Video

Lecture 9

Overview

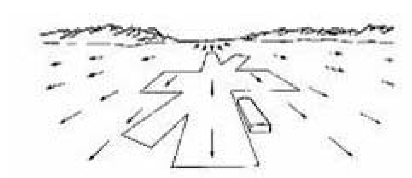
- Optical flow
- Motion Magnification
- Colorization

Overview

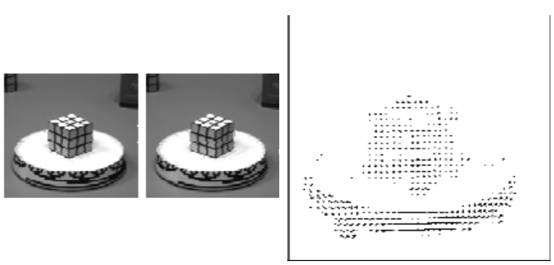
- Optical flow
- Motion Magnification
- Colorization

Optical flow

Combination of slides from Rick Szeliski, Steve Seitz, Alyosha Efros and Bill Freeman



Motion estimation: Optical flow



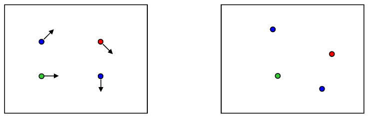
Will start by estimating motion of each pixel separately
Then will consider motion of entire image

Why estimate motion?

Lots of uses

- Track object behavior
- Correct for camera jitter (stabilization)
- Align images (mosaics)
- 3D shape reconstruction
- Special effects

Problem definition: optical flow



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

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

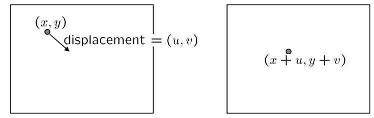
- Solve pixel correspondence problem
  - given a pixel in H, look for nearby pixels of the same color in I

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

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

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

In the limit as u and v go to zero, this becomes exact

$$0 = I_t + \nabla I \cdot [\tfrac{\partial x}{\partial t} \;\; \tfrac{\partial y}{\partial t}]$$

Optical flow equation

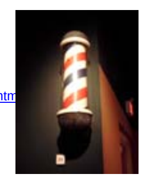
$$0 = I_t + \nabla I \cdot [u \;\; v]$$

Q: how many unknowns and equations per pixel?

2 unknowns, one equation

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

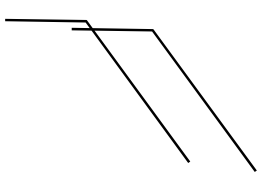


This explains the Barber Pole illusion

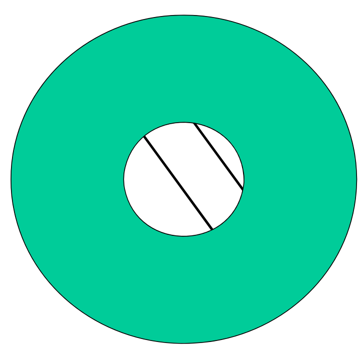
http://www.sandlotscience.com/Ambiguous/Barberpole_Illusion.htm
http://www.liv.ac.uk/~marcob/Trieste/barberpole.html

http://en.wikipedia.org/wiki/Barber's_po

Aperture problem



Aperture problem



Solving the aperture problem

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!

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

$$\underset{25\times2}{\begin{bmatrix} I_x(\mathbf{p_1}) & I_y(\mathbf{p_1}) \\ I_x(\mathbf{p_2}) & I_y(\mathbf{p_2}) \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}}) & I_y(\mathbf{p_{25}}) \end{bmatrix}}_{A} \underset{2\times1}{\begin{bmatrix} u \\ v \end{bmatrix}}_{d} = - \underset{25\times1}{\begin{bmatrix} I_t(\mathbf{p_1}) \\ I_t(\mathbf{p_2}) \\ \vdots \\ I_t(\mathbf{p_{25}}) \end{bmatrix}}_{b}$$

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!

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

$$\underset{75\times2}{\begin{bmatrix} I_x(\mathbf{p_1})[0] & I_y(\mathbf{p_1})[0] \\ I_x(\mathbf{p_1})[1] & I_y(\mathbf{p_1})[1] \\ I_x(\mathbf{p_1})[2] & I_y(\mathbf{p_1})[2] \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}})[0] & I_y(\mathbf{p_{25}})[0] \\ I_x(\mathbf{p_{25}})[1] & I_y(\mathbf{p_{25}})[1] \\ I_x(\mathbf{p_{25}})[2] & I_y(\mathbf{p_{25}})[2] \end{bmatrix}}_{A} \underset{2\times1}{\begin{bmatrix} u \\ v \end{bmatrix}}_{d} = - \underset{75\times1}{\begin{bmatrix} I_t(\mathbf{p_1})[0] \\ I_t(\mathbf{p_1})[1] \\ I_t(\mathbf{p_1})[2] \\ \vdots \\ I_t(\mathbf{p_{25}})[0] \\ I_t(\mathbf{p_{25}})[1] \\ I_t(\mathbf{p_{25}})[2] \end{bmatrix}}_{b}$$

Note that RGB is not enough to disambiguate because R, G & B are correlated

Just provides better gradient

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

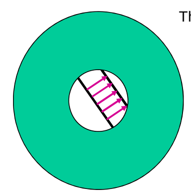
- minimum least squares solution given by solution (in d) of:

$$\underset{2\times2}{(A^T A)} \ \underset{2\times1}{d} = \underset{2\times1}{A^T b}$$

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

- The summations are over all pixels in the K x K window
- This technique was first proposed by Lukas & Kanade (1981)

Aperture Problem and Normal Flow



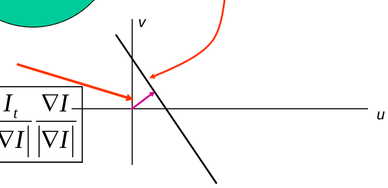
The gradient constraint:

$$\boxed{\begin{aligned} I_x u + I_y v + I_t &= 0 \\ \nabla I \bullet \vec{U} &= 0 \end{aligned}}$$

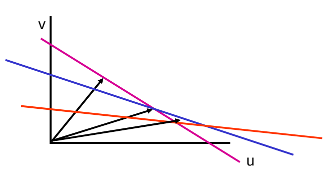
Defines a line in the *(u,v)* space

Normal Flow:

$$\boxed{u_\perp = -\frac{I_t}{|\nabla I|} \frac{\nabla I}{|\nabla I|}}$$



Combining Local Constraints



$$\nabla I^1 \bullet U = -I_t^1$$
$$\nabla I^2 \bullet U = -I_t^2$$
$$\nabla I^3 \bullet U = -I_t^3$$

etc.

Conditions for solvability

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

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

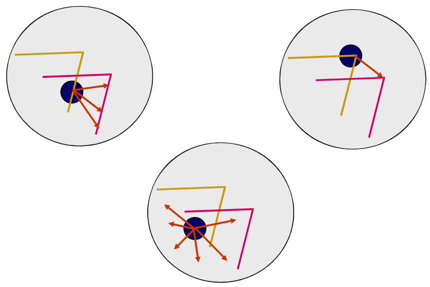
When is This Solvable?

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

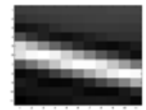
$A^T A$ is solvable when there is no aperture problem

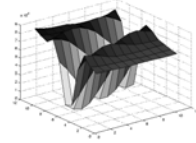
$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y] = \sum \nabla I (\nabla I)^T$$

Local Patch Analysis



Edge



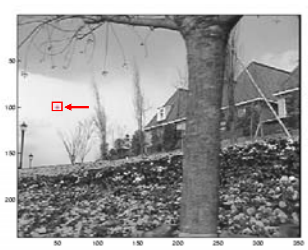


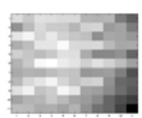
$\sum \nabla I (\nabla I)^T$

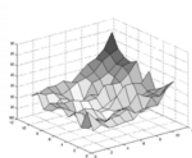
– large gradients, all the same

– large λ_1, small λ_2

Low texture region





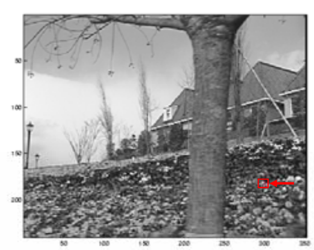


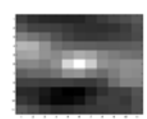
$\sum \nabla I (\nabla I)^T$

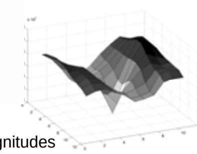
– gradients have small magnitude

– small λ_1, small λ_2

High textured region







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

– gradients are different, large magnitudes

– large λ_1, large λ_2

Observation

This is a two image problem BUT

- Can measure sensitivity by just looking at one of the images!
- This tells us which pixels are easy to track, which are hard
  - very useful later on when we do feature tracking...

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

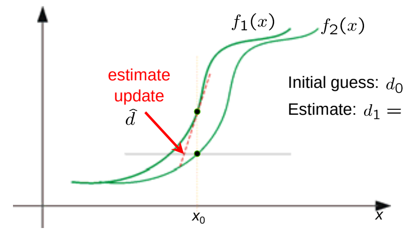
- Brightness constancy is not satisfied
- The motion is not small
- A point does not move like its neighbors
  - window size is too large
  - what is the ideal window size?

Iterative Refinement

Iterative Lukas-Kanade Algorithm

1. Estimate velocity at each pixel by solving Lucas-Kanade equations
2. Warp H towards I using the estimated flow field
   - *use image warping techniques*
3. Repeat until convergence

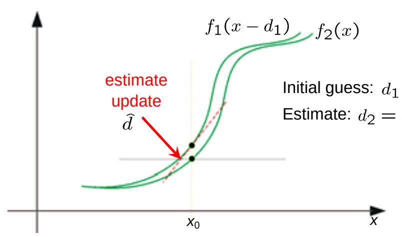
Optical Flow: Iterative Estimation



Initial guess: $d_0 = 0$
Estimate: $d_1 = d_0 + \hat{d}$

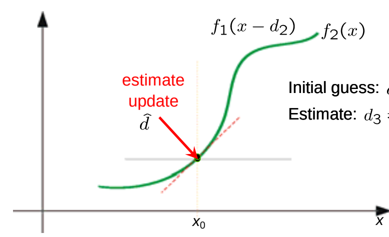
(using *d* for *displacement* here instead of *u*)

Optical Flow: Iterative Estimation



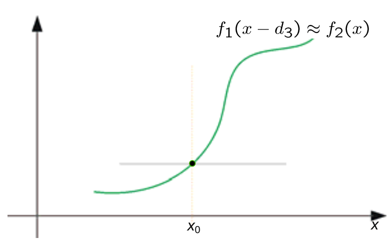
Initial guess: d_1
Estimate: $d_2 = d_1 + \hat{d}$

Optical Flow: Iterative Estimation



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

Optical Flow: Iterative Estimation



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)

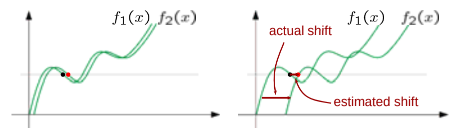
Revisiting the small motion assumption

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

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?

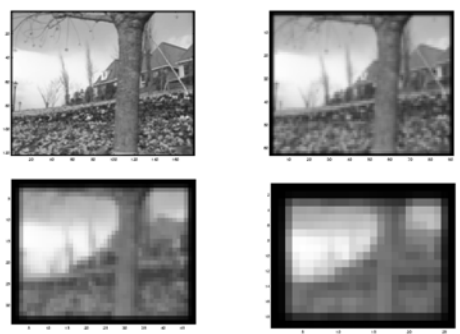


nearest match is correct (no aliasing)

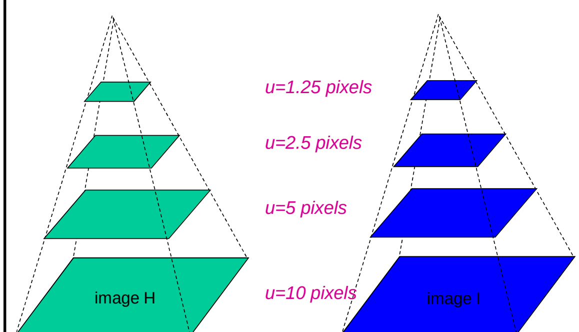
nearest match is incorrect (aliasing)

To overcome aliasing: coarse-to-fine estimation.

Reduce the resolution!



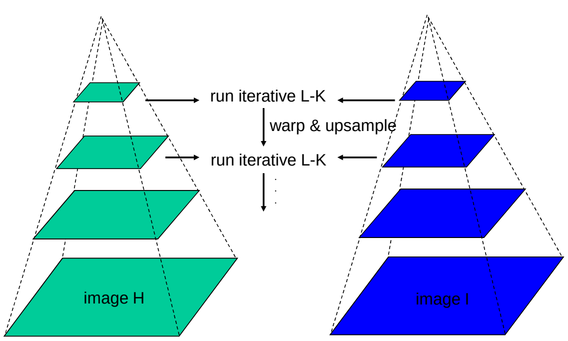
Coarse-to-fine optical flow estimation



Gaussian pyramid of image H

Gaussian pyramid of image I

Coarse-to-fine optical flow estimation



Gaussian pyramid of image H

Gaussian pyramid of image I

Beyond Translation

So far, our patch can only translate in (u,v)

What about other motion models?
- rotation, affine, perspective

Same thing but need to add an appropriate Jacobian

See Szeliski's survey of Panorama stitching

$$\mathbf{A}^{\mathbf{T}}\mathbf{A} = \sum_i \mathbf{J}\nabla\mathbf{I}(\nabla\mathbf{I})^{\mathbf{T}}\mathbf{J}^{\mathbf{T}}$$

$$\mathbf{A}^{\mathbf{T}}\mathbf{b} = -\sum_i \mathbf{J}^{\mathbf{T}} I_t (\nabla\mathbf{I})^{\mathbf{T}}$$

Recap: Classes of Techniques

Feature-based methods (e.g. SIFT+Ransac+regression)

- Extract visual features (corners, textured areas) and track them over multiple frames
- Sparse motion fields, but possibly robust tracking
- Suitable especially when image motion is large (10-s of pixels)

Direct-methods (e.g. optical flow)

- 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)

Overview

- Optical flow
- Motion Magnification
- Colorization

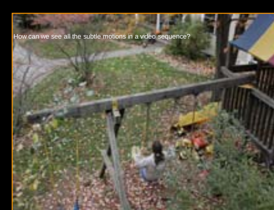
Motion Magnification

Ce Liu Antonio Torralba William T. Freeman
Frédo Durand Edward H. Adelson

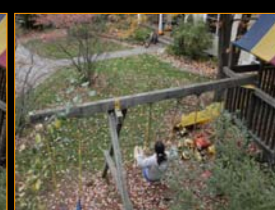
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology

SIGGRAPH2005

Motion Microscopy



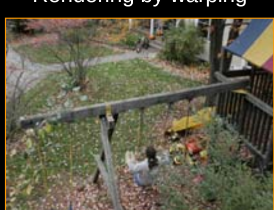
Original sequence



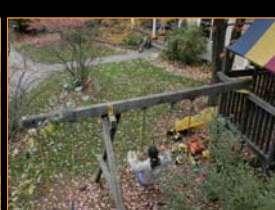
Magnified sequence

Naïve Approach

- Magnify the estimated optical flow field
- Rendering by warping

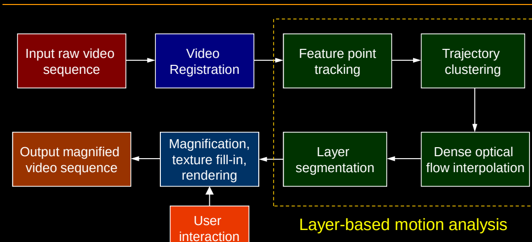


Original sequence



Magnified by naïve approach

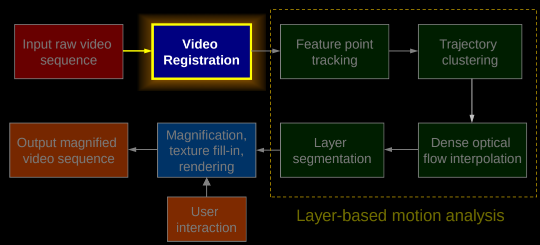
Layer-based Motion Magnification Processing Pipeline



Stationary camera, stationary background

Layer-based Motion Magnification
Video Registration

SIGGRAPH2005



Stationary camera, stationary background

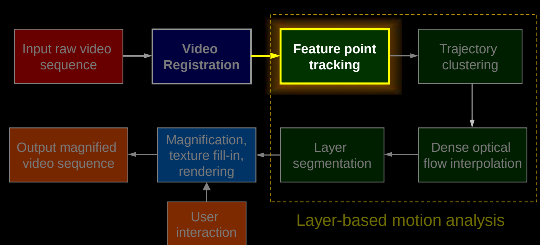
Robust Video Registration

SIGGRAPH2005

- Find feature points with Harris corner detector on the reference frame
- Brute force tracking feature points
- Select a set of robust feature points with inlier and outlier estimation (most from the rigid background)
- Warp each frame to the reference frame with a global affine transform

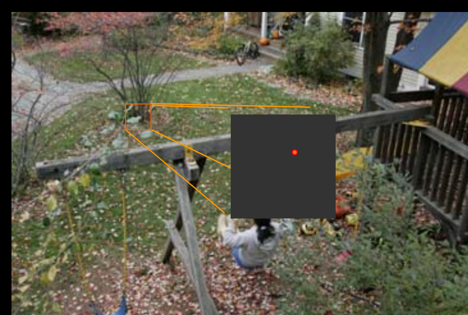
Motion Magnification Pipeline
Feature Point Tracking

SIGGRAPH2005



Challenges (1)

SIGGRAPH2005



Adaptive Region of Support

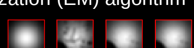
SIGGRAPH2005

- Brute force search

Confused by occlusion !

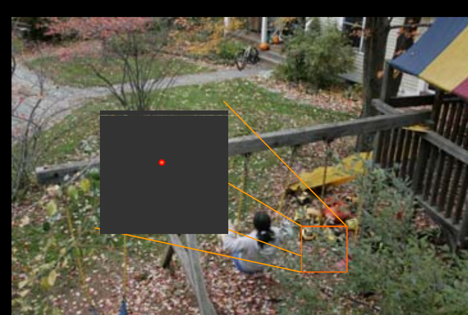
- Learn adaptive region of support using expectation-maximization (EM) algorithm

region of support



Challenges (2)

SIGGRAPH2005



Trajectory Pruning

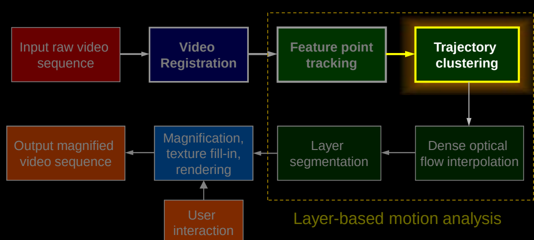
- Tracking with adaptive region of support Nonsense at full occlusion!

- Outlier detection and removal by interpolation

Comparison

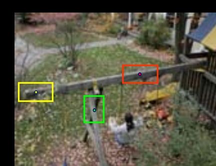
With adaptive region of support and trajectory pruning

Motion Magnification Pipeline
Trajectory Clustering

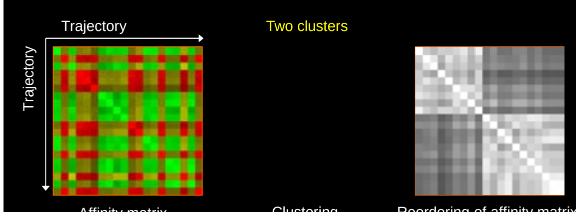


Normalized Complex Correlation

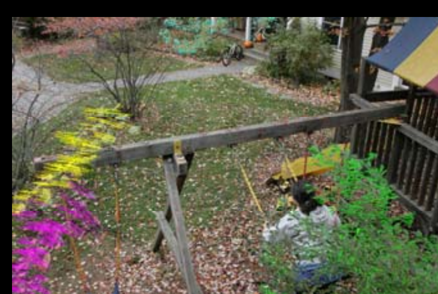
- The similarity metric should be independent of phase and magnitude
- Normalized complex correlation



Spectral Clustering

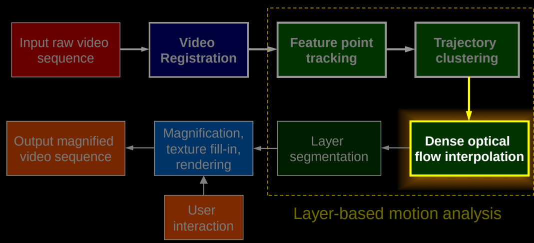


Clustering Results



Motion Magnification Pipeline
Dense Optical Flow Field

SIGGRAPH2005



From Sparse Feature Points to Dense Optical Flow Field

SIGGRAPH2005

- Interpolate dense optical flow field using locally weighted linear regression

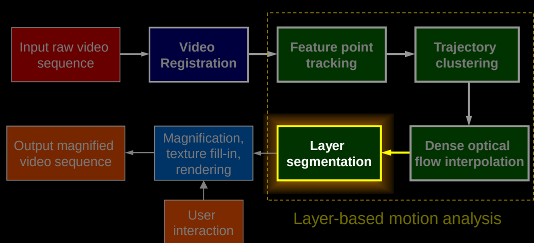
Dense optical flow field of sparse (feature) points

Cluster 1: leaves

Cluster 2: swing

Motion Magnification Pipeline
Layer Segmentation

SIGGRAPH2005



Motion Layer Assignment

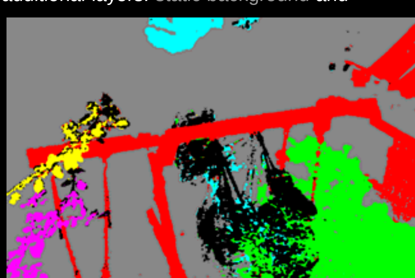
SIGGRAPH2005

- Assign each pixel to a motion cluster layer, using four cues:
  - **Motion likelihood**—consistency of pixel's intensity if it moves with the motion of a given layer (dense optical flow field)
  - **Color likelihood**—consistency of the color in a layer
  - **Spatial connectivity**—adjacent pixels favored to belong the same group
  - **Temporal coherence**—label assignment stays constant over time
- Energy minimization using graph cuts

Segmentation Results

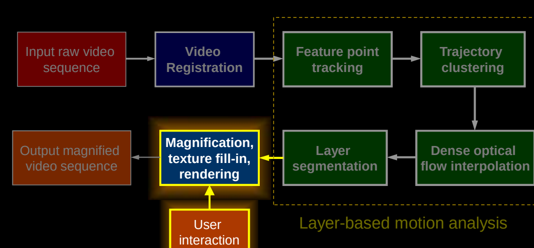
SIGGRAPH2005

- Two additional layers: static background and



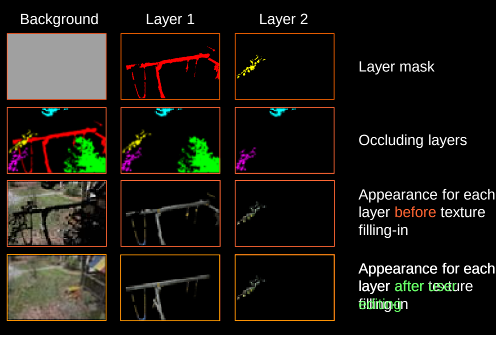
Motion Magnification Pipeline
Editing and Rendering

SIGGRAPH2005



Layered Motion Representation for Motion Processing

SIGGRAPH2005



Video

SIGGRAPH2005

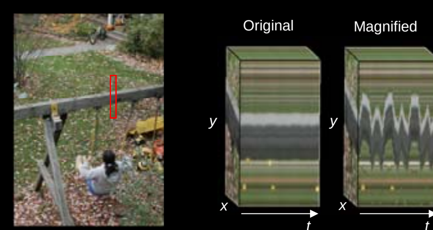
Motion Magnification

Is the Baby Breathing?

SIGGRAPH2005

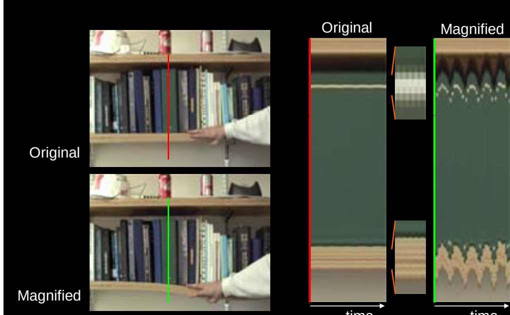
Are the Motions Real?

SIGGRAPH2005



Are the Motions Real?

SIGGRAPH2005



Applications

SIGGRAPH2005

- Education
- Entertainment
- Mechanical engineering
- Medical diagnosis

Conclusion

SIGGRAPH2005

- Motion magnification
  - A motion microscopy technique
- Layer-based motion processing system
  - Robust feature point tracking
  - Reliable trajectory clustering
  - Dense optical flow field interpolation
  - Layer segmentation combining multiple cues

Thank you!

SIGGRAPH2005

Motion Magnification

Ce Liu Antonio Torralba William T. Freeman Frédo Durand Edward H. Adelson

Computer Science and Artificial Intelligence Laboratory

Massachusetts Institute of Technology

Overview

- Optical flow
- Motion Magnification
- Colorization

Colorization Using Optimization

Anat Levin Dani Lischinski Yair Weiss

School of Computer Science and Engineering

The Hebrew University of Jerusalem, Israel

Colorization

Colorization: a computer-assisted process of adding color to a monochrome image or movie. (Invented by Wilson Markle, 1970)

Motivation

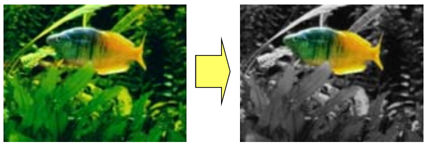
- **Colorizing black and white movies and TV shows**

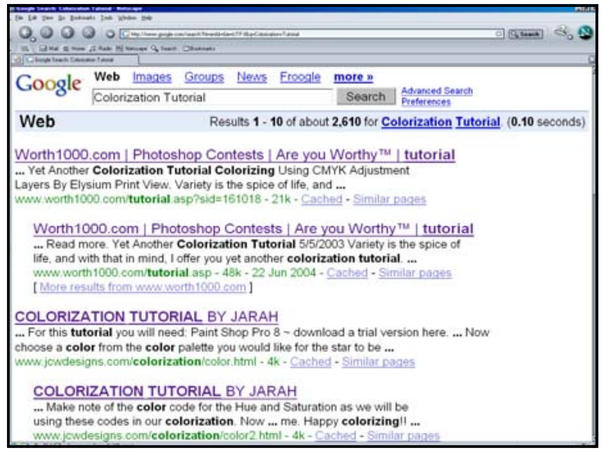
Earl Glick (Chairman, Hal Roach Studios), 1984:
"You couldn't make Wyatt Earp today for $1 million an episode. But for $50,000 a segment, you can turn it into color and have a brand new series with no residuals to pay"

Motivation

- Colorizing black and white movies and TV shows

- Recoloring color images for special effects





Typical Colorization Process

Images from:
"Yet Another Colorization Tutorial"
http://www.worth1000.com/tutorial.asp?sid=161018

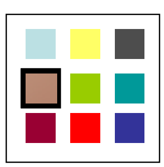
Typical Colorization Process

- Delineate region boundary

Images from:
"Yet Another Colorization Tutorial"
http://www.worth1000.com/tutorial.asp?sid=161018

Typical Colorization Process

- Delineate region boundary
- Choose region color from palette.



Images from:
"Yet Another Colorization Tutorial"
http://www.worth1000.com/tutorial.asp?sid=161018

Typical Colorization Process

- Delineate region boundary
- Choose region color from palette.

Images from:
"Yet Another Colorization Tutorial"
http://www.worth1000.com/tutorial.asp?sid=161018

Typical Colorization Process

- Delineate region boundary
- Choose region color from palette.

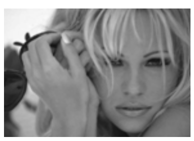
Images from:
"Yet Another Colorization Tutorial"

http://www.worth1000.com/tutorial.asp?sid=161018

Video Colorization Process

- Delineate region boundary
- Choose region color from palette.
- Track regions across video frames

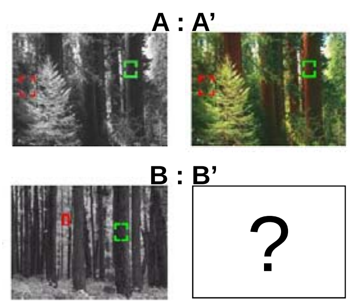
Colorization Process Discussion



Time consuming and labor intensive

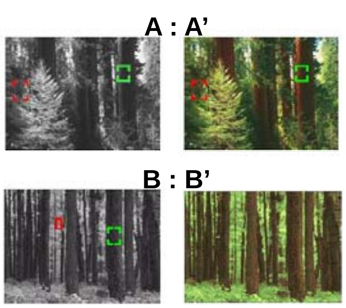
- Fine boundaries.
- Failures in tracking.

Colorization by Analogy



Hertzmann et al. 2001, Welsh et al. 2002

Colorization by Analogy



Hertzmann et al. 2001, Welsh et al. 2002

Colorization by Analogy - Discussion

- Indirect artistic control
- No spatial continuity constraint

Our approach

Our approach

Artist scribbles desired colors inside regions

Our approach

Colors are propagated to all pixels

"Nearby pixels with similar intensities should have the same color"

Propagation using Optimization

$$Y \Rightarrow U, V$$

Intensity channel

Color channels

"Neighboring pixels with similar intensities should have similar colors"

Propagation using Optimization

$$Y \Rightarrow U, V$$

Intensity channel

Color channels

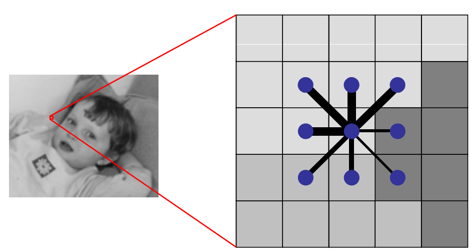
$$\boxed{J(U) = \sum_r \left(U(r) - \sum_{s \in N(r)} w_{rs} U(s) \right)^2}$$

- Minimize difference between color at a pixel and an *affinity-weighted average* of the neighbors

Affinity Functions

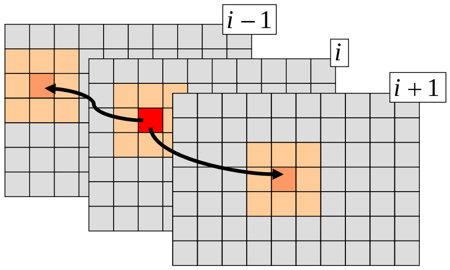
$$w_{rs} \propto e^{-(Y(r)-Y(s))^2/\sigma_r^2}$$

σ_r proportional to local variance



Affinity Functions in Space-Time

$$w_{rs} \propto e^{-(Y(r)-Y(s))^2/2\sigma_r^2}$$



Minimizing cost function

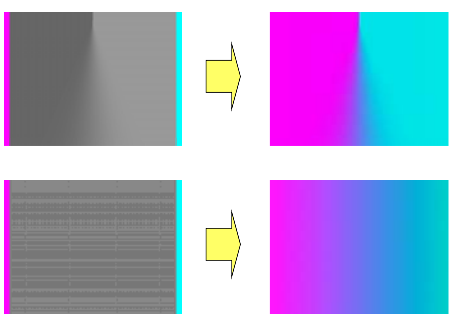
Minimize:

$$J(U) = \sum_r \left(U(r) - \sum_{s \in N(r)} w_{rs} U(s) \right)^2$$

Subject to ***labeling constraints***

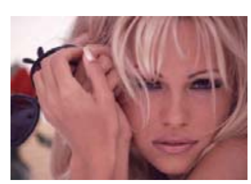
Since cost is quadratic, minimum can be found by solving sparse system of linear equations.

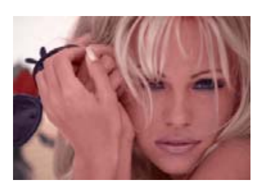
Color Interpolation



Coloring Stills

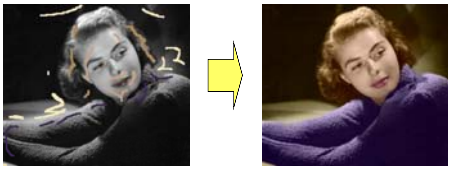
Coloring Stills



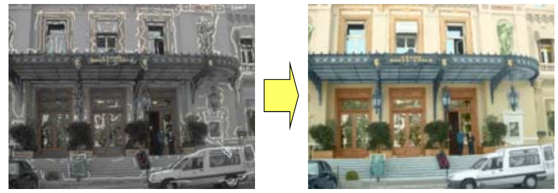


Original **Colorized**

Coloring Stills

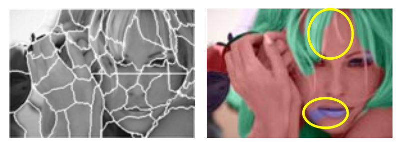


Coloring Stills



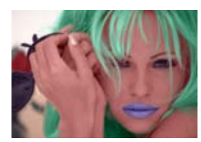
Colorization Challenges

Segmentation?



NCuts Segmentation
(Shi & Malik 97)

Segmentation aided
colorization



Our result

Recoloring

Affinity between pixels – based on intensity AND color similarities.

Recoloring

Recoloring

c.f. “Poisson image editing” Perez et al. SIGGRAPH 2003

Colorizing Video

13 out of 92 frames

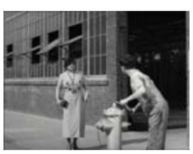
Colorizing Video

16 out of 101 frames

Matting as Colorization

Red channel<->matte

Matting as Colorization



Future Work:

- **Import image segmentation advantages: affinity functions, optimization techniques.**
- **Alternative color spaces, propagating hue and saturation differently**

Summary

- Interface: User scribbles color on a small number of pixels
- Colors propagate in space-time volume respecting intensity boundaries
- Convincing colorization with a small amount of user effort

Code & examples available:

www.cs.huji.ac.i/~yweiss/Colorization/