# Motion

- What is the function of motion?

- What should we compute?

# Motion functions and computationas

The computation and perception of motion is complex because there are many different behavioral functions and means of motion:

- optomotor response: control eye, head, and body to stabilize gaze
- visual control of limbs
- self motion: navigating through the environment
- tracking, detecting, and identifying objects
- computation of 3-D structure and position
- object and scene segmentation

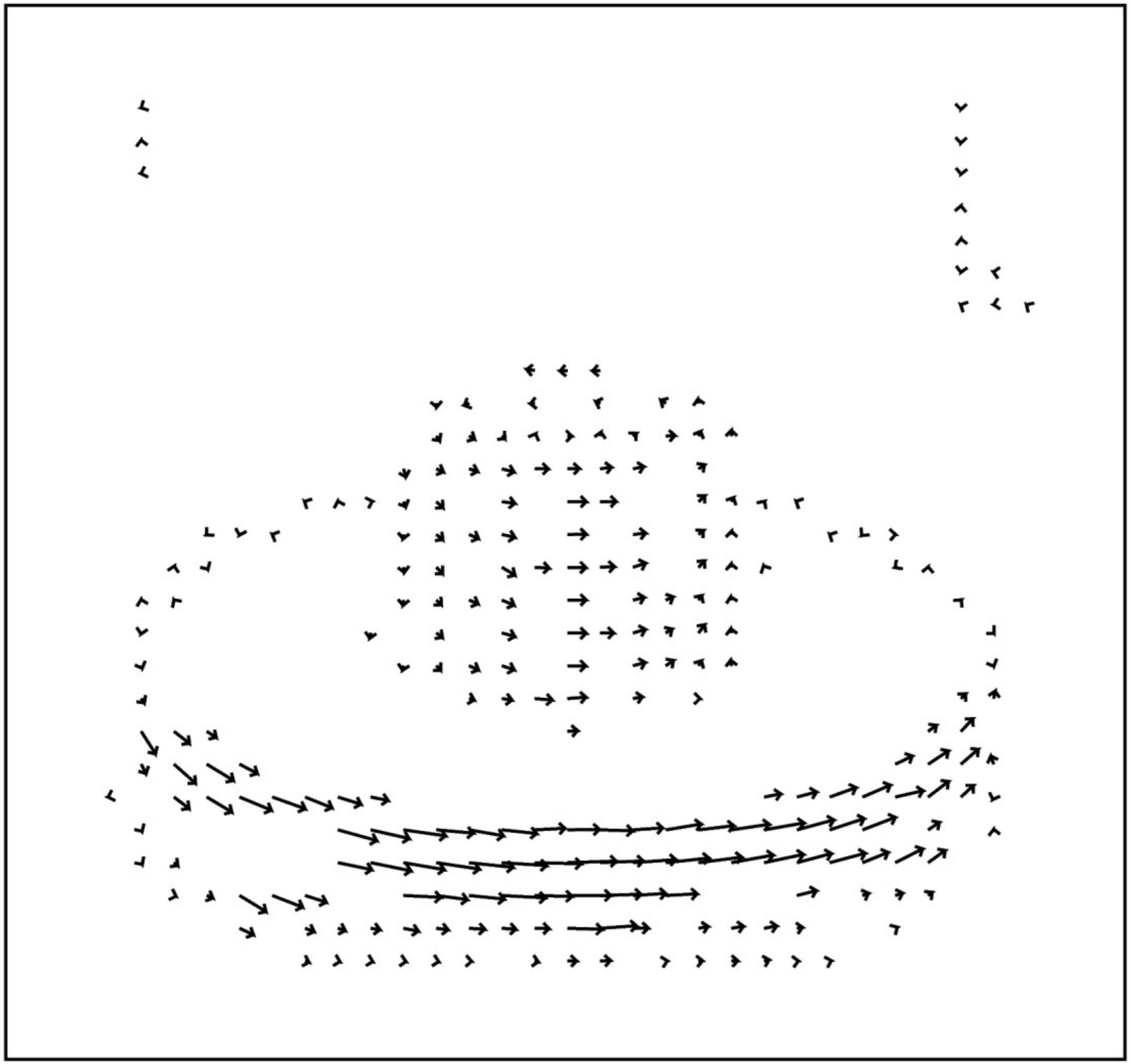
Some of what makes motion computation very complex:

- time-varying intensity on retina is not necessarily motion
- must distinguish between motion of object vs motion of self
- motion on retina is 2D projection of 3D object
- local motion often conflicts with global motion
- the visual scene can be complex
- noise and image variation limits ability to detect change
- true motion is actually changing

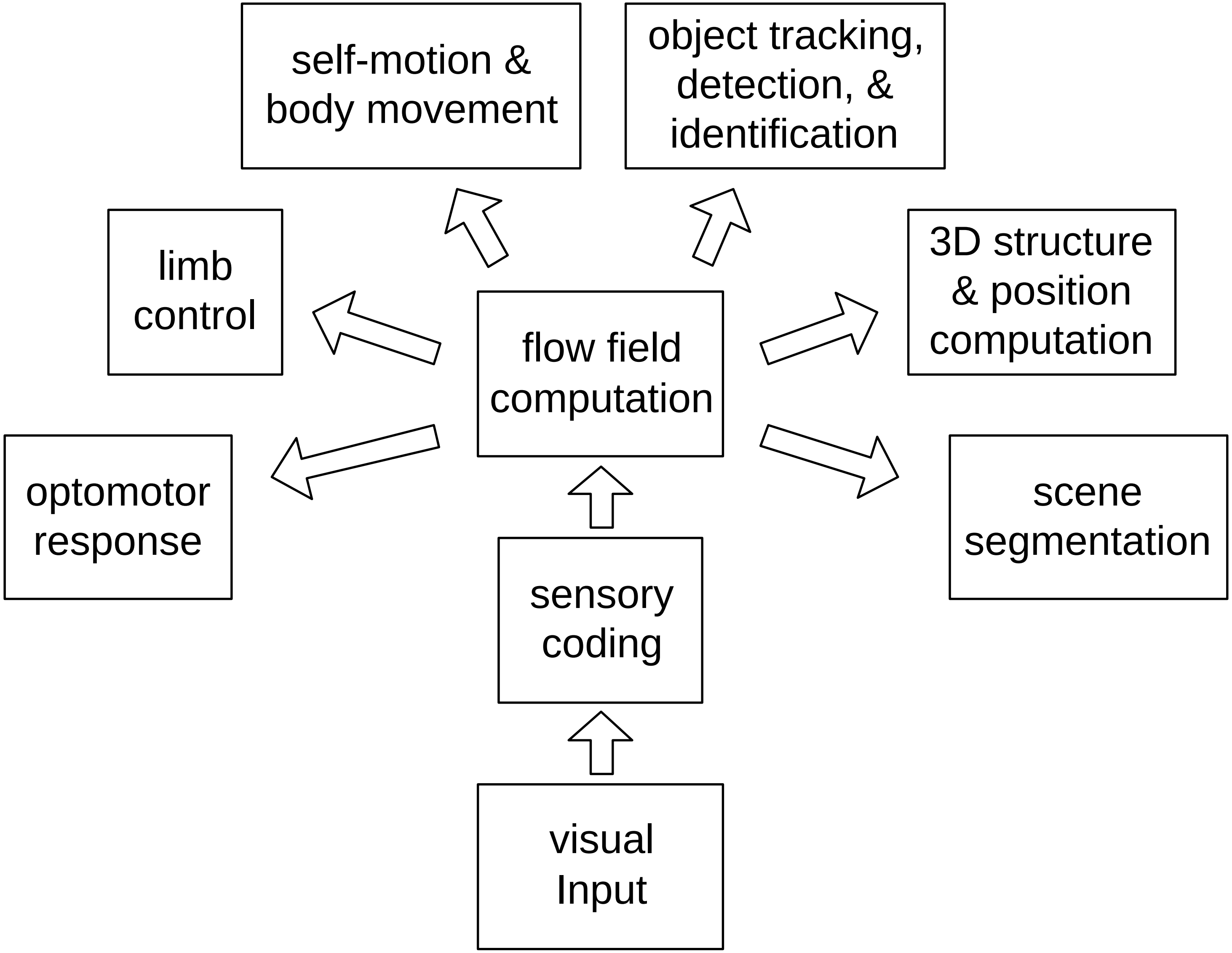
# One answer: compute motion flow fields

- A flow field assigns a motion vector to each point on the image.
- This is an inherently ambiguous problem (e.g. what should the motion be on the table surface?)
- Flow fields generally represent 2D motion, i.e. motion of the projected image.

# Horn and Schunk flow field solution

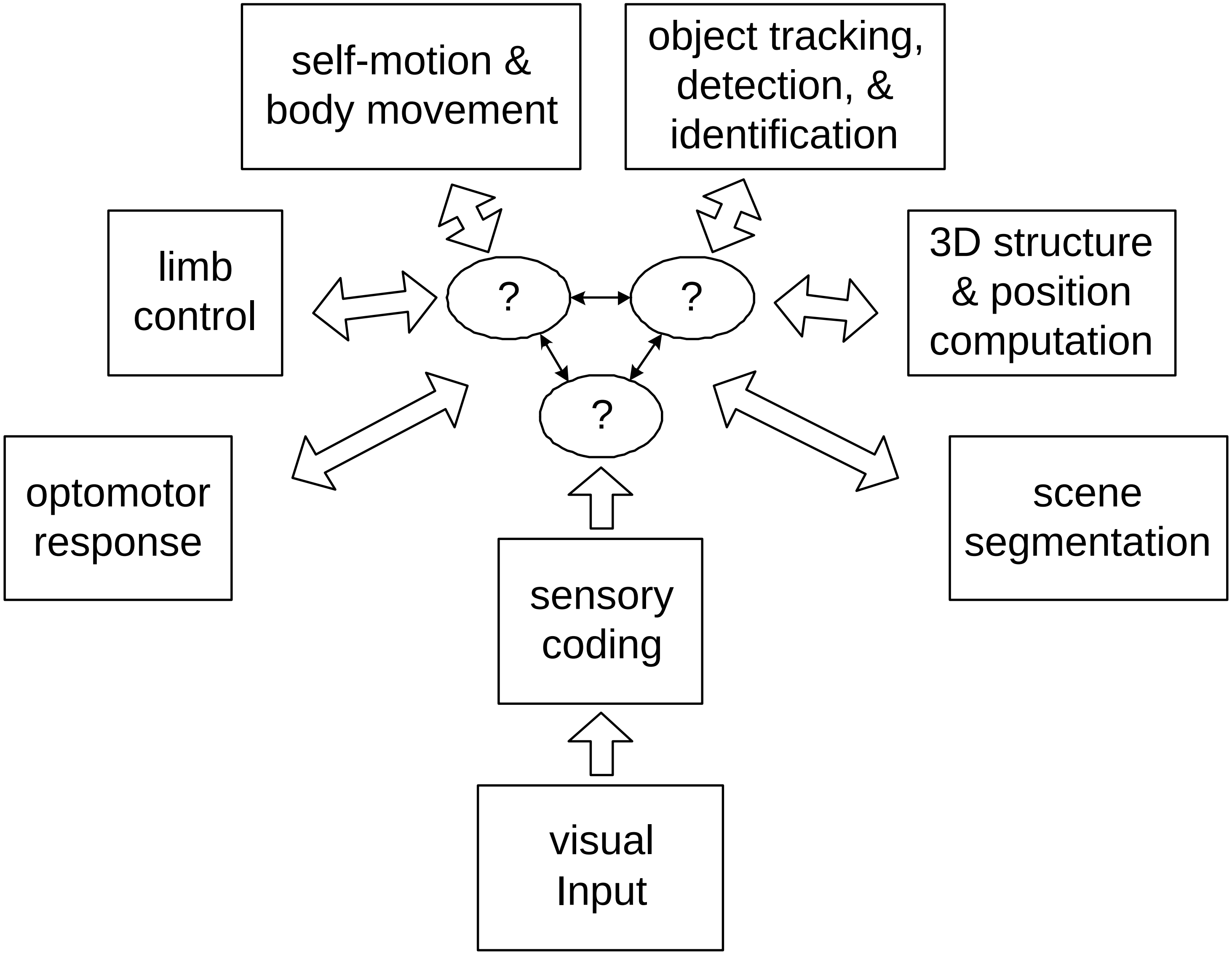


# General or Specialized Algorithms?




- Can we achieve higher level computations from a flow field?

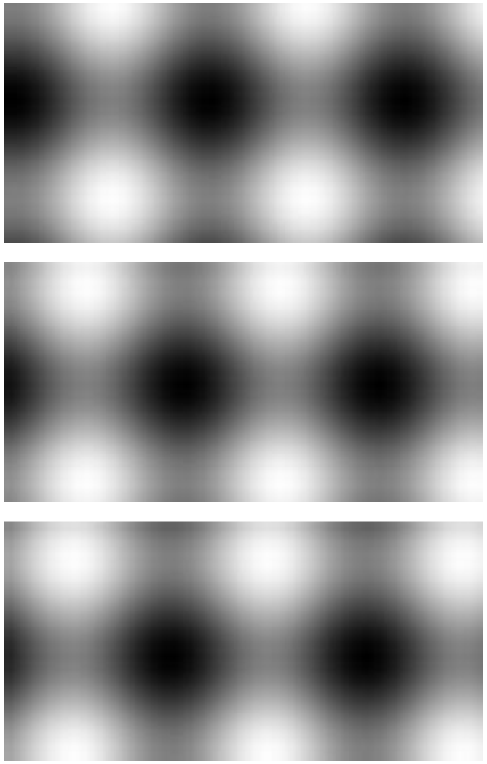
# What should we compute?




- estimation ideal flowfield depends on higher level information
- other representations might be sufficient for required tasks

# Estimating optical flow

Idea: estimate motion using spatio-temporal brightness changes.



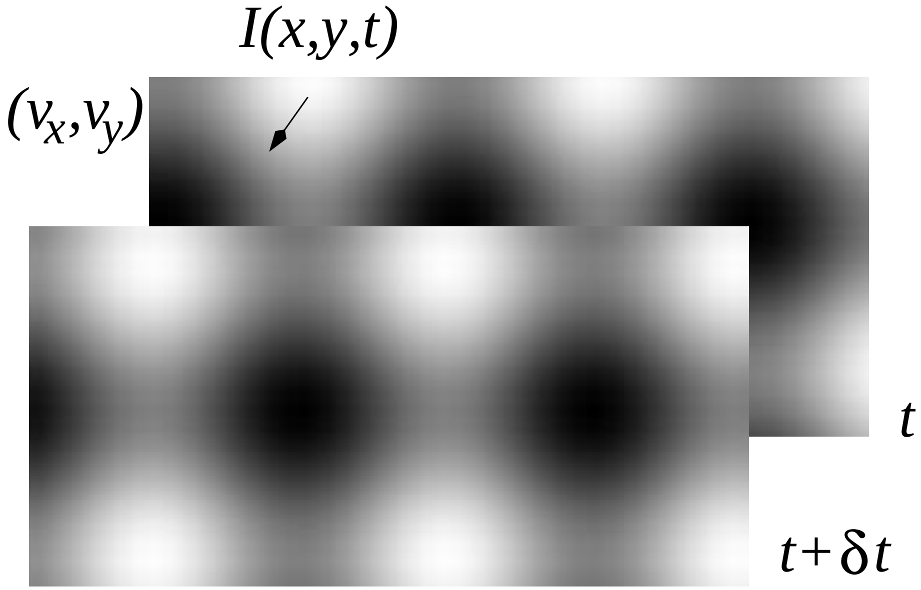
$t$

$t + \delta t$

$t + 2\delta t$

- assume in small regions displacement of point does not change brightness
- direction in which brightness is constant is diretion of motion
- How do we express this mathematically?

# The motion gradient equation




Assume a point $I(x,y,t)$ does not change intensity after $\delta t$

$$I(x,y,t) = I(x+\delta x, y+\delta y, t+\delta t)$$

We can also describe this by Talyor expanding around the current point

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

Subtracting $I(x,y,t)$ from both sides and dividing through by $\delta t$

$$\frac{\partial I}{\partial x}\frac{\delta x}{\delta t} + \frac{\partial I}{\partial y}\frac{\delta y}{\delta t} + \frac{\partial I}{\partial t} + \mathcal{O}(\delta t) = 0$$

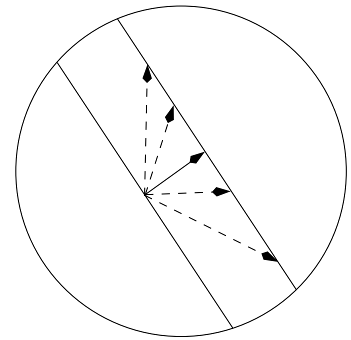
# Constraint lines from motion gradient equation

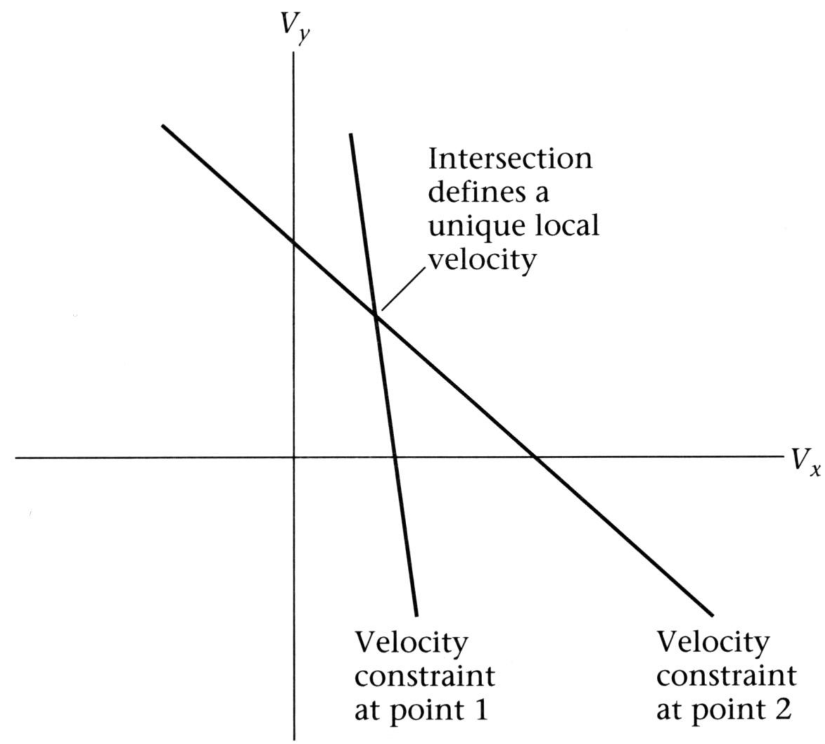
In the limit as $\delta t \rightarrow 0$ we have the *motion gradient equation*

$$\begin{aligned}\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} &= 0 \\ I_x u + I_y v + I_t &= 0\end{aligned}$$

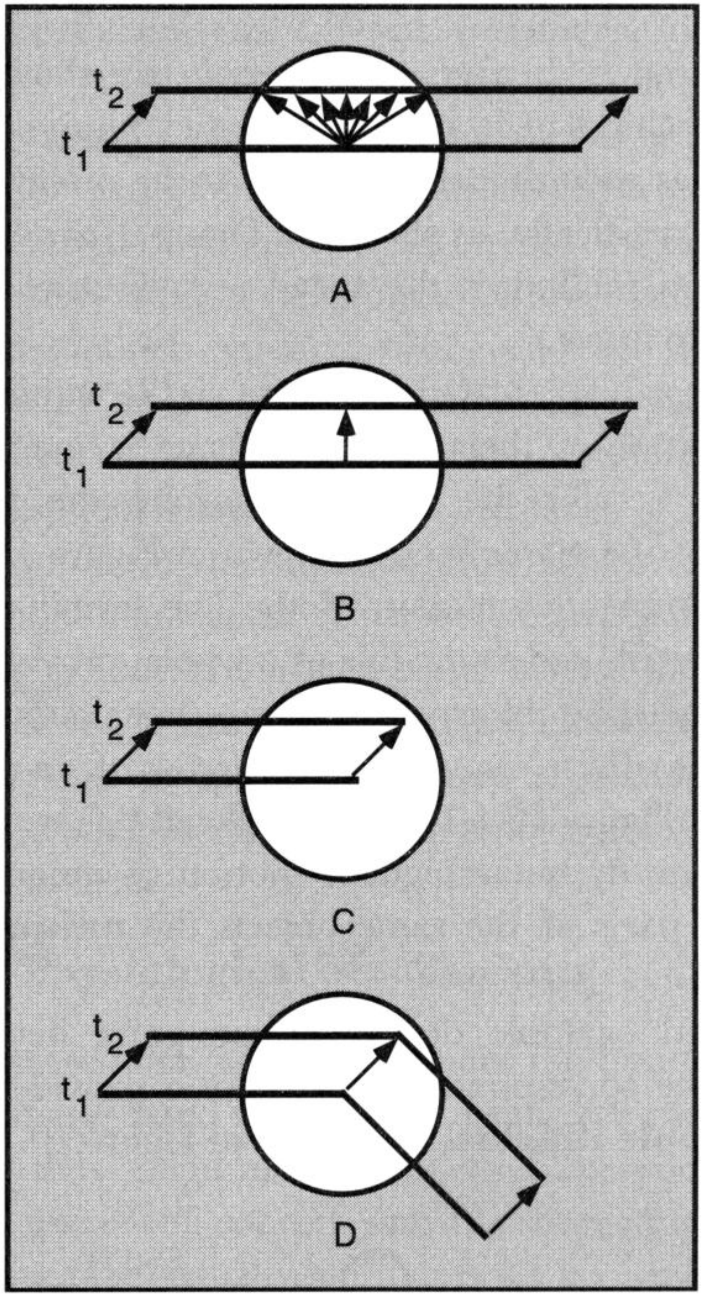
$$\Rightarrow \quad (I_x, I_y) \cdot (u, v) = -I_t$$

- $I_a = \delta I / \delta a$, $u = dx/dt$, $v = dy/dt$
- relates intensity changes to motion
- one equation and two unknowns
- mathematical expression of the *aperture problem*
- Each point defines a constraint line





# The aperture problem

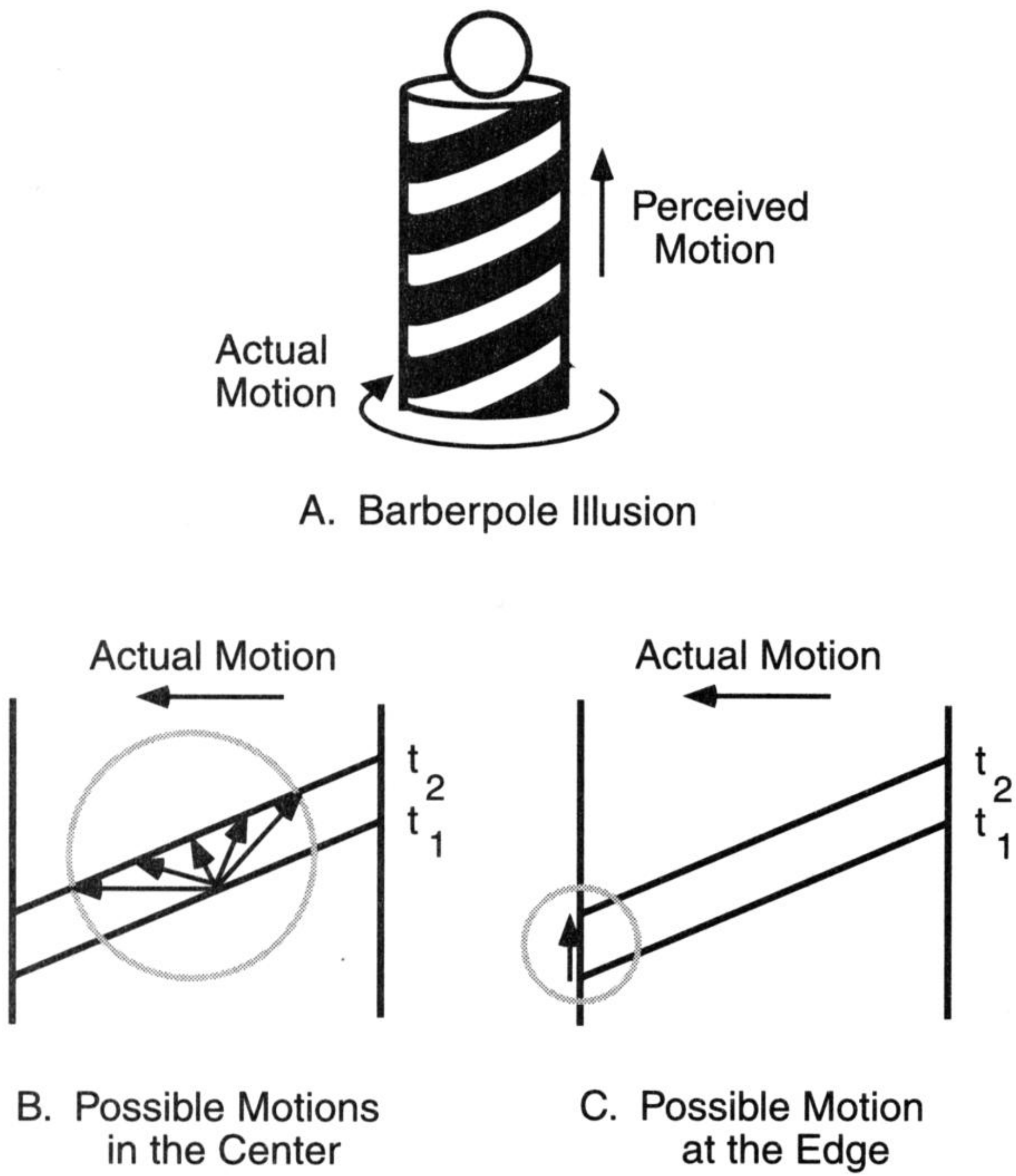



(A) Motion of line is ambiguous because of ambiguous correspondence

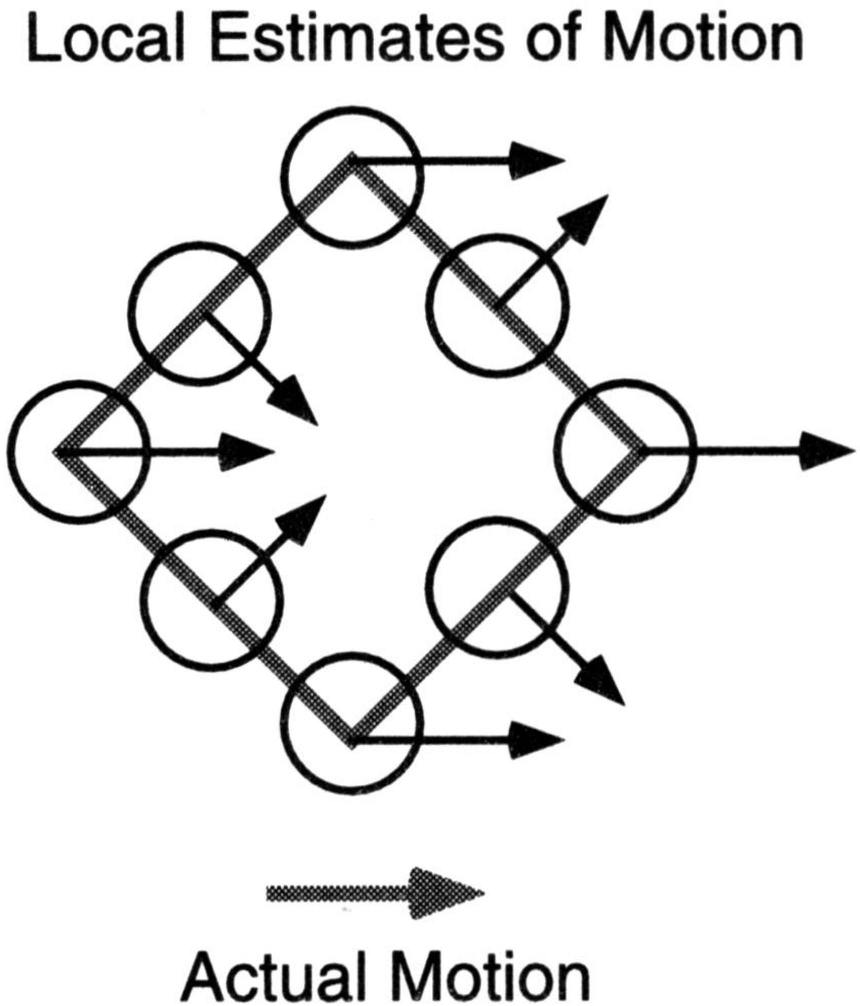
(B) Perceived motion is upward, unless unique points are visible (C,D)

This is an example of the important distinction between local and global motion.

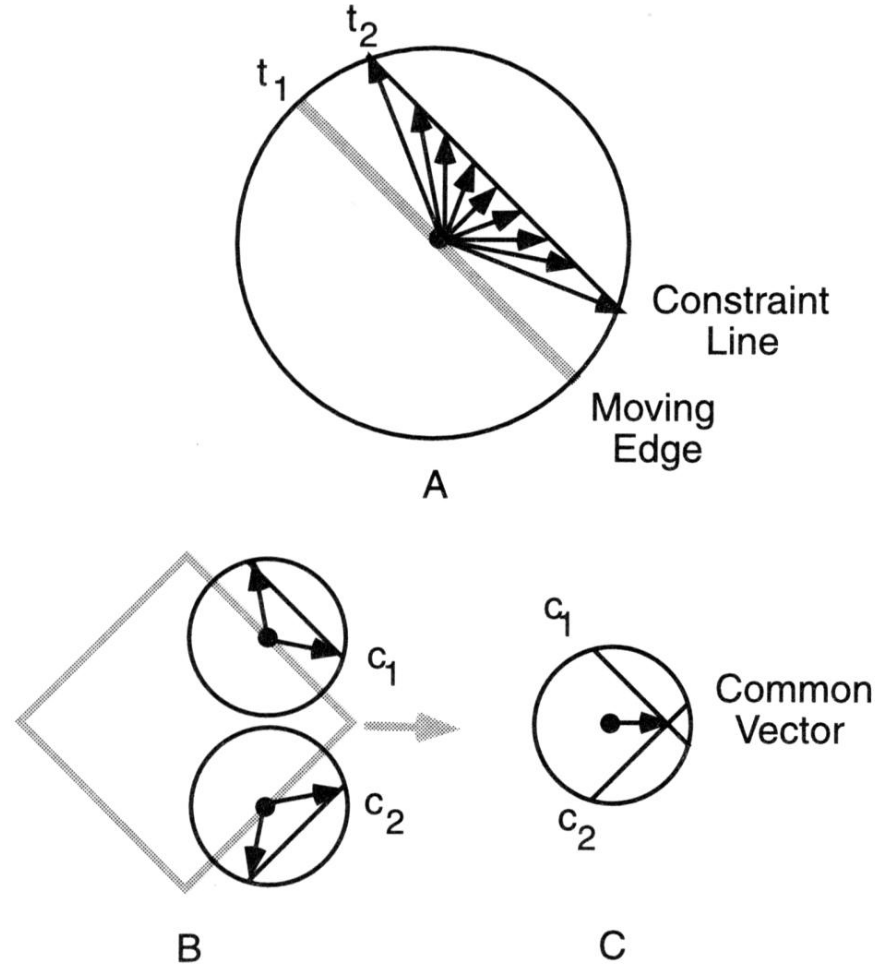
# The barberpole illusion



# Resolving ambiguity in local motion estimates

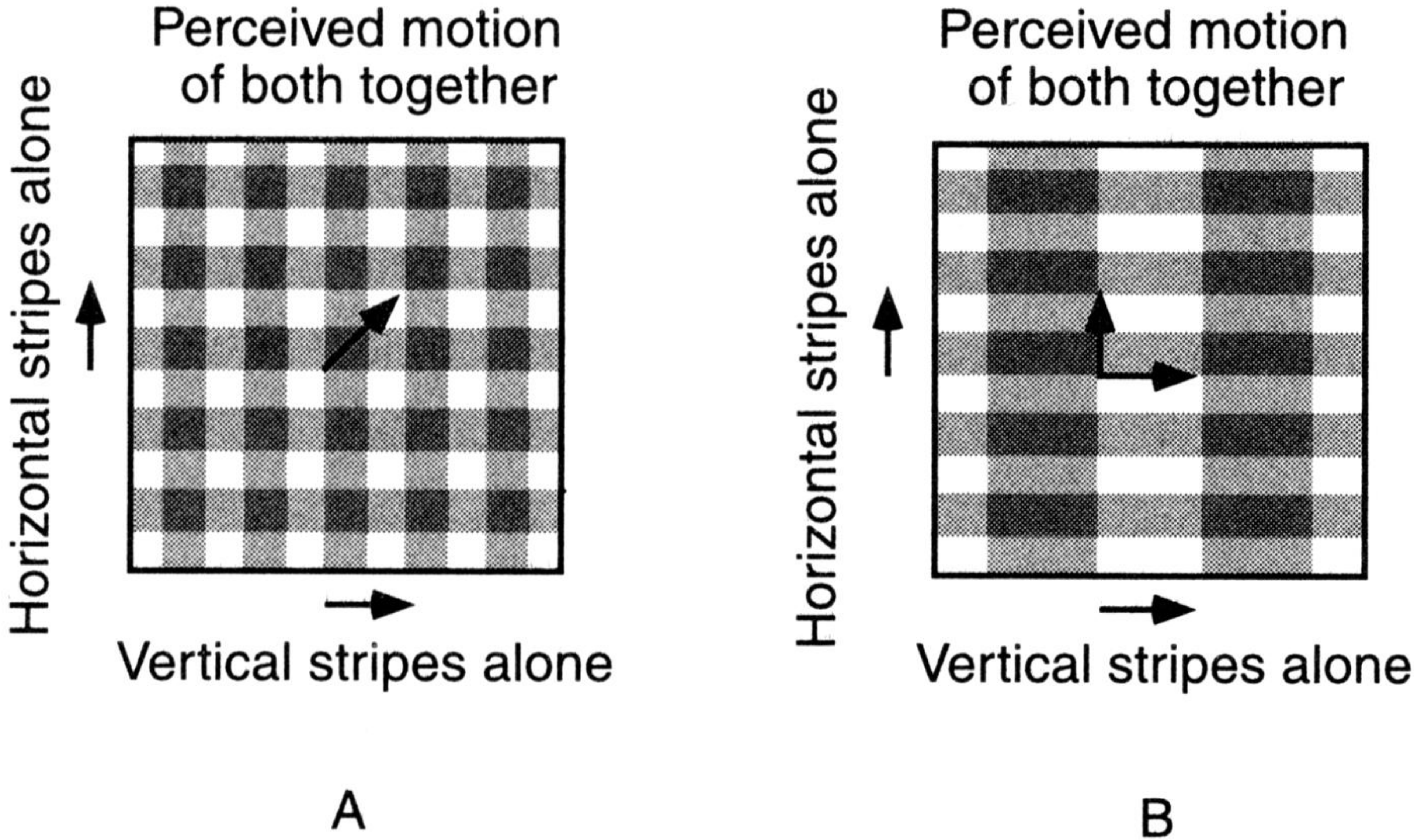



Local estimates are ambiguous.



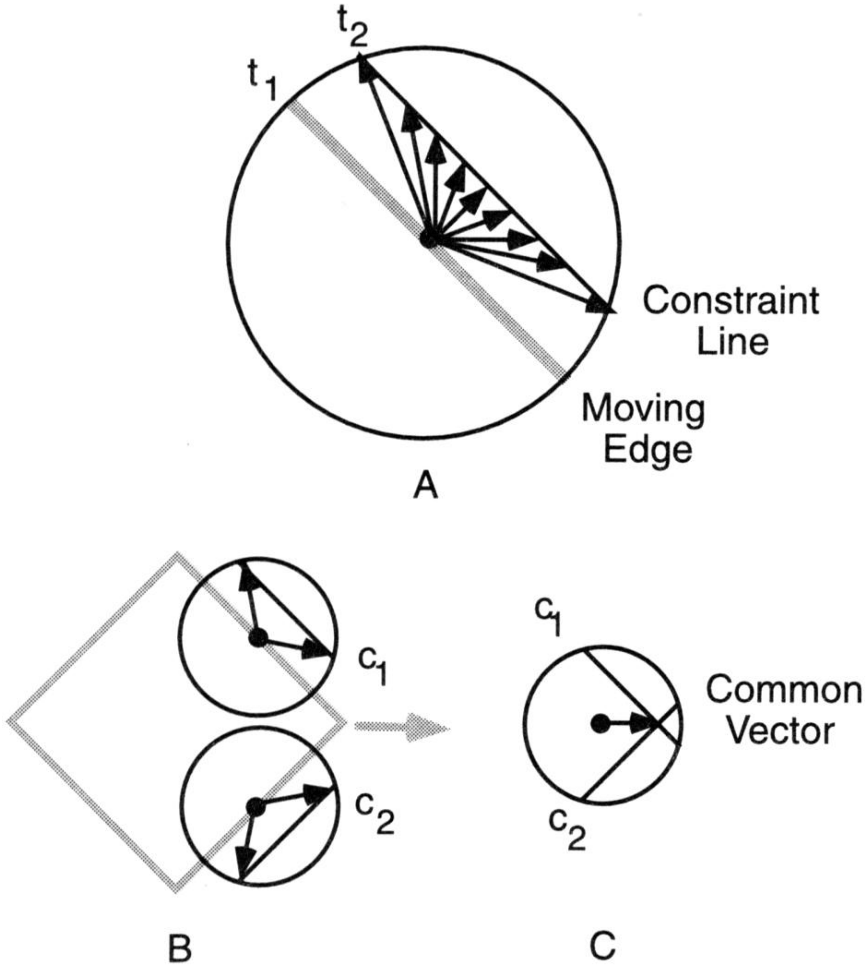

Only consistent interpretation is common points.

# Object motion: Plaids




Plaid motion
Plaid components
Transparent surface motion

# Contraint lines only work in cases of minimal noise




Need a different approach to contrain solutions.

# Regularization of ill-posed problems

Motion estimation is an *ill-posed* problem

- solution is not unique
- need to add constraints
- local smoothness is most natural

Idea: minimize the global cost function by *regularization*,
i.e. find $u(x, y), v(x, y)$ that minimizes

$$\int\int (I_x u + I_y v + I_t)^2 + \alpha^2 (u_x^2 + u_y^2 + v_x^2 + v_y^2)\, dx\, dy$$

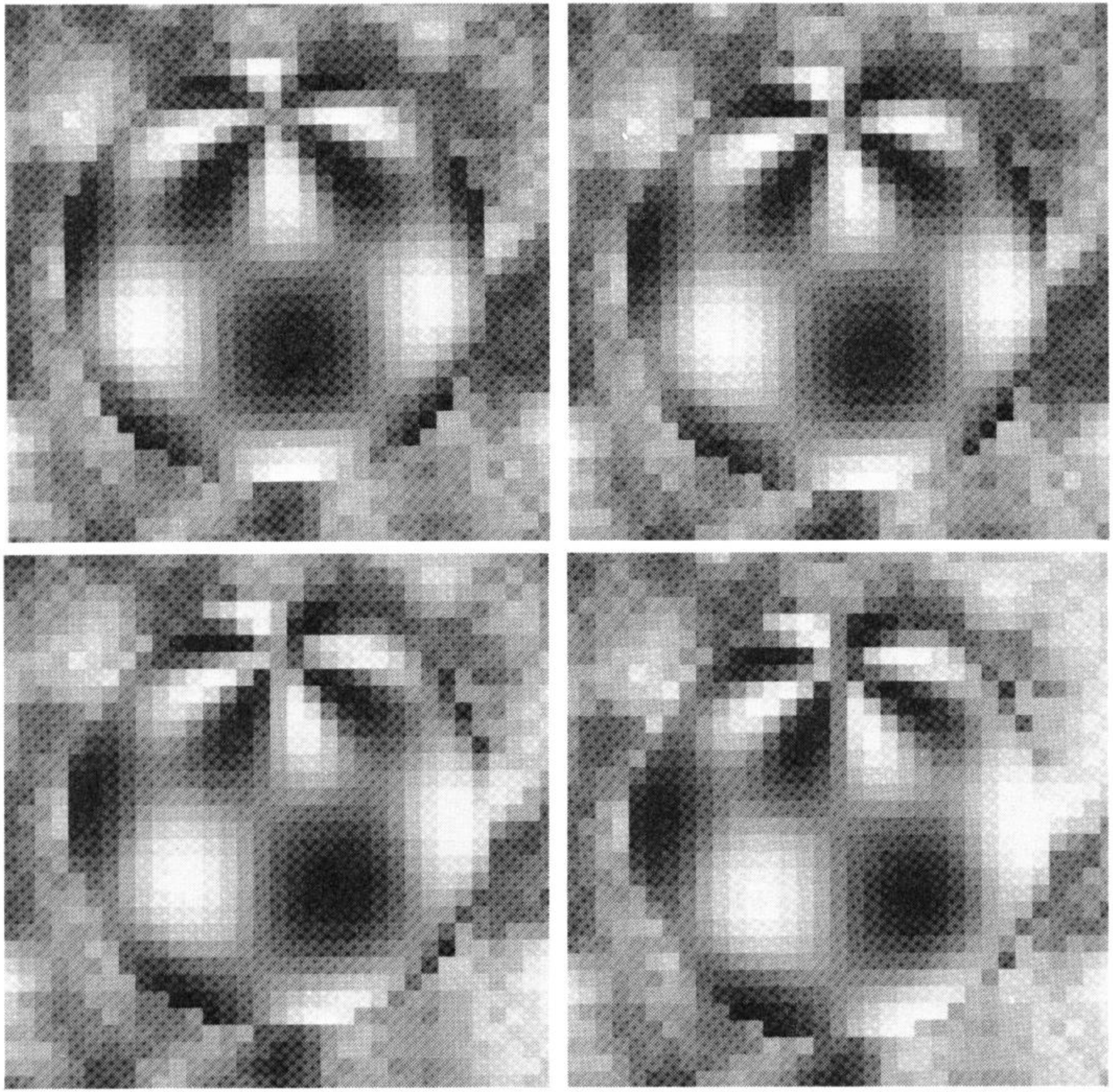
- first term minimizes deviation from motion gradient equation
- second term minimizes the square of the magnitude of the gradient of the optical flow.
- large $\alpha \Rightarrow$ greater smoothness

# Implemtation

Have to solve iteratively. Issues:

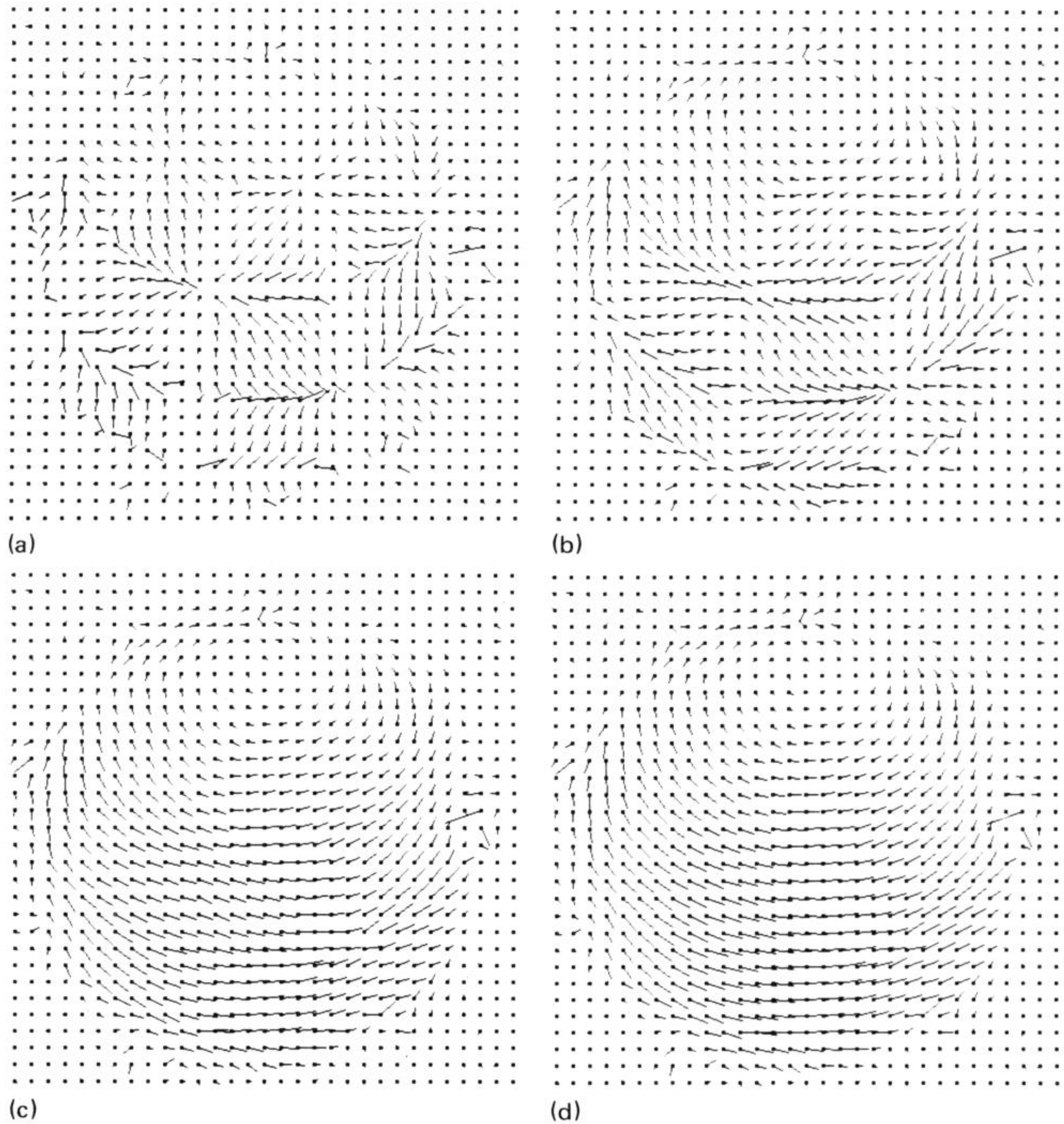
- how to estimate the derivatives
  - simplest first-order differences
  - second-order differences
- noise in the image
  - often need spatiotemporal pre-smoothing
  - combine velocities at multiple spatial scales
  - combine velocities over time

# Example: Horn and Schunk (1981)



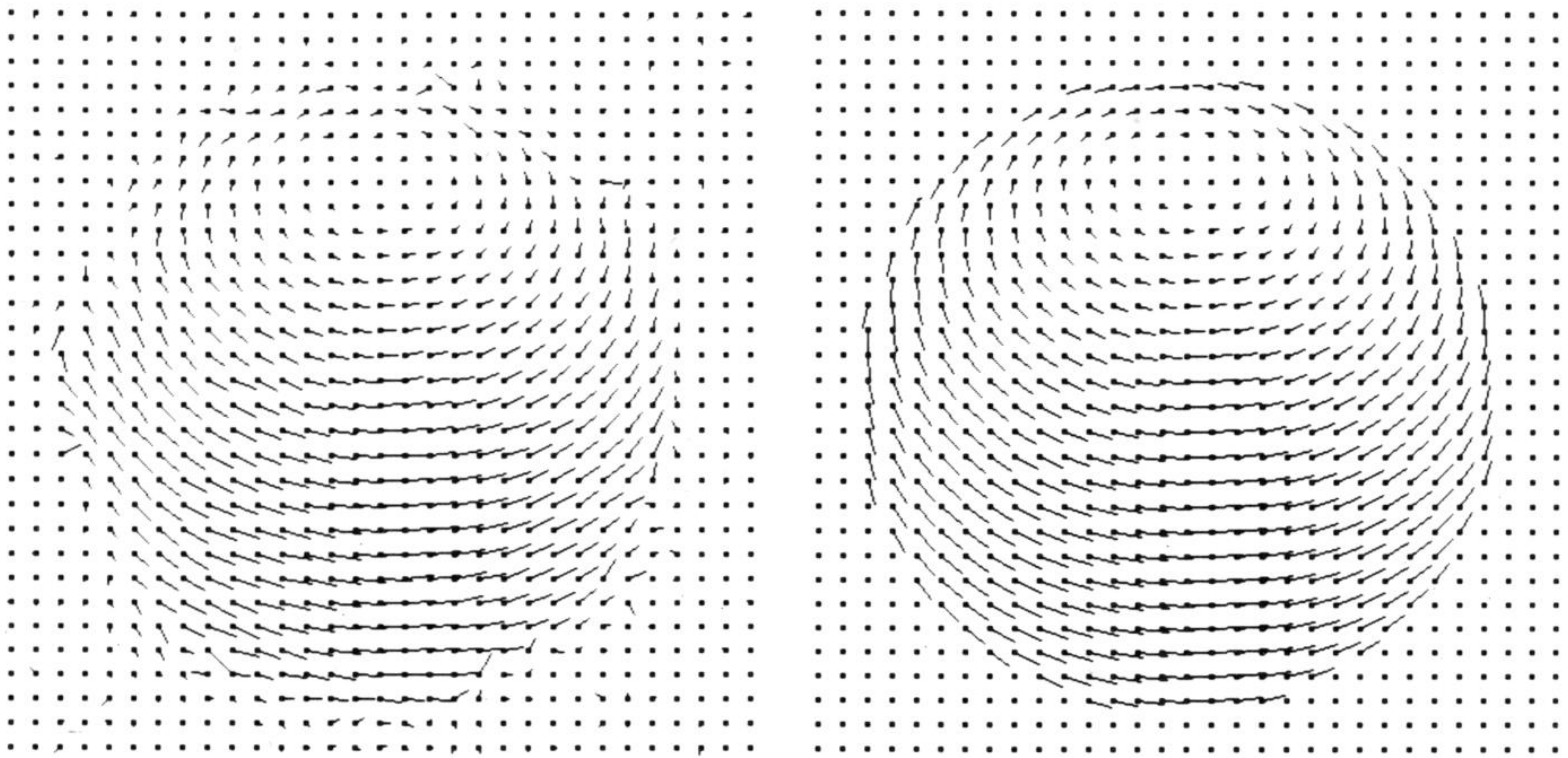
Slowly rotating synthetic sphere on noisy background.

# Solution of optical flow is iterative




- Iterations 1,4,16, and 64.
- Note regularization of non-smooth local velocities

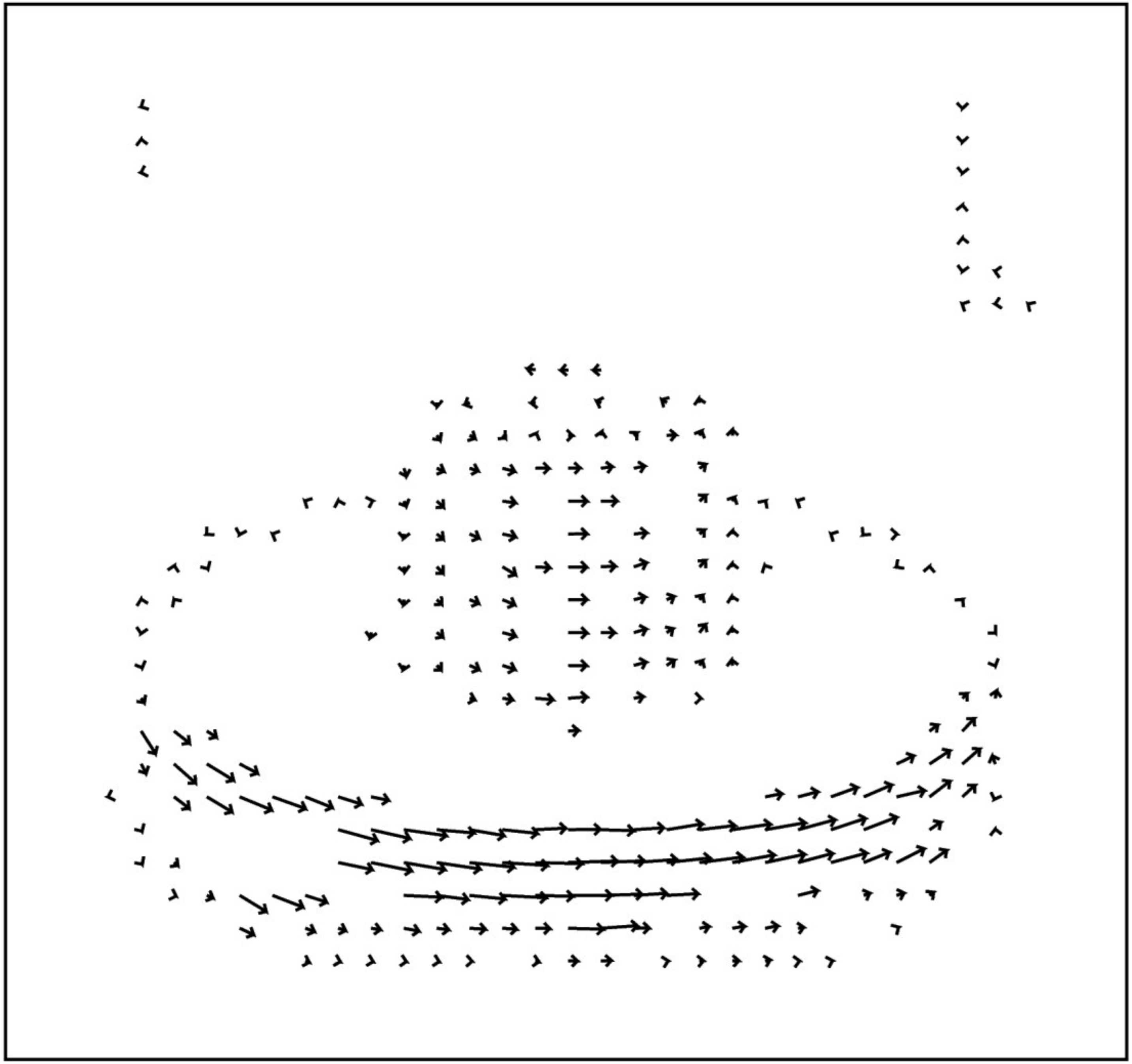
# Accuracy improves with more iterations



After many more iterations.

# Realistic examples

# Horn on Schunk algorithm flow field solution



# Another approach to smoothness

Weighted least squares (Lucas and Kanade, 1981): minimize motion gradient constraint over a local area $\Omega$.

$$\int_\Omega w^2(x,y)(v_x I_x + v_y I_y + I_t)^2 \, dx \, dy$$

w(x,y) denotes a window function that gives greater weight to constraints at the center of the neighborhood ($\Omega$), e.g. a $5 \times 5$ Gaussian window. This can be written more compactly in vector notation

$$\sum_{\mathbf{x} \in \Omega} W^2(\mathbf{x})[\nabla I(\mathbf{x}, t) \cdot \mathbf{v} + I_t(\mathbf{x}, t)]^2$$

This has the advantage that it can be expressed as a *least squares problem*

$$A^T W^2 A \mathbf{v} = A^T W^2 \mathbf{b}$$

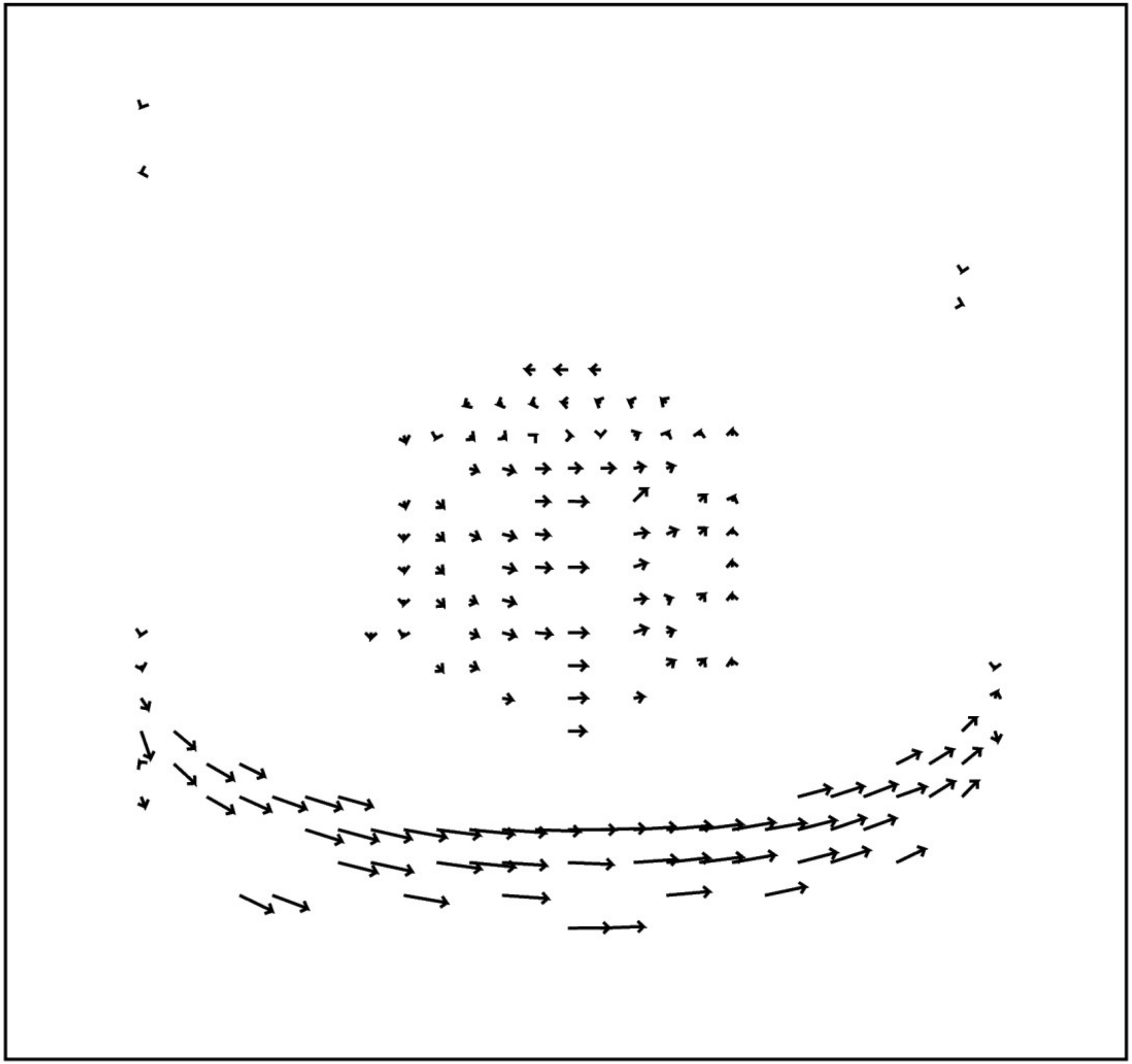
where

$$\begin{aligned} A &= [\nabla I(\mathbf{x}_1), \ldots, \nabla I(\mathbf{x}_n)]^T \\ W &= \text{diag}[W(\mathbf{x}_1), \ldots, W(\mathbf{x}_n)] \\ \mathbf{b} &= -[I_t(\mathbf{x}_1), \ldots, I_t(\mathbf{x}_n)]^T \end{aligned}$$

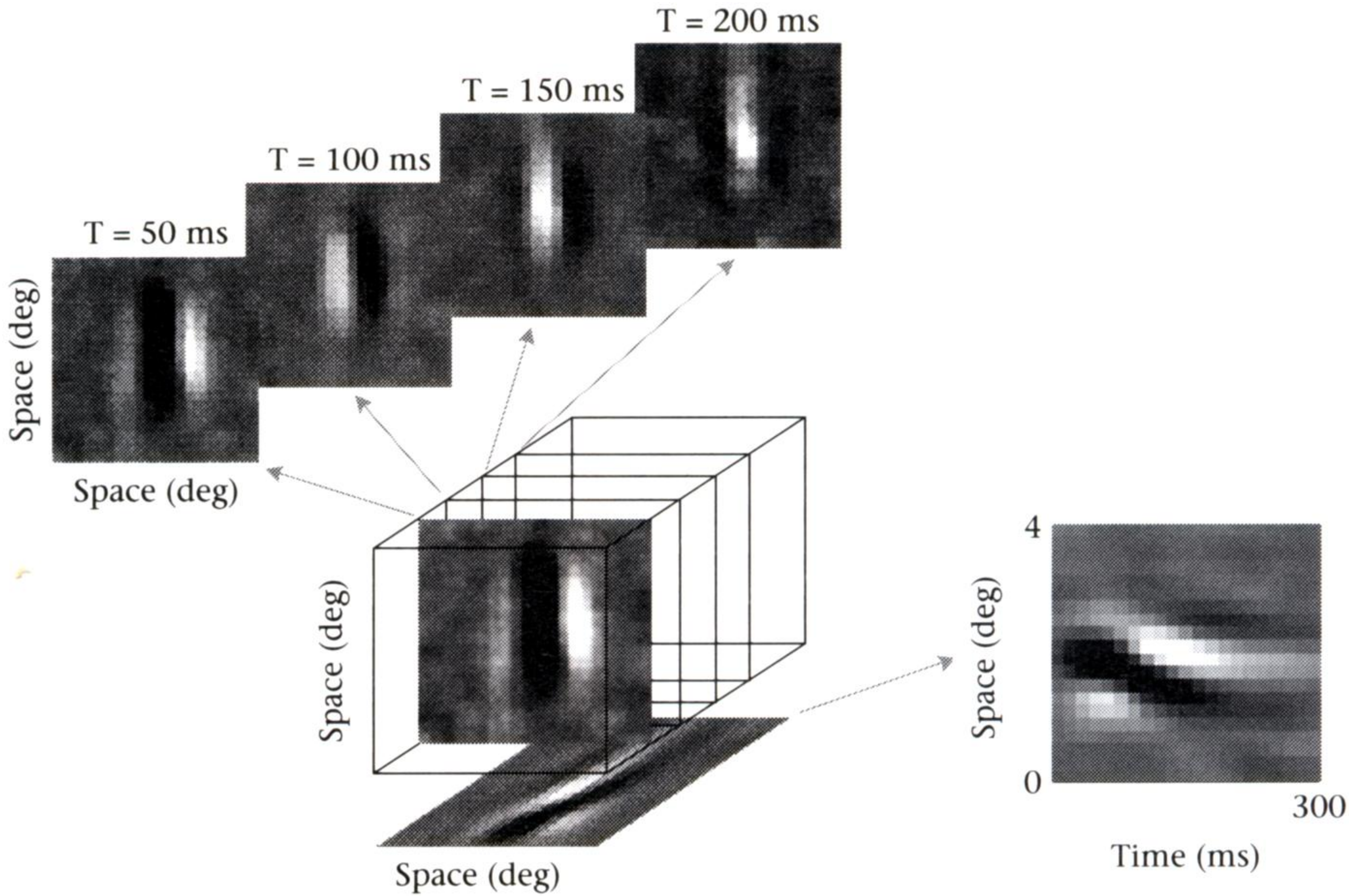
The solution is just a matrix inversion:

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

# Lucas and Kande flow field



# A motion energy algorithm

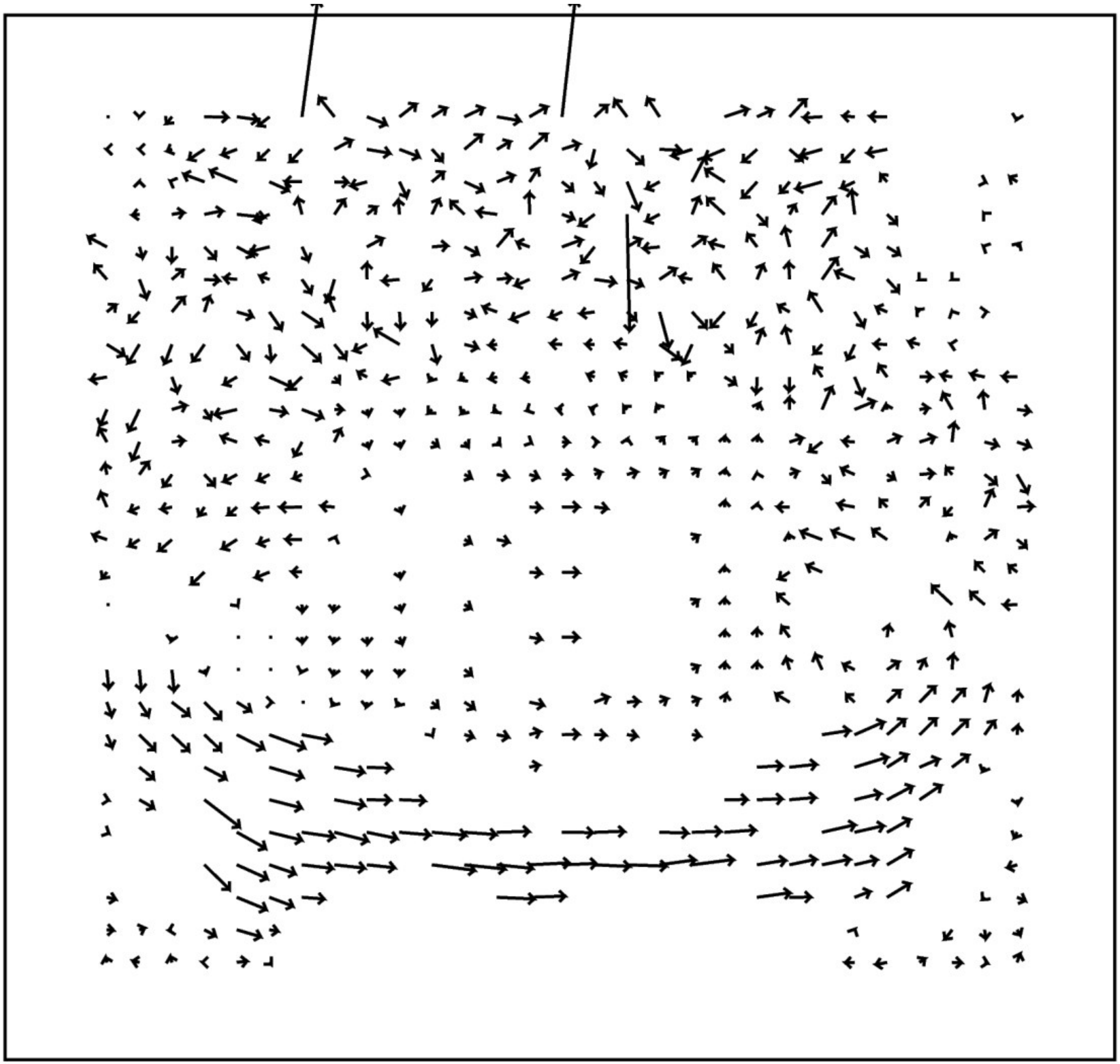


# Sketch of motion energy algorithm

Heeger (1987, 1988)

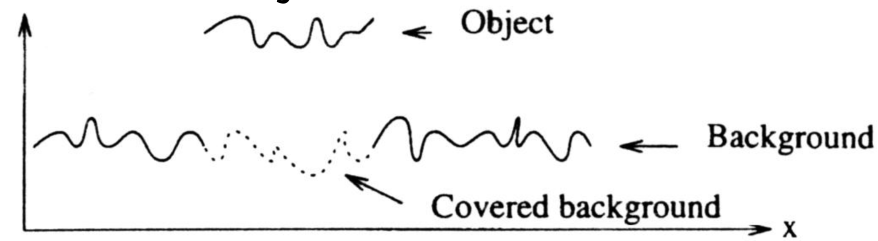
- convolve image with spatio-temporal energy filters
- use 12 Gabor motion energy filters at each of several spatial scales
- also set to different spatial orientations and temporal frequencies
- each level of Gaussian pyramid optimized for different speeds (e.g. 0-1.25, 1.25-2.5, and 2.5-5 pixels/frame)
- estimated local velocity is a least squares between predicted motion energies of Gabor filters and observed motion energy

# Motion energy model flow field solution



# Limitations and difficulties

- real images are noisy
- other sources of intensity changes
- motion isn't smooth at all scales (e.g. textures)
- motion is discontinuous at edges
- effects of object occulsion




- motion is too fast

# Velocity detection in the fly

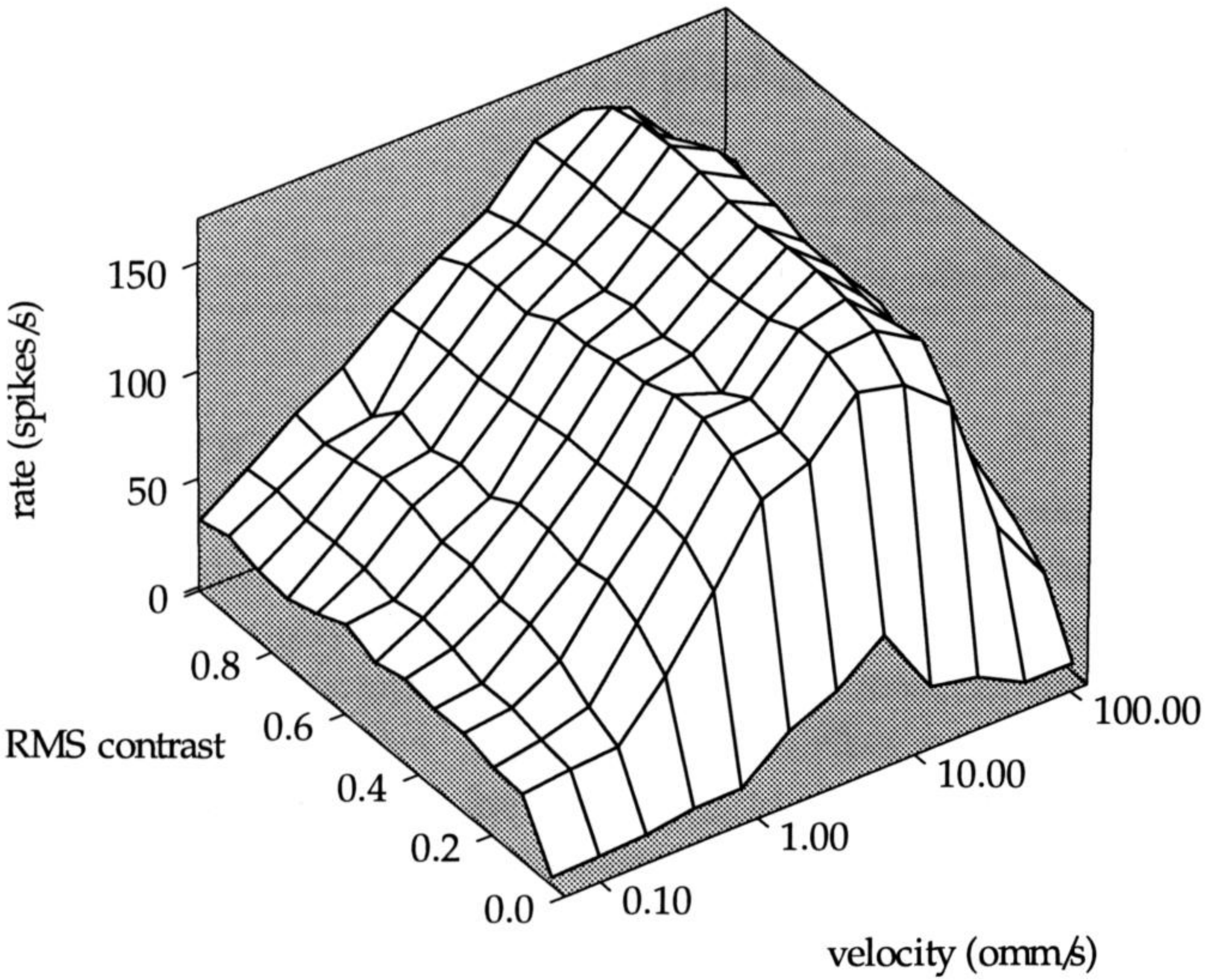
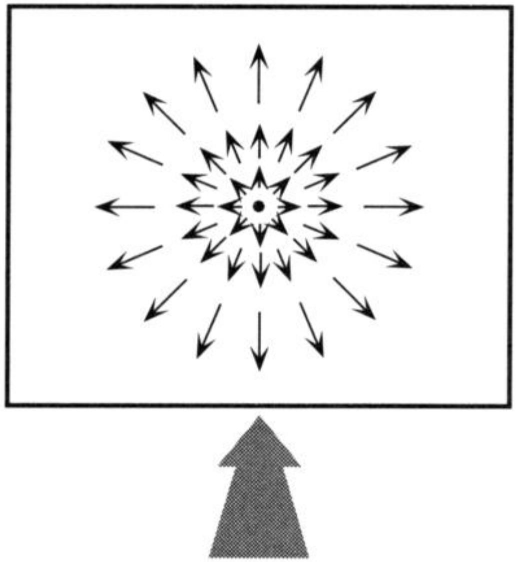
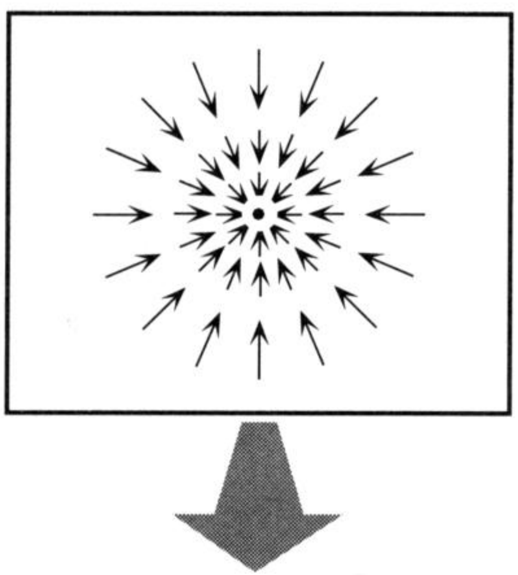



*Fig. 4: Firing rate of H1 as a function of both contrast and velocity. Velocity was presented alternatingly in null direction (3 s, v=8 omm/s, and preferred direction (1 s, v as given by axis). Rate in this figure is the average response to the 1 second test stimulus.*

# Motion fields: the focus of optical expansion



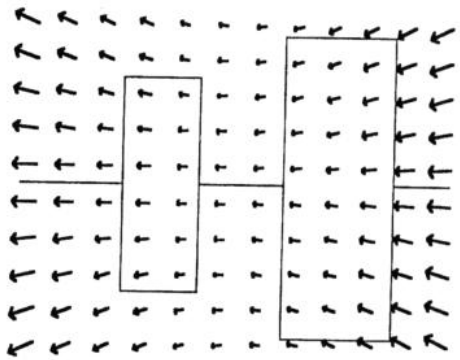
A. Motion Toward



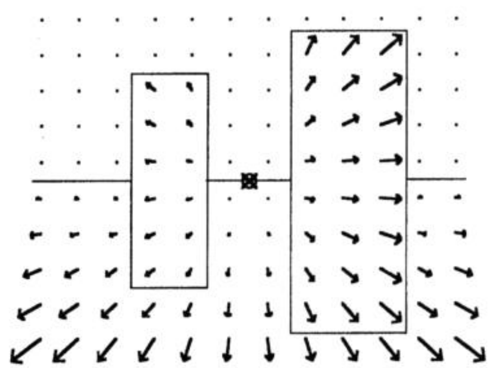
B: Motion Away

# Motion fields: the focus of optical expansion

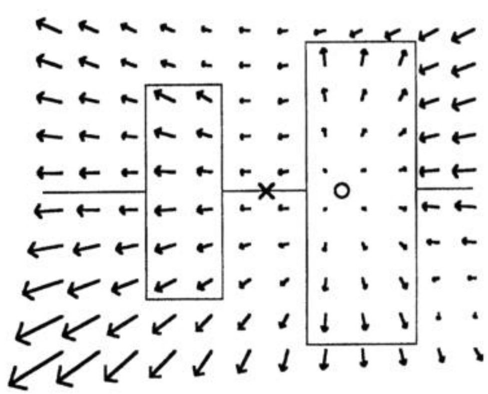
A. Flow field from rightward eye rotation



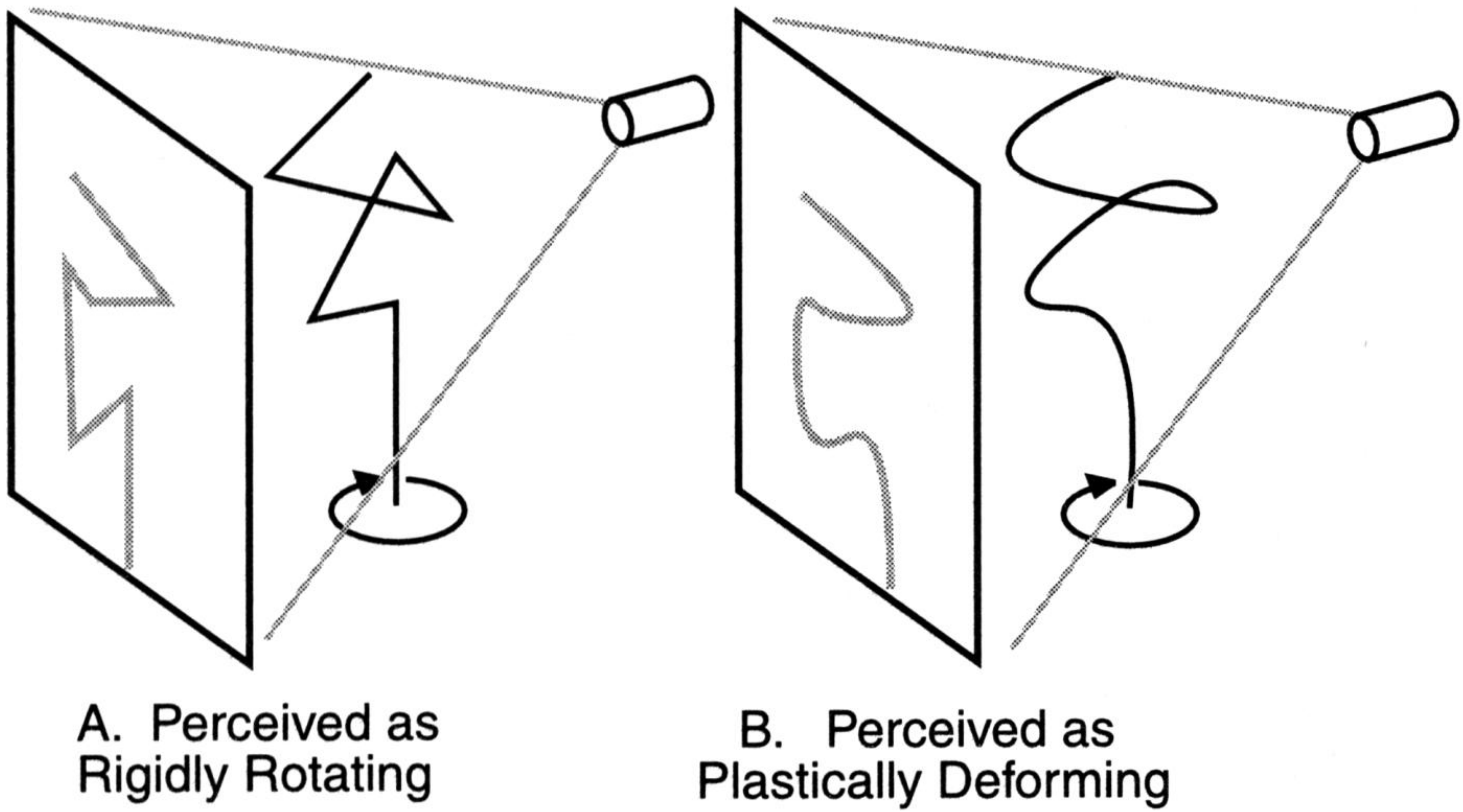
B. Flow field from moving toward X.



C. Flow field from moving toward X while tracking O.



# Higher order motion: The kinetic depth effect

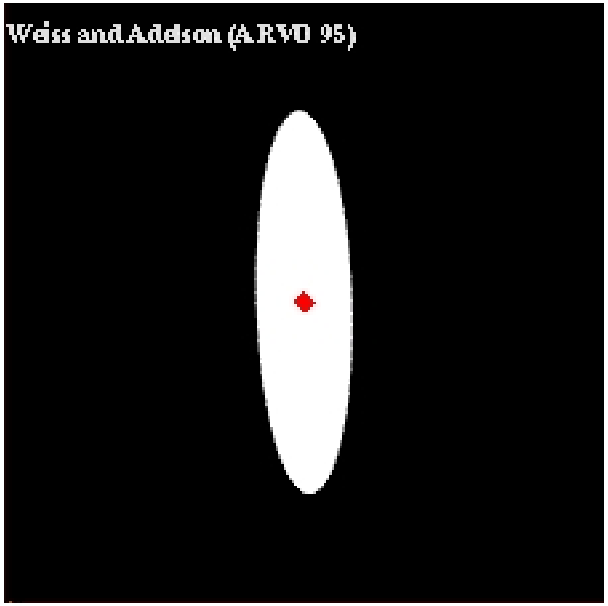


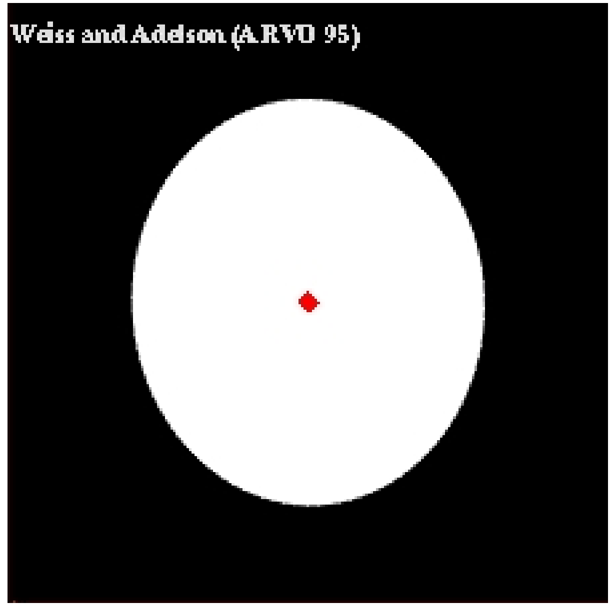

Sphere demo

# More higher level motion demonstrations

- Biological motion
- Stereokinetic depth
- Second-order motion
- Implicit figure motion
- Shadow motion: Rising square
- Shadow motion: Ball in a box

# Motion demonstrations on ellipses

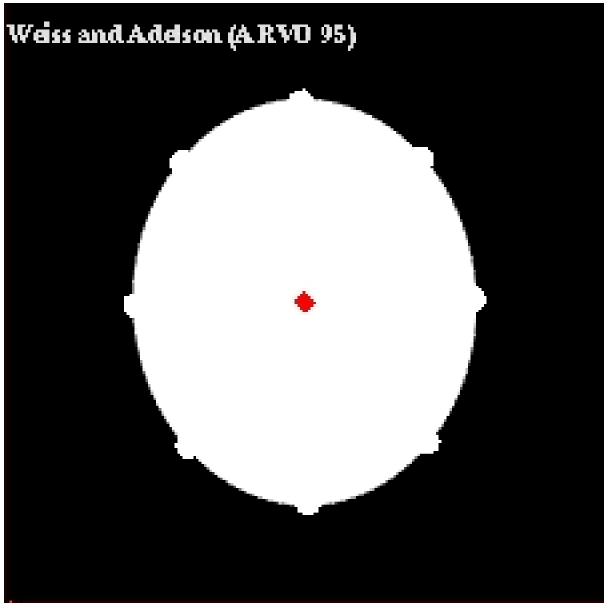


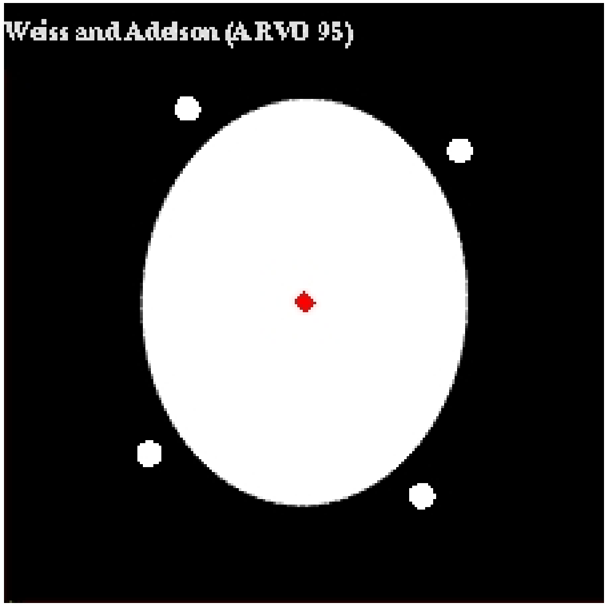




Percept:

- narrow ellipse appears rigid
- fat ellipse is deforms

# Adding texture changes percept






In neither case does the shape of the ellipse change during the animation.

Observation:

> If dots move with ellipse, it now appears rigid.

Why?