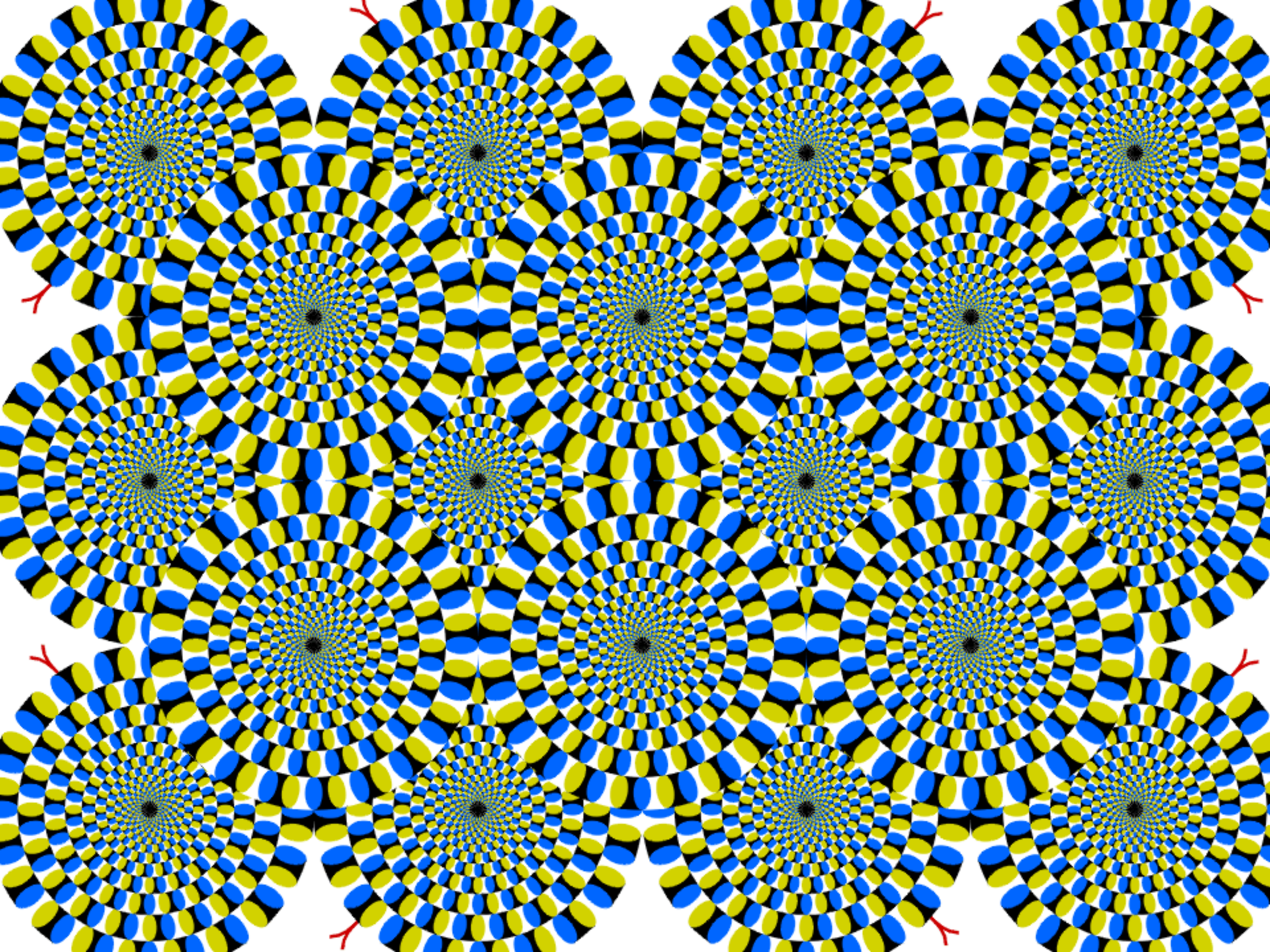
Digital Image Processing Introduction

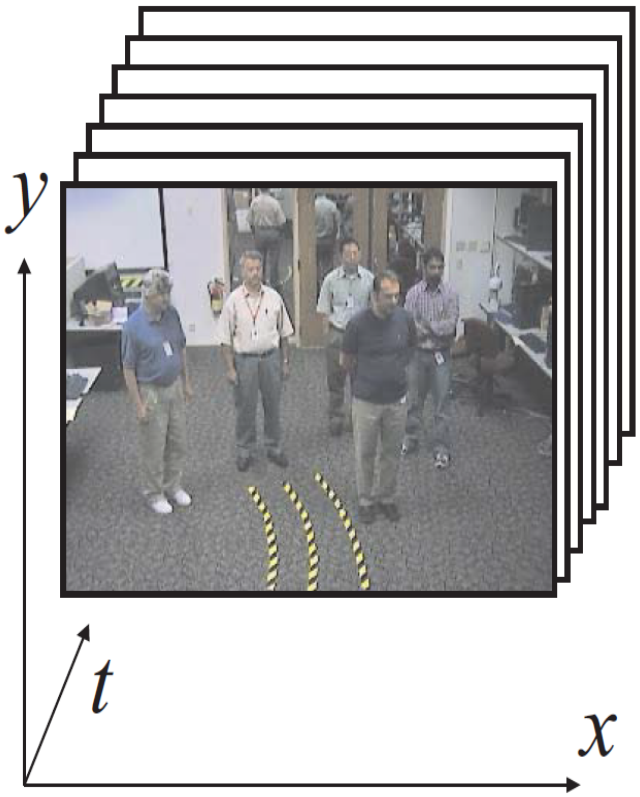
Object tracking & Motion detection in video sequences

Recomended link:

http://cmp.felk.cvut.cz/~hlavac/TeachPresEn/17CompVision3D/41ImageMotion.pdf



DYNAMIC SCENE ANALYSIS



The input to the dynamic scene analysis is a sequence of image frames $F(x, y, t)$ taken from the changing world.

x, y are spatial coordinates. Frames are usually captured at fixed time intervals.

t represents t-th frame in the sequence.

Typical Applikations

Motion detection. Often from a static camera. Common in surveillance systems. Often performed on the pixel level only (due to speed constraints).

Object localization. Focuses attention to a region of interest in the image. Data reduction. Often only interest points found which are used later to solve the correspondence problem.

Motion segmentation Images are segmented into region corresponding to different moving objects.

Three-dimensional shape from motion. Called also structure from motion. Similar problems as in stereo vision.

Object tracking. A sparse set of features is often tracked, e.g., corner points.

Pedestrians in underground station

Figure 1. Crowded Scene in an Underground Station. Pedestrians which cross the bright (blue) line at the foot of the elevator are being counted in this scenario.

Problem definition

Assuming that the scene illumination does not change, the image changes are due to **relative motion** between the scene objects and the camera (observer).

There are three possibilities:

- Stationary camera, moving objects.
- Moving camera, stationary objects.
- Moving camera, moving objects.

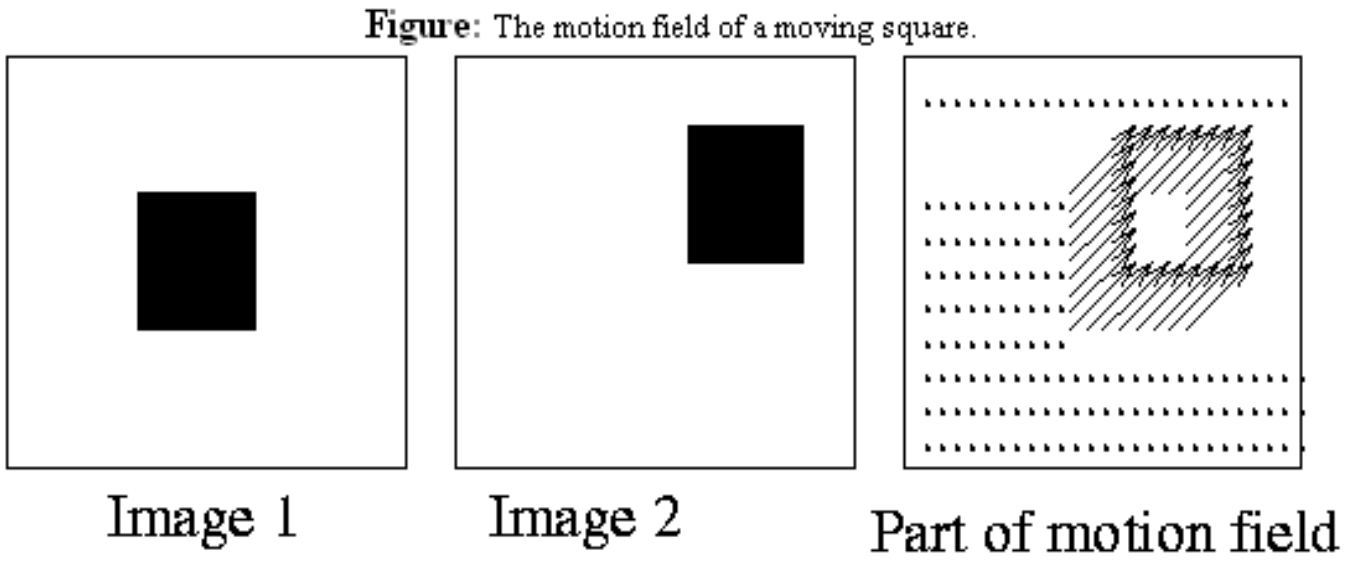
Motion field

- Motion field is a 2D representation in the image plane of a (generally) 3D motion of points in the scene (typically on surfaces of objects).

- *To the each point is assigned a velocity vector corresponding to the motion direction and velocity magnitude.*

Motion field

- Examle:



Estimating motion field

1. Matching of blocks.
2. Matching of objects. Interpretation needed. A sparse motion field.
3. Matching of interest points. A bit denser motion field. A more difficult interpretation. Problems with repetitive structures.
4. Optic flow, a differential technique matching intensities on a pixellevel, exploring spatial and time variations. Ideally, a dense motion field. Problems for textureless parts of images due to aperture problem (to be explained).

Segmentation in Video-sequences

- Background Subtraction
  - first must be defined a model of the background
  - this *background model* is compared against the current image and then the known background parts are subtracted away.
- The objects left after subtraction are presumably new foreground objects.

Establishing a Reference Images

Establishing a Reference Images:

– Difference will erase static object:

- When a dynamic object move out completely from its position, the back ground in replaced.

a b c

FIGURE 10.50 Building a static reference image. (a) and (b) Two frames in a sequence. (c) Eastbound automobile subtracted from (a) and the background restored from the corresponding area in (b). (Jain and Jain.)

Object tracking

- Object tracking is the problem of determining (estimating) the positions and other relevant information of moving objects in image sequences.
-
- Two-frame tracking can be accomplished using:
  - correlation-based matching methods
  - feature-based methods
  - optical flow techniques
  - change-based moving object detection methods

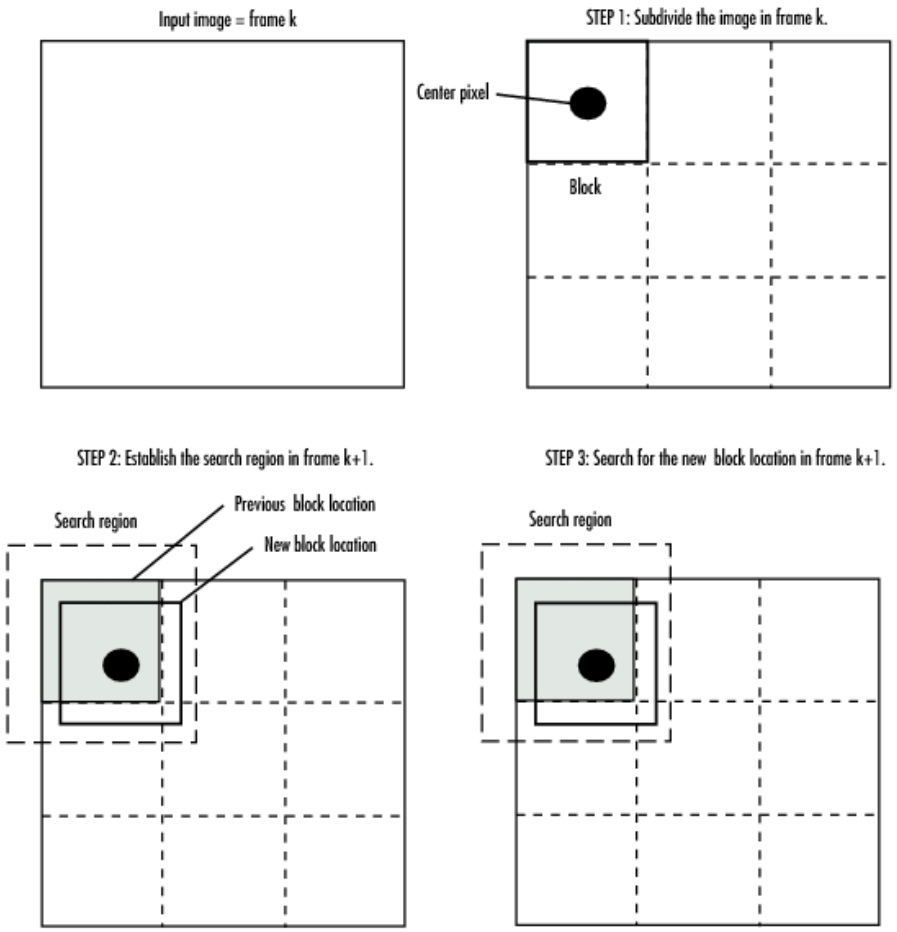
Main difficulties in reliable tracking of moving objects

- Rapid appearance changes caused by
  - image noise,
  - illumination changes,
  - non-rigid motion,
  - ...
- Non-stable background
- Interaction between multiple multiple objects
- ...

Block matching method

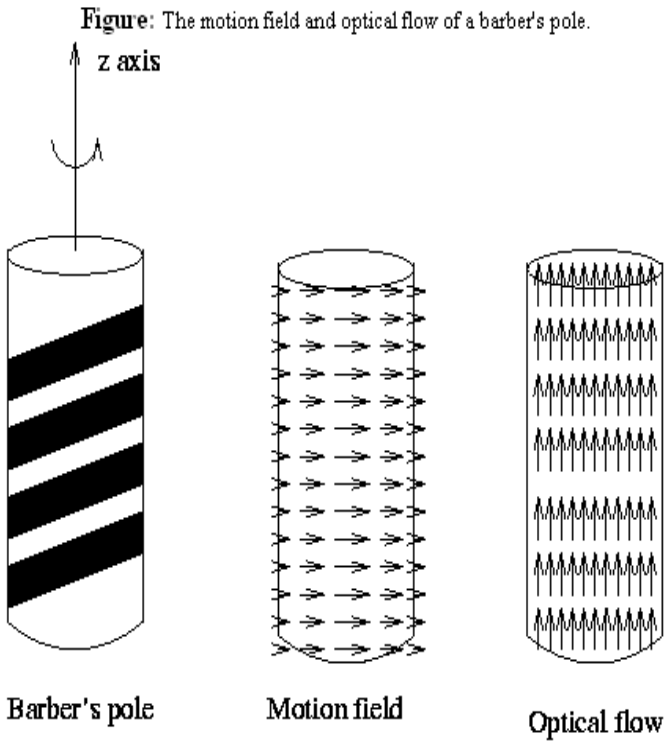
- Correlation-based tracking is BM method
- For a given region in one frame, find the corresponding region in the next frame by finding the maximum correlation score (or other block matching criteria) in a search region
- Blocks are typically squared and overlapped

Block matching method

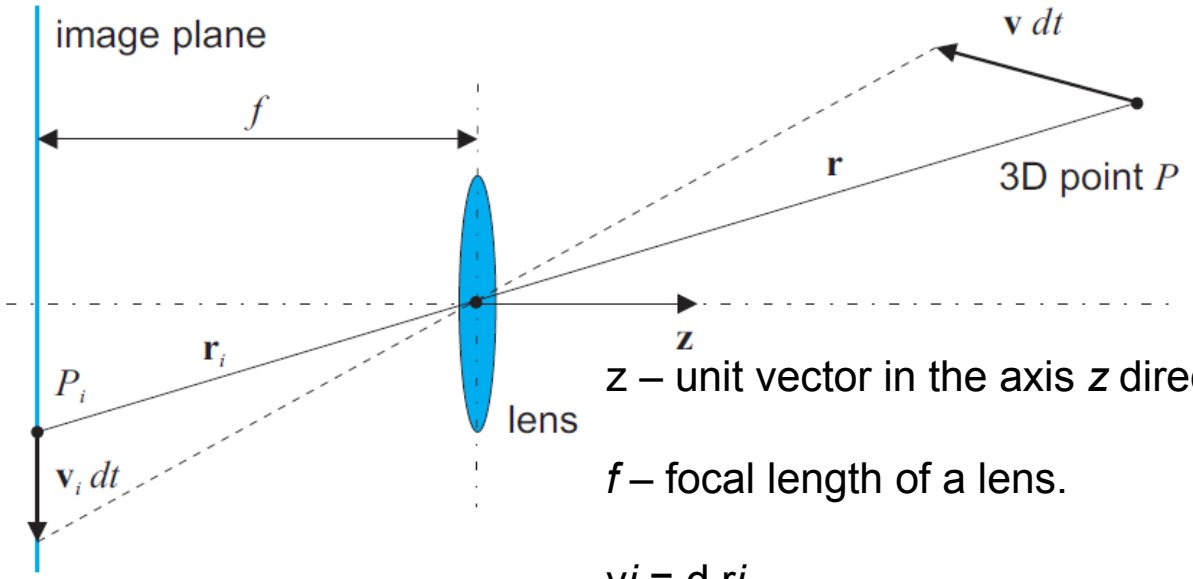


Visible Motion and True Motion

- Generally, optical flow corresponds to the motion field, but not always.
- For example, the motion field and optical flow of a rotating barber's pole are different...



Motion field / Optical flow



z – unit vector in the axis *z* direction.

f – focal length of a lens.

v*i* = d r*i*
d *t* – velocity in the image plane.

v = d r
d *t* – velocity in the 3D space.

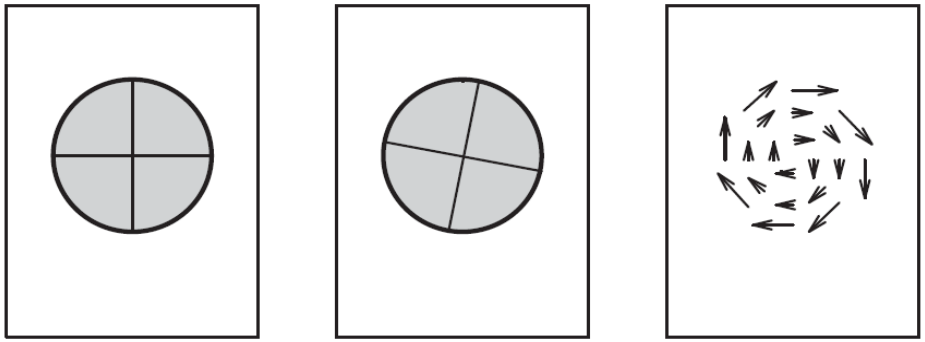
- Optical flow is a vector field, from which the motion field can be estimated under certain conditions.

Optical flow

OPTIC FLOW = apparent motion of the same (similar) intensity patterns

Ideally, the optic flow can approximate projections of the three-dimensional motion (i.e., velocity) vectors onto the image plane.

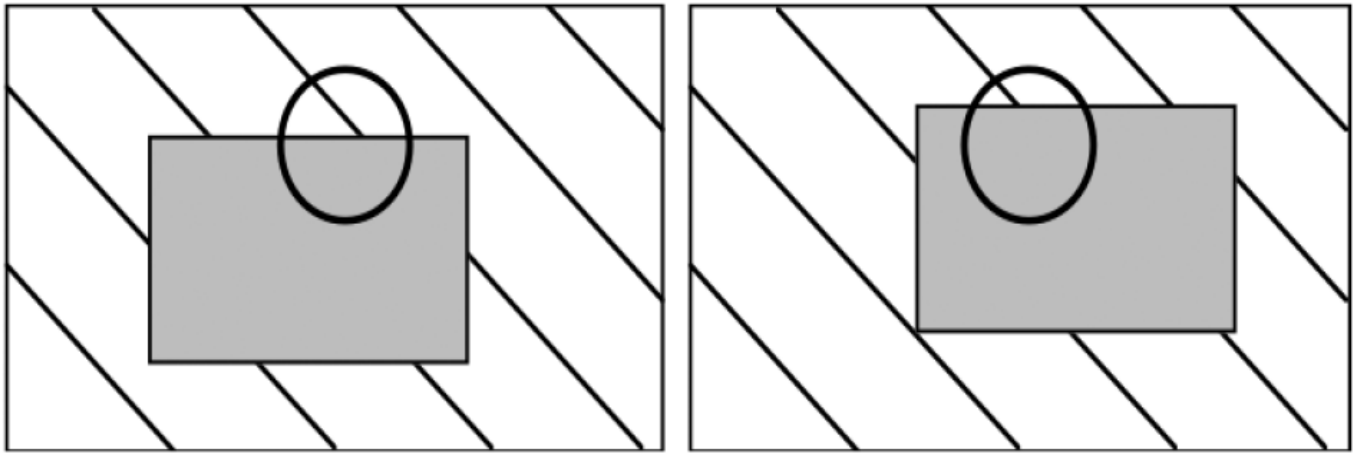
Hence, the optic flow is estimated instead of the motion field (since the latter is unobservable in general).



Time t_1 Time t_2 Optic flow

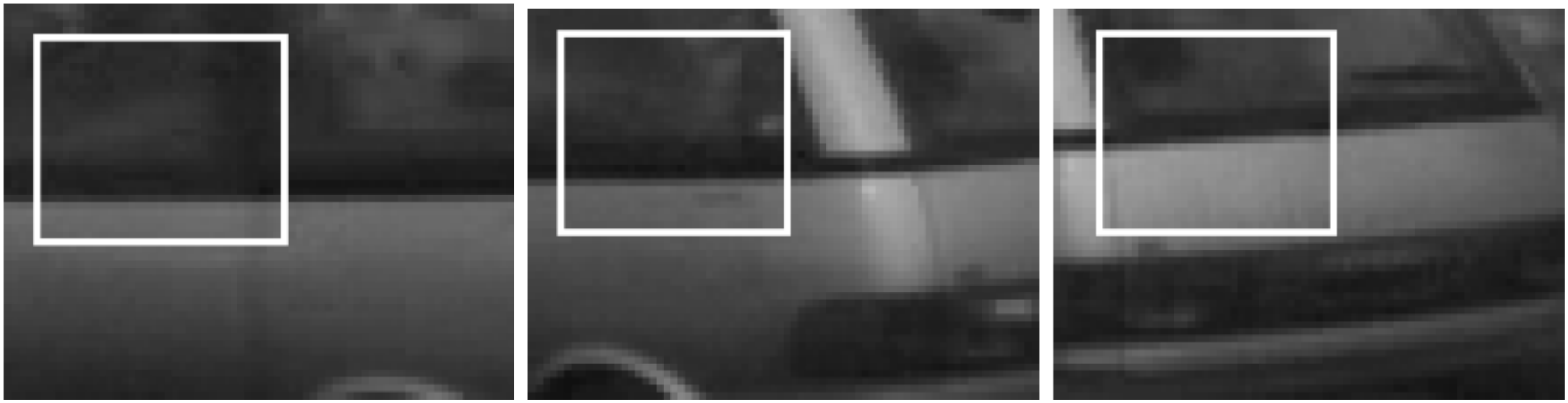
Aperture problem

- By analyzing projected images, we always observe a limited area of visible motion only (defined by aperture of the camera or of the area of influence of the used algorithm for motion analysis).



Aperture problem -example

• For example, assume that we either only see the inner rectangles, or the whole three images taken at times t, t + 1, t + 2. For the inner rectangles, we may conclude an upward translation with a minor rotation....



Aperture problem

- we are only able to measure the component of optical flow that is in the direction of the intensity gradient. We are unable to measure the component tangential to the intensity gradient.

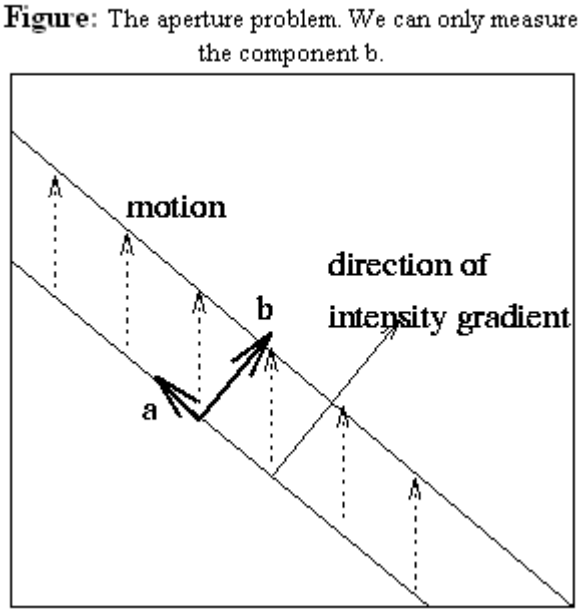
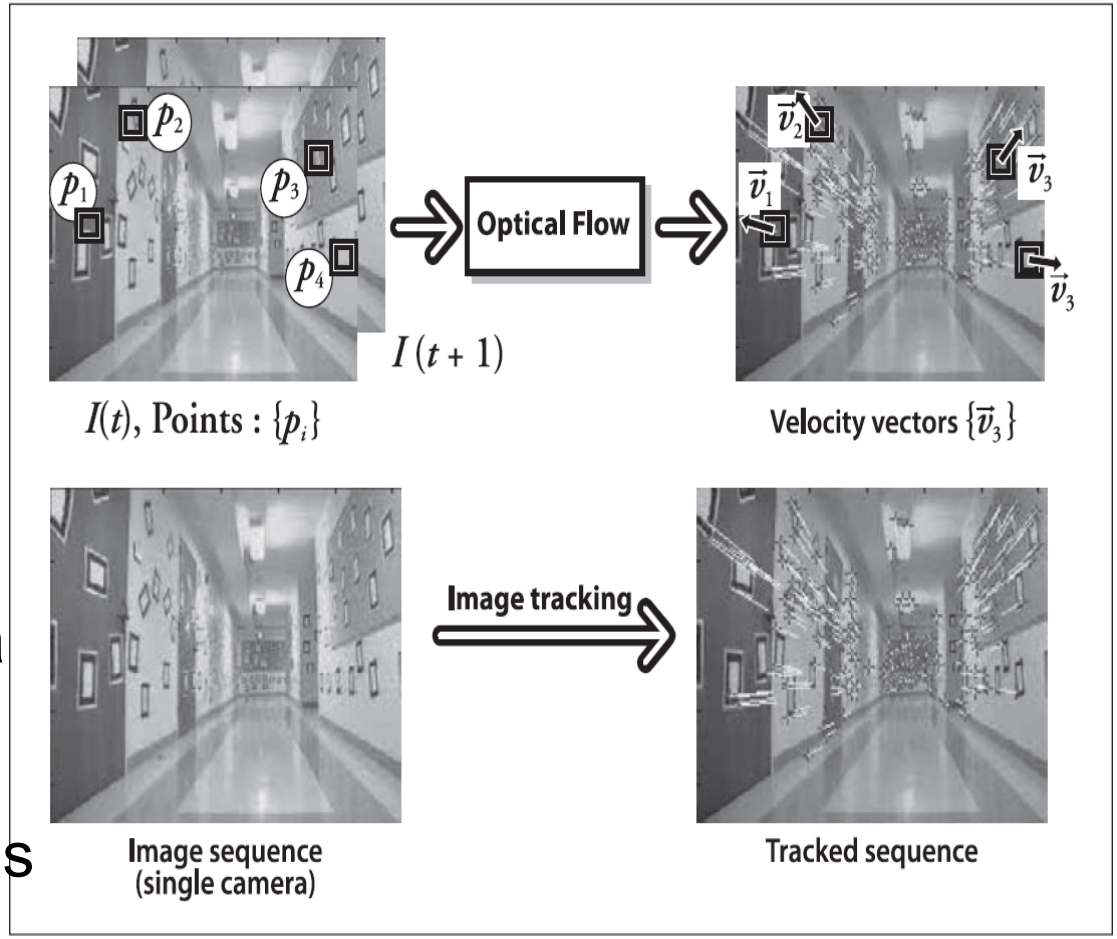


Figure: The aperture problem. We can only measure the component b.

Optical flow

- target features are tracked over time and their movement is converted into velocity vectors (upper right);
- lower panels show a single image of the hallway (left)and flow vectors (right) as the camera moves down the hall



- (original images courtesy of Jean-Yves Bouguet)

Optical flow

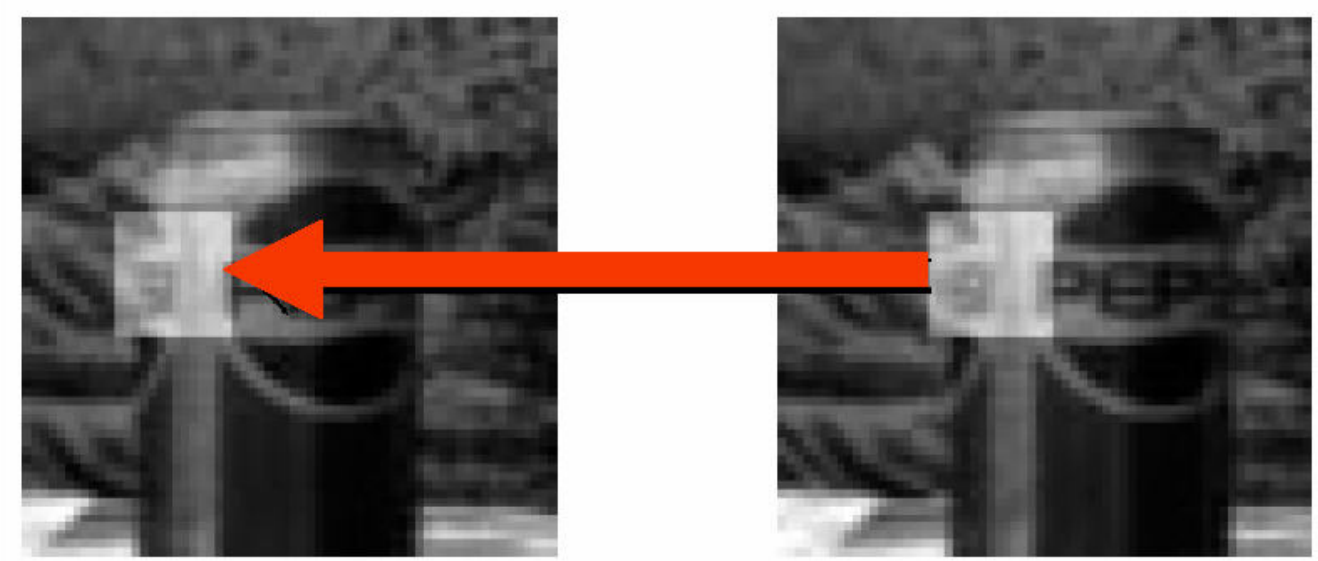
Common assumption - *Brightness constancy*:

The appearance of the image patches do not change (brightness constancy)

$$I(p_i, t) = I(p_i + \vec{v}_i, t + 1)$$

A pixel from the image of an object in the scene does not change in appearance as it (possibly) moves from frame to frame. For grayscale images this means we assume that the brightness of a pixel does not change as it is tracked from frame to frame.

Optical Flow Assumptions: Brightness constancy

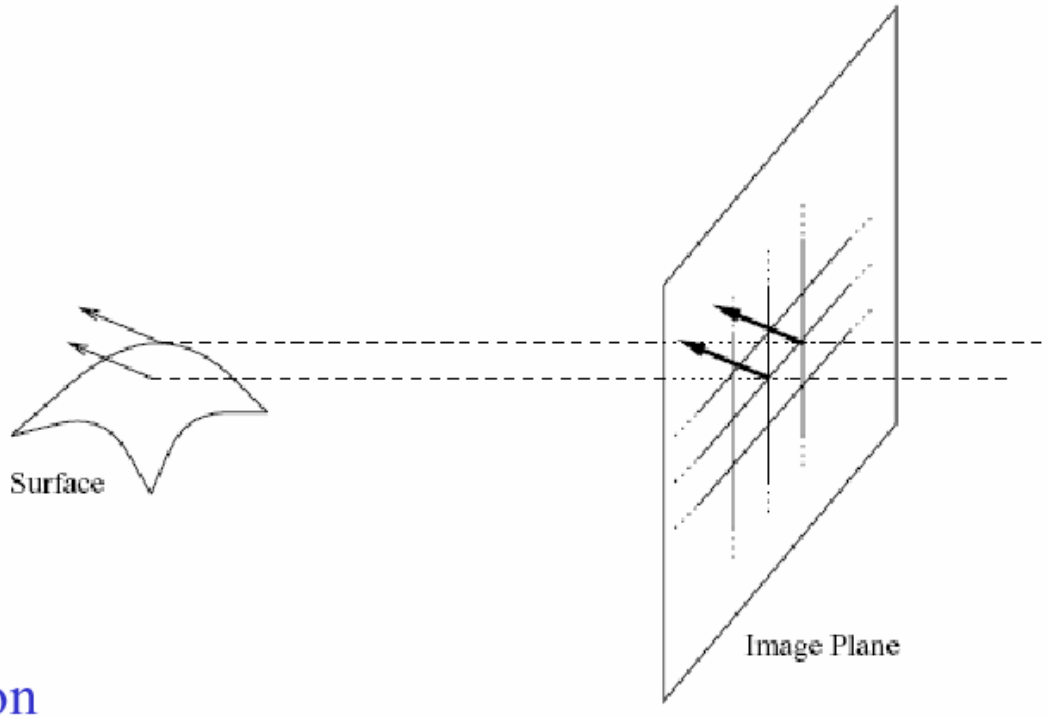


Assumption

Image measurements (e.g. brightness) in a small region remain the same although their location may change.

Optical Flow Assumptions:

Spatial Coherence

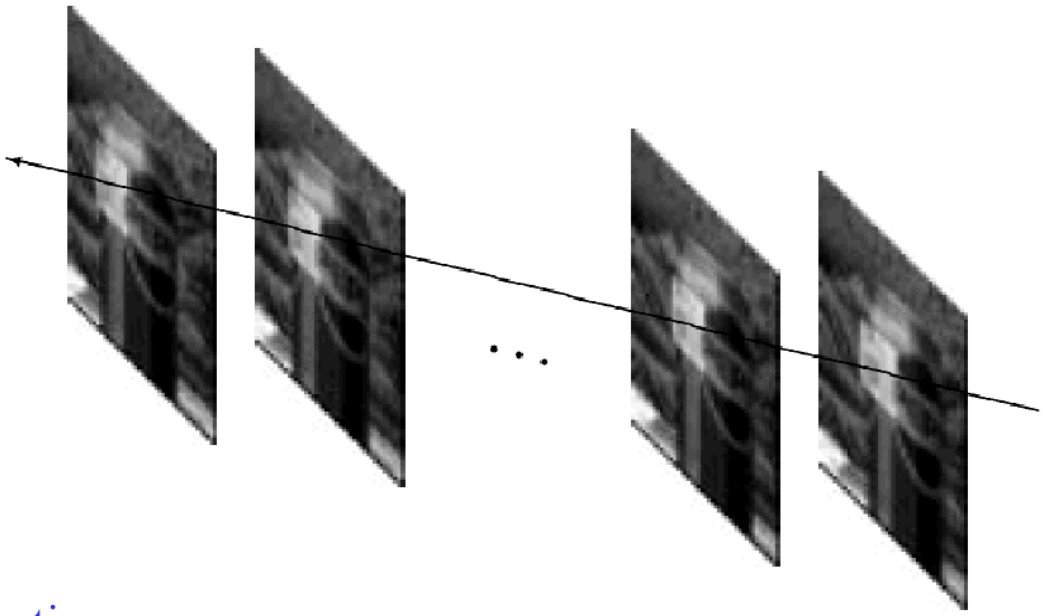


Assumption

* Neighboring points in the scene typically belong to the same surface and hence typically have similar motions.
* Since they also project to nearby points in the image, we expect spatial coherence in image flow.

Optical Flow Assumptions: Temporal persistence

- *Temporal persistence or "small movements"*. The image motion of a surface patch changes slowly in time. In practice, this means the temporal increments are fast enough relative to the scale of motion in the image that the object does not move much from frame to fra



Assumption:

The image motion of a surface patch changes gradually over time.

Optical Flow: 1D Case

Brightness Constancy Assumption:

$$f(t) \equiv I(\underbrace{x(t), t}) = I(x(t+dt), t+dt)$$

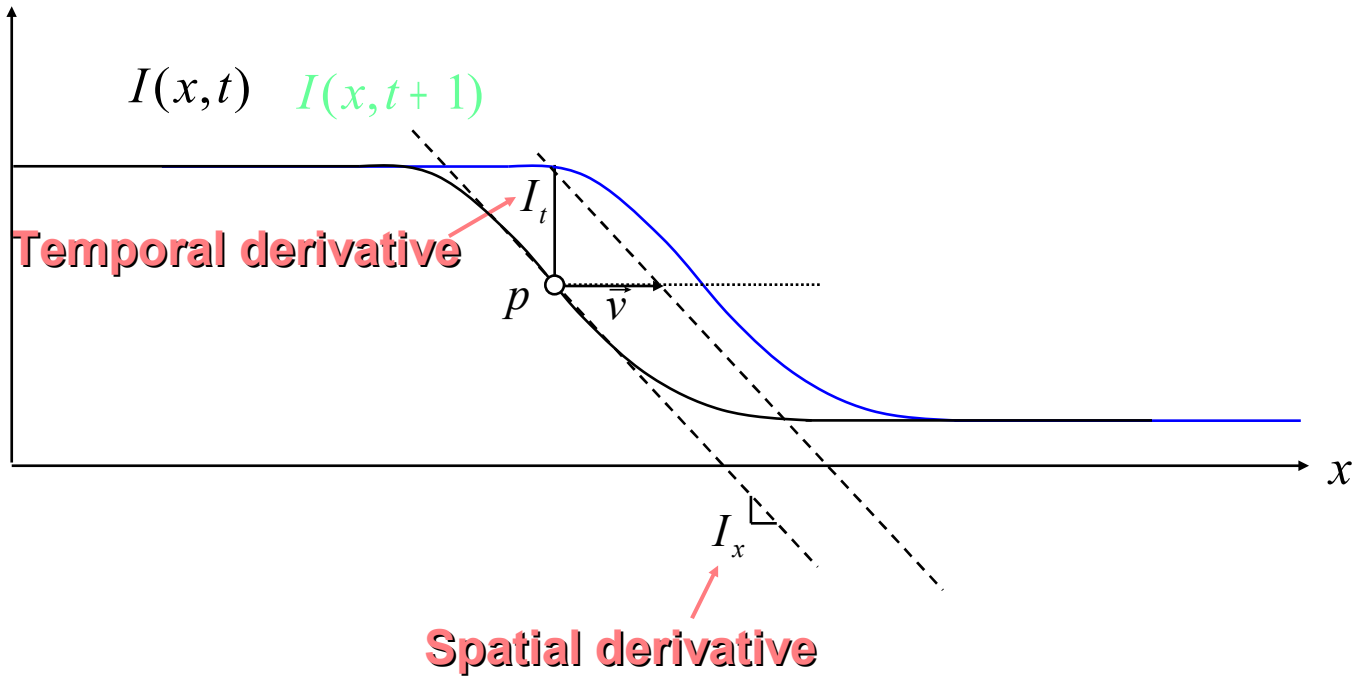
$$\frac{\partial f(x)}{\partial t} = 0$$ Because no change in brightness with time

$$\left.\frac{\partial I}{\partial x}\right|_t \left(\frac{\partial x}{\partial t}\right) + \left.\frac{\partial I}{\partial t}\right|_{x(t)} = 0$$

I_x v I_t

$$\Rightarrow v = \frac{I_t}{I_x}$$

Tracking in the 1D case:



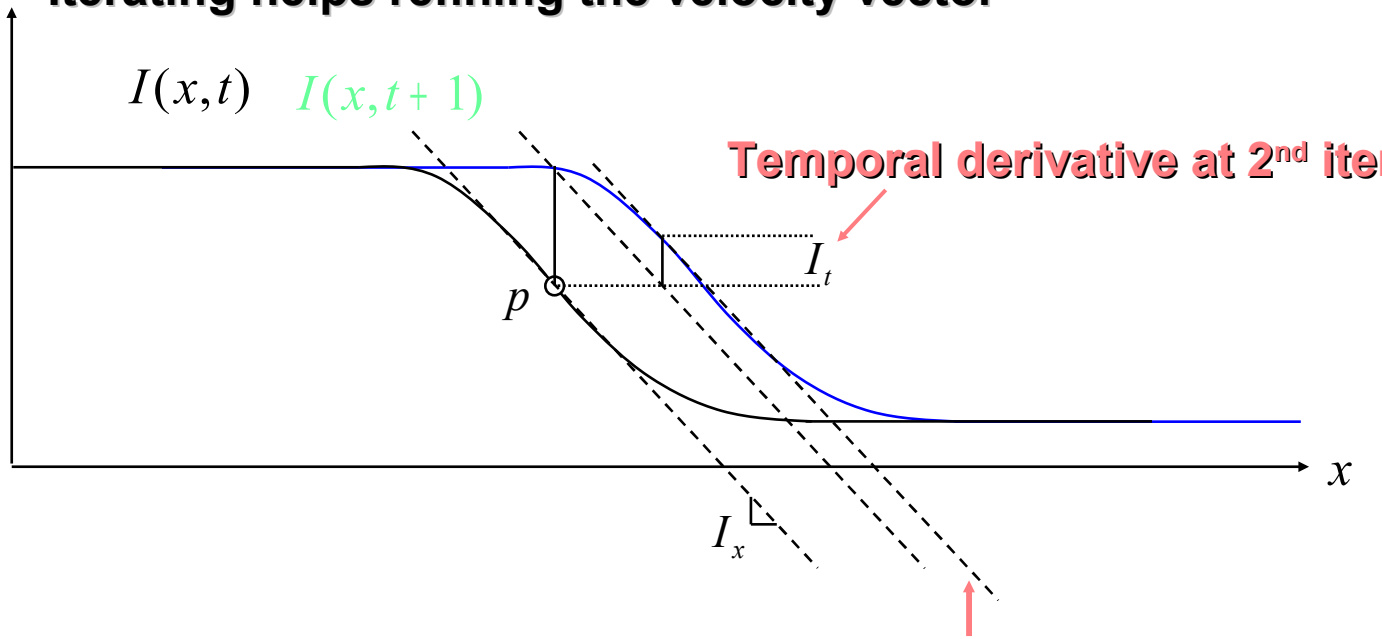
$$I_x = \left.\frac{\partial I}{\partial x}\right|_t \qquad I_t = \left.\frac{\partial I}{\partial t}\right|_{x=p} \qquad \Rightarrow \qquad \vec{v} \approx -\frac{I_t}{I_x}$$

Assumptions:

- **Brightness constancy**
- **Small motion**

Tracking in the 1D case:

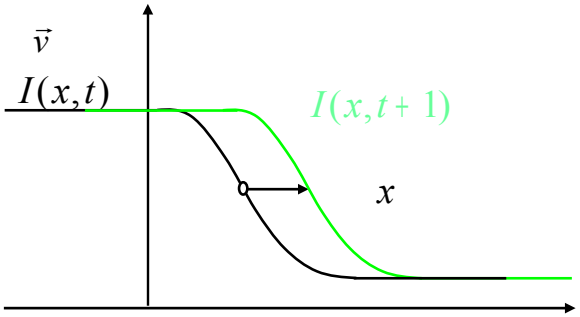
Iterating helps refining the velocity vector

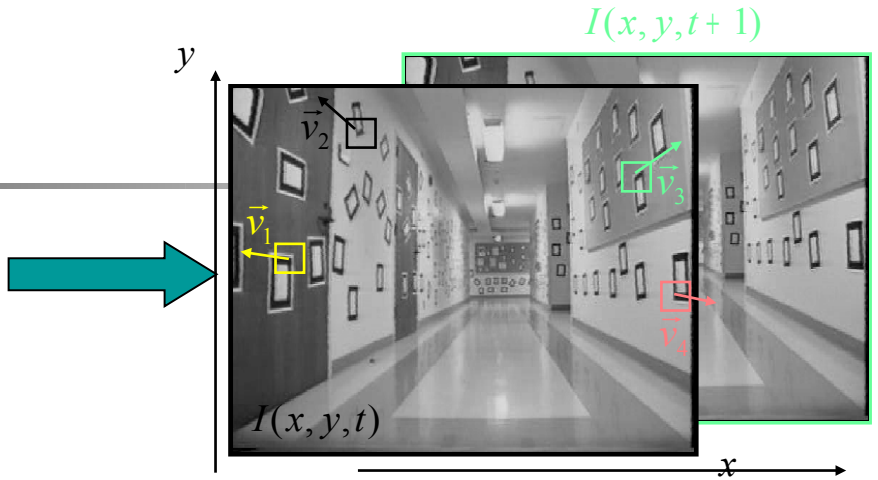


$$\vec{v} \leftarrow \vec{v}_{previous} - \frac{I_t}{I_x}$$

Converges in about 5 iterations

From 1D to 2D tracking





The Math is very similar:

$$\vec{v} \approx -\frac{I_t}{I_x}$$

$$\begin{cases} \vec{v} \approx -G^{-1}b \\ G = \sum_{\text{window around } p} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \\ b = \sum_{\text{window around } p} \begin{bmatrix} I_x I_t \\ I_y I_t \end{bmatrix} \end{cases}$$

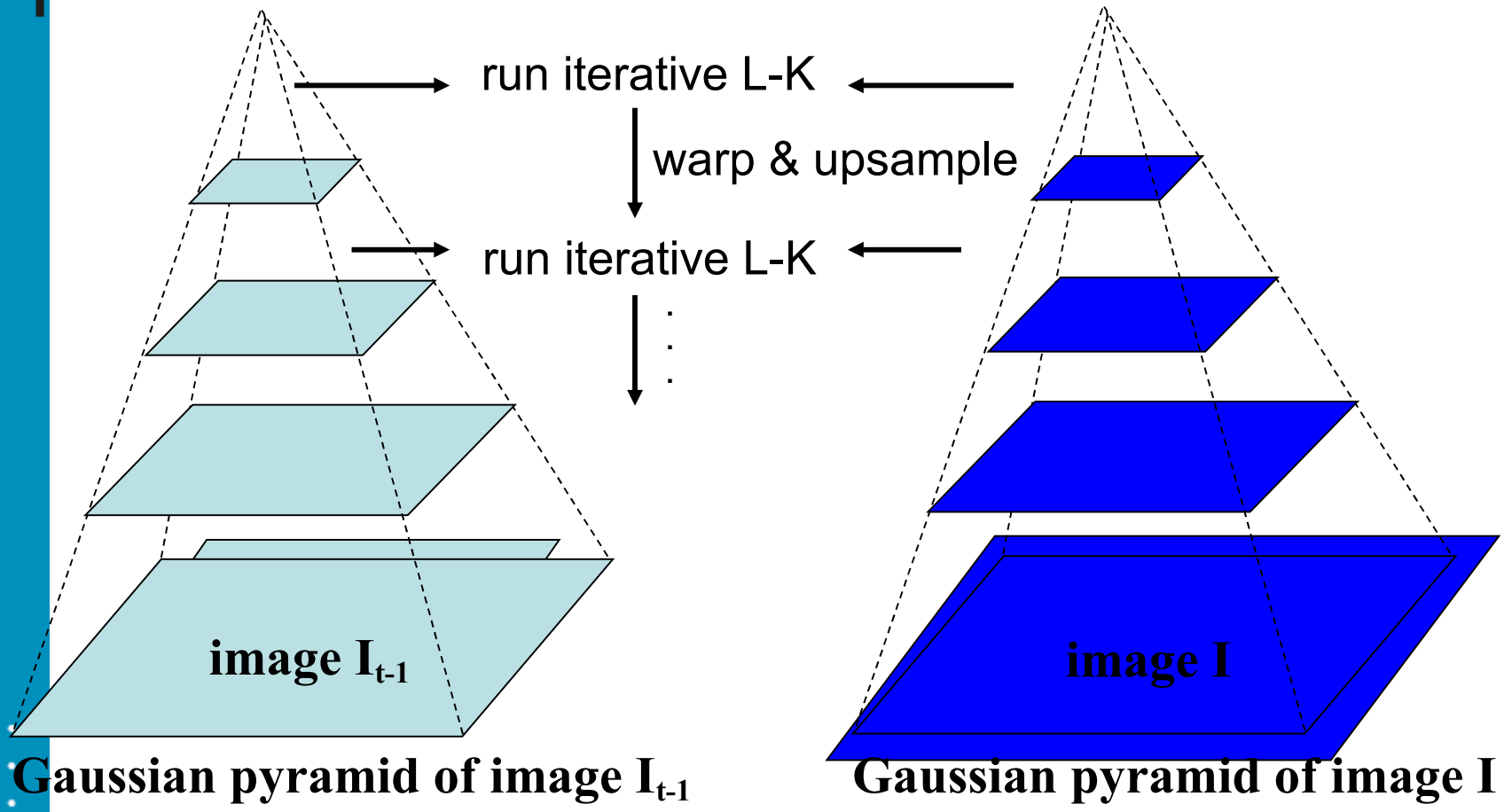
KLT in Pyramids

Problem:
we want a large window to catch large motions, but a large window too oft en breaks the coherent motion assumption!
To circumvent this problem, we can track first over larger spatial scales using an image pyramid and then refine the initial motion velocity assumptions by working our way down the levels of the image pyramid until we arrive at the raw image pixels.

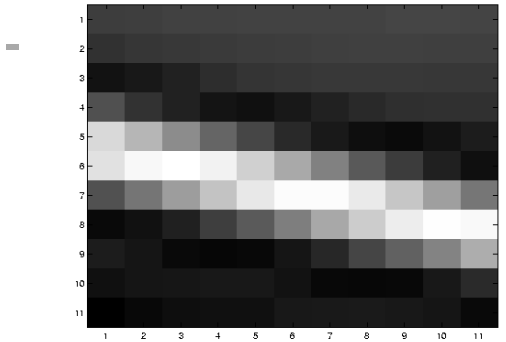
KLT in Pyramids

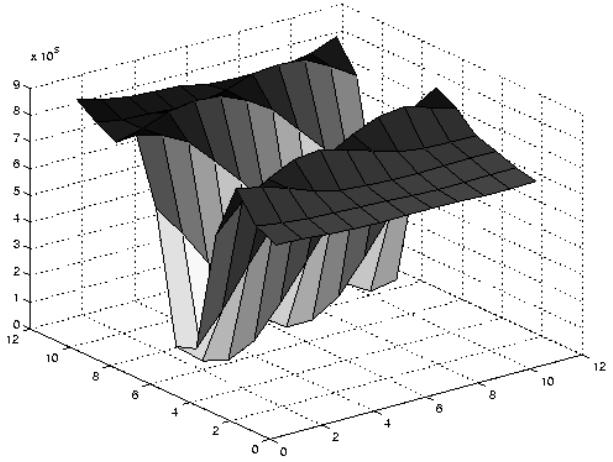
First solve optical flow at the top layer and
then to use the resulting motion estimates as the starting point for the next layer down.
We continue going down the pyramid in this manner until we reach the lowest level.
Thus we minimize the violations of our motion assumptions and so can track faster and longer motions.

Coarse-to-fine optical flow estimation



Edge



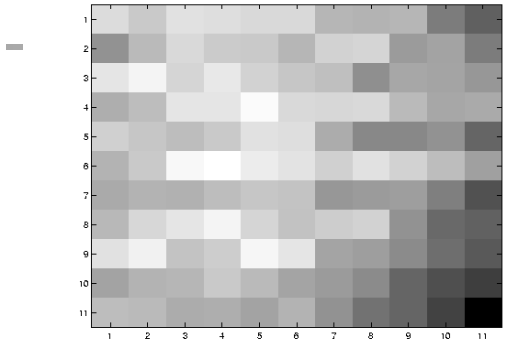


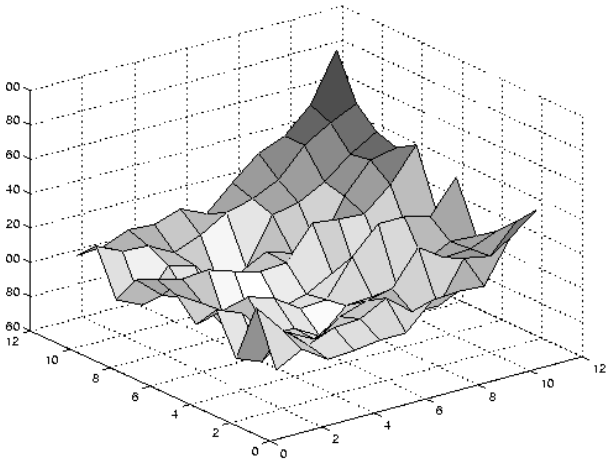
$$\sum \nabla I (\nabla I)^T$$

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

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Low texture region



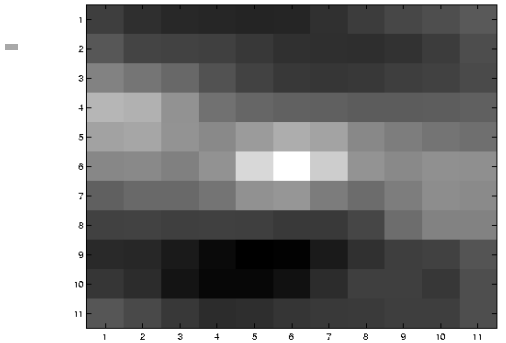


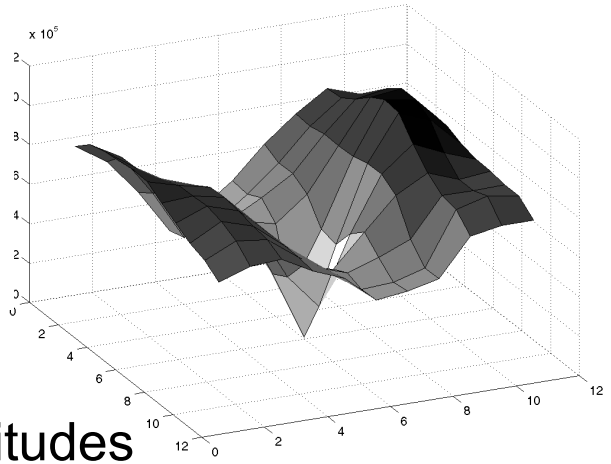
$$\sum \nabla I (\nabla I)^T$$

- gradients have small magnitude
- small λ_1, small λ_2

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

High textured region





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

- gradients are different, large magnitudes
- large λ_1, large λ_2

* From Khurram Hassan-Shafique CAP5415 Computer Vision 2003

Local features for tracking

- ...*There are many kinds of local features that one can track...*
- If strong derivatives are observed in two orthogonal directions then we can hope that this point is more likely to be unique.
- => many trackable features are called *corners*. Intuitively, corners—not edges—are the points that contain enough information to be picked out from one frame to the next.

Harris corner detection

- The definition relies on the matrix of the second-order derivatives (*∂2x*, *∂2 y*, *∂x ∂y*) of the image intensities.

Hessian matrix around a point:

$$H(p) = \begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x\, \partial y} \\ \frac{\partial^2 I}{\partial y\, \partial x} & \frac{\partial^2 I}{\partial y^2} \end{bmatrix}_p$$

Autocorrelation matrix of the second derivative images over a small window around each point:

$$M(x,y) = \begin{bmatrix} \sum_{-K \le i,j \le K} w_{i,j} I_x^2(x+i, y+j) & \sum_{-K \le i,j \le K} w_{i,j} I_x(x+i, y+j) I_y(x+i, y+j) \\ \sum_{-K \le i,j \le K} w_{i,j} I_x(x+i, y+j) I_y(x+i, y+j) & \sum_{-K \le i,j \le K} w_{i,j} I_y^2(x+i, y+j) \end{bmatrix}$$

Harris corner detection

-
- Harris's definition:
- Corners are places in the image where the autocorrelation matrix of the second derivatives has two large eigenvalues.
  - => there is texture (or edges) going in at least **two separate directions**
- these two eigenvalues do more than determine if a point is a good feature to track; they also provide an identifying signature for the point.

Shi and Tomasi: Good Features to Track

- Shi and Tomasi:
- Good corner results as long as the smaller of the two eigenvalues is greater than a minimum threshold.
- Shi and Tomasi's method was not only sufficient but in many cases gave more satisfactory results than Harris's method.

Global approach: Horn-Schunck

- A method for finding the optical flow pattern which assumes that the apparent velocity of the brightness pattern varies smoothly almost everywhere in the image.
- An iterative implementation computes the optical flow of each pixel.

Horn-Schunck Algorithm

- Two criteria:
  - Optical flow is smooth, $\mathbf{F}_s(u,v)$
  - Small error in optical flow constraint equation, $\mathbf{F}_h(u,v)$
- Minimize a combined error functional

$$\mathbf{F}_c(u,v) = \mathbf{F}_s(u,v) + \lambda\, \mathbf{F}_h(u,v)$$

  λ is a weighting parameter

- Variation calculus gives a pair of second order differential equations that can be solved iteratively

Advantages / Disadvantages of the Horn–Schunck algorithm

- Advantages :
  - it yields a high density of flow vectors, i.e. the flow information missing in inner parts of homogeneous objects is filled in from the motion boundaries.
- Disadvantages:
  - it is more sensitive to noise than local methods.
  - The Horn-Schunck method is very unstable in case of illumination artifacts in image sequences (due to the assumed brightness constancy).

Mean-Shift Tracking

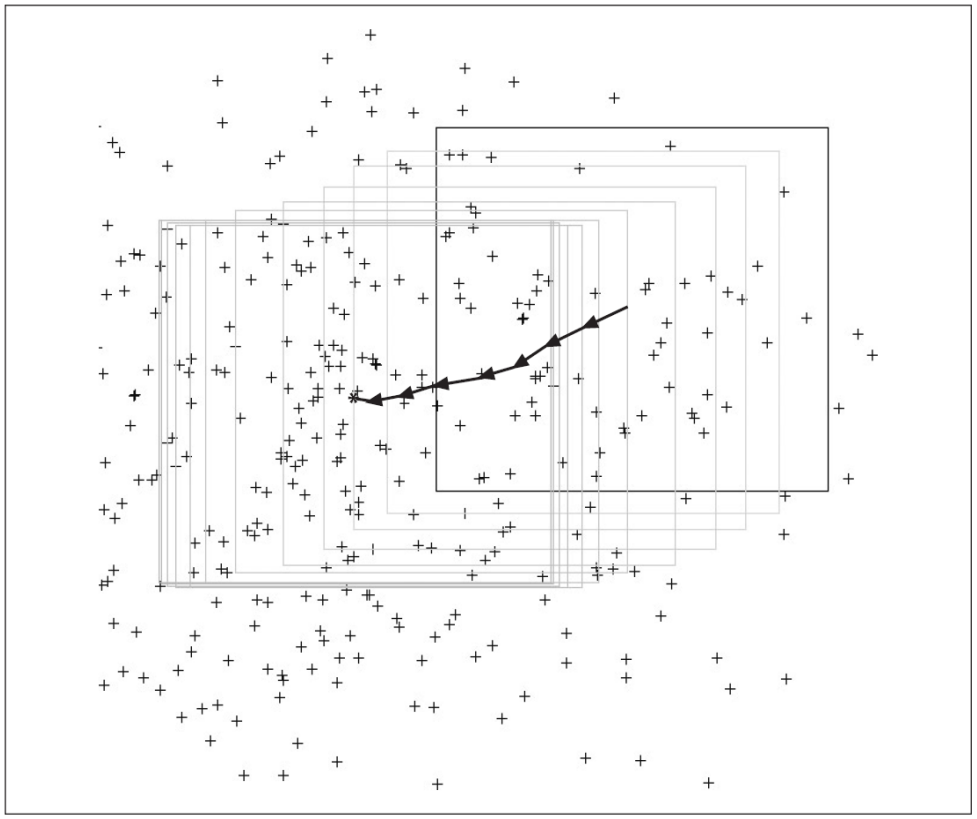
- **Mean-Shift in tracking task:**
  - track the motion of a cluster of interesting features.
- 1. choose the feature distribution to represent an object (e.g., color + texture),
- 2. start the mean-shift window over the feature distribution generated by the object
- 3. finally compute the chosen feature distribution over the next video frame.

Mean-Shift Tracking

- 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 fi rst place.
- => In this way, the mean-shift window tracks the movement of the object frame by frame.

Mean-shift algorithm in action

- An initial window is placed over a two-dimensional array of data points and is successively recentered over the mode (or local peak) of its data distribution until convergence



Mean-shift algorithm

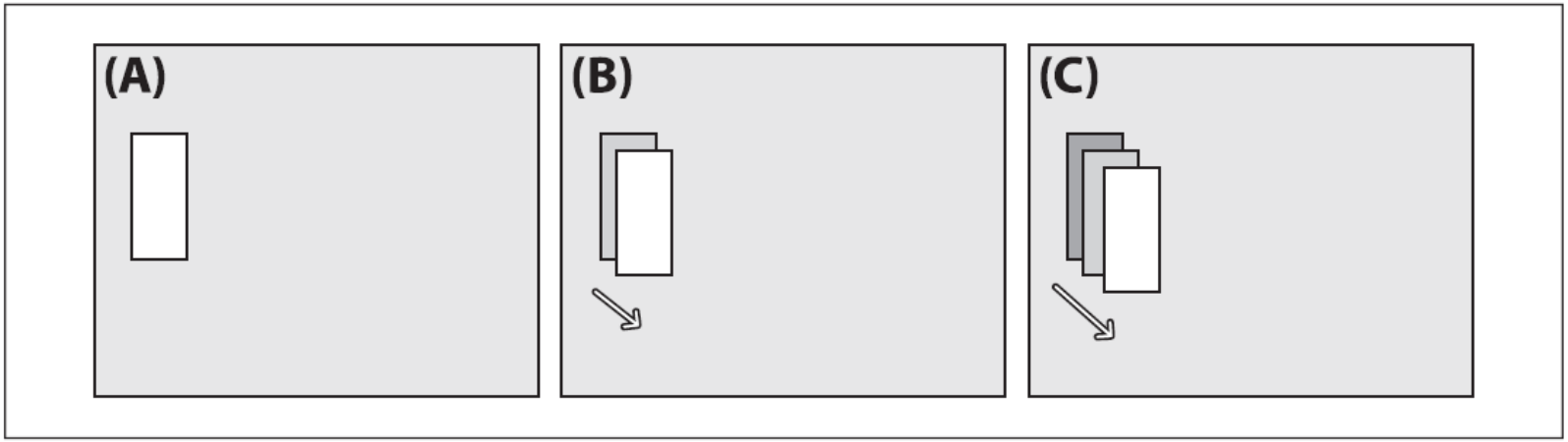
- The mean-shift algorithm runs as follows:
- 1. Choose a search window:
  - • its initial location;
  - • its type (uniform, polynomial, exponential, or Gaussian);
  - • its shape (symmetric or skewed, possibly rotated, rounded or rectangular);
  - • its size (extent at which it rolls off or is cut off).
- 2. Compute the window's (possibly weighted) center of mass.
- 3. Center the window at the center of mass.
- 4. Return to step 2 until the window stops moving (it always will).*

Camshift -
continuously adaptive mean-shift

- It differs from the meanshift in that the search window adjusts itself in size.

Motion Templates

- Track general movement and are especially applicable to gesture recognition.
- Gradient method
- Using motion templates requires a **silhouette** (or part of a silhouette) of an object.



Motion Templates Object silhouettes:

- **Object silhouettes** can be obtained in a number of ways:
  - 1. use a reasonably stationary camera and then employ frame-to-frame differencing . Th is will give you the moving edges of objects, which is enough to make motion templates work.
  - 2. You can use chroma keying. For example, if you have a known background, you can simply take as foreground anything that is not background.

Motion Templates
Object silhouettes:

To learn a background model from which you can isolate new foreground objects/people as silhouettes.

1. You can use active silhouetting techniques—for example, creating a wall of nearinfrared light and having a near-infrared-sensitive camera look at the wall. Any intervening object will show up as a silhouette.
2. You can use thermal images; then any hot object (such as a face) can be taken as foreground.
3. Finally, you can generate silhouettes by using the segmentation techniques ....

http://www.youtube.com/watch?v=-Wn7dv_nzbA

Tracking Cars Using Optical Flow Results -Mathworks

The model uses an optical flow estimation technique to estimate the motion vectors in each frame of the video sequence.

By thresholding and performing morphological closing on the motion vectors, the model produces binary feature images.

- The model locates the cars in each binary feature image using the Blob Analysis block.

Change-based blob tracking

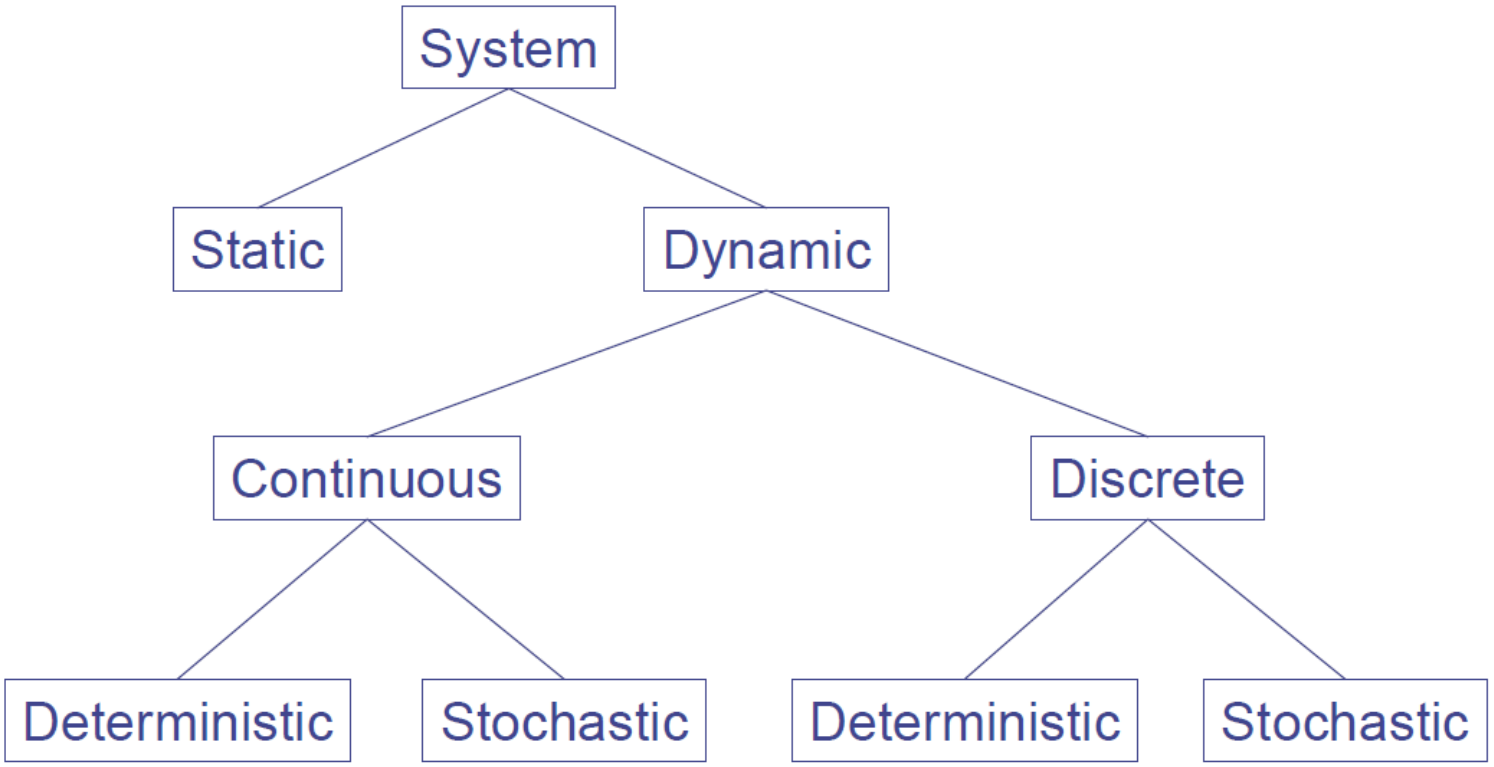
- Algorithm
  - Calculate the background
  - Image subtraction to detection motion blobs
  - Compute the difference image between two frames
  - Thresholding to find the blobs
  - Locate the blob center as the position of the object

Estimation - Prediction

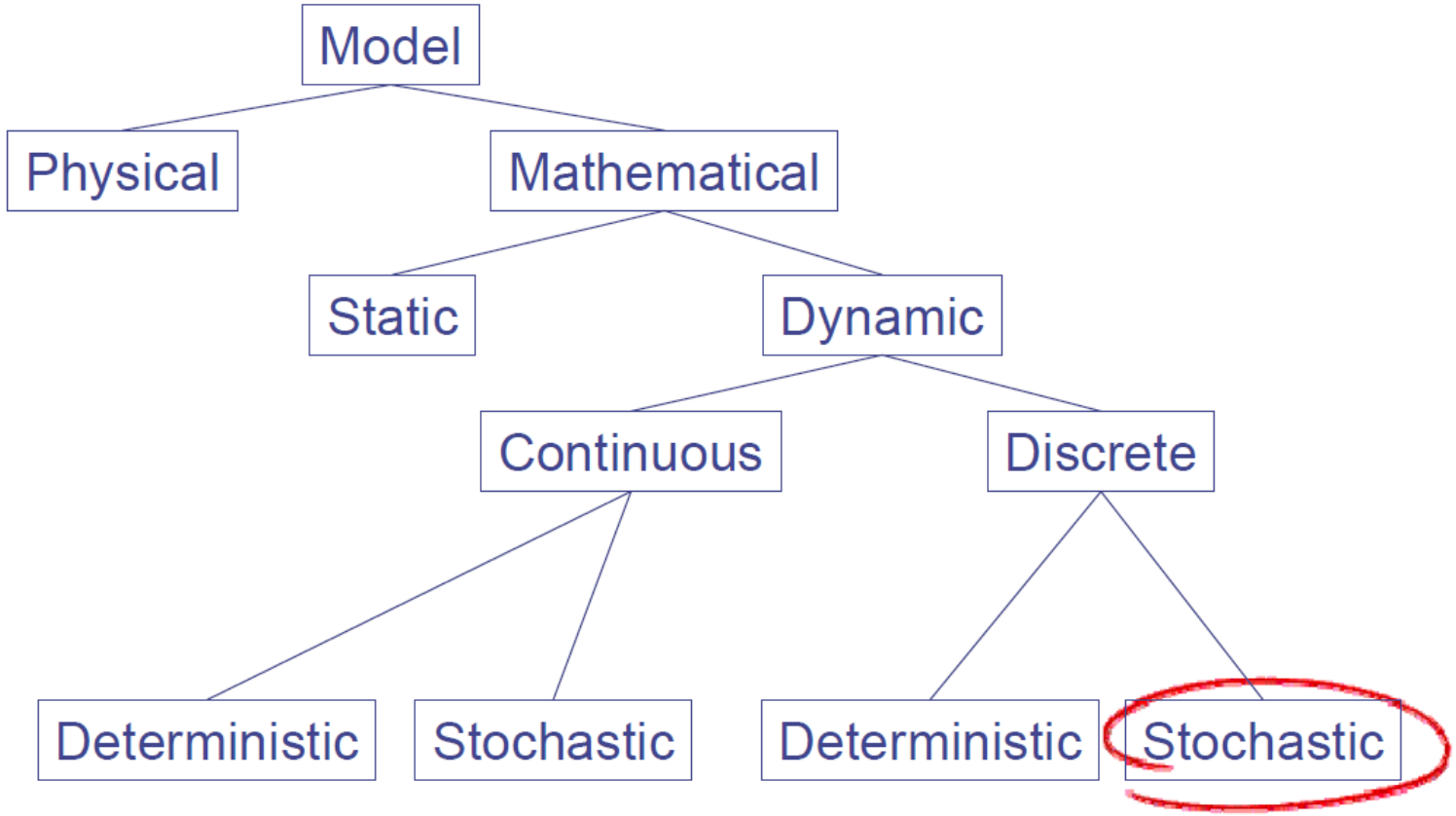
- In a long image sequence, if the dynamics of the moving object is known, prediction can be made about the positions of the objects in the current image. This information can be combined with the actual image observation to achieve more robust results

Prediction

- System



Model of system



Tracking formulated using the Hidden Markov model (HMM)

- Hidden Markov model (HMM):

x_k is called the hidden state and z_k is called the observation

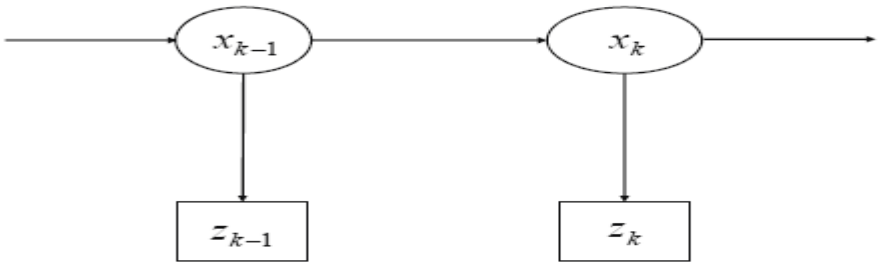
Markov chain

$$p(x_k \mid x_1,\ldots,x_{k-1}) = p(x_k \mid x_{k-1})$$

$$p(z_k \mid x_1,\ldots,x_k) = p(z_k \mid x_k)$$

As the result

$$p(x_1,\ldots,x_k,z_1,\ldots,z_k) = p(x_1)p(z_1 \mid x_1)\prod^{k}[p(x_i \mid x_{i-1})p(z_i \mid x_i)]$$



Tracking formulated using the Hidden Markov model

- The state *xk* usually consists of the position, the velocity, the acceleration, and other features of the object
- The observation *zk* is usually the video frame at current time instance.
- The transition probability is modeled using dynamic model such as constant velocity model, or constant acceleration model
- Example: A constant velocity dynamic model

The dynamic system

The dynamic system and the state

$$\mathbf{x}_k = \Phi \mathbf{x}_{k-1} + \xi$$

$$\mathbf{z}_k = H\mathbf{x}_k + \mu$$

where $\mathbf{x}_k$ is the a state vector at time instant k, e.g. $\mathbf{x}_k = [x_k, y_k, v_{xk}, v_{yk}]$,

$\mathbf{z}_k$ is the a observation vector at time instant k, e.g. $\mathbf{z}_k = [z_x, z_y]$,

Φ is the state transition matrix, e.g. $\Phi = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

H is the measurement matrix, e.g. $H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$

ξ is a random vector modelling the uncertainty of the model, assumed to be $N(0, Q)$

μ is a random vector modelling teh additive noise in the observation, assumed to be $N(0, R)$

The dynamic system

- general problem of trying to estimate the state ***x*** of a discrete-time controlled process is governed by the linear stochastic difference equation

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1},$$

with a measurement ***z*** that is:

$$z_k = Hx_k + v_k.$$

The random variables *wk* and *vk* represent the process and measurement noise (respectively).
They are assumed to be independent (of each other), white, and with normal probability distributions

$$p(w) \sim N(0, Q),$$

$$p(v) \sim N(0, R).$$

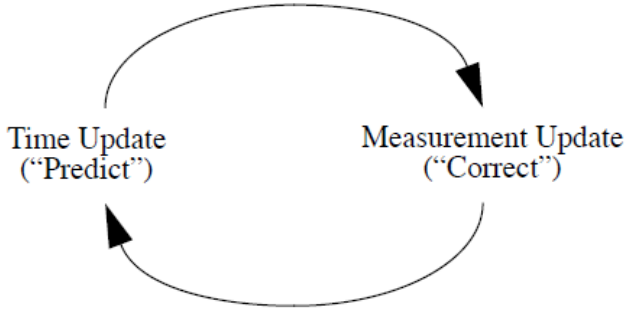
In practice, the process noise covariance Q and *measurement noise covariance R matrices* might *change with each time step or measurement, however here we assume they are constant.*

Kalman filter

- The Kalman filter is a set of mathematical equations that provides an efficient computational (recursive) means to estimate the state of a process, in a way that minimizes the mean of the squared error.

Kalman filter

- Two groups of the equations for the Kalman filter:
  - *time update* equations
  - and *measurement update* equations.



The time update equations are responsible for projecting forward (in time) the current state and error covariance estimates to obtain the *a priori* estimates for the next time step.

The measurement update equations are responsible for the feedback—i.e. for incorporating a new measurement into the *a priori* estimate to obtain an improved *a posteriori* estimate.

Kalman filter

- the time update equations can also be thought of as *predictor* equations
- the measurement update equations can be thought of as *corrector* equations.
  - *predictor-corrector* algorithm => solving numerical problems:
  -

Discrete Kalman filter time update equations:

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$$

$$P_k^- = AP_{k-1}A^T + Q$$

time update equations project the state and covariance estimates forward from time step k-1 to step k .

Discrete Kalman filter measurement update equations:

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$$

$$P_k = (I - K_k H)P_k^-$$