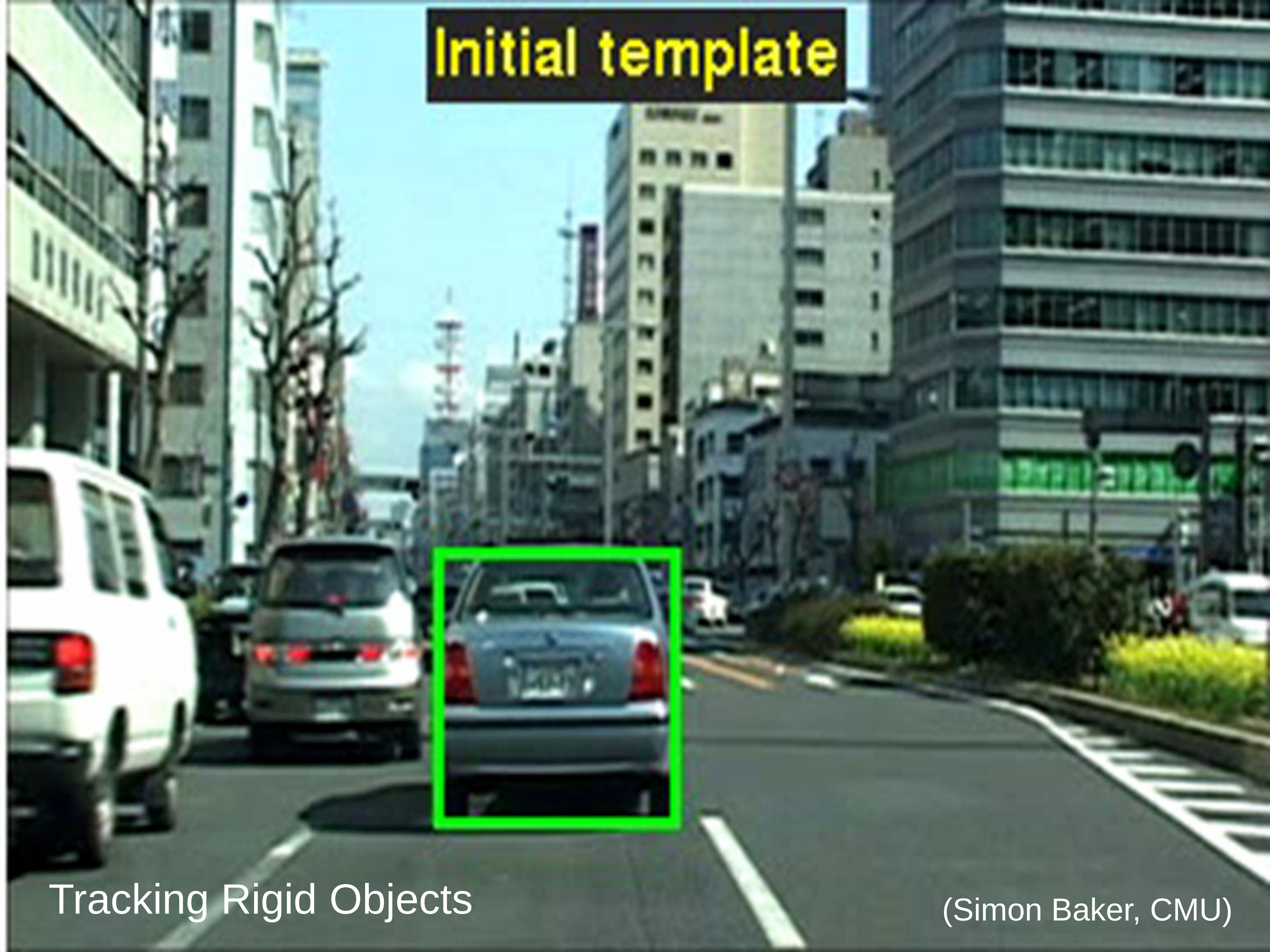
Optical Flow – What is that?

Optical flow is "the distribution of apparent velocities of movement of brightness patterns in an image" – Horn and Schunck 1980

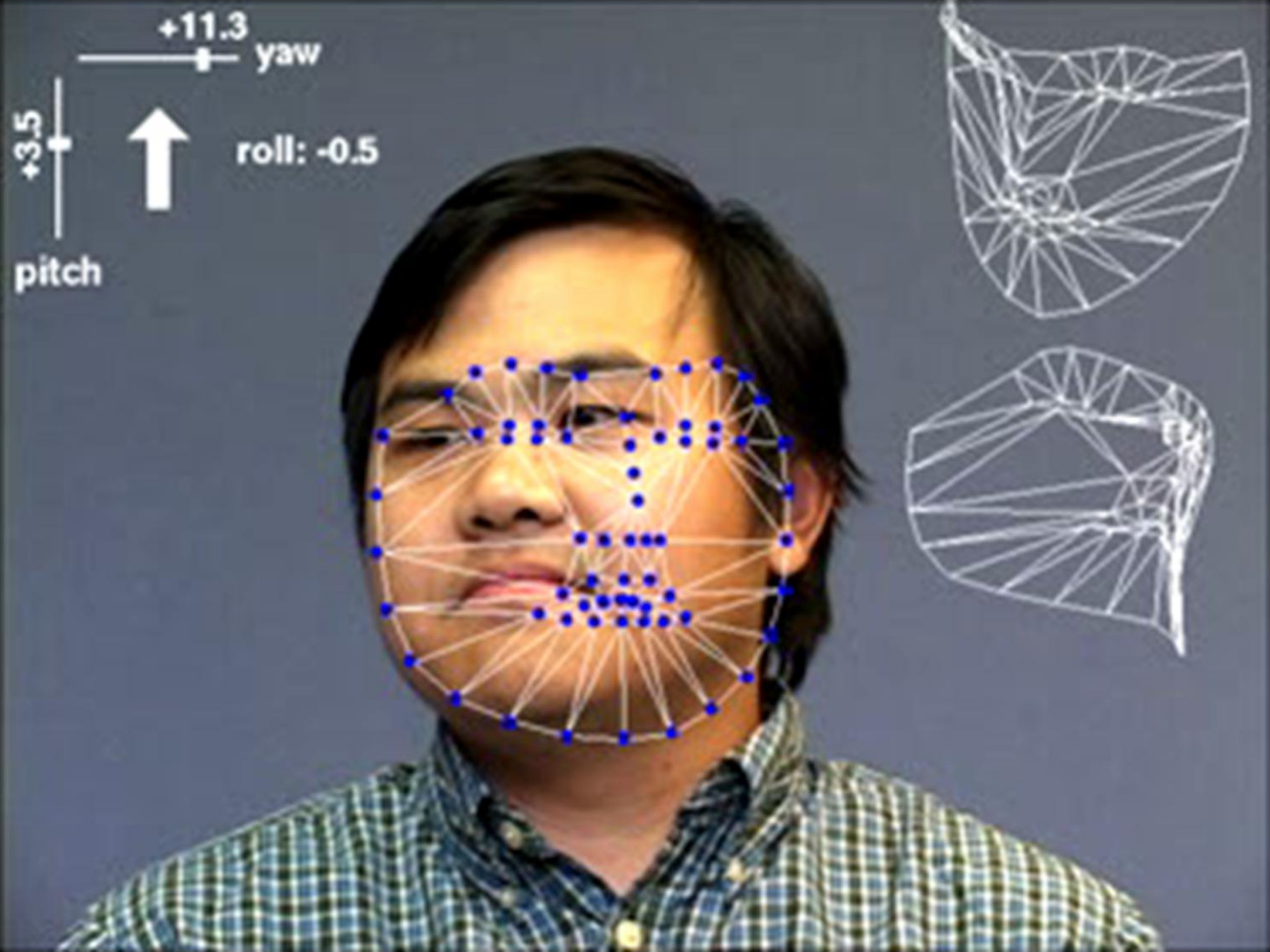
The optical flow field approximates the true motion field which is a "purely geometrical concept..., it is the [2D] projection into the image [plane] of [the sequence's] 3D motion vectors" – Horn and Schunk 1993



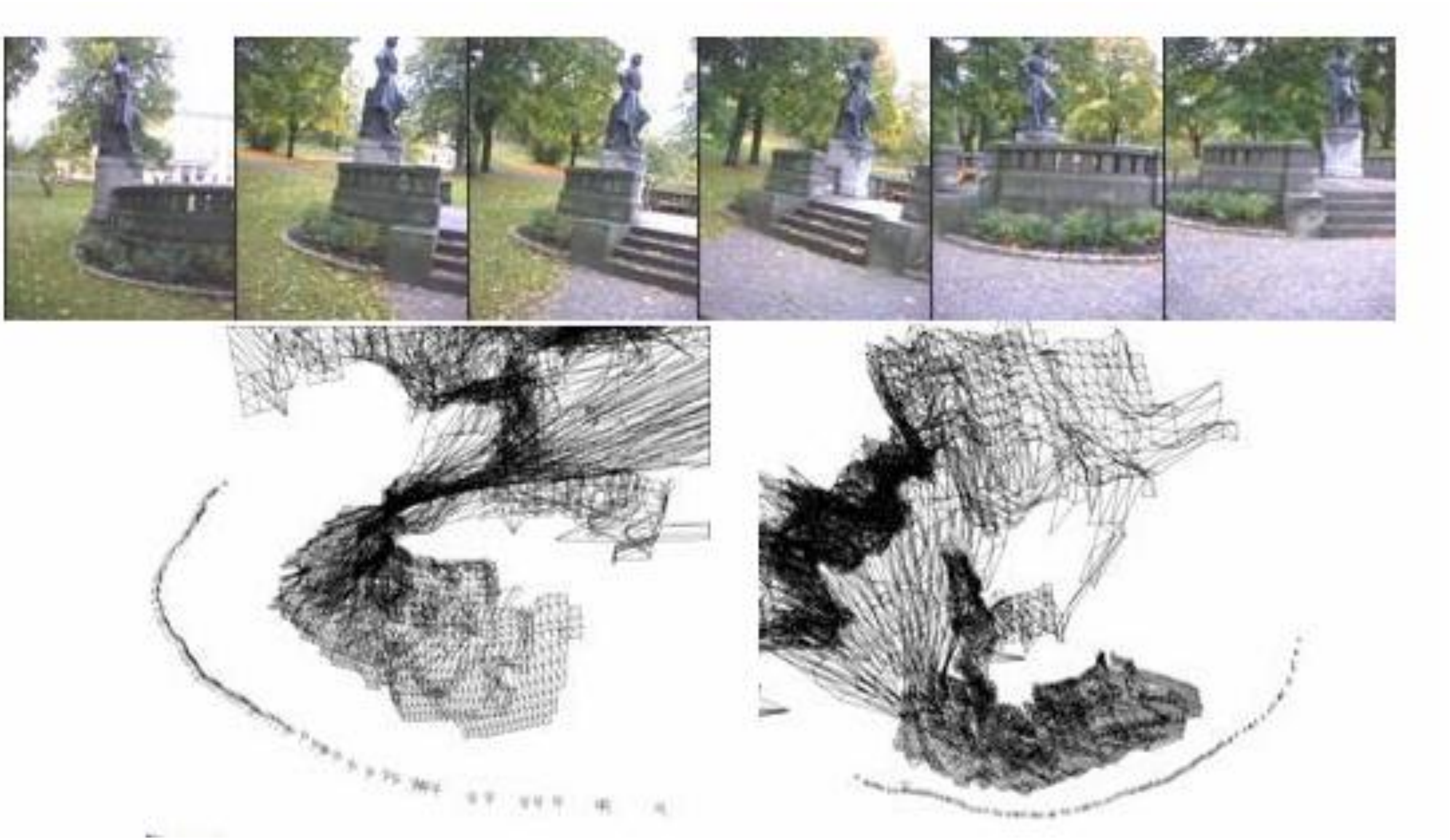
What can i use it for?


Initial template
Tracking Rigid Objects
(Simon Baker, CMU)

Tracking – Non-rigid Objects (Comaniciu et al, Siemens)


+11.3
yaw
+3.5
roll: -0.5
pitch

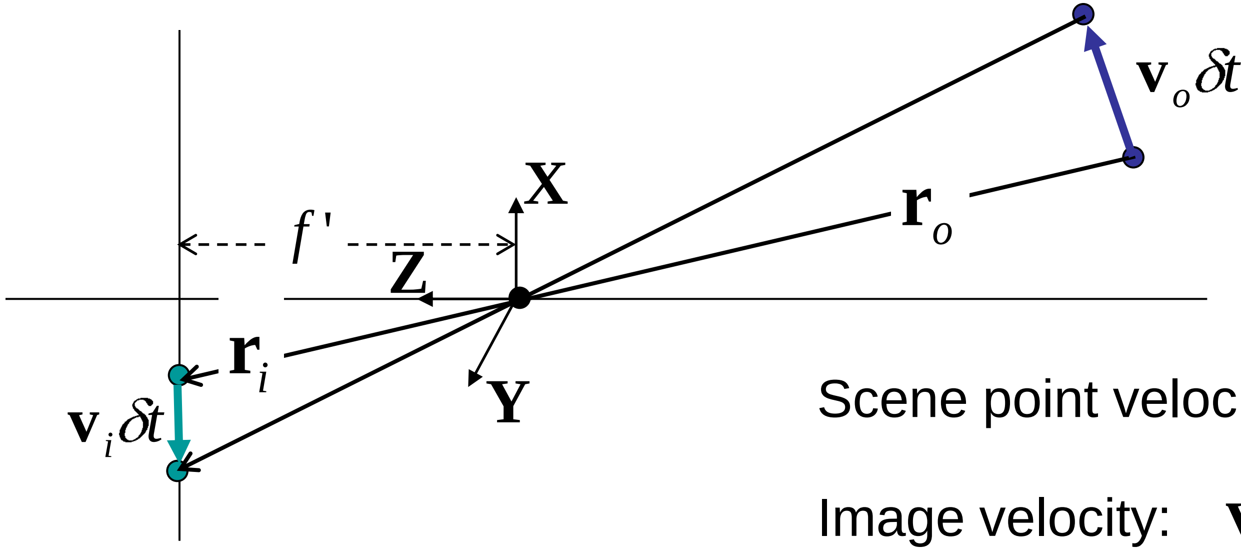
3D Structure from Motion



(David Nister, Kentucky)

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

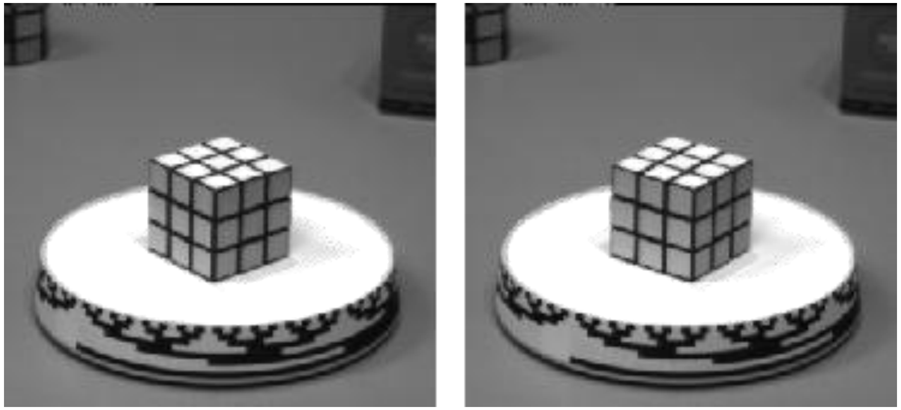
Perspective projection: $\dfrac{1}{f'}\mathbf{r}_i = \dfrac{\mathbf{r}_o}{\mathbf{r}_o \cdot \mathbf{Z}}$

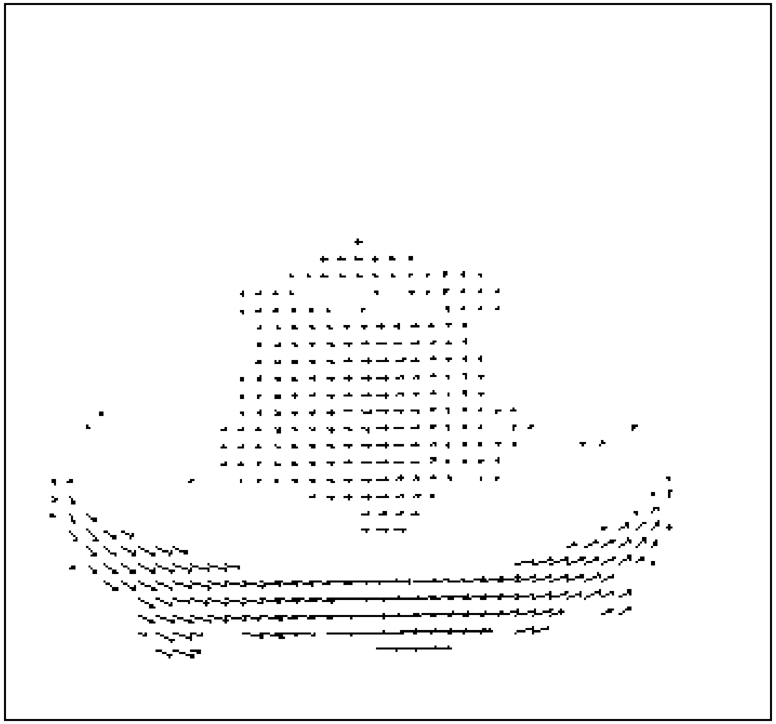
Motion field

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

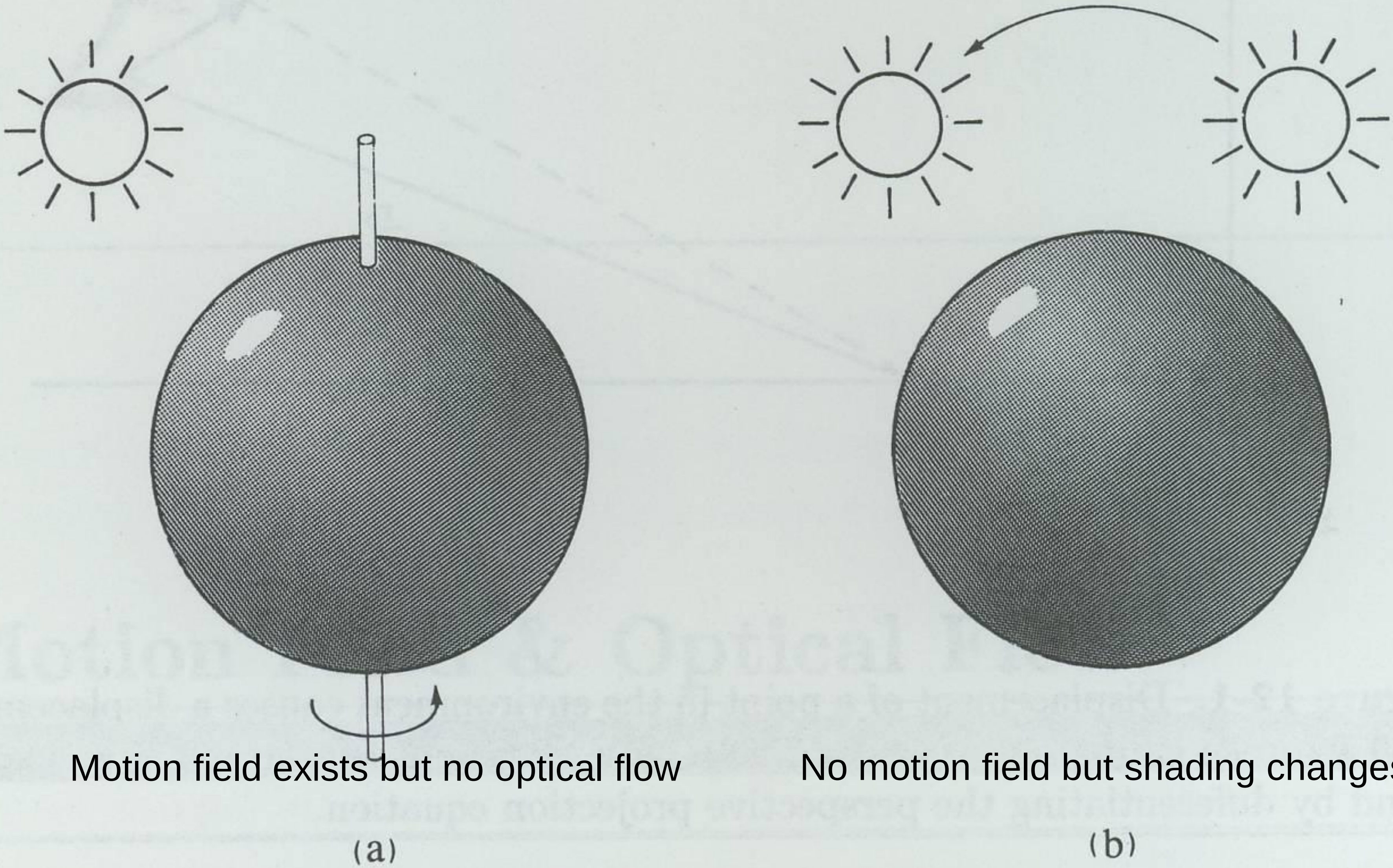
Optical Flow

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





Optical Flow ≠ Motion Field

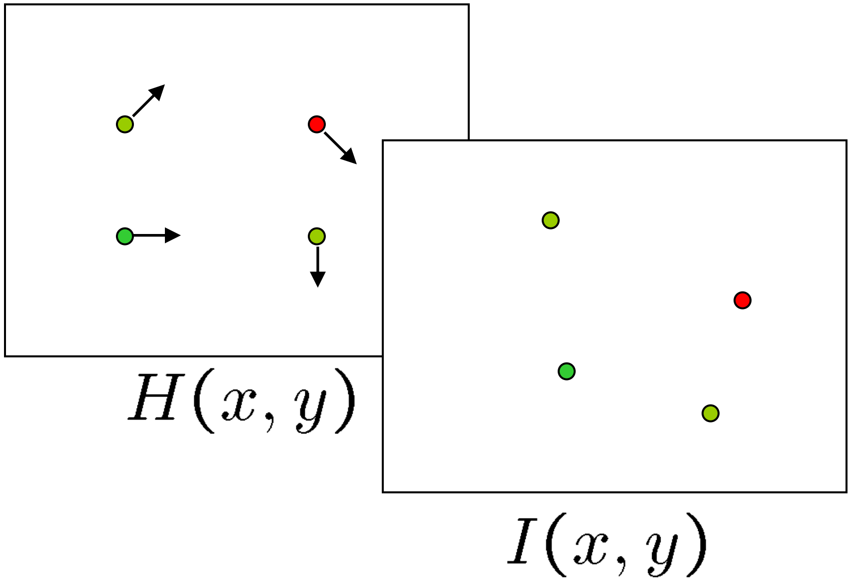


Motion field exists but no optical flow

No motion field but shading changes

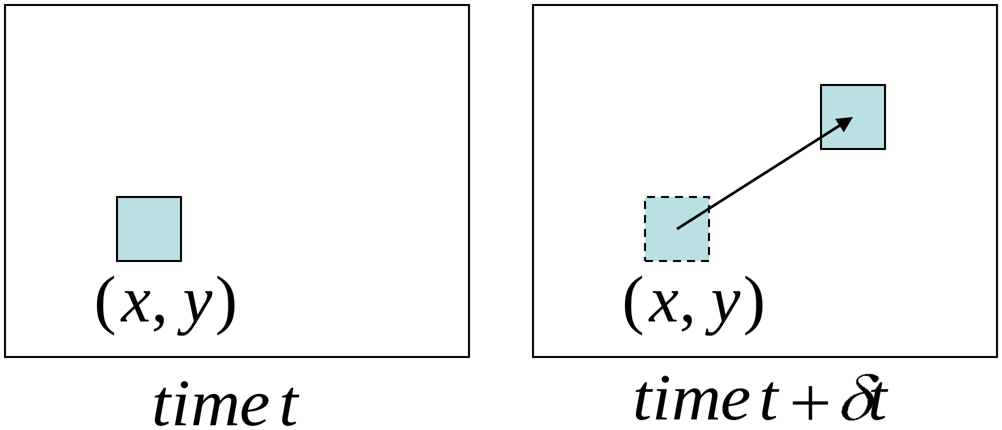
Problem Definition: Optical Flow

- How to estimate pixel motion from image H to image I?
  - Find pixel correspondences
    - 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 image I
    - For grayscale images, this is **brightness constancy**
  - **small motion**: points do not move very far

Optical Flow Constraint Equation



$(x + u\,\delta t, y + v\,\delta t)$

Optical Flow: Velocities (u, v)

Displacement:

$$(\delta x, \delta y) = (u\,\delta t, v\,\delta t)$$

– Assume brightness of patch remains same in both images:

$$E(x + u\,\delta t, y + v\,\delta t, t + \delta t) = E(x, y, t)$$

– Assume small motion: (Taylor expansion of LHS up to first order)

$$E(x, y, t) + \delta x \frac{\partial E}{\partial x} + \delta y \frac{\partial E}{\partial y} + \delta t \frac{\partial E}{\partial t} = E(x, y, t)$$

Optical Flow Constraint Equation

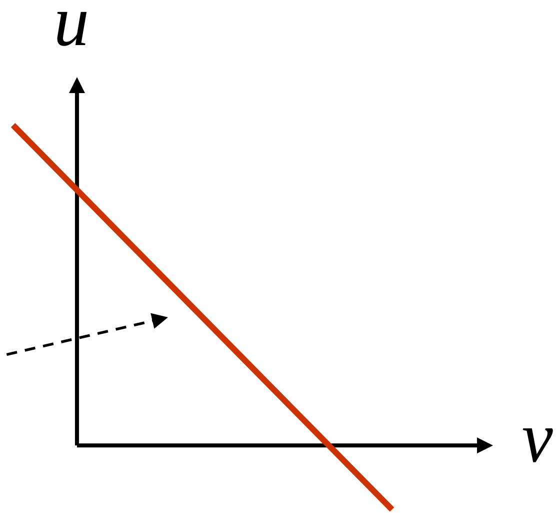
$$\delta x \frac{\partial E}{\partial x} + \delta y \frac{\partial E}{\partial y} + \delta t \frac{\partial E}{\partial t} = 0$$

Divide by δt and take the limit $\delta t \rightarrow 0$

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

Constraint Equation

$$E_x\, u + E_y\, v + E_t = 0$$



NOTE: (u,v) must lie on a straight line

We can compute E_x, E_y, E_t using gradient operators!

But, (u,v) cannot be found uniquely with this constraint!

Optical Flow Constraint

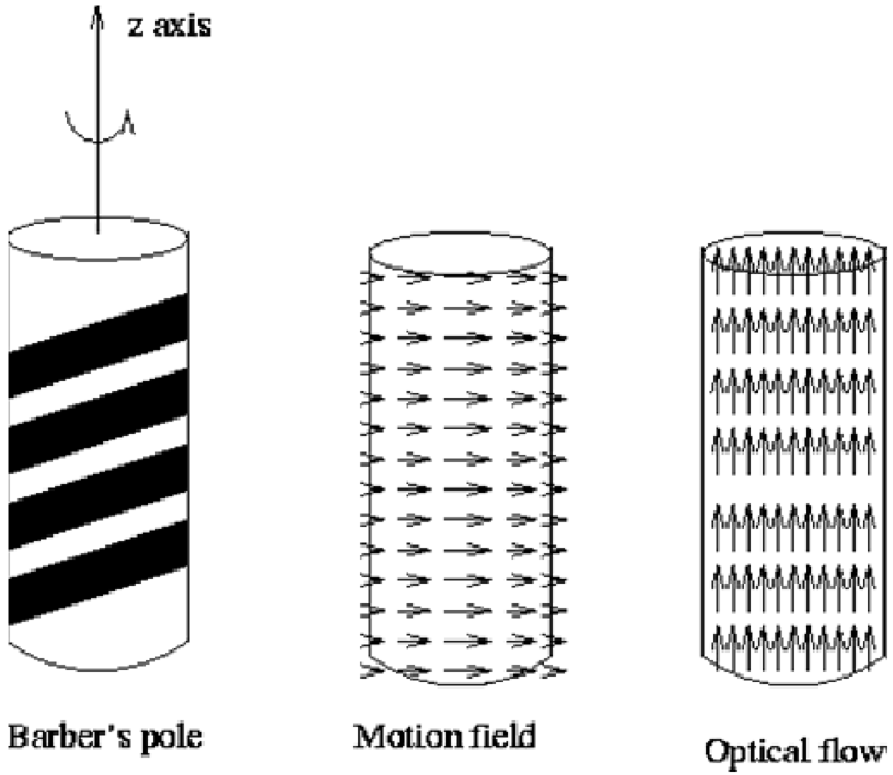
- Intuitively, what does this constraint mean?
  - The component of the flow in the gradient direction is determined.
  - The component of the flow parallel to an edge is unknown.

Topic: Aperture problem

- Optical Flow Constraint Equation
- Aperture problem.
- The Lucas & Kanade Algorithm

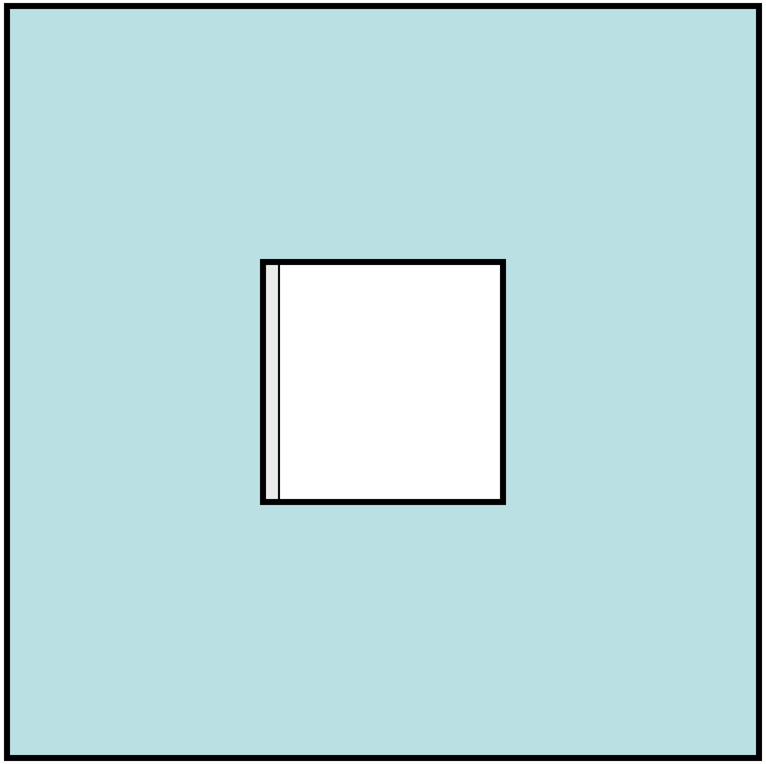
Optical Flow Constraint

Barber pole illusion



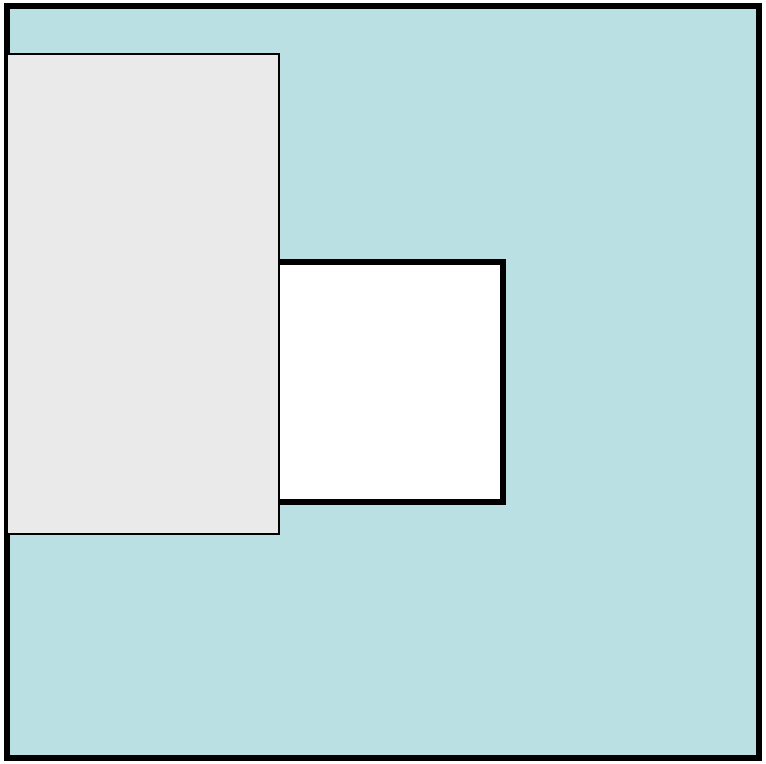
How does this show up visually? Known as the "Aperture Problem"

[Gary Bradski, Intel Research and Stanford SAIL]



Aperture Problem Exposed

[Gary Bradski, Intel Research and Stanford SAIL]



Motion along just
an edge is ambiguous

Computing Optical Flow

- Formulate Error in Optical Flow Constraint:

$$e_c = \iint_{image} (E_x u + E_y v + E_t)^2 \, dx \, dy$$

- We need additional constraints!

- Smoothness Constraint (as in shape from shading and stereo):

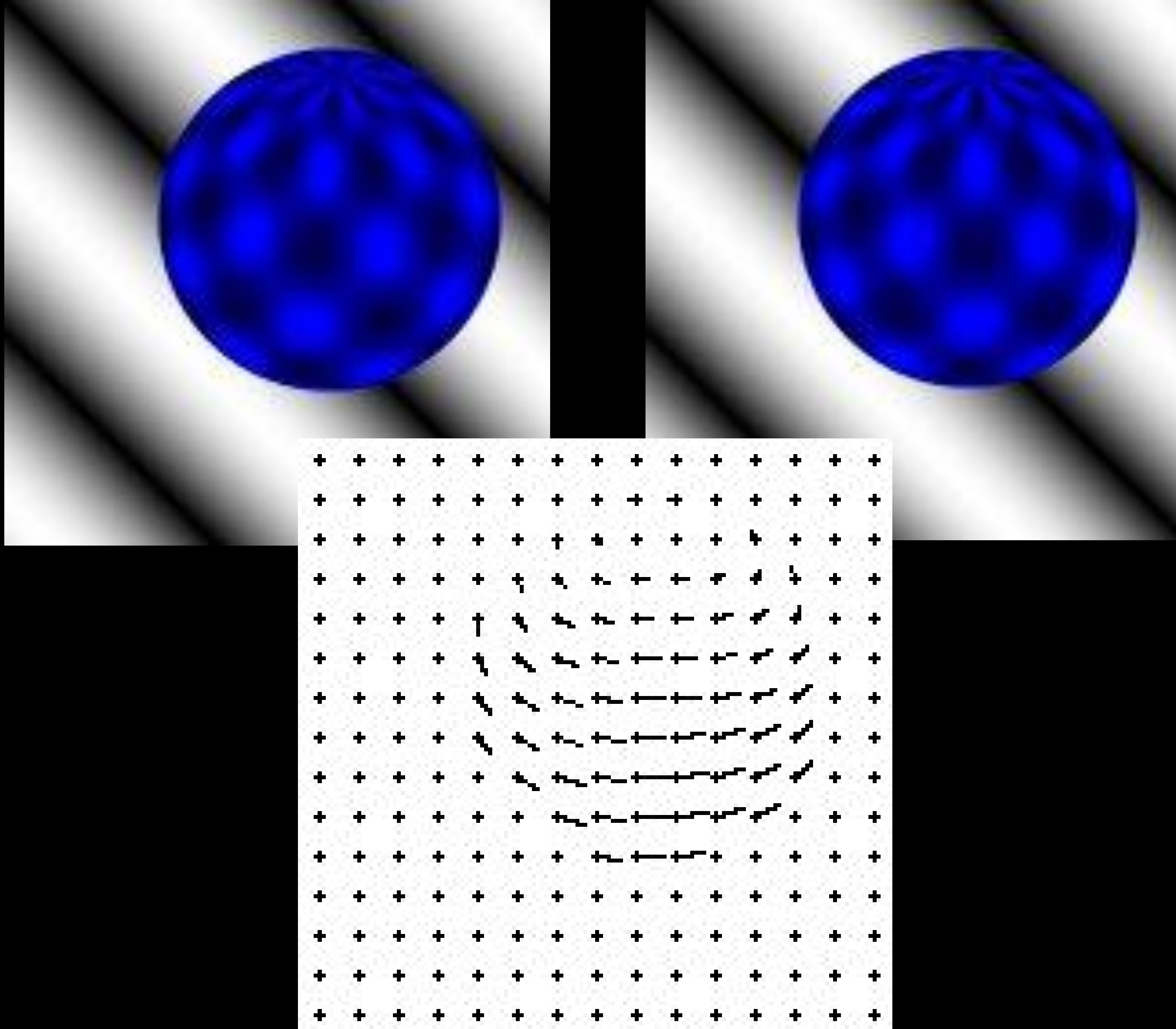
  Usually motion field varies smoothly in the image.
  So, penalize departure from smoothness:

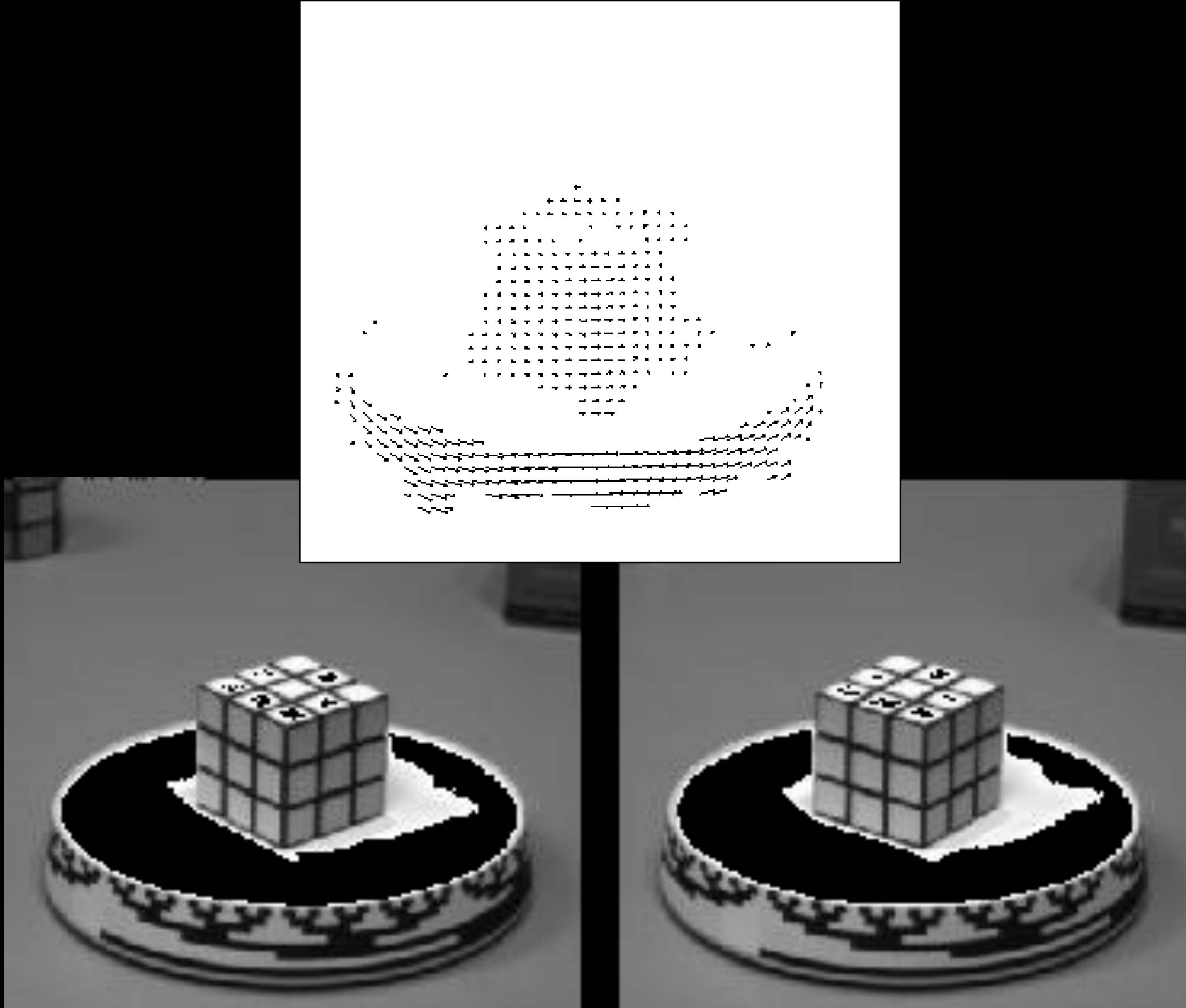
$$e_s = \iint_{image} (u_x^2 + u_y^2) + (v_x^2 + v_y^2) \;\; dx \, dy$$

- Find (u,v) at each image point that MINIMIZES:

$$e = e_s + \lambda e_c$$

weighting factor

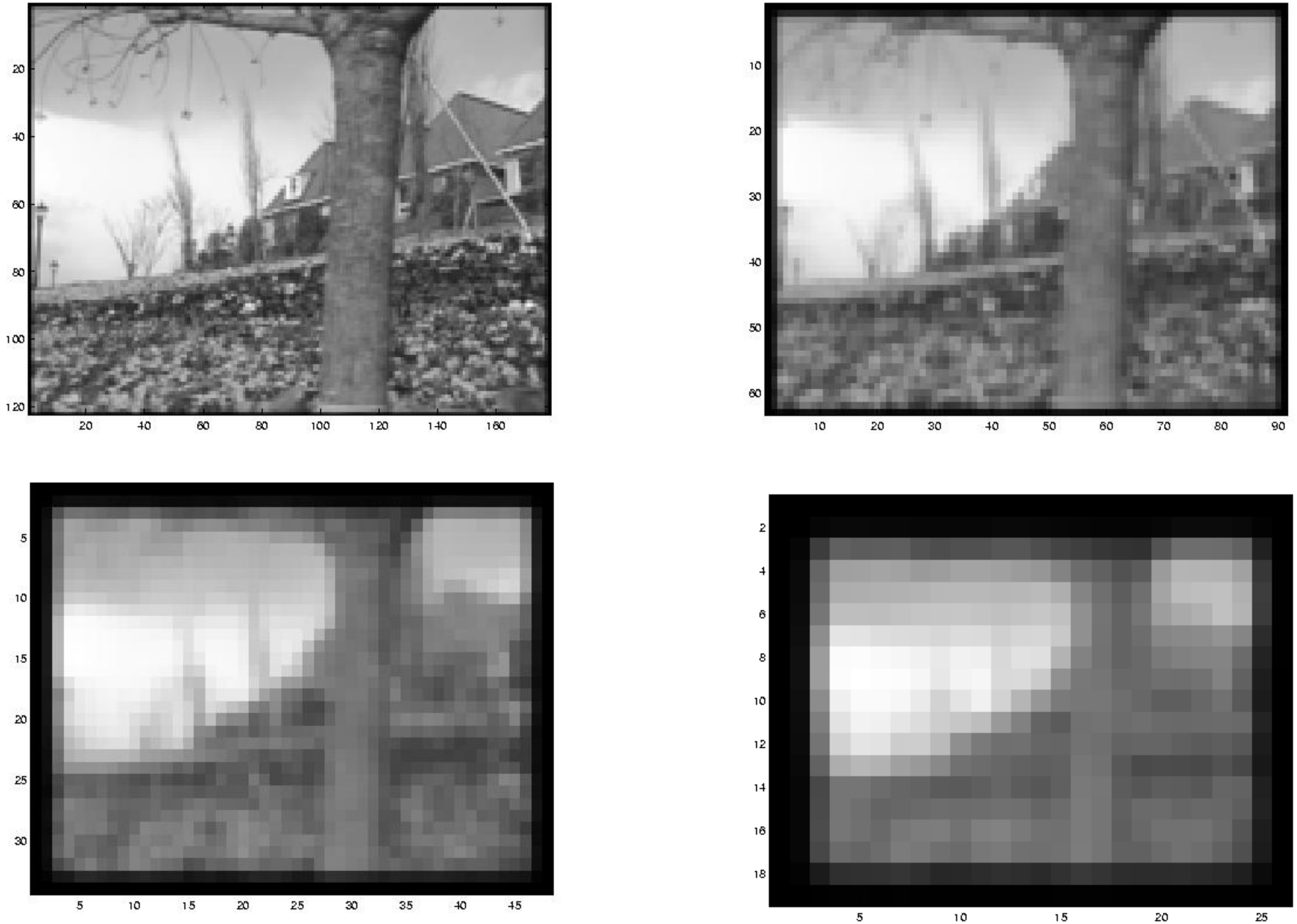




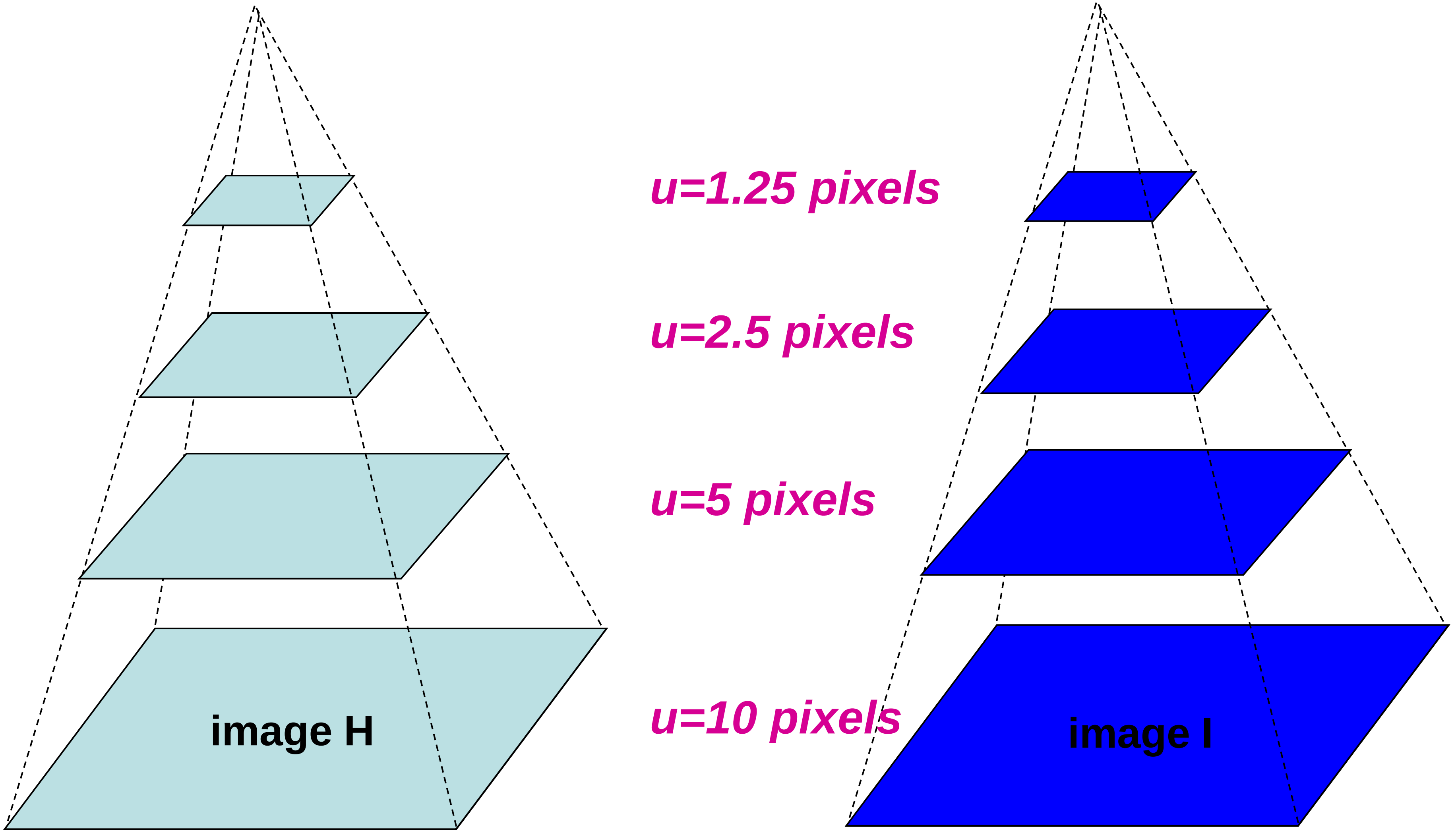
Revisiting the Small Motion Assumption

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

Reduce the Resolution!



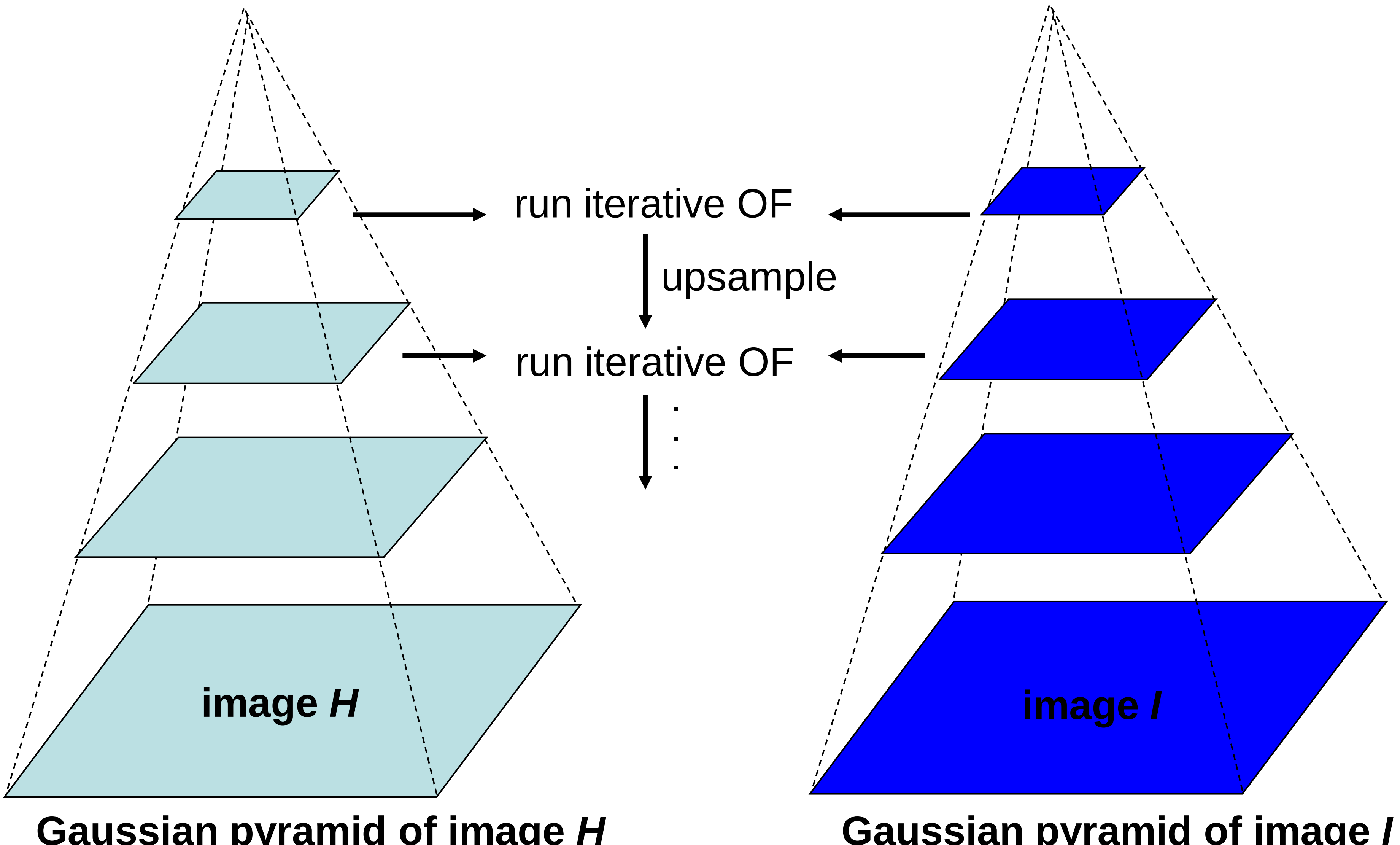
Coarse-to-fine Optical Flow Estimation



Gaussian pyramid of image *H*

Gaussian pyramid of image *I*

Coarse-to-fine Optical Flow Estimation



Types of OF methods

- Differential
  - Horn and Schunck [HS80], Lucas Kanade [LK81], Nagel [83].
- Region-based matching
  - Anandan [Anan87], Singh [Singh90], Digital video encoding standards.
- Energy-based
  - Heeger [Heeg87]
- Phase-based
  - Fleet and Jepson [FJ90]

Open problem!
Current solutions are not good enough!

Topic: The Lucas & Kanade Algorithm

- Optical Flow Constraint Equation
- Aperture problem.
- The Lucas & Kanade Algorithm

The Lucas & Kanade Method

- 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!

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

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

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Lukas-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 Lukas & Kanade (1981)
    - described in Trucco & Verri reading

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

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?

- $\mathbf{A^T A}$ 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)

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Eigenvectors of A^TA

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

- Suppose (x,y) is on an edge. What is A^TA?
  - gradients along edge all point the same direction
  - gradients away from edge have small magnitude

$$\left(\sum \nabla I (\nabla I)^T\right) \approx k \nabla I \nabla I^T$$

$$\left(\sum \nabla I (\nabla I)^T\right) \nabla I = k \|\nabla I\| \nabla I$$

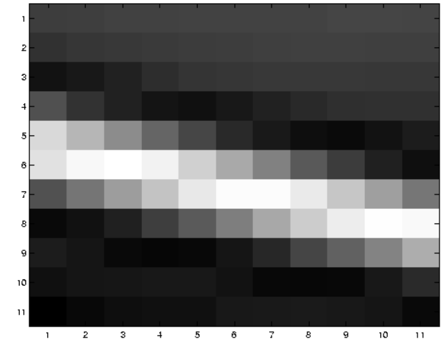
  - ∇I is an eigenvector with eigenvalue $k\|\nabla I\|$
  - What's the other eigenvector of A^TA?
    - let N be perpendicular to ∇I

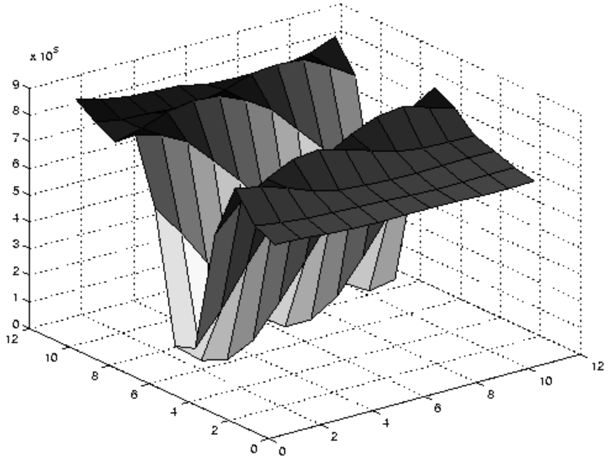
$$\left(\sum \nabla I (\nabla I)^T\right) N = 0$$

    - N is the second eigenvector with eigenvalue 0
- The eigenvectors of A^TA relate to edge direction and magnitude

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Edge





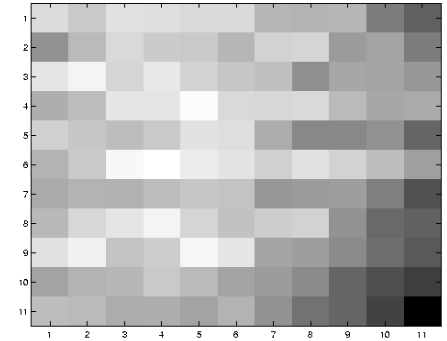
$$\sum \nabla I (\nabla I)^T$$

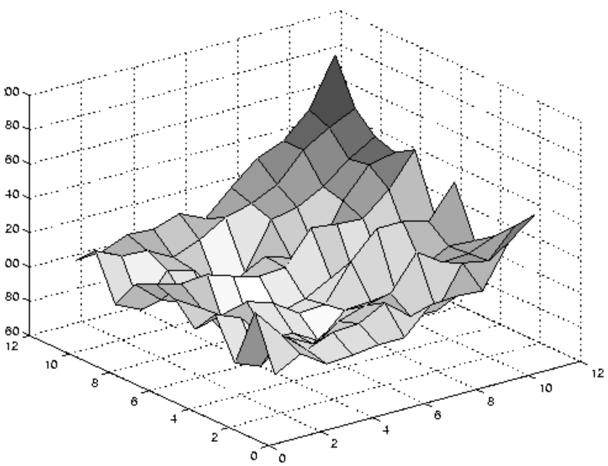
– large gradients, all the same

– large λ_1, small λ_2

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Low texture region





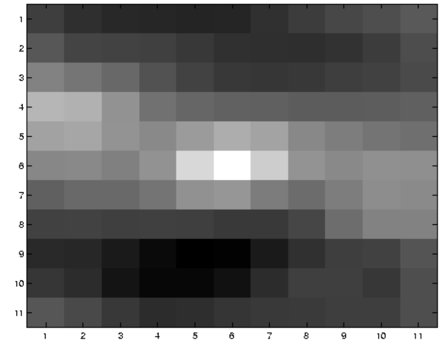
$$\sum \nabla I (\nabla I)^T$$

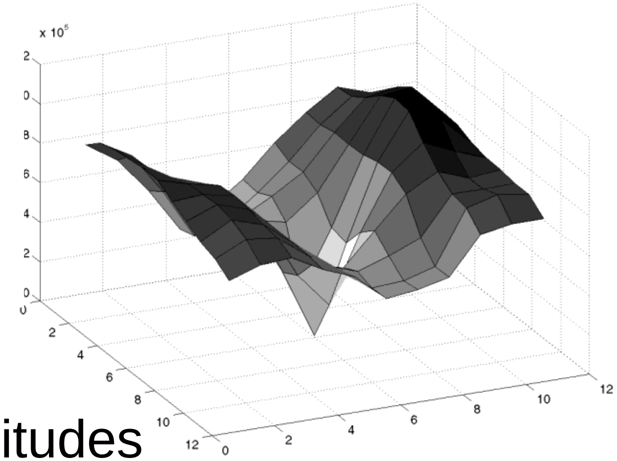
– gradients have small magnitude

– 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

Sparse Motion Field

- We are only confident in motion vectors of areas with two strong eigenvectors.
  - Optical flow.
- Not so confident when we have one or zero strong eigenvectors.
  - Normal flow (apperture problem).
  - Unknown flow (blank-wall problem).

Summing all up

- Optical flow:
  - Algorithms try to approximate the true motion field of the image plane.
  - The Optical Flow Constraint Equation needs an additional constraint (e.g. smoothness, constant local flow).
  - The Lucas Kanade method is the most popular Optical Flow Algorithm.
- What applications is this useful for?
- What about block matching?

Resources

- Barron, “Tutorial: Computing 2D and 3D Optical Flow.”, http://www.tina-vision.net/docs/memos/2004-012.pdf
- CVonline: Optical Flow - http://homepages.inf.ed.ac.uk/cgi/rbf/CVONLINE/entries.pl?TAG518
- Fast Image Motion Estimation Demo http://extra.cmis.csiro.au/IA/changs/motion/