计算机视觉

邬向前

计算学部

多模态智能及应用研究中心

电子邮箱: xqwu@hit.edu.cn

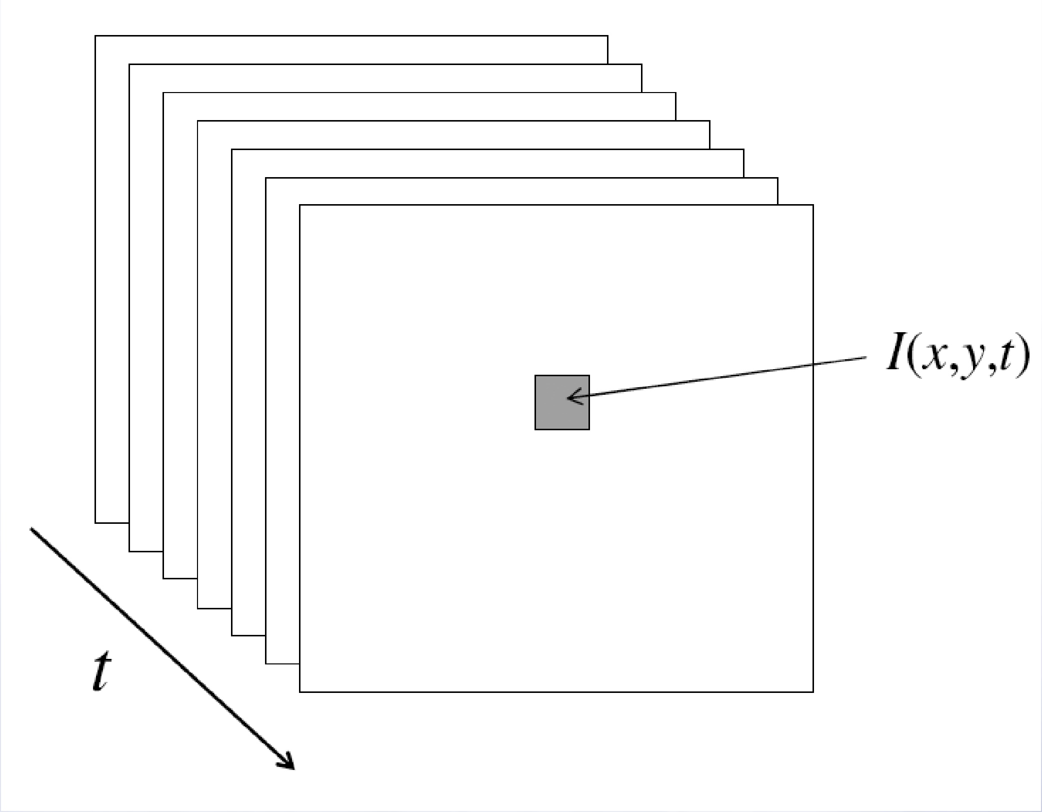
Lecture 7 – Optical Flow

Overview

- Segmentation in Video
- Optical flow

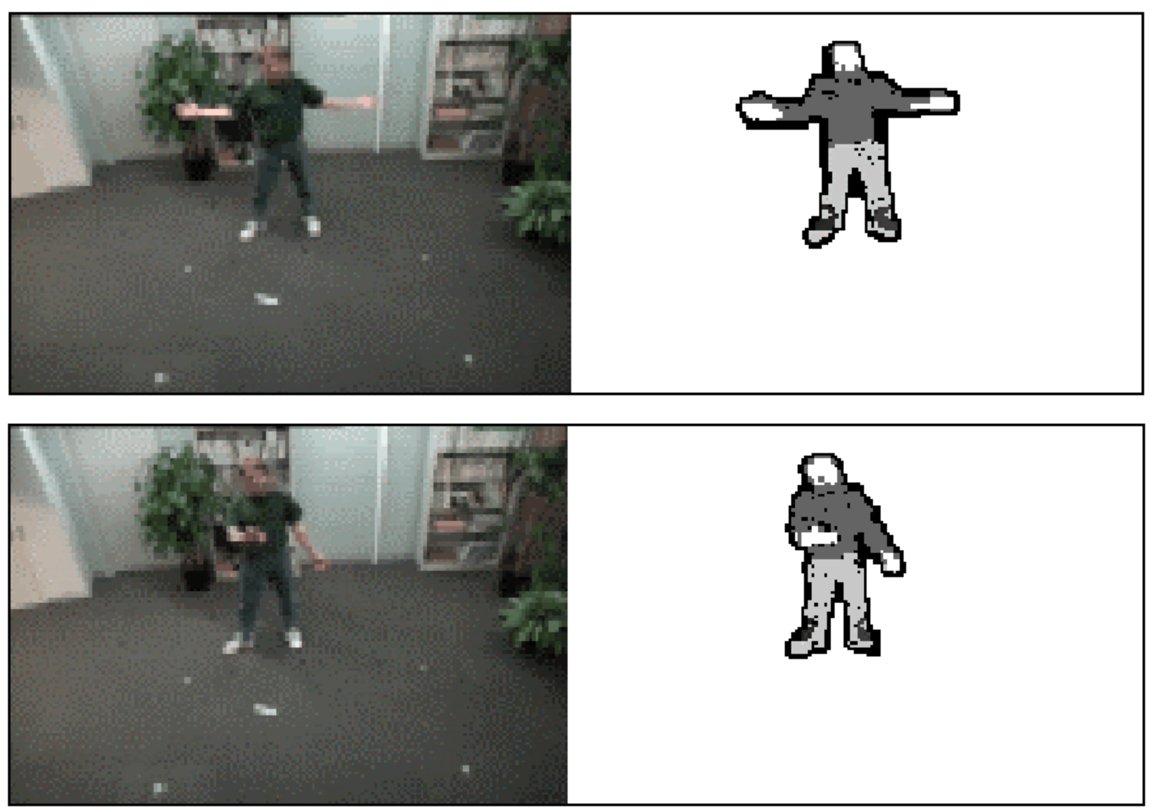
Video

- A video is a sequence of frames captured over time
- Now our image data is a function of space (x, y) and time (t)



Applications of segmentation to video

- Background subtraction
  - A static camera is observing a scene
  - Goal: separate the static *background* from the moving *foreground*



Applications of segmentation to video

- Background subtraction
  - Form an initial background estimate

Calculate background: Average a series of preceding images.

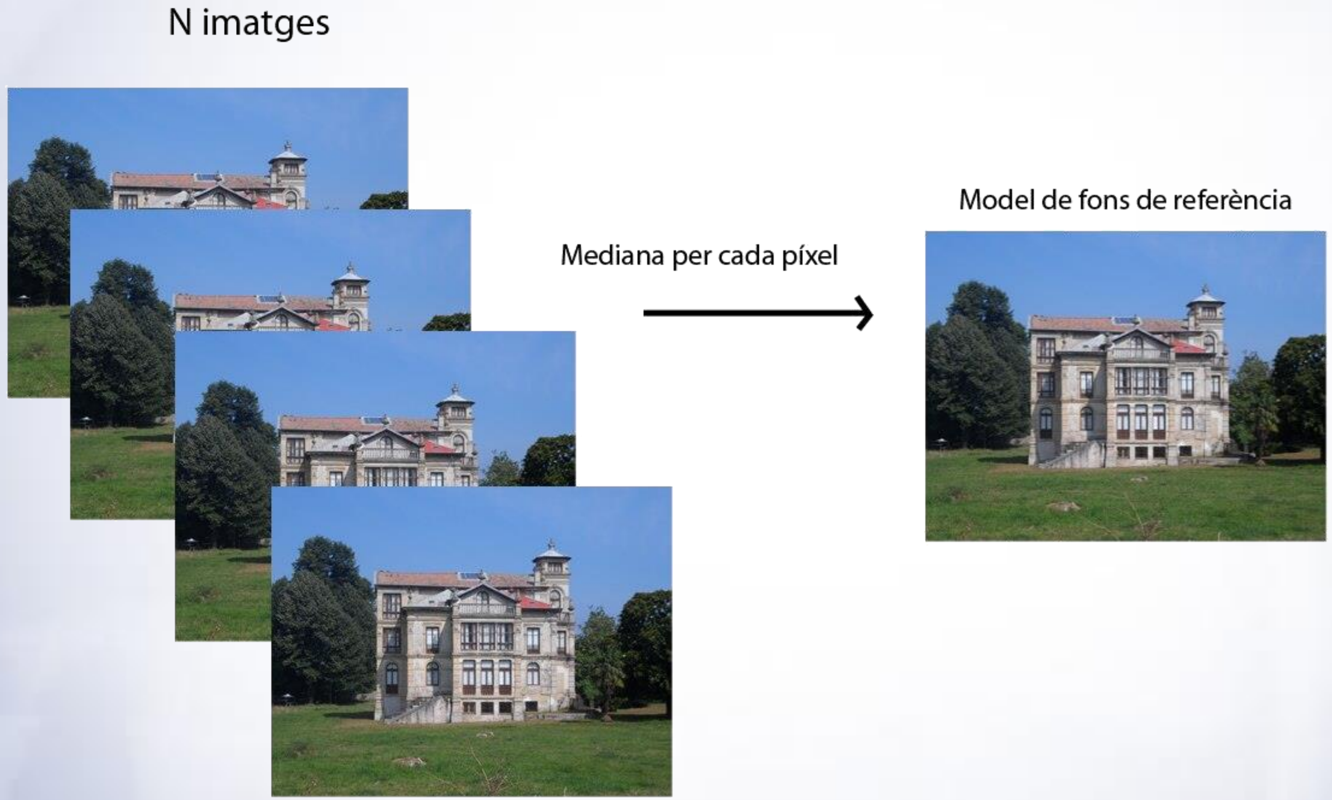
$$\frac{1}{N}\sum_{i=1}^{N} V(x,y,t-i)$$

Background image at time t:

$$B(x,y,t)=\frac{1}{N}\sum_{i=1}^{N} V(x,y,t-i)$$

Applications of segmentation to video

- Background subtraction



Applications of segmentation to video

- Background subtraction
  - Form an initial background estimate

$$B(x,y,t) = \frac{1}{N}\sum_{i=1}^{N} V(x,y,t-i)$$

  - Subtract the background estimate from the frame

$$V(x,y,t) - B(x,y,t)$$

  - Label as foreground each pixel where the magnitude of the difference >Th

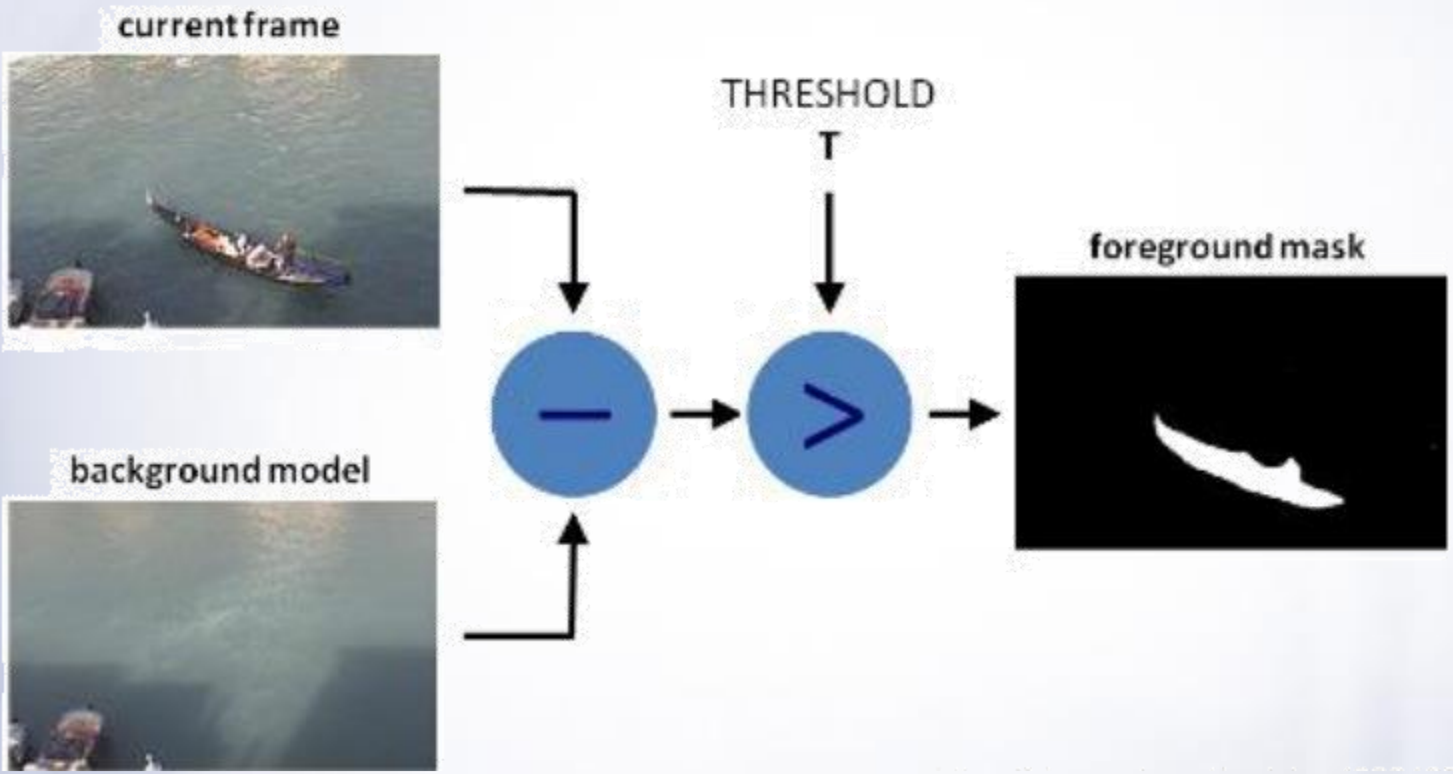
$$|V(x,y,t) - B(x,y,t)| > \mathrm{Th}$$

Applications of segmentation to video

- Background subtraction
  - Form an initial background estimate
  - For each frame:
    - Update estimate using a moving average
    - Subtract the background estimate from the frame
    - Label as foreground each pixel where the magnitude of the difference is greater than some threshold
    - Use median filtering to “clean up” the results

Applications of segmentation to video

- Background subtraction

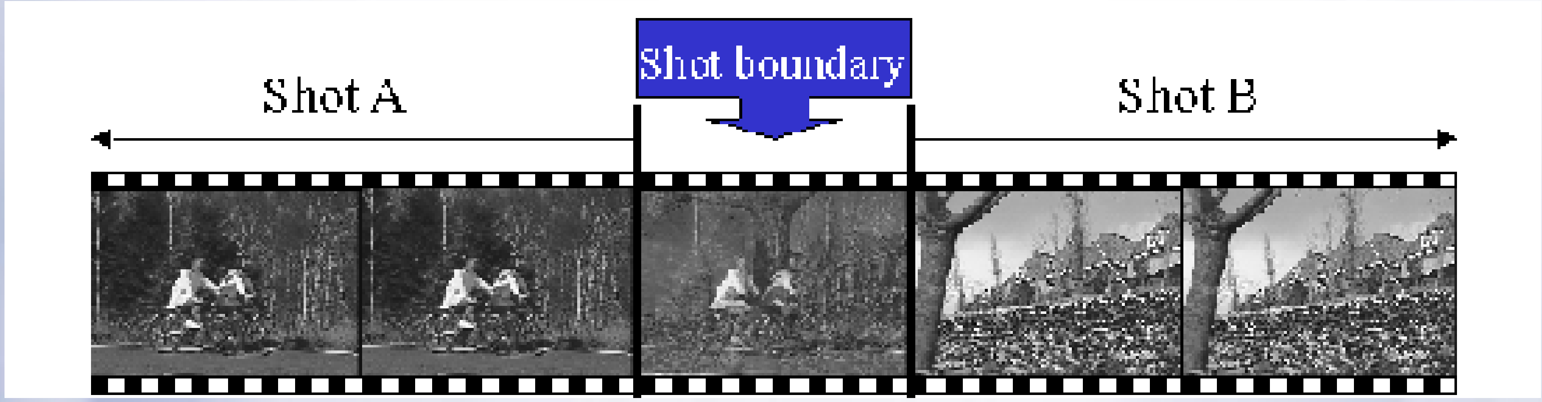


Applications of segmentation to video

- Background subtraction
  - Pros:
    - Simple;
    - To some extent, overcomes the influence of environmental light;
  - Cons:
    - Can not be used for moving Cameras;
    - Difficult to update the background image in real time

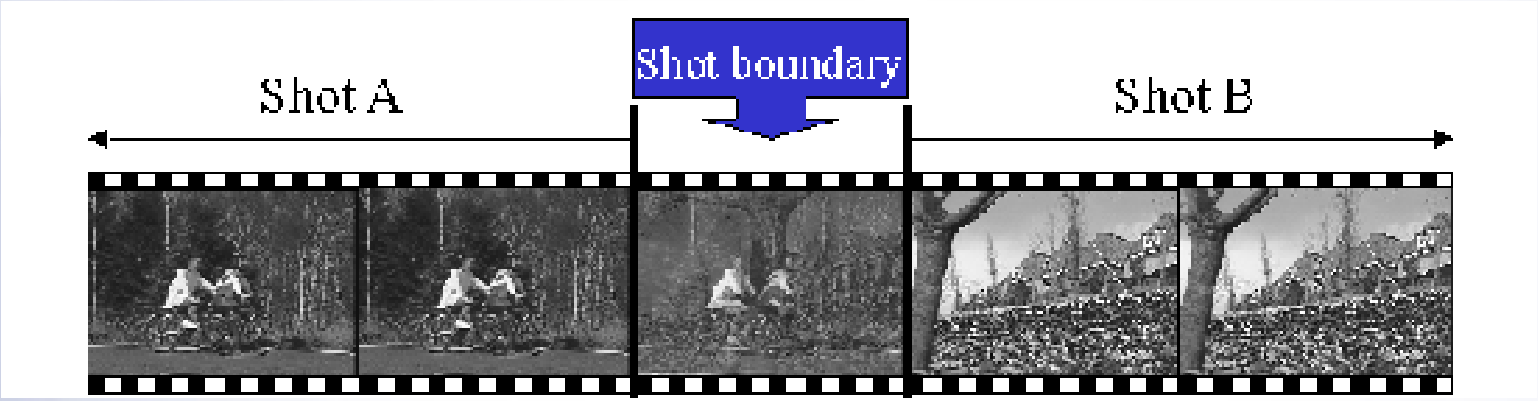
Applications of segmentation to video

- Background subtraction
- Shot boundary detection
  - Commercial video is usually composed of *shots* or sequences showing the same objects or scene
  - Goal: segment video into shots for summarization and browsing (each shot can be represented by a single keyframe in a user interface)
  - Difference from background subtraction: the camera is not necessarily stationary



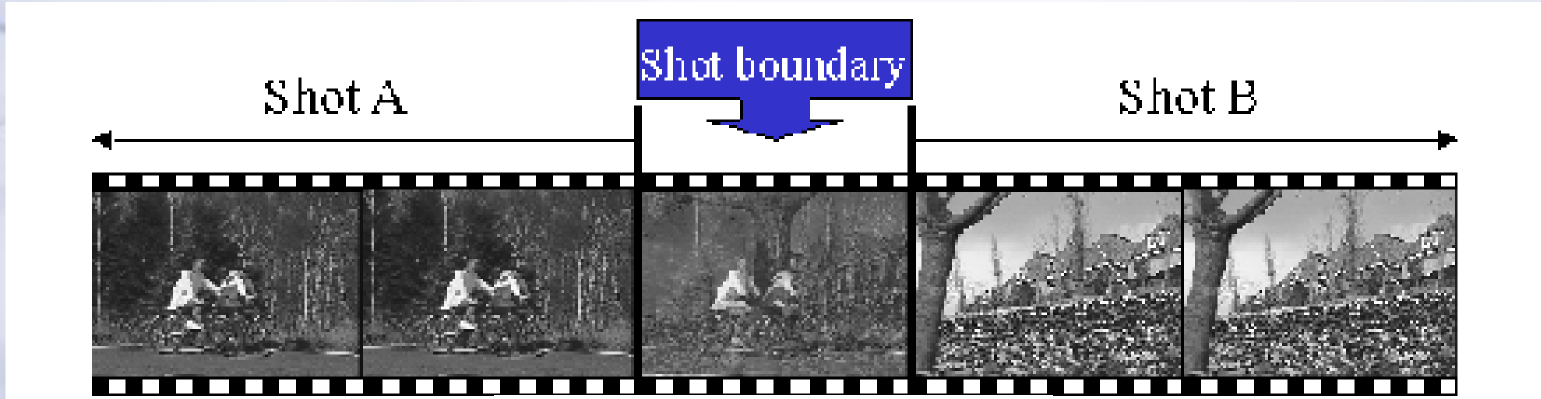
Applications of segmentation to video

- Background subtraction
- Shot boundary detection



a sudden transition from one shot to another

Applications of segmentation to video

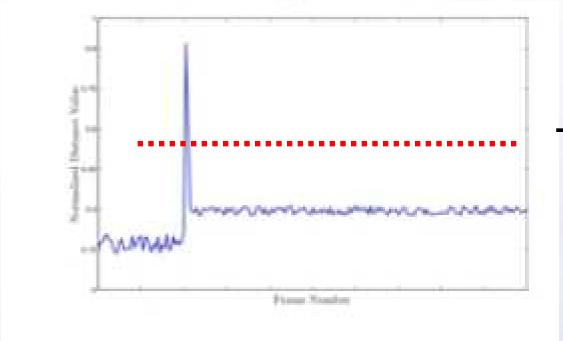


$$Z_n = \Psi(f_n)$$

$$d(f_n, f_{n+1}) = \left(\sum_{k=1} |Z_n(k) - Z_{n+1}(k)|^p \right)^{1/p}$$

fn： video frame, Ψ： feature extraction function

Zn： extracted features of video frame fn



Threshold

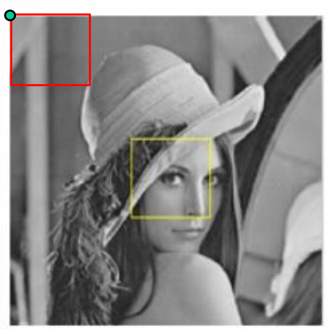
Applications of segmentation to video

- Background subtraction
- Shot boundary detection
  - For each frame
    - Compute the distance between the current frame and the previous one
      - » Mean absolute differences (MAD)

Similarity between two consecutive frames pixels

$from\ (i,j)\ search$

$m * n$



$S(x,y)$

$\mathrm{T}(x,y)$

$M * N$

Measure of similarity: $$D(i,j) = \frac{1}{M \times N}\sum_{s=1}^{M}\sum_{t=1}^{N}|S(i+s-1, j+t-1) - T(s,t)|$$

$$1 \le i \le m - M + 1 \qquad 1 \le j \le n - N + 1$$

Applications of segmentation to video

- Background subtraction
- Shot boundary detection
  - For each frame
    - Compute the distance between the current frame and the previous one
      - » Histogram differences (HD)

Similarity between the histograms of two consecutive frames

$$d(H_1,H_2)=\frac{\sum_I (H_1(I)-\bar{H}_1)(H_2(I)-\bar{H}_2)}{\sqrt{\sum_I (H_1(I)-\bar{H}_1)^2 \sum_I (H_2(I)-\bar{H}_2)^2}}$$

$$\bar{H}_k=\frac{1}{N}\sum_J H_k(J)$$

Correlation：

$$r(X,Y)=\frac{Cov(X,Y)}{\sqrt{Var[X]\,Var[Y]}}$$

$d(H_1,H_2)\rightarrow 1$，strong Correlation.

Applications of segmentation to video

- Background subtraction
- Shot boundary detection
  - For each frame
    - Compute the distance between the current frame and the previous one
      - » Mean absolute differences (MAD)
      - » Histogram differences (HD)
    - If the distance is greater than some threshold, classify the frame as a shot boundary

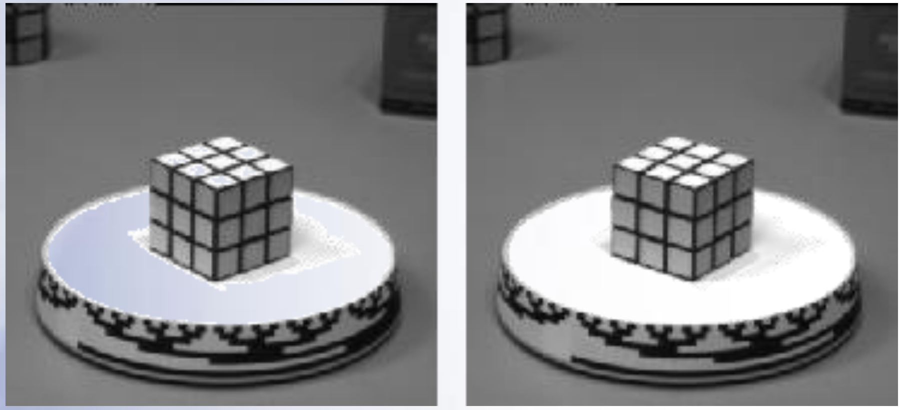
Applications of segmentation to video

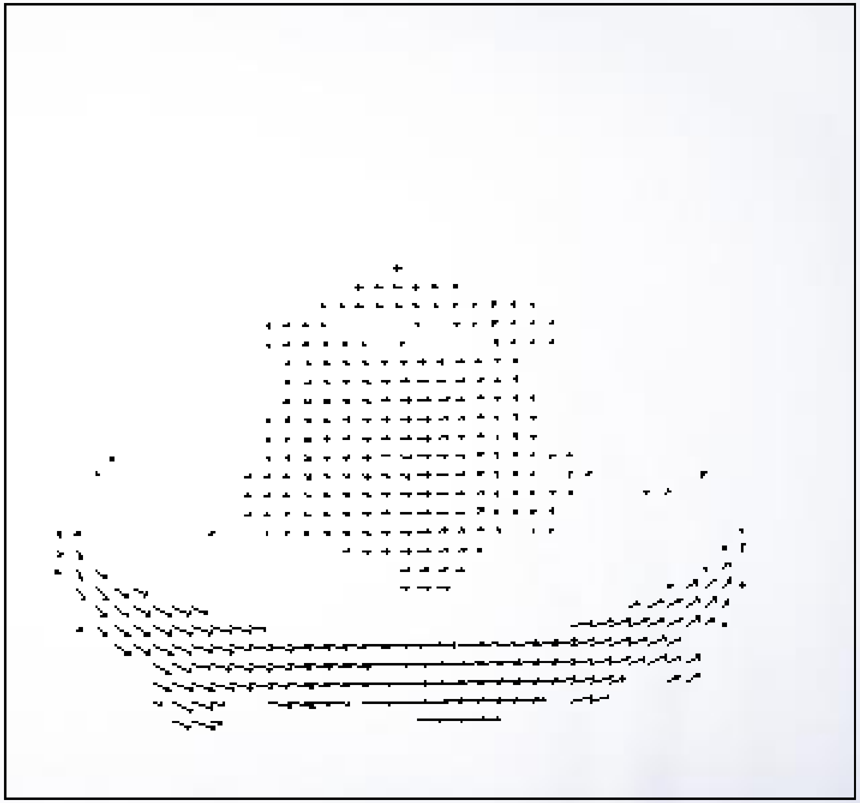
- Background subtraction
- Shot boundary detection
- Motion Segmentation

Overview

- Segmentation in Video
- Optical flow
  - Lucas-kanade
  - Horn-schunck
  - Optical flow pyramid

Motion estimation: Optical flow





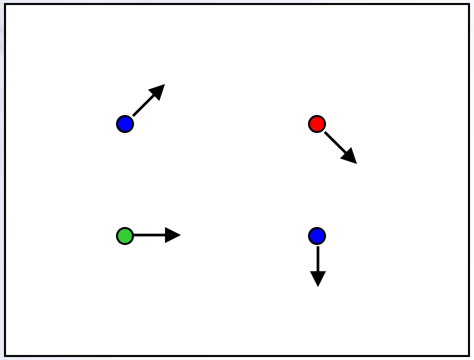
Will start by estimating motion of each pixel separately
Then will consider motion of entire image

Why estimate motion?

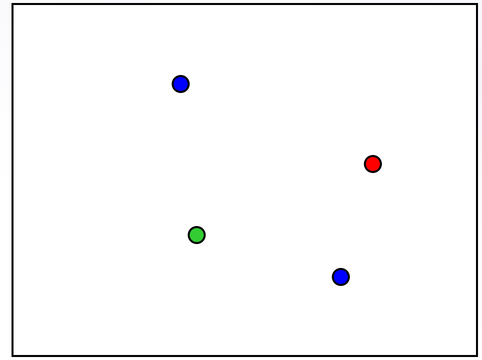
Lots of uses

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

Problem definition: optical flow



$H(x, y)$



$I(x, y)$

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

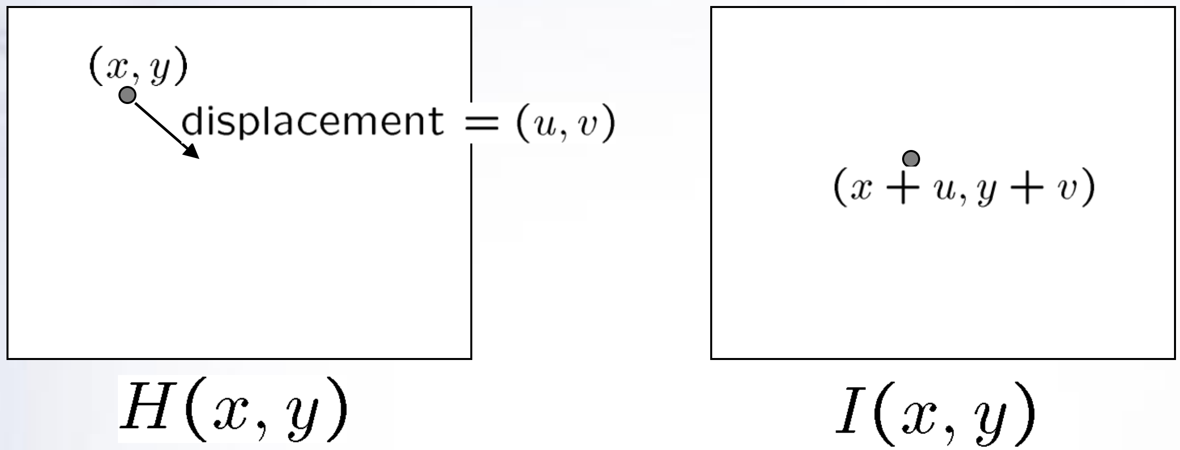
- Solve pixel correspondence problem
  - given a pixel in H, look for nearby pixels of the same color in I

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)



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

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

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

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

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

Lucas-kanade

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_pol

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!

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

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

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!

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

$$\begin{bmatrix} I_x(\mathbf{p_1})[0] & I_y(\mathbf{p_1})[0] \\ I_x(\mathbf{p_1})[1] & I_y(\mathbf{p_1})[1] \\ I_x(\mathbf{p_1})[2] & I_y(\mathbf{p_1})[2] \\ \vdots & \vdots \\ I_x(\mathbf{p_{25}})[0] & I_y(\mathbf{p_{25}})[0] \\ I_x(\mathbf{p_{25}})[1] & I_y(\mathbf{p_{25}})[1] \\ I_x(\mathbf{p_{25}})[2] & I_y(\mathbf{p_{25}})[2] \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p_1})[0] \\ I_t(\mathbf{p_1})[1] \\ I_t(\mathbf{p_1})[2] \\ \vdots \\ I_t(\mathbf{p_{25}})[0] \\ I_t(\mathbf{p_{25}})[1] \\ I_t(\mathbf{p_{25}})[2] \end{bmatrix}$$

$$\underset{75\times2}{A} \qquad \underset{2\times1}{d} \qquad \underset{75\times1}{b}$$

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

Lukas-Kanade flow

Prob: we have more equations than unknowns

$$\underset{25\text{x}2}{A}\ \underset{2\text{x}1}{d} = \underset{25\text{x}1}{b} \longrightarrow \text{minimize } \|Ad - b\|^2$$

Solution: solve least squares problem

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

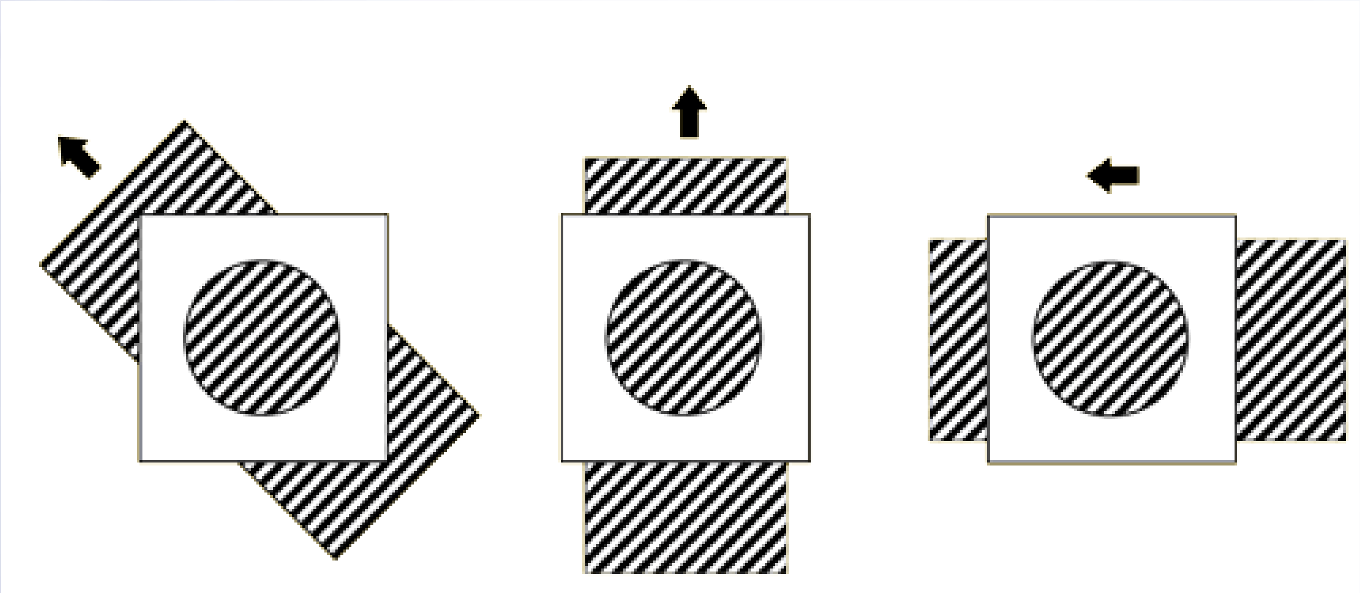
$$\underset{2\text{x}2}{(A^T A)}\ \underset{2\text{x}1}{d} = \underset{2\text{x}1}{A^T b}$$

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

To solve d=[u,v]

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

Aperture problem



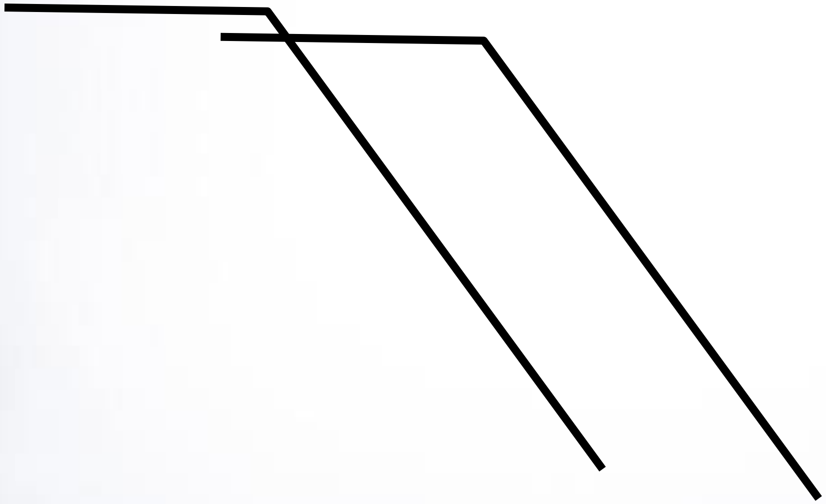
3 different moving stripes.

Observed from the hole, is same.

Thus direction of optical flow： multiple direction

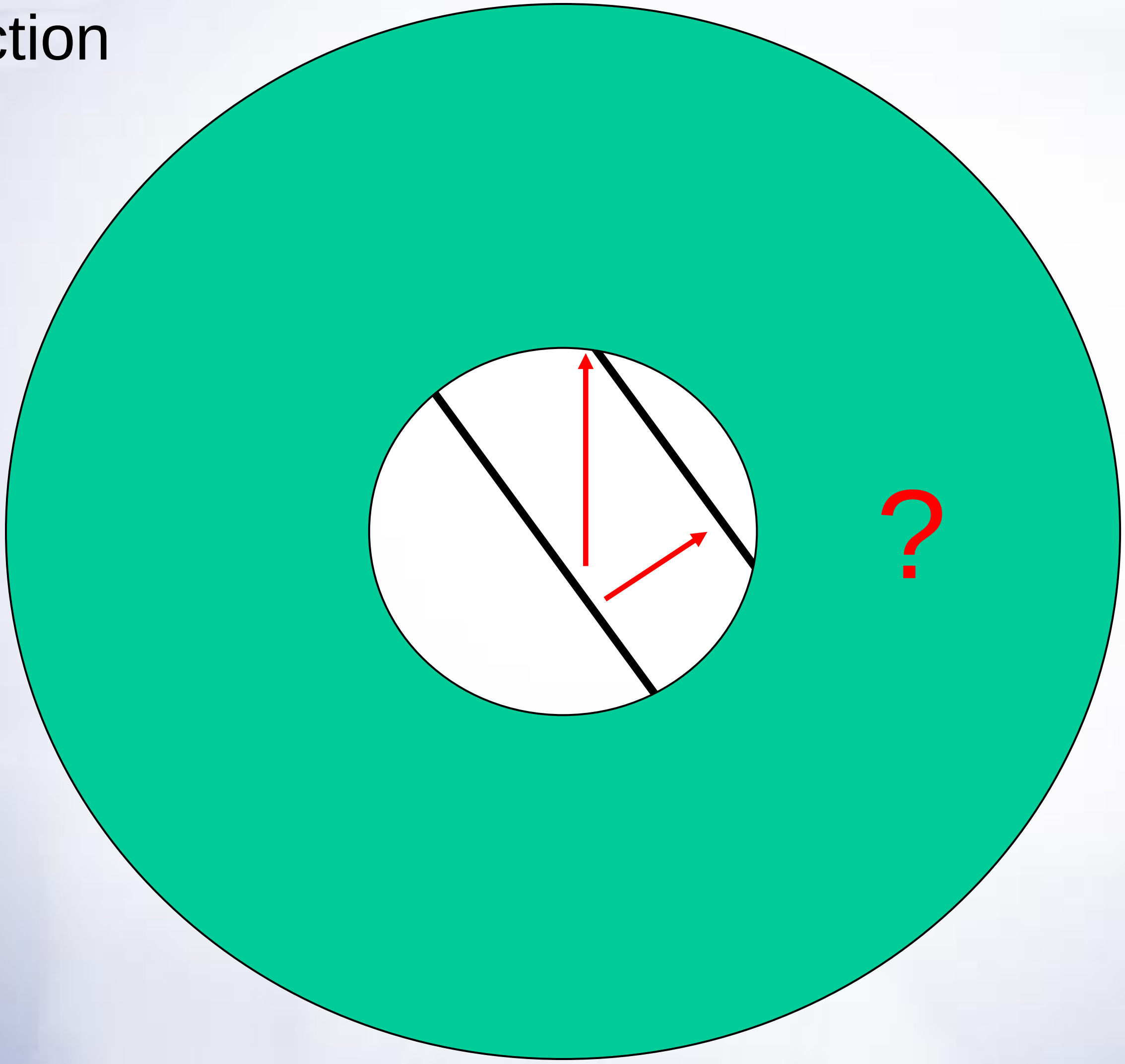
unable to estimate direction of optical flow

Aperture problem



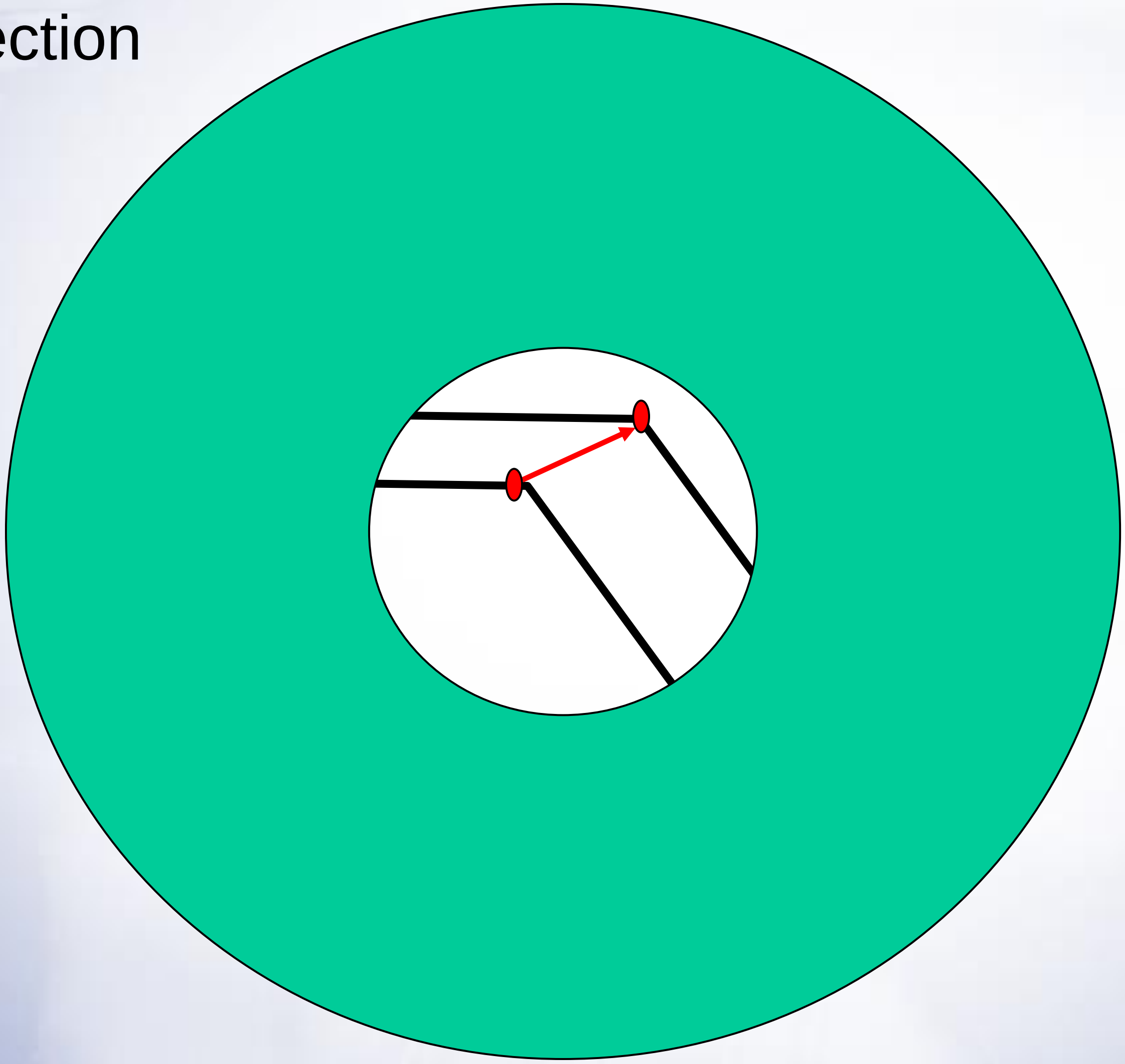
Aperture problem

Multi-direction



Aperture problem

single-direction



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? $\longrightarrow$ No aperture problem

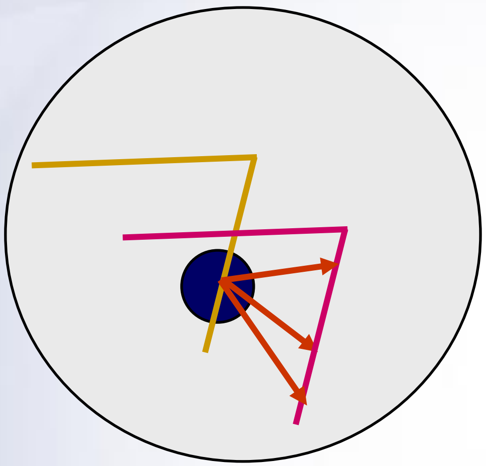
- A^TA should be invertible
- A^TA should not be too small due to noise
  – eigenvalues λ_1 and λ_2 of A^TA should not be too small
- A^TA should be well-conditioned
  – λ_1/λ_2 should not be too large (λ_1 = larger eigenvalue)

A^TA is solvable when there is no aperture problem

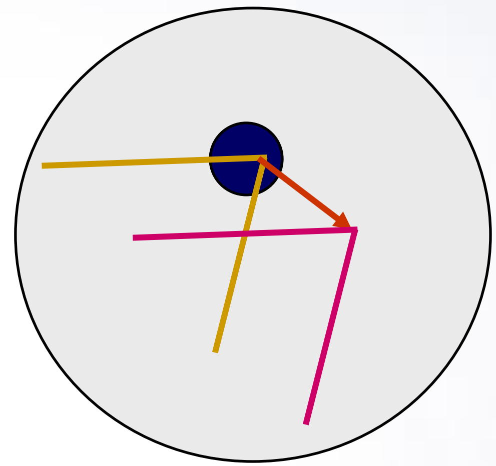
$$A^T A = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \ I_y] = \sum \nabla I (\nabla I)^T$$

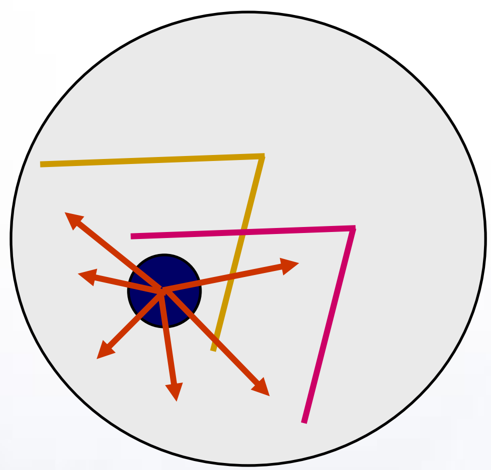
Local Patch Analysis

Egde： multi-direction ×



Corner： single-direction √





Flat： multi-direction ×

So need Corner detectot!
Corner point： A^TA Solvable

Aperture problem cause multi-direction when at egde、flat points

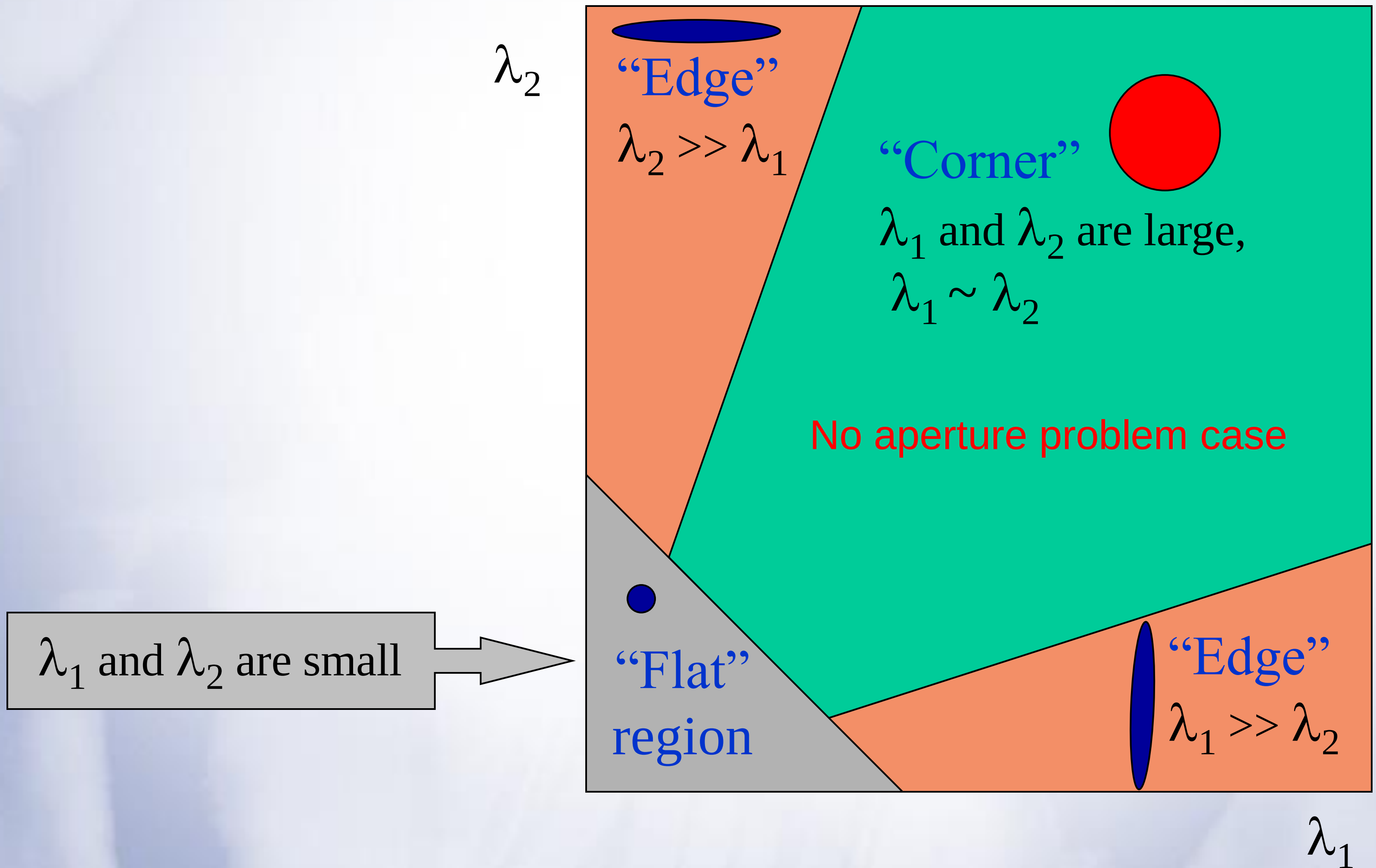
Eigenvectors of $A^T A$

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

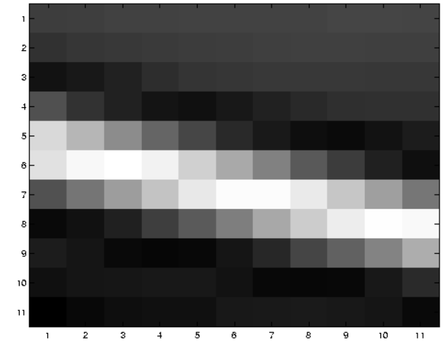
- Recall the Harris corner detector: $M = A^T A$ is the *second moment matrix*
- The eigenvectors and eigenvalues of *M* relate to edge direction and magnitude
  - The eigenvector associated with the larger eigenvalue points in the direction of fastest intensity change
  - The other eigenvector is orthogonal to it

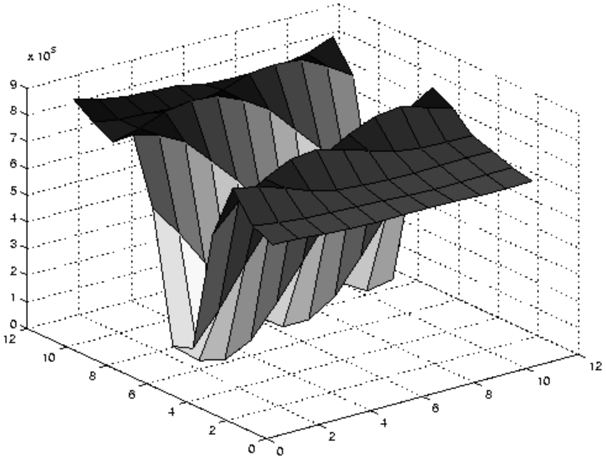
Interpreting the eigenvalues

Classification of image points using eigenvalues of the second moment matrix:



Edge



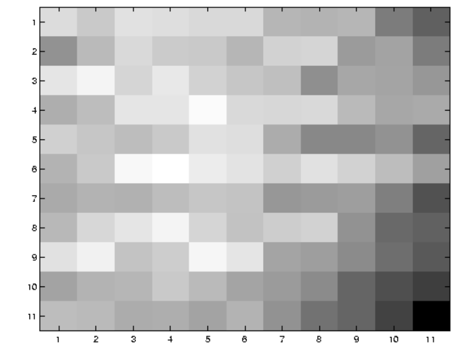


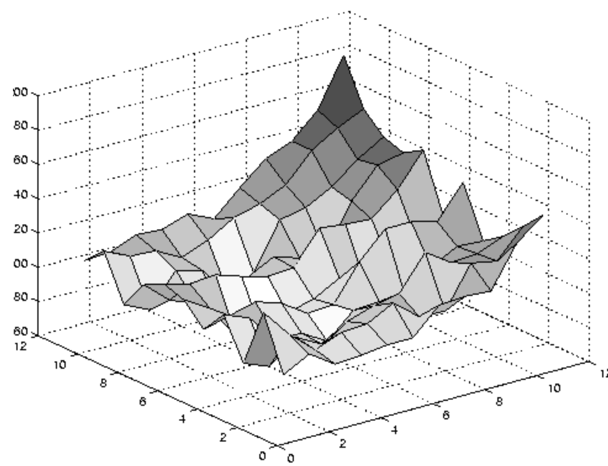
$$\sum \nabla I (\nabla I)^T$$

– large gradients, all the same

– large λ_1, small λ_2 ✗

Low texture region



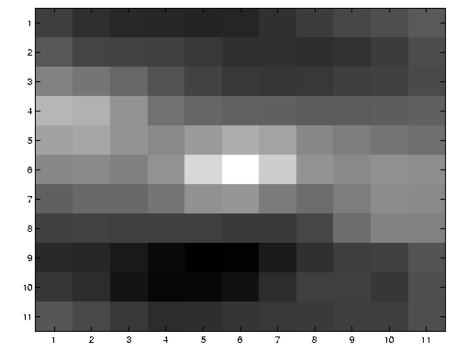


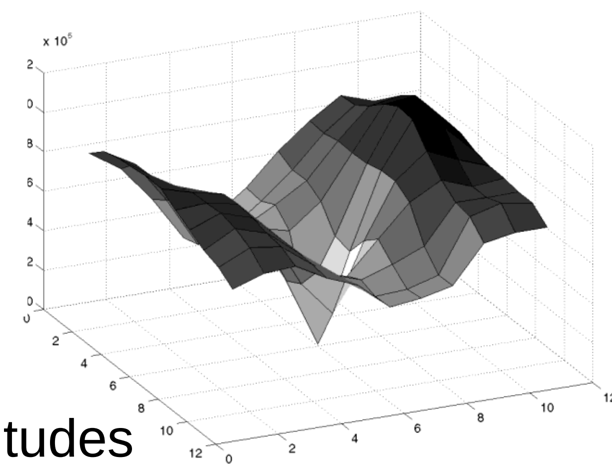
$$\sum \nabla I (\nabla I)^T$$

– gradients have small magnitude

– small λ_1, small λ_2 ✕

High textured region/Corner





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

– gradients are different, large magnitudes

– large λ_1, large λ_2 √

Horn-Schunck

Optical flow equation

Combining these two equations

$$0 = I(x+u, y+v) - H(x,y)$$

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

$$\approx I(x,y) + I_x u + I_y v - H(x,y)$$

$$\approx (I(x,y) - H(x,y)) + I_x u + I_y v$$

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

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

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

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

Horn-Schunck

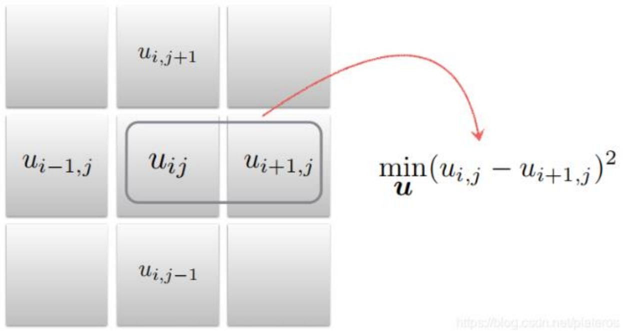
Horn-Schunck algorithm： global method

global constraint of smoothness

solve the aperture problem

LK constraint： $\min_{u,v}\left[I_x u_{ij} + I_y v_{ij} + I_t\right]^2$

Smooth constraint：



Global energy function:

$$E = \iint \left[(I_x u + I_y v + I_t)^2 + \alpha^2 \left(\|\nabla u\|^2 + \|\nabla v\|^2\right)\right] \mathrm{d}x\mathrm{d}y$$

Horn-Schunck

Discrete global energy function:

$$E = \sum_{i,j} E_s(i,j) + \lambda \sum_{i,j} E_d(i,j)$$

$$E_s(i,j) = \frac{1}{4}\left[(u_{i,j} - u_{i+1,j})^2 + (u_{i,j} - u_{i,j+1})^2 + (v_{i,j} - v_{i+1,j})^2 + (v_{i,j} - v_{i,j+1})^2\right]$$

$$E_d(i,j) = \|I_x u_{ij} + I_y v_{ij} + I_t\|^2$$

I_x, I_y, I_t: derivatives of image intensity values along x, y, t

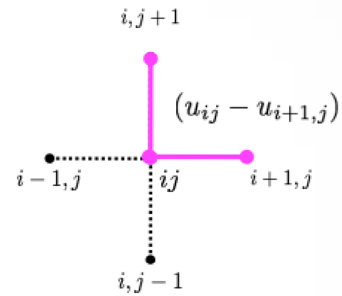
$\vec{V} = [u(x,y), v(x,y)]^\top$: optical flow vector (to be solved)

Horn-Schunck

Minimize discrete global energy function:

$$\sum_{ij}\left\{\frac{1}{4}\left[(u_{ij}-u_{i+1,j})^2+(u_{ij}-u_{i,j+1})^2+(v_{ij}-v_{i+1,j})^2+(v_{ij}-v_{i,j+1})^2\right]+\lambda\left[I_xu_{ij}+I_yv_{ij}+I_t\right]^2\right\}$$

$(u_{ij}^2-2u_{ij}u_{i+1,j}+u_{i+1,j}^2)$ $(u_{ij}^2-2u_{ij}u_{i,j+1}+u_{i,j+1}^2)$



(variable will appear four times in sum)

$$\frac{\partial E}{\partial u_{kl}}=2(u_{kl}-\bar{u}_{kl})+2\lambda(I_xu_{kl}+I_yv_{kl}+I_t)I_x$$

$$\frac{\partial E}{\partial v_{kl}}=2(v_{kl}-\bar{v}_{kl})+2\lambda(I_xu_{kl}+I_yv_{kl}+I_t)I_y$$

short hand for local average

$$\bar{u}_{ij}=\frac{1}{4}\left\{u_{i+1,j}+u_{i-1,j}+u_{i,j+1}+u_{i,j-1}\right\}$$

Horn-Schunck

Minimize discrete global energy function:

$$\frac{\partial E}{\partial u_{kl}} = 2(u_{kl} - \bar{u}_{kl}) + 2\lambda(I_x u_{kl} + I_y v_{kl} + I_t)I_x$$

$$\frac{\partial E}{\partial v_{kl}} = 2(v_{kl} - \bar{v}_{kl}) + 2\lambda(I_x u_{kl} + I_y v_{kl} + I_t)I_y$$

(set derivatives to zero and solve for unknowns u and v)

$$(1 + \lambda I_x^2)u_{kl} + \lambda I_x I_y v_{kl} = \bar{u}_{kl} - \lambda I_x I_t$$

$$\lambda I_x I_y u_{kl} + (1 + \lambda I_y^2)v_{kl} = \bar{v}_{kl} - \lambda I_y I_t$$

this is a linear system $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$ *how do you solve this?*

Horn-Schunck

Minimize discrete global energy function:

Matrix form：

$$\begin{bmatrix} 1+\lambda I_x^2 & \lambda I_x I_y \\ \lambda I_x I_y & 1+\lambda I_y^2 \end{bmatrix} \begin{bmatrix} u_{kl} \\ v_{kl} \end{bmatrix} = \begin{bmatrix} \bar{u}_{kl} - \lambda I_x I_t \\ \bar{v}_{kl} - \lambda I_y I_t \end{bmatrix}$$

$$\downarrow$$

$$\left\{1+\lambda\left(I_x^2+I_y^2\right)\right\} u_{kl} = \left(1+\lambda I_y^2\right)\bar{u}_{kl} - \lambda I_x I_y \bar{v}_{kl} - \lambda I_x I_t$$

$$\left\{1+\lambda\left(I_x^2+I_y^2\right)\right\} v_{kl} = \left(1+\lambda I_x^2\right)\bar{v}_{kl} - \lambda I_x I_y \bar{u}_{kl} - \lambda I_y I_t$$

$$\downarrow$$

$$u_{kl} = \bar{u}_{kl} - \frac{I_x\bar{u}_{kl}+I_y\bar{v}_{kl}+I_t}{\lambda^{-1}+I_x^2+I_y^2} I_x, \qquad v_{kl} = \bar{v}_{kl} - \frac{I_x\bar{u}_{kl}+I_y\bar{v}_{kl}+I_t}{\lambda^{-1}+I_x^2+I_y^2} I_y$$

Horn-Schunck

Algorithm flow:

- Initialize the optical flow, $u = 0, v = 0$
- Iteration:
  - Calculate the image gradient I_x, I_y
  - Calculate the difference between the two frames I_t
  - Solve the below two equations iteratively until convergence.

$$u_{kl} = \bar{u}_{kl} - \frac{I_x \bar{u}_{kl} + I_y \bar{v}_{kl} + I_t}{\lambda^{-1} + I_x^2 + I_y^2} I_x, \qquad v_{kl} = \bar{v}_{kl} - \frac{I_x \bar{u}_{kl} + I_y \bar{v}_{kl} + I_t}{\lambda^{-1} + I_x^2 + I_y^2} I_y$$

$$\bar{u}_{ij} = \frac{1}{4}\left\{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}\right\}$$

Optical flow pyramid

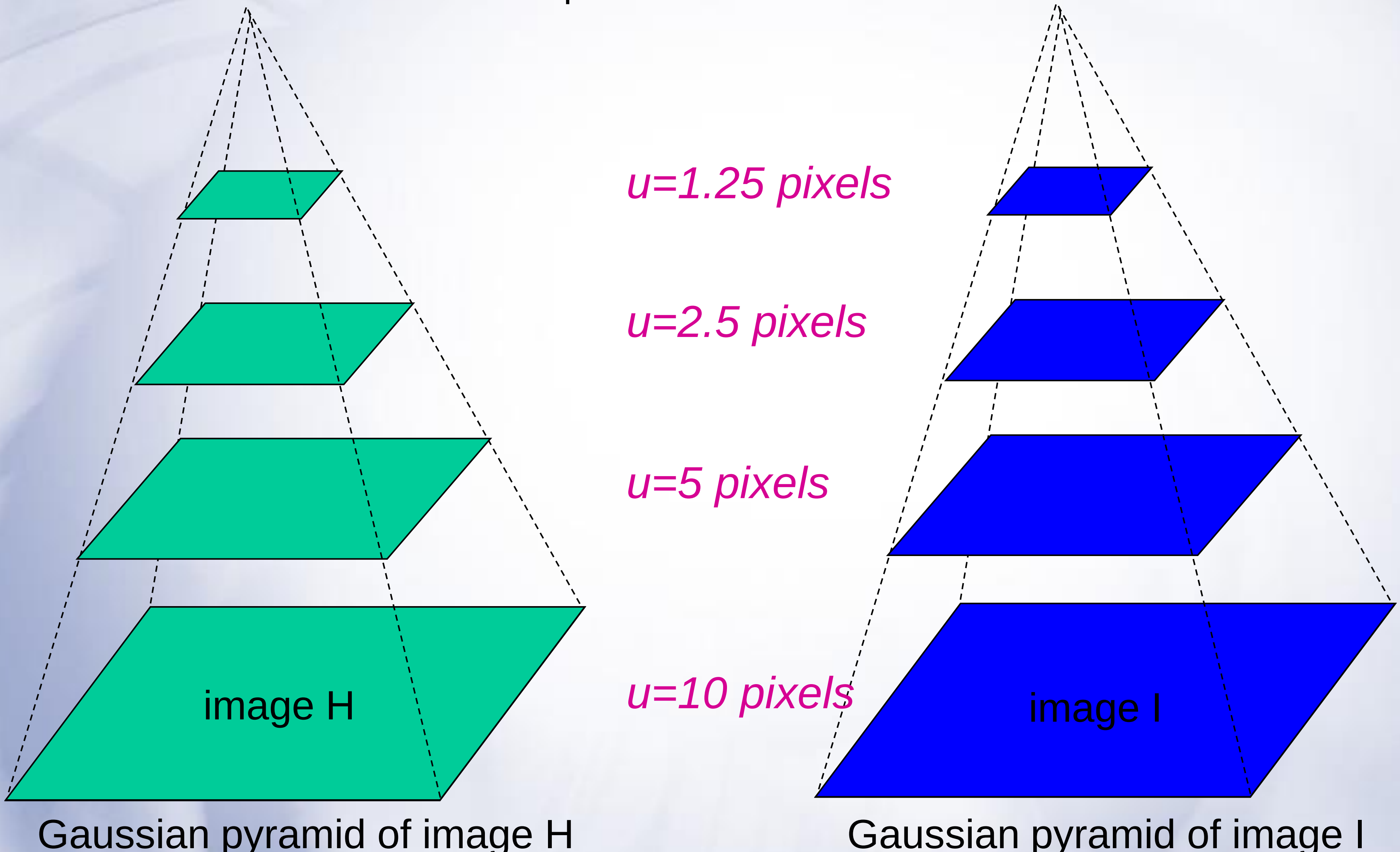
Optical flow pyramid

Both the LK and Horn-schunck algorithms have the assumption of small motion.

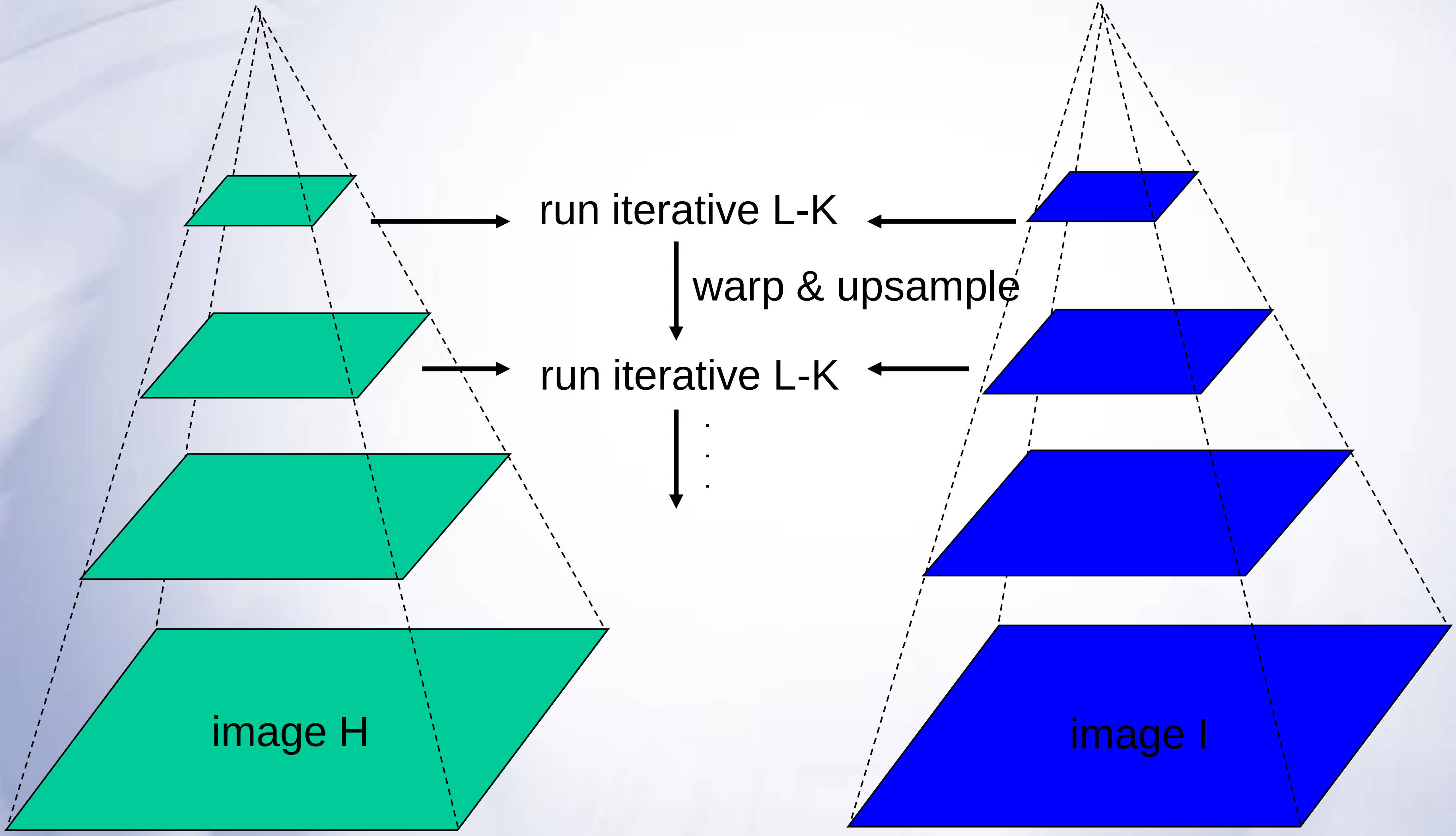
However, this assumption cannot be guaranteed under actual scenarios.

Coarse-to-fine optical flow estimation

Optical flow pyramid algorithm to improve the drawbacks associated with the small motion assumption.



Coarse-to-fine optical flow estimation



Gaussian pyramid of image H

Gaussian pyramid of image I

Optical flow pyramid

Generate layer image:

- Image1, image2 are scaled by a certain ratio
- Bottom layer is the original size image
- The algorithm calculates the optical flow from the top layer
- Result of the previous layer as next layer input

Therefore this process is also called coarse-to-fine.

Optical flow pyramid

Generate layer image:

$$\begin{aligned} I^L(x,y) \quad = \quad & \frac{1}{4} I^{L-1}(2x,2y) + \\ & \frac{1}{8}\left(I^{L-1}(2x-1,2y) + I^{L-1}(2x+1,2y) + I^{L-1}(2x,2y-1) + I^{L-1}(2x,2y+1)\right) + \\ & \frac{1}{16}\left(I^{L-1}(2x-1,2y-1) + I^{L-1}(2x+1,2y+1) + I^{L-1}(2x-1,2y+1) + I^{L-1}(2x+1,2y+1)\right). \end{aligned}$$

I_0: Layer 0 of image I , the original image with the highest resolution

I^{L-1}: L define number of layers. I^{L-1} denotes the image of layer $L-1$

$$\mathbf{g}^{L-1} = 2\left(\mathbf{g}^{\mathbf{L}} + \mathbf{d}^L\right).$$

I^{L-1}: g^{L-1} denotes the initial optical flow of layer $L-1$.
d^Ldenotes the estimated optical flow of the L layer.

Optical flow pyramid

Generate layer image:

greater resolution

↓

greater pixel displacement

↓

more layers needed

Initial optical flow layer, as the initial input for the next layer.

Using LK or HS iterative optical flow for each layer.

Techniques for estimate motion

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)