Feature Tracking and Video Textures

CS448V — Computational Video Manipulation

April 2019

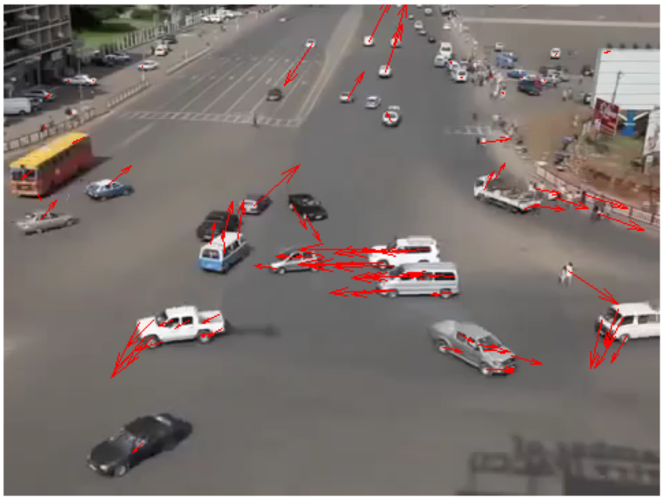
Feature Tracking

Why is motion of features useful?

Slide credit: Niebles and Krishna

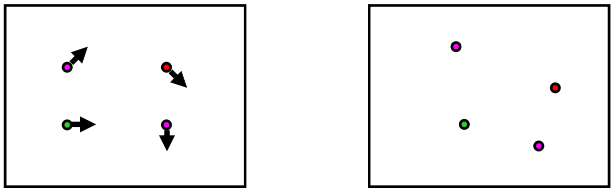
Feature Tracking

Why is motion of features useful?



Slide credit: Niebles and Krishna

Estimating Optical Flow



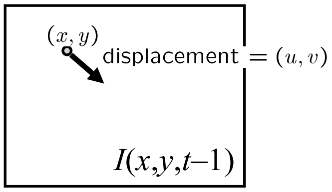
Given two subsequent frames, estimate the apparent motion field u(x,y), v(x,y) between them

Key assumptions

- **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

Slide credit: Savarese

The brightness constancy constraint



Brightness Constancy Equation:

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

Linearizing the right side using Taylor expansion:

Image derivative along x

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

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

Hence, $I_x \cdot u + I_y \cdot v + I_t \approx 0 \quad \rightarrow \nabla I \cdot \begin{bmatrix} u & v \end{bmatrix}^T + I_t = 0$

Slide credit: Savarese

Computing Derivatives

$\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$ first image

$\begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}$ second image

I_x

$\begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix}$ first image

$\begin{bmatrix} -1 & -1 \\ 1 & 1 \end{bmatrix}$ second image

I_y

$\begin{bmatrix} -1 & -1 \\ -1 & -1 \end{bmatrix}$ first image

$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$ second image

I_t

Slide credit: Savarese

The brightness constancy constraint

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

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

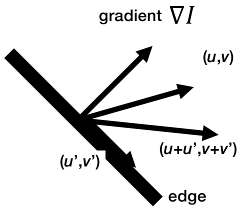
How many equations and unknowns per pixel?

One equation (this is a scalar equation!), two unknowns (u,v)

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

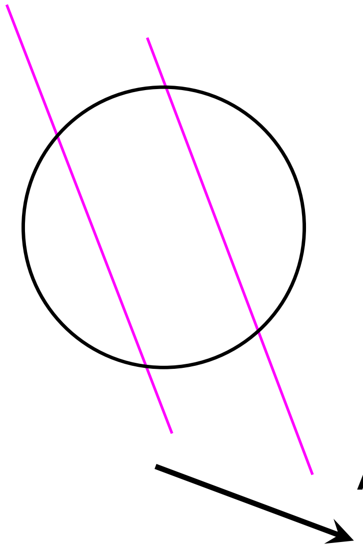
If (u, v) satisfies the equation,
so does ($u+u'$, $v+v'$) if

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



Slide credit: Savarese

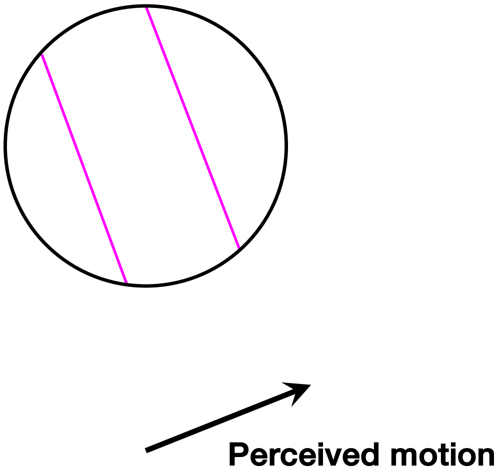
The aperture problem



Actual motion

Slide credit: Savarese

The aperture problem



Slide credit: Savarese

Solving the Ambiguity

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

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

$$\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}$$

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.

Slide credit: Savarese

Lucas-Kanade Flow

Overconstrained linear system:

$$\begin{bmatrix} I_x(\mathrm{p}_1) & I_y(\mathrm{p}_1) \\ I_x(\mathrm{p}_2) & I_y(\mathrm{p}_2) \\ \vdots & \vdots \\ I_x(\mathrm{p}_{25}) & I_y(\mathrm{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathrm{p}_1) \\ I_t(\mathrm{p}_2) \\ \vdots \\ I_t(\mathrm{p}_{25}) \end{bmatrix} \qquad \underset{25\text{x}2}{A} \; \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

Slide credit: Savarese

Lucas-Kanade Flow

Overconstrained linear system:

$$\begin{bmatrix} I_x(\mathrm{p}_1) & I_y(\mathrm{p}_1) \\ I_x(\mathrm{p}_2) & I_y(\mathrm{p}_2) \\ \vdots & \vdots \\ I_x(\mathrm{p}_{25}) & I_y(\mathrm{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathrm{p}_1) \\ I_t(\mathrm{p}_2) \\ \vdots \\ I_t(\mathrm{p}_{25}) \end{bmatrix} \qquad \underset{25\text{x}2}{A} \; \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b}$$

Least squares solution for *d* given by $(A^TA)\ d = A^Tb$

$$\underset{A^TA}{\begin{bmatrix} \sum I_xI_x & \sum I_xI_y \\ \sum I_xI_y & \sum I_yI_y \end{bmatrix}} \begin{bmatrix} u \\ v \end{bmatrix} = - \underset{A^Tb}{\begin{bmatrix} \sum I_xI_t \\ \sum I_yI_t \end{bmatrix}}$$

The summations are over all pixels in the 5 x 5 window

Slide credit: Savarese

Conditions for Solvability

Optimal (u, v) satisfies Lucas-Kanade equation

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

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

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)

Slide credit: Savarese

M = $A^T A$ is the *second moment matrix* ! (Harris corner detector…)

$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y] = \sum \nabla I (\nabla I)^T$$

- Eigenvectors and eigenvalues of $A^T A$ relate to edge direction and magnitude
  - The eigenvector associated with the larger eigenvalue points in the direction of fastest intensity change
  - The other eigenvector is orthogonal to it

Slide credit: Savarese

Edge

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

– gradients very large or very small

– large λ_1, small λ_2

Slide credit: Savarese

Low Texture Region

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

– gradients have small magnitude

– small λ_1, small λ_2

Slide credit: Savarese

High Texture Region

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

– gradients are different, large magnitudes

– large λ_1, large λ_2

Slide credit: Savarese

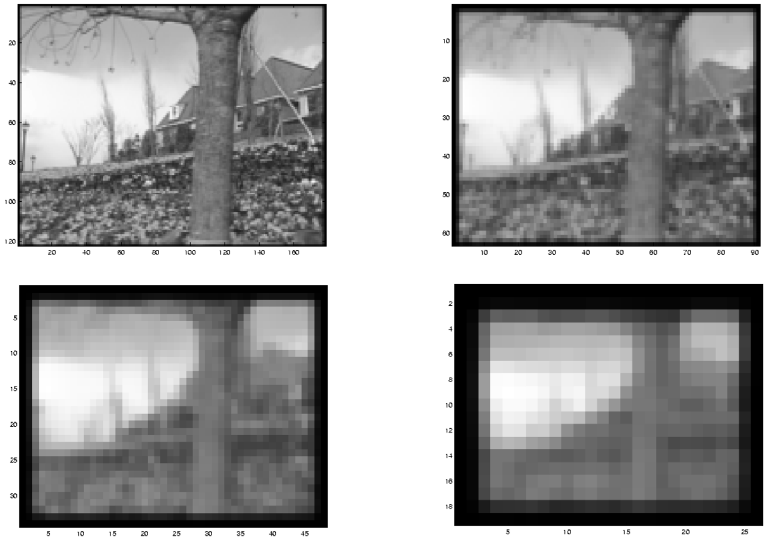
Revisiting Small Motion Assumption

Is this motion small enough?

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

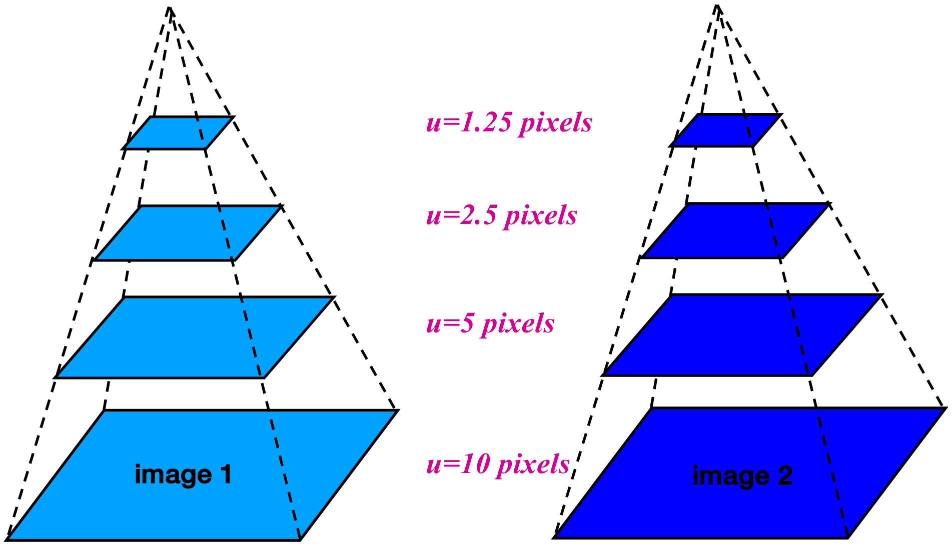
* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Reduce Resolution



* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

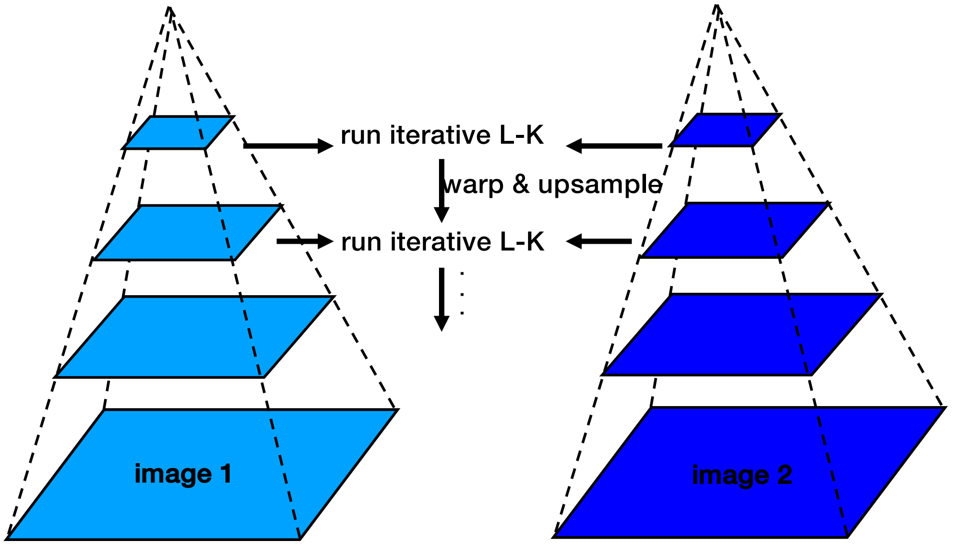
Course to Fine Estimation



Gaussian pyramid of image 1

Gaussian pyramid of image 2

Course to Fine Estimation



run iterative L-K

warp & upsample

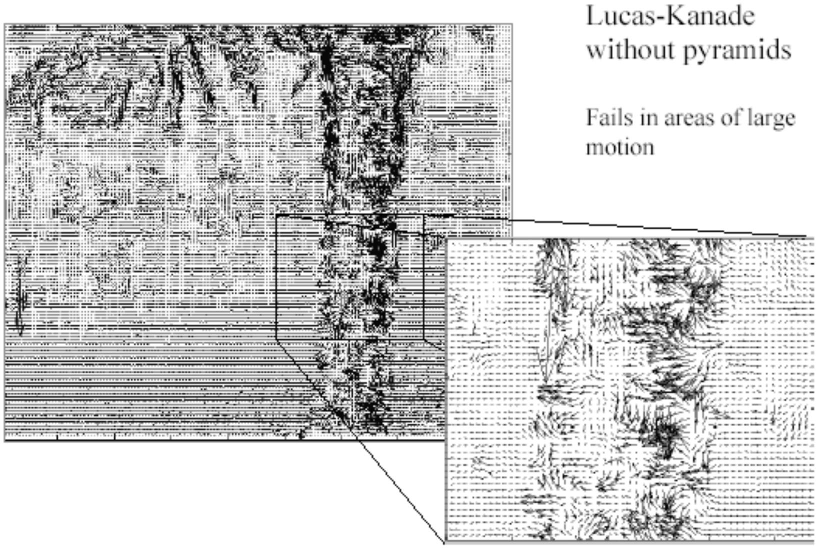
run iterative L-K

Gaussian pyramid of image 1 (t)

Gaussian pyramid of image 2 (t+1)

Source: Silvio Savarese

Optical Flow Results

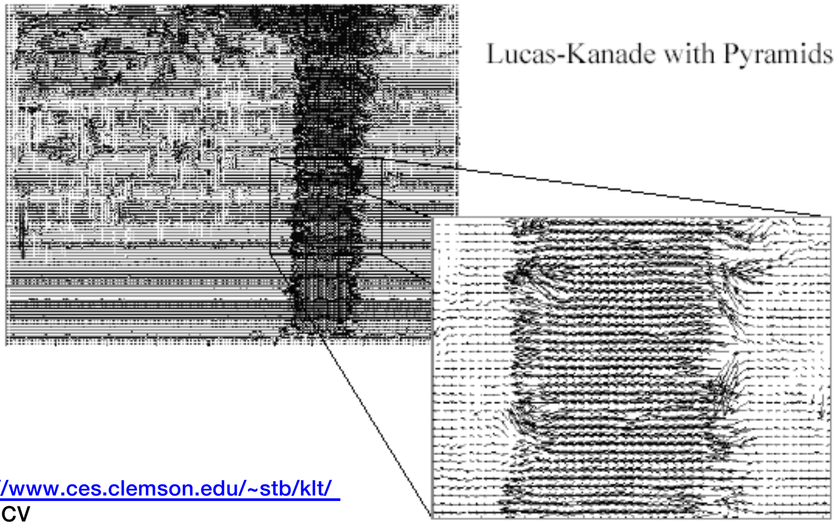


Lucas-Kanade without pyramids

Fails in areas of large motion

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

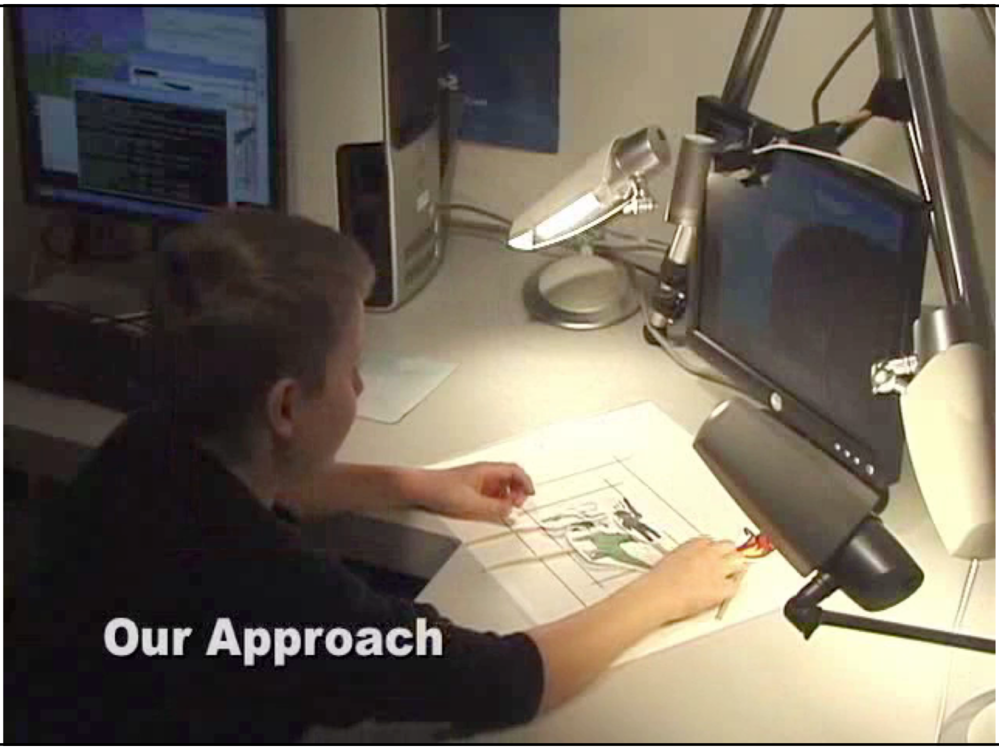
Optical Flow Results



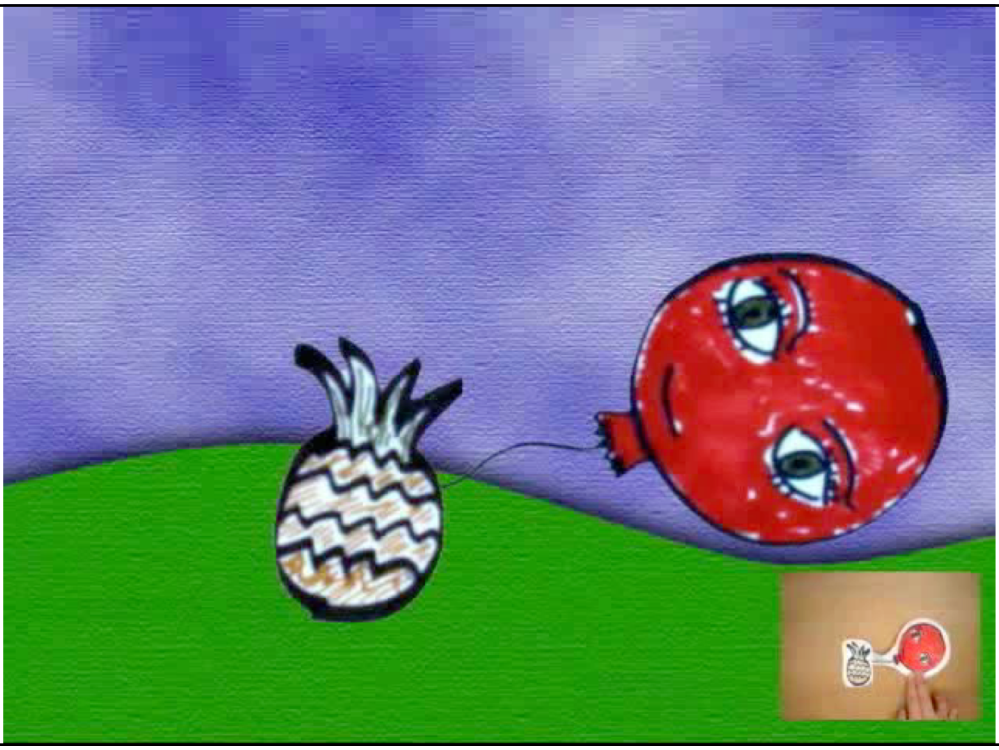
- http://www.ces.clemson.edu/~stb/klt/
- OpenCV

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

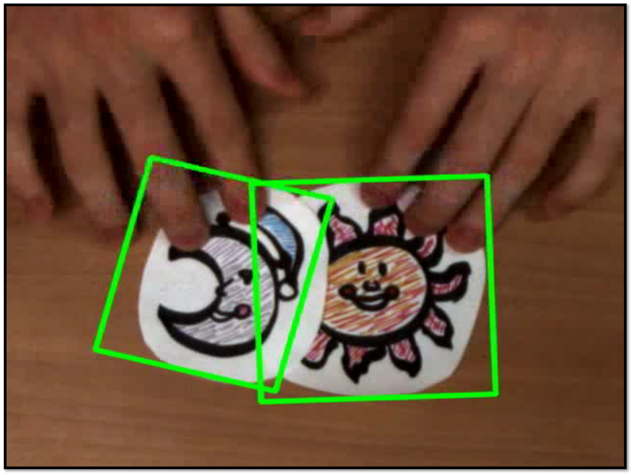
Video Puppetry: A Performative Interface for Cutout Animation. Connelly Barnes, David E. Jacobs, Jason Sanders, Dan B Goldman, Szymon Rusinkiewicz, Adam Finkelstein and Maneesh Agrawala, SIGGRAPH ASIA 2008.



Our Approach



Identification and Tracking: SIFT [Lowe 04]

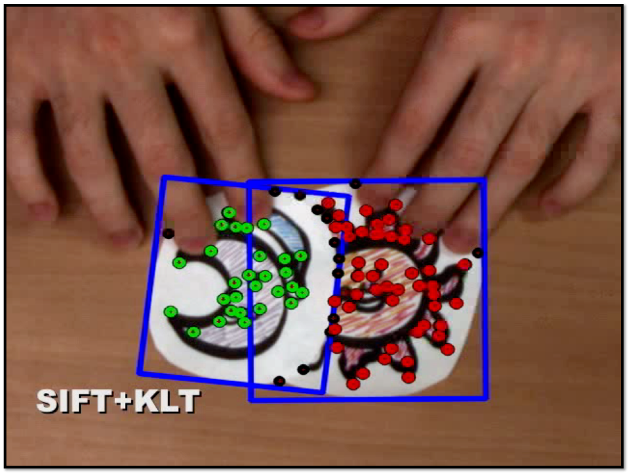


+ Identifies and locates puppets
- Not real time

Identification and Tracking: KLT [Tomasi 91]

+ Real time
- No identification

Identification and Tracking: SIFT + KLT



Group KLT points by puppet
Update transform from KLT motion
Use SIFT to correct KLT drift

Video Textures

Video Textures. Arno Schoedl, Richard Szeliski, David Salesin and Irfan Essa, SIGGRAPH 2000.

Weather Forecasting for Dummies™

Let's predict weather:

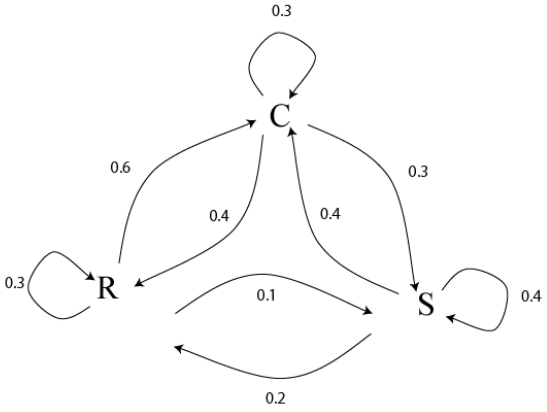
- Given today's weather only, we want to know tomorrow's
- Suppose weather can only be {Sunny, Cloudy, Raining}

The "Weather Channel" algorithm:

- Over a long period of time, record:
  - How often S followed by R
  - How often S followed by S
  - Etc.
- Compute percentages for each state:
  - P(R|S), P(S|S), etc.
- Predict the state with highest probability!
- It's a Markov Chain

Slide credit: Efros

Markov Chain



$$\begin{pmatrix} 0.3 & 0.6 & 0.1 \\ 0.4 & 0.3 & 0.3 \\ 0.2 & 0.4 & 0.4 \end{pmatrix}$$

What if we know today and yesterday's weather?

Slide credit: Efros

Text Synthesis

[Shannon,'48] proposed a way to generate English-looking text using N-grams:

- Assume a generalized Markov model
- Use a large text to compute prob. distributions of each letter given N-1 previous letters
- Starting from a seed repeatedly sample this Markov chain to generate new letters
- Also works for whole words

WE NEED TO EAT CAKE

Slide credit: Efros

Mark V. Shaney (Bell Labs)

Results (using `alt.singles` corpus):

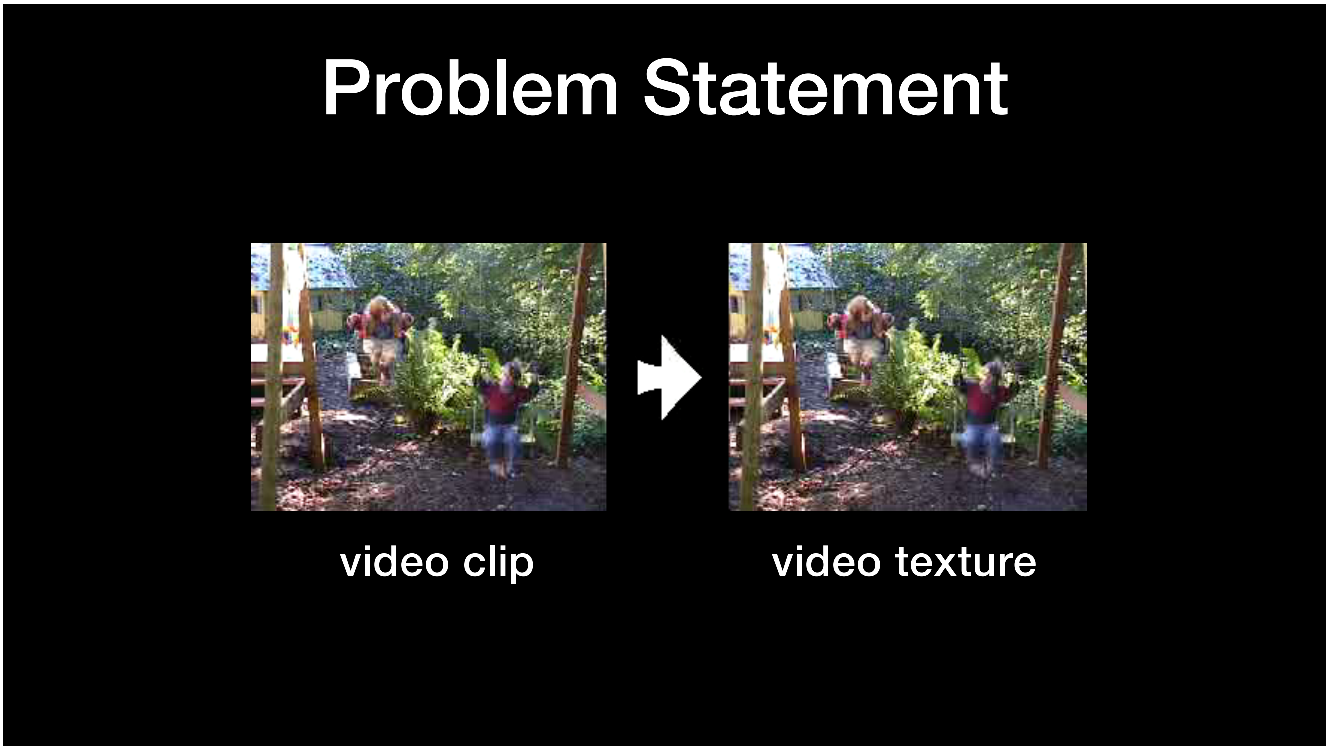
- *"As I've commented before, really relating to someone involves standing next to impossible."*
- *"One morning I shot an elephant in my arms and kissed him."*
- *"I spent an interesting evening recently with a grain of salt"*

Slide credit: Efros

Still Photos

Video Clips

Video Textures



Problem Statement

video clip

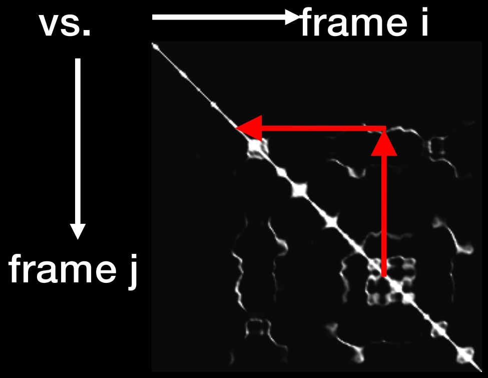
video texture

Our Approach

How do we find good transitions?

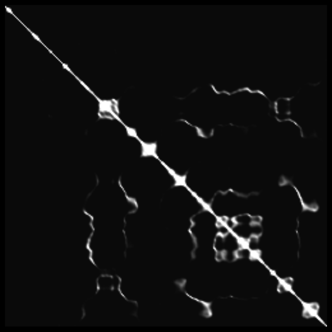
Finding Good Transitions

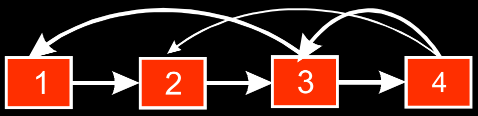
Compute L_2 distance $D_{i,j}$ between all frames



Similar frames make good transitions

Markov Chain Representation



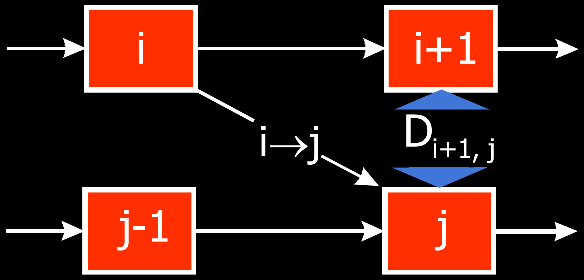


Similar frames make good transitions

Transition Costs

Transition from i to j if successor of i is similar to j

Cost function: $C_{i \to j} = D_{i+1,\, j}$



Transition Probabilities

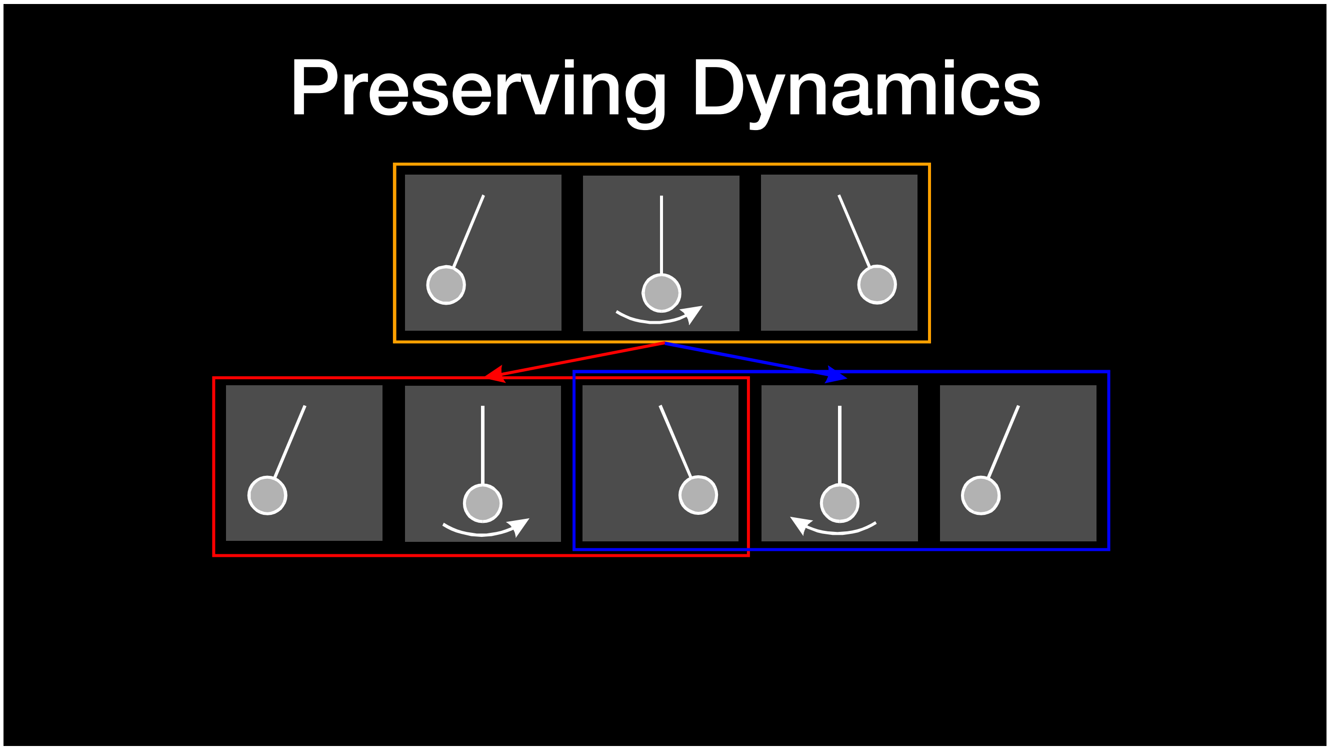
Probability for transition $P_{i \to j}$ inversely related to cost:

$$P_{i \to j} \sim \exp\left(- C_{i \to j} / \sigma^2 \right)$$

high σ

low σ

Preserving Dynamics

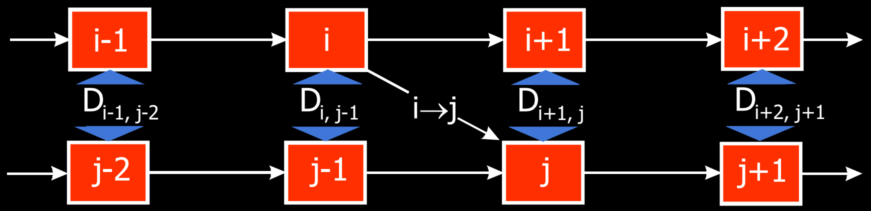


Preserving Dynamics

Preserving Dynamics

Cost for transition *i*→*j*

$$C_{i\to j} = \sum_{k=-N}^{N-1} w_k D_{i+k+1,\, j+k}$$



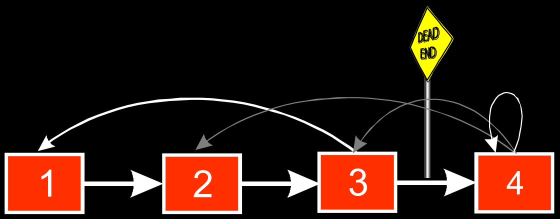
Preserving Dynamics – Effect

Cost for transition *i*→*j*

$$C_{i\to j} = \sum_{k=-N}^{N-1} w_k D_{i+k+1,\, j+k}$$

Dead Ends

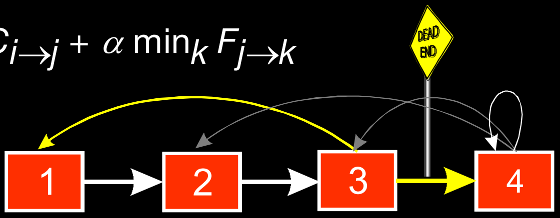
No good transition at the end of sequence

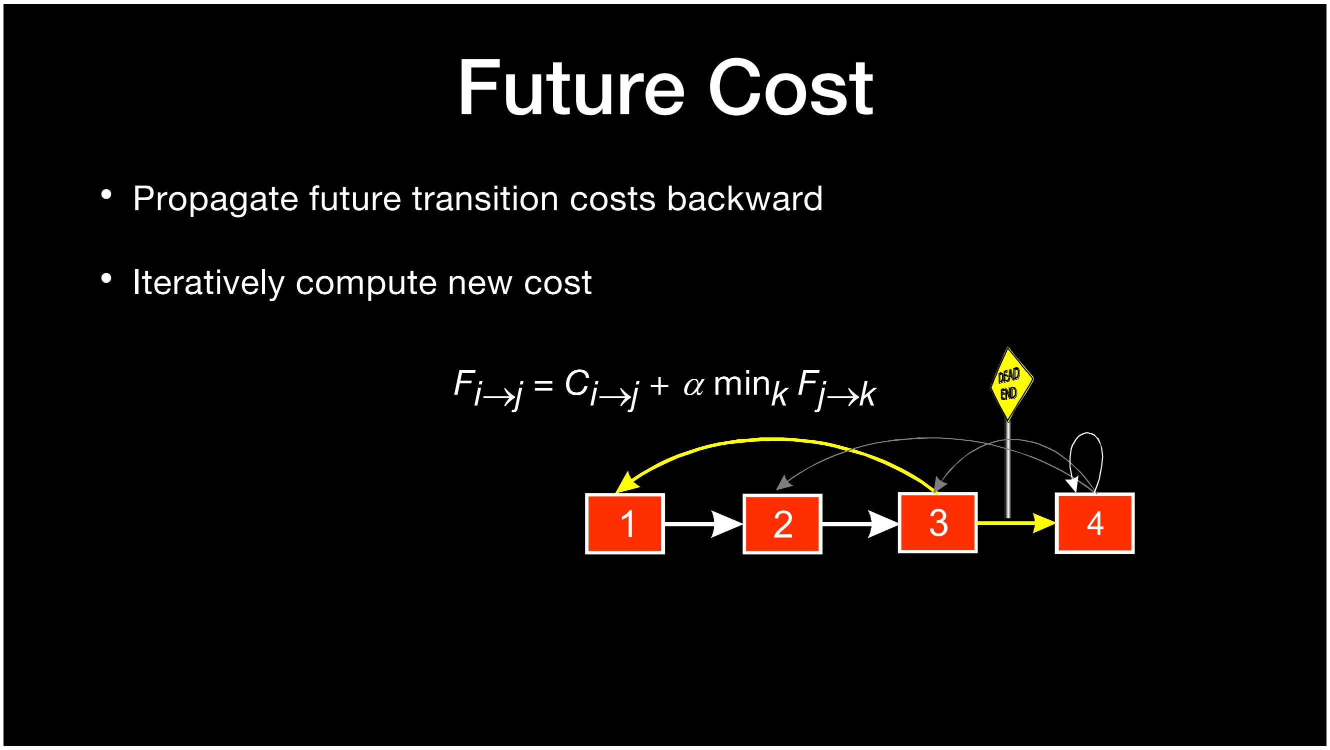


Future Cost

- Propagate future transition costs backward
- Iteratively compute new cost

$$F_{i \to j} = C_{i \to j} + \alpha \min_k F_{j \to k}$$

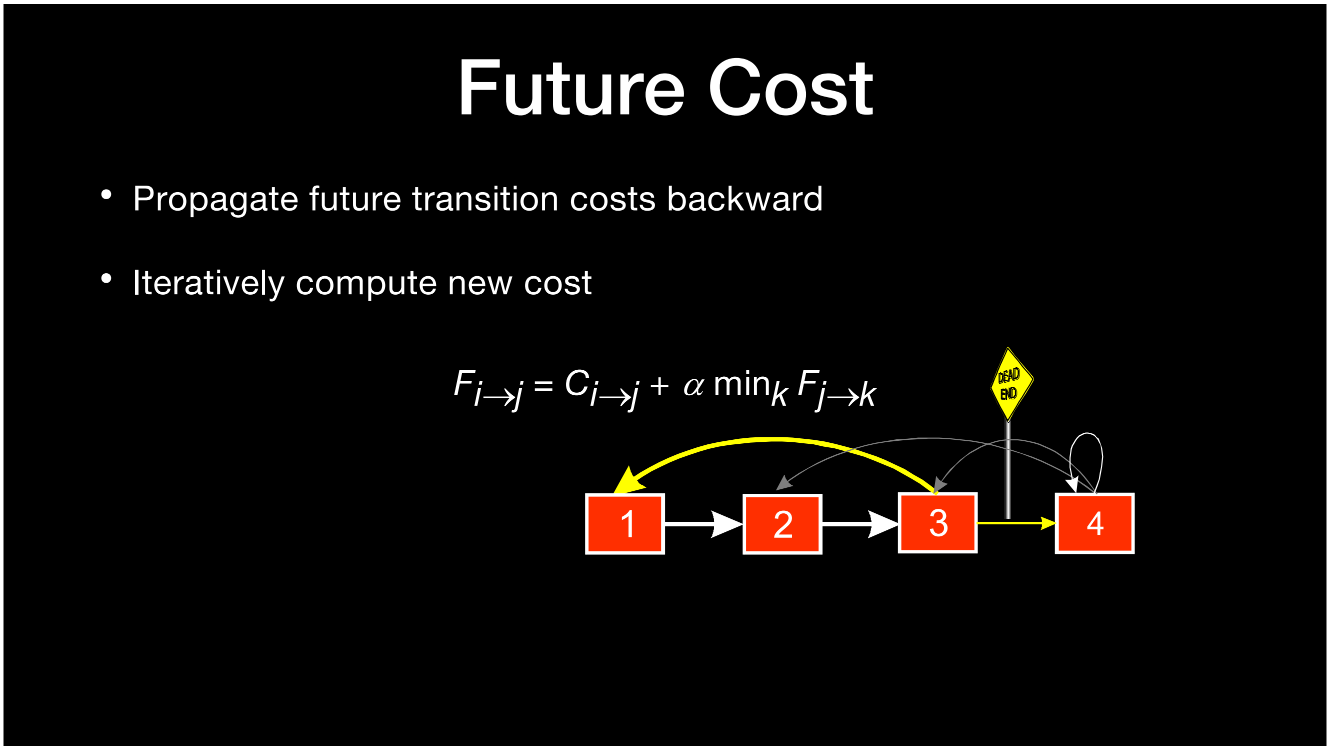




Future Cost

- Propagate future transition costs backward
- Iteratively compute new cost

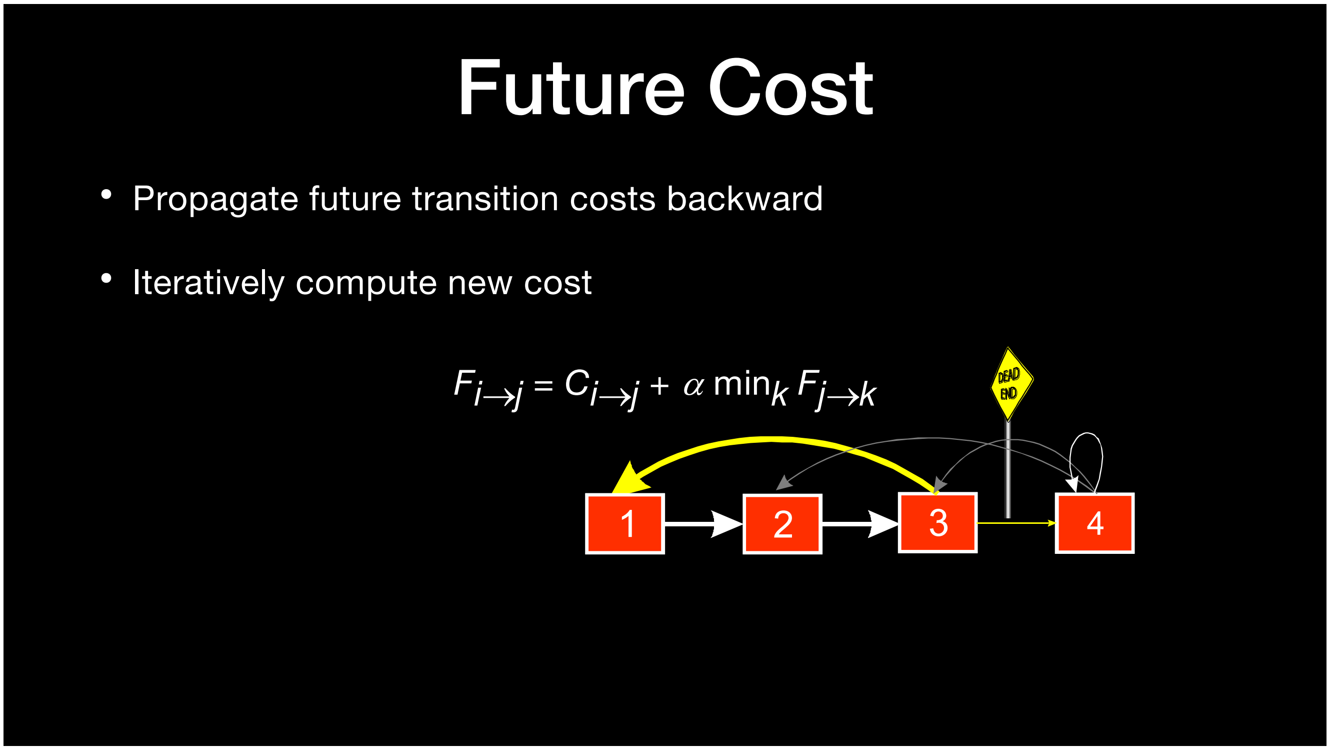
$$F_{i\to j} = C_{i\to j} + \alpha \min_k F_{j\to k}$$



Future Cost

- Propagate future transition costs backward
- Iteratively compute new cost

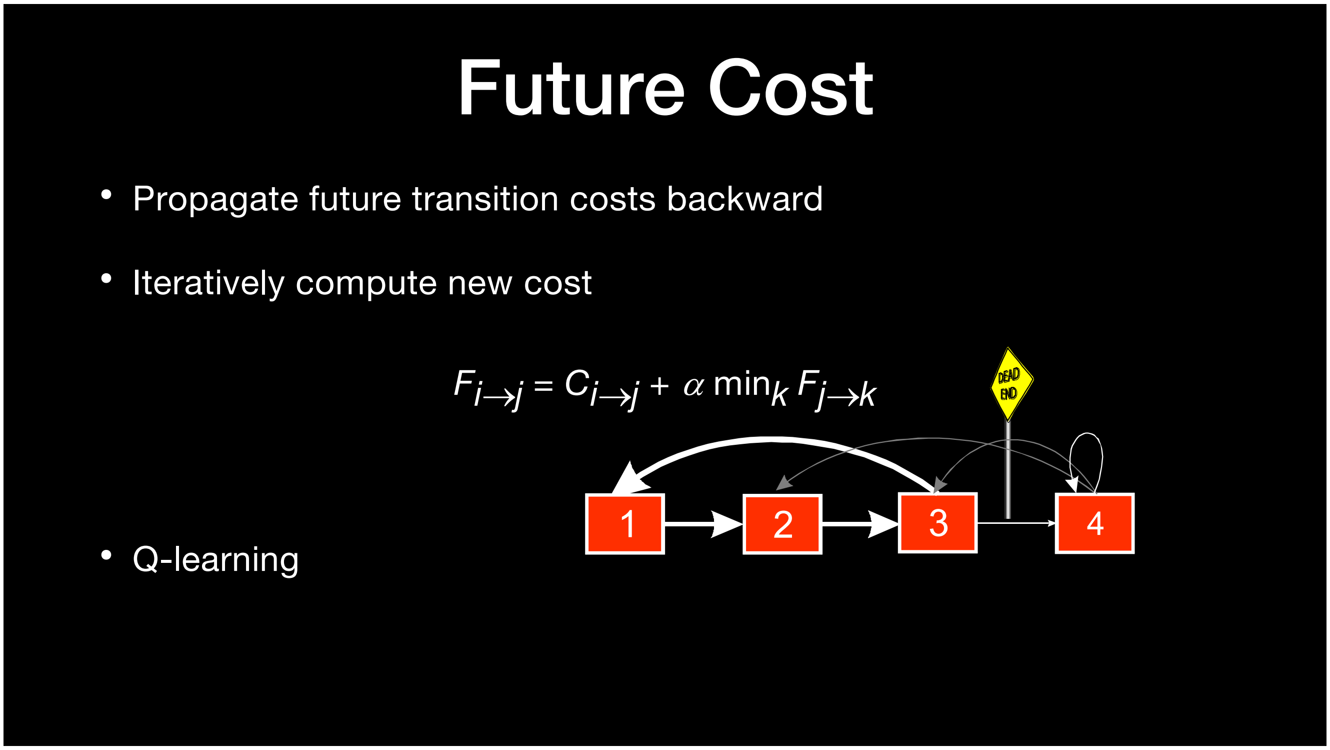
$$F_{i\to j} = C_{i\to j} + \alpha \min_k F_{j\to k}$$



Future Cost

- Propagate future transition costs backward
- Iteratively compute new cost

$$F_{i\to j} = C_{i\to j} + \alpha \min_k F_{j\to k}$$



Future Cost

- Propagate future transition costs backward
- Iteratively compute new cost

$$F_{i\to j} = C_{i\to j} + \alpha \min_k F_{j\to k}$$

- Q-learning

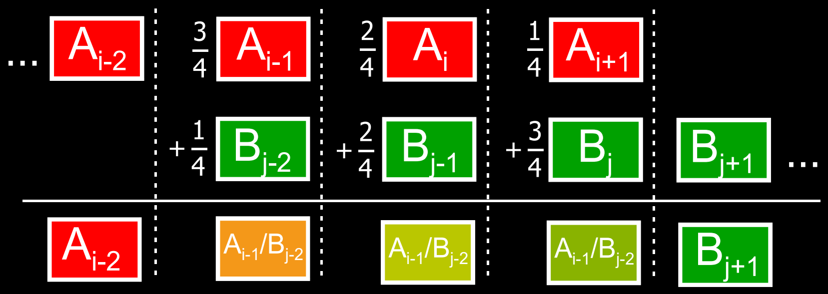
Future Cost – Effect

Visual Discontinuities

Problem: Visible “Jumps”

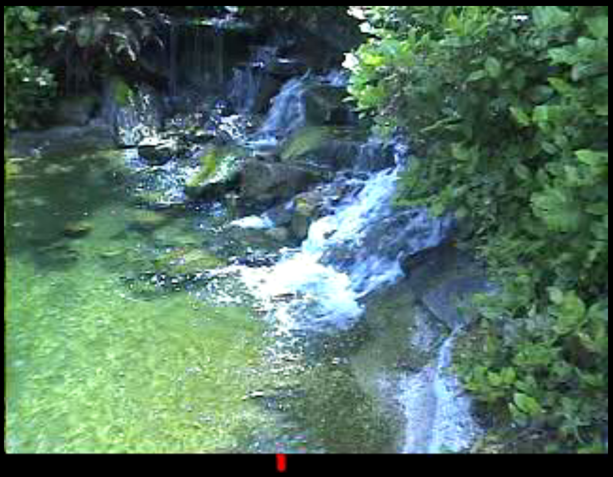
Crossfading

Solution: Crossfade from one sequence to the other



… A_{i-2}	$\frac{3}{4}$ A_{i-1}	$\frac{2}{4}$ A_i	$\frac{1}{4}$ A_{i+1}	
	$+\frac{1}{4}$ B_{j-2}	$+\frac{2}{4}$ B_{j-1}	$+\frac{3}{4}$ B_j	B_{j+1} …
A_{i-2}	A_{i-1}/B_{j-2}	A_{i-1}/B_{j-2}	A_{i-1}/B_{j-2}	B_{j+1}

Crossfading



Frequent Jump & Crossfading

Video Portrait

Useful for web pages

Video Portrait – 3D

Combine with IBR techniques

Region-Based Analysis

Divide video up into regions

Generate a video texture for each region

Automatic Region Analysis

What if motion regions overlap in space?

User-Controlled Video Textures

User selects target frame range

Time Warping

Lengthen / shorten video without affecting speed

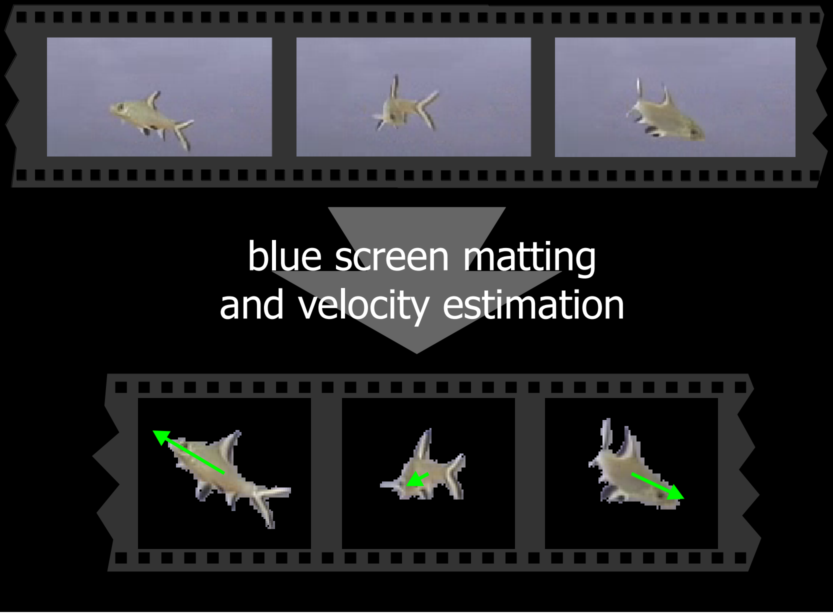
Video-Based Animation

Like sprites computer games

Extract sprites from real video

Interactively control desired motion

Video Sprite Extraction

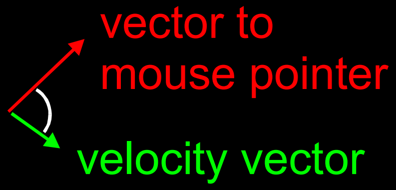


blue screen matting
and velocity estimation

Video Sprite Control

Augmented transition cost:

$$C^{\text{Animation}}_{i\to j} = \alpha \underbrace{C_{i\to j}}_{\text{Similarity term}} + \underbrace{\beta\ \text{angle}}_{\text{Control term}}$$



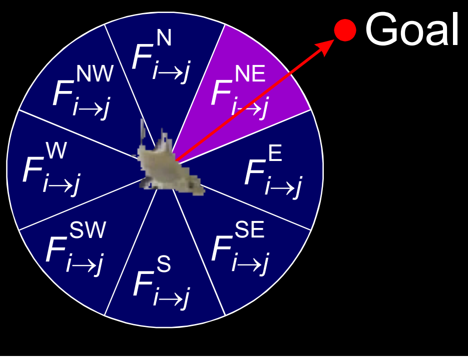
Video Sprite Control

Need future cost computation

Precompute future costs for a few angles.

Switch between precomputed angles according to user input

[GIT-GVU-00-11]



Interactive Fish

What would be required to create video sprite of a human?

Panoramic Video Textures

Panoramic
Video
Textures

Panoramic Video Textures. Aseem Agarwala, Ke Colin Zheng, Chris Pal, Maneesh Agrawala, Michael F. Cohen, Brian Curless, David Salesin, Richard Szeliski. SIGGRAPH 2005.

"Amateur" by Lasse Gjertsen

http://www.youtube.com/watch?v=JzqumbhfxRo

Michel Gondry Train Video

https://www.youtube.com/watch?v=0S43IwBF0uM