C. Feature Tracking and Optical Flow

Instructor: Shaifali Parashar

Many Slides from Fei-Fei Li’s lectures at Standford

Feature Tracking & Optical Flow

Feature tracking: Extract visual features (corners, textured areas) and “track” them over multiple frames

Optical Flow: Recover image motion at each pixel from spatio-temporal image brightness variations

Optical flow = apparent motion of brightness patterns

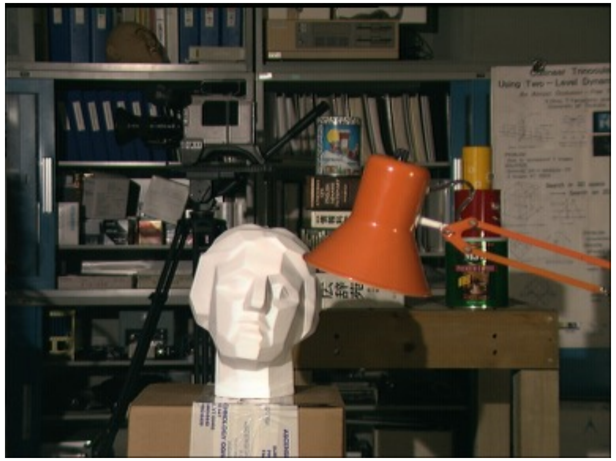
Optical Flow

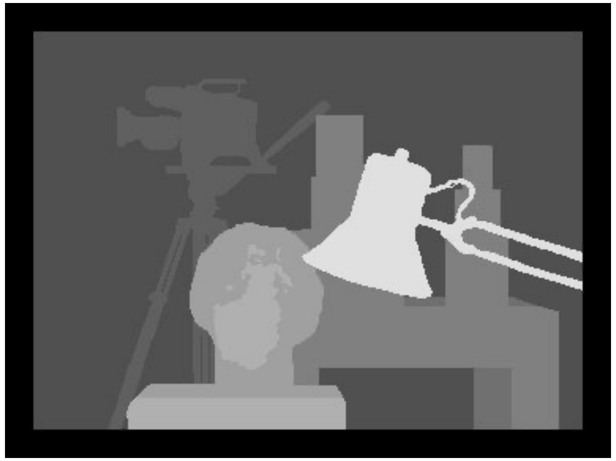
Why bother with motion?

G. Johansson, “Visual Perception of Biological Motion and a Model For Its Analysis", *Perception and Psychophysics 14, 201-211, 1973.*

Image processing goals

- Computing Field Properties
  - orientation
  - optical flow
  - …





Feature Tracking & Optical Flow

Challenges:

1. Figure out which features can be tracked
2. Efficiently track across frames
3. Some points may change appearance over time (e.g., due to rotation, moving into shadows, etc.)
4. Drift: small errors can accumulate as appearance model is updated
5. Points may appear or disappear: need to be able to add/delete tracked points

Feature Tracking & Optical Flow

Feature tracking: Extract visual features (corners, textured areas) and "track" them over multiple frames

Problems: Sparse, inexact feature alignment, low accuracy

Optical Flow: Recover image motion at each pixel from spatio-temporal image brightness variations

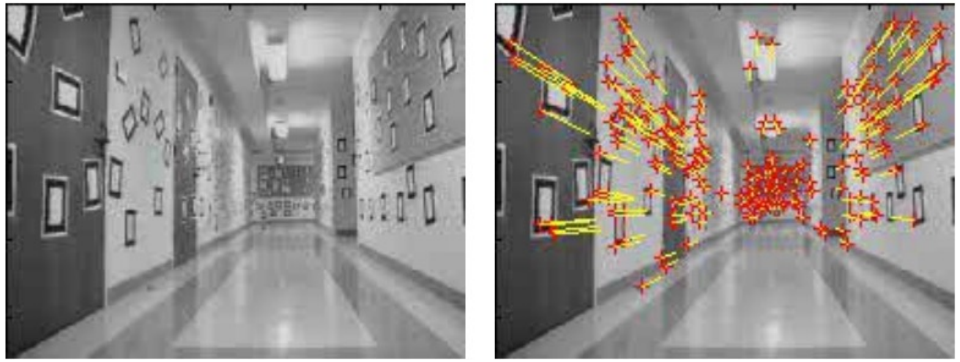
Advantage: Scale/rotation invariant, lighting invariant, can handle large movements (not really, not accurately for sure)

Registration using Lucas-Kanade Method

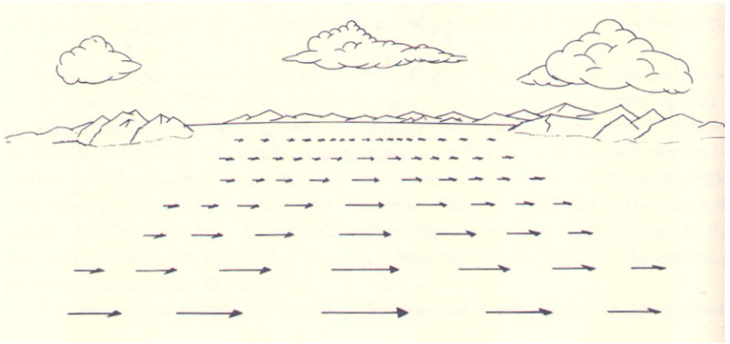
An iterative image registration technique with an application to stereo vision

Feature Tracking & Optical Flow

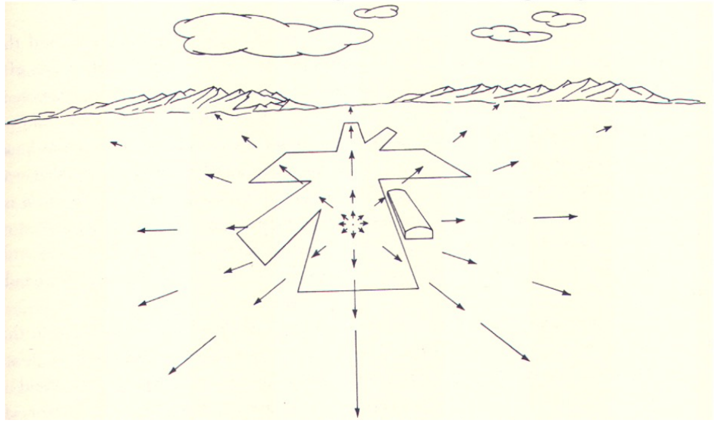
Feature-tracking



Optical flow from the side window of a car

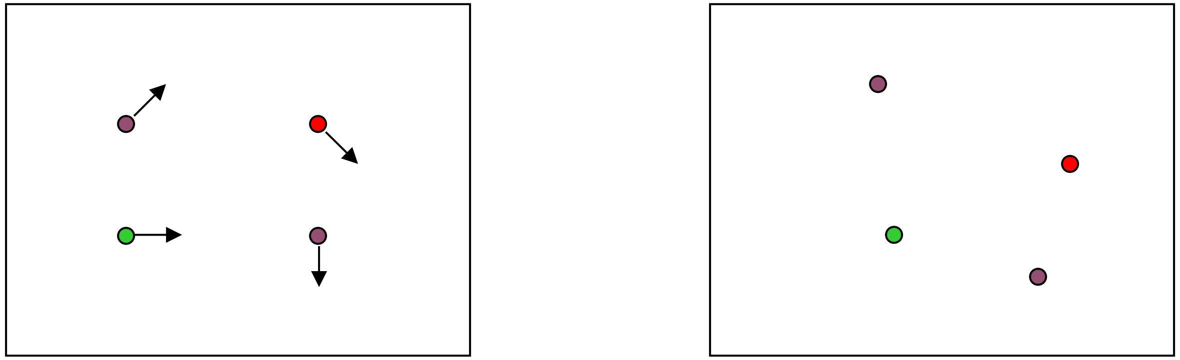


Optical flow for a pilot landing a plane



Applications:
3D reconstruction, Object segmentation, Motion Segmentation, Tracking, Super-resolution, Activity recognition

Optical Flow



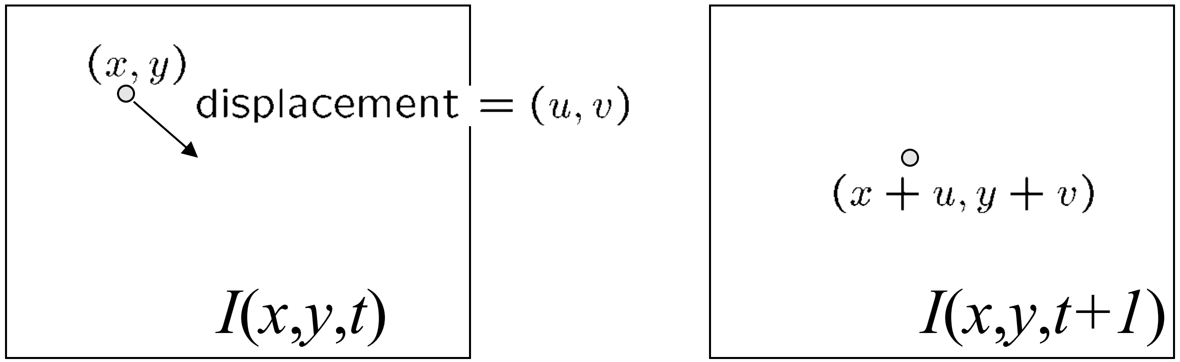
$I(x,y,t)$ $I(x,y,t+1)$

Given two subsequent frames, estimate the point translation

Key assumptions of Lucas-Kanade Tracker

- **Brightness constancy:** projection of the same point looks the same in every frame
- **Small motion:** points do not move very far
- **Spatial coherence:** points move like their neighbors

Optical Flow: Brightness Constancy



Brightness Constancy Equation: $I(x, y, t) = I(x+u, y+v, t+1)$

Take Taylor expansion of *I(x+u, y+v, t+1)* at *(x,y,t)* to linearize the right side:

Image derivative along x   Difference over frames

$$I(x+u, y+v, t+1) \approx I(x, y, t) + \boxed{I_x} \cdot u + I_y \cdot v + \boxed{I_t}$$

$$I(x+u, y+v, t+1) - I(x, y, t) = I_x \cdot u + I_y \cdot v + I_t$$

So: $I_x \cdot u + I_y \cdot v + I_t \approx 0 \quad \rightarrow \nabla \mathrm{I} \cdot [\mathrm{u} \;\; \mathrm{v}]^{\mathrm{T}} + \mathrm{I}_{\mathrm{t}} = 0$

Optical Flow: Brightness Constancy

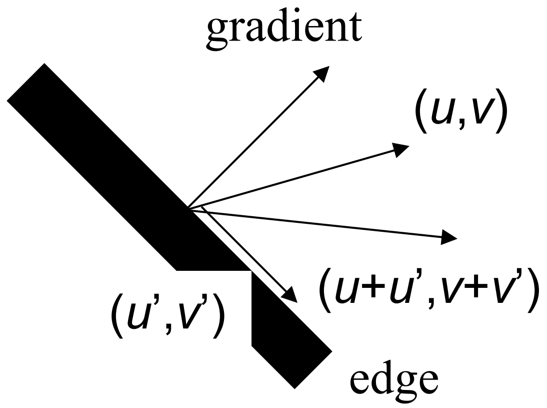
$$\nabla I \cdot [u \;\; v]^T + I_t = 0$$

1 equation, 2 unknowns (u,v), can't solve

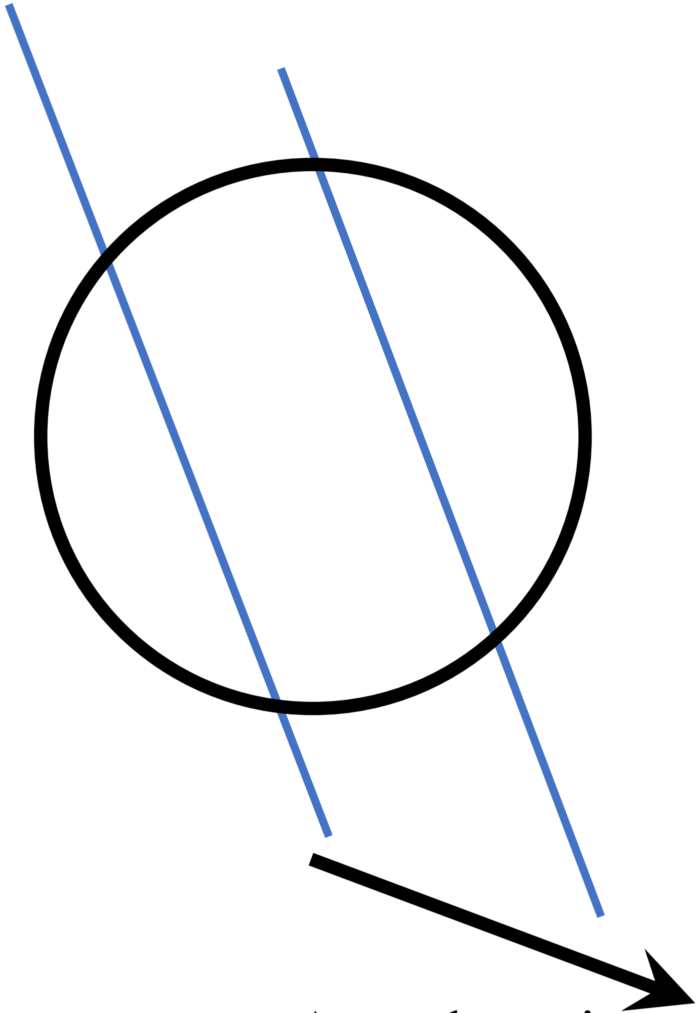
The component of the motion perpendicular to the gradient (i.e., parallel to the edge) cannot be measured

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

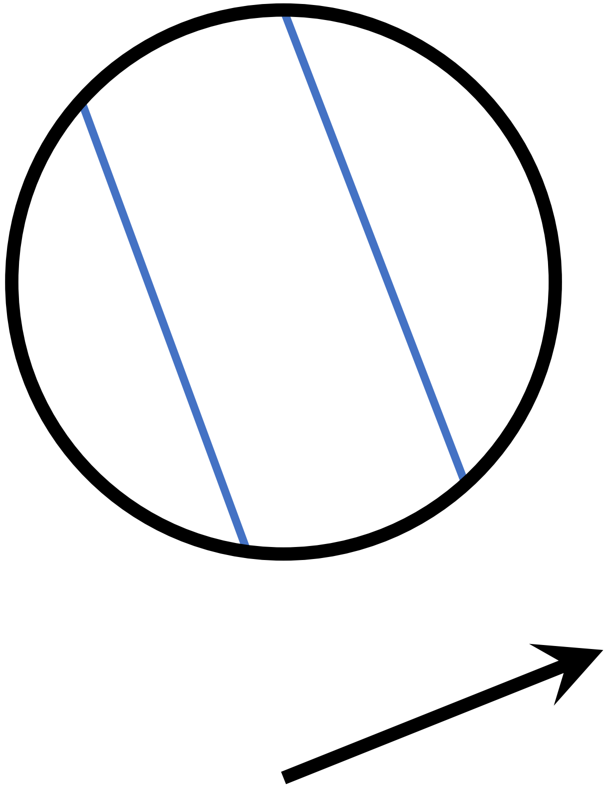
$$\nabla I \cdot [u' \;\; v']^T = 0$$



Optical Flow: Aperture Problem

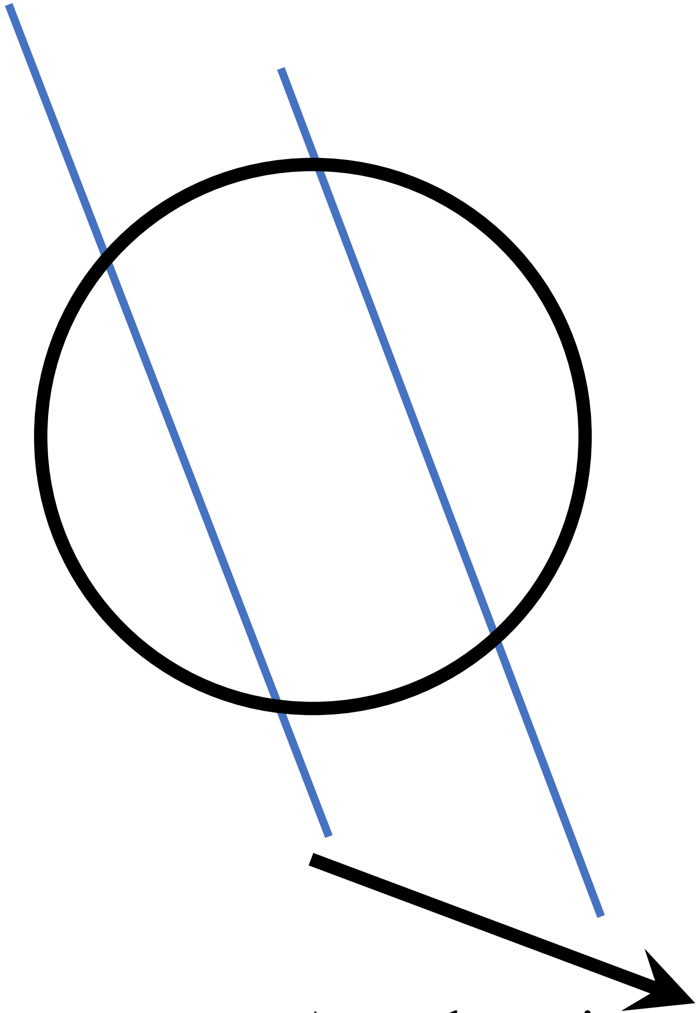


Actual motion

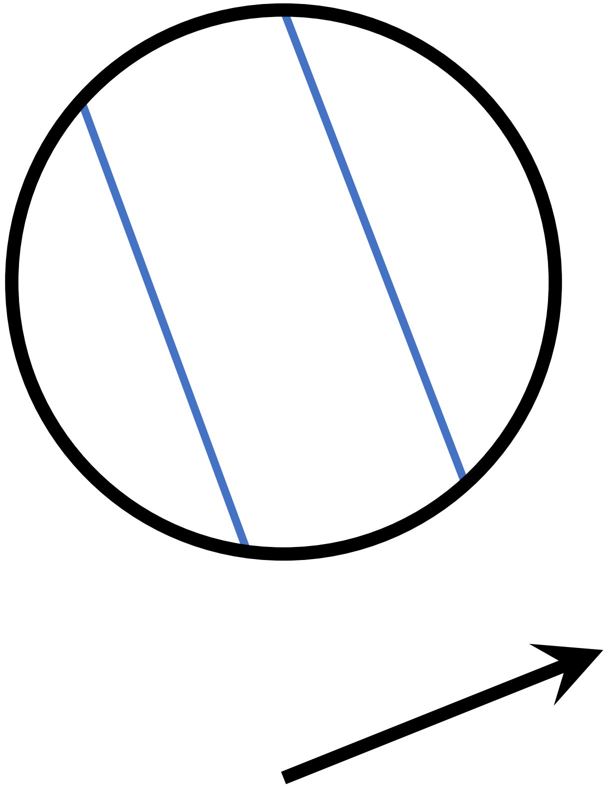


Perceived motion

Optical Flow: Aperture Problem



Actual motion



Perceived motion

Solution:
Look at neighbors

Optical Flow: Barber Pole Illusion

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

Optical Flow: Barber Pole Illusion

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

Optical Flow: Solving the issue

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

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

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

$$(A^T A) \; d = A^T b$$

Optical Flow: Solving the issue

When is this solvable? I.e., what are good points to track? Remember: Harris Corners

- $\mathbf{A^TA}$ should be invertible
- $\mathbf{A^TA}$ should not be too small due to noise
  - eigenvalues λ_1 and λ_2 of $\mathbf{A^TA}$ should not be too small
- $\mathbf{A^TA}$ should be well-conditioned
  - λ_1/λ_2 should not be too large (λ_1 = larger eigenvalue)

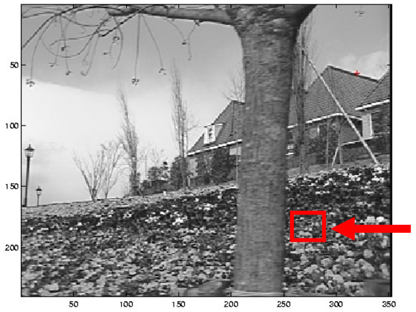
Low-texture region

– gradients are small magnitude
– small eigenvalues

Edge

– gradients are large and small
– large and small eigenvalue

High-texture region



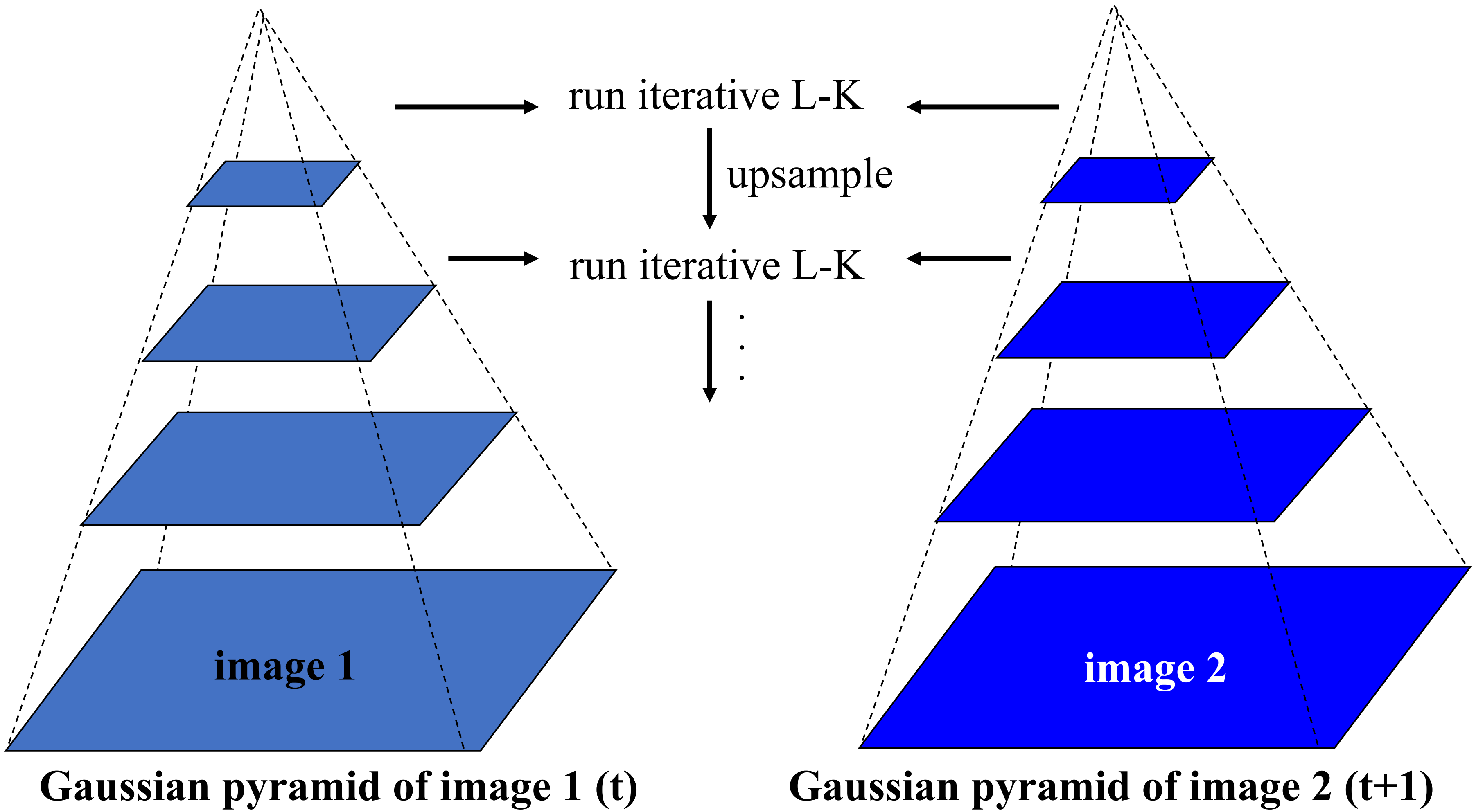
– gradients are different, large magnitudes
– large eigenvalues

Optical Flow- Lucas-Kanade

Assumptions:

1. Brightness consistency
2. Spatial coherence
3. Small motion (not really!)

Dealing with larger movements: coarse-to-fine registration



Optical Flow: State of the Art

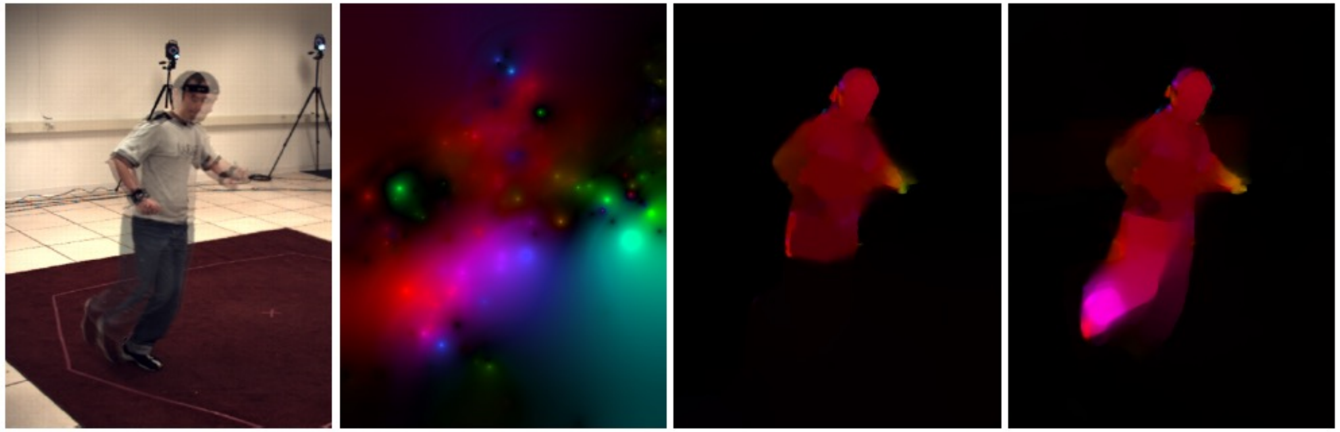
Start with something similar to Lucas-Kanade

+ gradient constancy

+ energy minimization with smoothing term

+ region matching

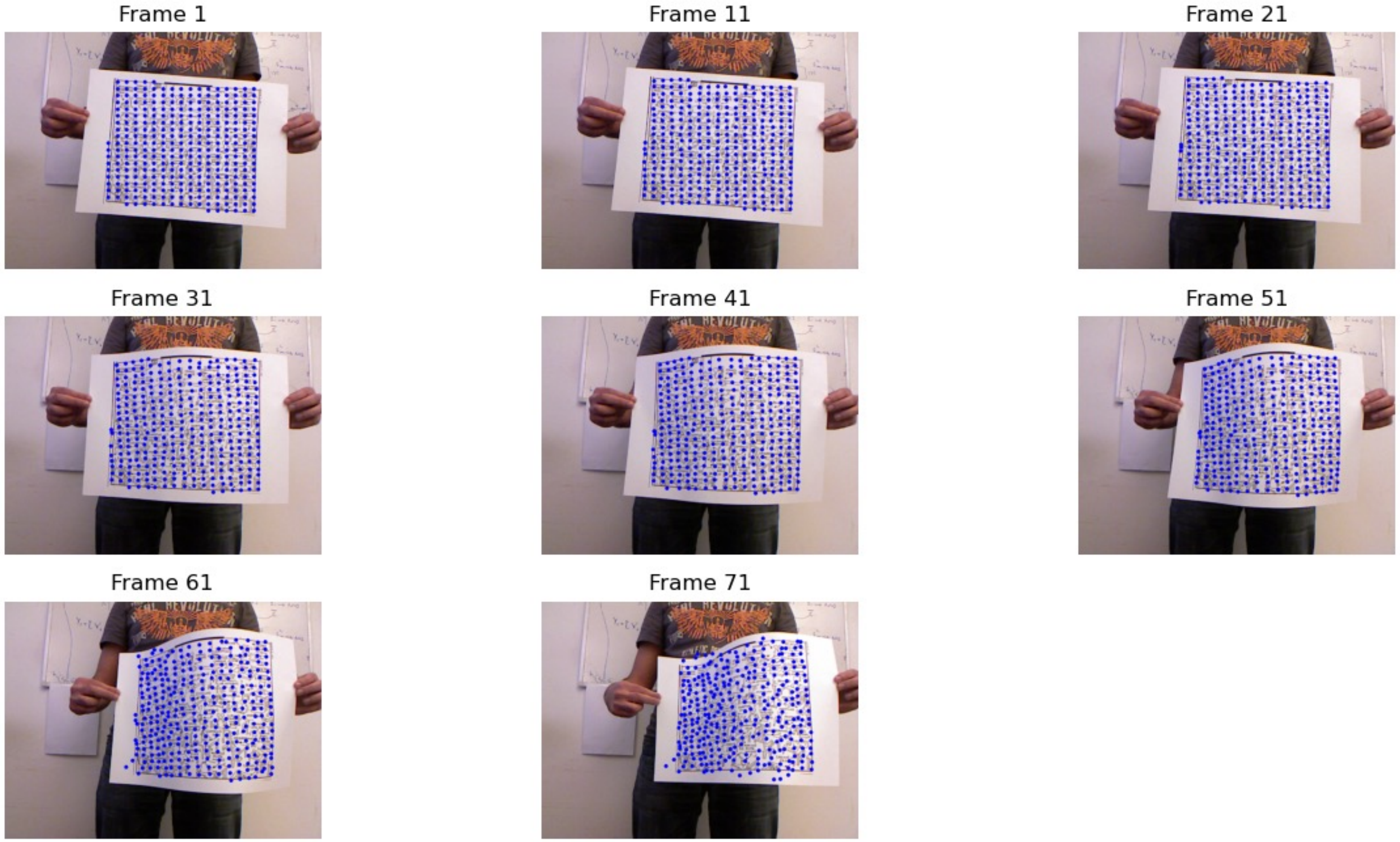
+ keypoint matching (long-range)



Region-based +Pixel-based +Keypoint-based

Large displacement optical flow, Brox et al., CVPR 2009

Optical Flow: State of the Art



PIPS: ECCV-2022

Feature Tracking: Shi-Tomasi Feature Tracker

Find good features using eigenvalues of second-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.

Feature Tracking: Shi-Tomasi Feature Tracker

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.

Feature Tracking: Shi-Tomasi Feature Tracker

1. Find a good point to track (harris corner)
2. Use intensity second moment matrix and difference across frames to find displacement
3. Iterate and use coarse-to-fine search to deal with larger movements
4. When creating long tracks, check appearance of registered patch against appearance of initial patch to find points that have drifted

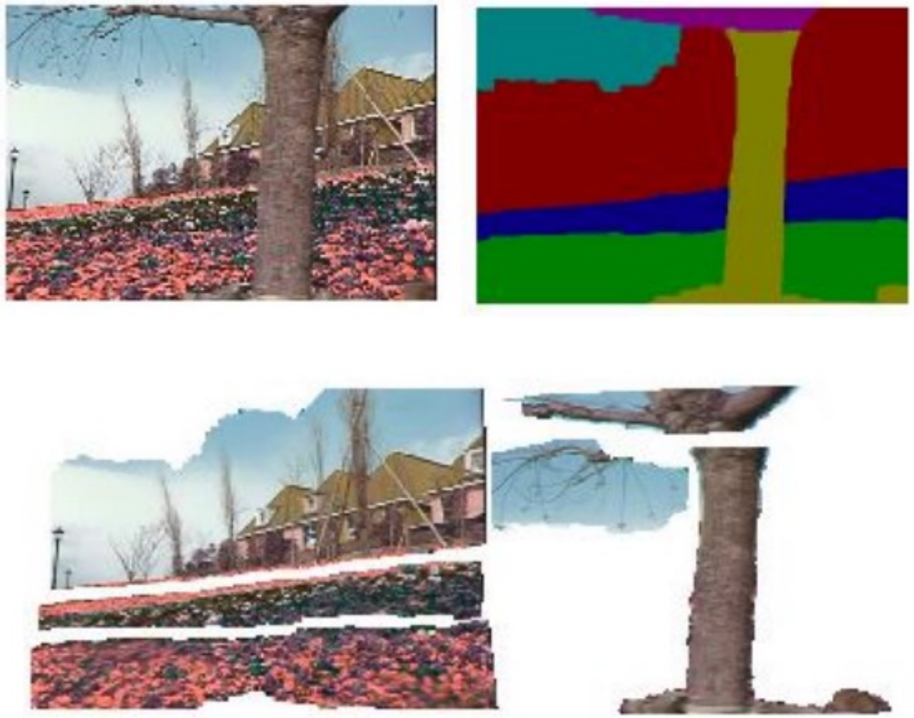
Implementation issues:

- 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)

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

Motion Segmentation

- Create layers (with coherent affine motion) and track them



J. Wang and E. Adelson. Layered Representation for Motion Analysis. *CVPR 1993*.

Affine Motion

$$u(x,y) = a_1 + a_2 x + a_3 y$$
$$v(x,y) = a_4 + a_5 x + a_6 y$$

- Substituting into the brightness constancy equation:

$$\boxed{I_x(a_1 + a_2 x + a_3 y) + I_y(a_4 + a_5 x + a_6 y) + I_t \approx 0}$$

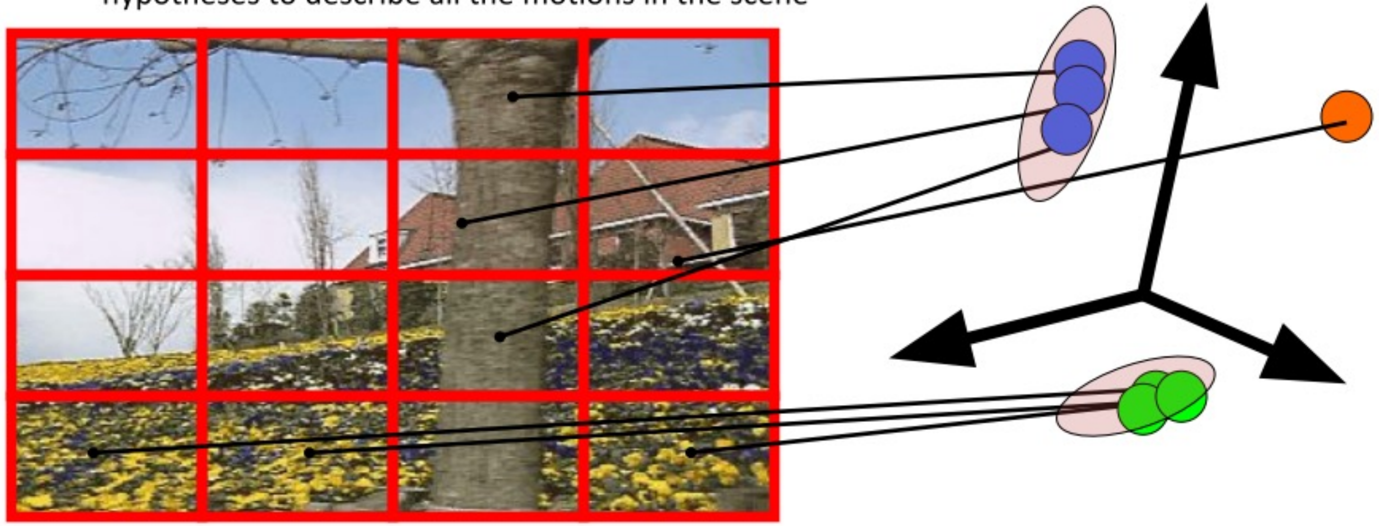
- Each pixel provides 1 linear constraint in 6 unknowns
- Least squares minimization:

$$\boxed{Err(\vec{a}) = \sum \left[I_x(a_1 + a_2 x + a_3 y) + I_y(a_4 + a_5 x + a_6 y) + I_t \right]^2}$$

Source: Silvio Savarese

How to estimate layers?

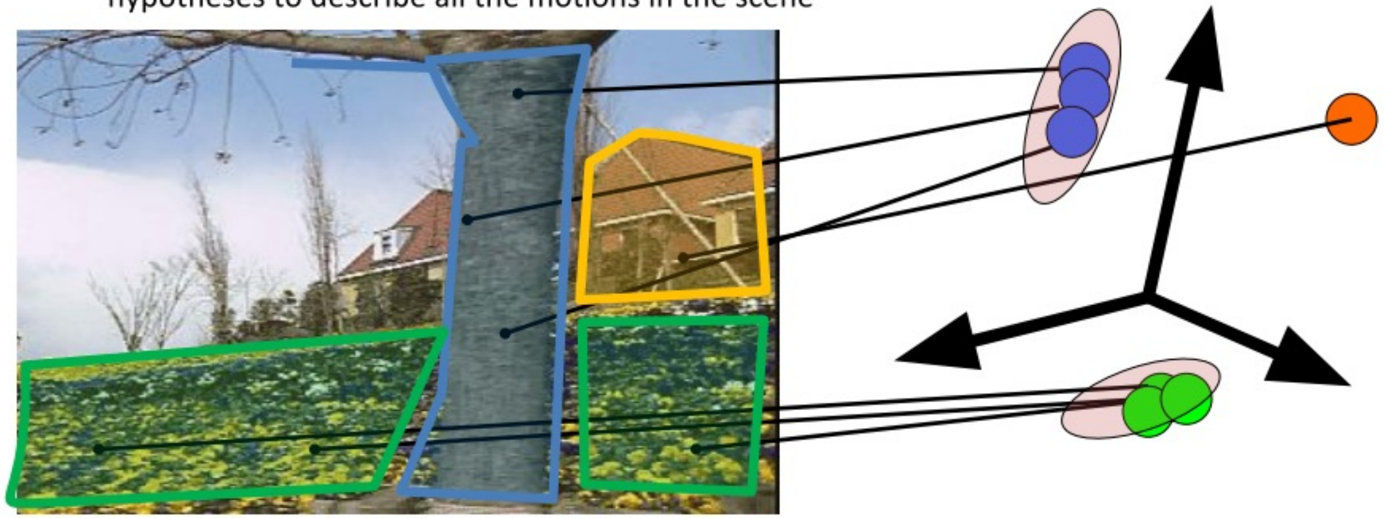
1. Obtain a set of initial affine motion hypotheses
   - Divide the image into blocks and estimate affine motion parameters in each block by least squares
   - Eliminate hypotheses with high residual error
2. Map into motion parameter space
3. Perform k-means clustering on affine motion parameters
   - Merge clusters that are close and retain the largest clusters to obtain a smaller set of hypotheses to describe all the motions in the scene



Source: Silvio Savarese

How to estimate layers?

1. Obtain a set of initial affine motion hypotheses
   - Divide the image into blocks and estimate affine motion parameters in each block by least squares
   - Eliminate hypotheses with high residual error
2. Map into motion parameter space
3. Perform k-means clustering on affine motion parameters
   - Merge clusters that are close and retain the largest clusters to obtain a smaller set of hypotheses to describe all the motions in the scene



Source: Silvio Savarese

How to estimate layers?

1. Obtain a set of initial affine motion hypotheses
   - Divide the image into blocks and estimate affine motion parameters in each block by least squares
   - Eliminate hypotheses with high residual error

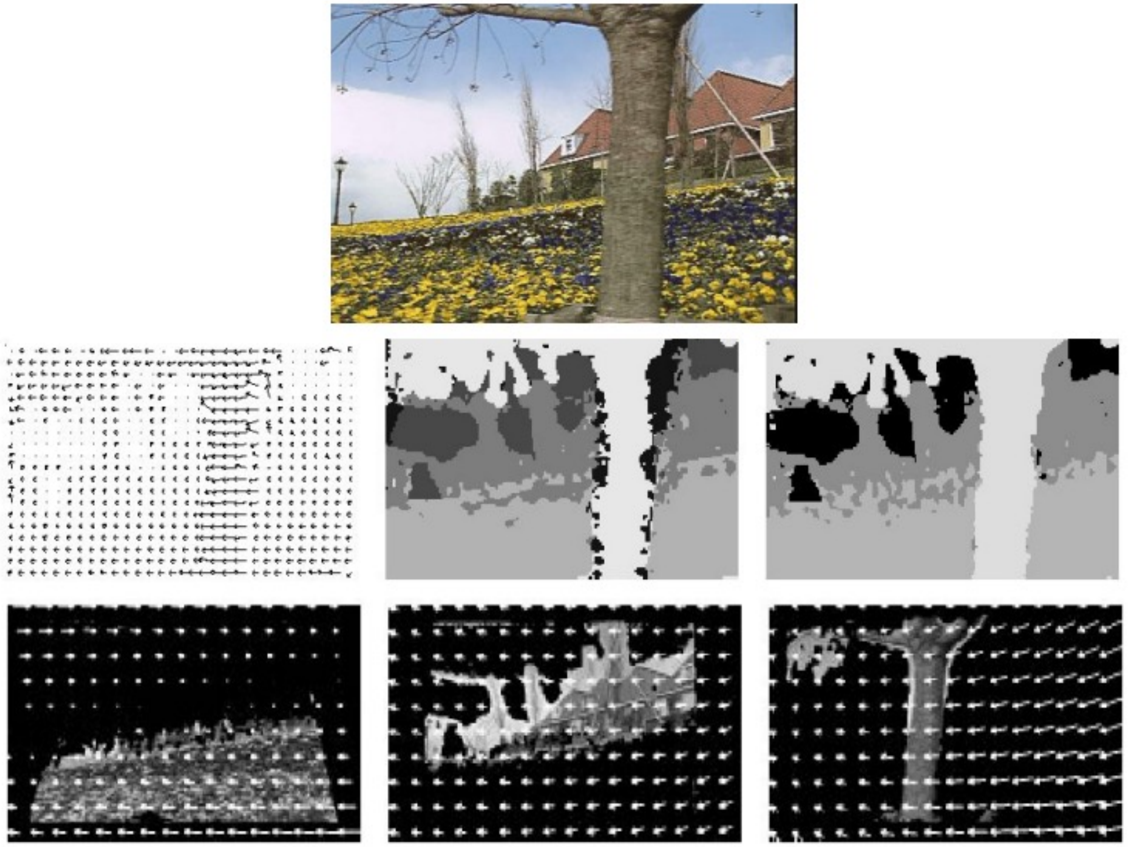
- Map into motion parameter space
- Perform k-means clustering on affine motion parameters
  - Merge clusters that are close and retain the largest clusters to obtain a smaller set of hypotheses to describe all the motions in the scene

2. Iterate until convergence:

- Assign each pixel to best hypothesis
  - Pixels with high residual error remain unassigned
- Perform region filtering to enforce spatial constraints
- Re-estimate affine motions in each region

Source: Silvio Savarese

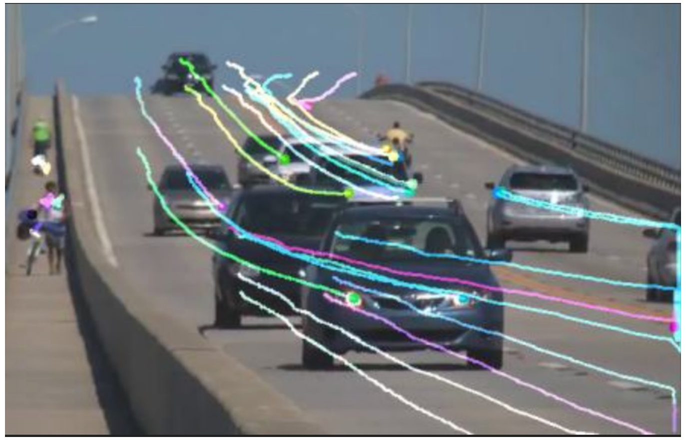
How to estimate layers?



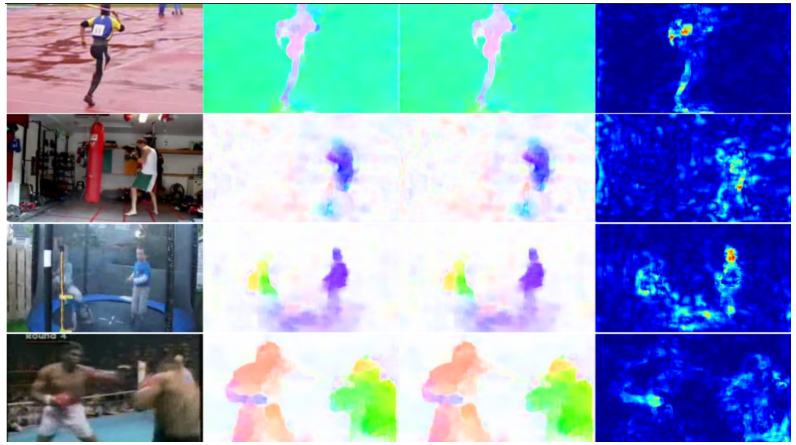
J. Wang and E. Adelson. Layered Representation for Motion Analysis. *CVPR 1993*.

Source: Silvio Savarese

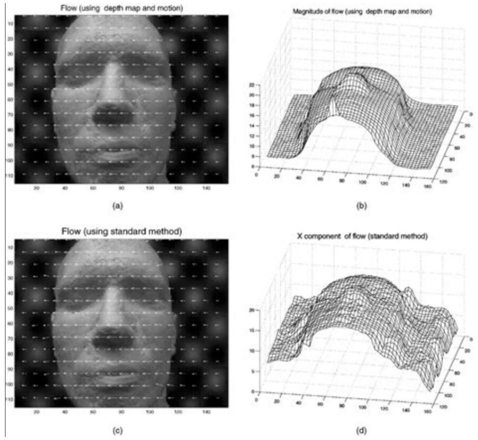
Applications-Optical Flow



Trajectory estimation



Motion estimation



3D Reconstruction