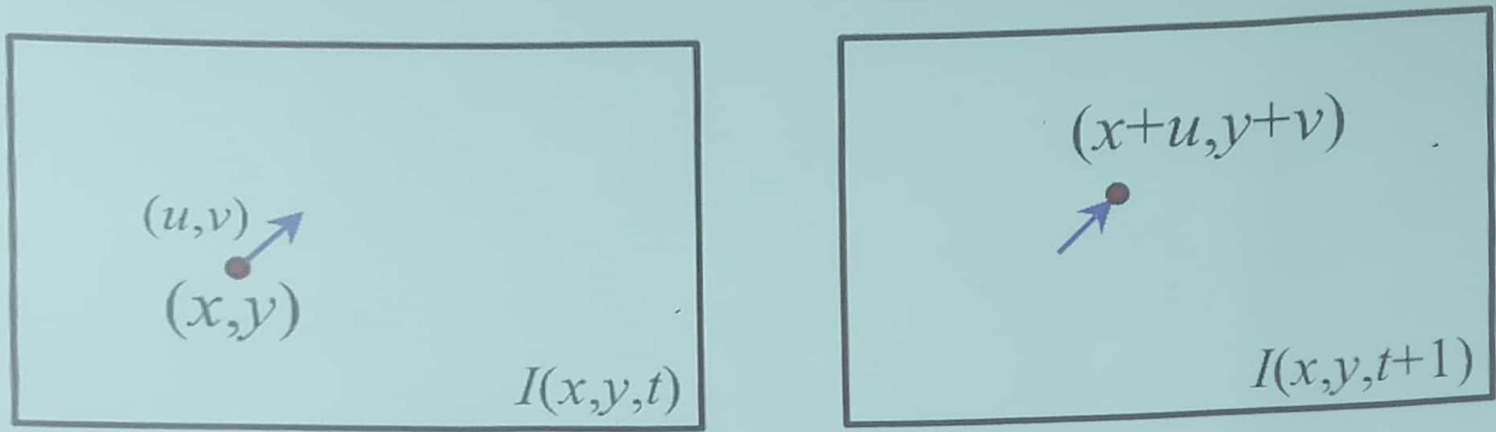
Feature Tracking vs. Optical Flow

Feature Tracking: Extract visual features (corners, textured areas) and "track" them over multiple frames (**sparse corr.**)

Optical Flow: Recover image motion at each pixel from spatio-temporal image brightness variations (**dense corr.**)

- Relationship to stereo matching, SFM

Optical Flow



- Brightness Constancy Assumption

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

Ambiguity of Motion

Can we use this equation to recover image motion (u,v) at each pixel?

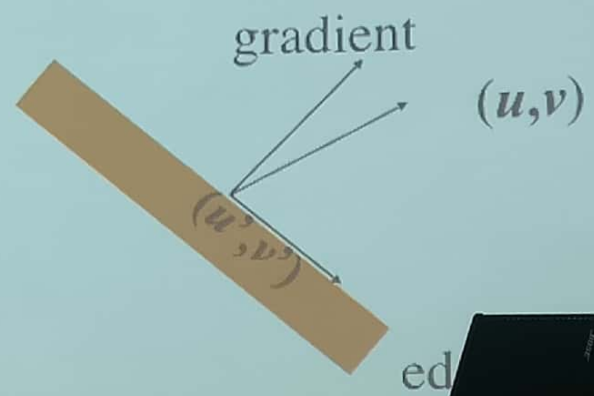
$$\nabla I \cdot [u \ \ v]^T + I_t = 0$$

- How many equations and unknowns per pixel?
  - One equation (this is a scalar equation!), two unknowns (u,v)

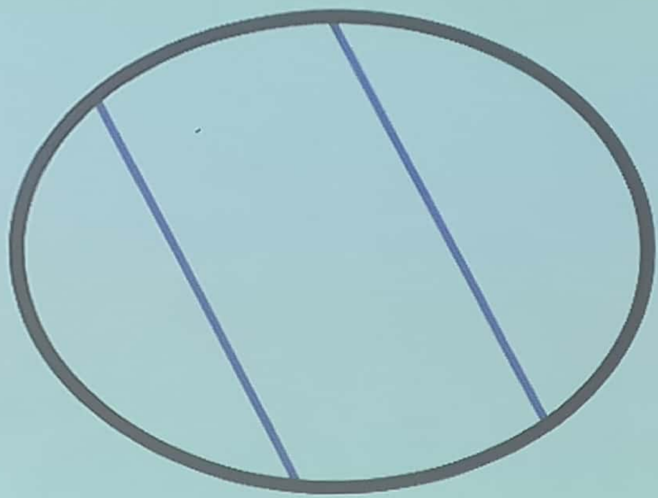
The component of the motion perpendicular to the gradient (i.e., parallel to the edge) cannot be measured

If (u, v) satisfies the equation,
so does $(u+u', v+v')$ if

$$\nabla I \cdot [u' \ \ v']^T = 0$$

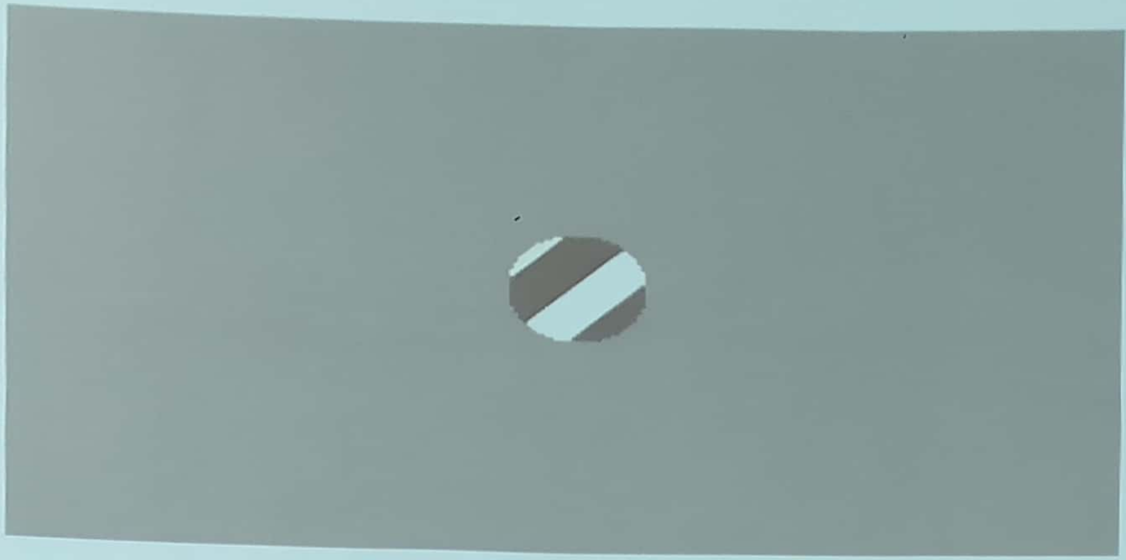


The Aperture Problem



Scanned by CamScanner

Motion Ambiguity



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


Motion Ambiguity
• Motion perpendicular to gradient direction is not discernible.
http://en.wikipedia.org/wiki/Barberpole_illusion
BOSE

Solving the Ambiguity...

B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 674–679, 1981.

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

$$\begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix}$$

Matching Patches Across Images

- Overconstrained linear system

$$\begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix} \qquad \underset{25\times2}{A} \; \underset{2\times1}{d} = \underset{25\times1}{b}$$

Least squares solution for d given by $(A^T A)\ d = A^T b$

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

The summations are over all pixels in the K x K window

Conditions for Solvability

Optimal (u, v) satisfies Lucas-Kanade equation

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

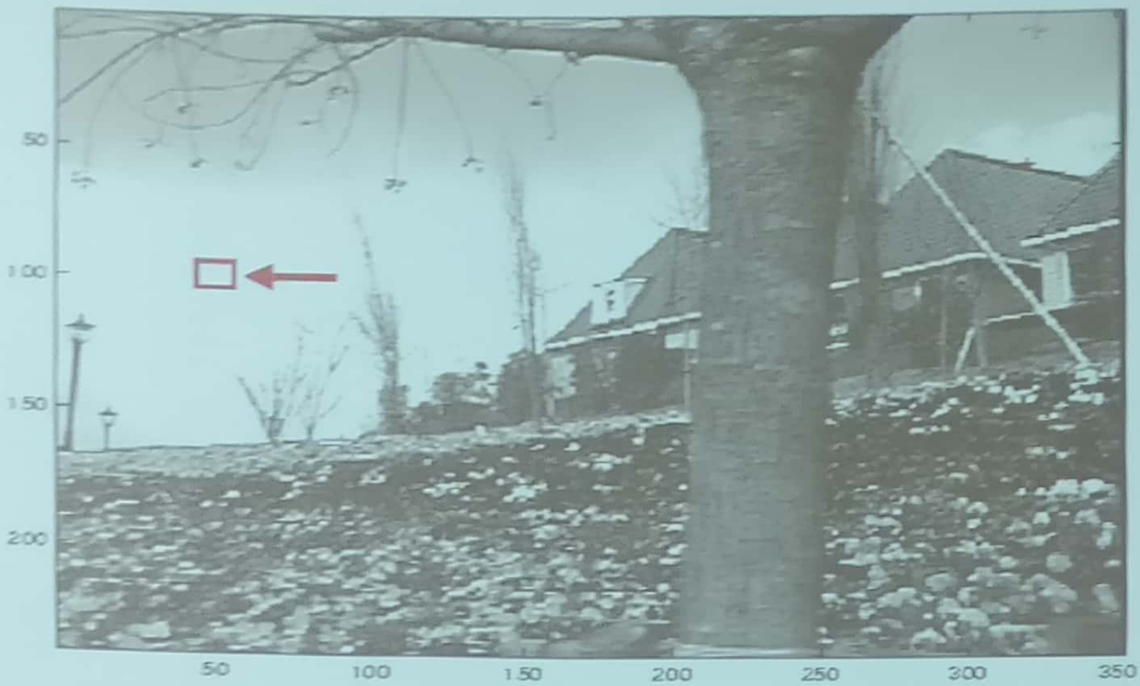
$$A^T A \qquad\qquad\qquad A^T b$$

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?

Low-Texture Region



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

– gradients have small magnitude

– small λ_1, small λ_2

High-Texture Region

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

– gradients are different, large magnitudes

– large λ_1, large λ_2

Dealing with Larger Movements: Iterative Refinement

Original (x,y) position

1. Initialize (x',y') = (x,y)
2. Compute (u,v) by

$I_t = I(x', y', t+1) - I(x, y, t)$

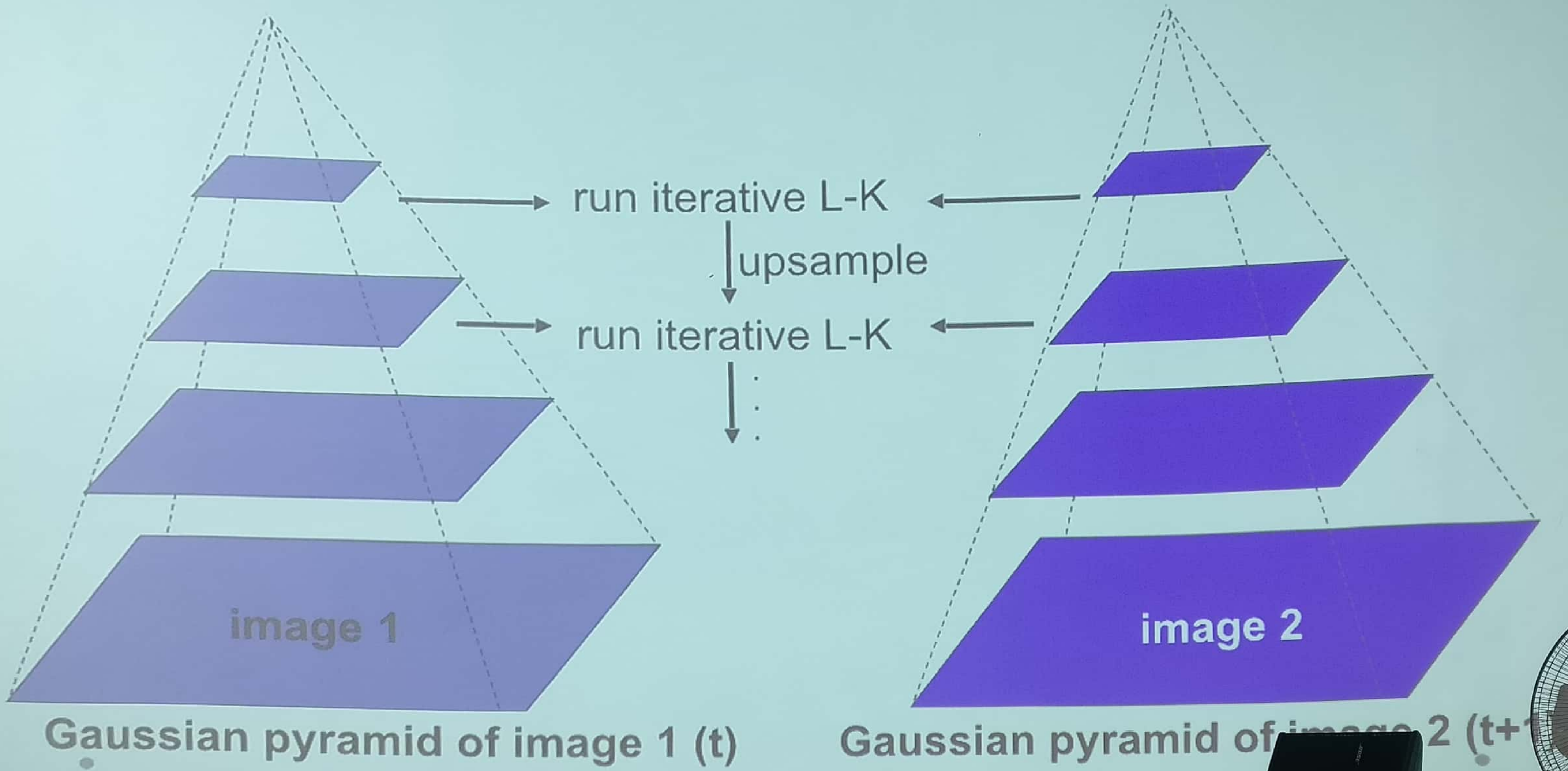
$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

2nd moment matrix for feature patch in first image

displacement

1. Shift window by (u, v): $x' = x' + u;\ y' = y' + v;$
2. Recalculate I_t
3. Repeat steps 2-4 until small change
   - Use interpolation for subpixel values

Dealing with Larger Movements: Coarse-to-Fine Registration



Shi-Tomasi Feature Tracker

- Find good features using eigenvalues of 2nd-moment matrix (e.g., Harris detector or threshold on the smallest eigenvalue)
  - Key idea: “good” features to track are the ones whose motion can be estimated reliably
- Track from frame to frame with Lucas-Kanade
  - This amounts to assuming a translation model for frame-to-frame feature movement
- Check consistency of tracks by *affine* registration to the first observed instance of the feature
  - Affine model is more accurate for larger displacements
  - Comparing to the first frame helps to minimize drift

J. Shi and C. Tomasi. Good Features to Track. CVPR 1994

Tracking Example

Figure 1: Three frame details from Woody Allen's *Manhattan*. The details are from the 1st, 11th, and 21st frames of a subsequence from the movie.

Figure 2: The traffic sign windows from frames 1,6,11,16,21 as tracked (top), and warped by the computed deformation matrices (bottom).

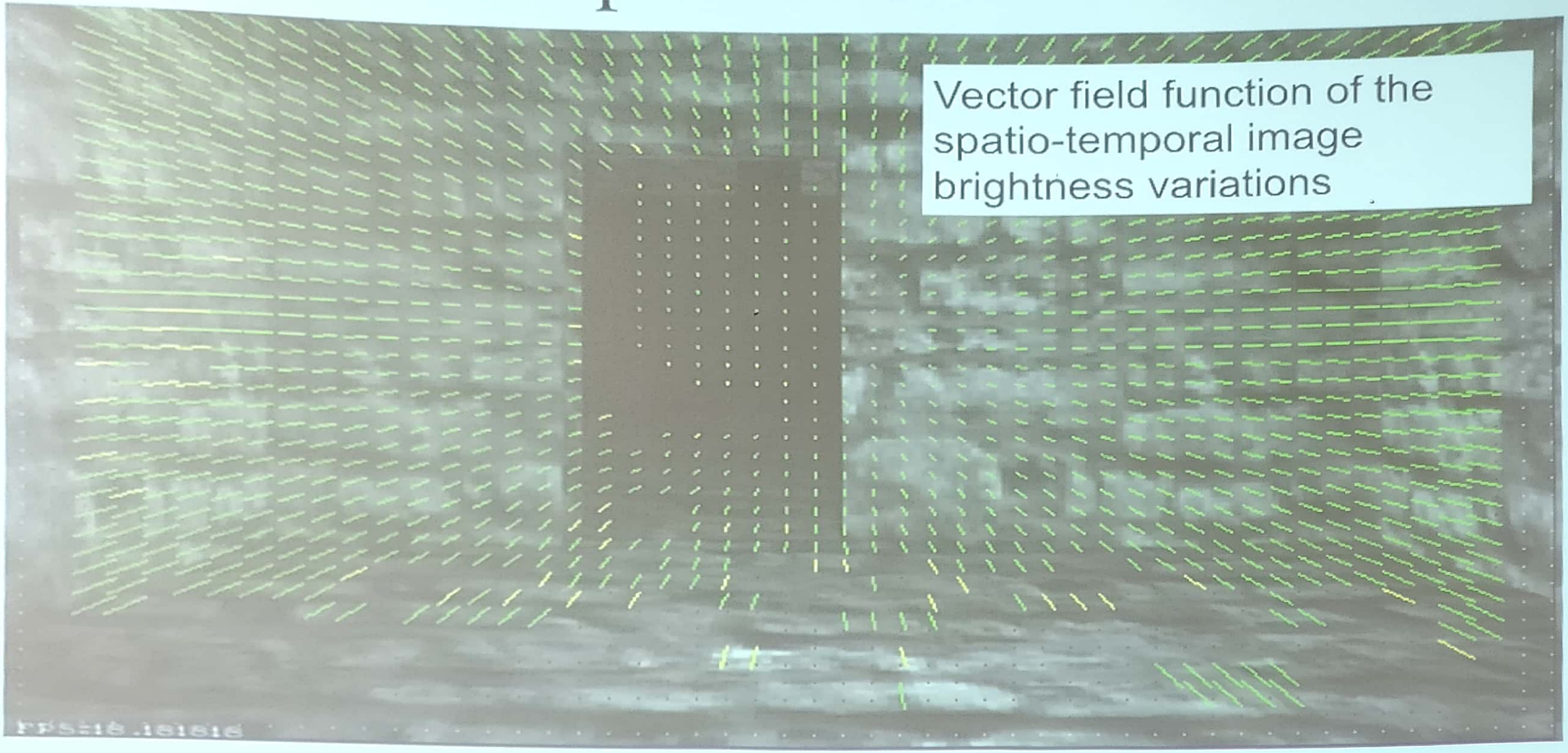
J. Shi and C. Tomasi. Good Features to Track. CVPR 1994

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)

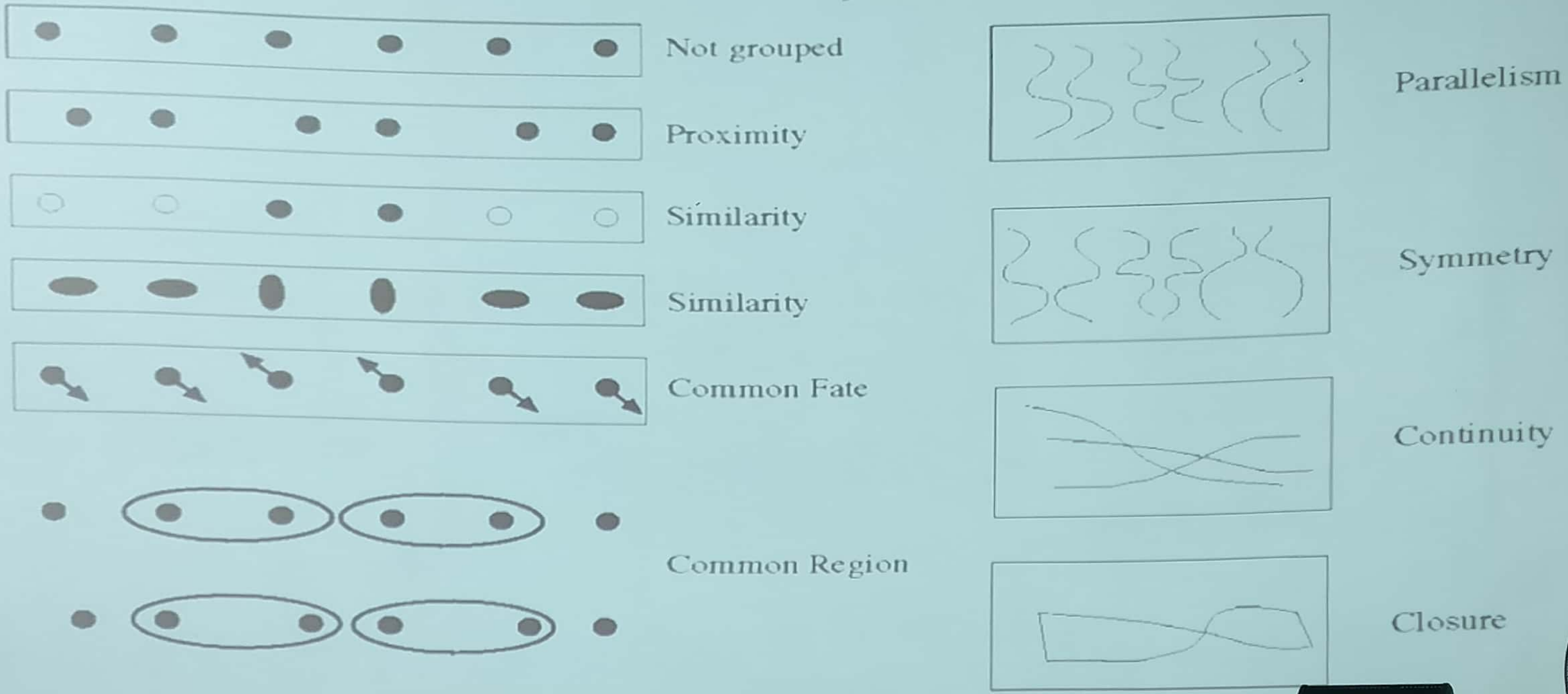
Optical flow



Picture courtesy of Selim Temizer - Learning and Intelligent Systems (LIS) Group, MIT

Motion and Perceptual Organization

- Sometimes, motion is the only cue

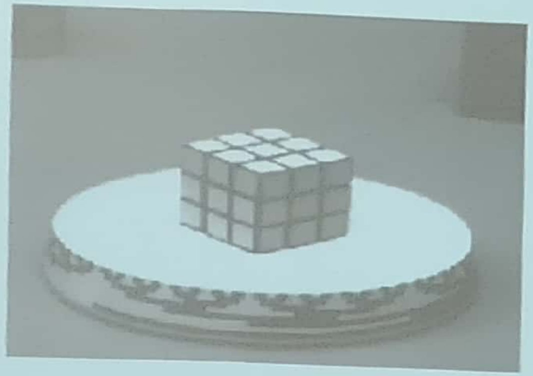


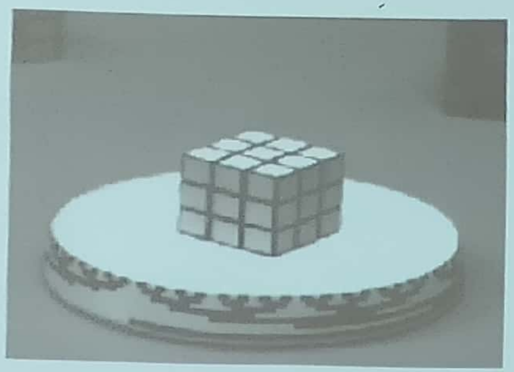
Uses of Motion Estimation

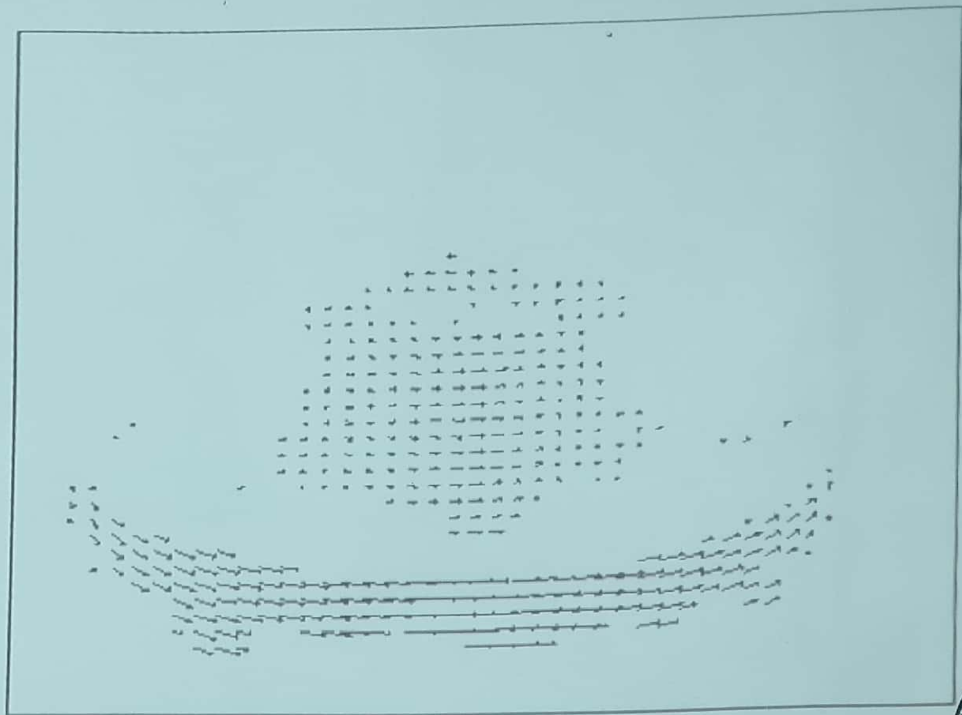
- Estimating 3D structure
- Segmenting objects based on motion cues
- Learning and tracking dynamical models
- Recognizing events and activities
- Improving video quality (motion stabilization)
- Video Compression (MPEG-4)

Motion Field

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







What would the motion field of a non-rotating ball moving towards the c ok like

Lucas-Kanade Optical Flow

- Same as Lucas-Kanade feature tracking, but for each pixel
  - As we saw, works better for textured pixels
- Operations can be done one frame at a time, rather than pixel by pixel
  - Efficient

Iterative Refinement

- Iterative Lukas-Kanade Algorithm
  1. Estimate displacement at each pixel by solving Lucas-Kanade equations
  2. Warp I(t) towards I(t+1) using the estimated flow field
     - Basically, just interpolation
  3. Repeat until convergence

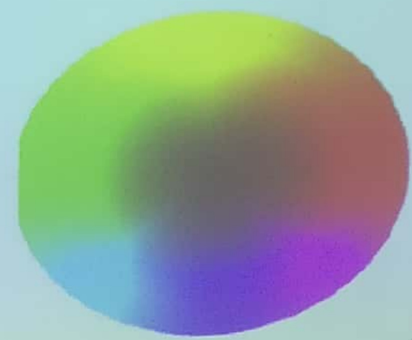
* From Khurram Hassan-Shafique CA… Vision 2003

Other methods for optical flow

Start with something similar to Lucas-Kanade

+ gradient constancy
+ energy minimization with smoothing term
+ region matching
+ keypoint matching (long-range)

Region-based +Pixel-based +Keypoint-based



Color map used to visualize flo

Summary

- Major contributions from Kanade Lucas, Tomasi
  - Tracking feature points
  - Optical flow

- Key ideas
  - By assuming brightness constancy, truncated Taylor expansion leads to simple and fast patch matching across frames
  - Coarse-to-fine registration