C280, Computer Vision

Prof. Trevor Darrell
trevor@eecs.berkeley.edu

Lecture 9: Motion

Roadmap

- Previous: Image formation, filtering, local features, (Texture)…
- Tues: Feature-based Alignment
  - Stitching images together
  - Homographies, RANSAC, Warping, Blending
  - Global alignment of planar models
- **Today: Dense Motion Models**
  - **Local motion / feature displacement**
  - **Parametric optic flow**
- No classes next week: ICCV conference
- Oct 6th: Stereo / ‘Multi-view’: Estimating depth with known inter-camera pose
- Oct 8th: ‘Structure-from-motion’: Estimation of pose and 3D structure
  - Factorization approaches
  - Global alignment with 3D point models

Last Time: Alignment

- Homographies
- Rotational Panoramas
- RANSAC
- Global alignment
- Warping
- Blending

Today: Motion and Flow

- Motion estimation
- Patch-based motion (optic flow)
- Regularization and line processes
- Parametric (global) motion
- Layered motion models

Why estimate visual motion?

- Visual Motion can be annoying
  - Camera instabilities, jitter
  - Measure it; remove it (stabilize)
- Visual Motion indicates dynamics in the scene
  - Moving objects, behavior
  - Track objects and analyze trajectories
- Visual Motion reveals spatial layout
  - Motion parallax

Szeliski

Classes of Techniques

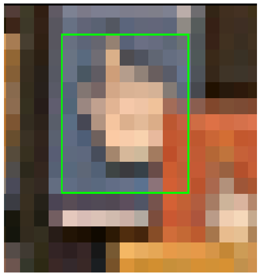
- ***Feature-based methods***
  - Extract visual features (corners, textured areas) and track them
  - Sparse motion fields, but possibly robust tracking
  - Suitable especially when image motion is large (10s of pixels)
- ***Direct-methods***
  - Directly recover image motion from spatio-temporal image brightness variations
  - Global motion parameters directly recovered without an intermediate feature motion calculation
  - Dense motion fields, but more sensitive to appearance variations
  - Suitable for video and when image motion is small (< 10 pixels)

Szeliski

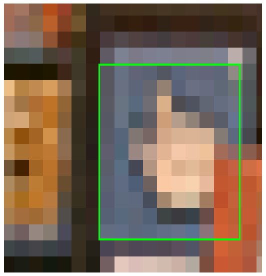
Patch-based motion estimation

Image motion

How do we determine correspondences?



I



J

Assume all change between frames is due to **motion:**

$$J(x, y) \approx I(x + u(x, y), y + v(x, y))$$

The Brightness Constraint

- Brightness Constancy Equation:

$$J(x, y) \approx \boxed{I(x+u(x, y), y+v(x, y))}$$

Or, equivalently, minimize :

$$E(u, v) = (J(x, y) - I(x+u, y+v))^2$$

Linearizing (assuming small *(u,v)*)
using Taylor series expansion:

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

Szeliski

Gradient Constraint (or the Optical Flow Constraint)

$$E(u,v) = (I_x \cdot u + I_y \cdot v + I_t)^2$$

Minimizing:

$$\frac{\partial E}{du} = \frac{\partial E}{dv} = 0$$

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

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

In general

$$I_x, I_y \neq 0$$

Hence,

$$I_x \cdot u + I_y \cdot v + I_t \approx 0$$

Szeliski

Patch Translation [Lucas-Kanade]

Assume a single velocity for all pixels within an image patch

$$E(u,v) = \sum_{x,y\in\Omega} \left(I_x(x,y)u + I_y(x,y)v + I_t\right)^2$$

Minimizing

$$\begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = -\begin{pmatrix} \sum I_x I_t \\ \sum I_y I_t \end{pmatrix}$$

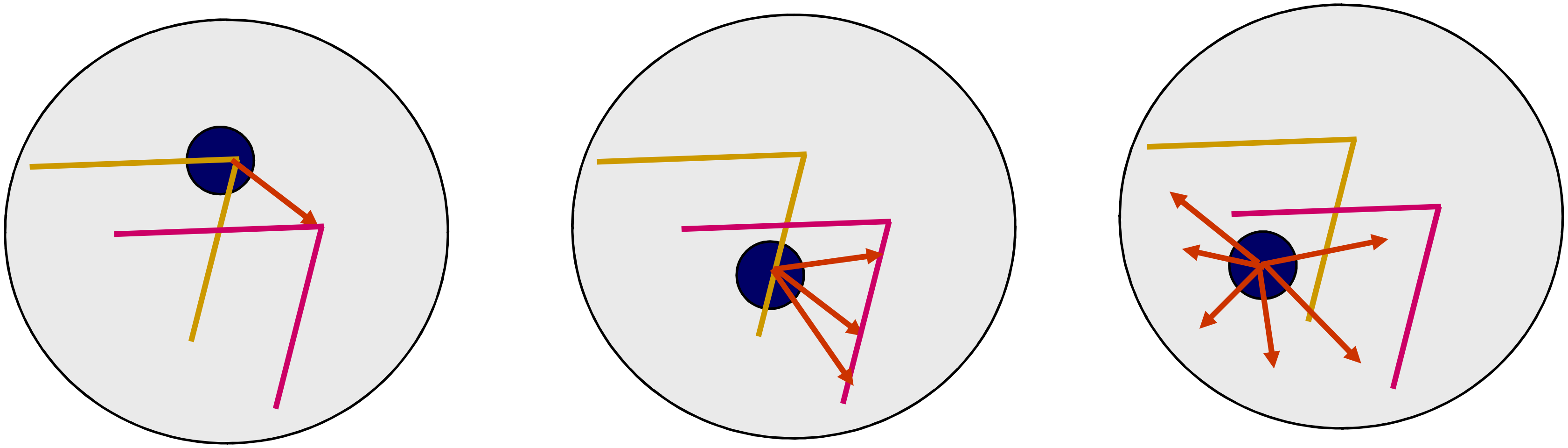
$$\left(\sum \nabla I \nabla I^T\right)\vec{U} = -\sum \nabla I I_t$$

LHS: sum of the 2x2 outer product of the gradient vector

Szeliski

Local Patch Analysis

- How *certain* are the motion estimates?



Szeliski

The Aperture Problem

Let $M = \sum (\nabla I)(\nabla I)^T$ and $b = \begin{bmatrix} -\sum I_x I_t \\ -\sum I_y I_t \end{bmatrix}$

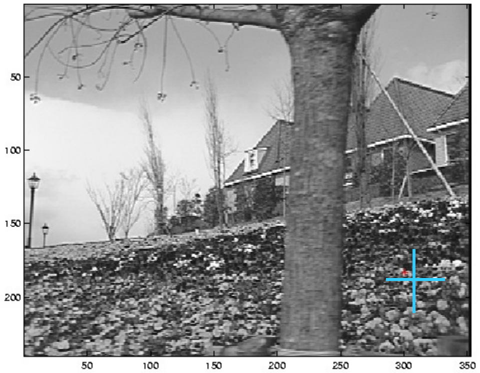
- Algorithm: At each pixel compute U by solving $MU = b$

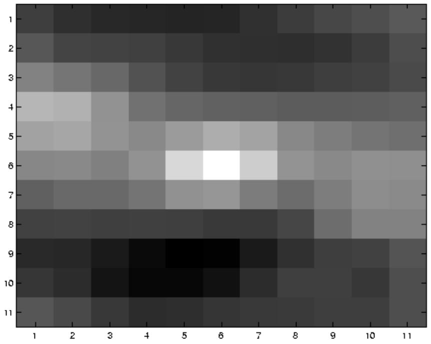
- *M* is singular if all gradient vectors point in the same direction
  - e.g., along an edge
  - of course, trivially singular if the summation is over a single pixel or there is no texture
  - i.e., only *normal flow* is available (aperture problem)

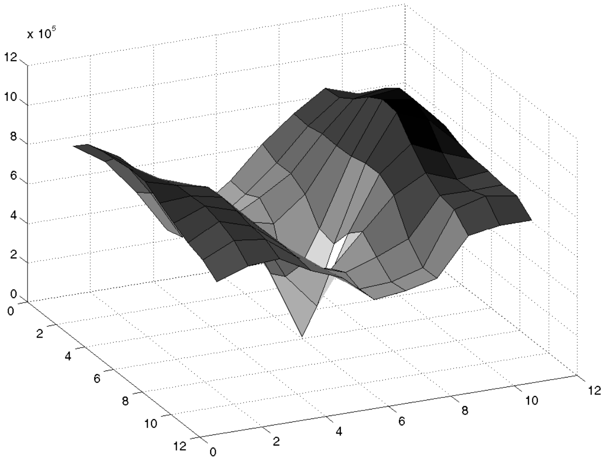
- Corners and textured areas are OK

Szeliski

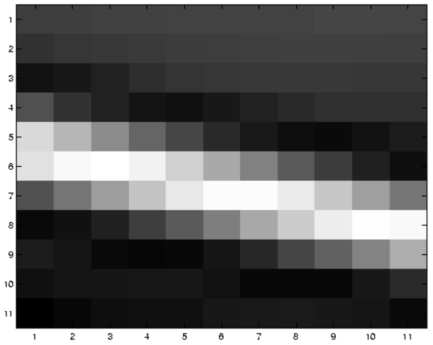
SSD Surface – Textured area

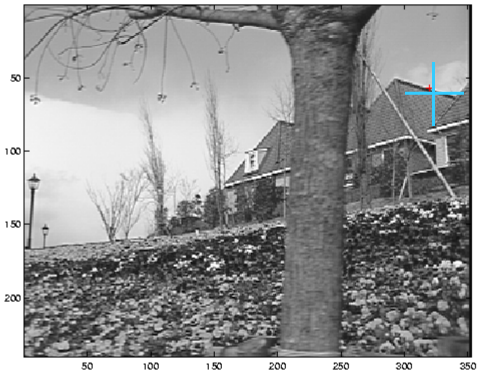


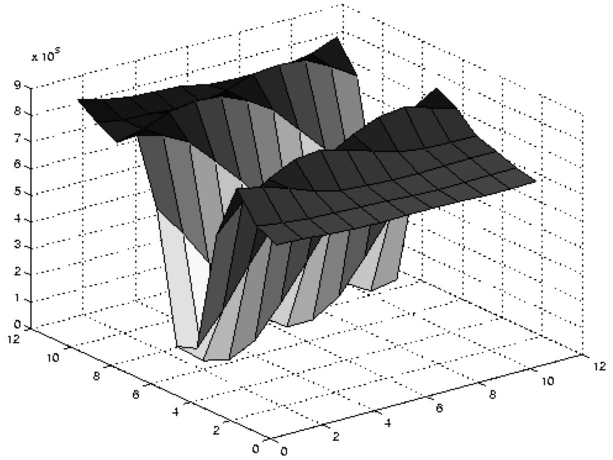




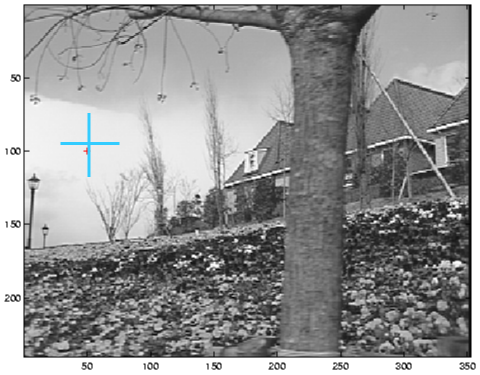
SSD Surface -- Edge

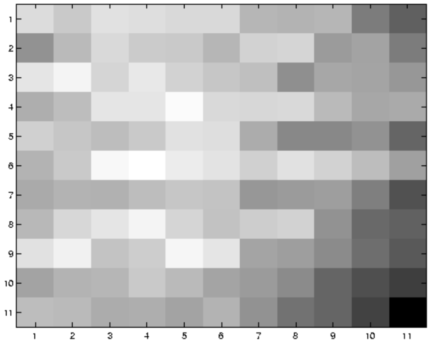


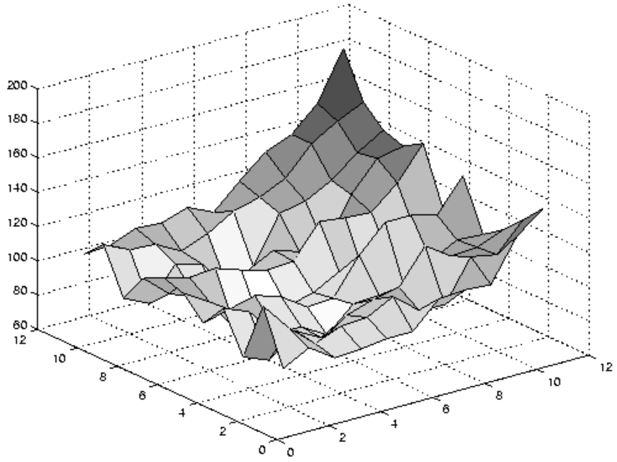




SSD – homogeneous area







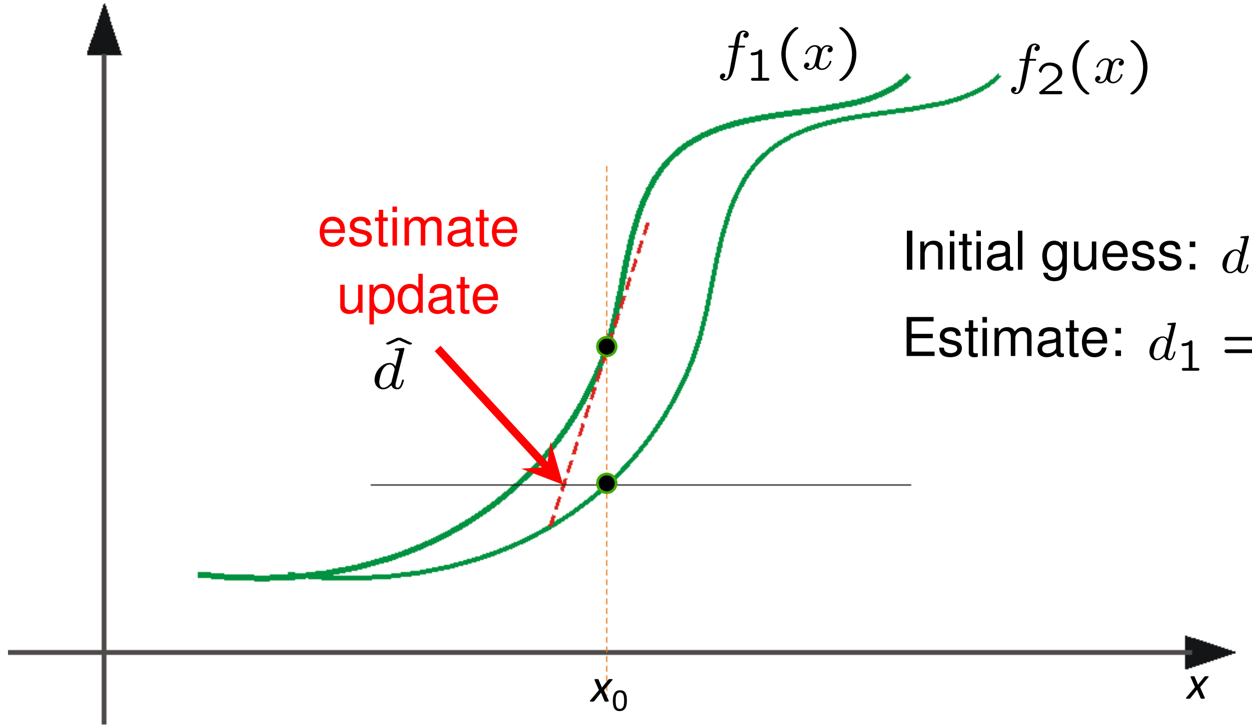
Iterative Refinement

- Estimate velocity at each pixel using one iteration of Lucas and Kanade estimation
- Warp one image toward the other using the estimated flow field

  (easier said than done)
- Refine estimate by repeating the process

Szeliski

Optical Flow: Iterative Estimation



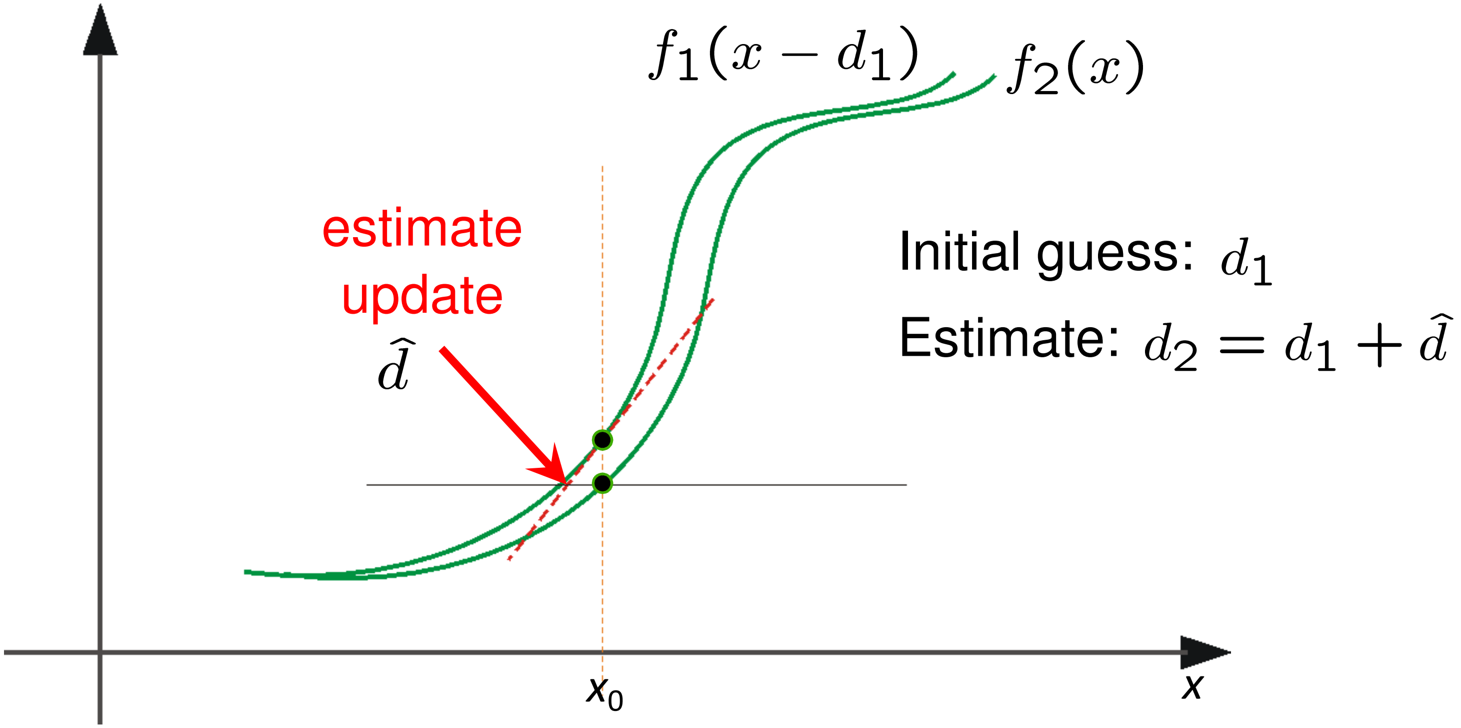
Initial guess: $d_0 = 0$

Estimate: $d_1 = d_0 + \hat{d}$

(using *d* for *displacement* here instead of *u*)

Szeliski

Optical Flow: Iterative Estimation

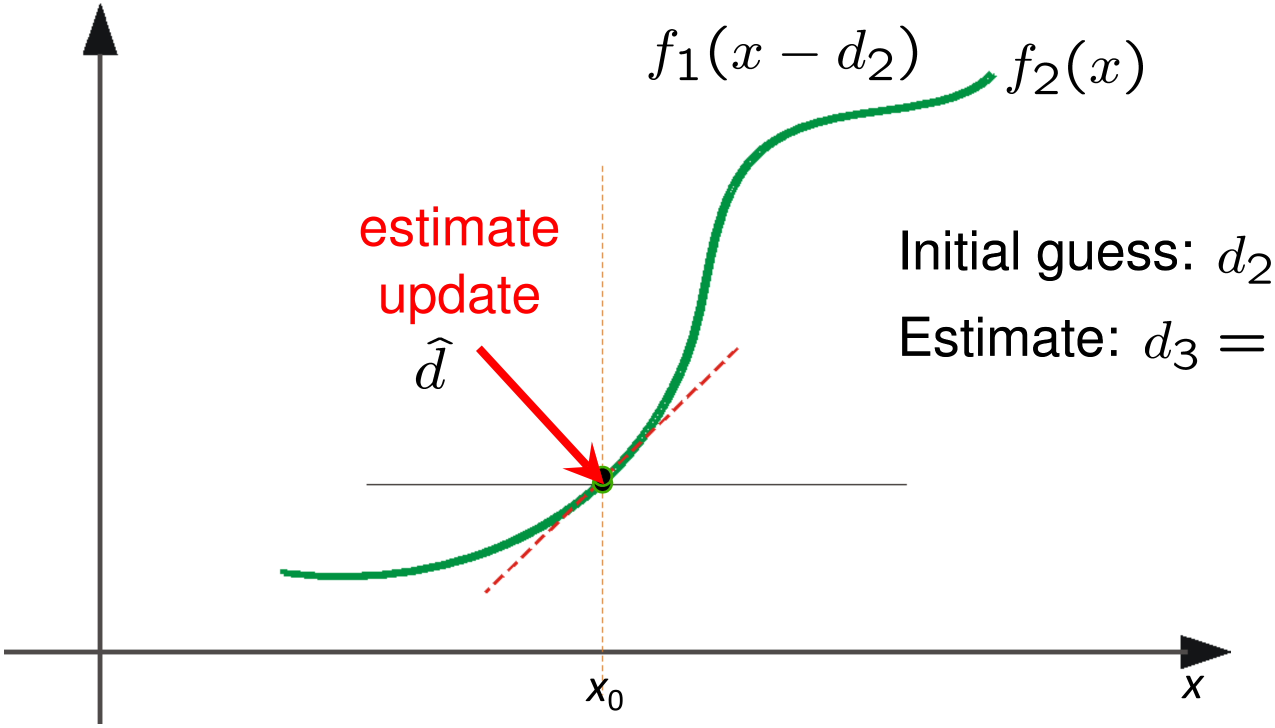


Initial guess: d_1

Estimate: $d_2 = d_1 + \hat{d}$

Szeliski

Optical Flow: Iterative Estimation

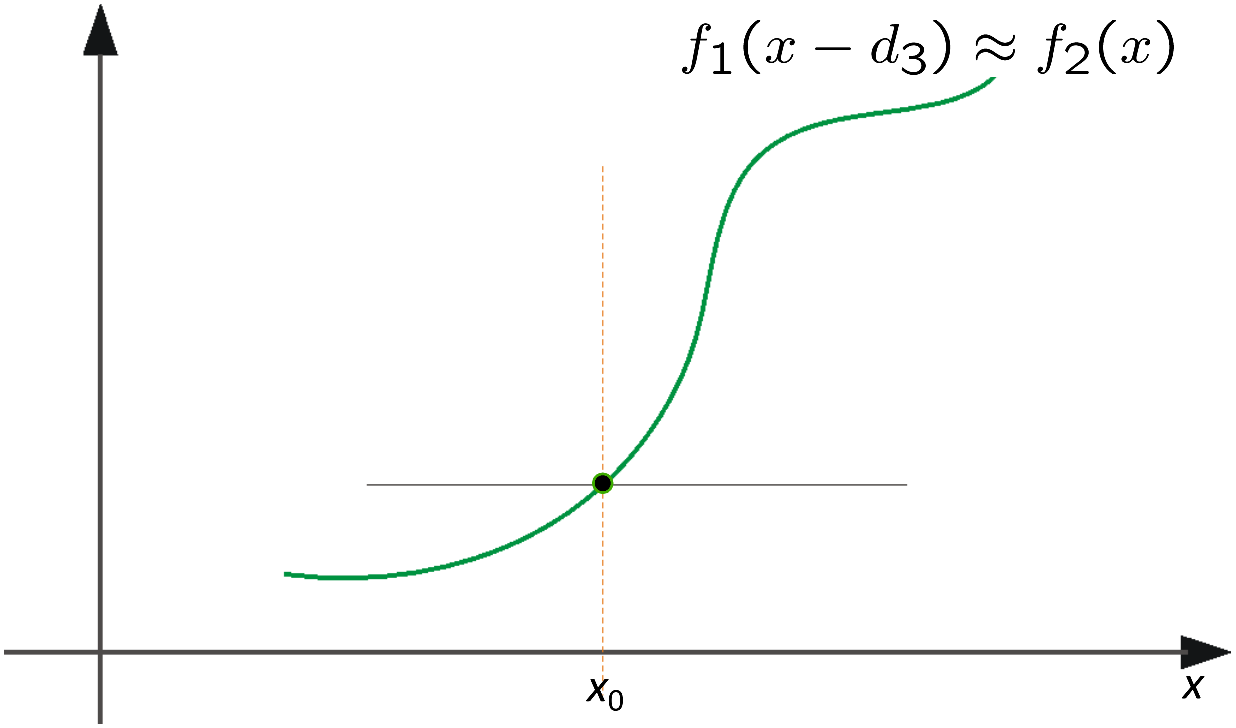


Initial guess: d_2

Estimate: $d_3 = d_2 + \hat{d}$

Szeliski

Optical Flow: Iterative Estimation



Szeliski

Optical Flow: Iterative Estimation

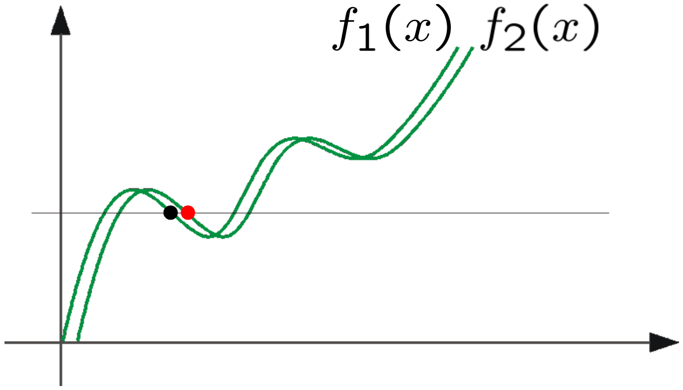
- Some Implementation Issues:
  - Warping is not easy (ensure that errors in warping are smaller than the estimate refinement)
  - Warp one image, take derivatives of the other so you don’t need to re-compute the gradient after each iteration.
  - Often useful to low-pass filter the images before motion estimation (for better derivative estimation, and linear approximations to image intensity)

Szeliski

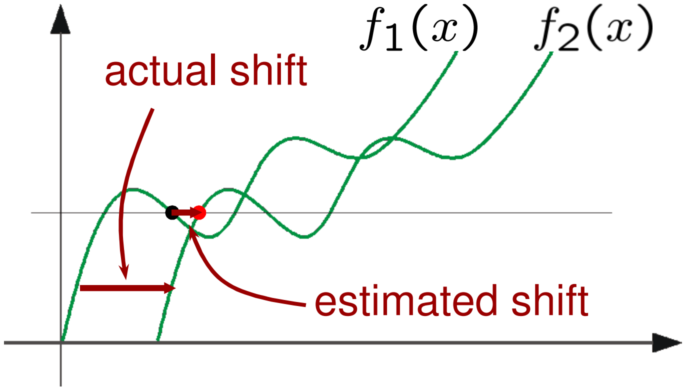
Optical Flow: Aliasing

Temporal aliasing causes ambiguities in optical flow because images can have many pixels with the same intensity.

I.e., how do we know which ‘correspondence’ is correct?



nearest match is correct
(no aliasing)



nearest match is incorrect
(aliasing)

To overcome aliasing: coarse-to-fine estimation.

Szeliski

Limits of the gradient method

Fails when intensity structure in window is poor

Fails when the displacement is large (typical operating range is motion of 1 pixel)

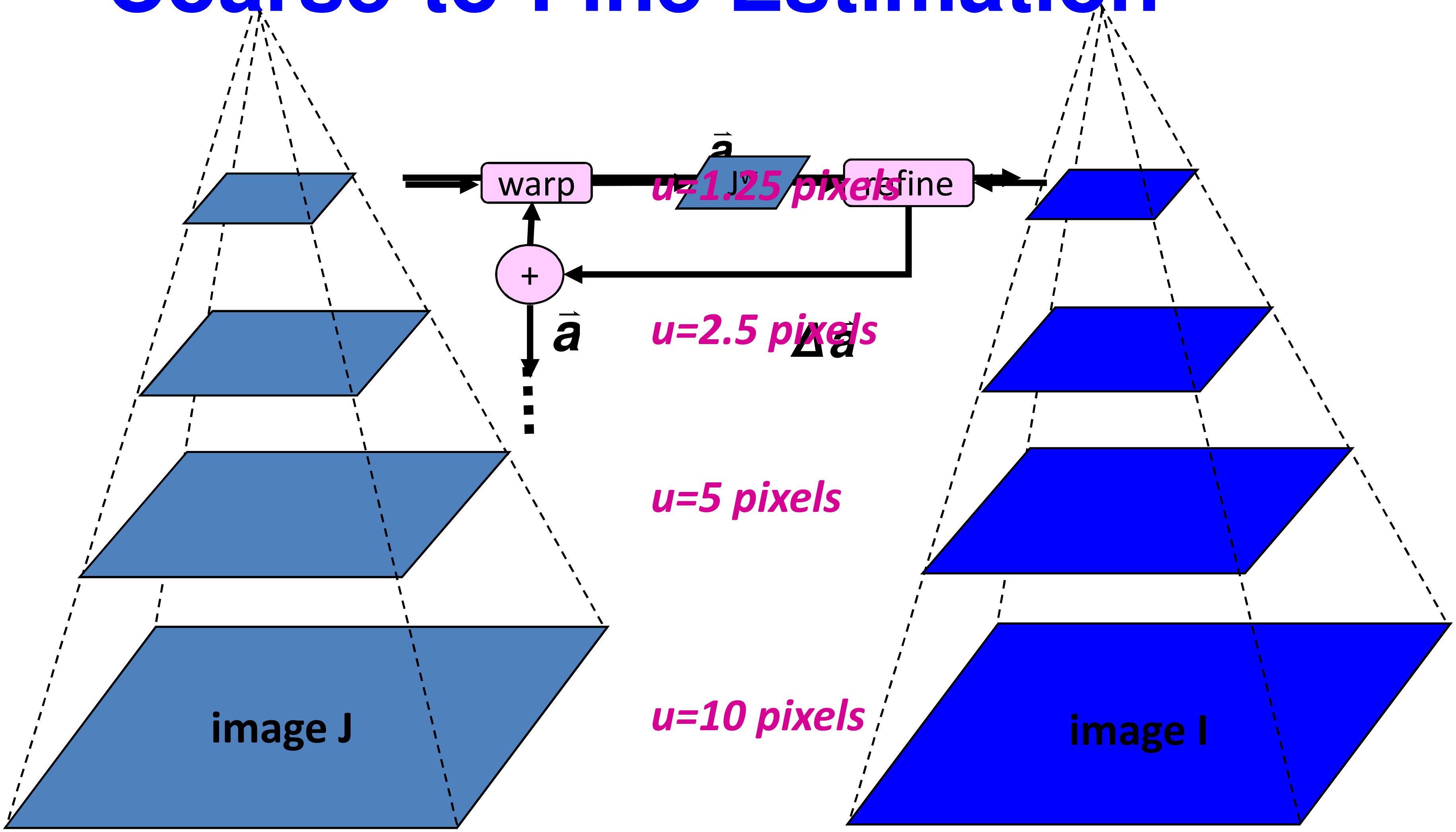
Linearization of brightness is suitable only for small displacements

- Also, brightness is not strictly constant in images

actually less problematic than it appears, since we can pre-filter images to make them look similar

Szeliski

Coarse-to-Fine Estimation

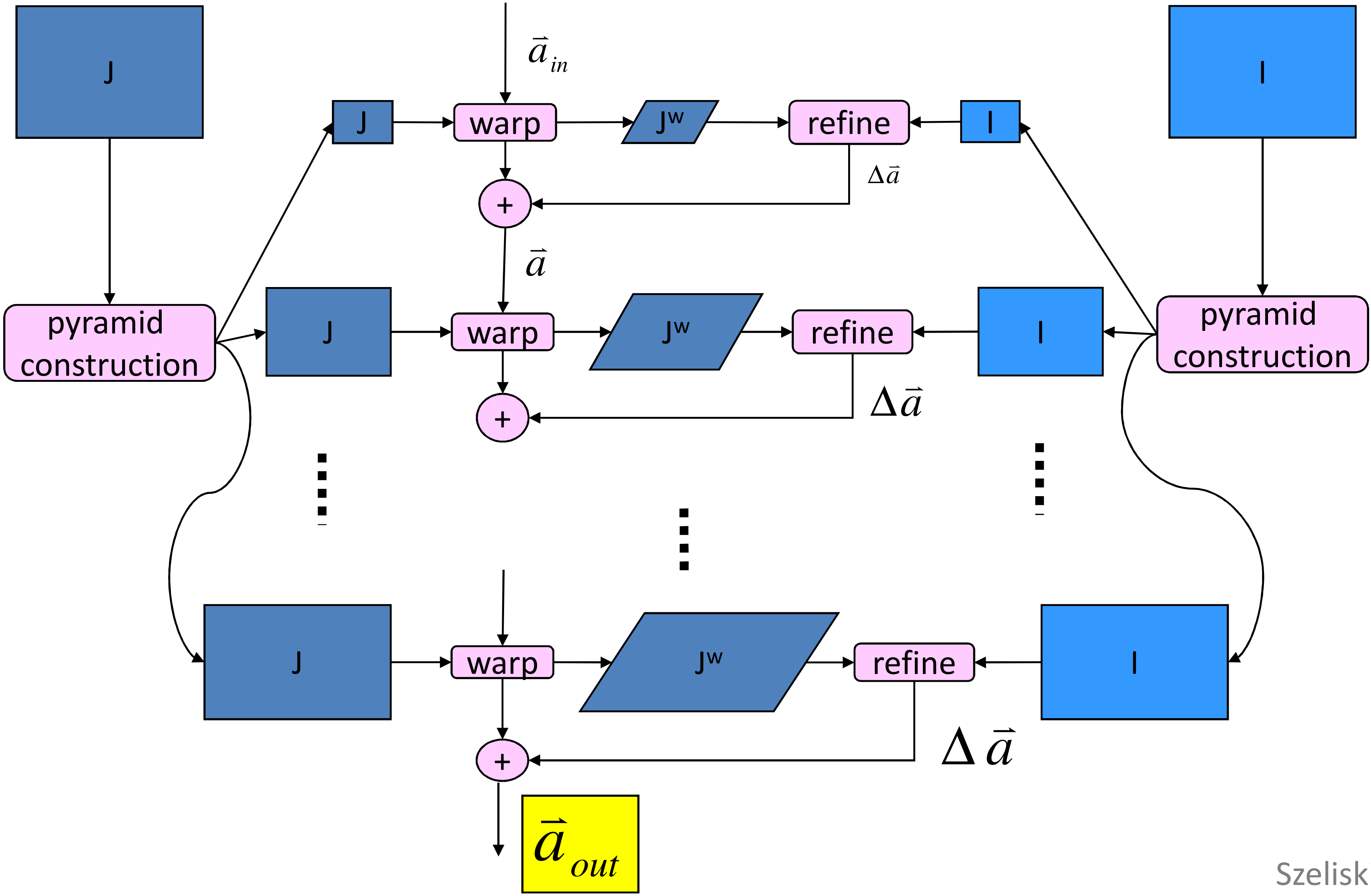


Pyramid of image J

Pyramid of image I

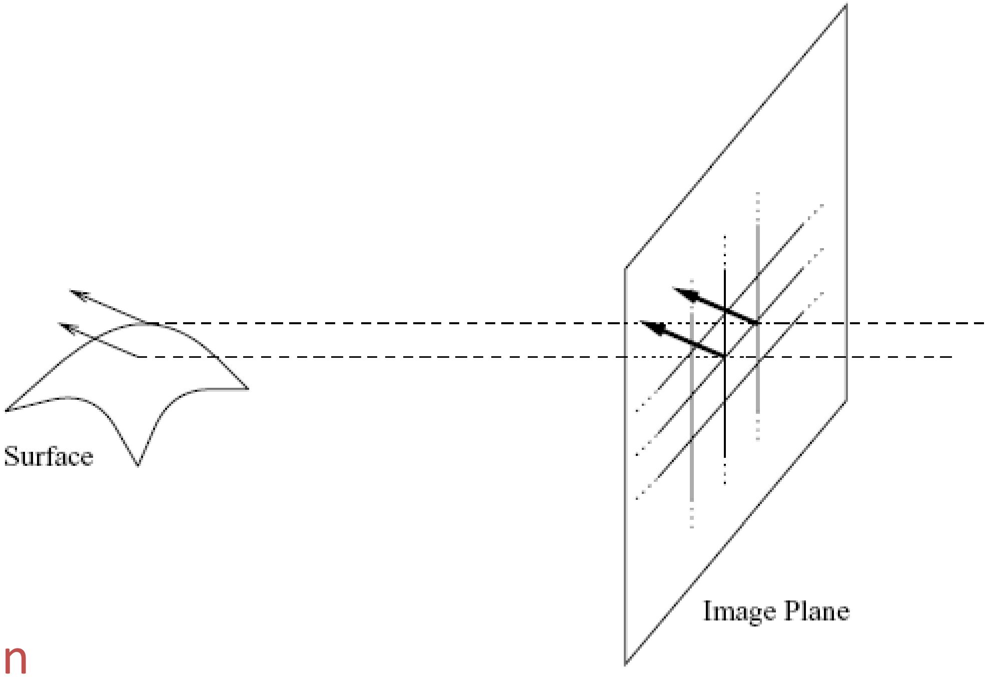
Szeliski

Coarse-to-Fine Estimation



Szeliski

Spatial Coherence



Assumption

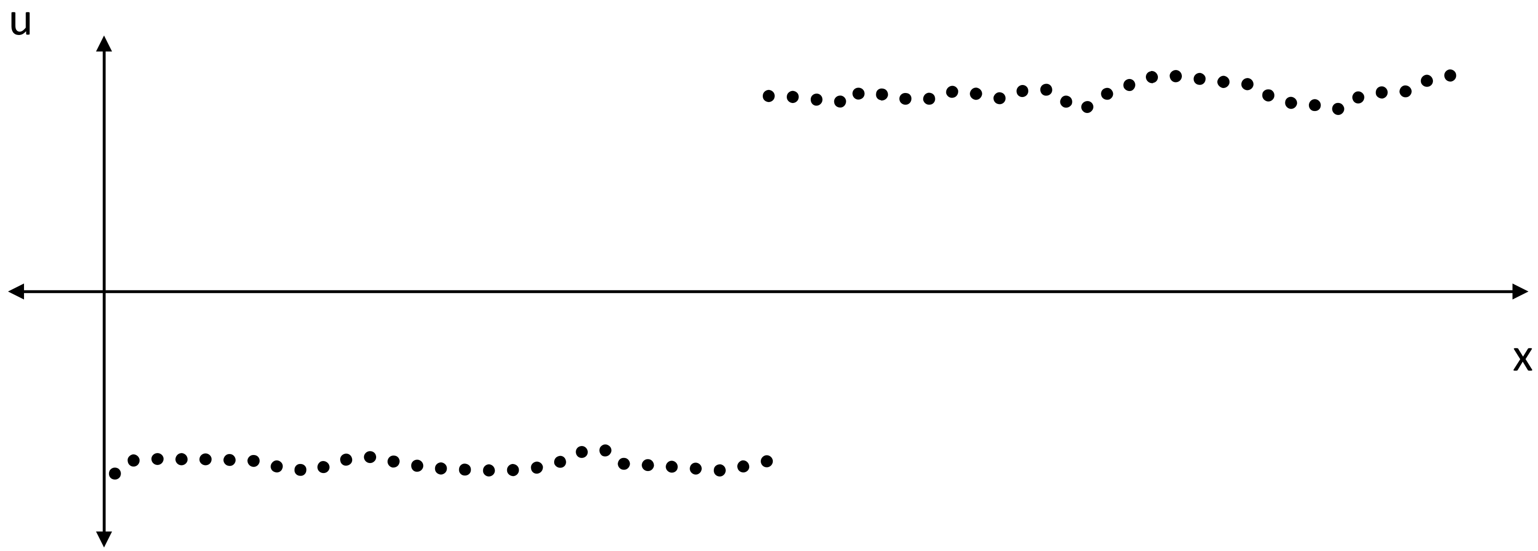
* Neighboring points in the scene typically belong to the same surface and hence typically have similar motions.

* Since they also project to nearby points in the image, we expect spatial coherence in image flow.

Black

Formalize this Idea

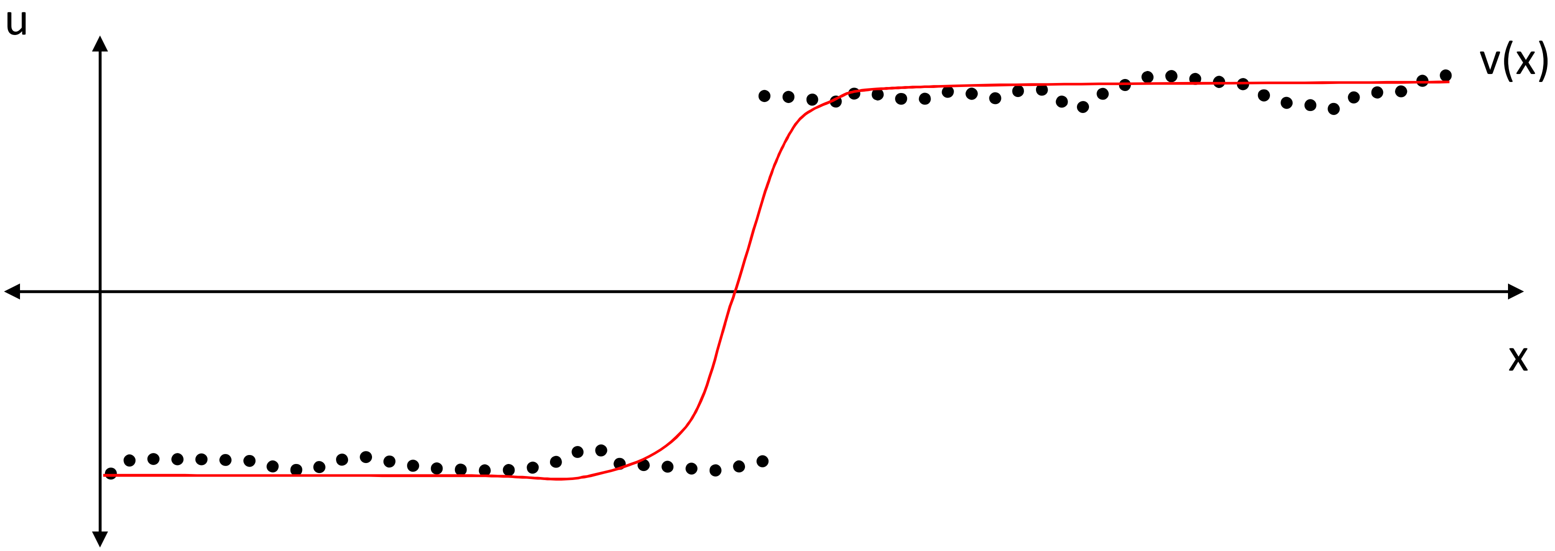
Noisy 1D signal:



Noisy measurementsu(x)

Regularization

Find the “best fitting” smoothed function v(x)

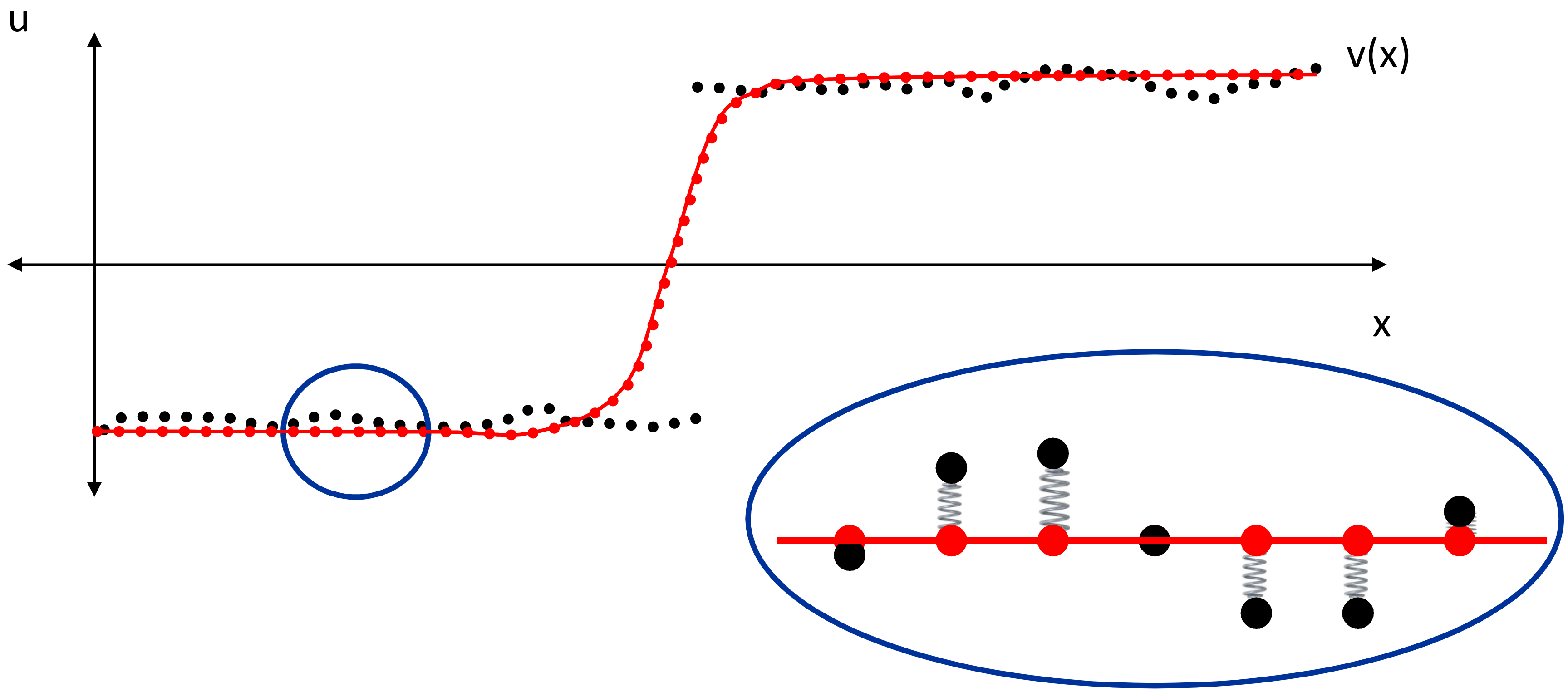


Noisy measurementsu(x)

Black

Membrane model

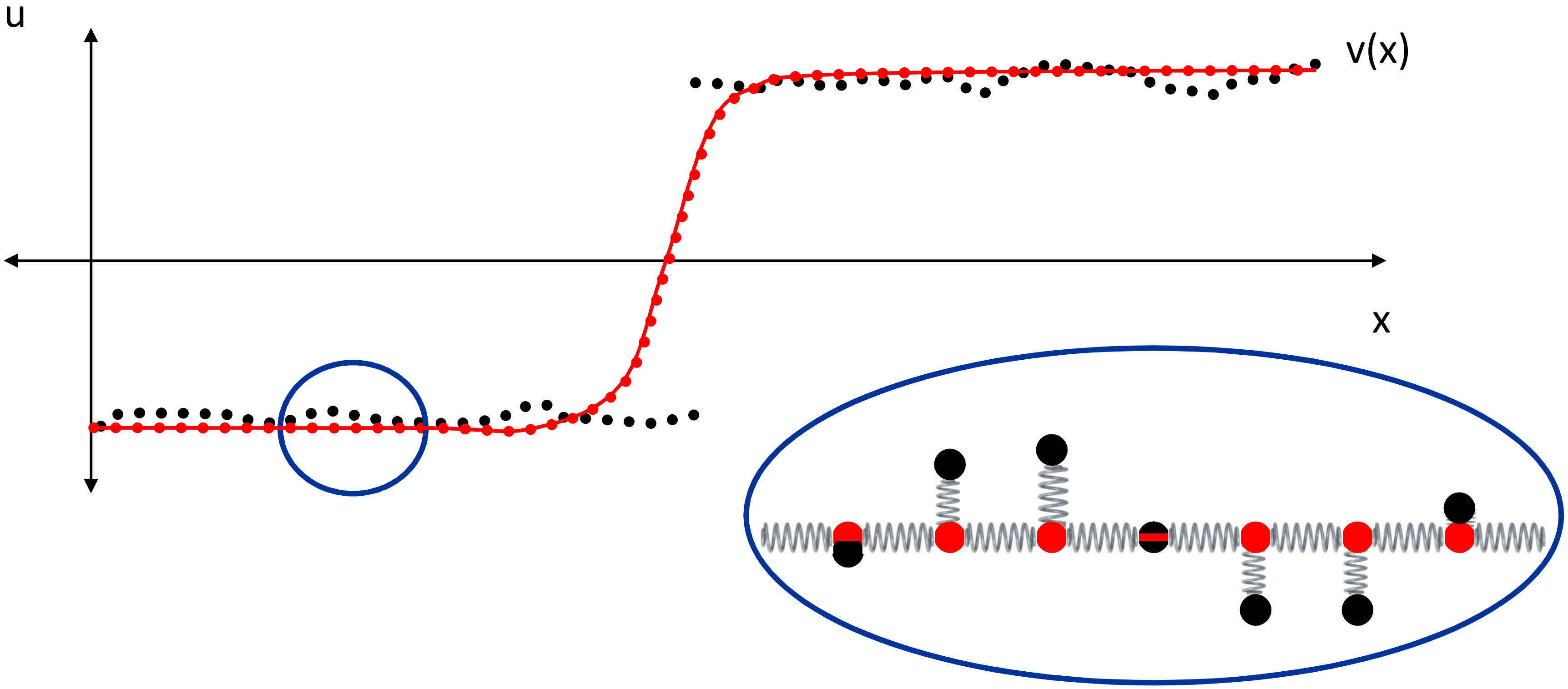
Find the “best fitting” smoothed function v(x)



Black

Membrane model

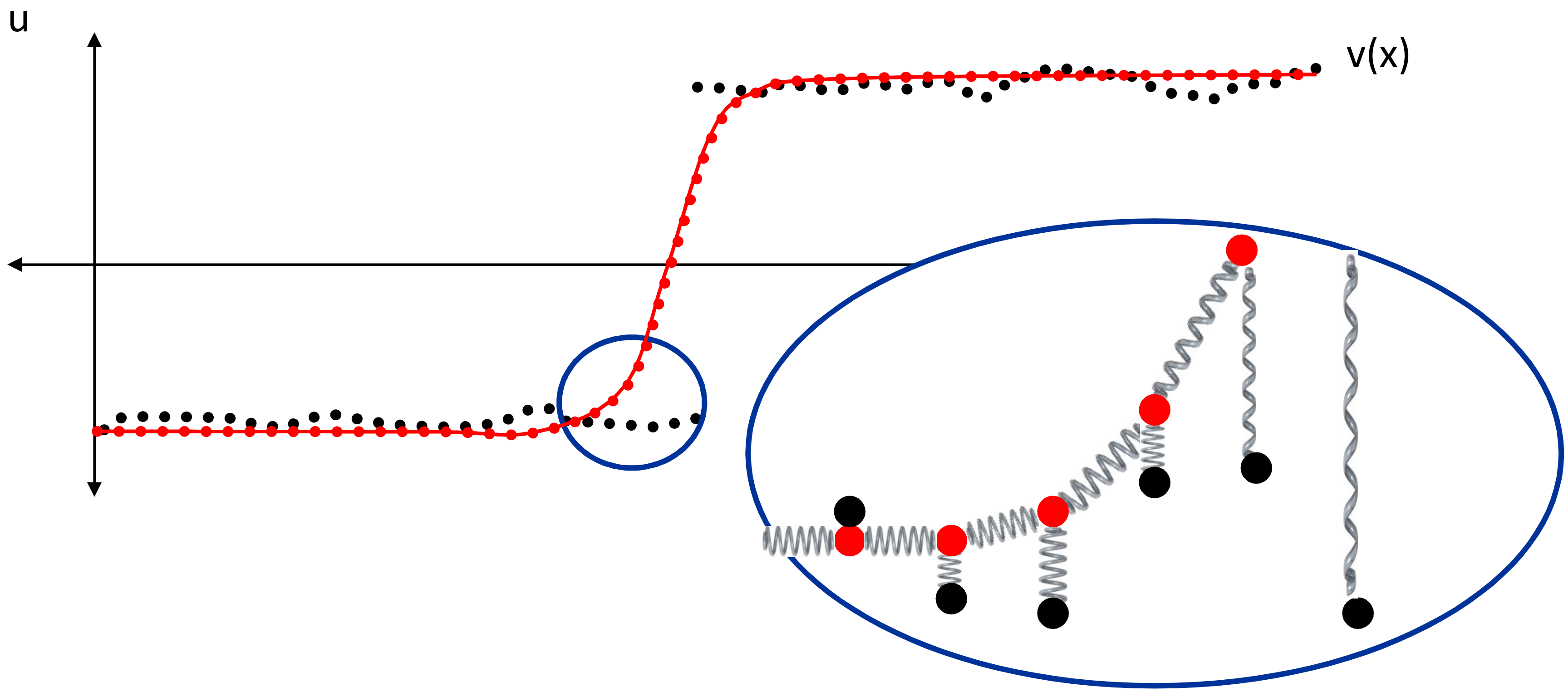
Find the “best fitting” smoothed function v(x)



Black

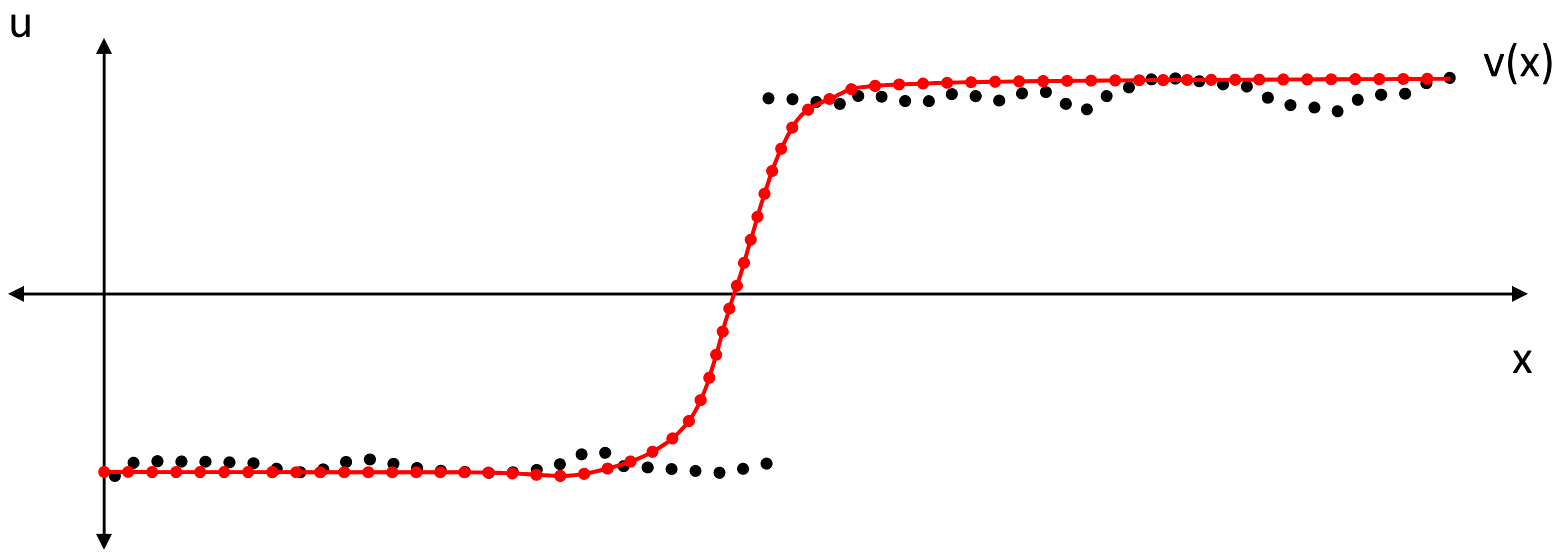
Membrane model

Find the “best fitting” smoothed function v(x)



Black

Regularization



Minimize:

Faithful to the data

Spatial smoothness assumption

$$E(v)=\sum_{x=1}^{N}(v(x)-u(x))^2+\lambda\sum_{x=1}^{N-1}(v(x+1)-v(x))^2$$

Black

Bayesian Interpretation

$$E(v) = \sum_{x=1}^{N} (v(x) - u(x))^2 + \lambda \sum_{x=1}^{N-1} (v(x+1) - v(x))^2$$

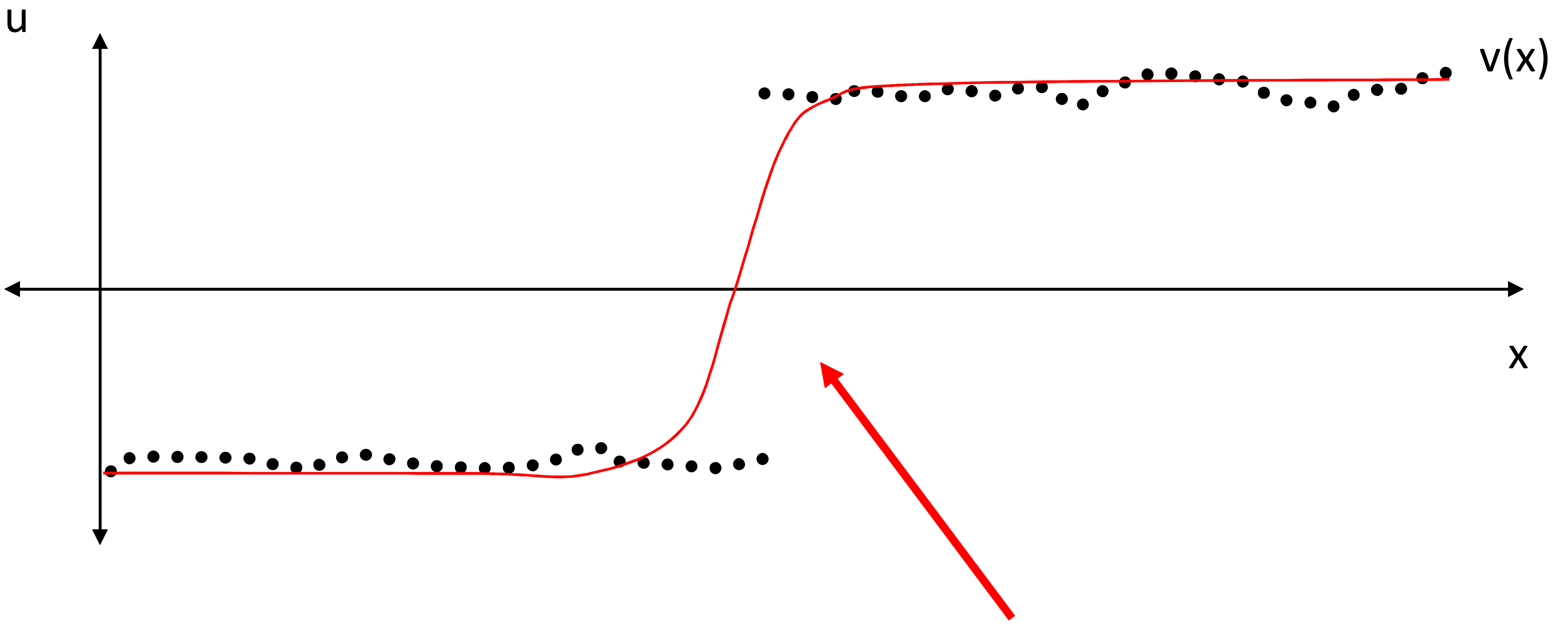
$$p(v \mid u) \propto p(u \mid v)\, p(v)$$

$$u(x) = v(x) + \eta \quad \eta \sim N(0, \sigma_1) \qquad p(u \mid v) = \prod_{x=1}^{N} \frac{1}{\sqrt{2\pi}\sigma_1} \exp(-\frac{1}{2}(u(x) - v(x))^2 / \sigma_1^2)$$

$$v(x) = v(x+1) + \eta_2 \quad \eta_2 \sim N(0, \sigma_2) \qquad p(v) = \prod_{x=1}^{N-1} \frac{1}{\sqrt{2\pi}\sigma_2} \exp(-\frac{1}{2}(v_x(x))^2 / \sigma_2^2)^{\lambda}$$

Black

Discontinuities

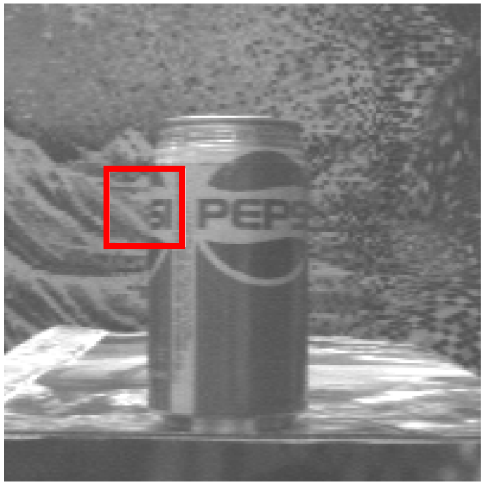


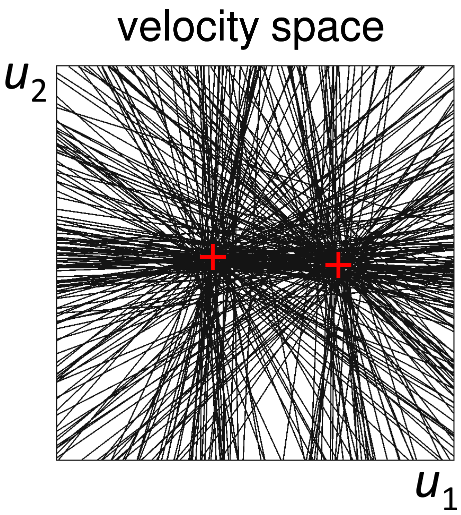
What about this discontinuity?
What is happening here?
What can we do?

Robust Estimation

Noise distributions are often non-Gaussian, having much heavier tails. Noise samples from the tails are called outliers.

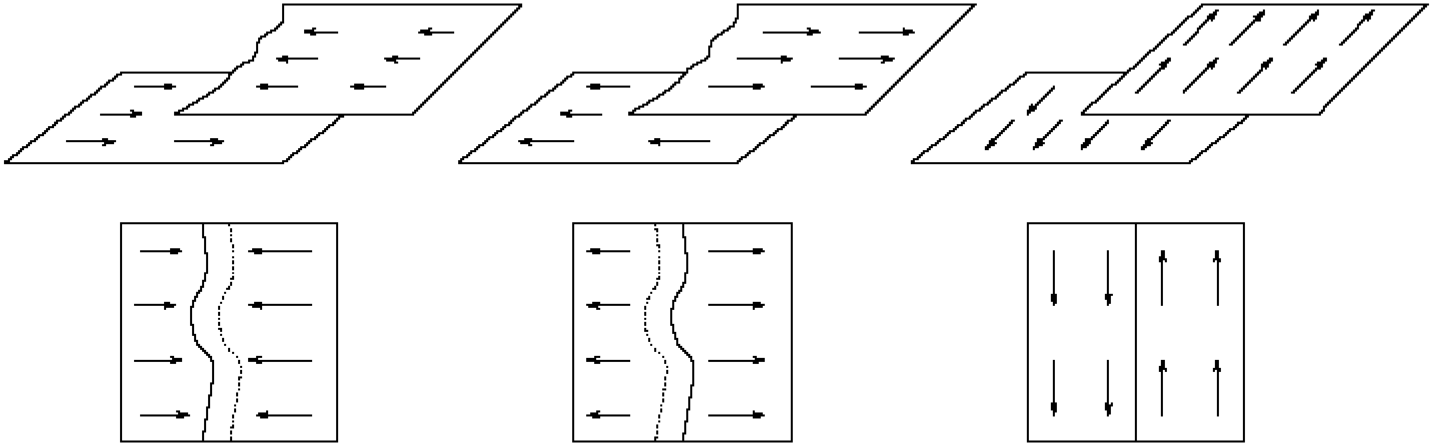
- Sources of outliers (multiple motions):
  - specularities / highlights
  - jpeg artifacts / interlacing / motion blur
  - multiple motions (occlusion boundaries, transparency)





Black

Occlusion



occlusion

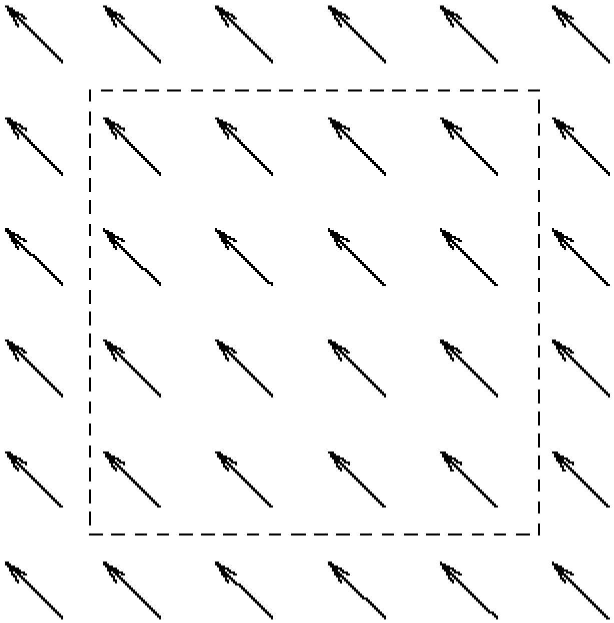
disocclusion

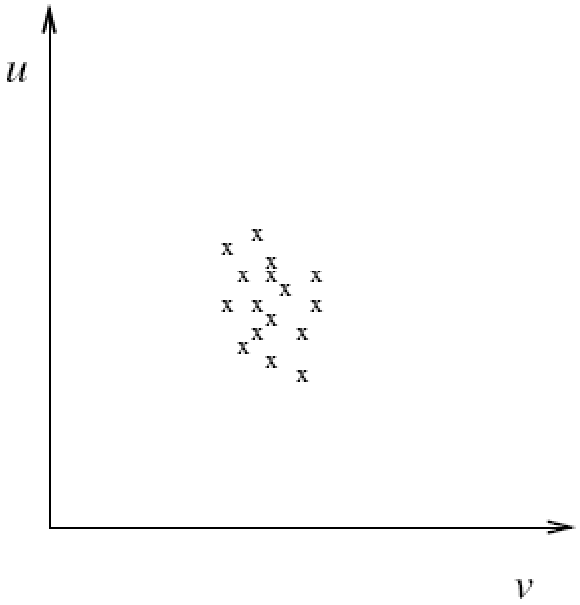
shear

Multiple motions within a finite region.

Black

Coherent Motion

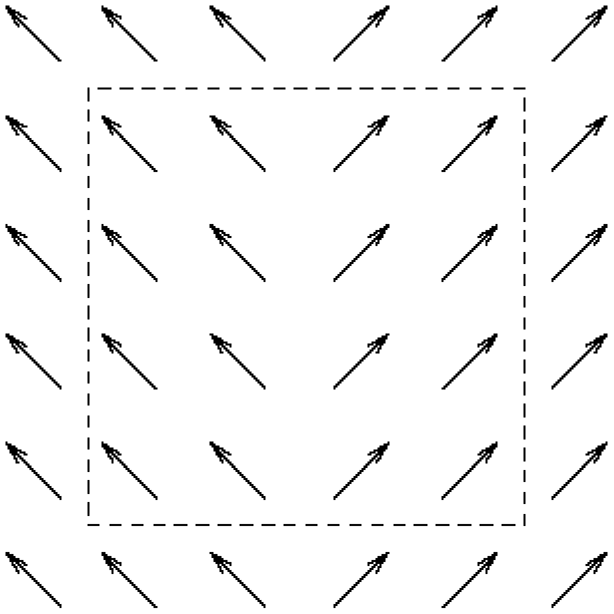


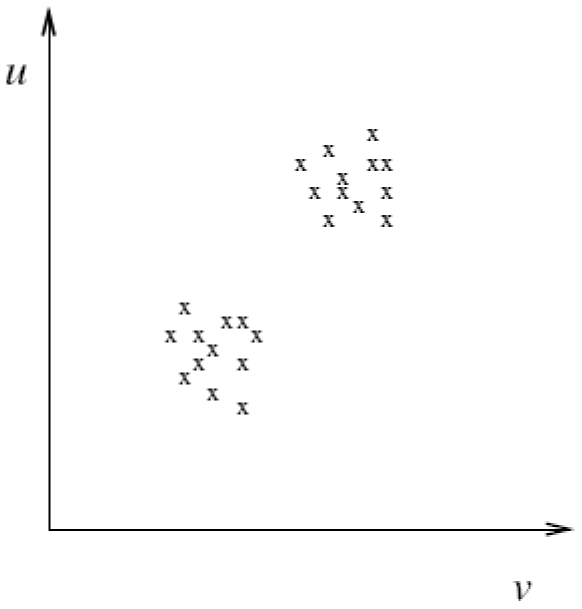


Possibly Gaussian.

Black

Multiple Motions

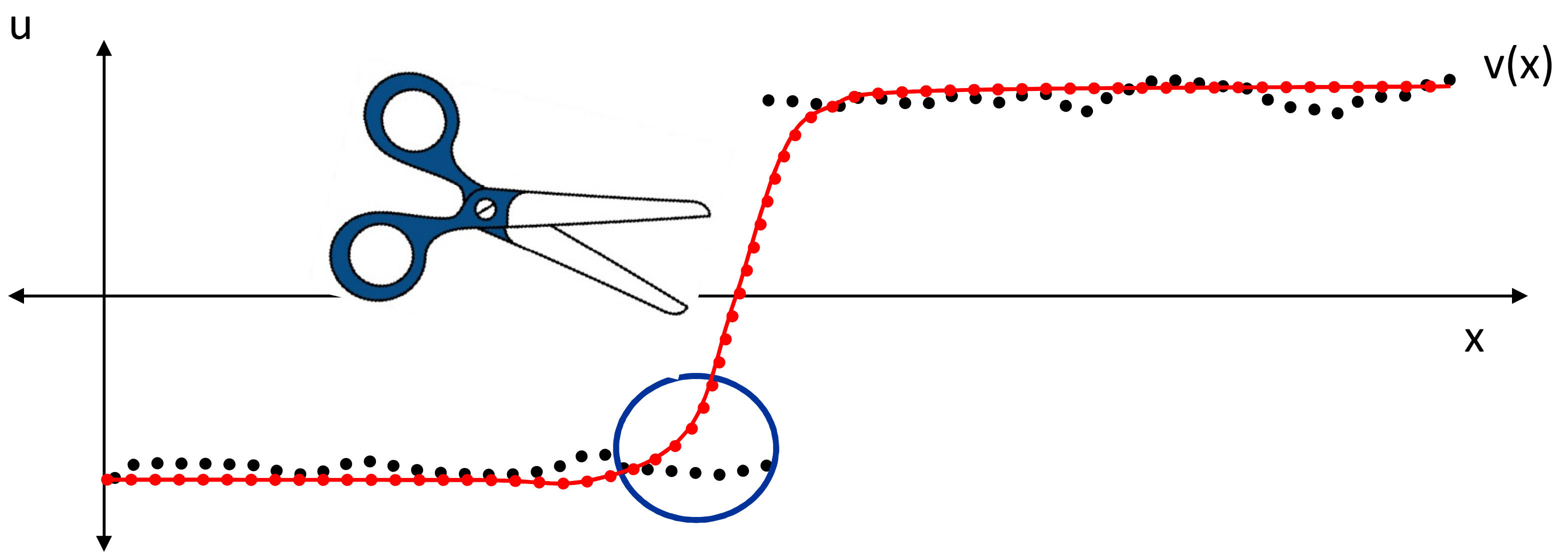




Definitely not Gaussian.

Black

Weak membrane model



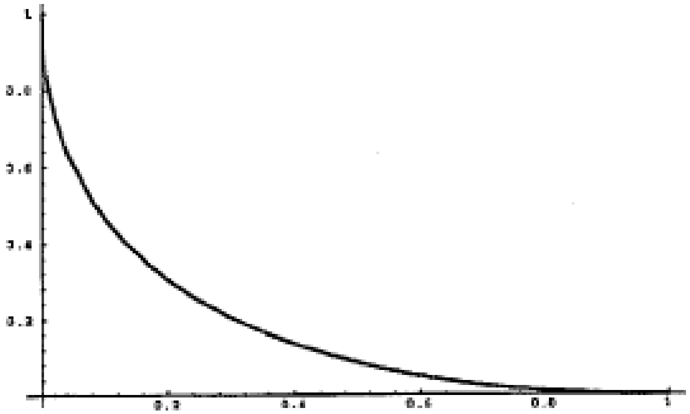
$$E(v,l)=\sum_{x=1}^{N}(v(x)-u(x))^2+\lambda\sum_{x=1}^{N-1}\left[l(x)(v(x+1)-v(x))^2+\beta(1-l(x))\right]$$

$$l(x)\in\{0,1\}$$

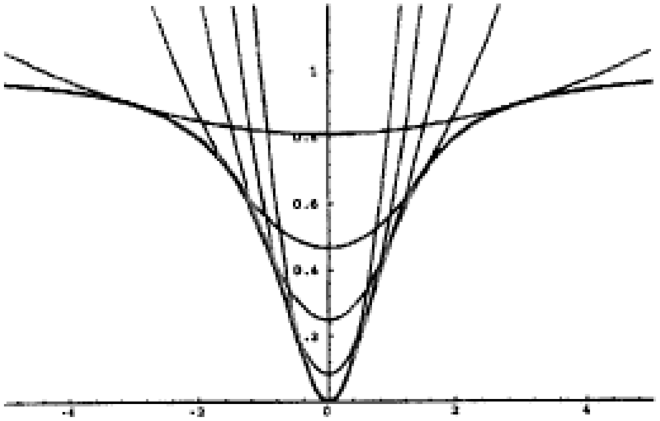
Black

Analog line process

Penalty function



Family of quadratics



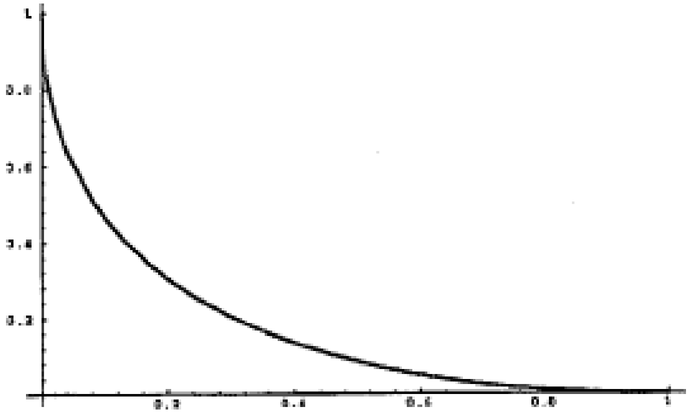
$$E(v,l)=\sum_{x=1}^{N}(v(x)-u(x))^2+\lambda\sum_{x=1}^{N-1}\left[l(x)(v(x+1)-v(x))^2+\Psi(l(x))\right]$$

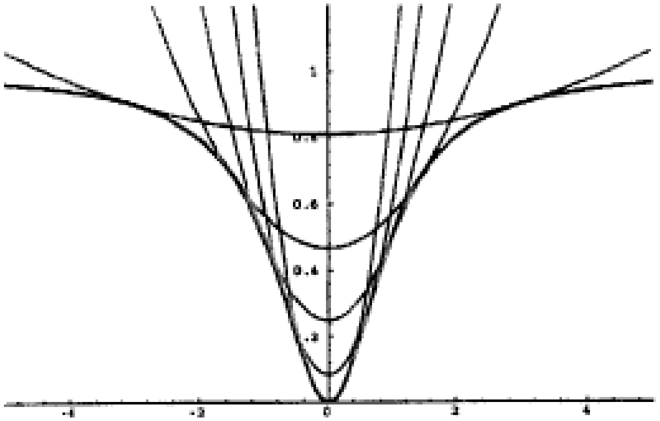
$$0\le l(x)\le 1$$

Black

Analog line process

Infimum defines a robust error function.





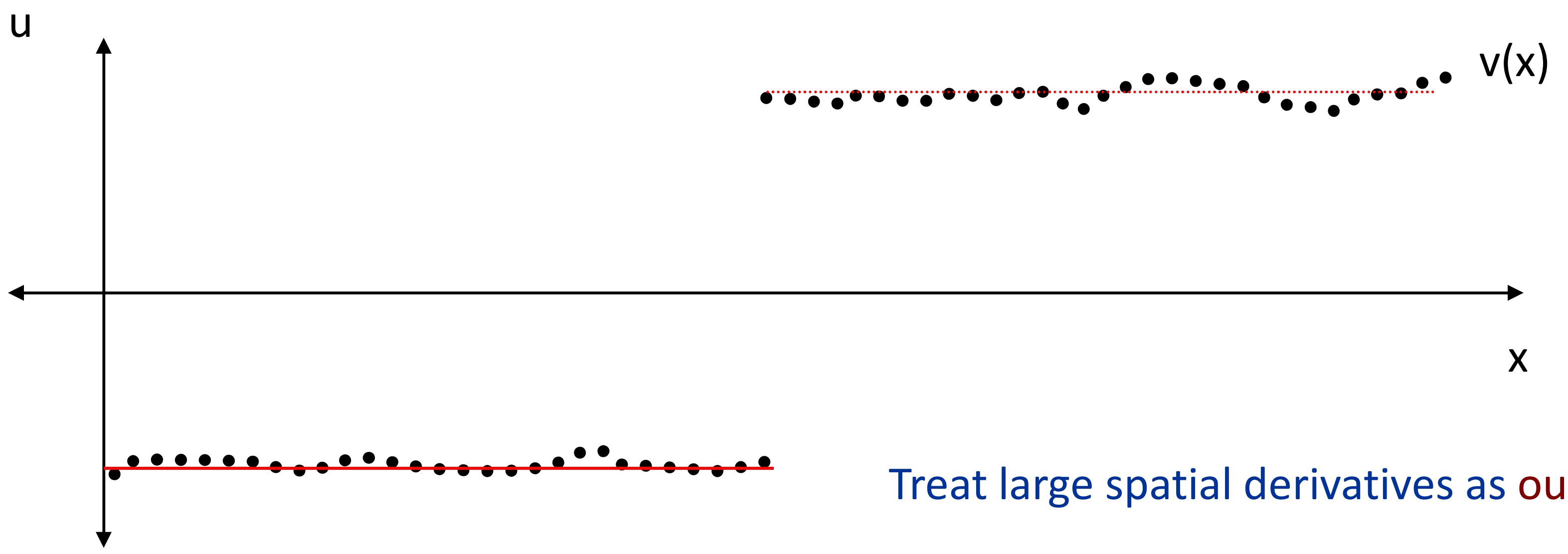
Minima are the same:

$$E(v,l)=\sum_{x=1}^{N}(v(x)-u(x))^2+\lambda\sum_{x=1}^{N-1}\left[l(x)(v(x+1)-v(x))^2+\Psi(l(x))\right]$$

$$E(v)=\sum_{x=1}^{N}(v(x)-u(x))^2+\lambda\sum_{x=1}^{N-1}\rho(v(x+1)-v(x),\sigma_2)$$

Black

Robust Regularization



Treat large spatial derivatives as outliers.

Minimize:

$$E(v) = \sum_{x=1}^{N} \rho(v(x) - u(x), \sigma_1) + \lambda \sum_{x=1}^{N-1} \rho(v(x+1) - v(x), \sigma_2)$$

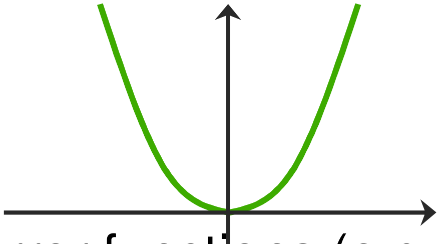
Black

Robust Estimation

Problem: Least-squares estimators penalize deviations between data & model with quadratic error f^n (extremely sensitive to outliers)

error penalty function

$$\rho(\epsilon) = \epsilon^2$$



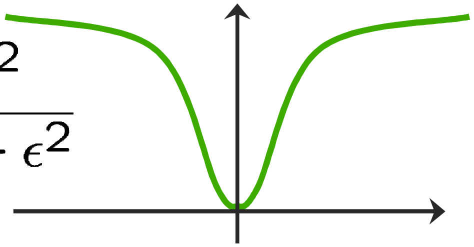
influence function

$$\psi(\epsilon) = \frac{\partial \rho(\epsilon)}{\partial \epsilon} = 2\,\epsilon$$

Redescending error functions (e.g., Geman-McClure) help to reduce the influence of outlying measurements.

error penalty function

$$\rho(\epsilon;\, s) = \frac{\epsilon^2}{s + \epsilon^2}$$

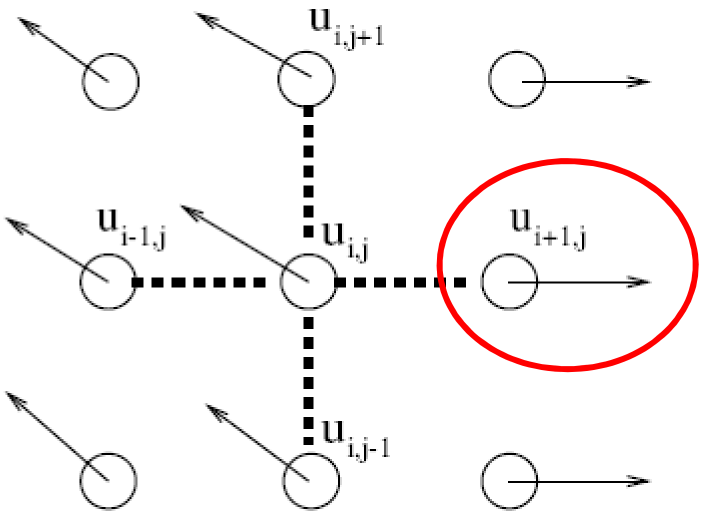


influence function

$$\psi(\epsilon;\, s) = \frac{2\,\epsilon\, s}{(s + \epsilon^2)^2}$$

Black

Optical flow



Outlier with respect to neighbors.

Robust formulation of spatial coherence term

$$E_S(u,v) = \rho(u_x) + \rho(u_y) + \rho(v_x) + \rho(v_y)$$

Black

“Dense” Optical Flow

$$E_D(\mathbf{u}(\mathbf{x})) = \rho(I_x(\mathbf{x})u(\mathbf{x}) + I_y(\mathbf{x})v(\mathbf{x}) + I_t(\mathbf{x}), \sigma_D)$$

$$E_S(u,v) = \sum_{\mathbf{y} \in G(\mathbf{x})} [\rho(u(\mathbf{x}) - u(\mathbf{y}), \sigma_S) + \rho(v(\mathbf{x}) - v(\mathbf{y}), \sigma_S)]$$

Objective function:

$$E(\mathbf{u}) = \sum_{\mathbf{x}} E_D(\mathbf{u}(\mathbf{x})) + \lambda E_S(\mathbf{u}(\mathbf{x}))$$

When ρ is quadratic = “Horn and Schunck”

Black

Optimization

$$u^{(n+1)} = u^{(n)} - \omega \frac{1}{T(u)} \frac{\partial E}{\partial u}$$

$$v^{(n+1)} = v^{(n)} - \omega \frac{1}{T(v)} \frac{\partial E}{\partial v}$$

$$\frac{\partial E}{\partial u_s} = \psi(I_x u_s + I_u v_s + I_t, \sigma_D) I_x + \lambda \sum_{n \in G(s)} \psi(u_s - u_n, \sigma_S)$$

T(u)= max of second derivative

Black

Optimization

- Gradient descent
- Coarse-to-fine (pyramid)
- Deterministic annealing

Black

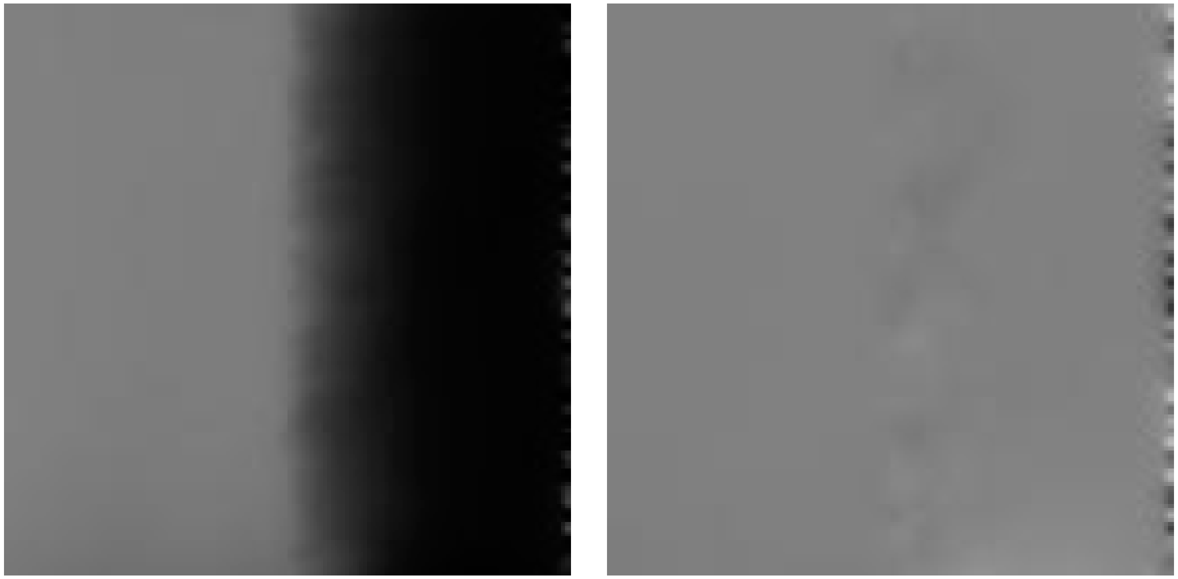
Example

Black

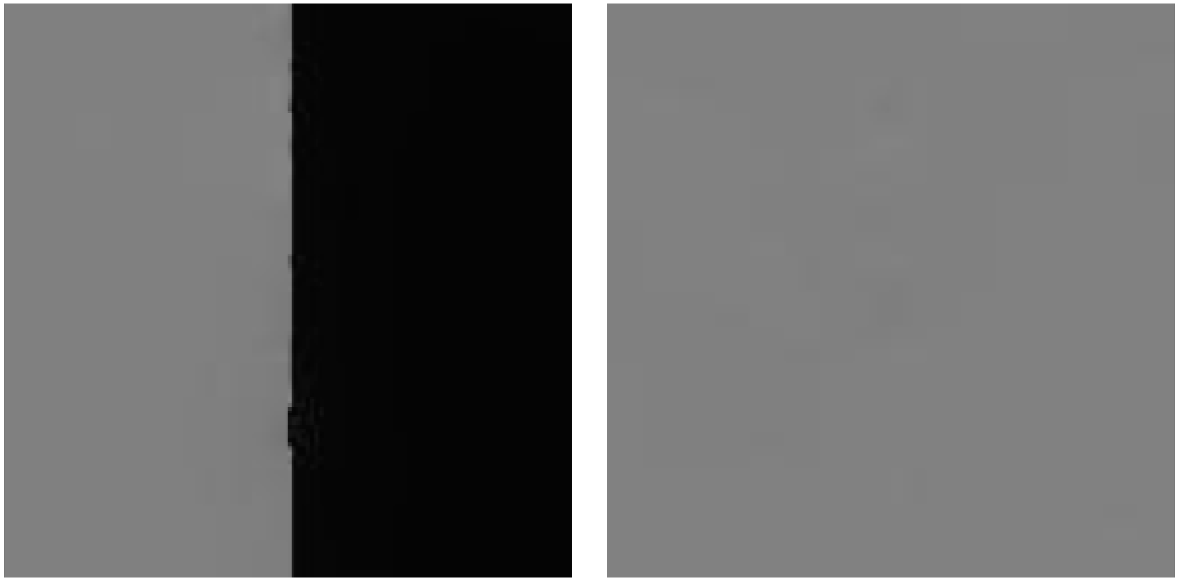
Example

Black

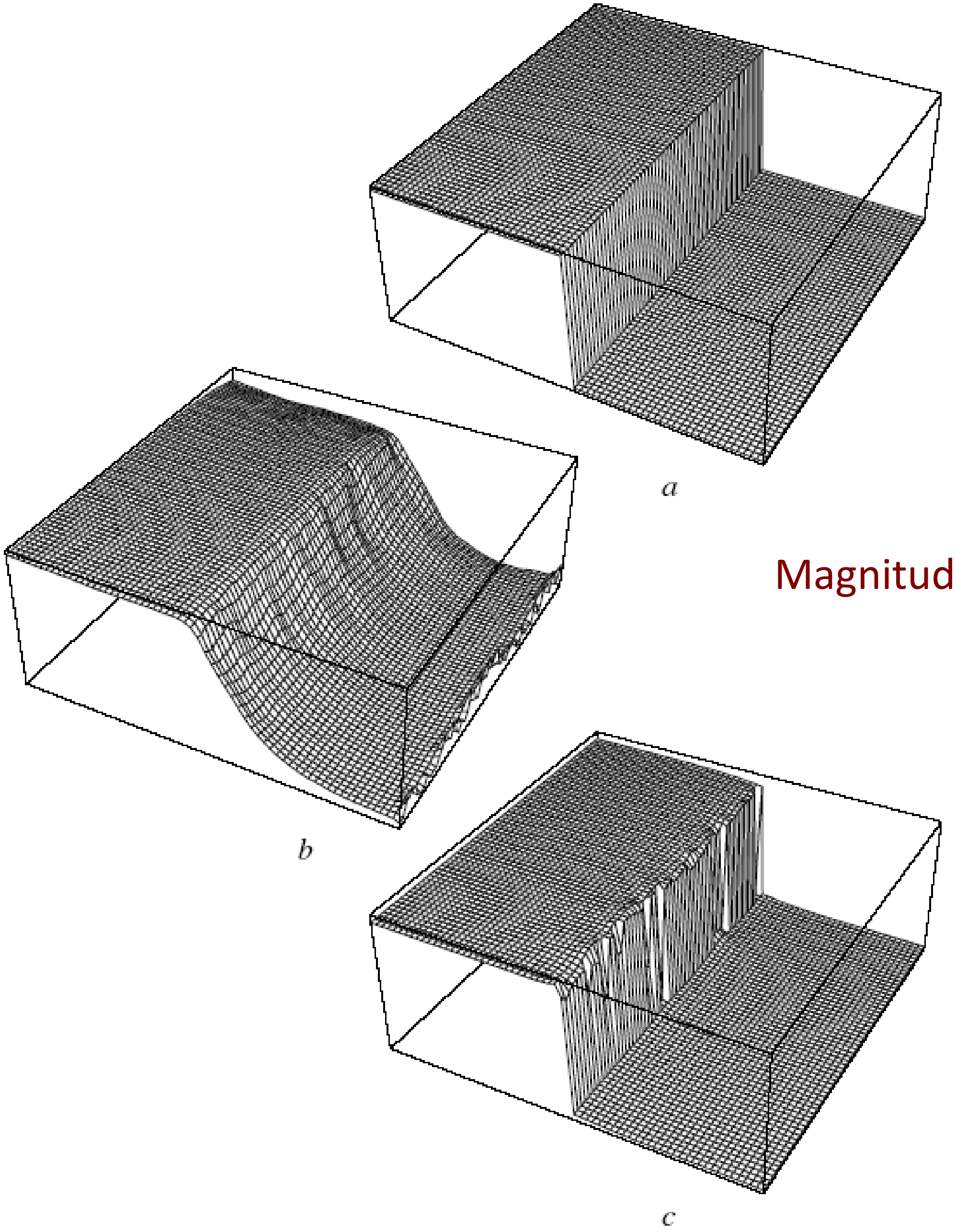
Quadratic:



Robust:



Black

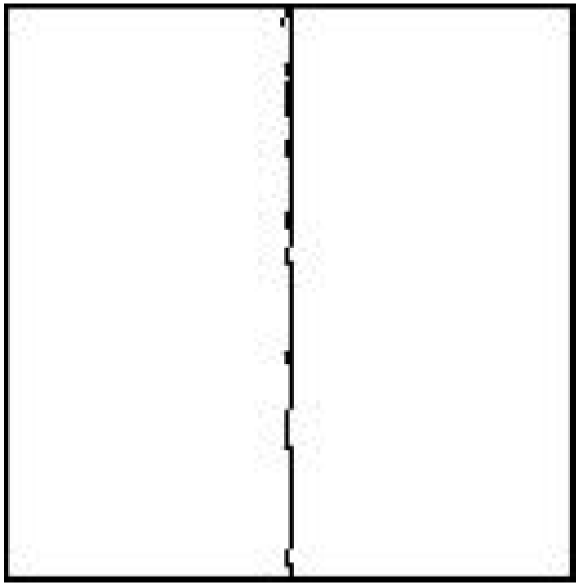


Magnitude of horizontal flow

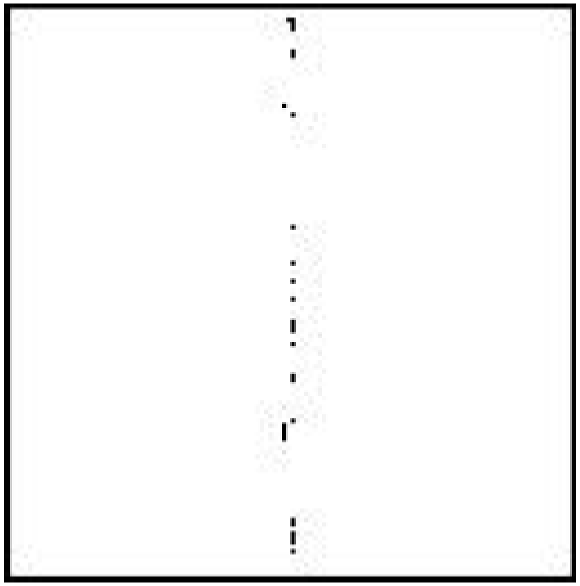
Black

Outliers

Points where the influence is reduced



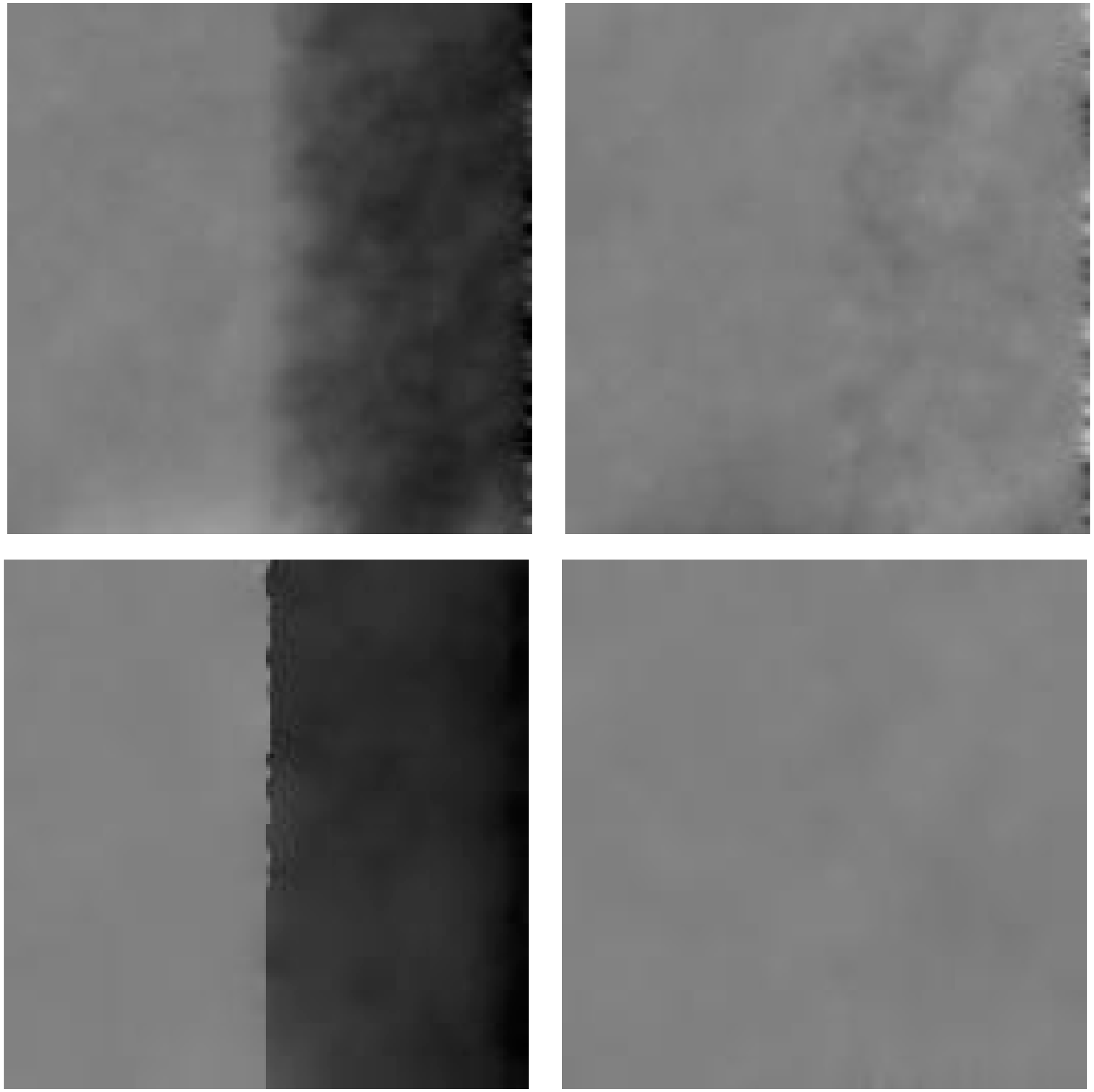
Spatial term



Data term

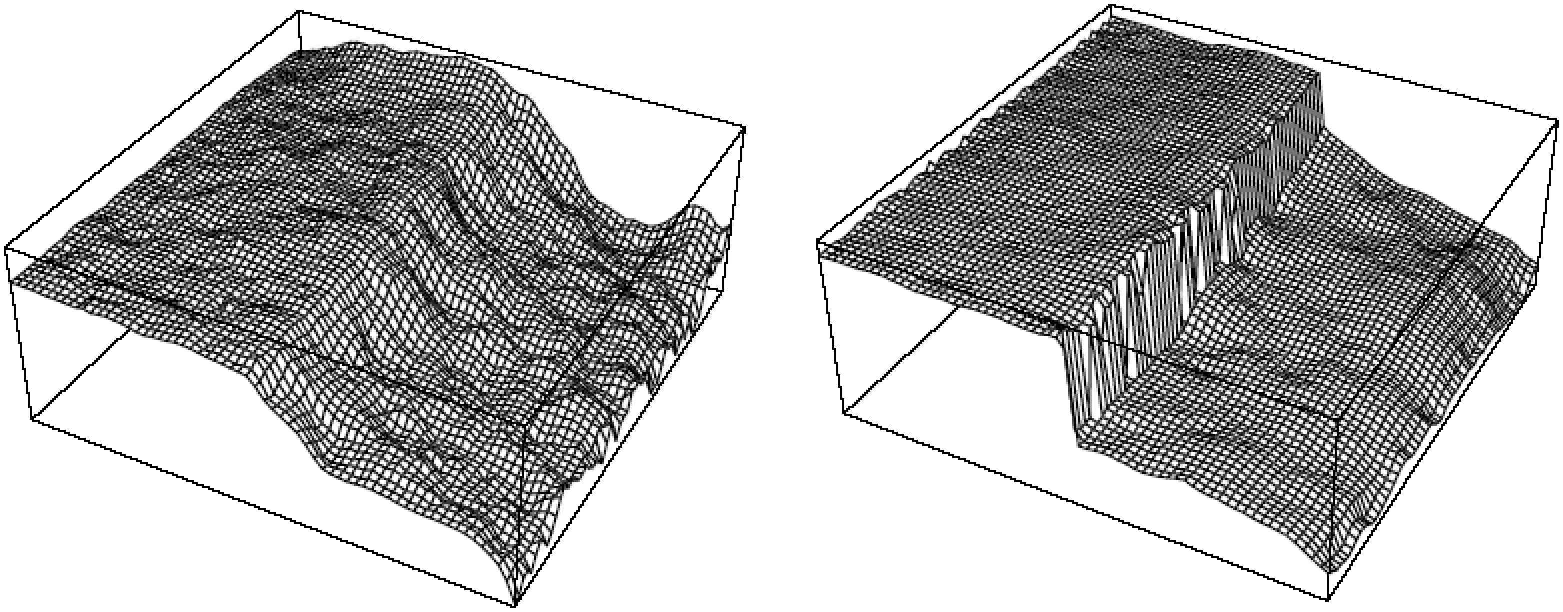
Black

With 5% uniform random noise added to the images.



Black

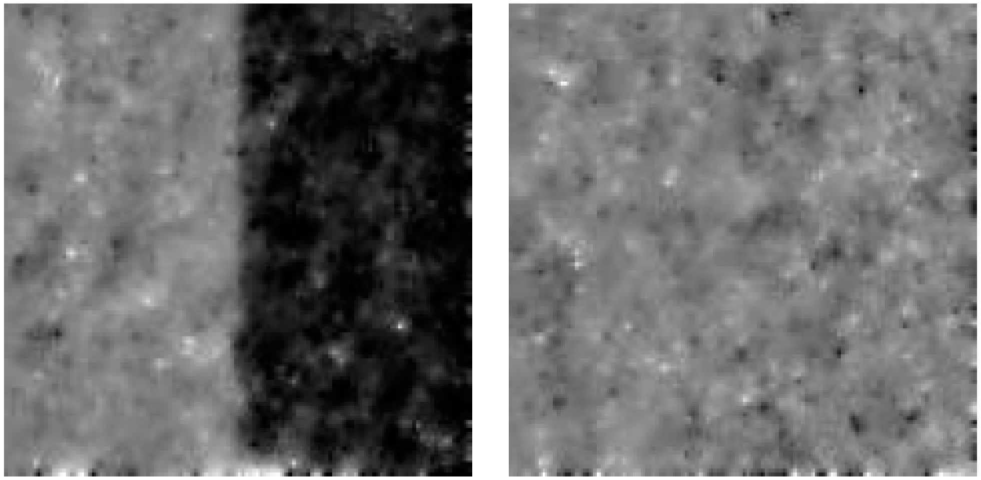
Horizontal Component



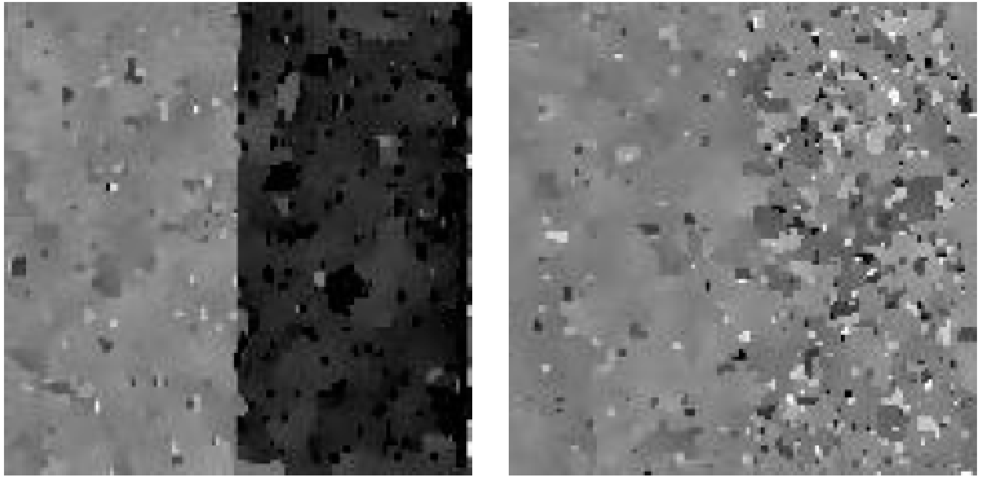
Black

More Noise

Quadratic:

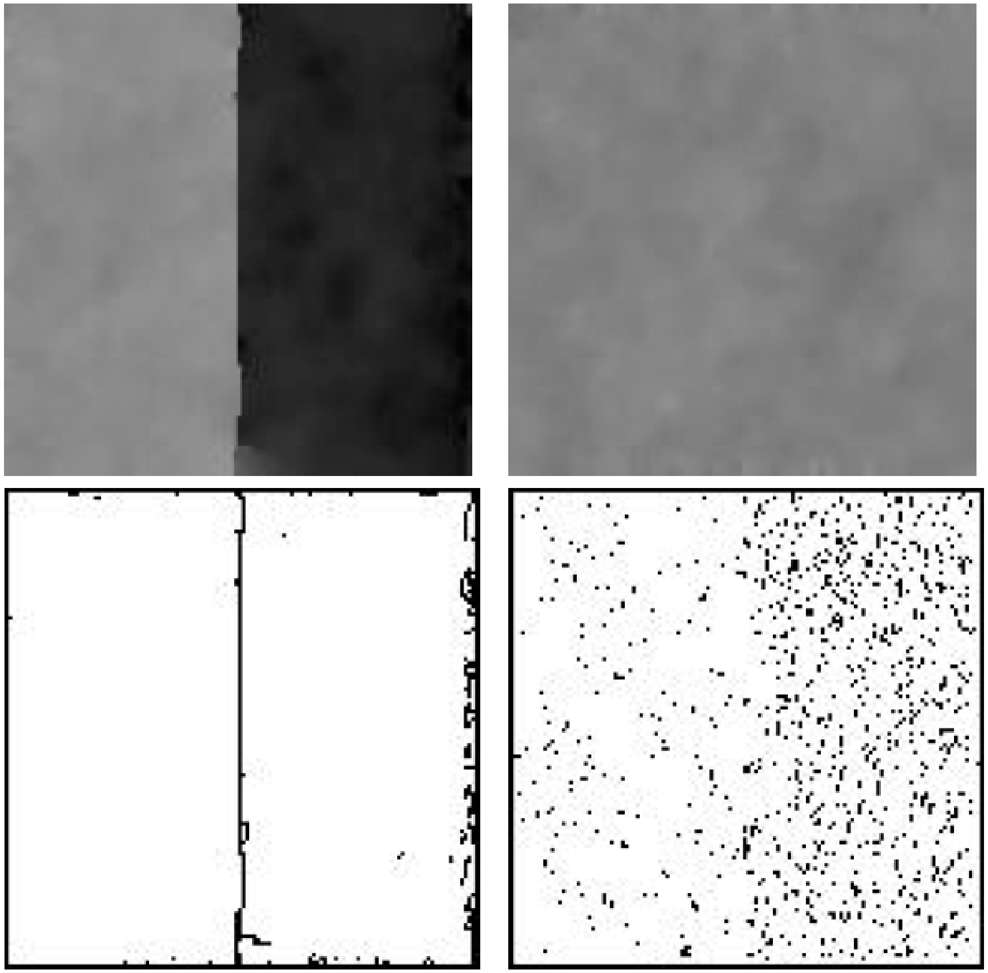


Quadratic data term, robust spatial term:



Black

Both Terms Robust



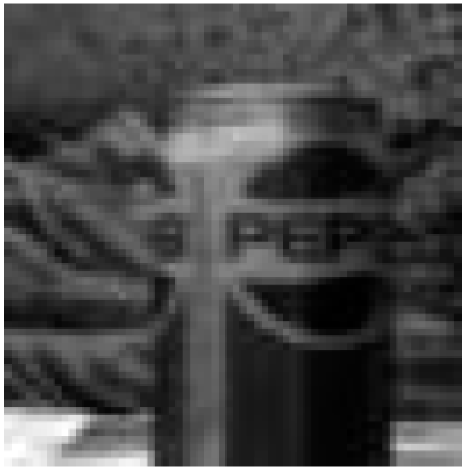
Spatial and
data outliers:

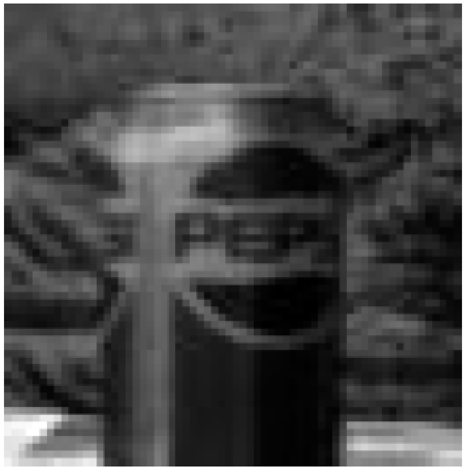
Black

Pepsi

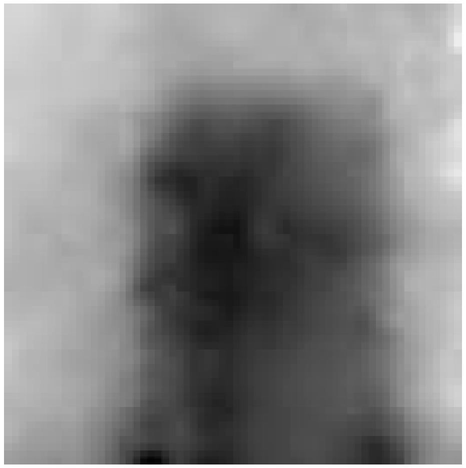
Black

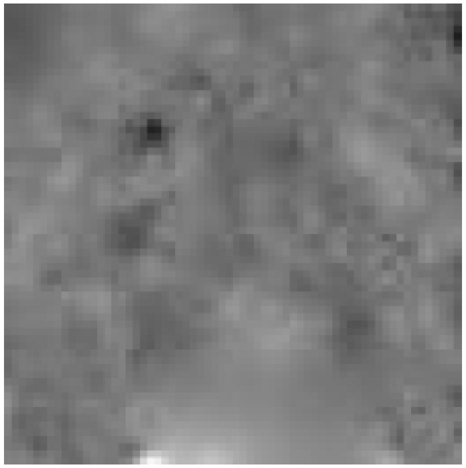
Real Sequence



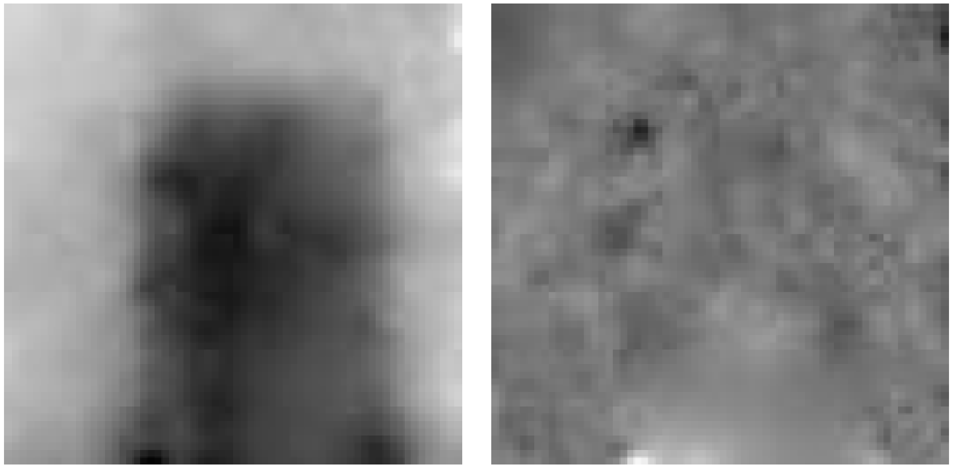


Deterministic annealing.
First stage (large σ):

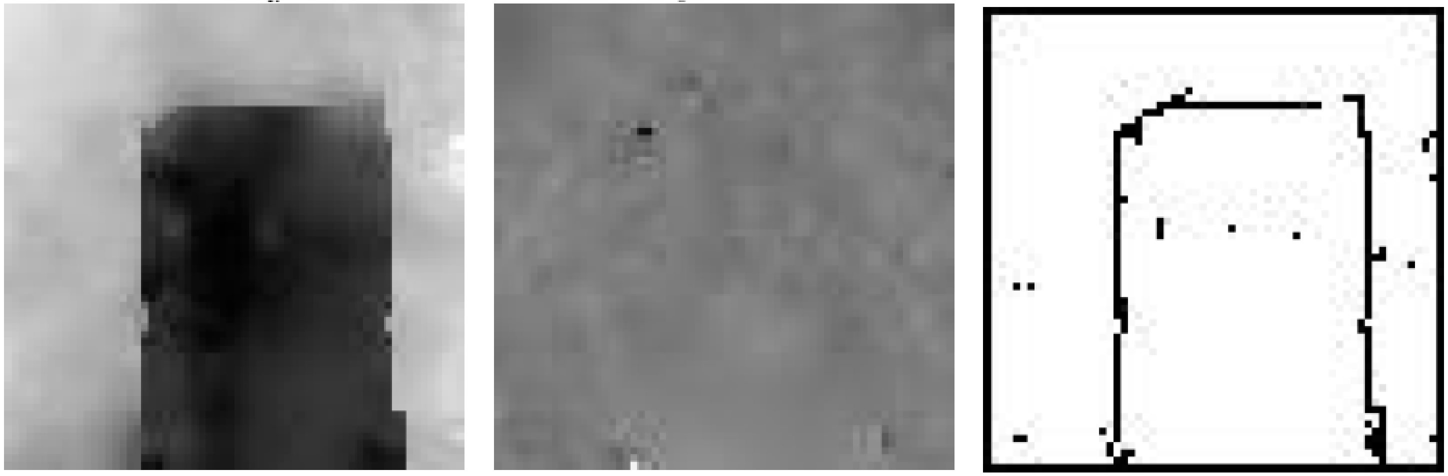




Real Sequence



Final result after annealing:



Black

Parametric motion estimation

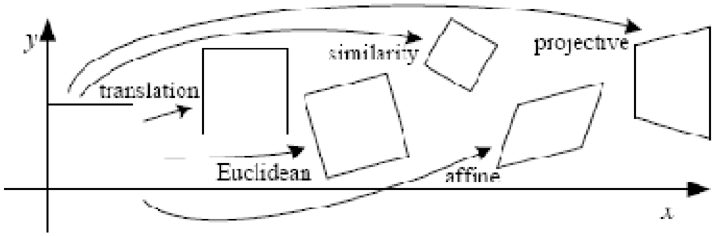
Global (parametric) motion models

- *2D Models:*
- Affine
- Quadratic
- Planar projective transform (Homography)

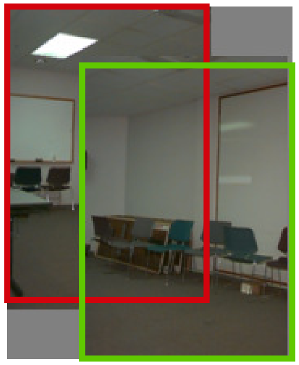
- *3D Models:*
- Instantaneous camera motion models
- Homography+epipole
- Plane+Parallax

Szeliski

Motion models

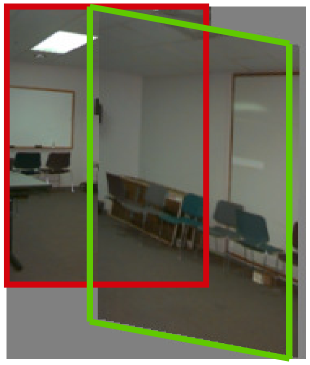


Translation



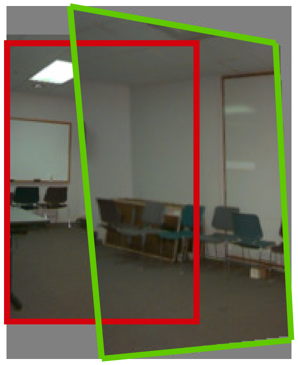
2 unknowns

Affine



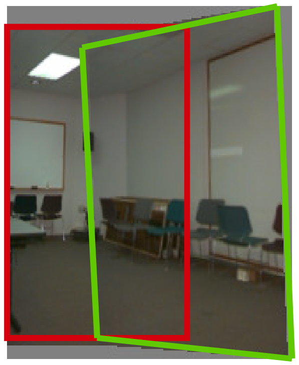
6 unknowns

Perspective



8 unknowns

3D rotation



3 unknowns

Szeliski

Example: 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 B.C. Equation:

$$\boxed{I_x \cdot u + I_y \cdot v + I_t \approx 0}$$

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

Each pixel provides 1 linear constraint in 6 *global* unknowns

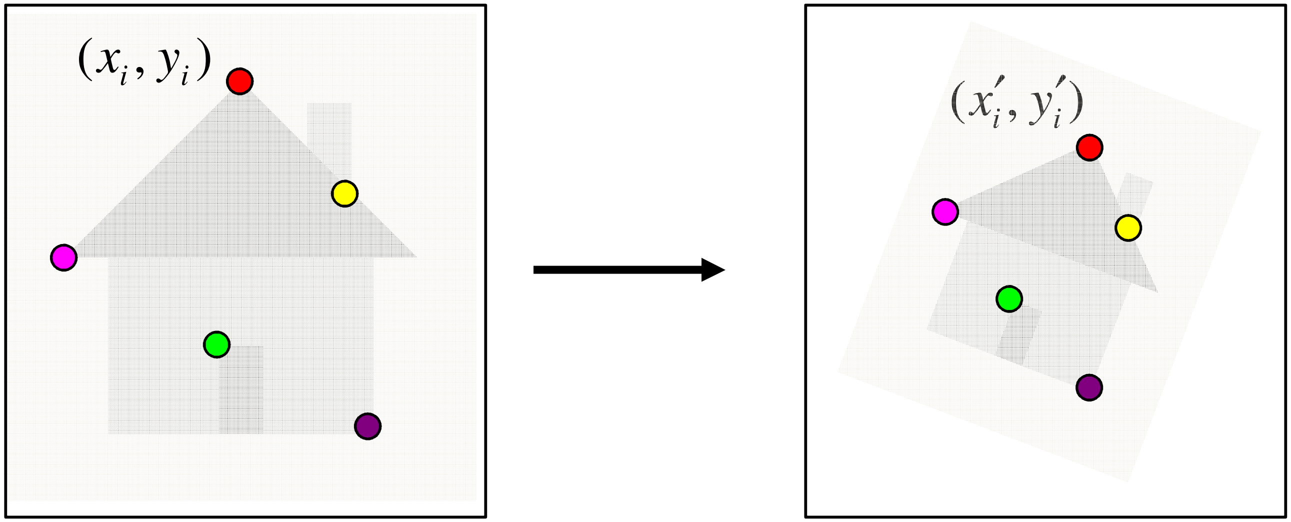
Least Square Minimization (over all pixels):

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

Szeliski

Last lecture: Alignment / motion warping

- “Alignment”: Assuming we know the correspondences, how do we get the transformation?



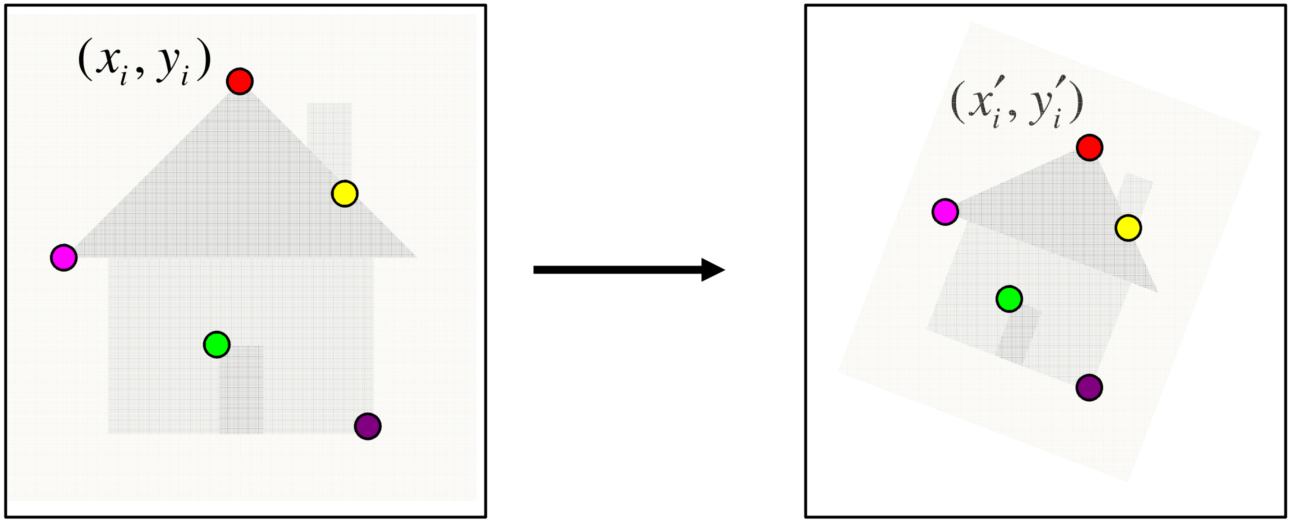
e.g., affine model in abs. coords…

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

- ***Expressed in terms of absolute coordinates of corresponding points…***
- ***Generally presumed features separately detected in each frame***

Today: “flow”, “parametric motion”

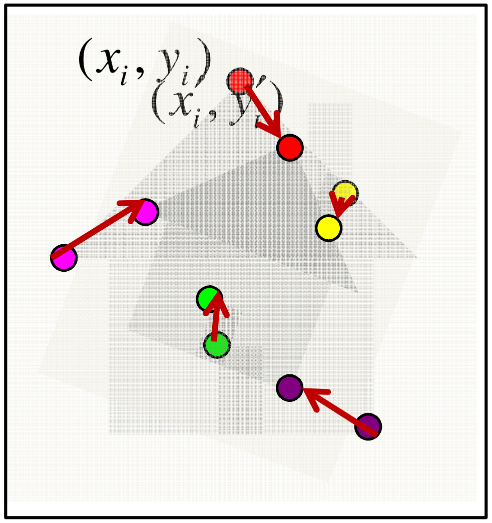
- Two views presumed in temporal sequence…**track** or analyze **spatio-temporal gradient**



- ***Sparse or dense in first frame***
- ***Search in second frame***
- ***Motion models expressed in terms of position change***

Today: “flow”, “parametric motion”

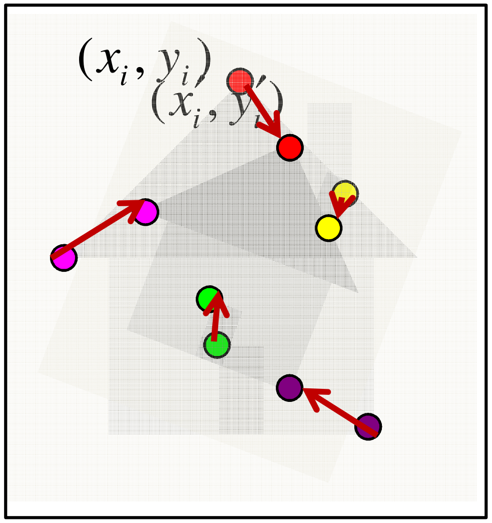
- Two views presumed in temporal sequence…**track** or analyze **spatio-temporal gradient**



- *Sparse or dense in first frame*
- *Search in second frame*
- *Motion models expressed in terms of position change*

Today: “flow”, “parametric motion”

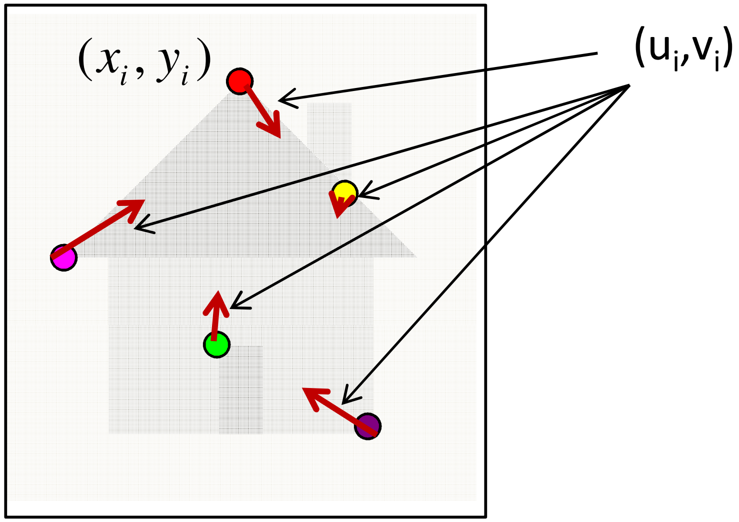
- Two views presumed in temporal sequence…**track** or analyze **spatio-temporal gradient**



- ***Sparse or dense in first frame***
- ***Search in second frame***
- ***Motion models expressed in terms of position change***

Today: “flow”, “parametric motion”

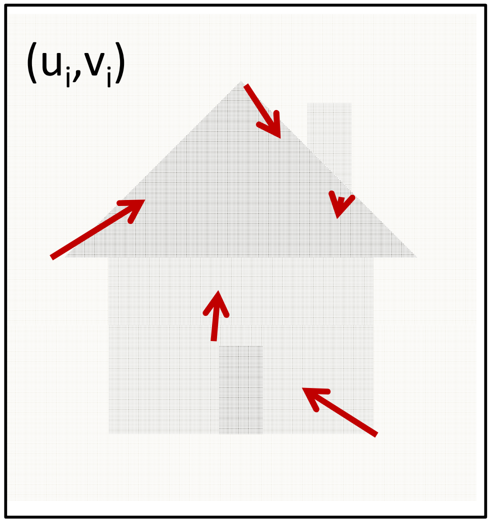
- Two views presumed in temporal sequence…**track** or analyze **spatio-temporal gradient**



- *Sparse or dense in first frame*
- *Search in second frame*
- *Motion models expressed in terms of position change*

Today: “flow”, “parametric motion”

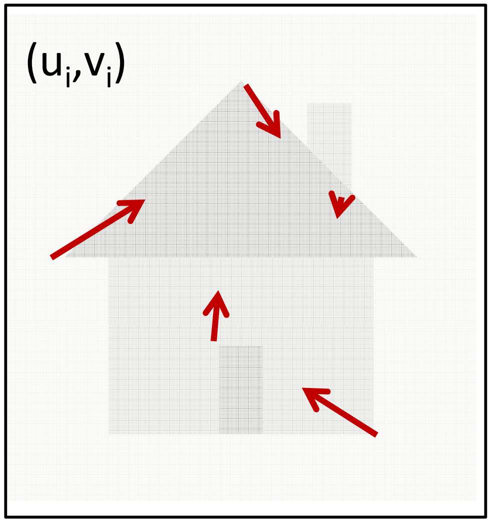
- Two views presumed in temporal sequence…**track** or analyze **spatio-temporal gradient**



- ***Sparse or dense in first frame***
- ***Search in second frame***
- ***Motion models expressed in terms of position change***

Today: “flow”, “parametric motion”

- Two views presumed in temporal sequence…**track** or analyze **spatio-temporal gradient**



Previous Alignment model:

$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

Now, Displacement model:

$$\begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} a_2 & a_3 \\ a_5 & a_6 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} a_1 \\ a_4 \end{bmatrix}$$

$$u(x, y) = a_1 + a_2 x + a_3 y$$

$$v(x, y) = a_4 + a_5 x + a_6 y$$

- ***Sparse or dense in first frame***
- ***Search in second frame***
- ***Motion models expressed in terms of position change***

Other 2D Motion Models

Quadratic – instantaneous approximation to planar motion

$$u = q_1 + q_2 x + q_3 y + q_7 x^2 + q_8 xy$$
$$v = q_4 + q_5 x + q_6 y + q_7 xy + q_8 y^2$$

Projective – exact planar motion

$$x' = \frac{h_1 + h_2 x + h_3 y}{h_7 + h_8 x + h_9 y}$$
$$y' = \frac{h_4 + h_5 x + h_6 y}{h_7 + h_8 x + h_9 y}$$

and

$$u = x' - x, \quad v = y' - y$$

Szeliski

3D Motion Models

Instantaneous camera motion:

Global parameters: $\Omega_X, \Omega_Y, \Omega_Z, T_X, T_Y, T_Z$

Local Parameter: $Z(x, y)$

$$u = -xy\Omega_X + (1+x^2)\Omega_Y - y\Omega_Z + (T_X - T_Z x)/Z$$
$$v = -(1+y^2)\Omega_X + xy\Omega_Y - x\Omega_Z + (T_Y - T_Z x)/Z$$

Homography+Epipole

Global parameters: $h_1, \ldots, h_9, t_1, t_2, t_3$

Local Parameter: $\gamma(x, y)$

$$x' = \frac{h_1 x + h_2 y + h_3 + \gamma t_1}{h_7 x + h_8 y + h_9 + \gamma t_3}$$
$$y' = \frac{h_4 x + h_5 y + h_6 + \gamma t_1}{h_7 x + h_8 y + h_9 + \gamma t_3}$$
$$\text{and}: \quad u = x' - x, \quad v = y' - y$$

Residual Planar Parallax Motion

Global parameters: t_1, t_2, t_3

Local Parameter: $\gamma(x, y)$

$$u = x^w - x = \frac{\gamma}{1+\gamma t_3}(t_3 x - t_1)$$
$$v = y^w - x = \frac{\gamma}{1+\gamma t_3}(t_3 y - t_2)$$

Szeliski

Discrete Search vs. Gradient Based

- Consider image I translated by u_0, v_0

$$I_0(x, y) = I(x, y)$$
$$I_1(x + u_0, y + v_0) = I(x, y) + \eta_1(x, y)$$

$$\begin{aligned} E(u,v) &= \sum_{x,y} (I(x, y) - I_1(x + u, y + v))^2 \\ &= \sum_{x,y} (I(x, y) - I(x - u_0 + u, y - v_0 + v) - \eta_1(x, y))^2 \end{aligned}$$

- The discrete search method simply searches for the best estimate.
- The gradient method linearizes the intensity function and solves for the estimate

Szeliski

Correlation and SSD

- For larger displacements, do template matching
  - Define a small area around a pixel as the template
  - Match the template against each pixel within a search area in next image.
  - Use a match measure such as correlation, normalized correlation, or sum-of-squares difference
  - Choose the maximum (or minimum) as the match
  - Sub-pixel estimate (Lucas-Kanade)

Szeliski

Shi-Tomasi feature tracker

1. Find good features (min eigenvalue of 2×2 Hessian)
2. Use Lucas-Kanade to track with pure translation
3. Use affine registration with first feature patch
4. Terminate tracks whose dissimilarity gets too large
5. Start new tracks when needed

Szeliski

Tracking results

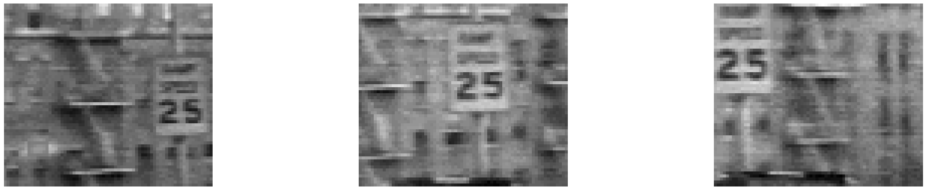


Figure 1: Three frame details from Woody Allen's *Manhattan*. The details are from the 1st, 11th, and 21st frames of a subsequence from the movie.

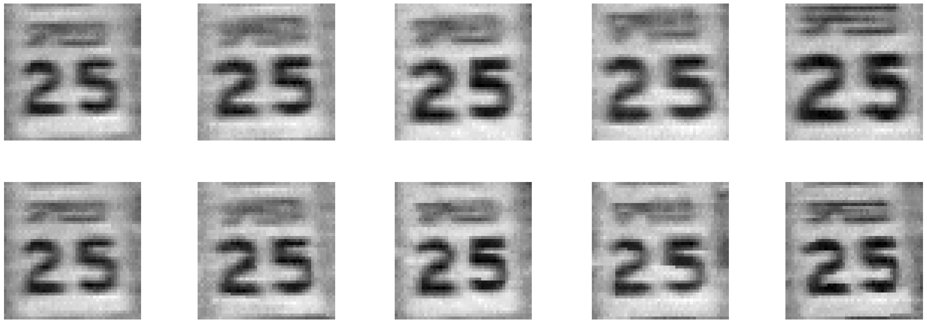


Figure 2: The traffic sign windows from frames 1,6,11,16,21 as tracked (top), and warped by the computed deformation matrices (bottom).

Szeliski

Tracking - dissimilarity

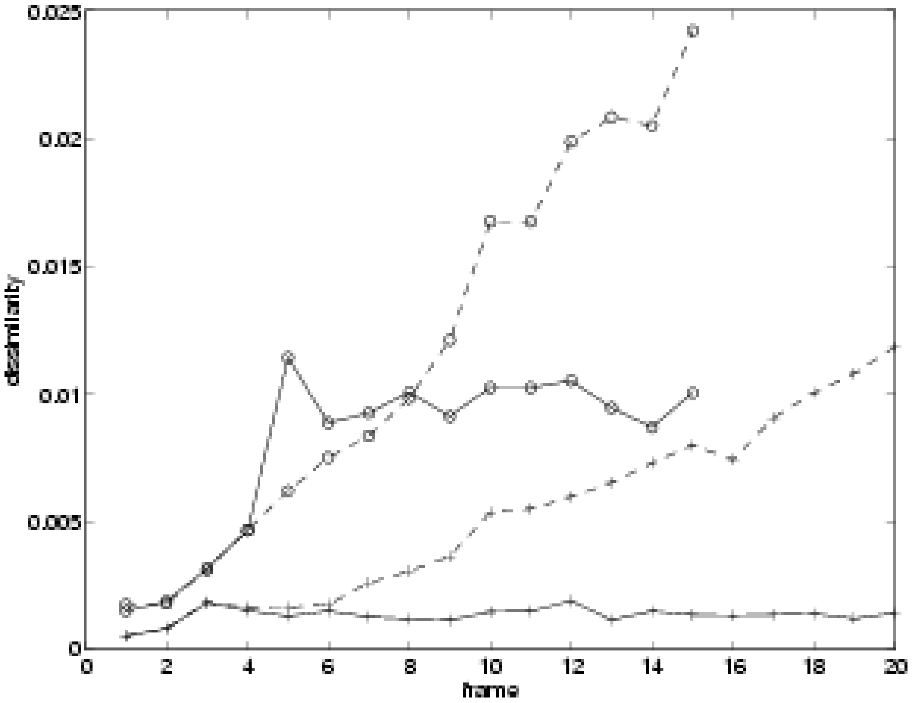


Figure 3: Pure translation (dashed) and affine motion (solid) dissimilarity measures for the window sequence of figure 1 (plusses) and 4 (circles).

Szeliski

Tracking results

Figure 13: Labels of some of the features in figure 11.

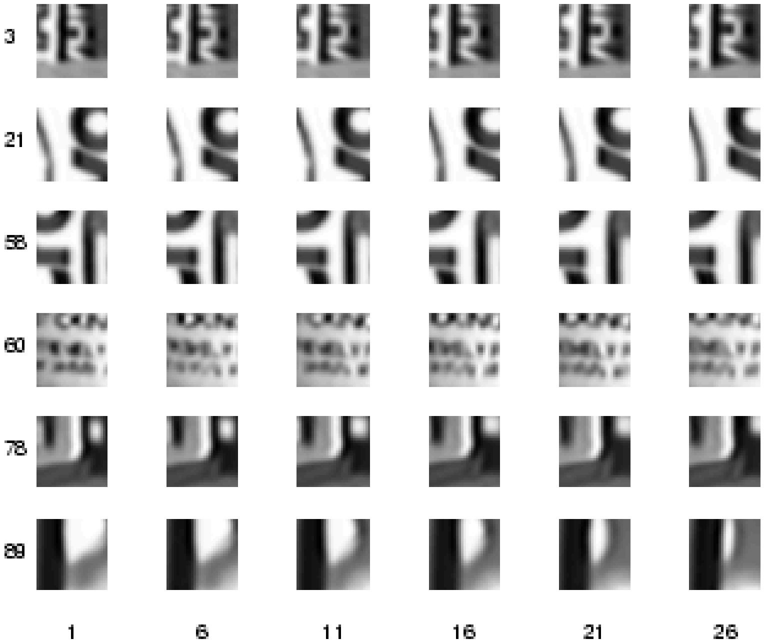


Figure 14: Six sample features through six sample frames.

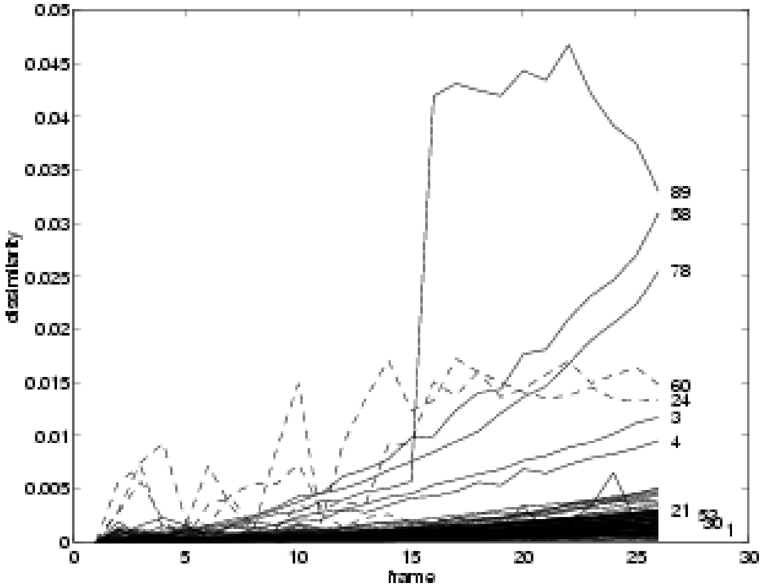


Figure 15: Affine motion dissimilarity for the features in figure 11. Notice the good discrimination between good and bad features. Dashed plots indicate aliasing (see text).

Features 24 and 60 deserve a special discussion, and

Szeliski

How well do these techniques work?

A Database and Evaluation Methodology for Optical Flow

Simon Baker, Daniel Scharstein, J.P Lewis, Stefan Roth, Michael Black, and Richard Szeliski

ICCV 2007

http://vision.middlebury.edu/flow/

Limitations of *Yosemite*

- Only sequence used for quantitative evaluation

Yosemite

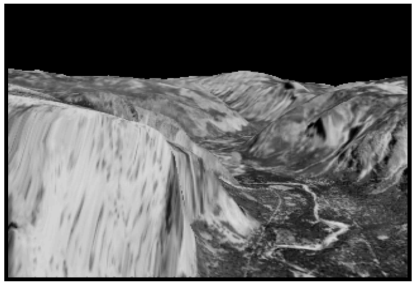


Image 7

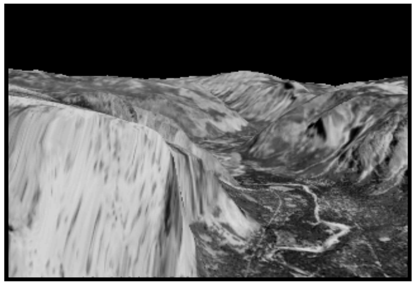
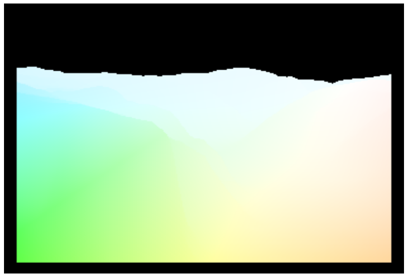
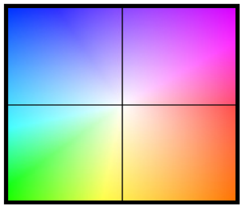


Image 8



Ground-Truth Flow



Flow Color Coding

- **Limitations:**
- Very simple and synthetic
- Small, rigid motion
- Minimal motion discontinuities/occlusions

Szeliski

Limitations of *Yosemite*

- Only sequence used for quantitative evaluation

Yosemite

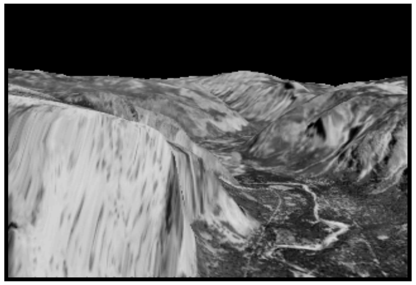


Image 7

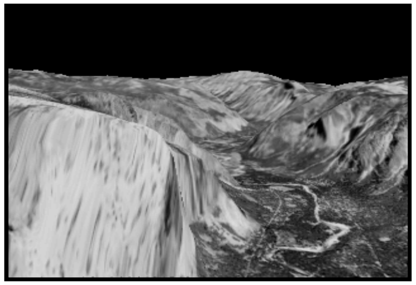
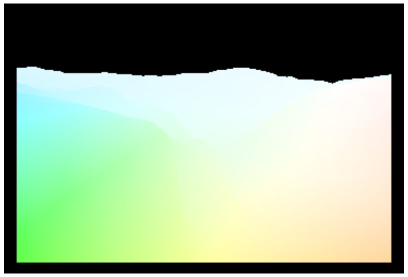
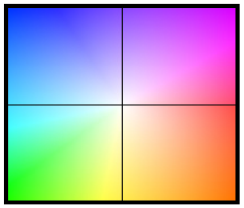


Image 8



Ground-Truth Flow



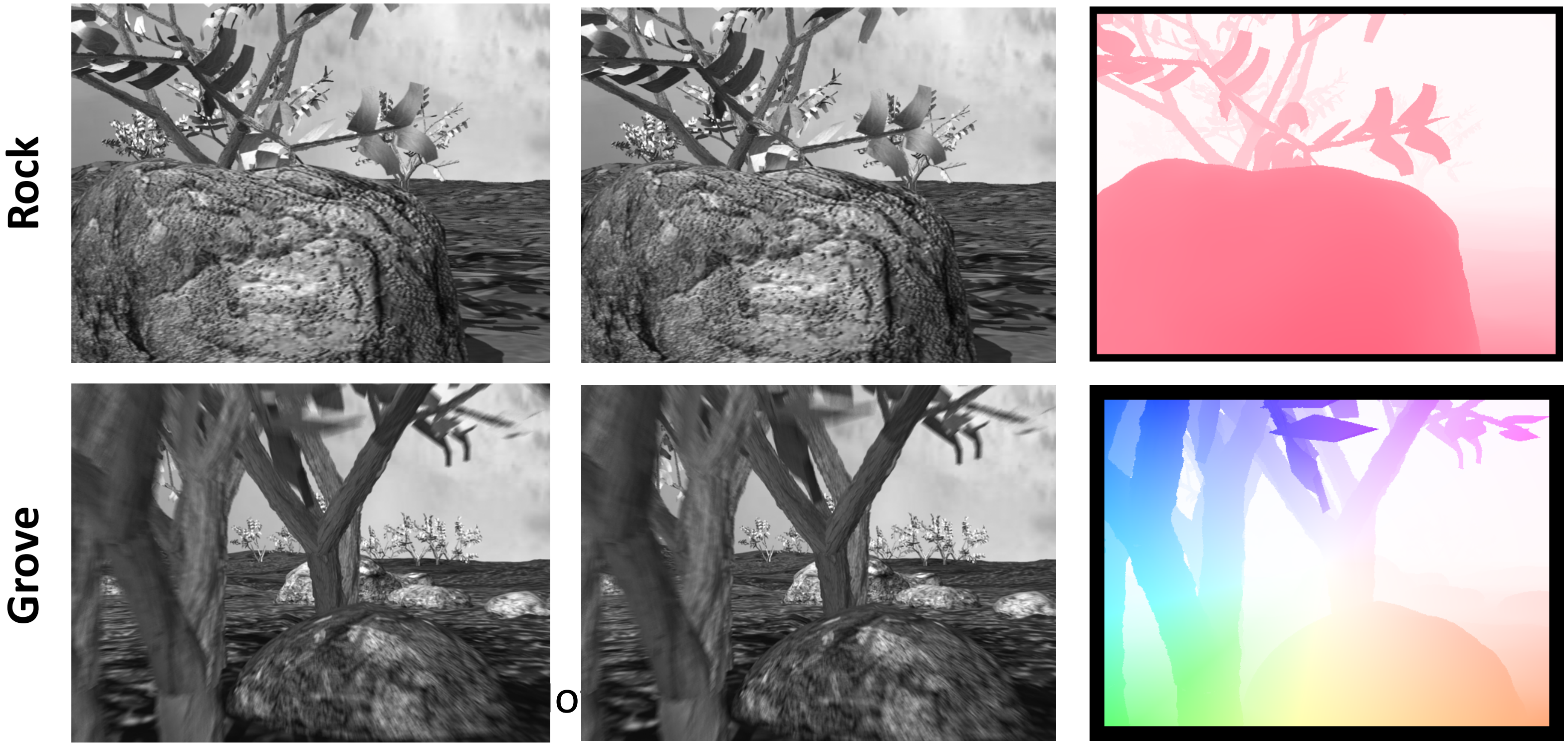
Flow Color Coding

- **Current challenges:**
- Non-rigid motion
- Real sensor noise
- Complex natural scenes
- Motion discontinuities
- Need **more challenging** and **more realistic** benchmarks

Szeliski

Realistic synthetic imagery

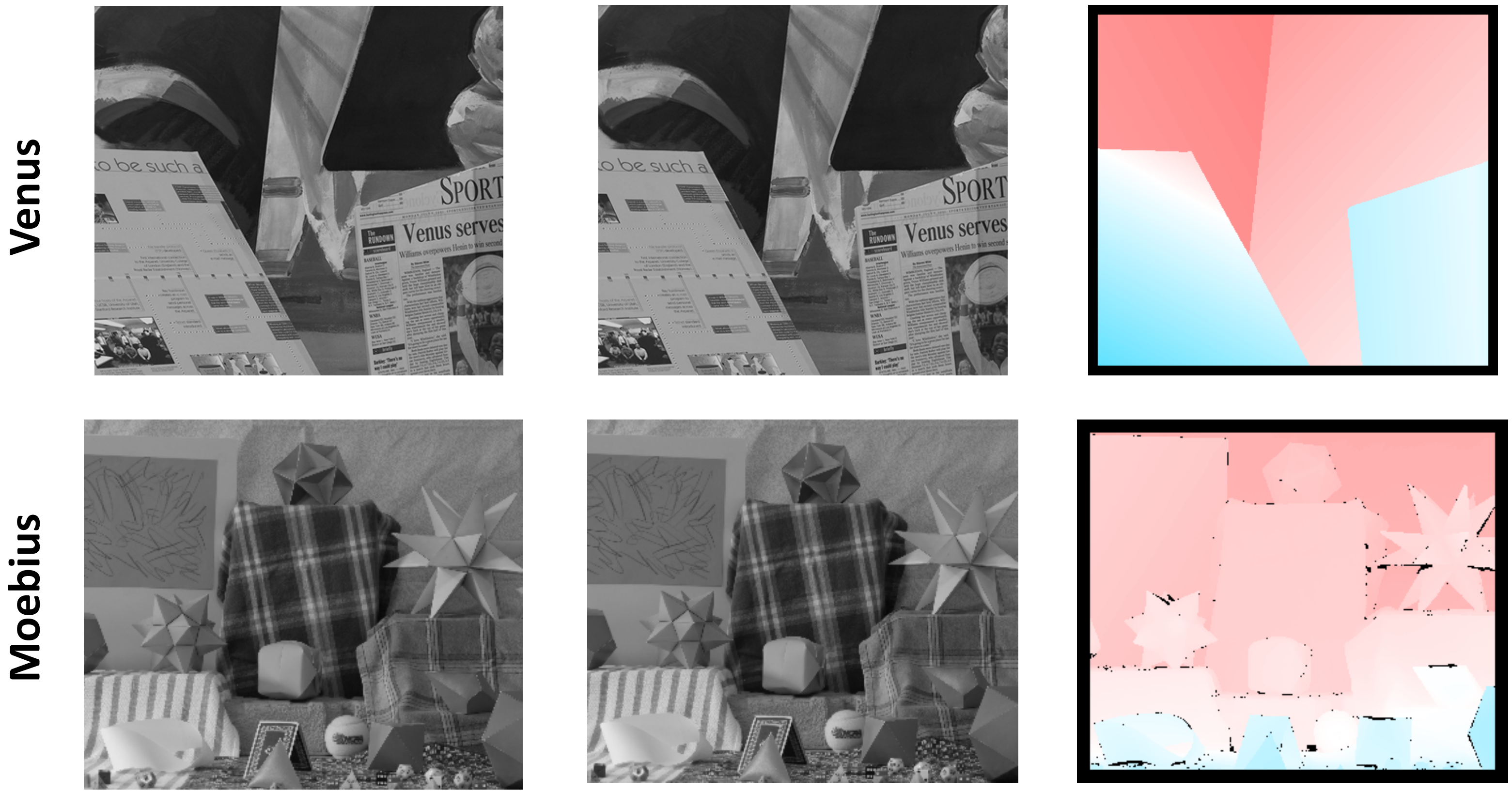
- Randomly generate scenes with “trees” and “rocks”
- Significant occlusions, motion, texture, and blur
- Rendered using Mental Ray and “lens shader” plugin



Szeliski

Modified stereo imagery

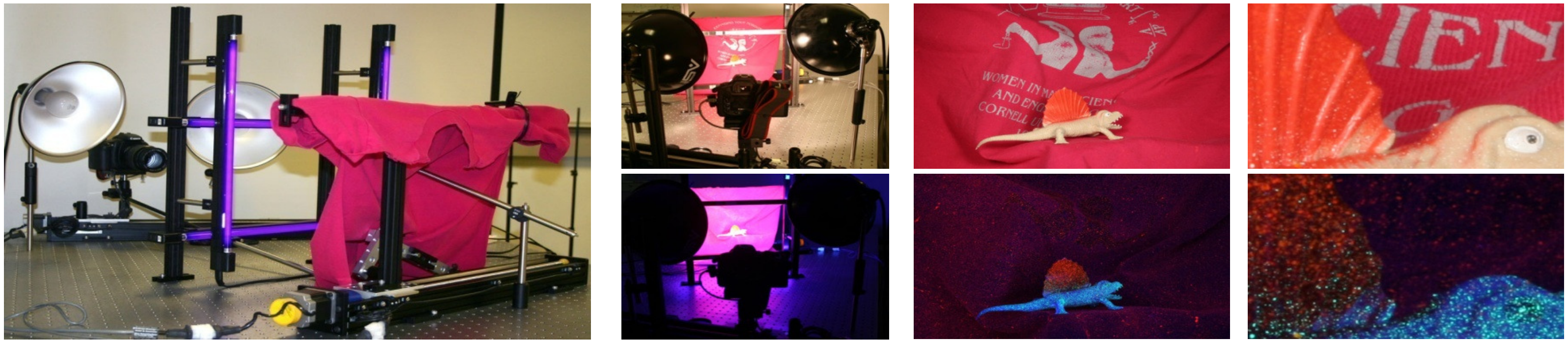
- Recrop and resample ground-truth stereo datasets to have appropriate motion for OF



Szeliski

Dense flow with hidden texture

- Paint scene with textured fluorescent paint
- Take 2 images: One in visible light, one in UV light
- Move scene in very small steps using robot
- Generate ground-truth by tracking the UV images



Visible

UV

Setup Lights Image Cropped

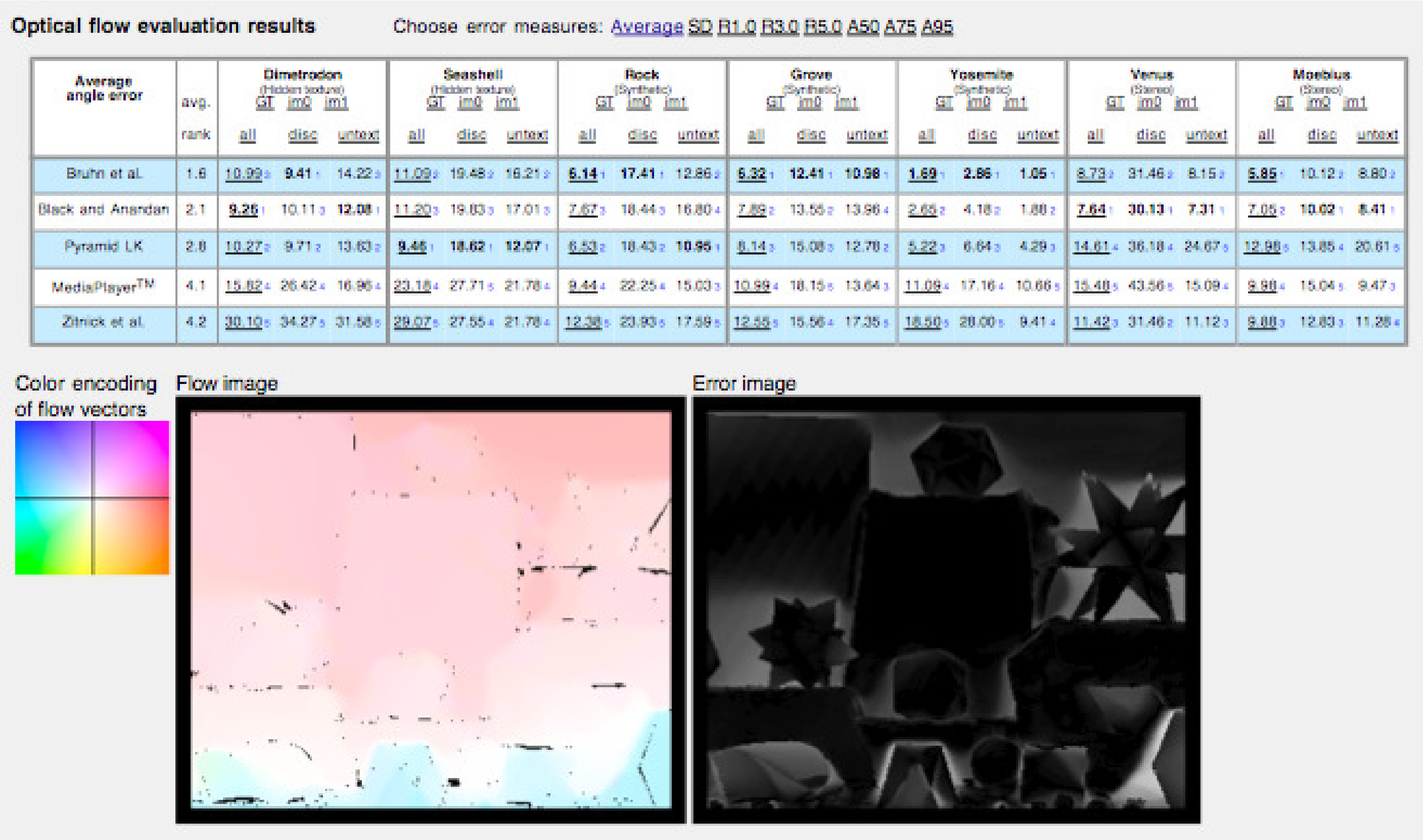
Szeliski

Experimental results

- **Algorithms:**
- **Pyramid LK:** OpenCV-based implementation of Lucas-Kanade on a Gaussian pyramid
- **Black and Anandan:** Author’s implementation
- **Bruhn *et al*.:** Our implementation
- **MediaPlayer$^{\mathrm{TM}}$:** Code used for video frame-rate upsampling in Microsoft MediaPlayer
- **Zitnick *et al*.:** Author’s implementation

Szeliski

Experimental results



Optical flow evaluation results

Choose error measures: Average SD R1.0 R3.0 R5.0 A50 A75 A95

Average angle error	avg.	Dimetrodon (Hidden texture) GT im0 im1			Seashell (Hidden texture) GT im0 im1			Rock (Synthetic) GT im0 im1			Grove (Synthetic) GT im0 im1			Yosemite (Synthetic) GT im0 im1			Venus (Stereo) GT im0 im1			Moebius (Stereo) GT im0 im1		
	rank	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext	all	disc	untext
Bruhn et al.	1.6	10.99 3	**9.41** 1	14.22 3	11.09 2	19.48 2	16.21 2	**6.14** 1	**17.41** 1	12.86 2	**6.32** 1	**12.41** 1	**10.98** 1	**1.69** 1	**2.86** 1	**1.05** 1	8.73 2	31.46 2	8.15 2	**5.85** 1	10.12 2	8.80 2
Black and Anandan	2.1	**9.26** 1	10.11 3	**12.08** 1	11.20 3	19.83 3	17.01 3	7.67 3	18.44 3	16.80 4	7.89 2	13.55 2	13.96 4	2.65 2	4.18 2	1.88 2	**7.64** 1	**30.13** 1	**7.31** 1	7.05 2	**10.02** 1	**8.41** 1
Pyramid LK	2.8	10.27 2	9.71 2	13.63 2	**9.46** 1	**18.62** 1	**12.07** 1	6.53 2	18.43 2	**10.95** 1	8.14 3	15.08 3	12.78 2	5.22 3	6.64 3	4.29 3	14.61 4	36.18 4	24.67 5	12.98 5	13.85 4	20.61 5
MediaPlayer^{TM}	4.1	15.82 4	26.42 4	16.96 4	23.18 4	27.71 5	21.78 4	9.44 4	22.25 4	15.03 3	10.99 4	18.15 5	13.64 3	11.09 4	17.16 4	10.66 5	15.48 5	43.56 5	15.09 4	9.88 4	15.04 5	9.47 3
Zitnick et al.	4.2	30.10 5	34.27 5	31.58 5	29.07 5	27.55 4	21.78 4	12.38 5	23.93 5	17.59 5	12.55 5	15.56 4	17.35 5	18.50 5	28.00 5	9.41 4	11.42 3	31.46 2	11.12 3	9.88 3	12.83 3	11.28 4

Szeliski

Conclusions

- **Difficulty:** Data substantially more challenging than **Yosemite**
- **Diversity: S**ubstantial variation in difficulty across the various datasets
- **Motion GT vs Interpolation:** Best algorithms for one are not the best for the other
- **Comparison with Stereo:** Performance of existing flow algorithms appears weak

Szeliski

Layered Scene Representations

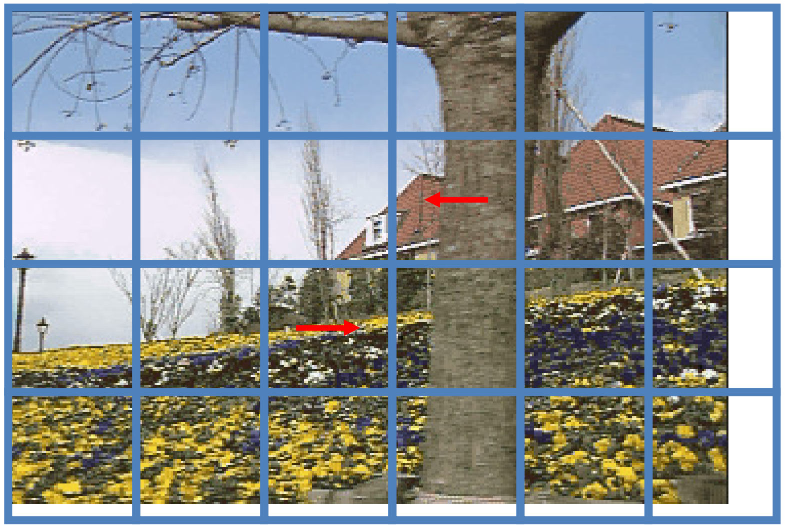
Motion representations

- How can we describe this scene?

Szeliski

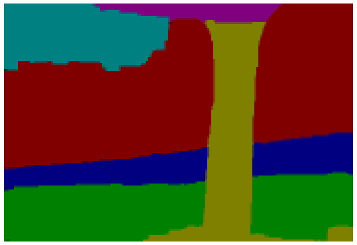
Block-based motion prediction

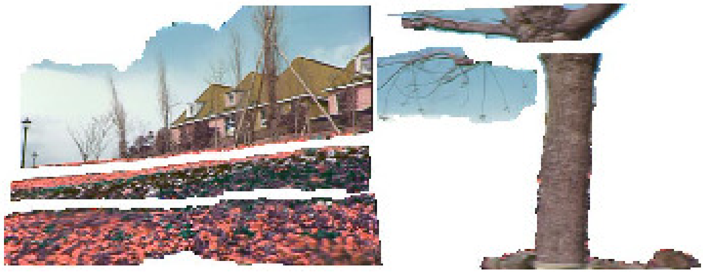
- Break image up into square blocks
- Estimate translation for each block
- Use this to predict next frame, code difference (MPEG-2)



Szeliski

Layered motion

- Break image sequence up into “layers”:
-  =



- Describe each layer’s motion

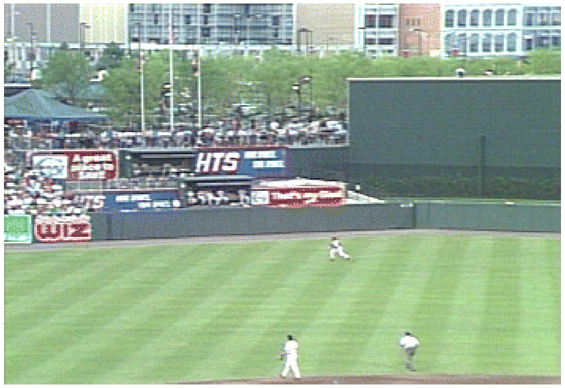
Szeliski

Layered motion

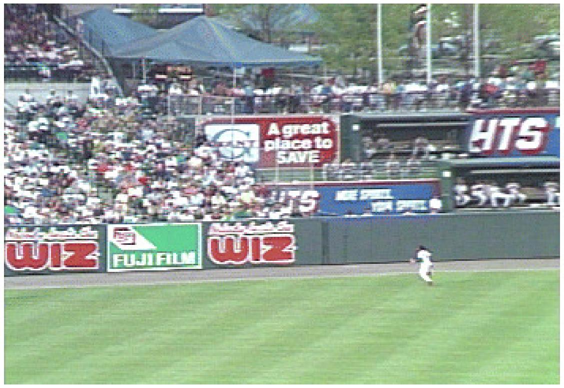
- Advantages:
- can represent occlusions / disocclusions
- each layer’s motion can be smooth
- video segmentation for semantic processing
- Difficulties:
- how do we determine the correct number?
- how do we assign pixels?
- how do we model the motion?

Szeliski

Layers for video summarization

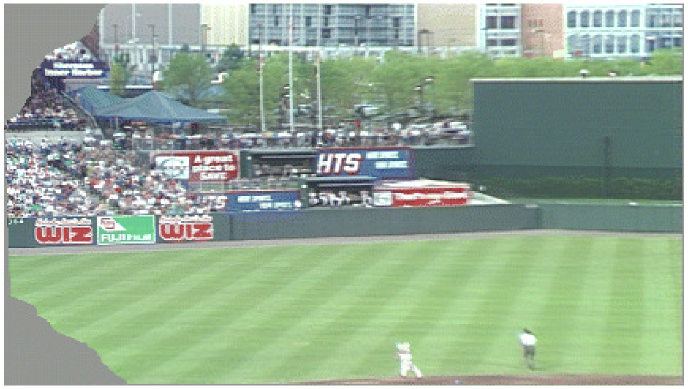


Frame 0

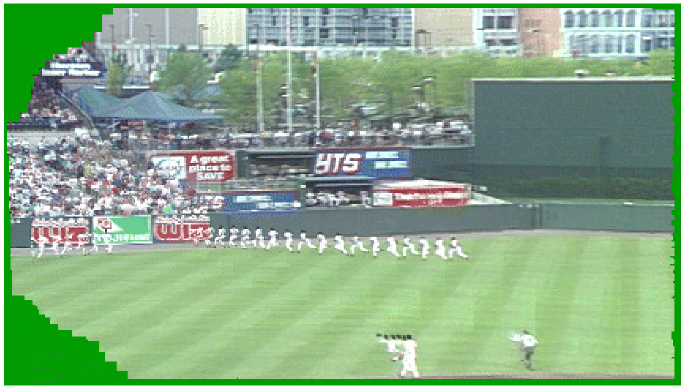


Frame 50

Frame 80



Background scene (players removed)

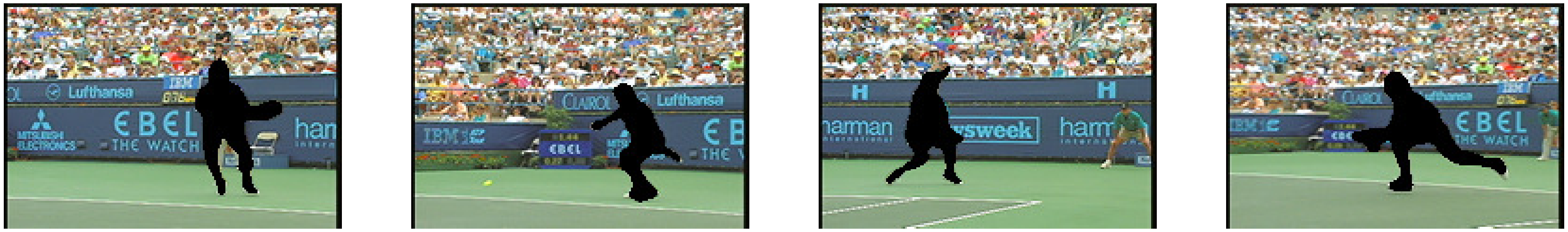


Complete synopsis of the video

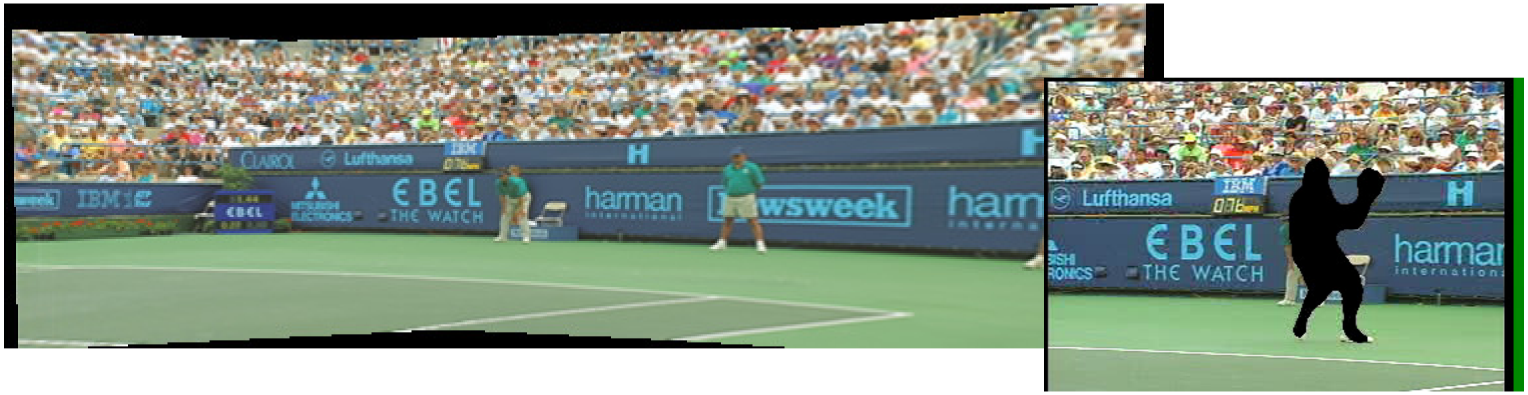
Szeliski

Background modeling (MPEG-4)

- Convert masked images into a background sprite for layered video coding

- 

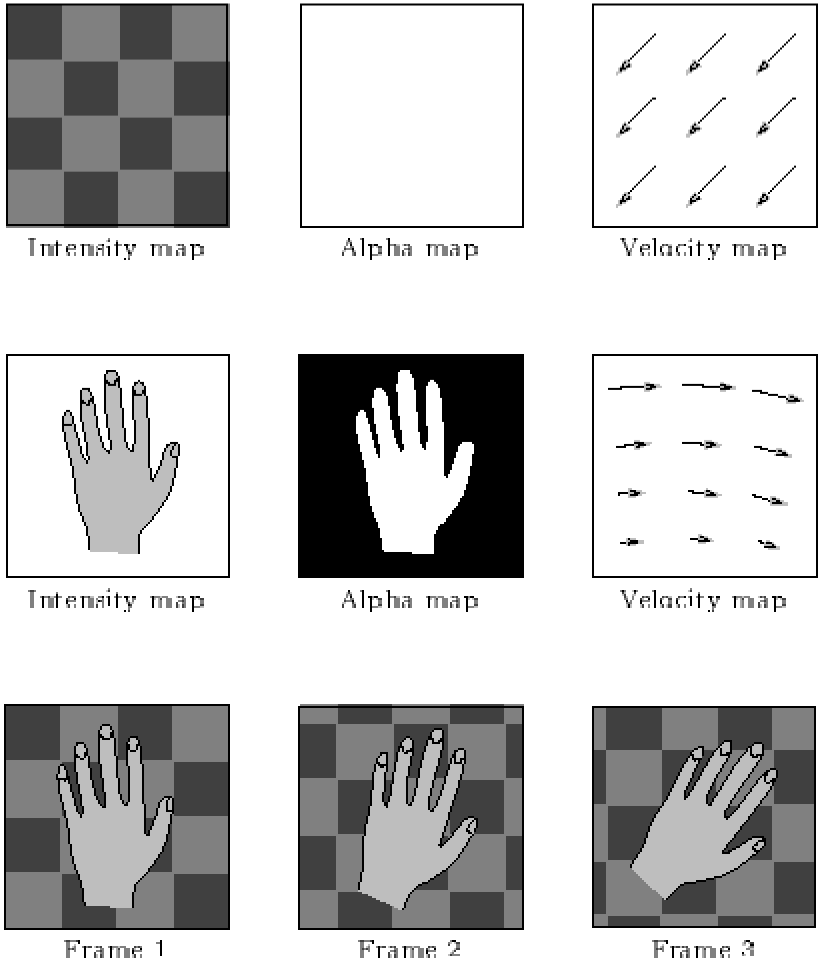
- =



Szeliski

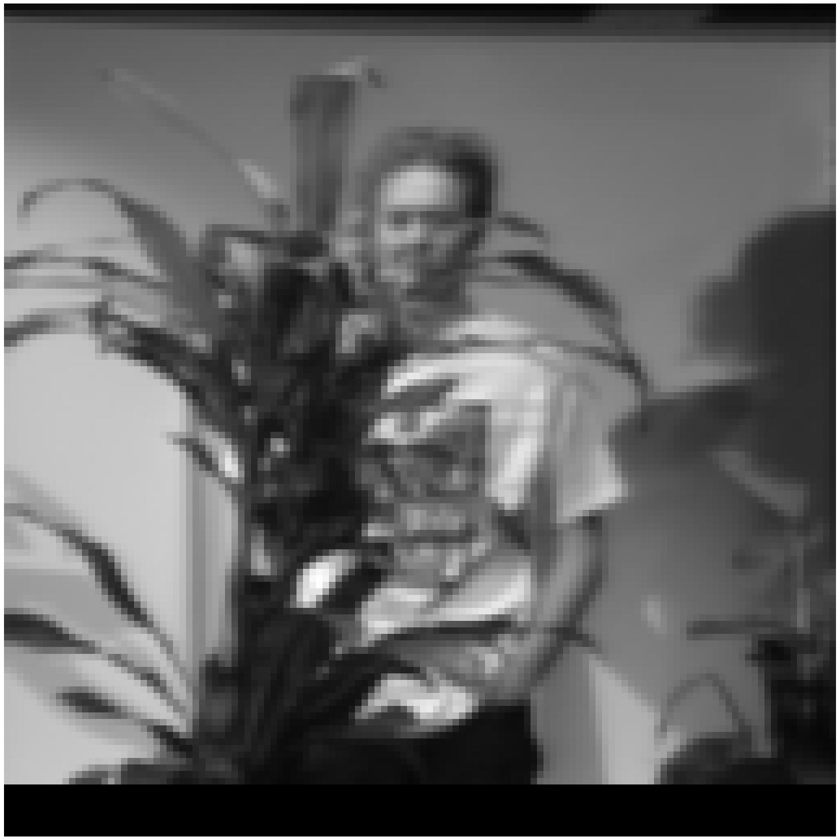
What are layers?

- [Wang & Adelson, 1994; Darrell & Pentland 1991]
- intensities
- alphas
- velocities

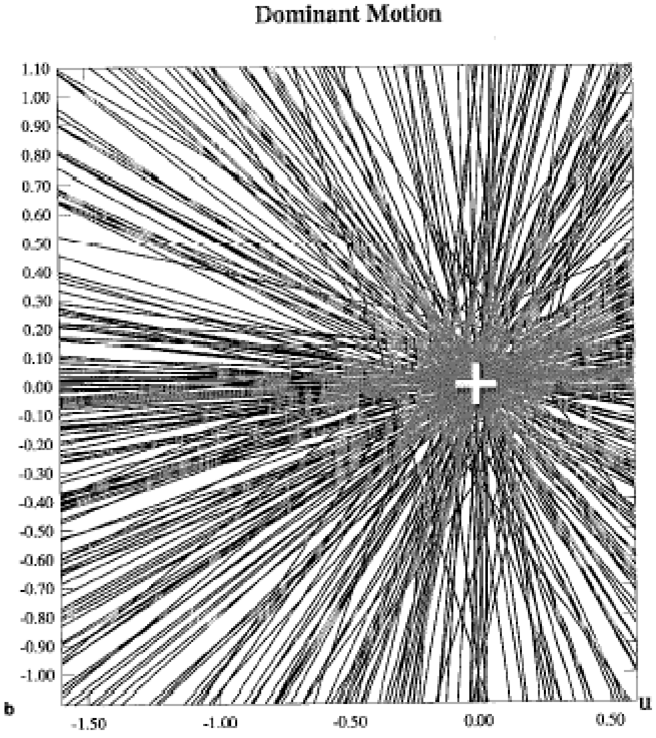


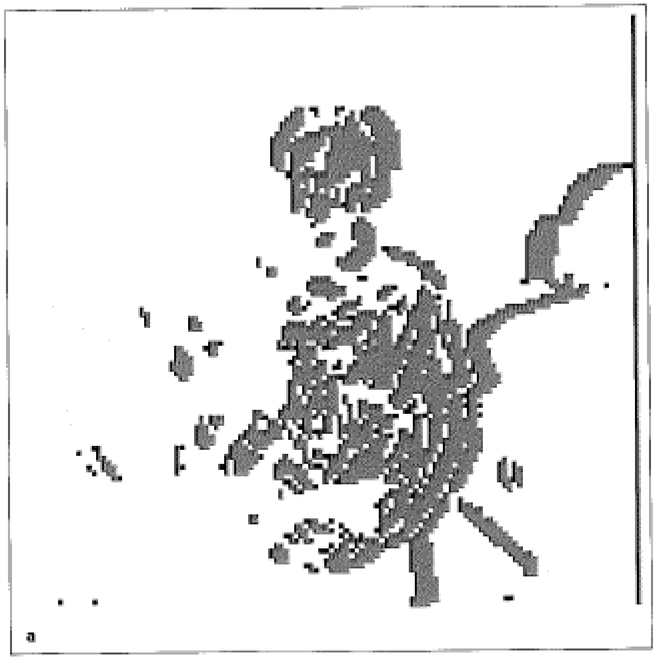
Szeliski

Fragmented Occlusion

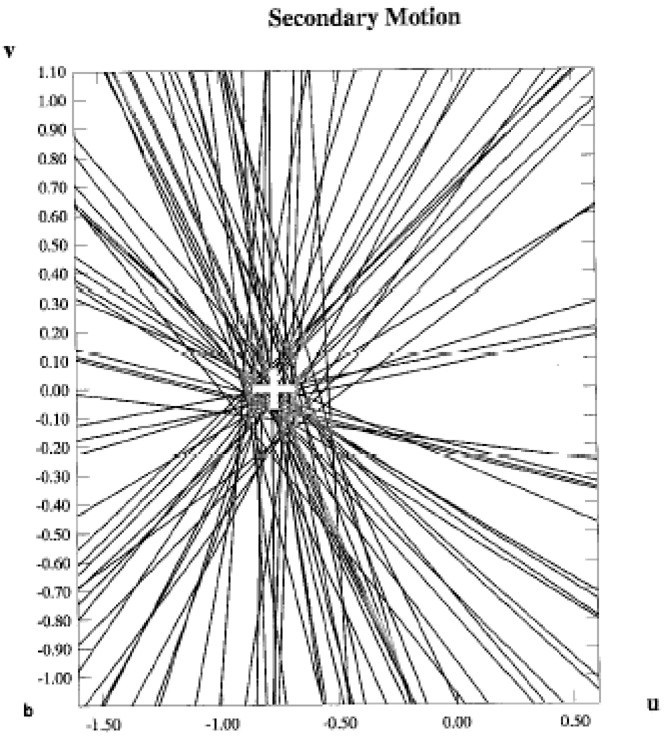


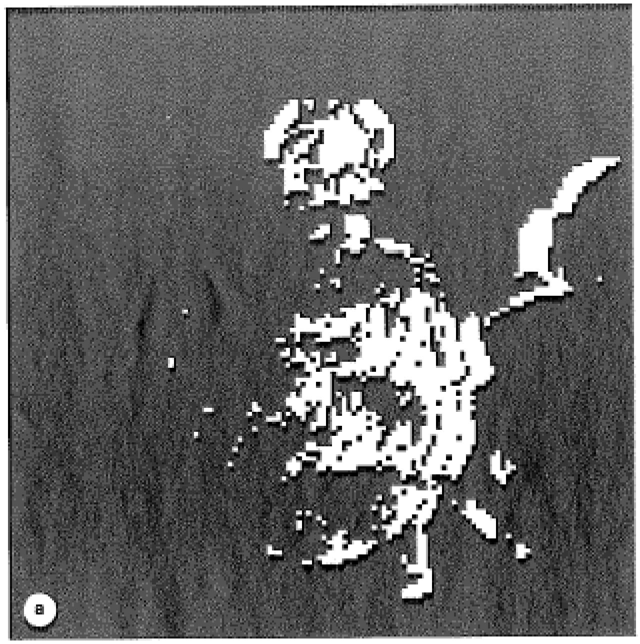
Results


Dominant Motion
v
1.10
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
-0.10
-0.20
-0.30
-0.40
-0.50
-0.60
-0.70
-0.80
-0.90
-1.00
b
u
-1.50
-1.00
-0.50
0.00
0.50

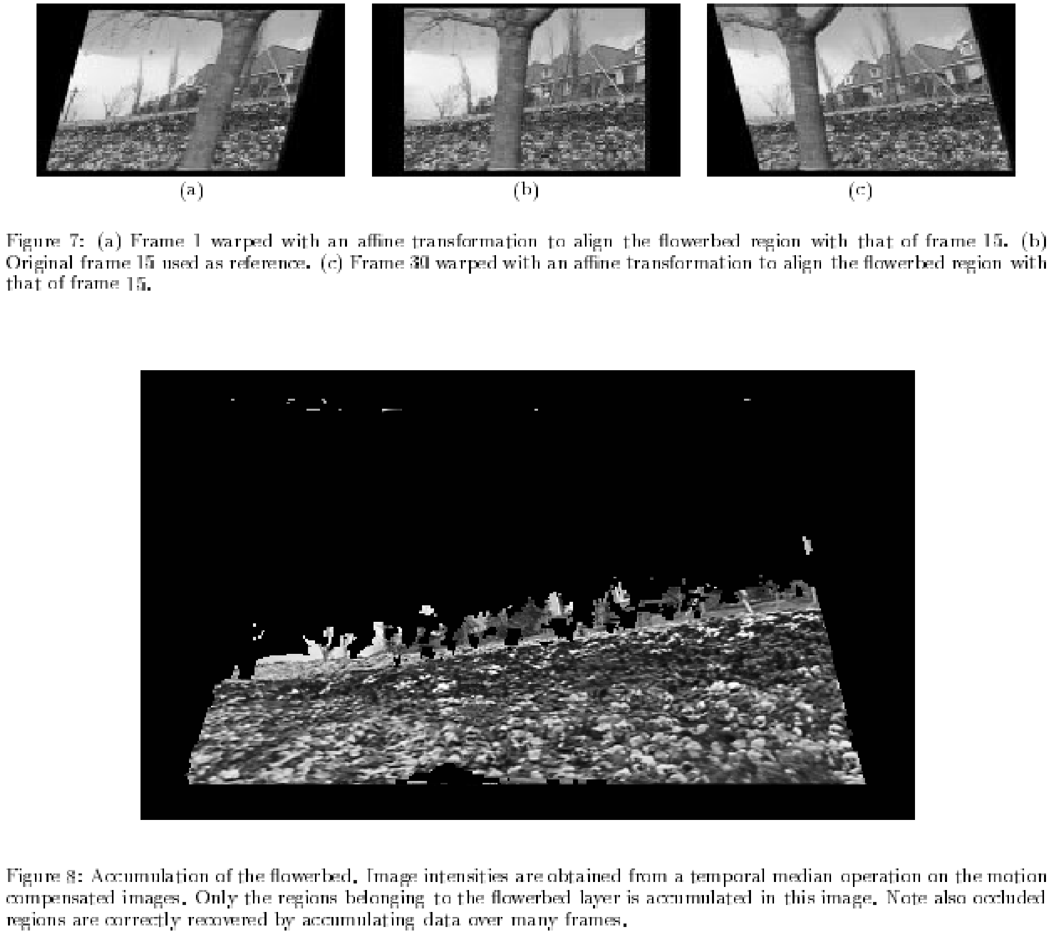

a

Results


Secondary Motion
v
1.10
1.00
0.90
0.80
0.70
0.60
0.50
0.40
0.30
0.20
0.10
0.00
-0.10
-0.20
-0.30
-0.40
-0.50
-0.60
-0.70
-0.80
-0.90
-1.00
b
-1.50
-1.00
-0.50
0.00
0.50
u


a

How do we form them?



(a) (b) (c)

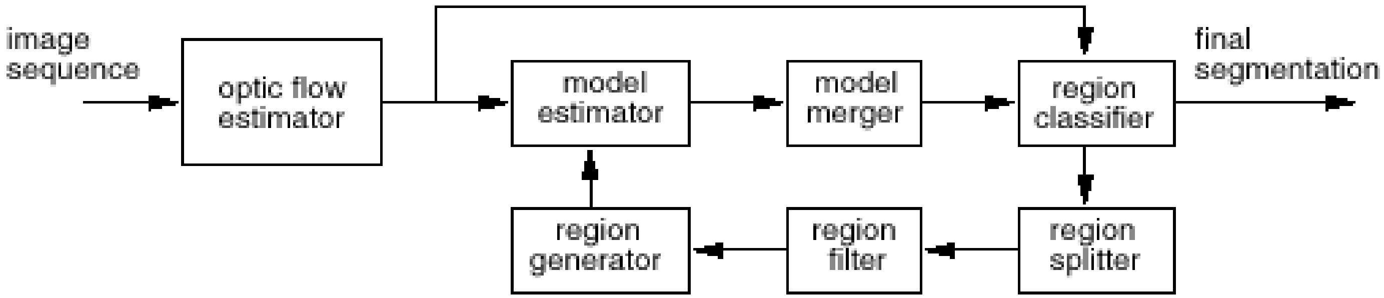
Figure 7: (a) Frame 1 warped with an affine transformation to align the flowerbed region with that of frame 15. (b) Original frame 15 used as reference. (c) Frame 30 warped with an affine transformation to align the flowerbed region with that of frame 15.

Figure 8: Accumulation of the flowerbed. Image intensities are obtained from a temporal median operation on the motion compensated images. Only the regions belonging to the flowerbed layer is accumulated in this image. Note also occluded regions are correctly recovered by accumulating data over many frames.

Szeliski

How do we estimate the layers?

1. compute coarse-to-fine flow
2. estimate affine motion in blocks (regression)
3. cluster with *k-means*
4. assign pixels to best fitting affine region
5. re-estimate affine motions in each region…



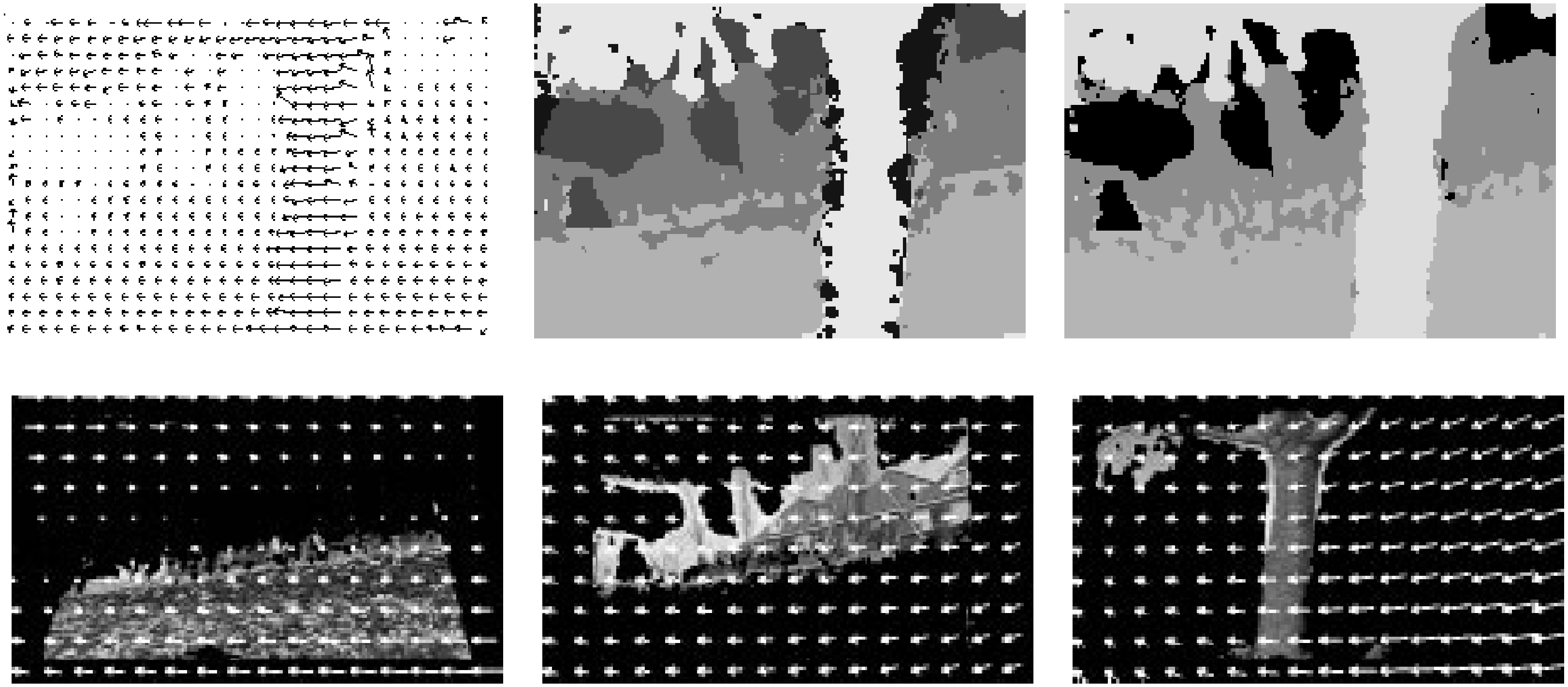
Szeliski

Layer synthesis

- For each layer:
- stabilize the sequence with the affine motion
- compute median value at each pixel
- Determine occlusion relationships

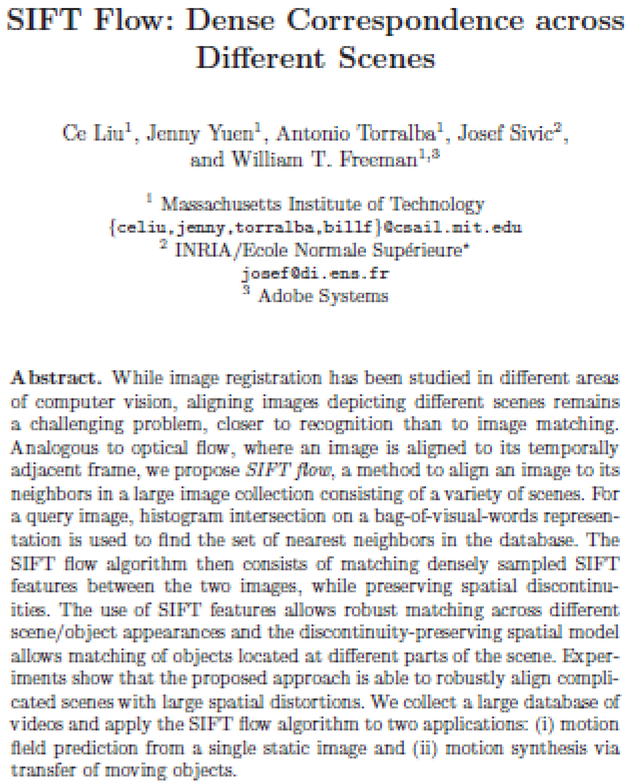
Szeliski

Result



Szeliski

Recent results: SIFT Flow



SIFT Flow: Dense Correspondence across Different Scenes

Ce Liu[1], Jenny Yuen[1], Antonio Torralba[1], Josef Sivic[2],
and William T. Freeman[1,3]

[1] Massachusetts Institute of Technology
{celiu,jenny,torralba,billf}@csail.mit.edu
[2] INRIA/Ecole Normale Supérieure*
josef@di.ens.fr
[3] Adobe Systems

Abstract. While image registration has been studied in different areas of computer vision, aligning images depicting different scenes remains a challenging problem, closer to recognition than to image matching. Analogous to optical flow, where an image is aligned to its temporally adjacent frame, we propose *SIFT flow*, a method to align an image to its neighbors in a large image collection consisting of a variety of scenes. For a query image, histogram intersection on a bag-of-visual-words representation is used to find the set of nearest neighbors in the database. The SIFT flow algorithm then consists of matching densely sampled SIFT features between the two images, while preserving spatial discontinuities. The use of SIFT features allows robust matching across different scene/object appearances and the discontinuity-preserving spatial model allows matching of objects located at different parts of the scene. Experiments show that the proposed approach is able to robustly align complicated scenes with large spatial distortions. We collect a large database of videos and apply the SIFT flow algorithm to two applications: (i) motion field prediction from a single static image and (ii) motion synthesis via transfer of moving objects.

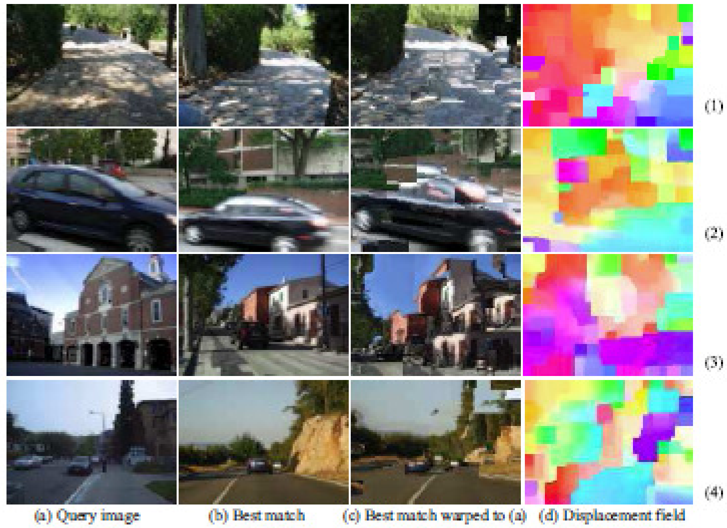
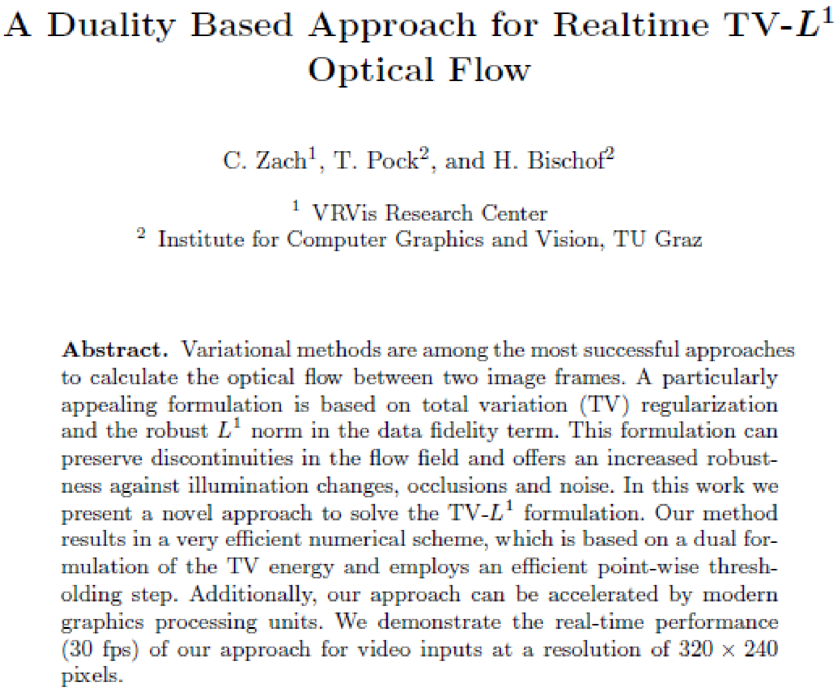


Fig. 1. Scene alignment using SIFT flow. (a) and (b) show images of similar scenes. (b) was obtained by matching (a) to a large image collection. (c) shows image (b) warped to align with (a) using the estimated dense correspondence field. (d) Visualization of pixel displacements using the color-coding scheme of [1]. Note the variation in scene appearance between (a) and (b). The visual resemblance of (a) and (c) demonstrates the quality of the scene alignment.

Recent GPU Implementation

- http://gpu4vision.icg.tugraz.at/
- Real time flow exploiting robust norm + regularized mapping



A Duality Based Approach for Realtime TV-L^1 Optical Flow

C. Zach[1], T. Pock[2], and H. Bischof[2]

[1] VRVis Research Center
[2] Institute for Computer Graphics and Vision, TU Graz

Abstract. Variational methods are among the most successful approaches to calculate the optical flow between two image frames. A particularly appealing formulation is based on total variation (TV) regularization and the robust L^1 norm in the data fidelity term. This formulation can preserve discontinuities in the flow field and offers an increased robustness against illumination changes, occlusions and noise. In this work we present a novel approach to solve the TV-L^1 formulation. Our method results in a very efficient numerical scheme, which is based on a dual formulation of the TV energy and employs an efficient point-wise thresholding step. Additionally, our approach can be accelerated by modern graphics processing units. We demonstrate the real-time performance (30 fps) of our approach for video inputs at a resolution of 320 × 240 pixels.

Today: Motion and Flow

- Motion estimation
- Patch-based motion (optic flow)
- Regularization and line processes
- Parametric (global) motion
- Layered motion models

Slide Credits

- Rick Szeliski
- Michael Black

Roadmap

- Previous: Image formation, filtering, local features, (Texture)…
- Tues: Feature-based Alignment
  - Stitching images together
  - Homographies, RANSAC, Warping, Blending
  - Global alignment of planar models
- **Today: Dense Motion Models**
  - **Local motion / feature displacement**
  - **Parametric optic flow**
- No classes next week: ICCV conference
- Oct 6th: Stereo / ‘Multi-view’: Estimating depth with known inter-camera pose
- Oct 8th: ‘Structure-from-motion’: Estimation of pose and 3D structure
  - Factorization approaches
  - Global alignment with 3D point models

Final project

- Significant novel implementation of technique related to course content
- Teams of 2 encouraged (document role!)
- Or journal length review article (no teams)
- Three components:
  - proposal document (no more than 5 pages)
  - in class results presentation (10 minutes)
  - final write-up (no more than 15 pages)

Project Proposals

- Due next Friday!
- No more than 5 pages
- Explain idea:
  - Motivation
  - Approach
  - Datasets
  - Evaluation
  - Schedule
- Proposal should convince me (and you) that project is interesting and doable given the resources you have.
- You can change topics after proposal (at your own risk!)
- I’ll consider proposal, final presentation, and report when evaluating project: a well-thought out proposal can have significant positive weight.
- Teams OK; overlap with thesis or other courses OK.

Project Ideas?

- Face annotation with online social networks?
- Classifying sports or family events from photo collections?
- Optic flow to recognize gesture?
- Finding indoor structures for scene context?
- Shape models of human silhouettes?
- Salesin: classify aesthetics?
  - Would gist regression work?