HA NOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Computer Vision

Chap 6: Motion and Tracking

What we will learn today?

- Optical flow
- Lucas-Kanade method
- Horn-Schunck method
- Pyramids for large motion
- Feature tracking
- Background subtraction for motion detection

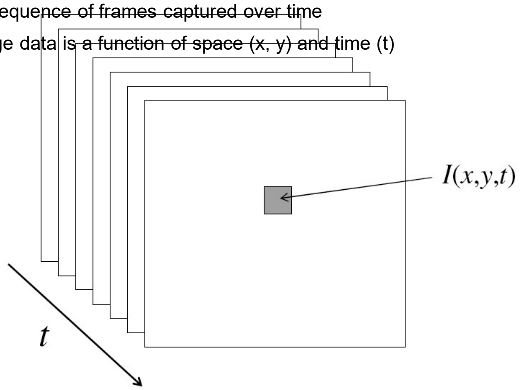
Reading: [Szeliski] Chapters: 8.4, 8.5

[Fleet & Weiss, 2005]
http://www.cs.toronto.edu/pub/jepson/teaching/vision/2503/opticalFlow.pdf

---

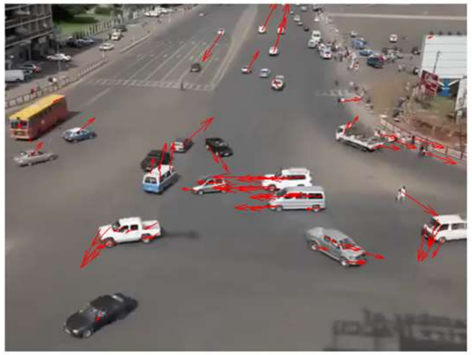
From images to videos

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



Why is motion useful?

Why is motion useful?



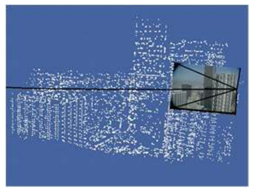
Motion and perceptual organization

Even “impoverished” motion data can evoke a strong percept

Uses of motion

- Estimating 3D structure
- Segmenting objects based on motion cues
- Video segmentation
- Recognizing events and activities
- Improving video quality (motion stabilization)
- …

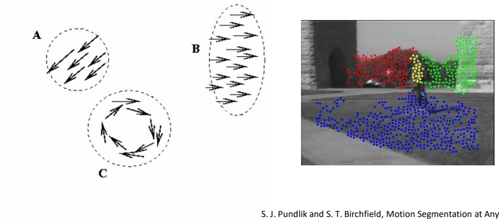
Estimating 3D structure



Source: Silvio Savarese

Segmenting objects based on motion cues

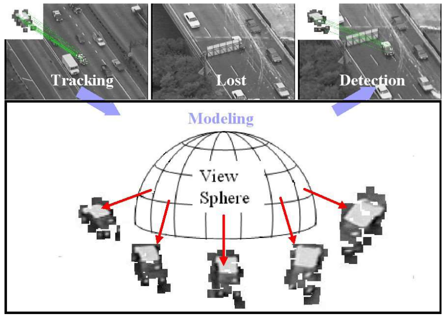
- Motion segmentation
  - Segment the video into multiple *coherently* moving objects



S. J. Pundlik and S. T. Birchfield, Motion Segmentation at Any Speed, Proceedings of the British Machine Vision Conference (BMVC) 2006

Source: Silvio Savarese

Tracking objects

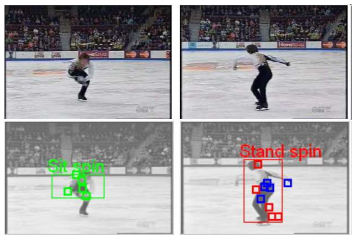


Z.Yin and R.Collins, "On-the-fly Object Modeling while Tracking," *IEEE Computer Vision and Pattern Recognition (CVPR '07)*, Minneapolis, MN, June 2007.

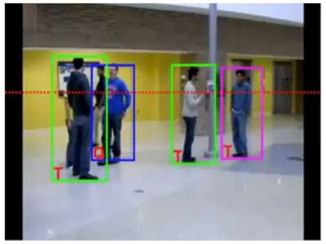
Source: Silvio Savarese

Recognizing events and activities

Crossing – Talking – Queuing – Dancing – jogging



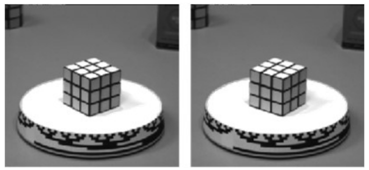
Juan Carlos Niebles, Hongcheng Wang and Li Fei-Fei, **Unsupervised Learning of Human Action Categories Using Spatial-Temporal Words**, *(BMVC)*, Edinburgh, 2006.

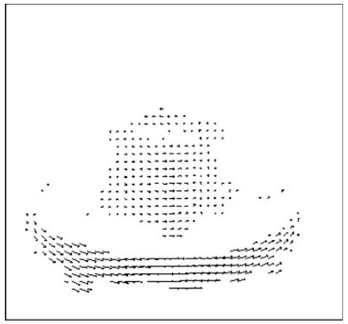


W. Choi & K. Shahid & S. Savarese WMC 2010

Motion field

- The motion field is the projection of the 3D scene motion into the image





Motion field + camera motion

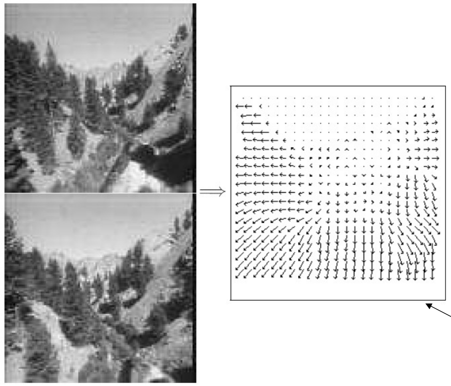


Figure 1.2: Two images taken from a helicopter flying through a canyon and the computed optical flow field.

Figure from Michael Black, Ph.D. Thesis

Length of flow vectors inversely proportional to depth Z of 3d point

points closer to the camera move more quickly across the image plane

Apparent motion != motion field

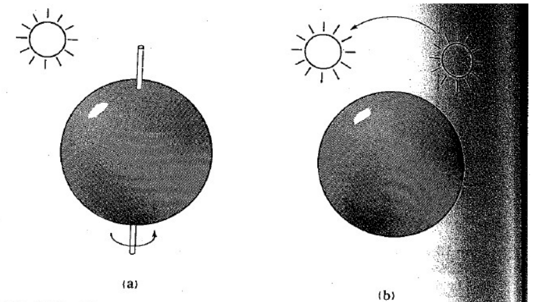
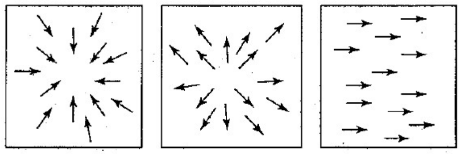


Figure 12-2. The optical flow is not always equal to the motion field. In (a) a smooth sphere is rotating under constant illumination—the image does not change, yet the motion field is nonzero. In (b) a fixed sphere is illuminated by a moving source—the shading in the image changes, yet the motion field is zero.

Figure from Horn book

Motion field + camera motion



Zoom out **Zoom in** **Pan right to left**

Motion estimation techniques

- Direct methods
  - Directly recover **image motion at each pixel** from spatio-temporal image brightness variations
  - Dense motion fields, but sensitive to appearance variations
  - Suitable for video and when **image motion is small**
- Feature-based methods
  - Extract visual features (corners, textured areas) and track them over multiple frames
  - Sparse motion fields, but more robust tracking
  - Suitable when image motion is large (10s of pixels)

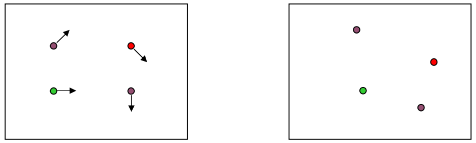
Optical flow

- Definition: optical flow is the apparent motion of brightness patterns in the image
- Ideally, optical flow would be the same as the motion field
- Note: apparent motion can be caused by lighting changes without any actual motion
  - Think of a uniform rotating sphere under fixed lighting vs. a stationary sphere under moving illumination

GOAL: Recover image motion at each pixel from optical flow

Problem definition: optical flow

$I(x,y,t)$

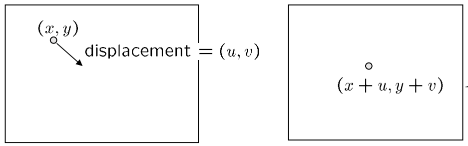


$I(x,y,t+1)$

- How to estimate pixel motion from image $I(x,y,t)$ to $I(x,y,t+1)$?
  - Solve pixel correspondence problem
    - Given a pixel in I(x,y,t), look for nearby pixels of the same color in *I(x,y,t+1)*
  - Key assumptions
    - Small motion: Points do not move very far
    - Color constancy: A point in *I(x,y,t)* looks the same in *I(x + u, y +v, t+1)*
      - For grayscale images, this is brightness constancy

Optical flow constraints (grayscale images)

$I(x,y,t)$



$I(x,y,t+1)$

- Let's look at these constraints more closely
  - Brightness constancy constraint (equation)

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

  - Small motion: (u and v are less than 1 pixel, or smoothly varying)

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

$$\begin{aligned} 0 &= I(x+u, y+v, t+1) - I(x,y,t) \\ &\approx I(x,y,t+1) + I_x u + I_y v - I(x,y,t) \end{aligned}$$

(Short hand: $I_x = \frac{\partial I}{\partial x}$ for t **or** $t+1$)

Optical flow equation

- Combining these two equations

$$
\begin{aligned}
0 &= I(x+u, y+v, t+1) - I(x,y,t) \\
&\approx I(x,y,t+1) + I_x u + I_y v - I(x,y,t) \\
&\approx [I(x,y,t+1) - I(x,y,t)] + I_x u + I_y v \\
&\approx I_t + I_x u + I_y v \\
&\approx I_t + \nabla I \cdot \langle u, v \rangle
\end{aligned}
$$

(Short hand: $I_x = \frac{\partial I}{\partial x}$ for t **or** $t+1$)

Optical flow equation

- Combining these two equations

$$
\begin{aligned}
0 &= I(x+u, y+v, t+1) - I(x,y,t) \\
&\approx I(x,y,t+1) + I_x u + I_y v - I(x,y,t) \\
&\approx \boxed{[I(x,y,t+1) - I(x,y,t)]} + I_x u + I_y v \\
&\approx \boxed{I_t} + I_x u + I_y v \\
&\approx I_t + \nabla I \cdot \langle u, v \rangle
\end{aligned}
$$

(Short hand: $I_x = \frac{\partial I}{\partial x}$ for t **or** $t+1$)

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

$$0 = I_t + \nabla I \cdot \langle u, v \rangle$$

Optical flow (velocities): (u, v)

Brightness constancy constraint equation

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

flow velocities

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

Image gradients
(at a point p)

temporal gradient

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

How do you compute …

$I_x = \frac{\partial I}{\partial x} \quad I_y = \frac{\partial I}{\partial y}$	$u = \frac{dx}{dt} \quad v = \frac{dy}{dt}$	$I_t = \frac{\partial I}{\partial t}$
spatial derivative	**optical flow**	**temporal derivative**
Forward difference Sobel filter Derivative-of-Gaussian filter …	(u, v) Solution lies on a line Cannot be found uniquely with a single constraint	frame differencing

How to compute gradients in x-y-t

$$I_x = \frac{1}{4}[(I_{x+1,y,t} + I_{x+1,y,t+1} + I_{x+1,y+1,t} + I_{x+1,y+1,t+1}) \\ -(I_{x,y,k} + I_{x,y,t+1} + I_{x,y+1,t} + I_{x,y+1,t+1})]$$

likewise for I_y and I_t

Brightness constancy constraint

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

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

➔ How many equations and unknowns per pixel?

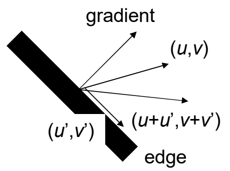
– One equation (this is a scalar equation!), two unknowns (u,v)

Need more constraints

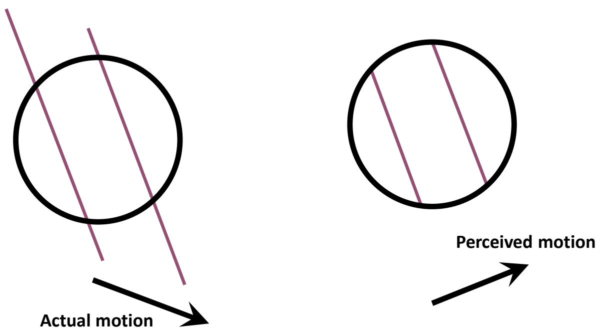
The component of the motion perpendicular to the gradient (i.e., parallel to the edge) cannot be measured

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

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



The aperture problem



Source: Silvio Savarese

The barber pole illusion

http://en.wikipedia.org/wiki/Barberpole_illusion

Source: Silvio Savarese

Horn-Schunck Optical Flow (1981)	Lucas-Kanade Optical Flow (1981)
brightness constancy small motion	method of differences
'smooth' flow (flow can vary from pixel to pixel)	**'constant' flow** (flow is constant for all pixels)
global method (dense)	local method (sparse)

http://www.cs.cmu.edu/~16385/

What we will learn today?

- Optical flow
- Lucas-Kanade method
- Horn-Schunck method
- Pyramids for large motion
- Feature tracking
- Background subtraction for motion detection

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

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

$$\begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix}$$

Source: Silvio Savarese

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.

Lucas-Kanade flow

- Overconstrained linear system

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

Least squares solution for *d* given by

$$(A^T A) \ d = A^T b$$

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

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

Source: Silvio Savarese

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? What are good points to track?

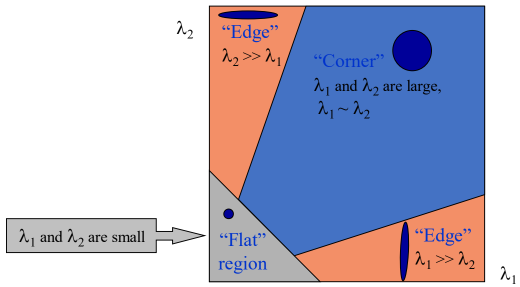
- **A^{T}A** should be invertible
- **A^{T}A** should not be too small due to noise
  – eigenvalues λ_1 and λ_2 of **A^{T}A** should not be too small
- **A^{T}A** should be well-conditioned
  – λ_1/λ_2 should not be too large (λ_1 = larger eigenvalue)

Does this remind you of anything?

Criteria for Harris corner detector

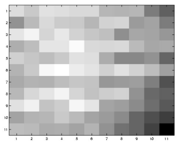
Interpreting the eigenvalues

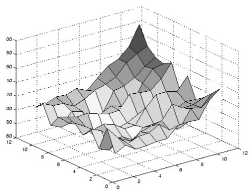
Classification of image points using eigenvalues of the second moment matrix:



Source: Silvio Savarese

Low texture region

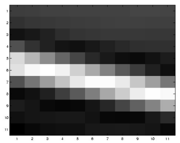


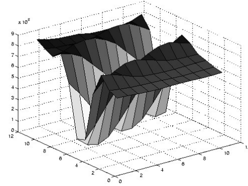


$$\sum \nabla I (\nabla I)^T$$

– gradients have small magnitude
– small λ_1, small λ_2

Edge

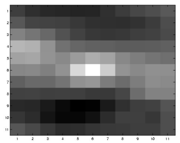
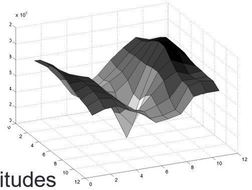




$$\sum \nabla I (\nabla I)^T$$

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

High textured region

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

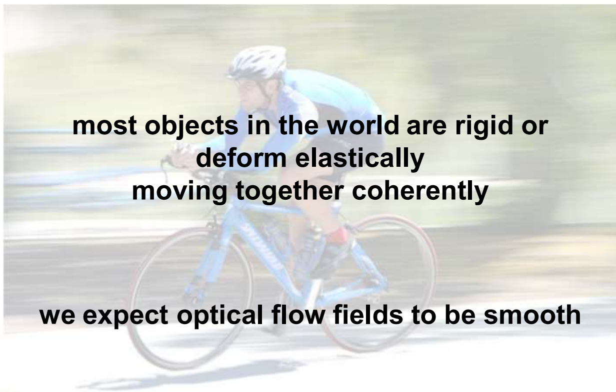
– gradients are different, large magnitudes

– large λ_1, large λ_2

What we will learn today?

- Optical flow
- Lucas-Kanade method
- Horn-Schunck method
- Pyramids for large motion
- Feature tracking
- Background subtraction for motion detection

Smoothness



Key idea
(of Horn-Schunck optical flow)

Enforce
brightness constancy

Enforce
smooth flow field

to compute optical flow

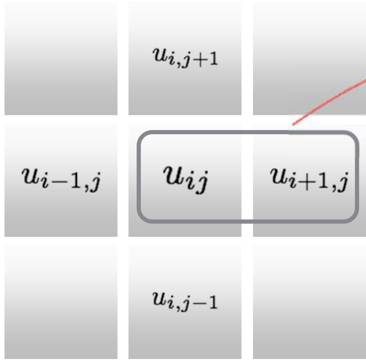
Enforce **brightness constancy**

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

For every pixel,

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

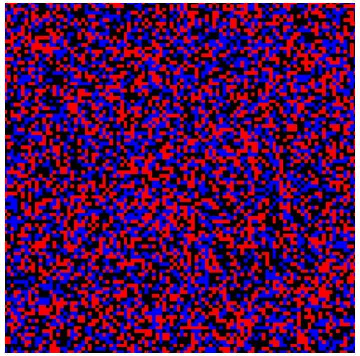
Enforce **smooth flow field**

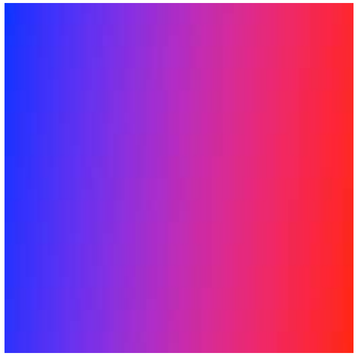


$$\min_{\boldsymbol{u}} (u_{i,j} - u_{i+1,j})^2$$

u-component of flow

Which flow field optimizes the objective? $\min_{\boldsymbol{u}} (u_{i,j} - u_{i+1,j})^2$





$\sum_{ij} (u_{ij} - u_{i+1,j})^2$ big ? $\sum_{ij} (u_{ij} - u_{i+1,j})^2$ small

Horn-Schunck optical flow

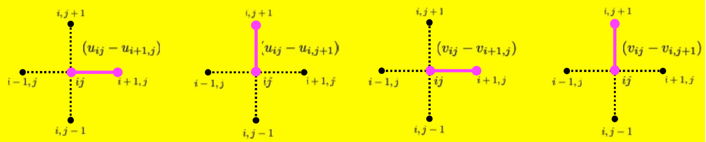
$$\min_{\boldsymbol{u},\boldsymbol{v}} \sum_{i,j} \left\{ \underbrace{E_s(i,j)}_{\text{smoothness}} + \underbrace{\lambda}_{\text{weight}} \underbrace{E_d(i,j)}_{\text{brightness constancy}} \right\}$$

HS optical flow objective function

Brightness constancy $E_d(i,j) = \left[I_x u_{ij} + I_y v_{ij} + I_t \right]^2$

Smoothness

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



How do we solve this minimization problem?

$$\min_{\boldsymbol{u},\boldsymbol{v}} \sum_{i,j} \left\{ E_s(i,j) + \lambda E_d(i,j) \right\}$$

Compute partial derivative, derive update equations

(gradient decent!)

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

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

Where are the extrema of E?

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

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

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

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

this is a linear system $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$

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

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

Rearrange to get update equations:

$$\underbrace{\hat{u}_{kl}}_{\text{new value}} = \underbrace{\bar{u}_{kl}}_{\text{old average}} - \frac{I_x \bar{u}_{kl} + I_y \bar{v}_{kl} + I_t}{\lambda^{-1} + I_x^2 + I_y^2} I_x$$

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

Recall: $\min_{\boldsymbol{u},\boldsymbol{v}} \sum_{i,j} \left\{ E_s(i,j) + \lambda E_d(i,j) \right\}$

When lambda is small (lambda inverse is big)…

$$\hat{u}_{kl} = \bar{u}_{kl} - \frac{\cancel{I_x \bar{u}_{kl} + I_y \bar{v}_{kl} + I_t}}{\cancel{\lambda^{-1}} + I_x^2 + I_y^2} I_x$$

goes to zero

new value

old average

$$\hat{v}_{kl} = \bar{v}_{kl} - \frac{\cancel{I_x \bar{u}_{kl} + I_y \bar{v}_{kl} + I_t}}{\cancel{\lambda^{-1}} + I_x^2 + I_y^2} I_y$$

goes to zero

…we only care about smoothness.

Horn-Schunck Optical Flow Algorithm

1. Precompute image gradients I_y I_x
2. Precompute temporal gradients I_t
3. Initialize flow field $\boldsymbol{u} = \boldsymbol{0}$ $\boldsymbol{v} = \boldsymbol{0}$
4. While not converged

 Compute flow field updates for each pixel:

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

What we will learn today?

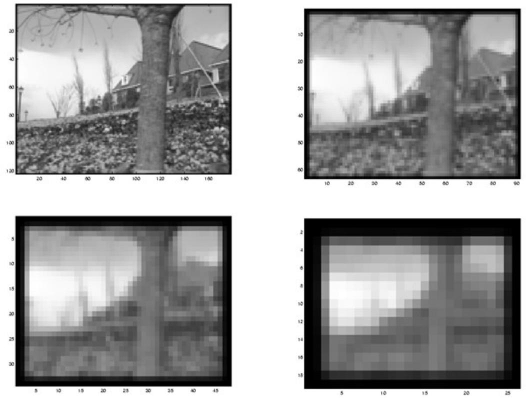
- Optical flow
- Lucas-Kanade method
- Horn-Schunck method
- Pyramids for large motion
- Feature tracking
- Background subtraction for motion detection

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?

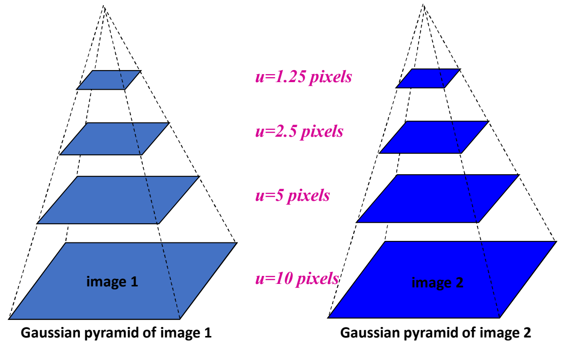
* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Reduce the resolution!



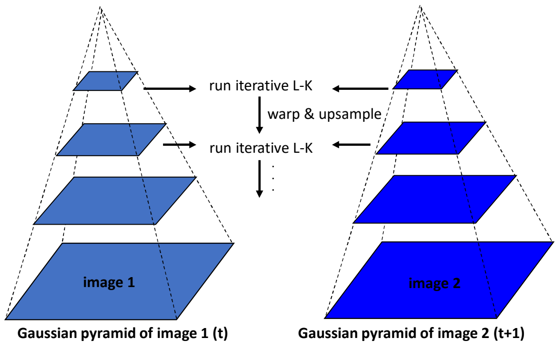
* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Coarse-to-fine optical flow estimation



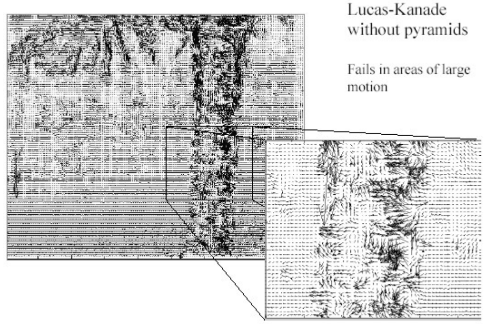
Source: Silvio Savarese

Coarse-to-fine optical flow estimation



Source: Silvio Savarese

Optical Flow Results

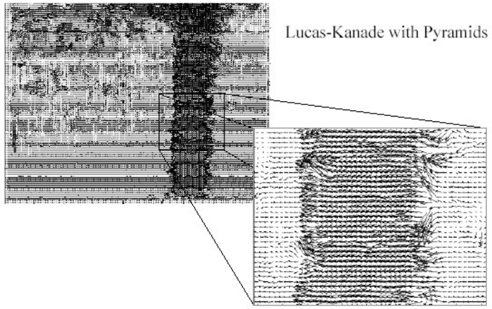


Lucas-Kanade without pyramids

Fails in areas of large motion

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Optical Flow Results

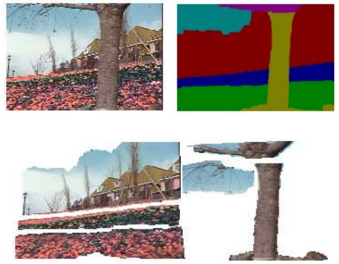


- http://www.ces.clemson.edu/~stb/klt/
- OpenCV

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

OF for motion segmentation

- Break image sequence into “layers” each of which has a coherent (affine) motion



J. Wang and E. Adelson. Layered Representation for Motion Analysis. *CVPR 1993.*

Source: Silvio Savarese

Motion estimation techniques

- Direct methods
  - Directly recover image motion at each pixel from spatio-temporal image brightness variations
  - Dense motion fields, but sensitive to appearance variations
  - Suitable for video and when image motion is small
- Feature-based methods
  - Extract visual features (corners, textured areas) and track them over multiple frames
  - Sparse motion fields, but more robust tracking
  - Suitable when image motion is large (10s of pixels)

What we will learn today?

- Optical flow
- Lucas-Kanade method
- Horn-Schunck method
- Pyramids for large motion
- Feature tracking
- Background subtraction for motion detection

Feature tracking: problem statement

Image sequence

Slide credit: Yonsei Univ.

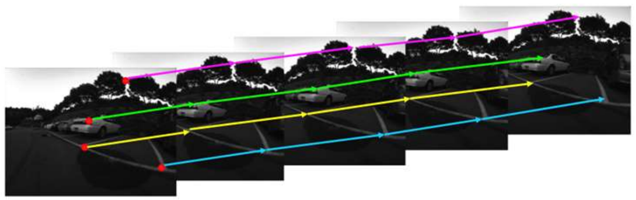
Problem statement

Feature point detection

Slide credit: Yonsei Univ.

Problem statement

Feature point tracking

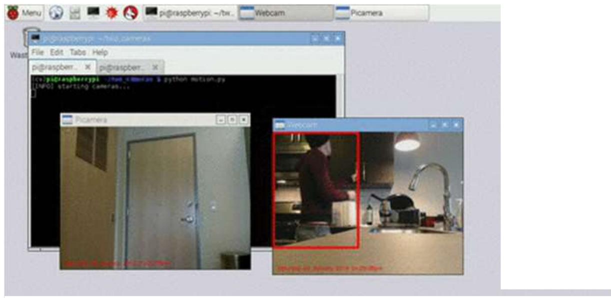


Slide credit: Yonsei Univ.

Single / Mutilple object tracking

Tracking with a fixed/moving camera

Tracking with multiple cameras



Challenges in feature tracking

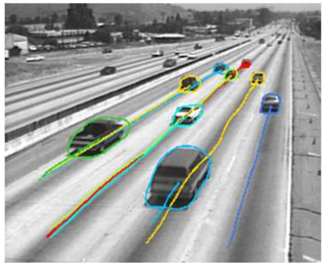
- Figure out which features can be tracked
  - Efficiently track across frames
- Some points may change appearance over time
  - e.g., due to rotation, moving into shadows, etc.
- Drift: small errors can accumulate as appearance model is updated
- Points may appear or disappear
  - need to be able to add/delete tracked points

What are good features to track?

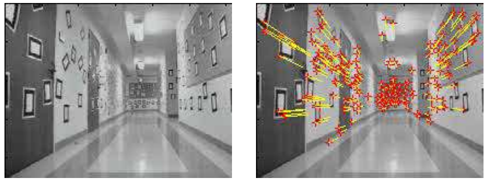
- Intuitively, we want to avoid smooth regions and edges.
- But is there a more is principled way to define good features?
  - What kinds of image regions can we detect easily and consistently? ➔Think about what you learnt earlier in the class.
  - Can measure "quality" of features from just a single image
- Hence: tracking Harris corners (or equivalent) guarantees small error sensitivity!

Optical flow can help track features

Once we have the features we want to track, lucas-kanade or other optical flow algorithm can help track those features



Feature-tracking



Courtesy of Jean-Yves Bouguet – Vision Lab, California Institute of Technology

Simple KLT tracker

1. Find a good point to track (harris corner)
2. For each Harris corner compute motion (translation or affine) between consecutive frames.
3. Link motion vectors in successive frames to get a track for each Harris point
4. Introduce new Harris points by applying Harris detector at every m (10 or 15) frames
5. Track new and old Harris points using steps 1-3

History of the

Kanade-Lucas-Tomasi (KLT) Tracker

An Iterative Image Registration Technique with an Application to Stereo Vision.

1981

Detection and Tracking of Feature Points.

1991

The original KLT algorithm

Good Features to Track.

1994

KLT tracker for fish

Video credit: Kanade

Tracking movement

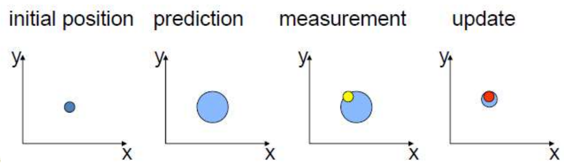
Video credit: Kanade

Other trackers

- Point tracking
  - KLT
  - Kalman
  - ..
- Kernel tracking
  - Mean-shift
  - KCF 2014
  - Struck 2014
  - TLD 2010
  - MIL 2009
  - Online boosting 2006
  - …

Kalman filter

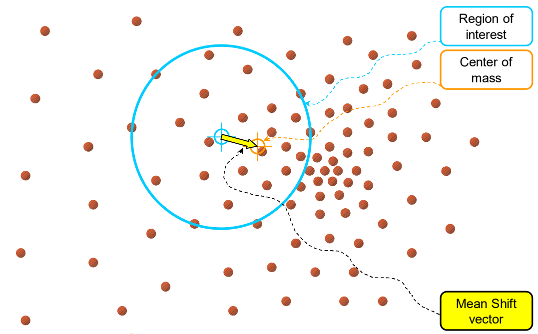
- Method for tracking linear dynamical models in Gaussian noise
- The predicted/corrected state distributions are Gaussian
  - Need to maintain the mean and covariance
  - Calculations are easy



Mean-shift

- The mean-shift algorithm is an efficient approach to tracking objects whose appearance is defined by histograms. (not limited to only color)
- Motivation
  - to track non-rigid objects, (like a walking person), it is hard to specify an explicit 2D parametric motion model.
  - Appearances of non-rigid objects can sometimes be modeled with color distributions

Intuitive Description

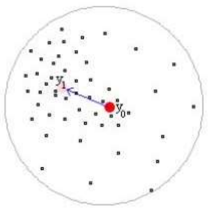


Objective : Find the densest region
Distribution of identical billiard balls

Stolen from: www.wisdom.weizmann.ac.il/~deniss/vision_spring04/files/mean_shift/mean_shift.ppt

Mean-shift vector

$$M_h(\mathbf{y}) = \left[\frac{1}{n_x} \sum_{i=1}^{n_x} \mathbf{x}_i \right] - \mathbf{y}_0$$

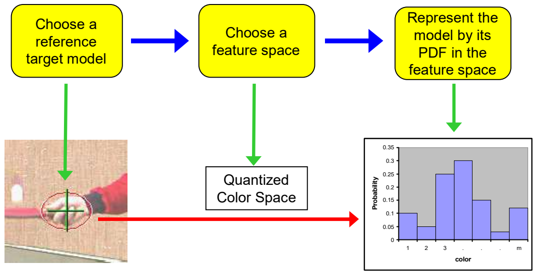


The mean shift vector always points to the **direction of the densest point** of data points

Mean-shift vector

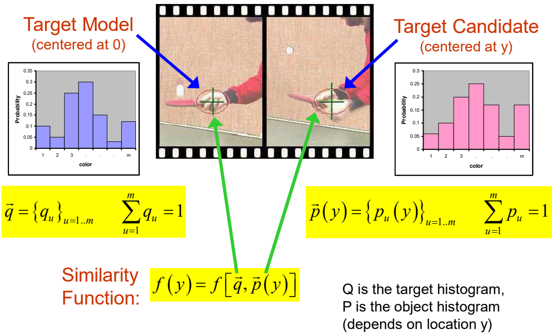
- Given:
  - Data points and approximate location of the mean of this data
- Task:
  - Estimate the exact location of the mean of the data by determining the shift vector from the initial mean.
- How ?
  - Compute mean shift vector from a region of interest
  - Estimate the new location
  - Repeat until, mean shift vector is zero

Mean-Shift Object Tracking
Target Representation



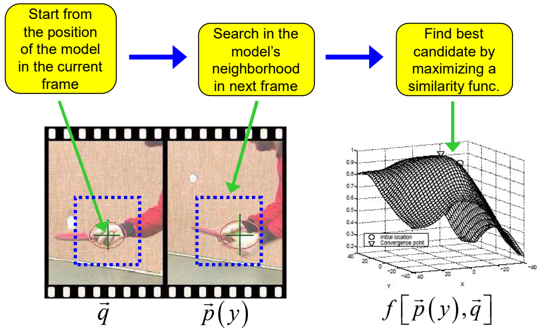
Source: www.cs.wustl.edu/~pless/559/lectures/lecture22_tracking.ppt

Mean-Shift Object Tracking
PDF Representation



Source: www.cs.wustl.edu/~pless/559/lectures/lecture22_tracking.ppt

Mean-Shift Object Tracking
Target Localization Algorithm



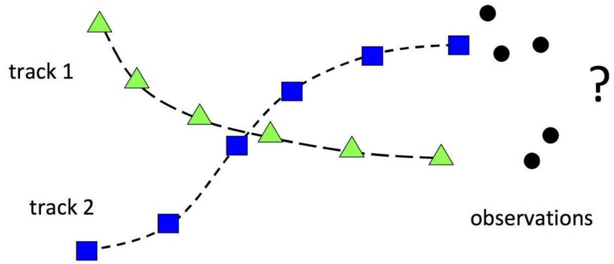
Stolen from: www.cs.wustl.edu/~pless/559/lectures/lecture22_tracking.ppt

Mean-shift - Summary

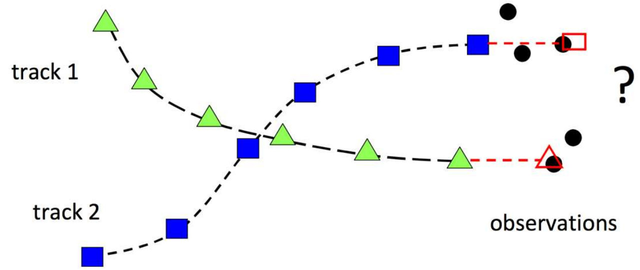
- Mean-Shift in tracking task:
  - ➢ track the motion of a cluster of interesting features.

1. Choose the feature distribution to represent an object (e.g., color + texture),
2. Start the mean-shift window over the feature distribution generated by the object
3. Finally compute the chosen feature distribution over the **next video frame**
   - Starting from the current window location, the mean-shift algorithm will find the new peak or mode of the feature distribution, which (presumably) is centered over the object that produced the color and texture in the first place.
   - In this way, the mean-shift window tracks the movement of the object frame by frame.

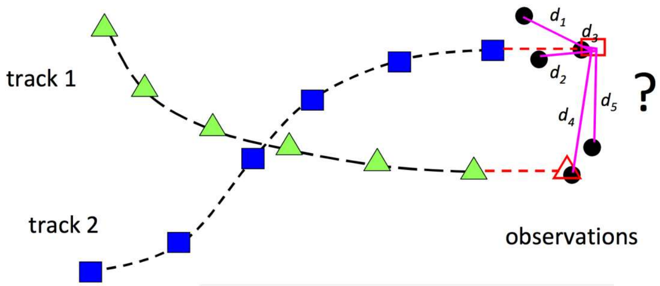
Object tracking – Multi target tracking



Object tracking – Multi target tracking



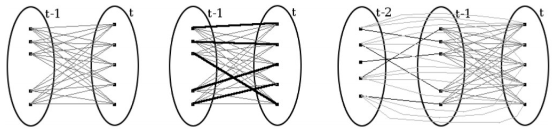
Object tracking – Multi target tracking



Object tracking – Multi target tracking

- 2D assignment problem (Bipartial matching problem)

$$\min_{x_{i,j}} \sum c_{i,j} x_{i,j}$$
$$s.t. \sum_{i:i>0} x_{i,j} = 1$$
$$\sum_{j:j>0} x_{i,j} = 1$$
$$x_{i,j} \in \{0, 1\}$$



- Hungarian method
- Auction method
- JVC method

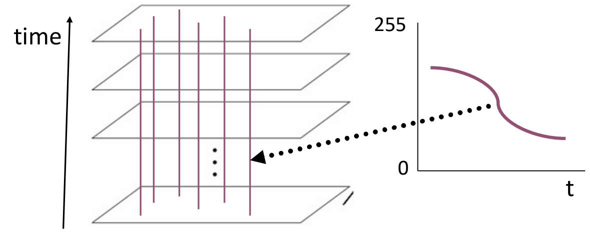
https://itu.dk/people/maau/teaching/visualisation/auction-algorithm/auction-algorithm.ht

https://cyberlab.engr.uconn.edu/wp-content/uploads/sites/2576/2018/09/Lecture_8.pdf

What we will learn today?

- Optical flow
- Lucas-Kanade method
- Horn-Schunck method
- Pyramids for large motion
- Feature tracking
- Background subtraction for motion detection

Video as an "Image Stack"



- Can look at video data as a spatio-temporal volume
  - If camera is stationary, each line through time corresponds to a single ray in space

Alyosha Efros, CMU

Background subtraction

- Given an image (mostly likely to be a video frame), we want to identify the **foreground objects** in that image!

⇒

Motivation

- In most cases, objects are of interest, not the scene.
- Makes our life easier: less processing costs, and less room for error.

Slide credit: Birgi Tamersoy

Background subtraction

- Simple techniques can do ok with static camera
- …But hard to do perfectly

- Widely used:
  - Traffic monitoring (counting vehicles, detecting & tracking vehicles, pedestrians),
  - Human action recognition (run, walk, jump, squat),
  - Human-computer interaction
  - Object tracking

Simple Approach

Image at time t: $I(x, y, t)$ $\Downarrow$ Background at time t: $B(x, y, t)$ $\Downarrow$

$|$  $-$  $| > Th$

1. Estimate the background for time t.
2. Subtract the estimated background from the input frame.
3. Apply a threshold, Th, to the absolute difference to get the **foreground mask**.

Frame Differencing

- Background is estimated to be the previous frame. Background subtraction equation then becomes:

$$B(x, y, t) = I(x, y, t-1)$$
$$\Downarrow$$
$$|I(x, y, t) - I(x, y, t-1)| > Th$$

- Depending on the object structure, speed, frame rate and global threshold, this approach may or may **not** be useful (usually **not**).

$|$  $-$  $| > Th$

Frame Differencing

$Th = 25$

$Th = 50$

$Th = 100$

$Th = 200$

Mean Filter

- In this case the background is the mean of the previous n frames:

$$B(x, y, t) = \frac{1}{n}\sum_{i=0}^{n-1} I(x, y, t-i)$$
$$\Downarrow$$
$$|I(x, y, t) - \frac{1}{n}\sum_{i=0}^{n-1} I(x, y, t-i)| > Th$$

- For $n = 10$:

Estimated Background

Foreground Mask

Median Filter

- Assuming that the background is more likely to appear in a scene, we can use the median of the previous n frames as the background model:

$$B(x, y, t) = median\{I(x, y, t - i)\}$$
$$\Downarrow$$
$$|I(x, y, t) - median\{I(x, y, t - i)\}| > Th \text{ where}$$
$$i \in \{0, \dots, n - 1\}.$$

- For $n = 10$:

Estimated Background

Foreground Mask

Slide credit: Birgi Tamersoy

Average/Median Image

Alyosha Efros, CMU

Background Subtraction

-

=

Alyosha Efros, CMU

Pros and cons

Advantages:

- Extremely easy to implement and use!
- All pretty fast
- Corresponding background models need not be constant, they change over time.

Disadvantages:

- Accuracy of frame differencing depends on object speed and frame rate
- Median background model: relatively high memory requirements
- Setting global threshold Th…

Background mixture models

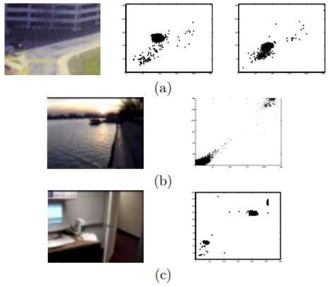
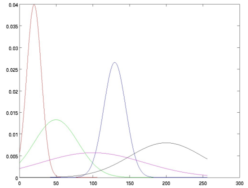


Figure 2: *This figure contains images and scatter plots of the red and green values of a single pixel from the image over time. It illustrates some of the difficulties involved in real environments. (a) shows two scatter plots from the same pixel taken 2 minutes apart. This would require two thresholds. (b) shows a bi-model distribution of a pixel values resulting from specularities on the surface of water. (c) shows another bi-modality resulting from monitor flicker.*



Idea: model each background pixel with a *mixture* of Gaussians; update its parameters over time.

Adaptive Background Mixture Models for Real-Time Tracking, Chris Stauffer & W.E.L. Grimson, CVPR98

What we have learned today

- Optical flow
- Lucas-Kanade method
- Horn-Schunck method
- Pyramids for large motion
- Feature tracking
- Background subtraction for motion detection

References

- CS131 Juan Carlos Niebles and Ranjay Krishna Stanford Vision and Learning Lab
- Fleet & Weiss, 2005 http://www.cs.toronto.edu/pub/jepson/teaching/vision/2503/opticalFlow.pdf
- Ioannis (Yannis) Gkioulekas, 16-385 Computer Vision – Lecture 24-25, Spring 2020, CMU
- Shapiro, EE596 – Optical flow, Universtiy of Washinton

https://homes.cs.washington.edu/~shapiro/EE596/notes/Optical_Flow.pdf