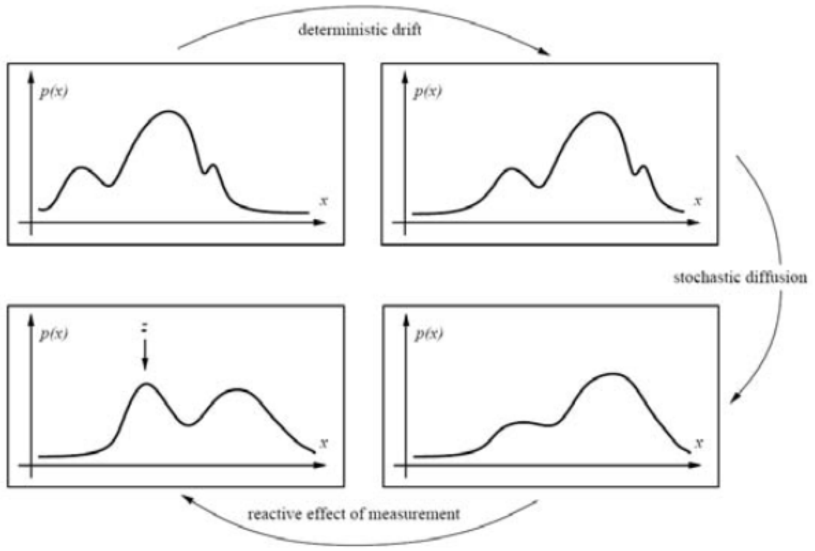
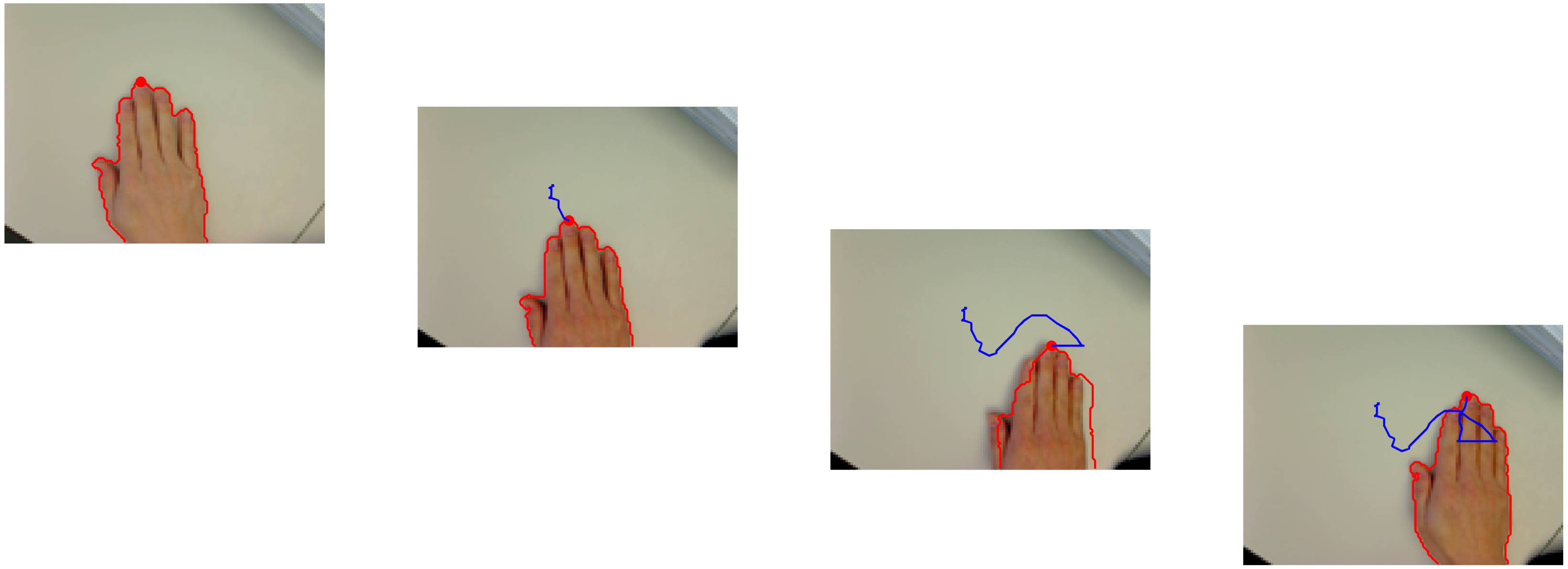


COMP9517: Computer Vision

Object Tracking

Motion Tracking

- Tracking is the problem of generating an inference about the motion of an object given a sequence of images



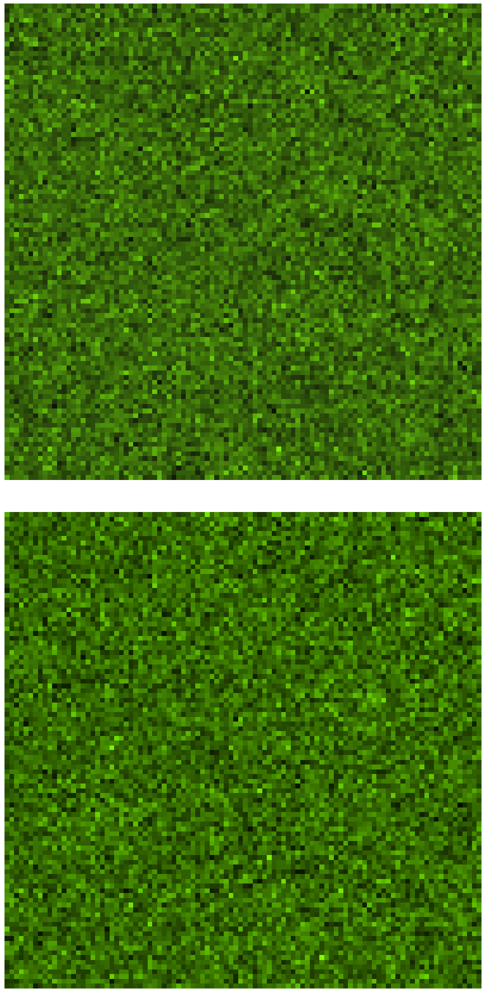
Applications

- **Motion capture**
  - Record motion of people to control cartoon characters in animations
  - Modify the motion record to obtain slightly different behaviours
- **Recognition from motion**
  - Determine the identity of a moving object
  - Assess what the object is doing
- **Surveillance**
  - Detect and track objects in a scene for security
  - Monitor their activities and warn if anything suspicious happens
- **Targeting**
  - Decide which objects to target in scene
  - Make sure the objects get hit

Difficulties in Tracking

- **Loss of information** caused by projection of the 3D world on a 2D image
- **Noise** in images
- Complex object **motion**
- **Non-rigid** or articulated nature of objects
- Partial and full object **occlusions**
- Complex object **shapes**
- Scene **illumination** changes
- **Real-time** processing requirements

Example Tracking Problem



Single moving microscopic particle

- Imaged with signal-to-noise ratio (SNR) of 1.5

Human visual motion perception

- Not so accurate and reproducible in quantification
- Good at integrating spatial and temporal information
- Powerful in making associations and predictions

Computer vision challenges

- Integration of spatial and temporal information
- Modeling and incorporation of prior knowledge
- Probabilistic rather than deterministic approach

Bayesian estimation methods…

Motion Assumptions

- When moving objects do not have unique texture or colour, the characteristics of the motion itself must be used to connect detected points into trajectories
- Assumptions about each moving object:
  - Location changes smoothly over time
  - Velocity (speed and direction) changes smoothly over time
  - Can be at only one location in space at any given time
  - Not in same location as another object at the same time

Topics

- **Bayesian inference**

  Using *probabilistic models* to perform tracking

- **Kalman filtering**

  Using *linear model assumptions* for tracking

- **Particle filtering**

  Using *nonlinear models* for tracking

- **Trajectory analysis**

  Using measures to *quantify motion*

Bayesian Inference

Problem Definition

- A moving object has a **state** which evolves over time

Random variable: X_i

Specific value: x_i

can contain any quantities of interest (position, velocity, acceleration, shape, intensity, colour, …)

- The state is **measured** at each time point

Random variable: Y_i

Specific value: y_i

in computer vision the measurements are typically features computed from the images

- Measurements are combined to estimate the state

Three Main Steps

- **Prediction**: use the measurements $(y_0, y_1, ..., y_{i-1})$ up to time $i-1$ to predict the state at time i

$$P(X_i \mid Y_0 = y_0, Y_1 = y_1, ..., Y_{i-1} = y_{i-1})$$

- **Association**: select the measurements at time i that are related to the object state

- **Correction**: use the incoming measurement y_i to update the state prediction

$$P(X_i \mid Y_0 = y_0, Y_1 = y_1, ..., Y_{i-1} = y_{i-1}, Y_i = y_i)$$

Independence Assumptions

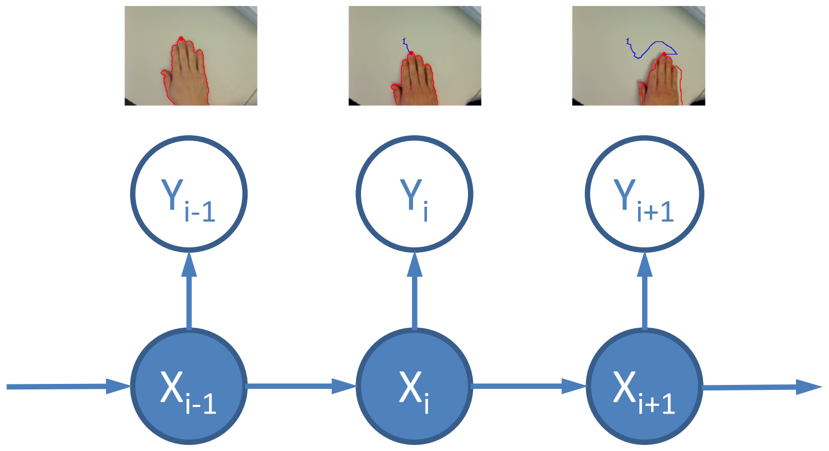
- Current state depends only on the immediate past

$$P(X_i \mid X_0, X_1, \ldots, X_{i-1}) = P(X_i \mid X_{i-1})$$

- Measurements depend only on the current state

$$P(Y_i, Y_j, \ldots, Y_k \mid X_i) = P(Y_i \mid X_i) P(Y_j, \ldots, Y_k \mid X_i)$$

These assumptions imply the tracking problem has the structure of inference on a hidden Markov model



Tracking by Bayesian Inference

- Prediction

$$P(X_i \mid y_0, y_1, \dots, y_{i-1}) = \int \boxed{P(X_i, X_{i-1} \mid y_0, y_1, \dots, y_{i-1})}\, dX_{i-1}$$

$$= \int P(X_i \mid X_{i-1}, \cancel{y_0, y_1, \dots, y_{i-1}}) P(X_{i-1} \mid y_0, y_1, \dots, y_{i-1})\, dX_{i-1}$$

$$= \int \underbrace{P(X_i \mid X_{i-1})}_{\text{dynamics model}} \underbrace{P(X_{i-1} \mid y_0, y_1, \dots, y_{i-1})}_{\text{posterior of previous time}}\, dX_{i-1}$$

$$P(X_i, X_{i-1} \mid y_0, y_1, \dots, y_{i-1}) = \frac{P(X_i, X_{i-1}, y_0, y_1, \dots, y_{i-1})}{P(y_0, y_1, \dots, y_{i-1})}$$

$$= \frac{P(X_i \mid X_{i-1}, y_0, y_1, \dots, y_{i-1}) P(X_{i-1}, y_0, y_1, \dots, y_{i-1})}{P(y_0, y_1, \dots, y_{i-1})}$$

$$= P(X_i \mid X_{i-1}, y_0, y_1, \dots, y_{i-1}) \frac{P(X_{i-1}, y_0, y_1, \dots, y_{i-1})}{P(y_0, y_1, \dots, y_{i-1})}$$

$$= P(X_i \mid X_{i-1}, y_0, y_1, \dots, y_{i-1}) P(X_{i-1} \mid y_0, y_1, \dots, y_{i-1})$$

Tracking by Bayesian Inference

- Correction

$$P(X_i \mid y_0, y_1, \ldots, y_i) = \frac{P(X_i, y_0, y_1, \ldots, y_i)}{P(y_0, y_1, \ldots, y_i)}$$

$$= \frac{P(y_i \mid X_i, \cancel{y_0, y_1, \ldots, y_{i-1}})P(X_i \mid y_0, y_1, \ldots, y_{i-1})P(y_0, y_1, \ldots, y_{i-1})}{P(y_0, y_1, \ldots, y_i)}$$

$$= P(y_i \mid X_i)P(X_i \mid y_0, y_1, \ldots, y_{i-1}) \underbrace{\frac{P(y_0, y_1, \ldots, y_{i-1})}{P(y_0, y_1, \ldots, y_i)}}_{\text{constant}}$$

$$\propto \underbrace{P(y_i \mid X_i)}_{\text{measurement model}} \underbrace{P(X_i \mid y_0, y_1, \ldots, y_{i-1})}_{\text{prediction of current state}}$$

Tracking by Bayesian Inference

In summary, tracking by Bayesian inference is done by iterative prediction and correction:

- Prediction

$$P(X_i \mid Y_{0:i-1}) = \int P(X_i \mid X_{i-1}) \underbrace{P(X_{i-1} \mid Y_{0:i-1})}_{\text{Posterior at time } i-1} \, dX_{i-1}$$

- Correction

$$\underbrace{P(X_i \mid Y_{0:i})}_{\text{Posterior at time } i} \propto P(Y_i \mid X_i) P(X_i \mid Y_{0:i-1})$$

$$\boxed{Y_{0:k} = (Y_0 = y_0, Y_1 = y_1, \ldots, Y_k = y_k)}$$

Tracking by Bayesian Inference

To make tracking by Bayesian inference work in practice you need to design two models:

- Dynamics model $P(X_i \mid X_{i-1})$
- Measurement model $P(Y_i \mid X_i)$

The specific design choices are application dependent

Tracking by Bayesian Inference

Final estimates are computed from the posterior:

- Example 1: expected a posteriori (EAP)

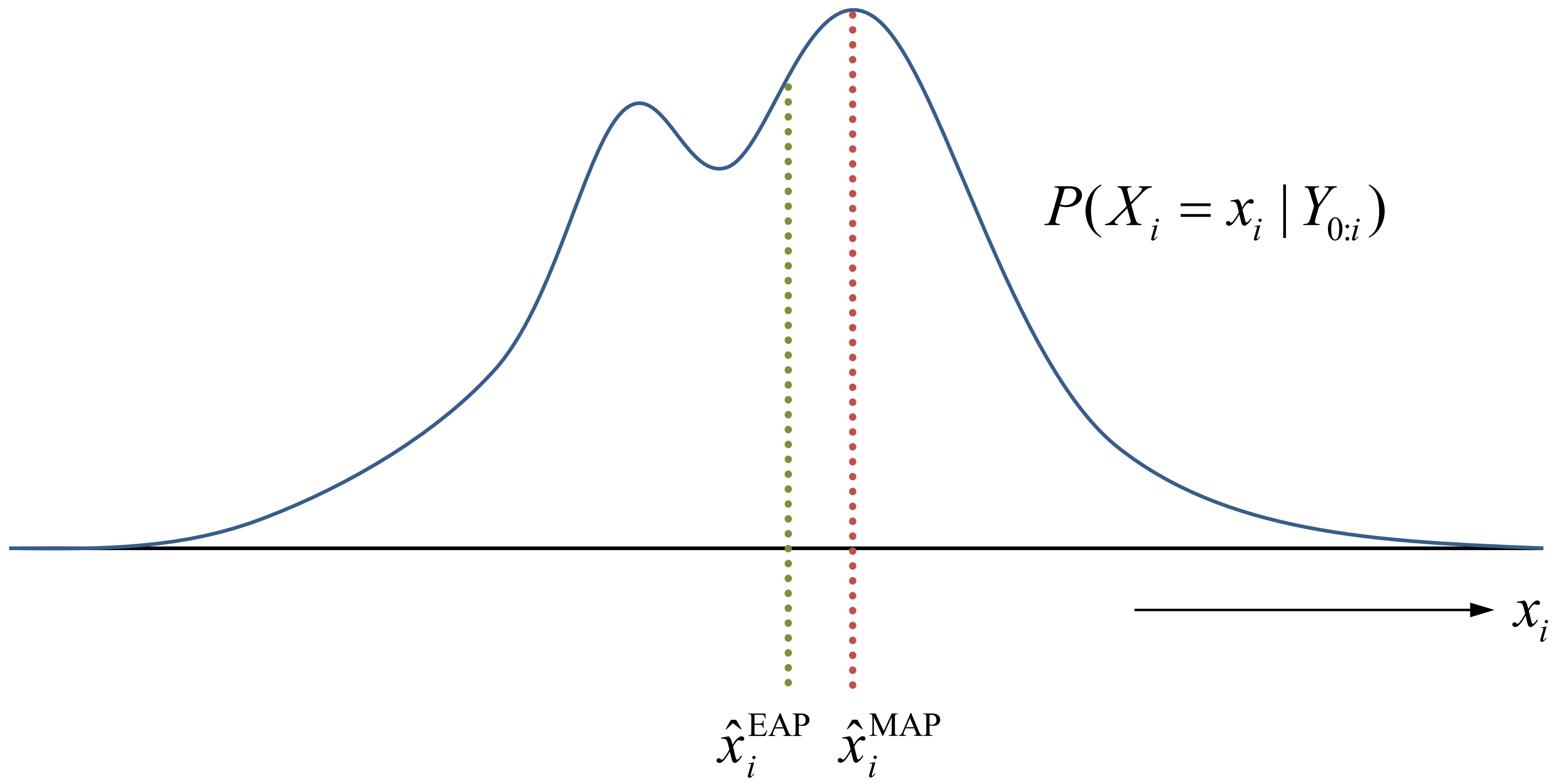
$$\hat{x}_i = \int x_i P(X_i = x_i \mid Y_{0:i}) dx_i$$

- Example 2: maximum a posteriori (MAP)

$$\hat{x}_i = \arg\max_{x_i} P(X_i = x_i \mid Y_{0:i})$$

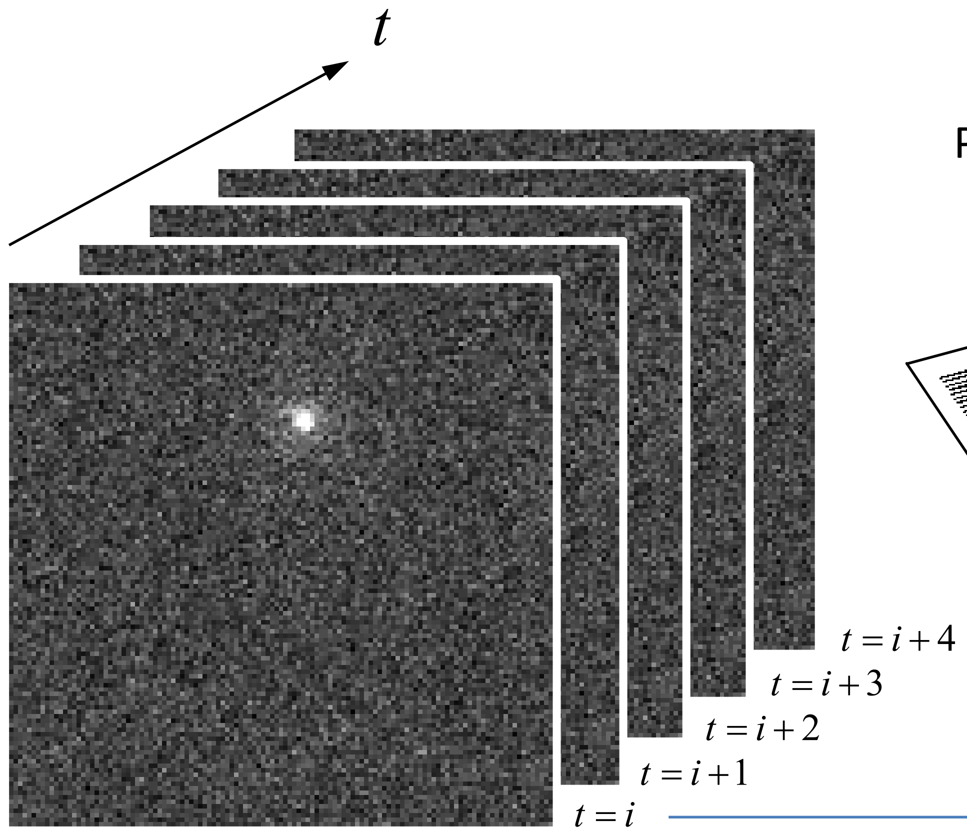
These are the most popular ones but others are possible

Tracking by Bayesian Inference

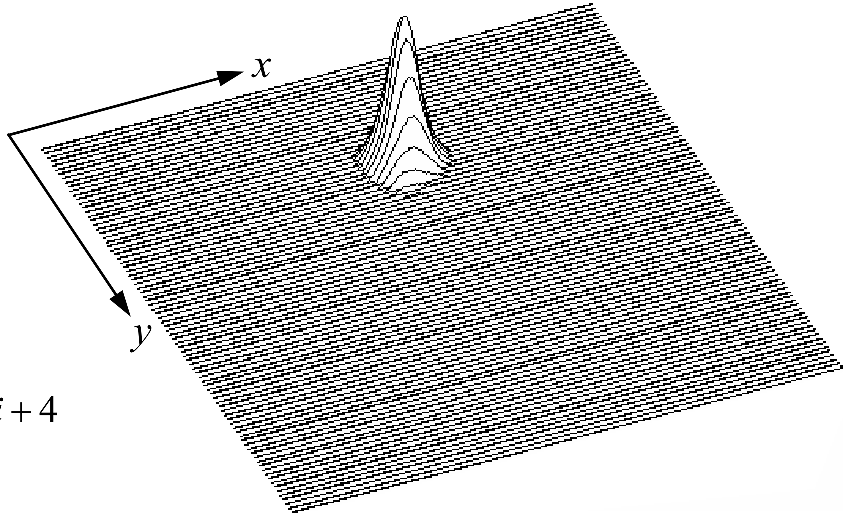


Bayesian Tracking Example

Estimating the coordinates of a moving particle:



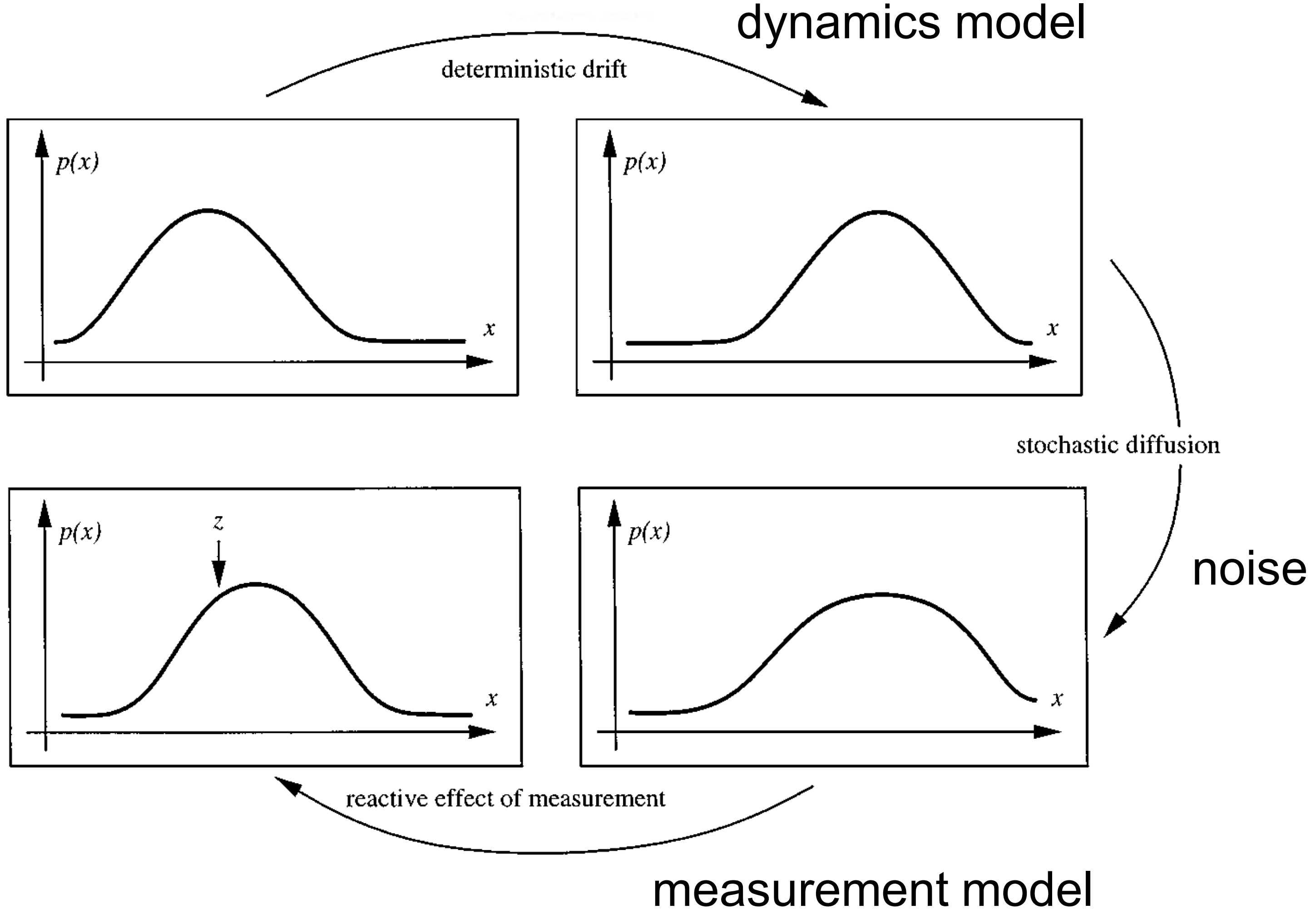
Posterior computed from the image



$$P(X_i \mid Y_{0:i})$$

Kalman Filtering

Probability Density Propagation



Linear / Gaussian Assumption

If we assume the dynamics (state transition) model and the measurement model to be linear, and the noise to be additive Gaussian, then all the probability densities will be Gaussians:

$$x \sim N(\mu, \Sigma)$$

- The state is advanced by multiplying with some known matrix and then adding a zero-mean normal random variable:

$$x_i = A x_{i-1} + q_{i-1}$$

- The measurement is obtained by multiplying the state by some matrix and then adding a zero-mean normal random variable:

$$y_i = H x_i + r_i$$

$$x_i \sim N(Ax_{i-1}, Q)$$

$$y_i \sim N(Hx_i, R)$$

Kalman Filtering

Prediction

1. Predict state

$$x_i^- = Ax_{i-1}$$

2. Predict covariance

$$P_i^- = AP_{i-1}A^T + Q$$

Correction

1. Compute Kalman gain

$$K_i = P_i^- H^T (H P_i^- H^T + R)^{-1}$$

2. Correct state with measurement

$$x_i = x_i^- + K_i (y_i - H x_i^-)$$

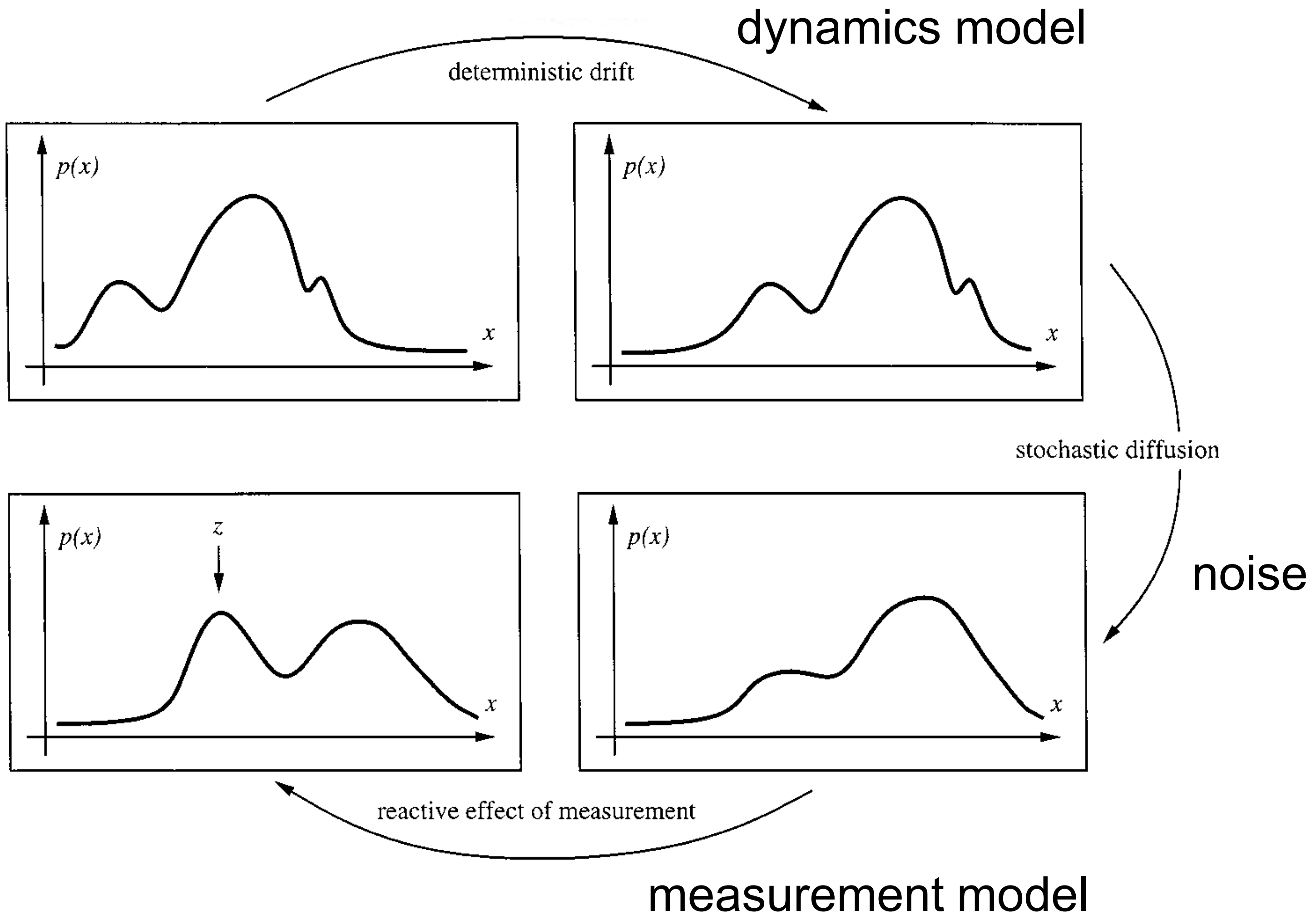
3. Correct covariance

$$P_i = (I - K_i H) P_i^-$$

$$i \to i+1$$

Particle Filtering

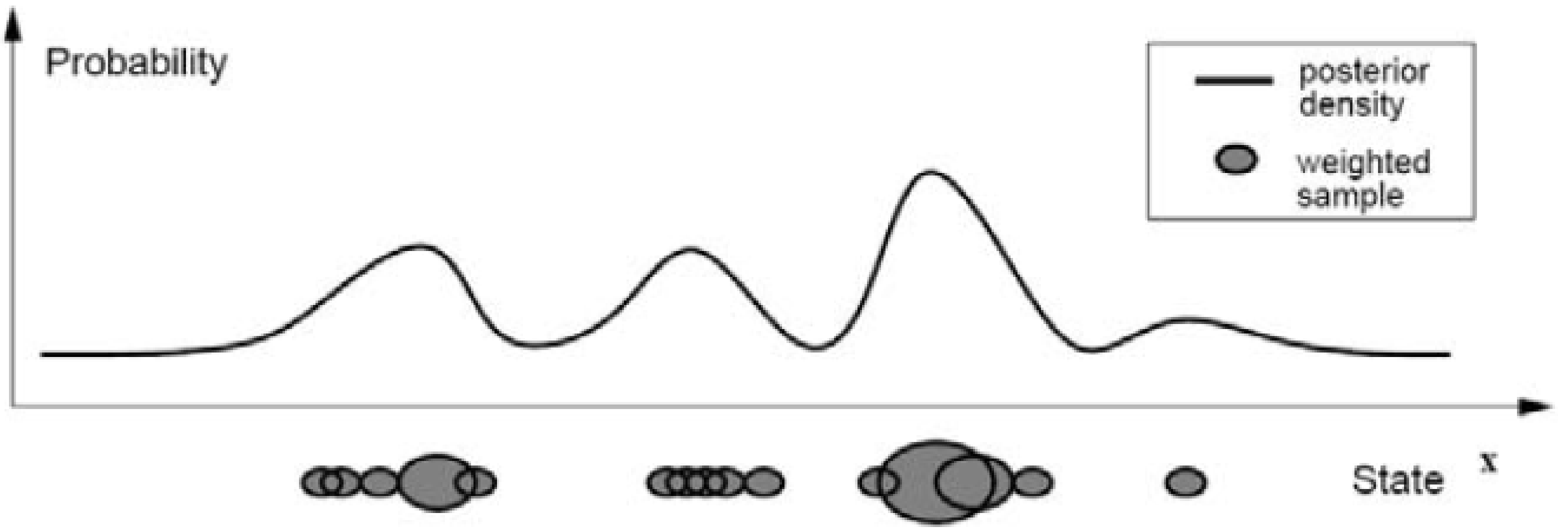
Probability Density Propagation



Non-Linear / Non-Gaussian Case

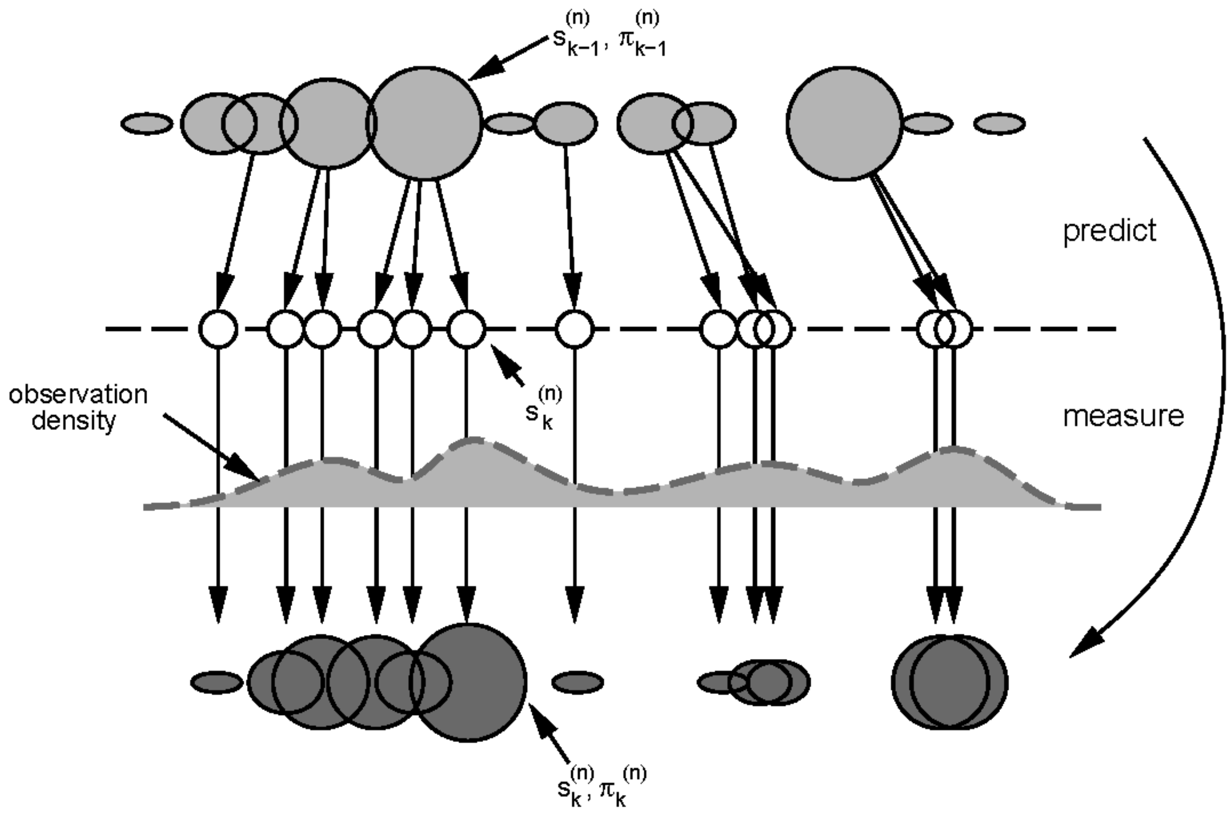
- Represent the conditional state density by a set of samples (particles) with corresponding weights (importance)

$$P(X_i \mid Y_{0:i}) \rightarrow \{s_i^{(n)}, \pi_i^{(n)}\}_{n=1}^{N}$$

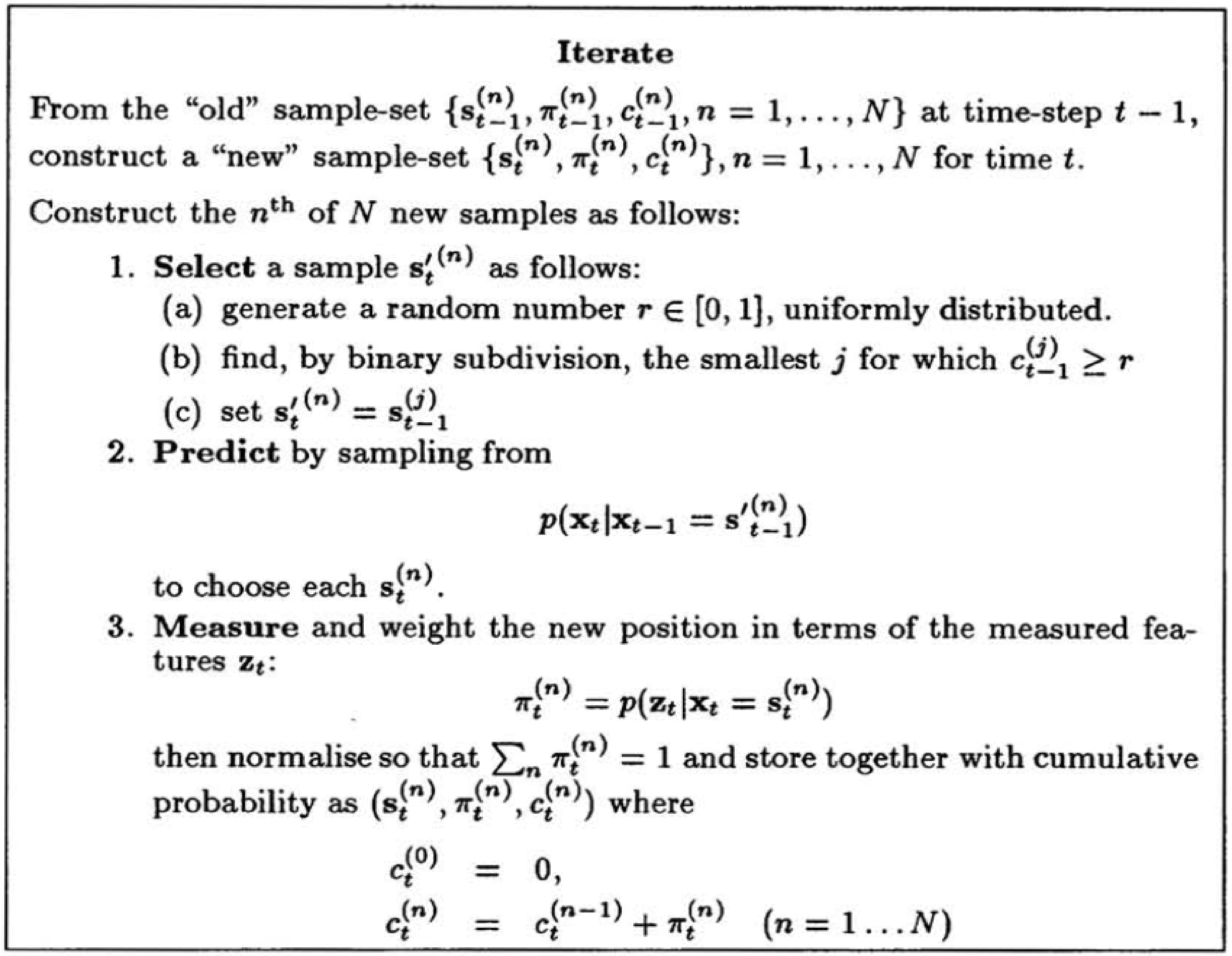


Particle Filtering

- Propagate each sample using the dynamics model and obtain its new weight using the measurement model



Particle Filtering Algorithm



Iterate

From the "old" sample-set $\{\mathbf{s}_{t-1}^{(n)}, \pi_{t-1}^{(n)}, c_{t-1}^{(n)}, n = 1, \ldots, N\}$ at time-step $t-1$, construct a "new" sample-set $\{\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)}\}, n = 1, \ldots, N$ for time t.

Construct the n^{th} of N new samples as follows:

1. **Select** a sample $\mathbf{s}_t'^{(n)}$ as follows:
   (a) generate a random number $r \in [0,1]$, uniformly distributed.
   (b) find, by binary subdivision, the smallest j for which $c_{t-1}^{(j)} \geq r$
   (c) set $\mathbf{s}_t'^{(n)} = \mathbf{s}_{t-1}^{(j)}$
2. **Predict** by sampling from

$$p(\mathbf{x}_t | \mathbf{x}_{t-1} = \mathbf{s}'^{(n)}_{t-1})$$

   to choose each $\mathbf{s}_t^{(n)}$.
3. **Measure** and weight the new position in terms of the measured features $\mathbf{z}_t$:

$$\pi_t^{(n)} = p(\mathbf{z}_t | \mathbf{x}_t = \mathbf{s}_t^{(n)})$$

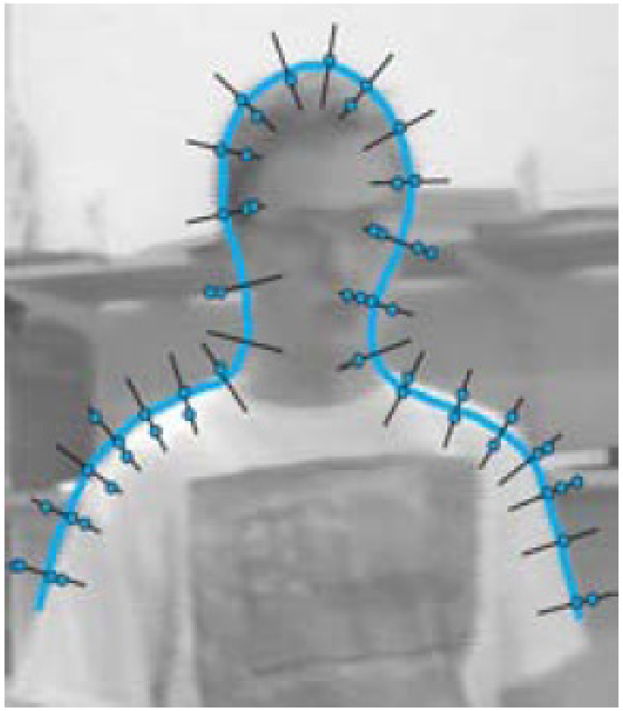
   then normalise so that $\sum_n \pi_t^{(n)} = 1$ and store together with cumulative probability as $(\mathbf{s}_t^{(n)}, \pi_t^{(n)}, c_t^{(n)})$ where

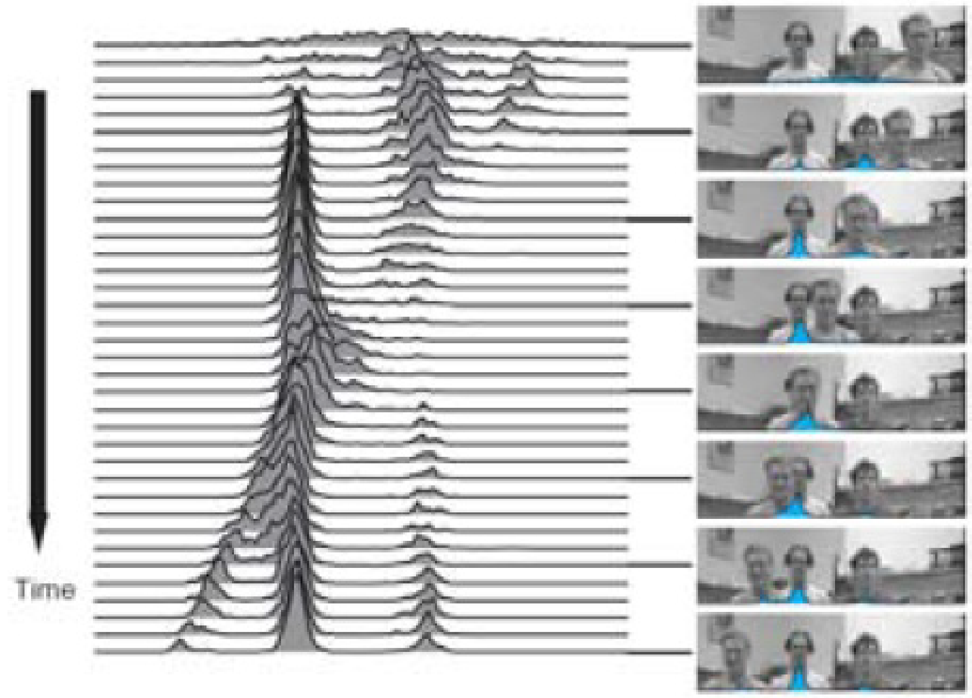
$$\begin{aligned} c_t^{(0)} &= 0, \\ c_t^{(n)} &= c_t^{(n-1)} + \pi_t^{(n)} \quad (n = 1 \ldots N) \end{aligned}$$

NIPS 1996

Example Application

Tracking of active contour representations of objects

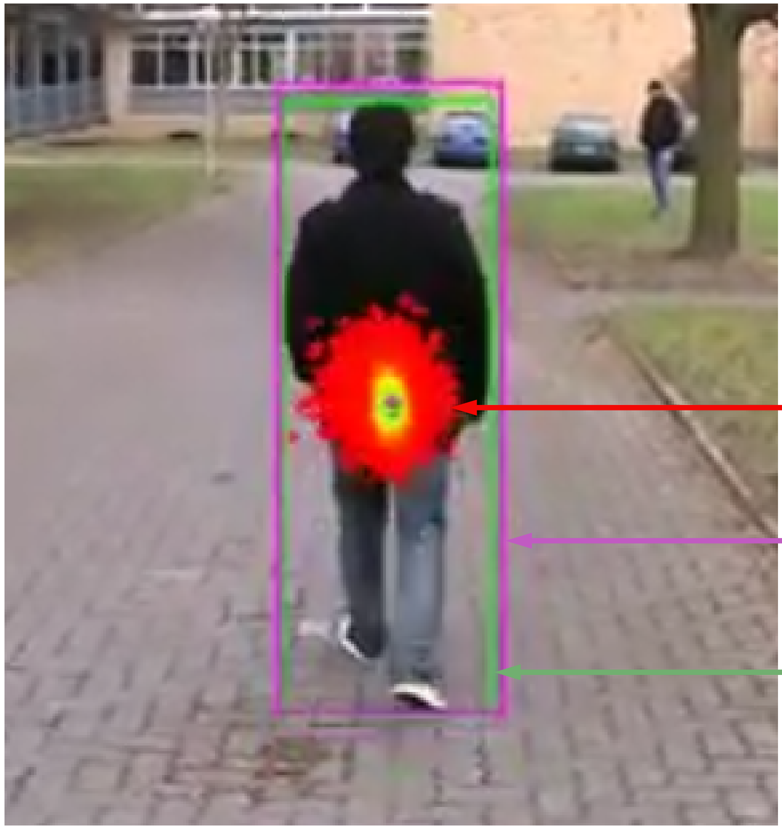




Particle filtering is also known variously as sequential Monte Carlo (SMC) filtering, bootstrap filtering, the condensation algorithm…

Example Application

Tracking of object location in the presence of clutter



Walking pedestrian represented by a state vector consisting of a center position and a bounding box:

$$s_i = (x, y, w, h)_i$$

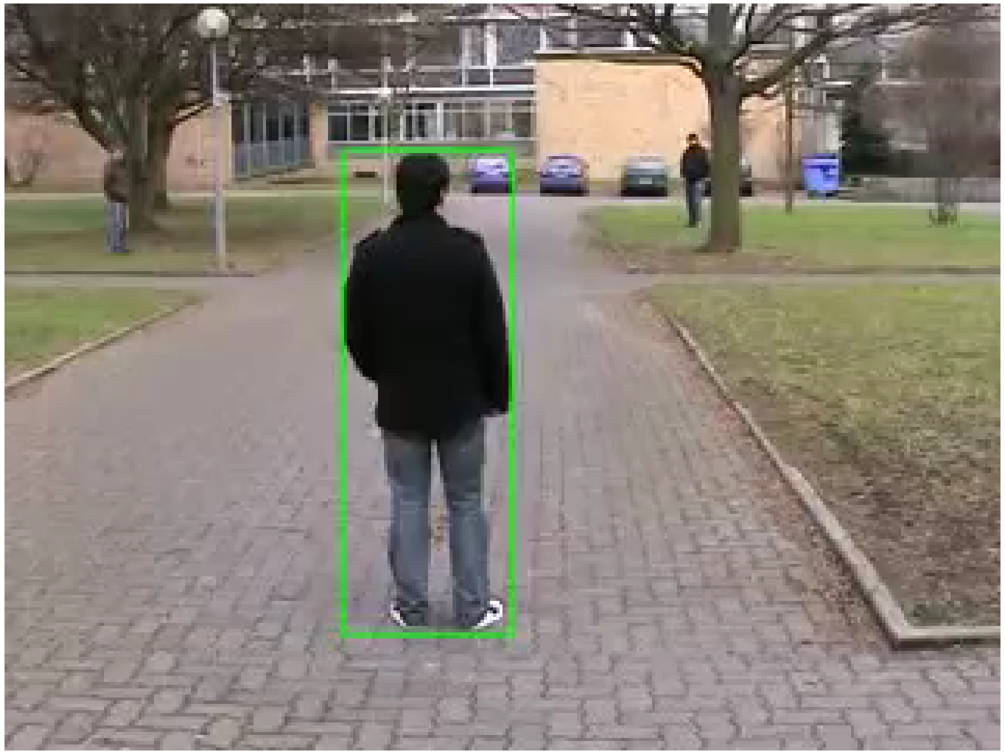
$s_i^{(n)}$ (samples)

$\hat{s}_i$ (estimated)

$\tilde{s}_i$ (truth/annotated)

Example Application

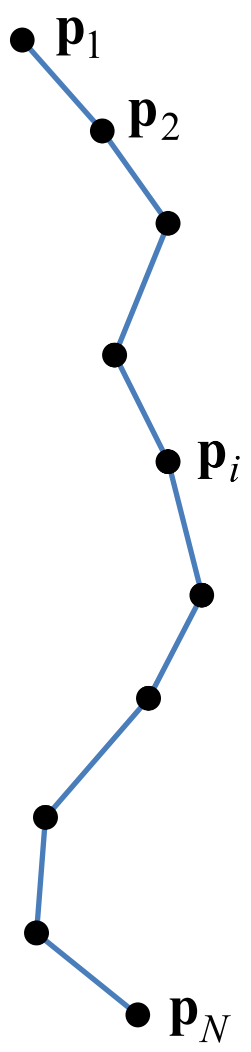
Tracking of object location in the presence of clutter



https://www.youtube.com/watch?v=j-duyzShJ_o

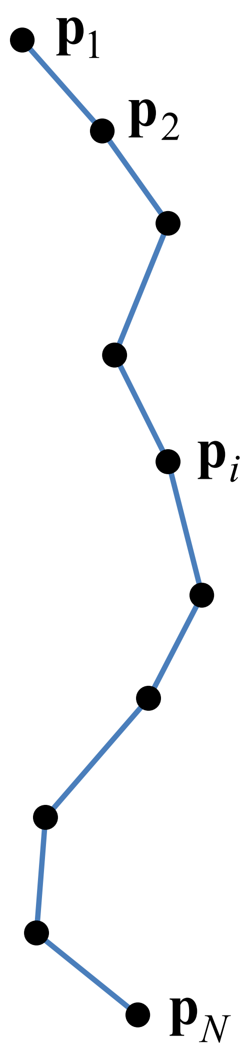
Trajectory Analysis

Motion Features



Measure	Definition
Total distance traveled	$d_{\text{tot}} = \sum_{i=1}^{N-1} d(\mathbf{p}_i, \mathbf{p}_{i+1})$
Net distance traveled	$d_{\text{net}} = d(\mathbf{p}_1, \mathbf{p}_N)$
Maximum distance traveled	$d_{\text{max}} = \max_i d(\mathbf{p}_1, \mathbf{p}_i)$
Total trajectory time	$t_{\text{tot}} = (N-1)\Delta t$
Confinement ratio	$r_{\text{con}} = d_{\text{net}}/d_{\text{tot}}$
Instantaneous angle	$\alpha_i = \arctan(y_{i+1}-y_i)/(x_{i+1}-x_i)$
Directional change	$\gamma_i = \alpha_i - \alpha_{i-1}$
Instantaneous speed	$v_i = d(\mathbf{p}_i, \mathbf{p}_{i+1})/\Delta t$
Mean curvilinear speed	$\bar{v} = \frac{1}{N-1}\sum_{i=1}^{N-1} v_i$
Mean straight-line speed	$v_{\text{lin}} = d_{\text{net}}/t_{\text{tot}}$
Linearity of forward progression	$r_{\text{lin}} = v_{\text{lin}}/\bar{v}$
Mean squared displacement	$\text{MSD}(n) = \frac{1}{N-n}\sum_{i=1}^{N-n} d^2(\mathbf{p}_i, \mathbf{p}_{i+n})$

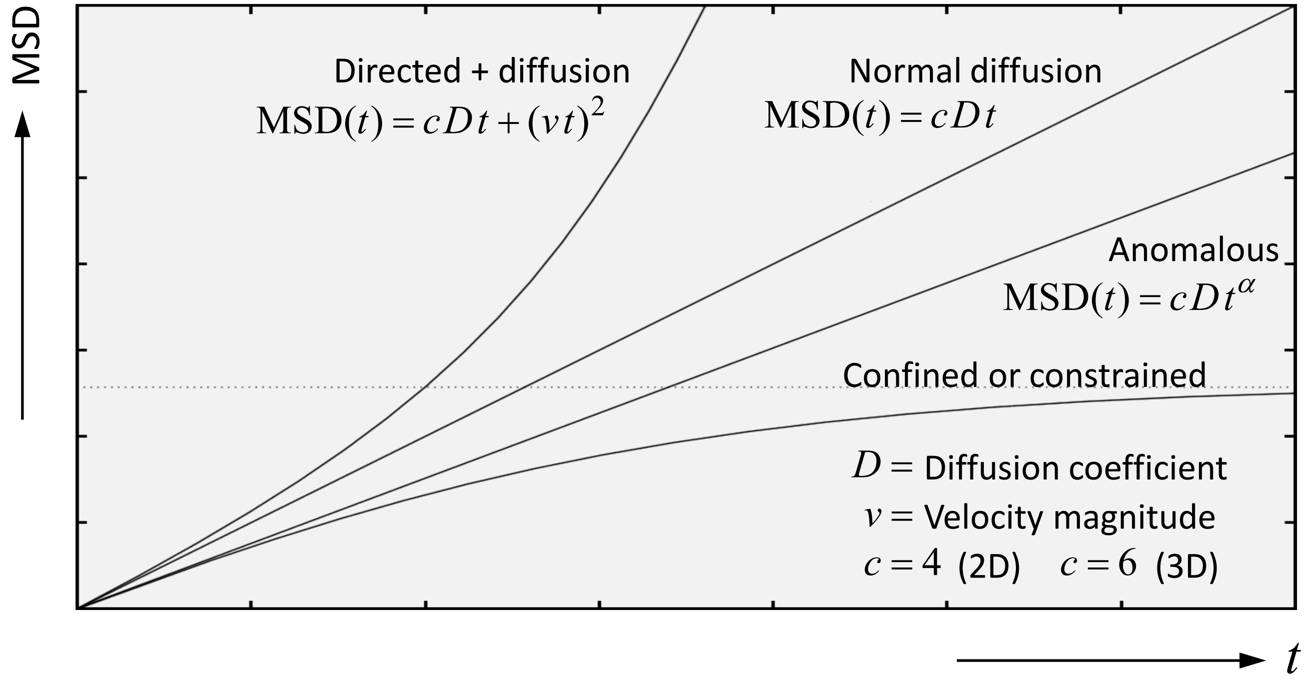
https://doi.org/10.1016/B978-0-12-391857-4.00009-4



MSD Analysis

Distance between track points: $d\left(\mathbf{p}_i, \mathbf{p}_j\right) = \| \mathbf{p}_j - \mathbf{p}_i \|_2$

MSD for a given time lag t: $\mathrm{MSD}(t) = \frac{1}{N-t} \sum_{i=1}^{N-t} d^2\left(\mathbf{p}_i, \mathbf{p}_{i+t}\right)$



References and Acknowledgements

- Chapters 5 and 8 of Szeliski 2010
- Chapter 18 of Forsyth and Ponce 2011
- Chapter 9 of Shapiro and Stockman 2001
- Paper by M. Isard and A. Blake 1998
  CONDENSATION: Conditional density propagation for visual tracking
  Available online via the UNSW Library
- Some images drawn from the above references