Optical Flow-Based Motion Estimation

Thanks to Steve Seitz, Simon Baker, Takeo Kanade, and anyone else who helped develop these slides.

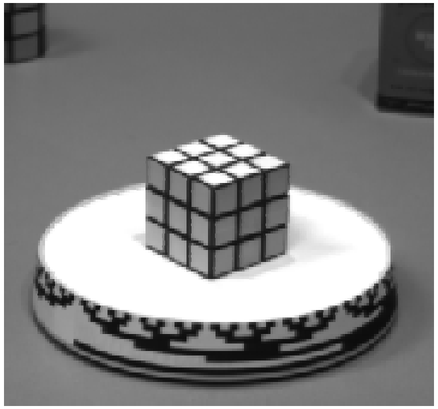
Why estimate motion?

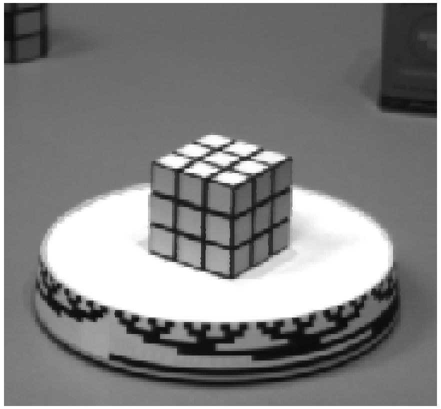
We live in a 4-D world

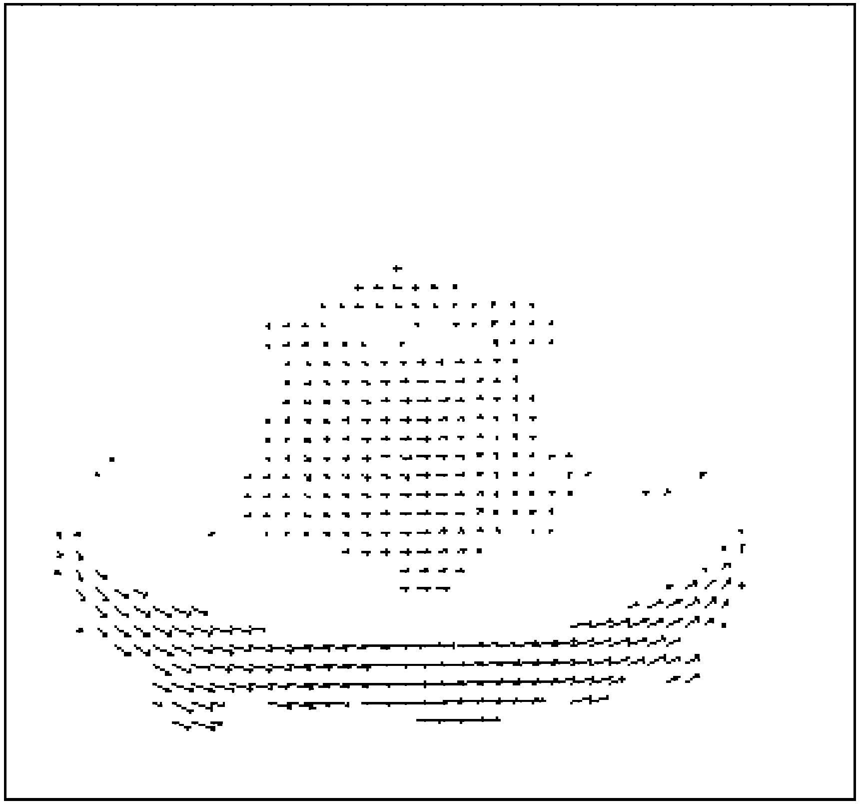
Wide applications

- Object Tracking
- Camera Stabilization
- Image Mosaics
- 3D Shape Reconstruction (SFM)
- Special Effects (Match Move)

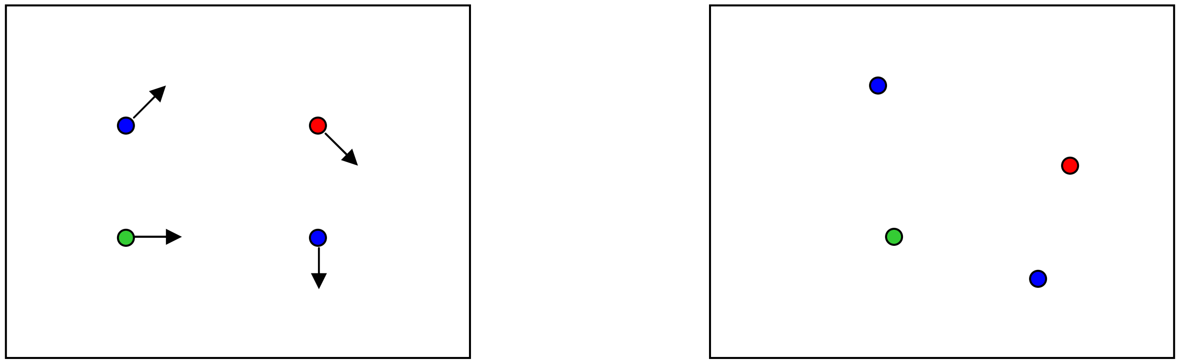
Optical flow







Problem definition: optical flow



$H(x, y)$ $I(x, y)$

How to estimate pixel motion from image H to image I?

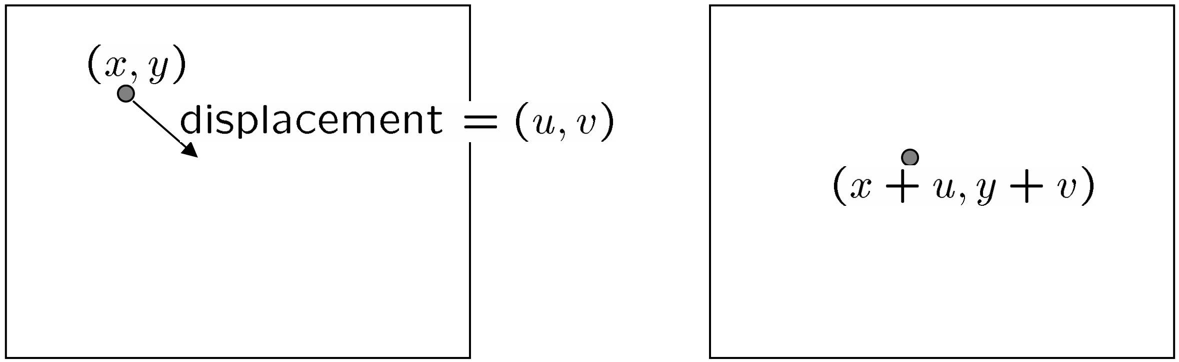
- Solve pixel correspondence problem
  - given a pixel in H, look for nearby pixels of the same color in I

Key assumptions

- **color constancy:** a point in H looks the same in I
  - For grayscale images, this is **brightness constancy**
- **small motion**: points do not move very far

This is called the **optical flow** problem

Optical flow constraints (grayscale images)



$H(x, y)$ $\quad$ $I(x, y)$

Let’s look at these constraints more closely

- brightness constancy: Q: what’s the equation?

$$H(x, y) = I(x+u, y+v)$$

- small motion: (u and v are less than 1 pixel)
  - suppose we take the Taylor series expansion of I:

$$I(x+u, y+v) = I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \text{higher order terms}$$

$$\approx I(x, y) + \frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v$$

Optical flow equation

Combining these two equations

$$0 = I(x+u, y+v) - H(x,y)$$
$$\approx I(x,y) + I_x u + I_y v - H(x,y)$$
$$\approx (I(x,y) - H(x,y)) + I_x u + I_y v$$
$$\approx I_t + I_x u + I_y v$$
$$\approx I_t + \nabla I \cdot [u \;\; v]$$

shorthand: $I_x = \frac{\partial I}{\partial x}$

The x-component of the gradient vector.

What is I_t ? The time derivative of the image at (x,y)

How do we calculate it?

Optical flow equation

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

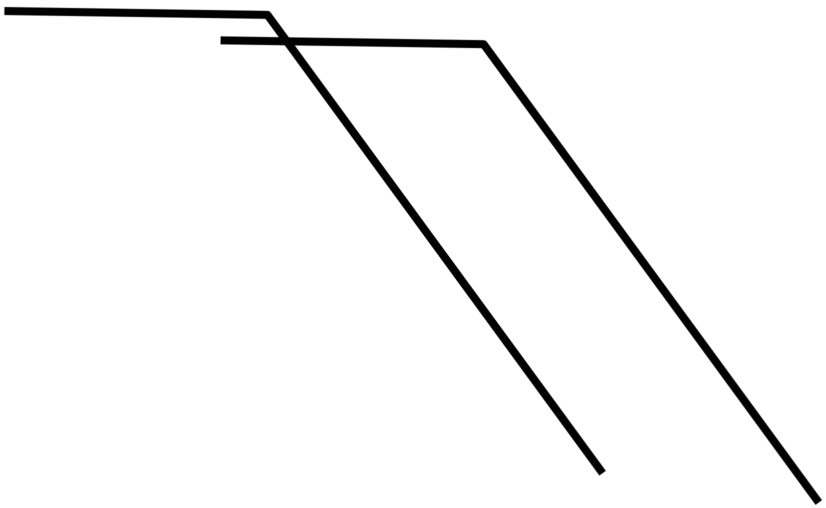
Q: how many unknowns and equations per pixel?

1 equation, but 2 unknowns (u and v)

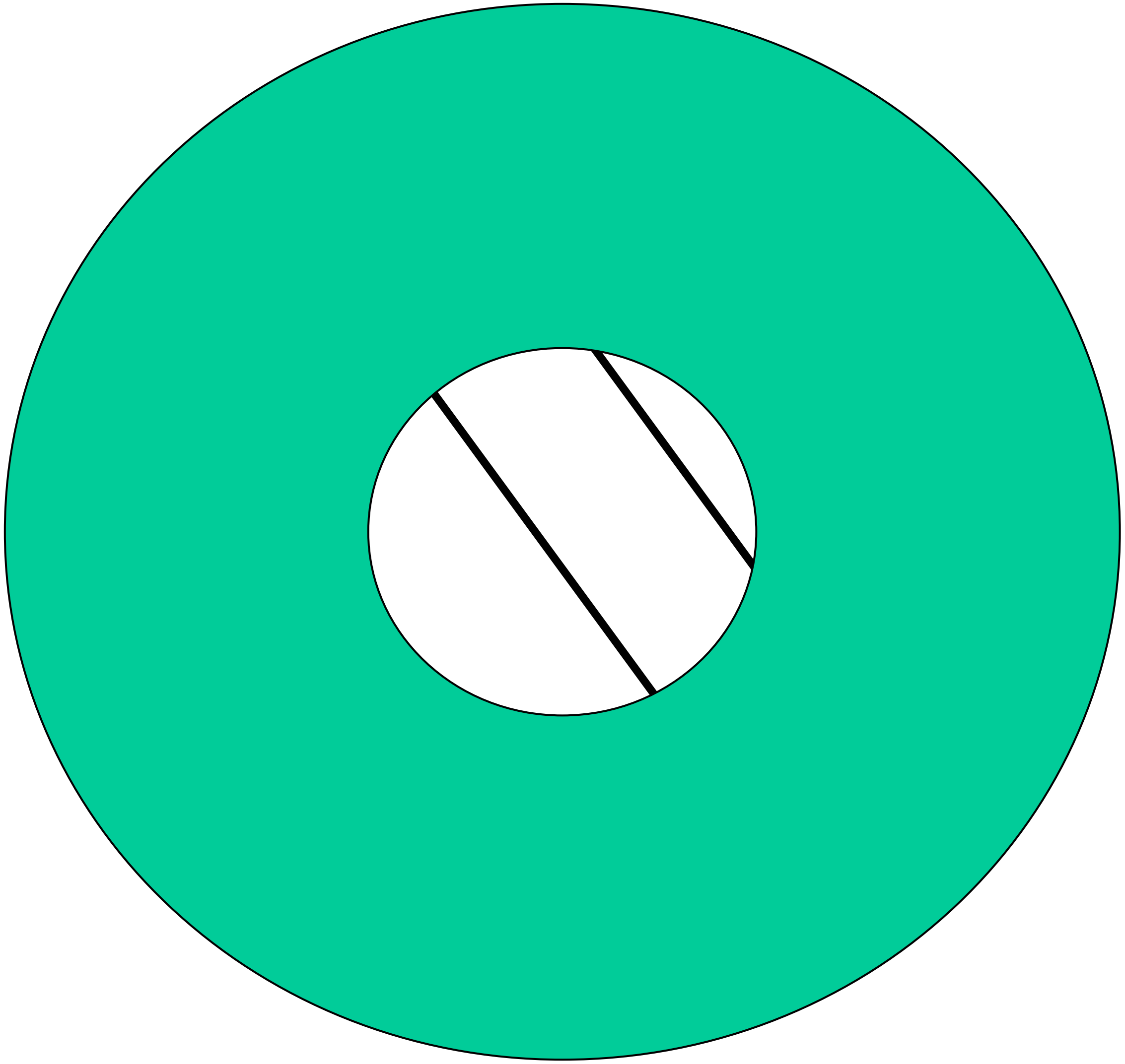
Intuitively, what does this constraint mean?

- The component of the flow in the gradient direction is determined
- The component of the flow parallel to an edge is unknown

Aperture problem



Aperture problem



Solving the aperture problem

Basic idea: assume motion field is smooth

Lukas & Kanade: assume locally constant motion

- pretend the pixel's neighbors have the same (u,v)
  - If we use a 5x5 window, that gives us 25 equations per pixel!

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

Many other methods exist. Here's an overview:

- Barron, J.L., Fleet, D.J., and Beauchemin, S, Performance of optical flow techniques**,** *International Journal of Computer Vision*, 12(1):43-77, 1994.

Lukas-Kanade flow

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!

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

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

RGB version

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*3 equations per pixel!

$$0 = I_t(\mathbf{p_i})[0,1,2] + \nabla I(\mathbf{p_i})[0,1,2] \cdot [u \ \ v]$$

$$\underbrace{\begin{bmatrix} I_x(\mathbf{p_1})[0] & I_y(\mathbf{p_1})[0] \\ I_x(\mathbf{p_1})[1] & I_y(\mathbf{p_1})[1] \\ I_x(\mathbf{p_1})[2] & I_y(\mathbf{p_1})[2] \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}})[0] & I_y(\mathbf{p_{25}})[0] \\ I_x(\mathbf{p_{25}})[1] & I_y(\mathbf{p_{25}})[1] \\ I_x(\mathbf{p_{25}})[2] & I_y(\mathbf{p_{25}})[2] \end{bmatrix}}_{\substack{A \\ 75\times 2}} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\substack{d \\ 2\times 1}} = - \underbrace{\begin{bmatrix} I_t(\mathbf{p_1})[0] \\ I_t(\mathbf{p_1})[1] \\ I_t(\mathbf{p_1})[2] \\ \vdots \\ I_t(\mathbf{p_{25}})[0] \\ I_t(\mathbf{p_{25}})[1] \\ I_t(\mathbf{p_{25}})[2] \end{bmatrix}}_{\substack{b \\ 75\times 1}}$$

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)

Conditions for solvability

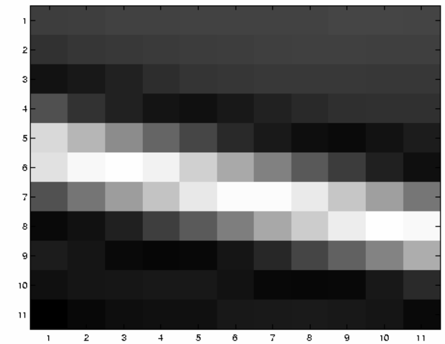
- Optimal (u, v) satisfies Lucas-Kanade equation

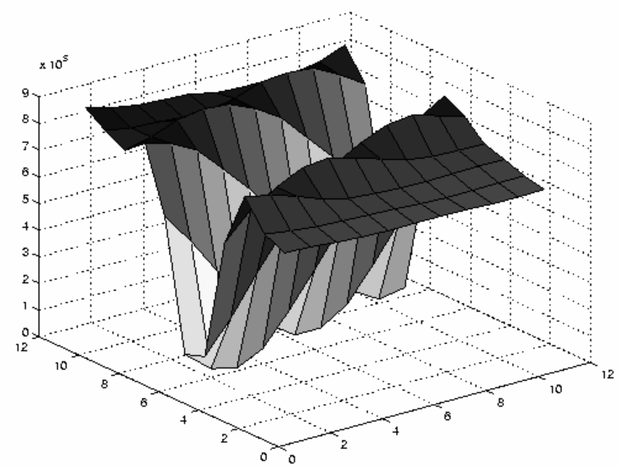
$$\underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b}$$

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)

Edges cause problems

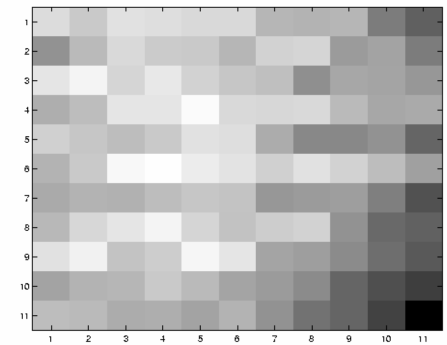


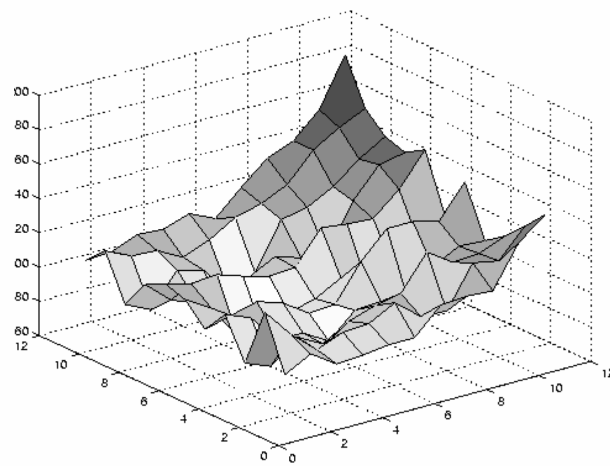


$$\sum \nabla I (\nabla I)^T$$

- large gradients, all the same
- large λ_1, small λ_2

Low texture regions don't work



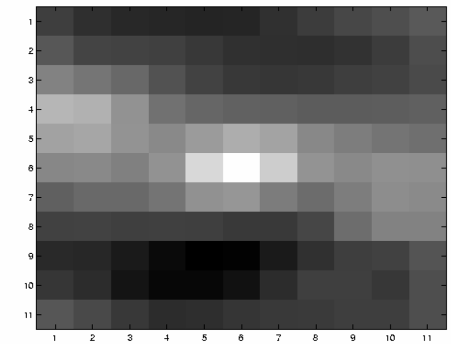


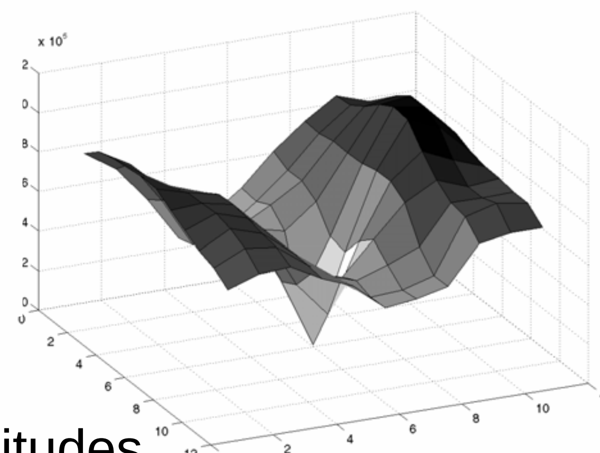
$$\sum \nabla I (\nabla I)^T$$

– gradients have small magnitude

– small λ_1, small λ_2

High textured region work best





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

– gradients are different, large magnitudes

– large λ_1, large λ_2

Errors in Lukas-Kanade

What are the potential causes of errors in this procedure?

- Suppose A^TA is easily invertible
- Suppose there is not much noise in the image

When our assumptions are violated

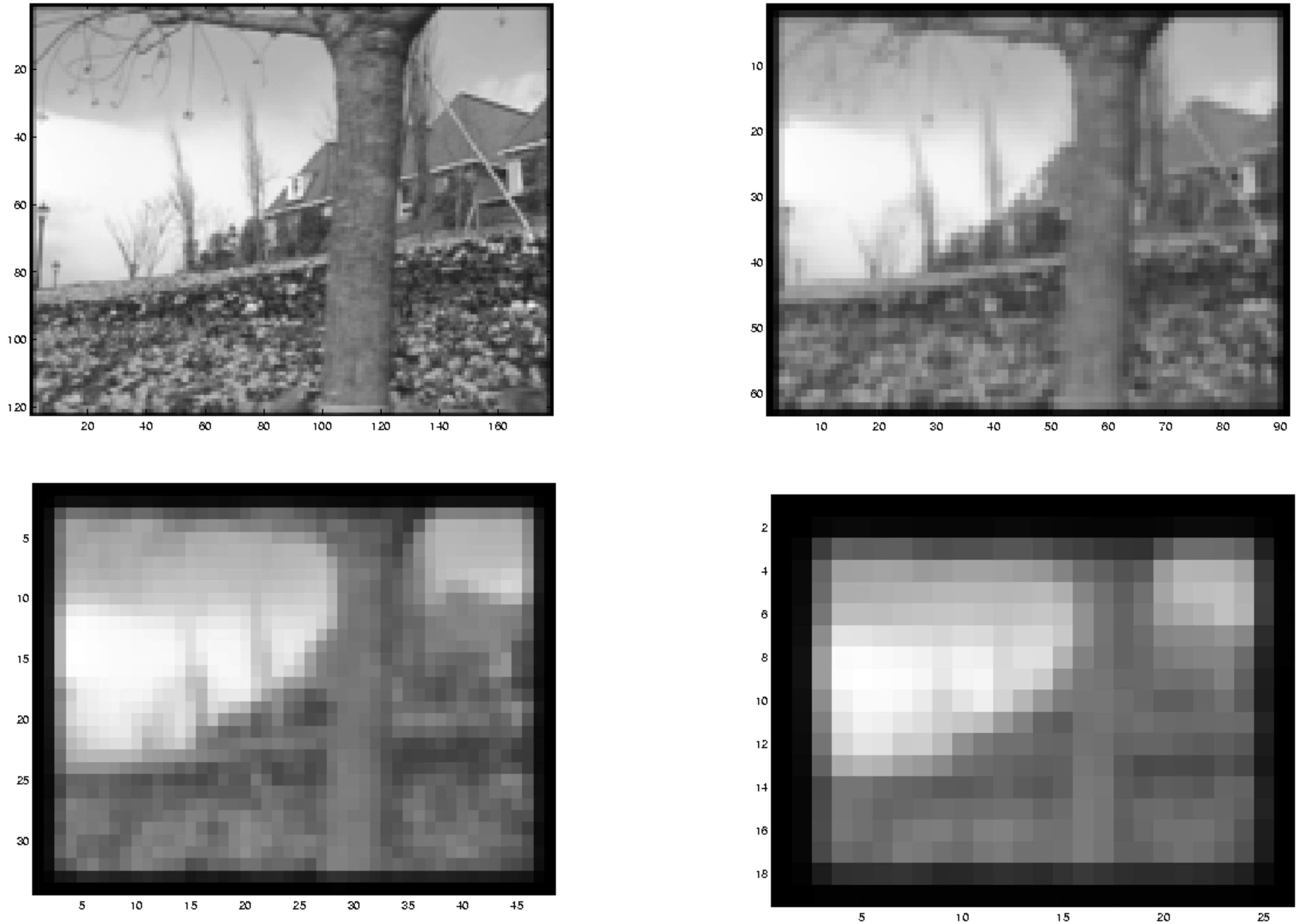
- Brightness constancy is **not** satisfied
- The motion is **not** small
- A point does **not** move like its neighbors
  - window size is too large
  - what is the ideal window size?

Revisiting the small motion assumption

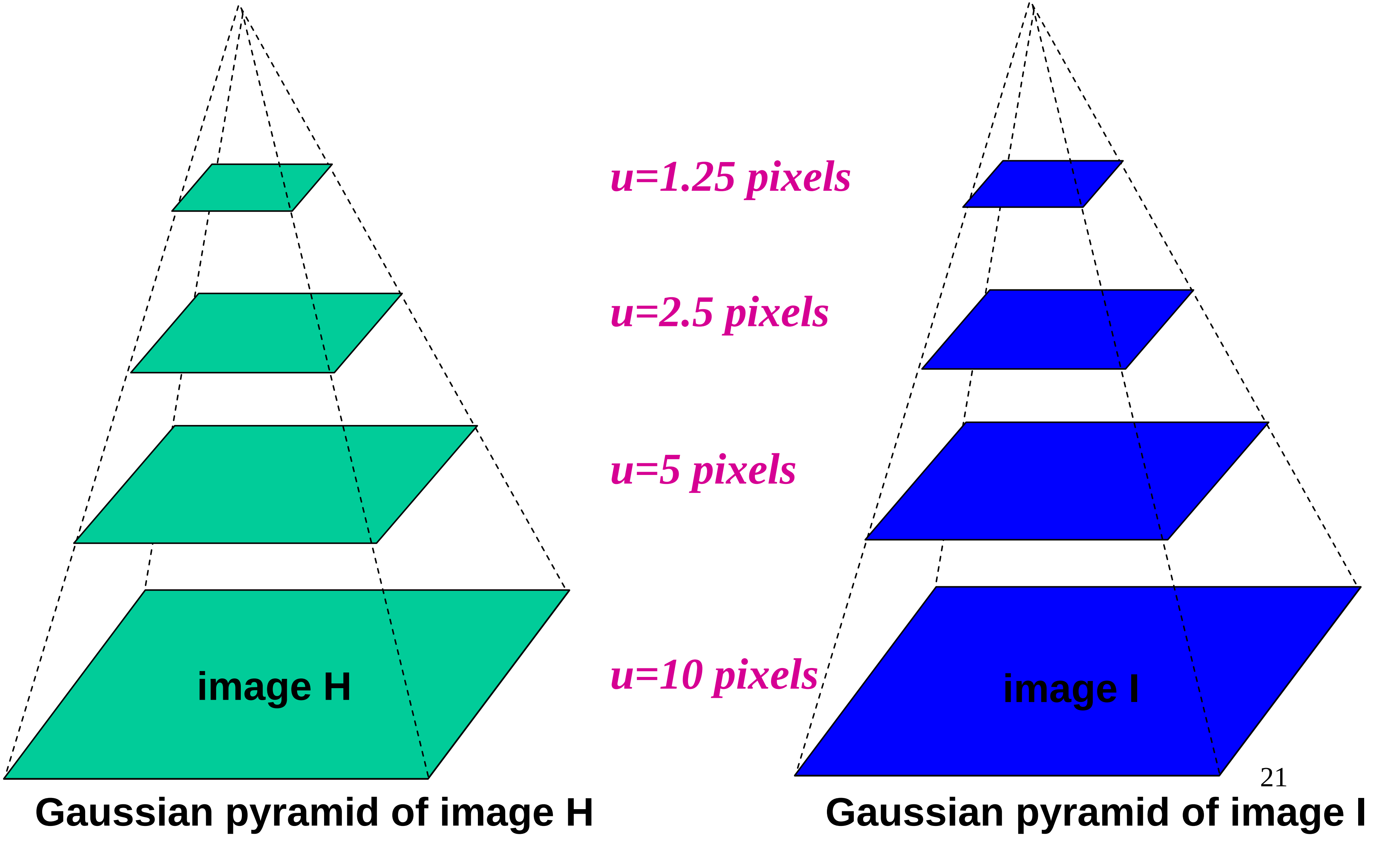
Is this motion small enough?

- Probably not—it's much larger than one pixel (2nd order terms dominate)
- How might we solve this problem?

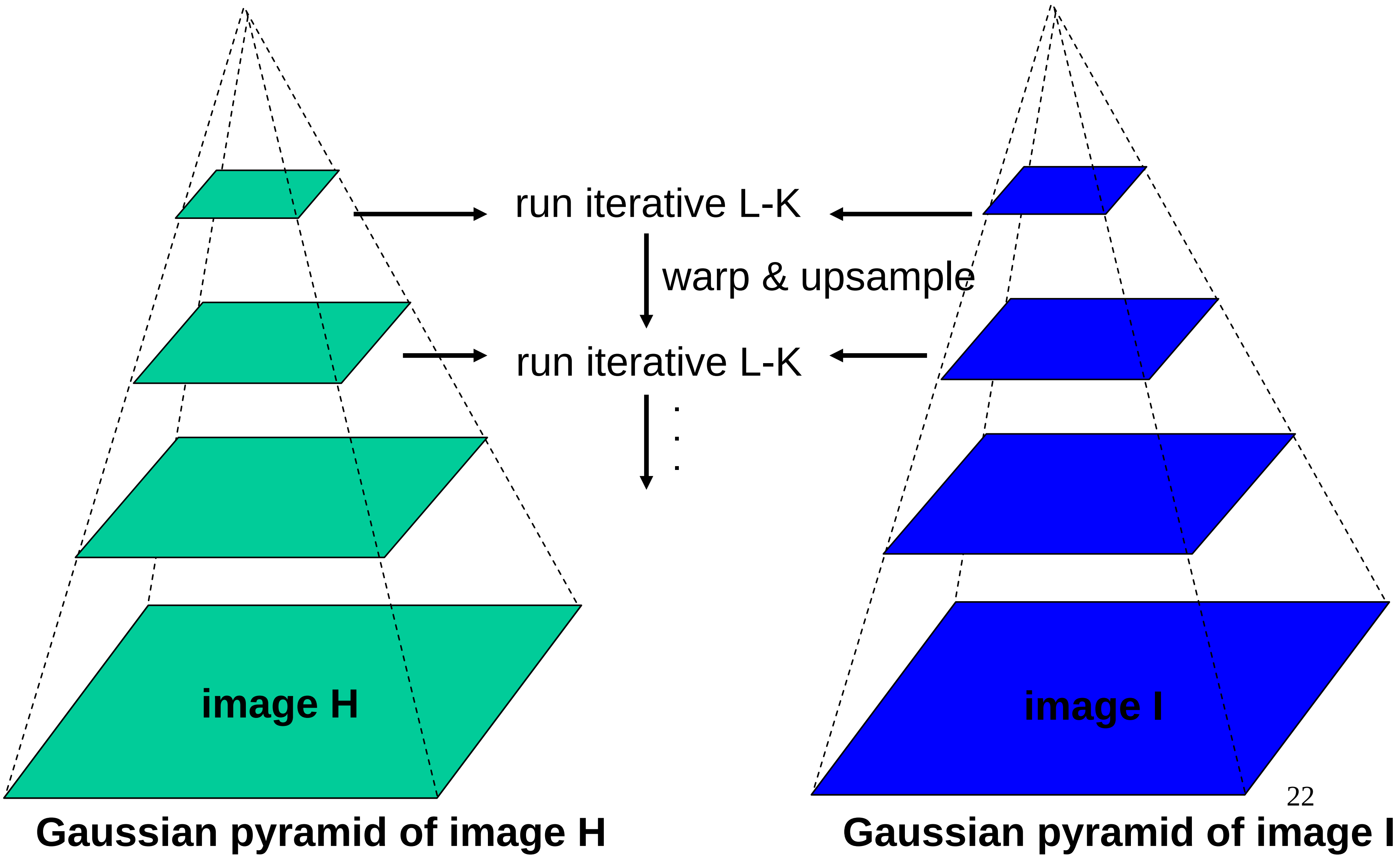
Reduce the resolution!



Coarse-to-fine optical flow estimation



Coarse-to-fine optical flow estimation

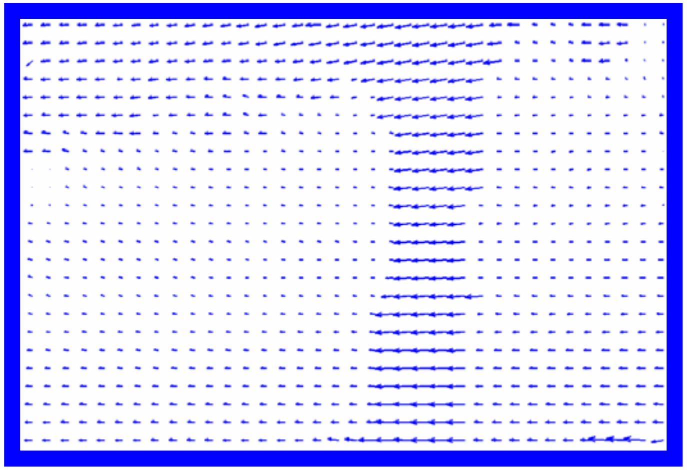


A Few Details

- Top Level
  - Apply L-K to get a flow field representing the flow from the first frame to the second frame.
  - Apply this flow field to warp the first frame toward the second frame.
  - Rerun L-K on the new warped image to get a flow field from it to the second frame.
  - Repeat till convergence.
- Next Level
  - Upsample the flow field to the next level as the first guess of the flow at that level.
  - Apply this flow field to warp the first frame toward the second frame.
  - Rerun L-K and warping till convergence as above.
- Etc.

The Flower Garden Video

What should the optical flow be?



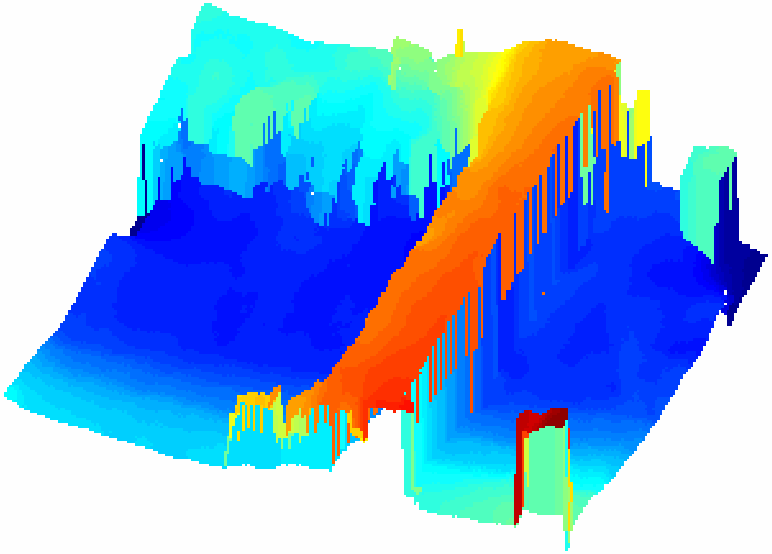
Robust Visual Motion Analysis:
Piecewise-Smooth Optical Flow

Ming Ye
Electrical Engineering
University of Washington

Structure From Motion

Rigid scene + camera translation

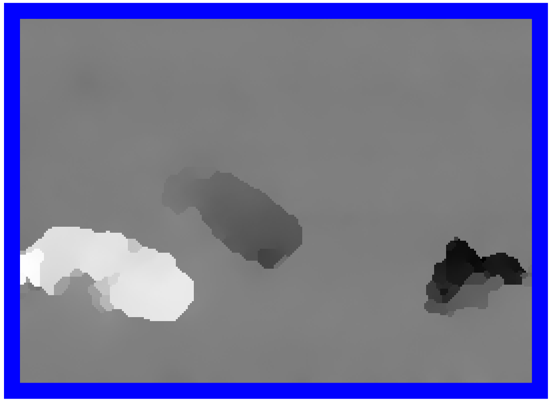
Estimated horizontal motion



Depth map

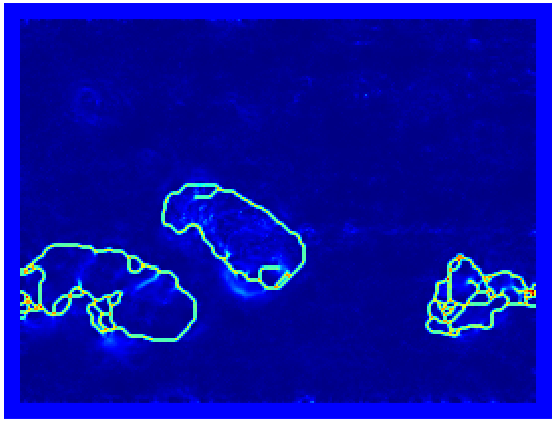
Scene Dynamics Understanding

- Surveillance
- Event analysis
- Video compression



Brighter pixels => larger speeds.

Estimated horizontal motion



Motion boundaries are smooth.

Motion smoothness

Target Detection and Tracking

A tiny airplane --- only observable by its distinct motion

Tracking results

Problem Statement

Assuming only brightness conservation and piecewise-smooth motion, find the optical flow to best describe the intensity change in three frames.

Approach: Matching-Based Global Optimization

- **Step 1. Robust local gradient-based method for high-quality initial flow estimate.**
- **Step 2. Global gradient-based method to improve the flow-field coherence.**
- **Step 3. Global matching that minimizes energy by a greedy approach.**

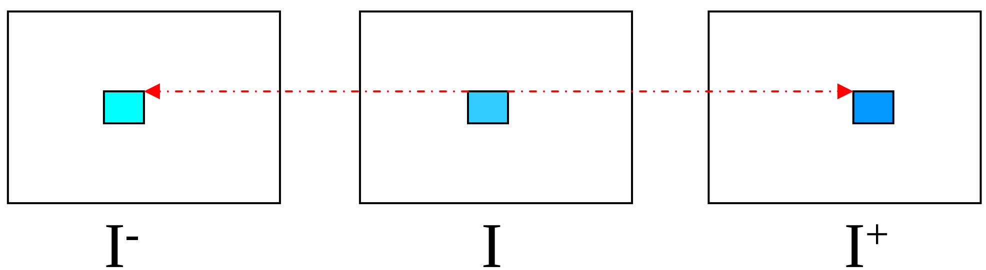
Global Energy Design

Global energy

$$E = \sum_{\text{all sites s}} E_B(V_s) + E_S(V_s)$$

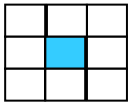
V_S is the optical flow field.
E_B is the brightness error.
E_S is the smoothness error.

I is the current frame, and I^- and I^+ are prev & next frame.
$I^-(V_s)$ is the warped intensity in prev frame.
E_B measures the minimum brightness difference between $|I^-(V_s)-I_s|$ and $|I^+(V_s)-I_s|$

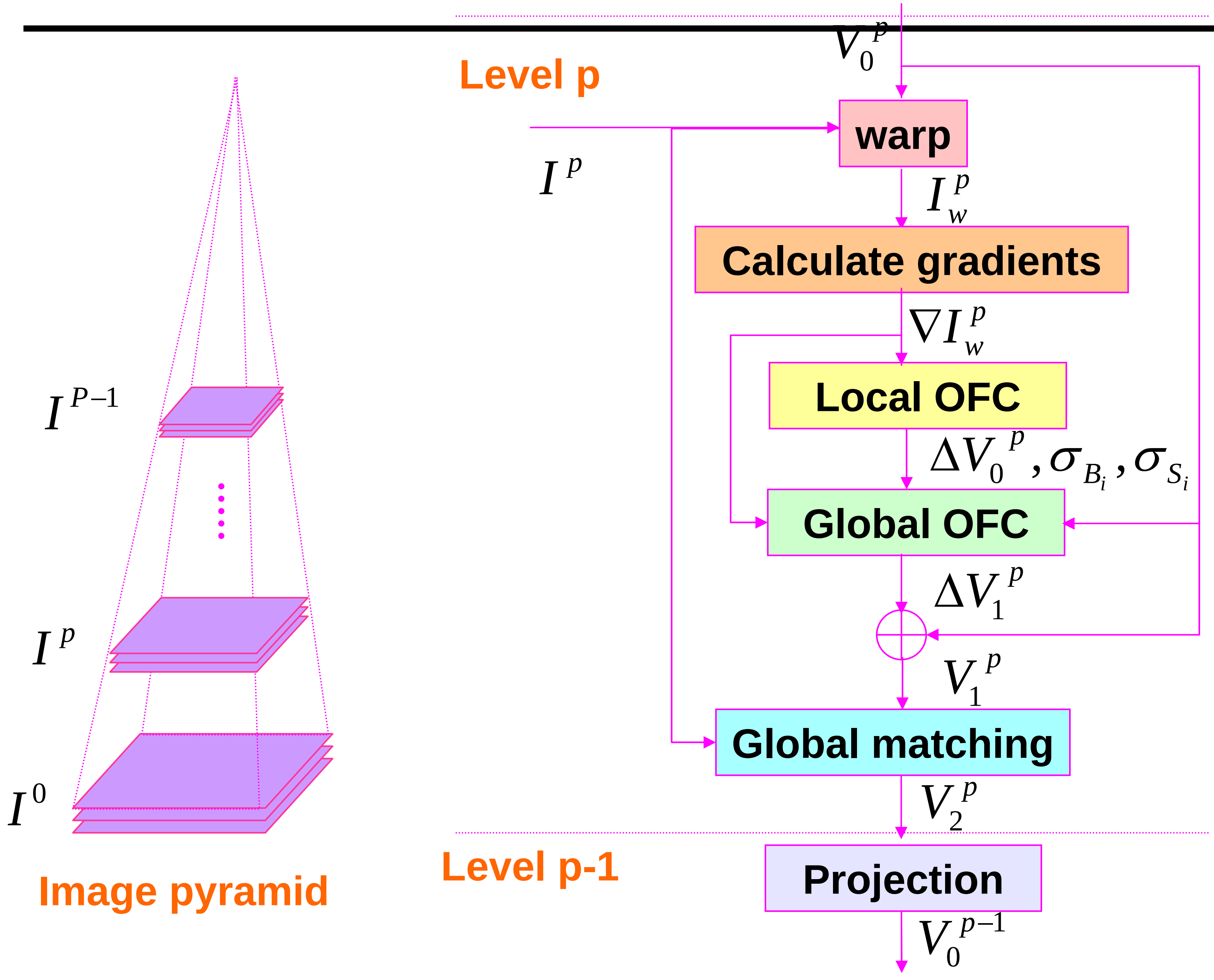


E_S is the flow smoothness error in a neighborhood about pixel s.

$$E_S(V_i) = \frac{1}{8} \sum_{n \in N_s^8} \rho(|V_s - V_n|, \sigma_{S_s})$$



Overall Algorithm



Advantages

Best of Everything

- Local OFC
  - High-quality initial flow estimates
  - Robust local scale estimates
- Global OFC
  - Improve flow smoothness
- Global Matching
  - The optimal formulation
  - Correct errors caused by poor gradient quality and hierarchical process

Results: fast convergence, high accuracy, simultaneous motion boundary detection

Experiments

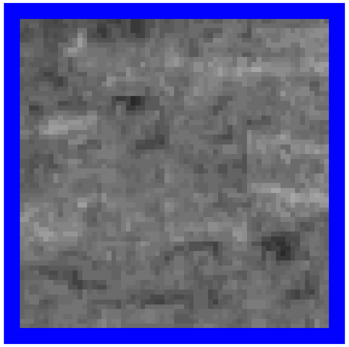
- **Experiments were run on several standard test videos.**

- **Estimates of optical flow were made for the middle frame of every three.**

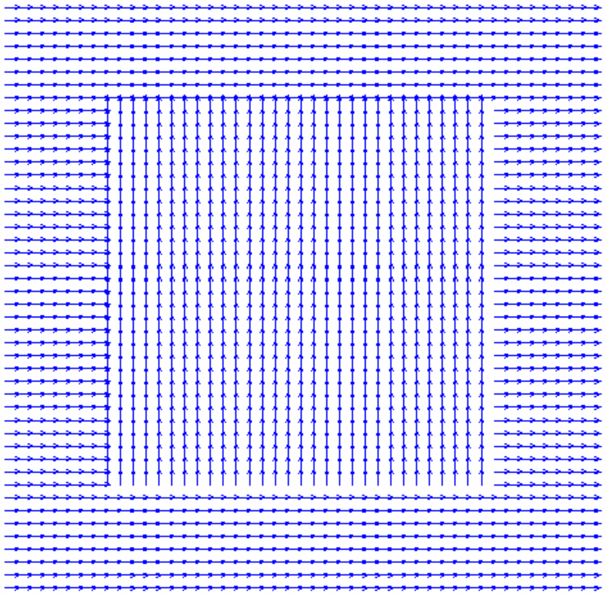
- **The results were compared with the Black and Anandan algorithm.**

TS: Translating Squares

Homebrew, ideal setting, test performance upper bound

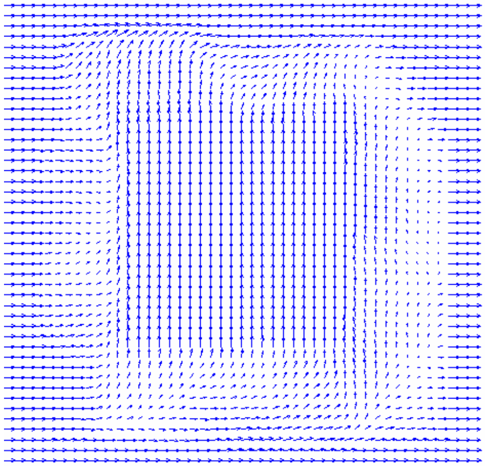


64x64, 1pixel/frame

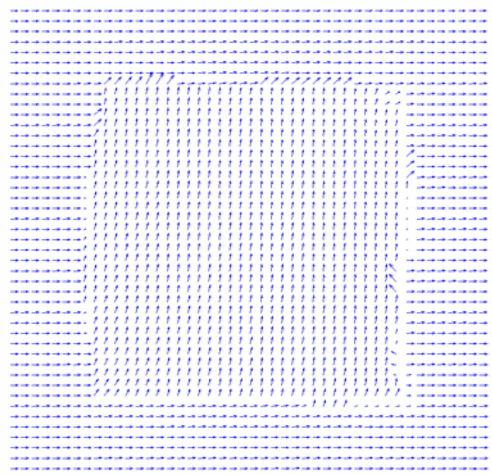


Groundtruth (cropped),
Our estimate looks the same

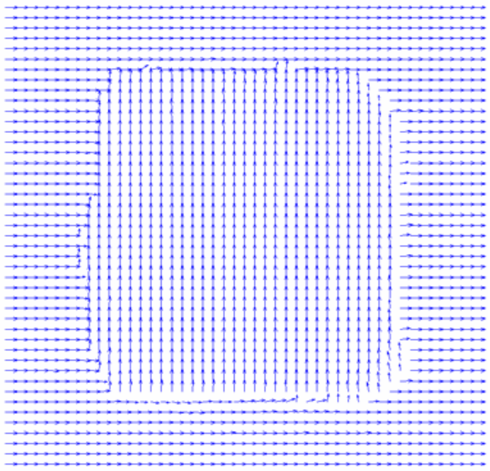
TS: Flow Estimate Plots



LS



BA



S1 (S2 is close)

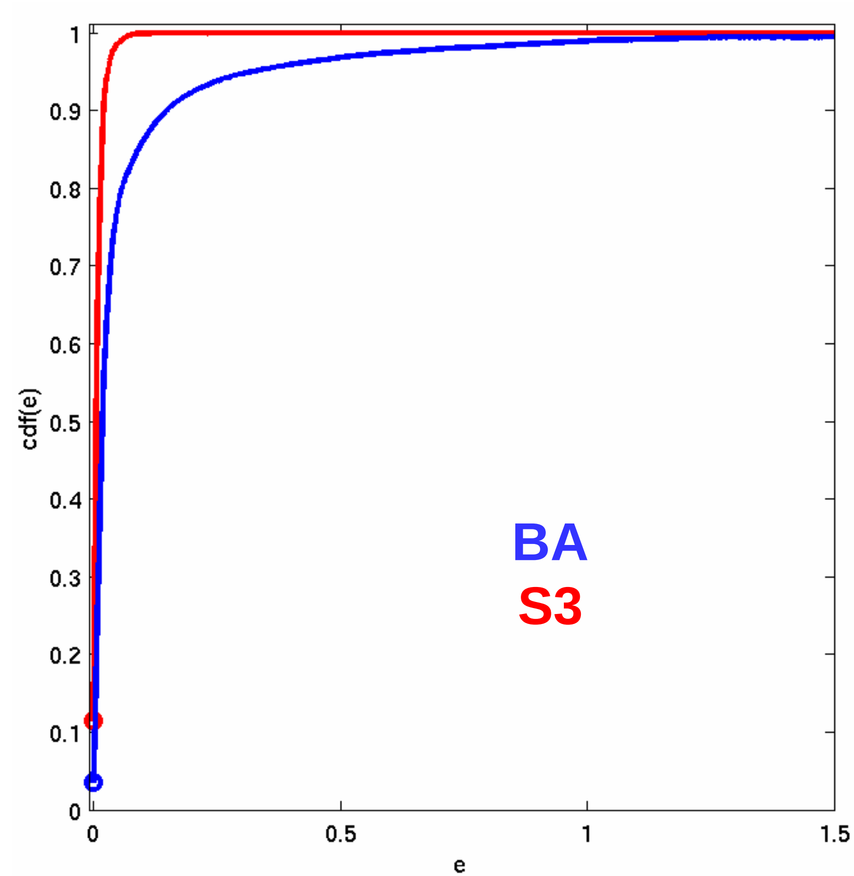
S3 looks the same as the groundtruth.

- S1, S2, S3: results from our Step I, II, III (final)

TT: Translating Tree

150x150 (Barron 94)

	$e_{\angle}(^\circ)$	$e_{\lvert\bullet\rvert}(\text{pix})$	$\bar{e}(\text{pix})$
BA	**2.60**	**0.128**	**0.0724**
S3	**0.248**	**0.0167**	**0.00984**

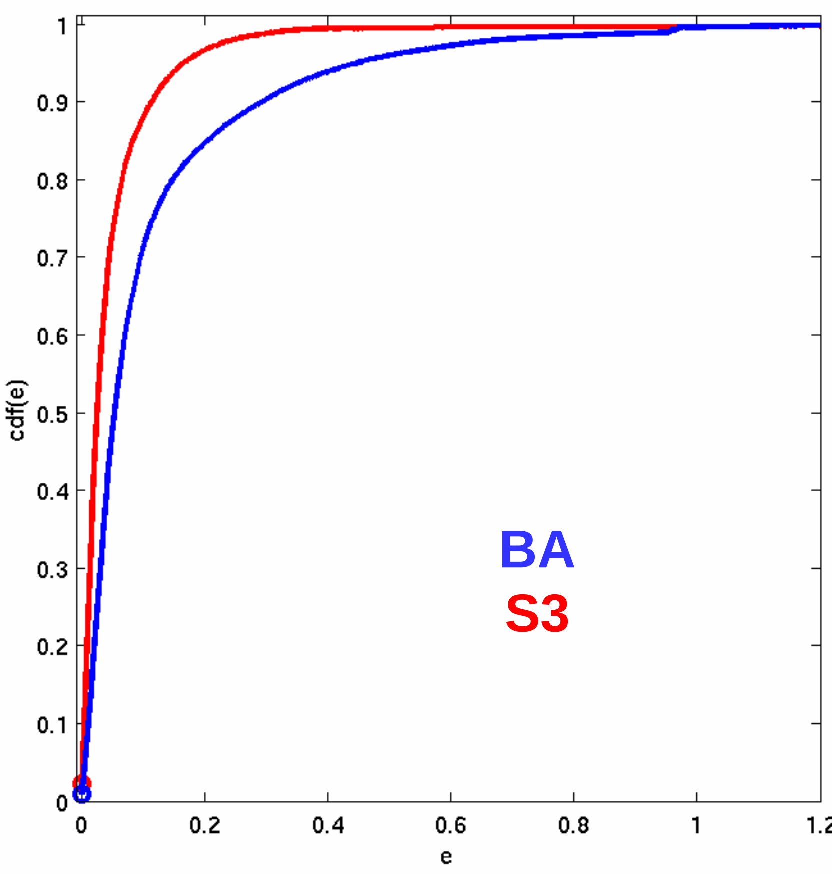


e: error in pixels, cdf: culmulative distribution function for all pixels

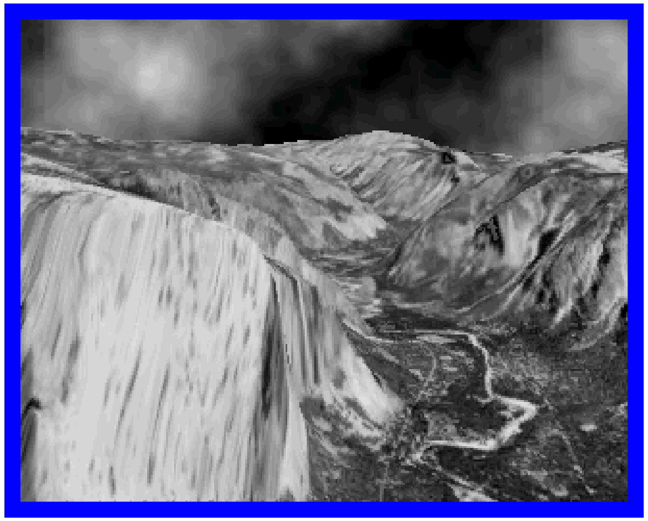
DT: Diverging Tree

150x150 (Barron 94)

	$e_{\angle}(^\circ)$	$e_{\|\bullet\|}(\text{pix})$	$\bar{e}(\text{pix})$
BA	6.36	0.182	0.114
S3	2.60	0.0813	0.0507

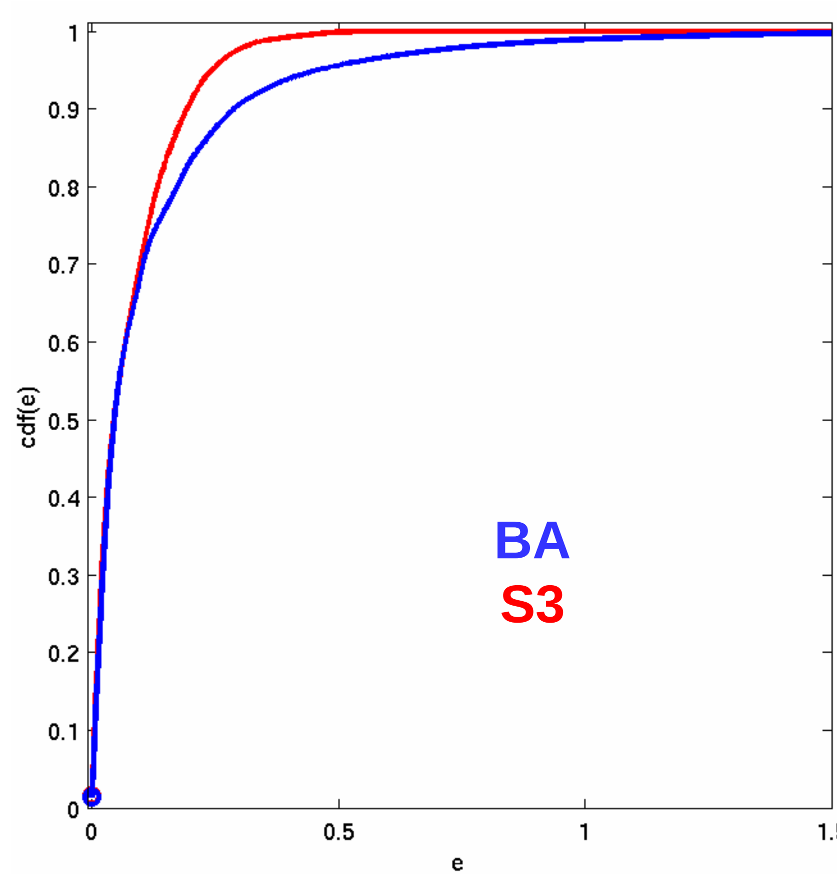


YOS: Yosemite Fly-Through



316x252 (Barron, cloud excluded)

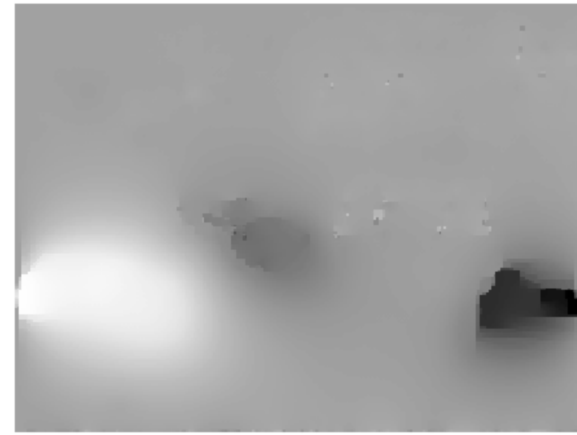
	$e_\angle(^\circ)$	$e_{\lvert\bullet\rvert}(\text{pix})$	$\bar{e}(\text{pix})$
BA	2.71	0.185	0.118
S3	1.92	0.120	0.0776



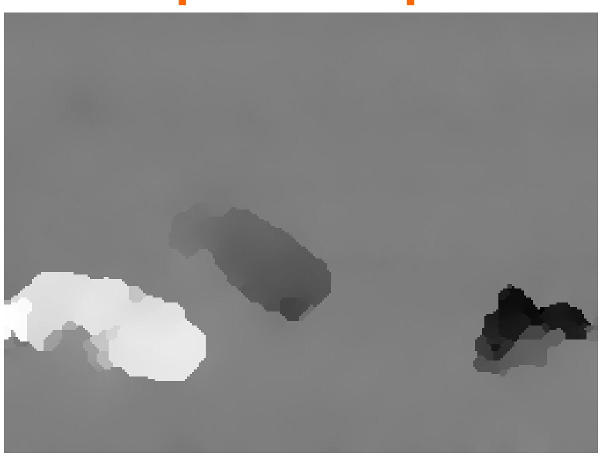
TAXI: Hamburg Taxi

256x190, (Barron 94)
max speed 3.0 pix/frame

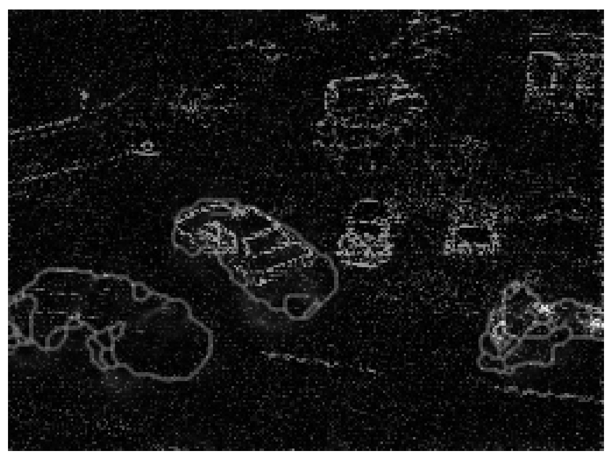
LMS



BA



Ours

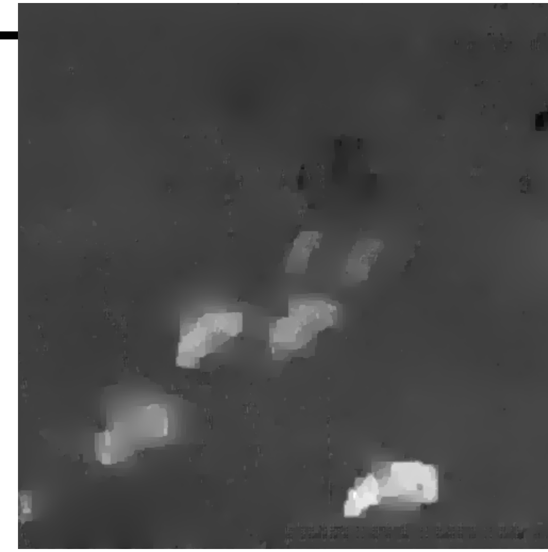


Error map

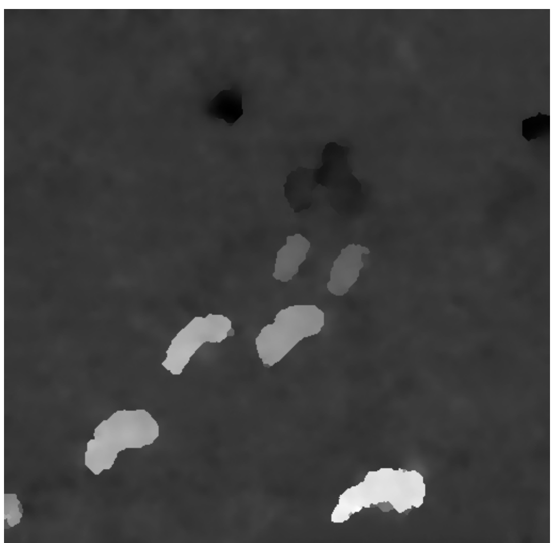
Smoothness error

Traffic

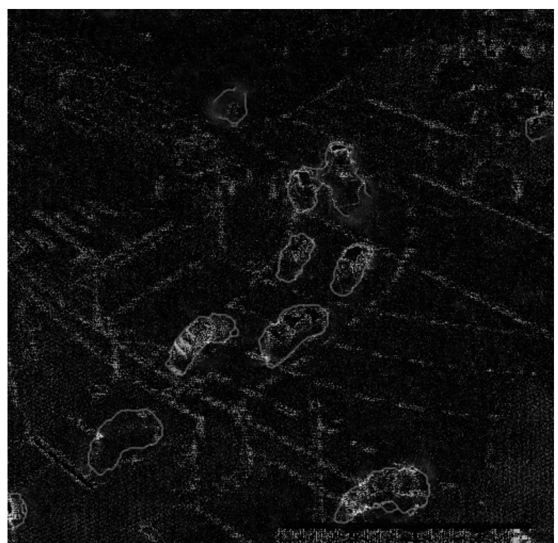
512x512
(Nagel)
max speed:
6.0 pix/frame



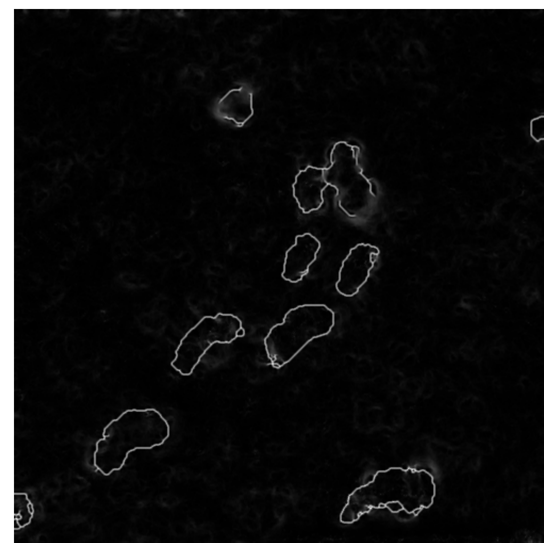
BA



Ours



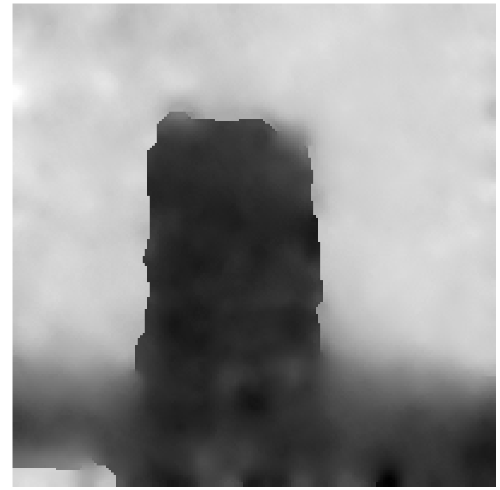
Error map



Smoothness error

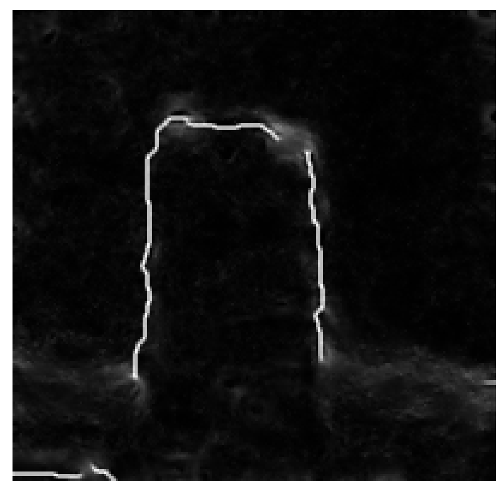
Pepsi Can

201x201 (Black) Max speed: 2pix/frame



Ours

BA

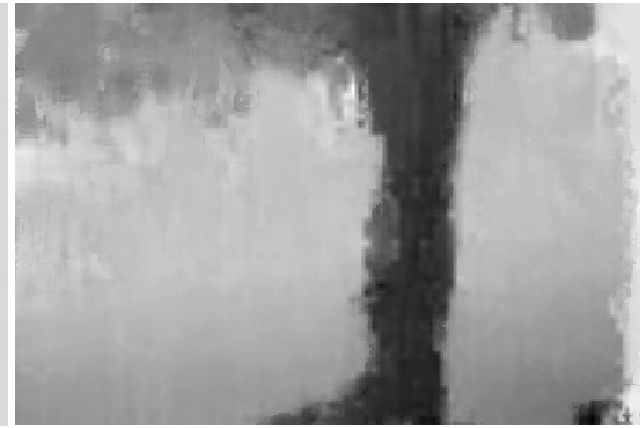


Smoothness error

FG: Flower Garden

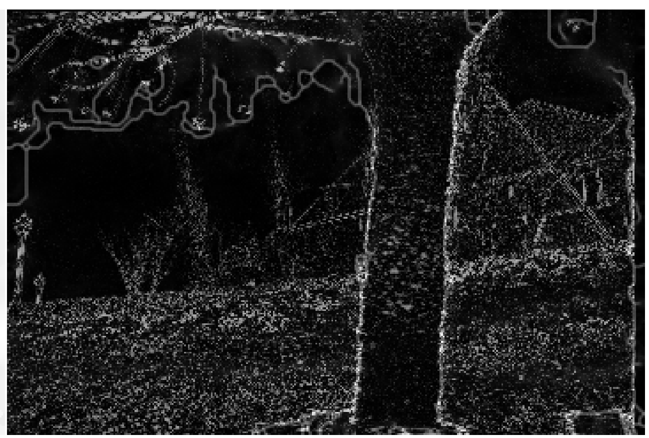
360x240 (Black)
Max speed: 7pix/frame

BA

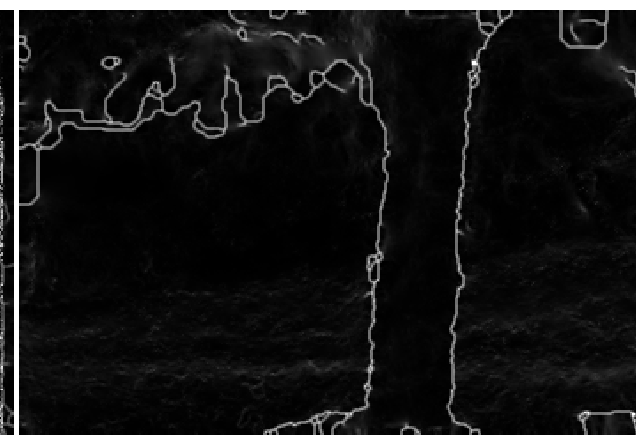


LMS

Ours



Error map



Smoothness error

MPEG Motion Compression

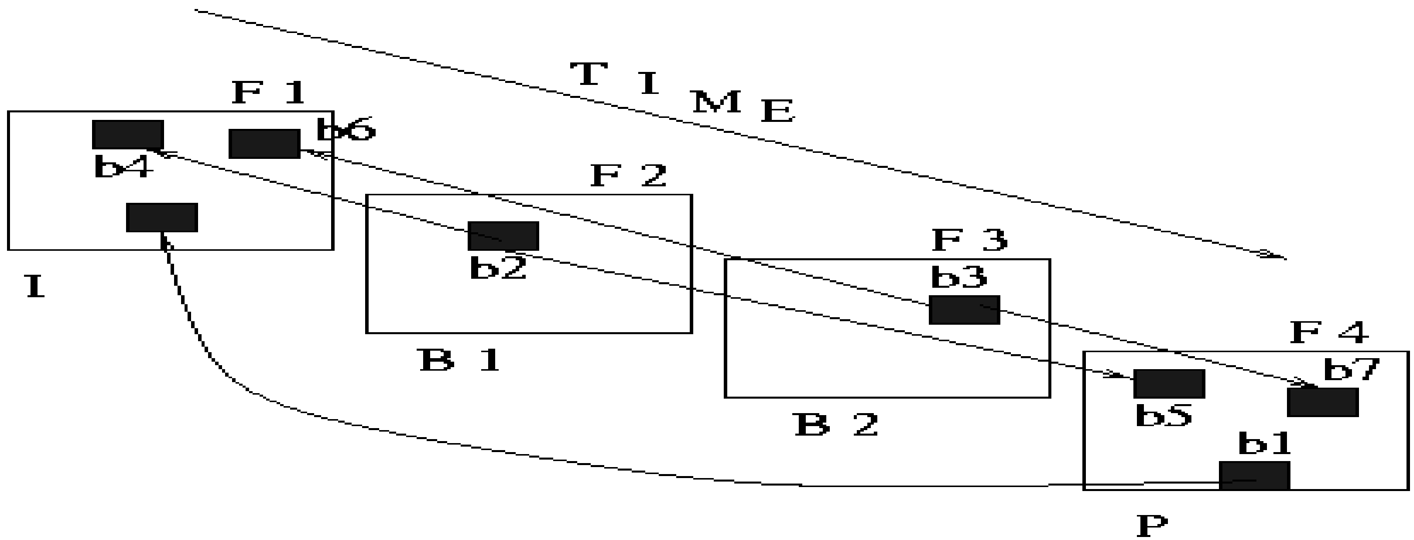
Some frames are encoded in terms of others.

I*ndependent frame* encoded as a still image using JPEG

P*redicted frame* encoded via flow vectors relative to the independent frame and difference image.

B*etween frame* encoded using flow vectors and independent and predicted frame.

MPEG compression method



F1 is independent. F4 is predicted. F2 and F3 are between.

Each block of I is matched to its closest match in P and represented by a motion vector and a block difference image.

Frames B1 and B2 between I and P are represented by two motion vectors per block referring to blocks in F1 and F4.

Example of compression

Assume frames are 512 x 512 bytes, or 32 x 32 blocks of size 16 x 16 pixels.

Frame A is ¼ megabytes = 250,000 bytes before JPEG

Frame B uses 32 x 32 =1024 motion vectors, or 2048 bytes only if delX and delY are represented as 1 byte integers.

Segmenting videos

Build video segment database

Scene change is a change of environment: newsroom to street

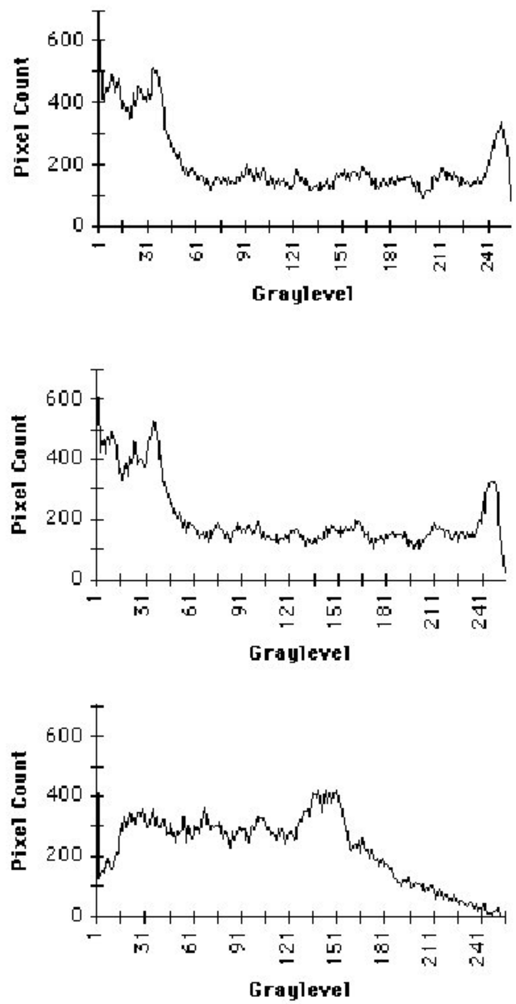
Shot change is a change of camera view of same scene

Camera pan and zoom, as before

Fade, dissolve, wipe are used for transitions

Scene change

Detect via histogram change



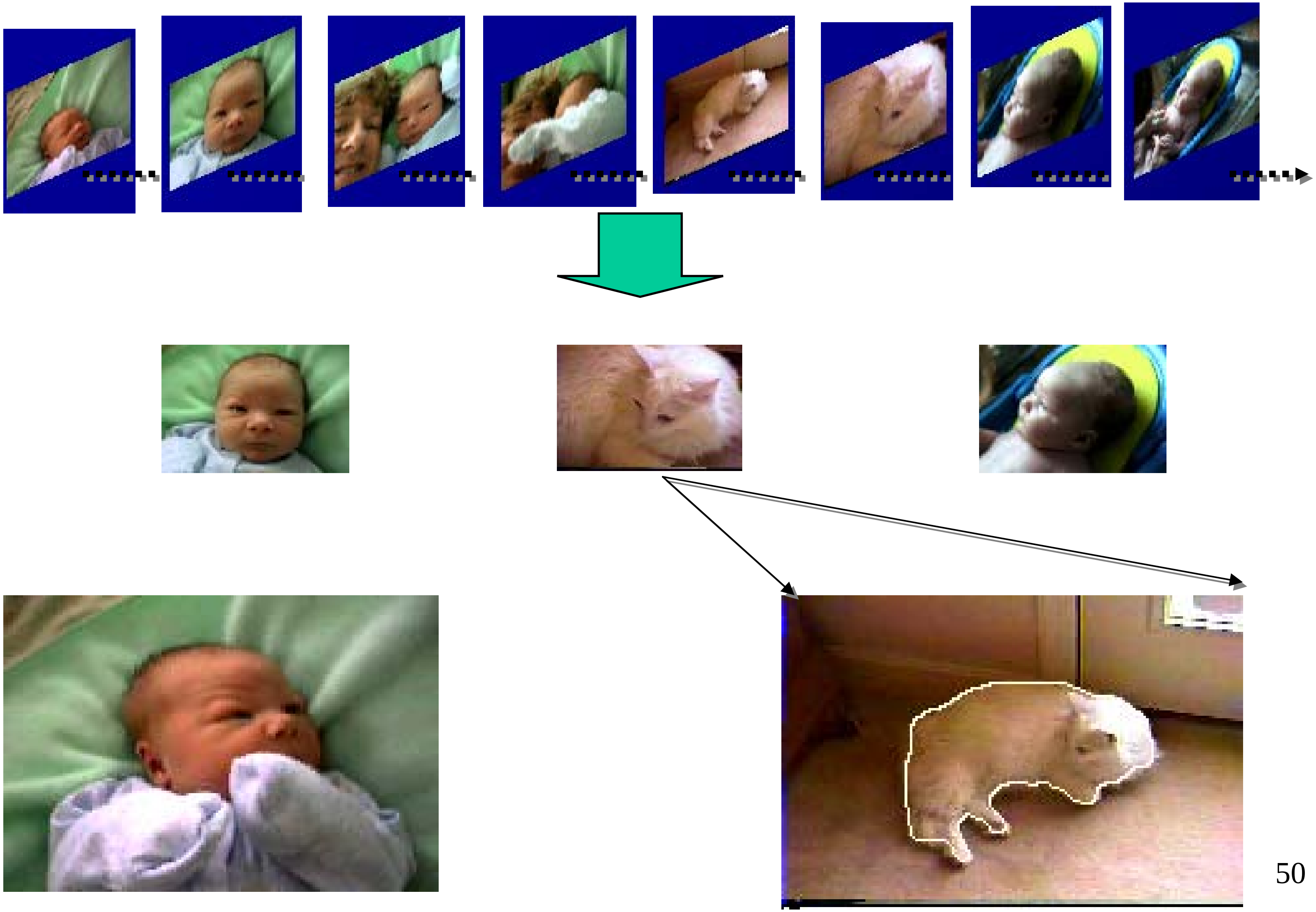
(Top) gray level histogram of intensities from frame 1 in newsroom.

(Middle) histogram of intensities from frame 2 in newsroom.

(Bottom) histogram of intensities from street scene.

Histograms change less with pan and zoom of same scene.

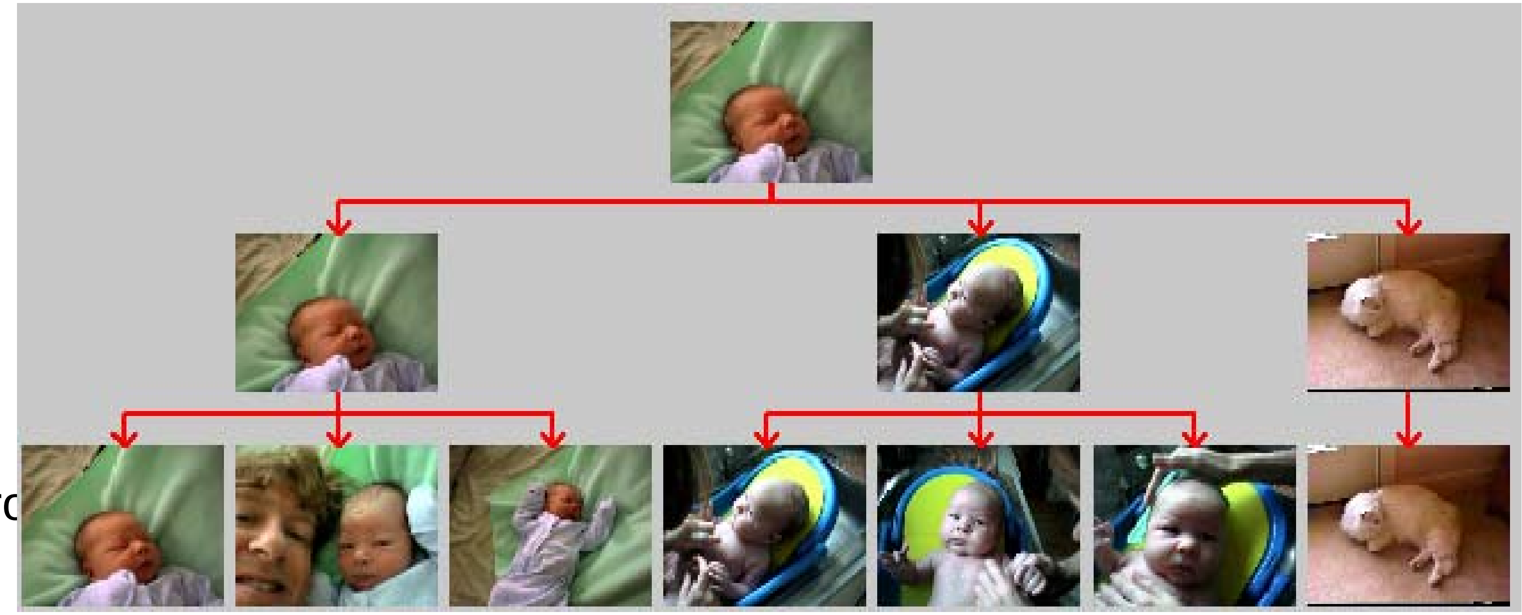
Daniel Gatica Perez's work on describing video content



Our problem: Finding Video Structure

Video Structure: hierarchical description of visual content

Table of Contents



Fro

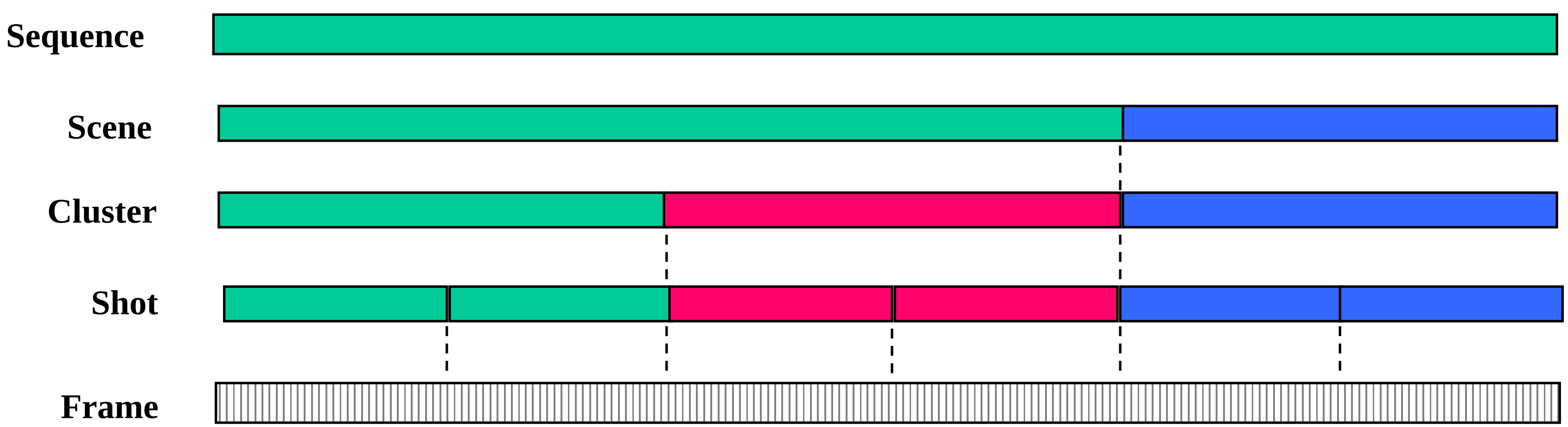
Hierarchical Structure in Video: Extensive Operators

Video Sequence

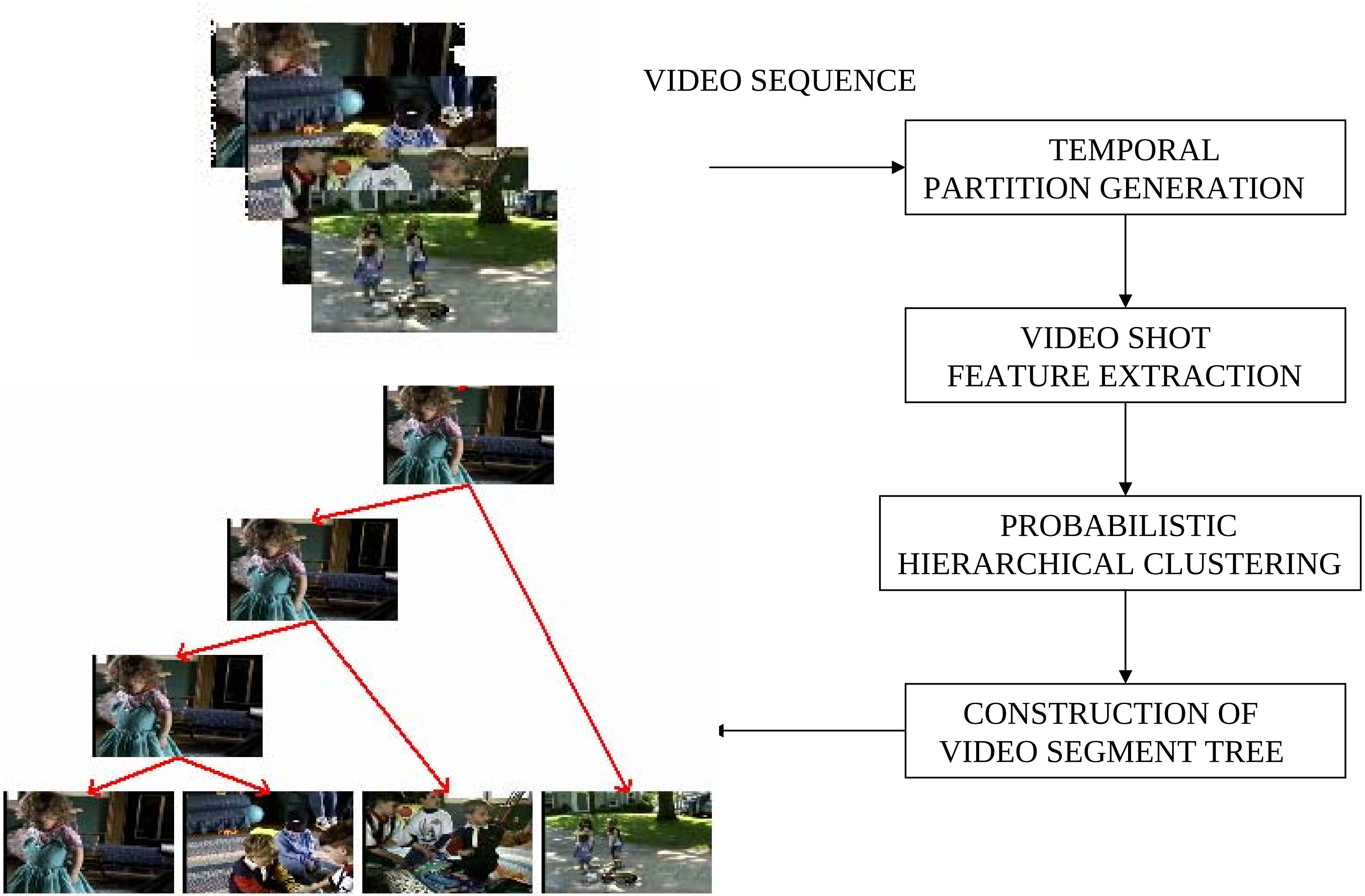
Scenes: Semantic Concept. Fair to use?

Clusters: Collection of temporally adjacent/visually similar shots

Shots: Consecutive frames recorded from a single camera



Daniel’s Approach



Video Structuring Results (I)

35 shots
9 clusters detected

Video Structuring Results (II)

12 shots
4 clusters

Tree-based Video Representation