Introduction to recursive Bayesian filtering

Michael Rubinstein

IDC

Problem overview

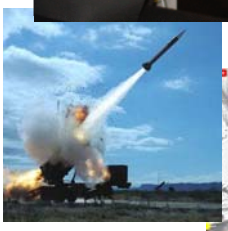
- Input
  - (Noisy) Sensor measurements
- Goal
  - Estimate most probable measurement at time k using measurements up to time k'
    - k'<k: **prediction**
    - k'>k: **smoothing**
    - k'=k: **filtering**
- Many problems require estimation of the state of systems that change over time using noisy measurements on the system

Applications

- Ballistics
- Robotics
  - Robot localization
- Tracking hands/cars/…
- Econometrics
  - Stock prediction
- Navigation

- Many more…





Challenges (why is it a hard problem?)

- Measurements
  - Noise
  - Errors
- Detection specific
  - Full/partial occlusions
  - Deformable objects
  - Entering/leaving the scene
  - Lighting variations
- Efficiency
- Multiple models and switching dynamics
- Multiple targets,
- …

Talk overview

- Background
  - Model setup
    - Markovian-stochastic processes
    - The state-space model
    - Dynamic systems
  - The Bayesian approach
  - Recursive filters
  - Restrictive cases + pros and cons
    - The Kalman filter
    - The Grid-based filter
- Particle filters
  - Monte Carlo integration
  - Importance sampling
- Multiple target tracking – BraMBLe ICCV 2001 (?)

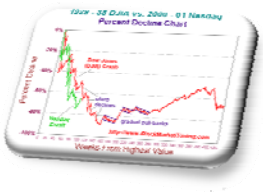
Stochastic Processes

- Deterministic process
  - Only one possible ‘reality’
- Random process
  - Several possible evolutions (starting point might be known)
  - Characterized by probability distributions
- Time series modeling
  - Sequence of random states/variables
  - Measurements available at discrete times

State space

- **The state vector** contains all available information to describe the investigated system
  - usually multidimensional: $X(k) \in R^{N_x}$
- **The measurement vector** represents observations related to the state vector $Z(k) \in R^{N_z}$
  - Generally (but not necessarily) of lower dimension than the state vector

State space



- Tracking:

$N_x = 3$

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix}$$

$N_x = 4$

$$\begin{bmatrix} x \\ v_x \\ y \\ v_y \end{bmatrix}$$

Econometrics:

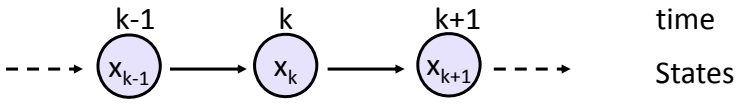
- Monetary flow
- Interest rates
- Inflation
- …

(First-order) Markov process

- The Markov property – the likelihood of a future state depends on present state only

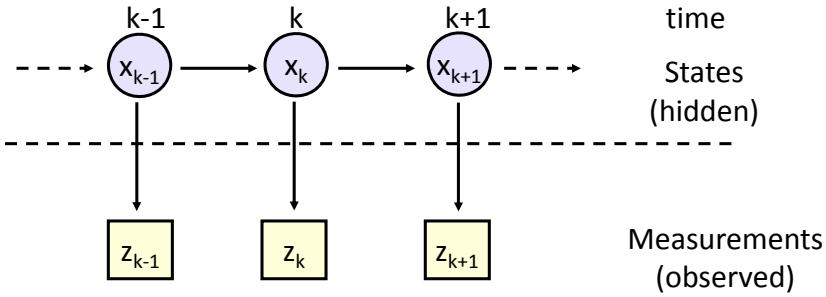
$$\Pr[X(k+h) = y \mid X(s) = x(s), \forall s \le k] = \Pr[X(k+h) = y \mid X(k) = x(k)], \forall h > 0$$

- Markov chain – A stochastic process with Markov property

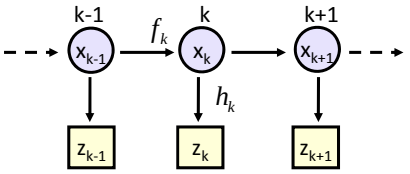


Hidden Markov Model (HMM)

- the state is not directly visible, but output dependent on the state is visible



Dynamic System



Stochastic diffusion

State equation: $x_k = f_k(x_{k-1}, v_k)$

- x_k state vector at time instant k
- f_k state transition function, $f_k : R^{N_x} \times R^{N_v} \to R^{N_x}$
- v_k i.i.d process noise

Observation equation: $z_k = h_k(x_k, w_k)$

- z_k observations at time instant k
- h_k observation function, $h_k : R^{N_x} \times R^{N_w} \to R^{N_z}$
- w_k i.i.d measurement noise

A simple dynamic system

- $X = [x, y, v_x, v_y]$ (4-dimensional state space)
- Constant velocity motion:

$$f(X, v) = [x + \Delta t \cdot v_x, y + \Delta t \cdot v_y, v_x, v_y] + v$$

$$v \sim N(0, Q) \quad Q = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & q^2 & 0 \\ 0 & 0 & 0 & q^2 \end{pmatrix}$$

- Only position is observed:

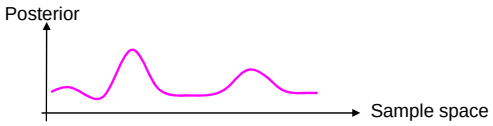
$$z = h(X, w) = [x, y] + w$$

$$w \sim N(0, R) \quad R = \begin{pmatrix} r^2 & 0 \\ 0 & r^2 \end{pmatrix}$$

The Bayesian approach


Thomas Bayes

- Construct the posterior probability density function $p(x_k \mid z_{1:k})$ of the state based on all available information



- By knowing the posterior many kinds of estimates for x_k can be derived
  - mean (expectation), mode, median, …
  - Can also give estimation of the accuracy (e.g. covariance)

Recursive filters

- For many problems, estimate is required each time a new measurement arrives
- **Batch** processing
  - Requires *all* available data
- **Sequential** processing
  - New data is processed upon arrival
  - Need not store the complete dataset
  - Need not reprocess all data for each new measurement
  - Assume no out-of-sequence measurements (solutions for this exist as well…)

Recursive Bayes filters

- Given:
  - System models in probabilistic forms

$$x_k = f_k(x_{k-1}, v_k) \leftrightarrow p(x_k \mid x_{k-1})$$
$$z_k = h_k(x_k, w_k) \leftrightarrow p(z_k \mid x_k)$$

Markovian process

Measurements are conditionally independent given the state

(known statistics of v_k, w_k)

  - Initial state $p(x_0 \mid z_0) = p(x_0)$ also known as the **prior**
  - Measurements $z_1, \ldots, z_k$

Recursive Bayes filters

- **Prediction** step (a-priori)

$$p(x_{k-1} \mid z_{1:k-1}) \rightarrow p(x_k \mid z_{1:k-1})$$

  - Uses the system model to predict forward
  - Deforms/translates/spreads state pdf due to random noise

- **Update** step (a-posteriori)

$$p(x_k \mid z_{1:k-1}) \rightarrow p(x_k \mid z_{1:k})$$

  - Update the prediction in light of new data
  - Tightens the state pdf

General prediction-update framework

- Assume $p(x_{k-1} \mid z_{1:k-1})$ is given at time k-1
- Prediction:

System model Previous posterior

$$p(x_k \mid z_{1:k-1}) = \int p(x_k \mid x_{k-1}) p(x_{k-1} \mid z_{1:k-1}) dx_{k-1} \quad (1)$$

- Using Chapman-Kolmogorov identity + Markov property

General prediction-update framework

- Update step $p(x_k \mid z_{1:k}) = p(x_k \mid z_k, z_{1:k-1})$

$$p(A \mid B, C) = \frac{p(B \mid A, C) p(A \mid C)}{p(B \mid C)}$$

$$= \frac{p(z_k \mid x_k, z_{1:k-1}) p(x_k \mid z_{1:k-1})}{p(z_k \mid z_{1:k-1})}$$

$$\frac{\text{likelihood} \times \text{prior}}{\text{evidence}}$$

Measurement model Current prior

$$= \frac{p(z_k \mid x_k) p(x_k \mid z_{1:k-1})}{p(z_k \mid z_{1:k-1})} \quad (2)$$

Normalization constant

Where $p(z_k \mid z_{1:k-1}) = \int p(z_k \mid x_k) p(x_k \mid z_{1:k-1}) dx_k$

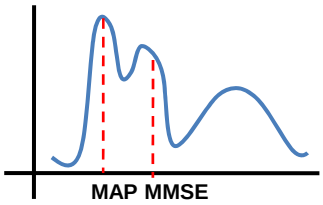
Generating estimates

- Knowledge of $p(x_k \mid z_{1:k})$ enables to compute optimal estimate with respect to any criterion. e.g.
  - Minimum mean-square error (MMSE)

$$\hat{x}_{k|k}^{MMSE} \equiv E[x_k \mid z_{1:k}] = \int x_k \, p(x_k \mid z_{1:k}) dx_k$$

  - Maximum a-posteriori

$$\hat{x}_{k|k}^{MAP} \equiv \arg\max_{x_k} p(x_k \mid z_k)$$



General prediction-update framework

➔So (1) and (2) give optimal solution for the recursive estimation problem!

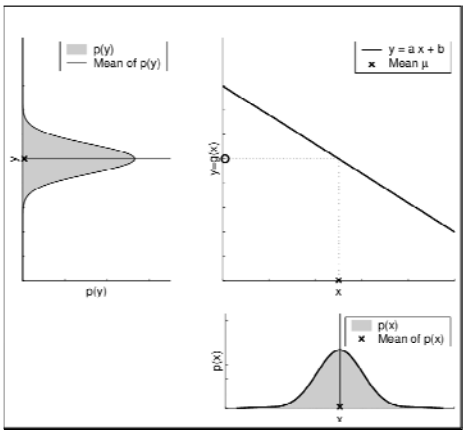
- Unfortunately no… only conceptual solution
  - integrals are intractable…
  - Can only implement the pdf to finite representation!
- However, optimal solution *does* exist for several restrictive cases

Restrictive case #1

- Posterior at each time step is Gaussian
  - Completely described by mean and covariance
- If $p(x_{k-1} \mid z_{1:k-1})$ is Gaussian it can be shown that $p(x_k \mid z_{1:k})$ is also Gaussian provided that:
  - v_k, w_k are Gaussian
  - f_k, h_k are linear

Restrictive case #1

- Why Linear?

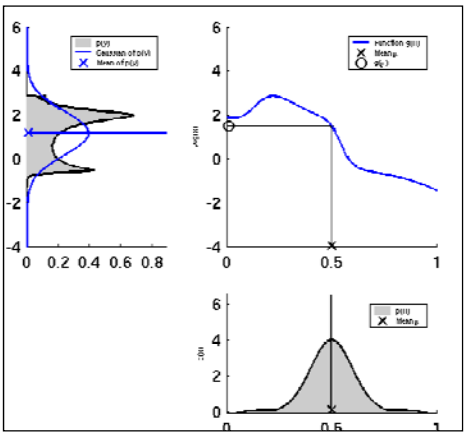


Yacov Hel-Or

$$y = Ax + B \Rightarrow p(y) \sim N\left(A\mu + B, A\Sigma A^T\right)$$

Restrictive case #1

- Why Linear?



Yacov Hel-Or

$$y = g(x) \Rightarrow p(y) \sim N(\ )$$

Restrictive case #1

- Linear system with additive noise

$$\begin{aligned} x_k &= F_k(x_{k-1}, w_k) \\ z_k &= H_k(x_k, w_k) \\ v_k &\sim N(0, Q_k) \\ w_k &\sim N(0, R_k) \end{aligned}$$

- Simple example again

$$f(X, v) = [x + \Delta t \cdot v_x, y + \Delta t \cdot v_y, v_x, v_y] + v$$

$$\begin{pmatrix} x_k \\ y_k \\ v_{x,k} \\ v_{y,k} \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{F} \begin{pmatrix} x_k \\ y_k \\ v_{x,k-1} \\ v_{y,k-1} \end{pmatrix} + N(0, Q_k)$$

$$z = h(X, w) = [x, y] + w$$

$$\begin{pmatrix} x_{obs} \\ y_{obs} \end{pmatrix} = \underbrace{\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}}_{H} \begin{pmatrix} x_k \\ y_k \\ v_{x,k} \\ v_{y,k} \end{pmatrix} + N(0, R_k)$$

The Kalman filter

Rudolf E. Kalman

$$p(x_{k-1} \mid z_{1:k-1}) = N(x_{k-1}; \hat{x}_{k-1|k-1}, P_{k-1|k-1})$$
$$p(x_k \mid z_{1:k-1}) = N(x_k; \hat{x}_{k|k-1}, P_{k|k-1})$$
$$p(x_k \mid z_{1:k}) = N(x_k; \hat{x}_{k|k}, P_{k|k})$$

$$N(x; \mu, \Sigma) = \mid 2\pi\Sigma \mid^{-1/2} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu) \right)$$

- Substituting into (1) and (2) yields the predict and update equations

The Kalman filter

Predict:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1}$$
$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k$$

Update:

$$S_k = H_k P_{k|k-1} H_k^T + R_k$$
$$K_k = P_{k|k-1} H_k^T S_k^{-1}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\left(z_k - H_k \hat{x}_{k|k-1}\right)$$
$$P_{k|k} = \left[I - K_k H_k\right] P_{k|k-1}$$

Intuition via 1D example

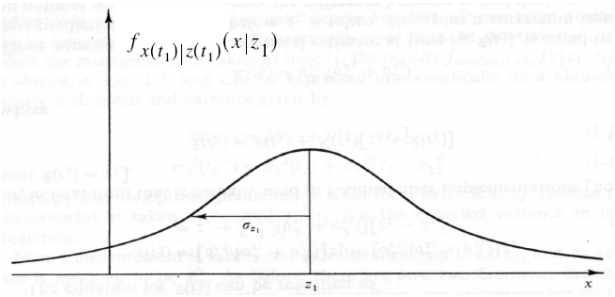
- Lost at sea
  - Night
  - No idea of location
  - For simplicity – let's assume 1D

* Example and plots by Maybeck, *"Stochastic models, estimation and control, volume 1"*

Example – cont'd

- Time t1: Star Sighting
  - Denote x(t1)=z1
- Uncertainty (inaccuracies, human error, etc)
  - Denote $\sigma 1$ (normal)
- Can establish the conditional probability of x(t1) given measurement z1

Example – cont'd

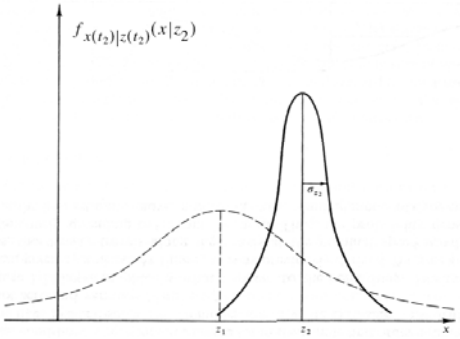


- Probability for any location, based on measurement
- For Gaussian density – 68.3% within $\pm\sigma 1$
- Best estimate of position: Mean/Mode/Median

$$\hat{x}(t_1) = z_1 \qquad \sigma_x^2(t_1) = \sigma_{z_1}^2$$

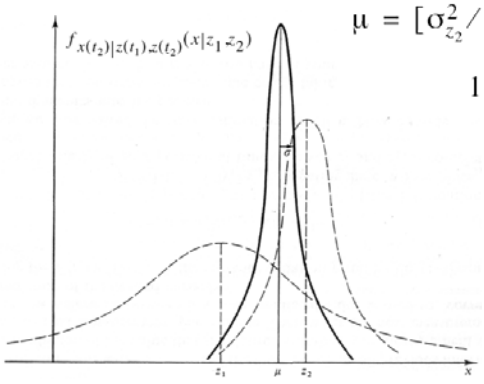
Example – cont'd

- Time $t2 \cong t1$: friend (more trained)
  - $x(t2)=z2$, $\sigma(t2)=\sigma 2$
  - Since she has higher skill: $\sigma 2<\sigma 1$



Example – cont'd

- f(x(t2)|z1,z2) also Gaussian



$$\mu = [\sigma_{z_2}^2/(\sigma_{z_1}^2 + \sigma_{z_2}^2)]z_1 + [\sigma_{z_1}^2/(\sigma_{z_1}^2 + \sigma_{z_2}^2)]z_2$$

$$1/\sigma^2 = (1/\sigma_{z_1}^2) + (1/\sigma_{z_2}^2)$$

Example – cont'd

$$\mu = [\sigma_{z_2}^2/(\sigma_{z_1}^2 + \sigma_{z_2}^2)]z_1 + [\sigma_{z_1}^2/(\sigma_{z_1}^2 + \sigma_{z_2}^2)]z_2$$

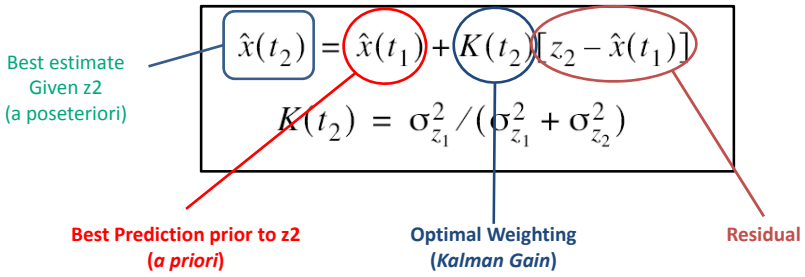
$$1/\sigma^2 = (1/\sigma_{z_1}^2) + (1/\sigma_{z_2}^2)$$

- σ less than both $\sigma 1$ and $\sigma 2$
- $\sigma 1 = \sigma 2$: average
- $\sigma 1 > \sigma 2$: more weight to z2
- Rewrite:

$$\hat{x}(t_2) = [\sigma_{z_2}^2/(\sigma_{z_1}^2 + \sigma_{z_2}^2)]z_1 + [\sigma_{z_1}^2/(\sigma_{z_1}^2 + \sigma_{z_2}^2)]z_2$$
$$= z_1 + [\sigma_{z_1}^2/(\sigma_{z_1}^2 + \sigma_{z_2}^2)][z_2 - z_1]$$

Example – cont'd

- The Kalman update rule:



The Kalman filter

Predict:

$$\hat{x}_{k|k-1} = F_k \hat{x}_{k-1|k-1}$$
$$P_{k|k-1} = F_k P_{k-1|k-1} F_k$$

Update:

$$S_k = H_k P_{k|k-1} H_k^T + R$$
$$K_k = P_{k|k-1} H_k^T S_k^{-1}$$
$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\left(z_k - H_k \hat{x}_{k|k-1}\right)$$
$$P_{k|k} = \left[I - K_k H_k\right] P_{k|k-1}$$

```
% --- Time update ("predict")

% Project the state forward
Xk1 = Ak*s.Xk;

% Project the prediction covariance forward
P1 = Ak*s.P*Ak' + Qk;

% --- Measurement update ("correct")

% Calculate the Kalman gain.
% K large: more weight goes to the measurement.
% K low: more weight goes to the model prediction.
K = P1*s.H'*inv(s.H*P1*s.H' + Rk);

% Update estimate with measurement Zk
s.Xk = Xk1 + K*(Zk-s.H*Xk1);

% Update the error covariance
s.P = P1 - K*s.H*P1;
```

$$K(t_2) = \sigma_{z_1}^2 / (\sigma_{z_1}^2 + \sigma_{z_2}^2)$$
$$\hat{x}(t_2) = \hat{x}(t_1) + K(t_2)[z_2 - \hat{x}(t_1)]$$
$$\frac{\sigma_{z_1}^2 \sigma_{z_2}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2} = \left(1 - \frac{\sigma_{z_1}^2}{\sigma_{z_1}^2 + \sigma_{z_2}^2}\right)\sigma_{z_1}^2$$

Kalman gain

$$
\begin{aligned}
S_k &= H_k P_{k|k-1} H_k^T + R_k \\
K_k &= P_{k|k-1} H_k^T S_k^{-1} \\
\hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k\left(z_k - H_k \hat{x}_{k|k-1}\right) \\
P_{k|k} &= \left[I - K_k H_k\right] P_{k|k-1}
\end{aligned}
$$

- Small measurement error:

$$\lim_{\mathbf{R}_k \to 0} K_k = H_k^{-1} \Rightarrow \lim_{\mathbf{R}_k \to 0} \hat{x}_{k|k} = H_k^{-1} z_k$$

- Small prediction error:

$$\lim_{\mathbf{P}_k \to 0} K_k = 0 \Rightarrow \lim_{\mathbf{P}_k \to 0} \hat{x}_{k|k} = \hat{x}_{k|k-1}$$

The Kalman filter

- Pros
  - Optimal closed-form solution to the tracking problem (under the assumptions)
    - No algorithm can do better in a linear-Gaussian environment!
  - All 'logical' estimations collapse to a unique solution
  - Simple to implement
  - Fast to execute
- Cons
  - If either the system or measurement model is non-linear → the posterior will be non-Gaussian

Restrictive case #2

- The state space (domain) is discrete and finite
- Assume the state space at time k-1 consists of states $x_{k-1}^i, i = 1..N_s$
- Let $\Pr(x_{k-1} = x_{k-1}^i \mid z_{1:k-1}) = w_{k-1|k-1}^i$ be the conditional probability of the state at time k-1, given measurements up to k-1

The Grid-based filter

- The posterior pdf at k-1 can be expressed as sum of delta functions

$$p(x_{k-1} \mid z_{1:k-1}) = \sum_{i=1}^{N_s} w_{k-1|k-1}^i \delta(x_{k-1} - x_{k-1}^i)$$

- Again, substitution into (1) and (2) yields the predict and update equations

The Grid-based filter

- Prediction

$$p(x_k \mid z_{1:k-1}) = \int p(x_k \mid x_{k-1})\, p(x_{k-1} \mid z_{1:k-1}) dx_{k-1} \qquad (1)$$

$$p(x_k \mid z_{1:k-1}) = \sum_{i=1}^{N_s} \sum_{j=1}^{N_s} p(x_k^i \mid x_{k-1}^j) w_{k-1|k-1}^j \delta(x_{k-1} - x_{k-1}^i)$$

$$= \sum_{i=1}^{N_s} w_{k|k-1}^i \delta(x_{k-1} - x_{k-1}^i)$$

$$w_{k|k-1}^i = \sum_{j=1}^{N_s} w_{k-1|k-1}^j p(x_k^i \mid x_{k-1}^j)$$

- New prior is also weighted sum of delta functions
- New prior weights are reweighting of old posterior weights using state transition probabilities

The Grid-based filter

- Update

$$p(x_k \mid z_{1:k}) = \frac{p(z_k \mid x_k)\, p(x_k \mid z_{1:k-1})}{p(z_k \mid z_{1:k-1})} \qquad (2)$$

$$p(x_k \mid z_{1:k}) = \sum_{i=1}^{N_s} w_{k|k}^i \delta(x_{k-1} - x_{k-1}^i)$$

$$w_{k|k}^i = \frac{w_{k|k-1}^i p(z_k \mid x_k^i)}{\sum_{j=1}^{N_s} w_{k|k-1}^j p(z_k \mid x_k^j)}$$

- Posterior weights are reweighting of prior weights using likelihoods (+ normalization)

The Grid-based filter

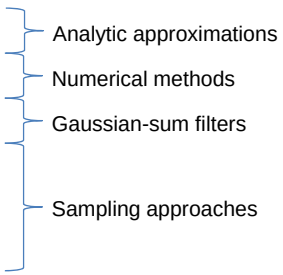
- Pros:
  - $p(x_k \mid x_{k-1}), p(z_k \mid x_k)$ assumed known, but no constraint on their (discrete) shapes
  - Easy extension to varying number of states
  - Optimal solution for the discrete-finite environment!
- Cons:
  - Curse of dimensionality
    - Inefficient if the state space is large
  - Statically considers *all* possible hypotheses

Suboptimal solutions

- In many cases these assumptions do not hold
  - Practical environments are nonlinear, non-Gaussian, continuous

➔Approximations are necessary…

- Extended Kalman filter (EKF)
- Approximate grid-based methods
- Multiple-model estimators
- Unscented Kalman filter (UKF)
- Particle filters (PF)
- …



The extended Kalman filter

- The idea: local linearization of the dynamic system might be sufficient description of the nonlinearity
- The model: nonlinear system with additive noise

$$
\begin{aligned}
x_k &= f_k(x_{k-1}) + v_k \\
z_k &= h_k(x_k) + w_k \\
v_k &\sim N(0, Q_k) \\
w_k &\sim N(0, R_k)
\end{aligned}
$$

The extended Kalman filter

- *f, h* are approximated using a first-order Taylor series expansion (eval at state estimations)

Predict:

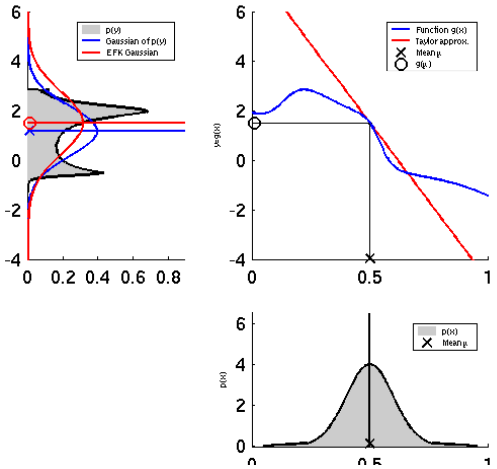
$$
\begin{aligned}
\hat{x}_{k|k-1} &= f_k(\hat{x}_{k-1|k-1}) \\
P_{k|k-1} &= \hat{F}_k P_{k-1|k-1} \hat{F}_k^T + Q_k
\end{aligned}
$$

Update:

$$
\begin{aligned}
S_k &= \hat{H}_k P_{k|k-1} \hat{H}_k^T + R_k \\
K_k &= P_{k|k-1} \hat{H}_k^T S_k^{-1} \\
\hat{x}_{k|k} &= \hat{x}_{k|k-1} + K_k \left(z_k - h_k(\hat{x}_{k|k-1}) \right) \\
P_{k|k} &= \left[I - K_k \hat{H}_k \right] P_{k|k-1}
\end{aligned}
$$

$$
\begin{aligned}
\hat{F}_k[i, j] &= \left. \frac{\partial f_k[i]}{\partial x_k[j]} \right|_{x_k = \hat{x}_{k-1|k-1}} \\
\hat{H}_k[i, j] &= \left. \frac{\partial h_k[i]}{\partial x_k[j]} \right|_{x_k = \hat{x}_{k|k-1}}
\end{aligned}
$$

The extended Kalman filter



Yacov Hel-Or

The extended Kalman filter

- Pros
  - Good approximation when models are near-linear
  - Efficient to calculate
    (de facto method for navigation systems and GPS)
- Cons
  - Only approximation (optimality not proven)
  - Still a single Gaussian approximations
    - Nonlinearity → non-Gaussianity (e.g. bimodal)
  - If we have multimodal hypothesis, and choose incorrectly – can be difficult to recover
  - Inapplicable when *f,h* discontinuous

Intermission

Questions?

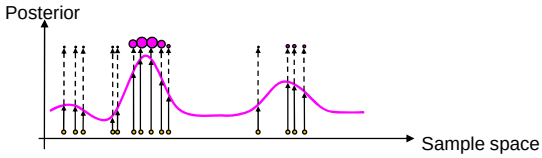
Particle filtering

- Family of techniques
  - Condensation algorithms (MacCormick&Blake, ‘99)
  - Bootstrap filtering (Gordon et al., ‘93)
  - Particle filtering (Carpenter et al., ‘99)
  - Interacting particle approximations (Moral ‘98)
  - Survival of the fittest (Kanazawa et al., ‘95)
  - Sequential Monte Carlo methods (SMC,SMCM)
  - SIS, SIR, ASIR, RPF, ….
- Statistics introduced in 1950s. Incorporated in vision in Last decade

Particle filtering

- Many variations, one general concept:

Represent the posterior pdf by a set of randomly chosen weighted samples (particles)



- Randomly Chosen = Monte Carlo (MC)
- As the number of samples become very large – the characterization becomes an equivalent representation of the true pdf

Particle filtering

- Compared to previous methods
  - Can represent any arbitrary distribution
    - multimodal support
  - Keep track several hypotheses simultaneously
  - **Approximate representation of complex model rather than exact representation of simplified model**
- The basic building-block: *Importance Sampling*

Monte Carlo integration

- Evaluate complex integrals using probabilistic techniques
- Assume we are trying to estimate a complicated integral of a function *f* over some domain *D*:

$$F = \int_D f(\vec{x}) d\vec{x}$$

- Also assume there exists some PDF *p* defined over *D*

Monte Carlo integration

- Then

$$F = \int_D f(\vec{x}) d\vec{x} = \int_D \frac{f(\vec{x})}{p(\vec{x})} p(\vec{x}) d\vec{x}$$

- But

$$\int_D \frac{f(\vec{x})}{p(\vec{x})} p(\vec{x}) d\vec{x} = E\left[\frac{f(\vec{x})}{p(\vec{x})}\right], x \sim p$$

- This is true for <u>any</u> PDF *p* over D!

Monte Carlo integration

- Now, if we have i.i.d random samples $\vec{x}_1,\ldots,\vec{x}_N$ sampled from *p*, then we can approximate $E\left[\frac{f(\vec{x})}{p(\vec{x})}\right]$ by

$$F_N = \frac{1}{N}\sum_{i=1}^{N}\frac{f(\vec{x}_i)}{p(\vec{x}_i)}$$

- Guaranteed by law of large numbers:

$$N \to \infty, F_N \overset{a.s}{\to} E\left[\frac{f(\vec{x})}{p(\vec{x})}\right] = F$$

Importance Sampling (IS)

- What about $p(\vec{x}) = 0$?
- If *p* is very small, f / p can be arbitrarily large, 'damaging' the average
  - Design *p* such that f / p is bounded
  - Rule of thumb: take p similar to f as possible

Importance weights

***Importance* or *proposal* density**

- The effect: get more samples in 'important' areas of *f*, i.e. where *f* is large

Convergence of MC integration

Pafnuty Lvovich Chebyshev

- Chebyshev's inequality: let X be a random variable with expected value μ and std σ. For any real number k>0,

$$\Pr\{|X-\mu|\geq k\sigma\}\leq\frac{1}{k^2}$$

- For example, for $k=\sqrt{2}$, it shows that at least half the values lie in interval $(\mu-\sqrt{2}\sigma,\mu+\sqrt{2}\sigma)$
- Let $y_i=\frac{f(x_i)}{p(x_i)}$, then MC estimator is $F_N=\frac{1}{N}\sum_{i=1}^{N}y_i$

Convergence of MC integration

- By Chebyshev's,

$$\Pr\{|F_N-E[F_N]|\geq\left(\frac{V[F_N]}{\delta}\right)^{1/2}\}\leq\delta\qquad(k=1/\sqrt{\delta})$$

$$V[F_N]=V\left[\frac{1}{N}\sum_{i=1}^{N}y_i\right]=\frac{1}{N^2}V\left[\sum_{i=1}^{N}y_i\right]=\frac{1}{N^2}\sum_{i=1}^{N}V[y_i]=\frac{1}{N}V[y]$$

➔ $$\Pr\{|F_N-F|\geq\frac{1}{\sqrt{N}}\left(\frac{V[y]}{\delta}\right)^{1/2}\}\leq\delta$$

- Hence, for a fixed threshold, the error decreases at rate $1/\sqrt{N}$

Convergence of MC integration

- Meaning
  1. To cut the error in half, it is necessary to evaluate 4 times as many samples
  2. Convergence rate is independent of the integrand dimension!
     - On contrast, the convergence rate of grid-based approximations decreases as N_x increases

IS for Bayesian estimation

$$E(f(X)) = \int_X f(x_{0:k}) p(x_{0:k} \mid z_{1:k}) dx_{0:k}$$
$$= \int_X f(x_{0:k}) \frac{p(x_{0:k} \mid z_{1:k})}{q(x_{0:k} \mid z_{1:k})} q(x_{0:k} \mid z_{1:k}) dx_{0:k}$$

- We characterize the posterior pdf using a set of samples (particles) and their weights
$$\{x_{0:k}^i, w_k^i\}_{i=1}^N$$
- Then the joint posterior density at time k is approximated by
$$p(x_{0:k} \mid z_{1:k}) \approx \sum_{i=1}^N w_k^i \delta(x_{0:k} - x_{0:k}^i)$$

IS for Bayesian estimation

- We draw the samples from the importance density $q(x_{0:k} \mid z_{1:k})$ with importance weights

$$w_k^i \propto \frac{p(x_{0:k} \mid z_{1:k})}{q(x_{0:k} \mid z_{1:k})}$$

- Sequential update (after some calculation…)

Particle update

$$x_k^i \sim q(x_k \mid x_{k-1}^i, z_k)$$

Weight update

$$w_k^i = w_{k-1}^i \frac{p(z_k \mid x_k^i)\, p(x_k^i \mid x_{k-1}^i)}{q(x_k^i \mid x_{k-1}^i, z_k)}$$

Sequential Importance Sampling (SIS)

$$\left[\{x_k^i, w_k^i\}_{i=1}^N\right] = \mathrm{SIS}\left[\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^N, z_k\right]$$

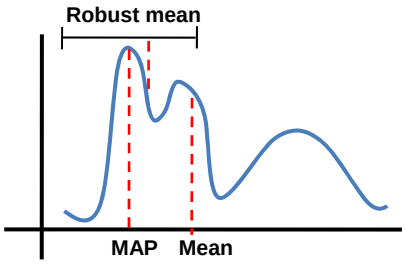
- FOR i=1:N
  - Draw $x_k^i \sim q(x_k \mid x_{k-1}^i, z_k)$
  - Update weights $w_k^i = w_{k-1}^i \frac{p(z_k \mid x_k^i)\, p(x_k^i \mid x_{k-1}^i)}{q(x_k^i \mid x_{k-1}^i, z_k)}$
- END
- Normalize weights

State estimates

- Any function $f(x_k)$ can be calculated by discrete pdf approximation

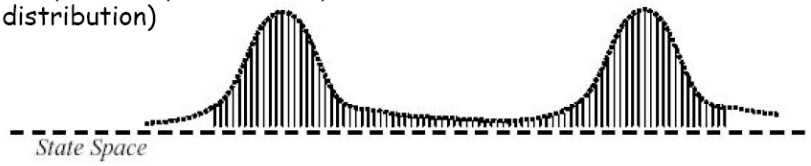
$$E\left[f(x_k)\right] = \frac{1}{N}\sum_{i=1}^{N} w_k^i f(x_k^i)$$

- Example:
  - Mean (simple average)
  - MAP estimate: particle with largest weight
  - Robust mean: mean within window around MAP estimate



Choice of importance density

Sample from prior belief q(x) (for instance, the uniform distribution)



Compute importance weights, w(x) = p(x) /q(x)

Hsiao et al.

Choice of importance density

- Most common (suboptimal): the transitional prior

$$q(x_k \mid x_{k-1}^i, z_k) = p(x_k \mid x_{k-1}^i)$$

$$\Rightarrow w_k^i = w_{k-1}^i \frac{p(z_k \mid x_k^i)\, p(x_k^i \mid x_{k-1}^i)}{q(x_k^i \mid x_{k-1}^i, z_k)} = w_{k-1}^i p(z_k \mid x_k^i)$$

Grid filter weight update: $w_{k|k}^i = \dfrac{w_{k|k-1}^i p(z_k \mid x_k^i)}{\sum_{j=1}^{N_s} w_{k|k-1}^j p(z_k \mid x_k^j)}$

The degeneracy phenomenon

- Unavoidable problem with SIS: after a few iterations most particles have negligible weights
  - Large computational effort for updating particles with very small contribution to $p(x_k \mid z_{1:k})$
- Measure of degeneracy - the effective sample size:

$$N_{eff} = \frac{1}{\sum_{i=1}^{N} (w_k^i)^2}$$

  - Uniform: $N_{eff} = N$, severe degeneracy: $N_{eff} = 1$

Resampling

- The idea: when degeneracy is above some threshold, eliminate particles with low importance weights and multiply particles with high importance weights

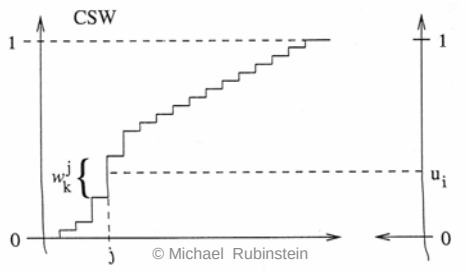
$$\{x_k^i, w_k^i\}_{i=1}^N \rightarrow \{x_k^{i*}, \tfrac{1}{N}\}_{i=1}^N$$

- The new set is generated by sampling with replacement from the discrete representation of $p(x_k \mid z_{1:k})$ such that $\Pr\{x_k^{i*} = x_k^j\} = w_k^j$

Resampling

$$\left[\{x_k^{i*}, w_k^i\}_{i=1}^N\right] = \text{RESAMPLE}\left[\{x_k^i, w_k^i\}_{i=1}^N\right]$$

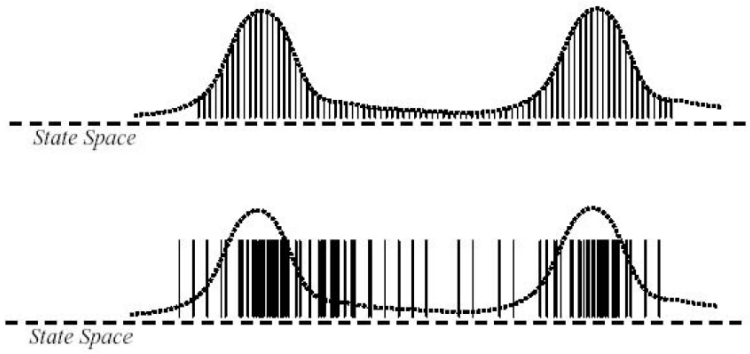
- Generate N i.i.d variables $u_i \sim U[0,1]$
- Sort them in ascending order
- Compare them with the cumulative sum of normalized weights



Ristic et al.

Resampling

- Complexity: O(NlogN)
  - O(N) sampling algorithms exist



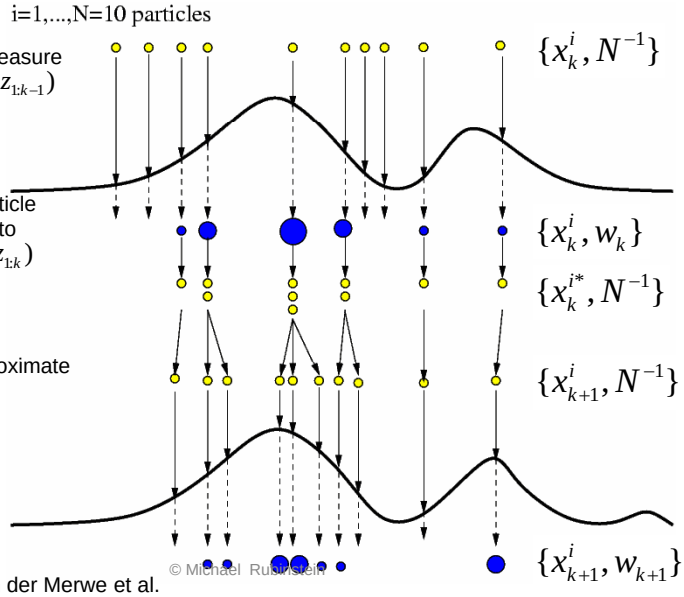
Hsiao et al.

Generic PF

$$\left[\{x_k^i, w_k^i\}_{i=1}^N\right] = \mathrm{PF}\left[\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^N, z_k\right]$$

- Apply SIS filtering $\left[\{x_k^i, w_k^i\}_{i=1}^N\right] = \mathrm{SIS}\left[\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^N, z_k\right]$
- Calculate N_{eff}
- IF $N_{eff} < N_{thr}$
  - $\left[\{x_k^i, w_k^i\}_{i=1}^N\right] = \mathrm{RESAMPLE}\left[\{x_k^i, w_k^i\}_{i=1}^N\right]$
- END

Generic PF



Van der Merwe et al.

PF variants

- Sampling Importance Resampling (SIR)
- Auxiliary Sampling Importance Resampling (ASIR)
- Regularized Particle Filter (RPF)
- Local-linearization particle filters
- Multiple models particle filters (maneuvering targets)
- …

Sampling Importance Resampling (SIR)

- A.K.A Bootstrap filter, Condensation

- **Initialize** $\{x_0^i, w_0^i\}_{i=1}^N$ from prior distribution X_0
- For k > 0 do
  - **Resample** $\{x_{k-1}^i, w_{k-1}^i\}_{i=1}^N$ into $\{x_{k-1}^{i*}, \frac{1}{N}\}_{i=1}^N$
  - **Predict** $x_k^i \sim p(x_k \mid x_{k-1} = x_{k-1}^{i*})$
  - **Reweight** $w_k^i = p(z_k \mid x_k = x_k^i)$
  - **Normalize** weights
  - **Estimate** $\hat{x}_k$ (for display)

Red pill or blue pill?

1. We had enough – show us some videos!
2. 15 minute walk through a multiple-target-tracking system

Multiple Targets (MTT/MOT)

- Previous challenges
  - Full/partial occlusions
  - Entering/leaving the scene
  - …
- And in addition
  - Estimating the number of objects
  - Computationally tractable for multiple simultaneous targets
  - Interaction between objects
  - Many works on multiple single-target filters

BraMBLe: A Bayesian Multiple-Blob Tracker

M. Isard and J. MacCormick

Compaq Systems Research Center

ICCV 2001

Some slides taken from Qi Zhao
Some images taken from Isard and MacCormick

BraMBLE

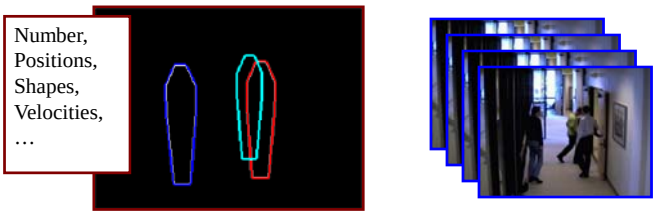
- First rigorous particle filter implementation with variable number of targets
- Posterior distribution is constructed over possible object configurations and number

- *Sensor*: single static camera
- *Tracking*: SIR particle filter
- *Performance*: real-time for 1-2 simultaneous objects

The BraMBLe posterior

$$p(x_k \mid z_{1:k})$$

State at frame *k* Image Sequence



State space

- Hypothesis configuration:

$$X_k = (m_k, x_k^1, x_k^2, ..., x_k^m)$$

- Object configuration:

$$N_x = 1 + 13M_{\max}$$

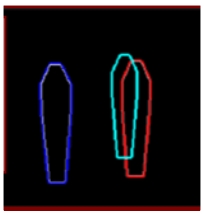
$$x_k^i = (\phi_k^i, \mathbf{X}_k^i, \mathbf{V}_k^i, \mathbf{S}_k^i)$$

identifier

position $\mathbf{X} = (x, z)$

velocity $\mathbf{V} = (v_x, v_z)$

shape $\mathbf{S} = (w_f, w_w, w_s, w_h, h, \theta, \alpha_w, \alpha_s)$

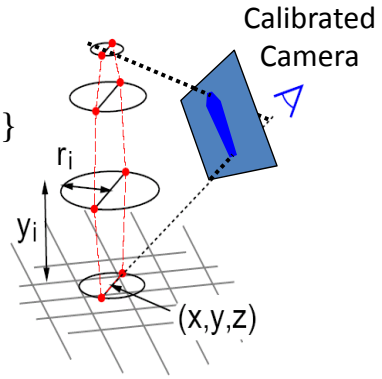


Object model

- A person is modeled as a *generalized-cylinder* with vertical axis in the world coords

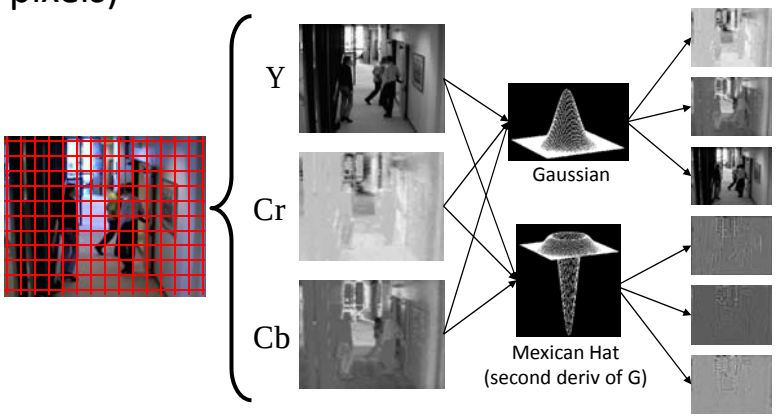
$$\mathbf{S} = (w_f, w_w, w_s, w_h, h, \theta, \alpha_w, \alpha_s)$$

$$(r_i, y_i) = \{(w_f, 0), (w_w\theta, \alpha_w h), (w_s\theta, \alpha_s h), (w_h, h)\}$$



Observation likelihood $p(\mathbf{Z}_t \mid \mathbf{X}_t)$

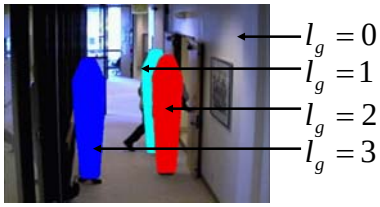
- Image overlaid with rectangular Grid (e.g. 5 pixels)



Observation likelihood $p(\mathbf{Z}_t \mid \mathbf{X}_t)$

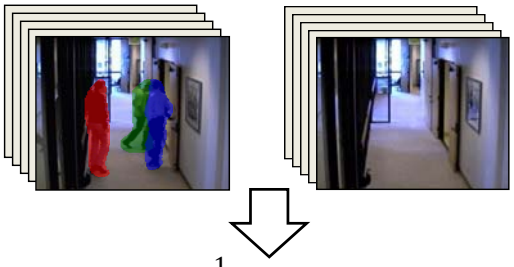
- The response values are assumed conditionally independent given X

$$\mathrm{p}(\mathbf{Z} \mid \mathbf{X}) = \prod_g p(z_g \mid \mathbf{X}) = \prod_g p(z_g \mid l_g)$$



Appearance models

- GMMs for background and foreground are trained using kmeans



$$p(z_g \mid l_g = 0) = \frac{1}{K}\sum_k N(\mu_g^k, \Sigma_g^k + \Delta_B) + \tau_B \qquad K = 4$$

$$p(z_g \mid l_g \neq 0) = \frac{1}{K}\sum_k N(\mu_F^k, \Sigma_F^k) + \tau_F \qquad K = 16$$

Observation likelihood

$$\log\left(\frac{p(z_g \mid l_g \neq 0)}{p(z_g \mid l_g = 0)}\right)$$

System (prediction) model $p(X_t \mid X_{t-1})$

- The number of objects can change:
  - Each object has a constant probability λ_r to remain in the scene.
  - At each time step, there is constant probability λ_i that a new object will enter the scene.
- $X_{t-1}^{n'} = (m_{t-1}^{n'}, \tilde{x}_{t-1}^{n',1}, \ldots) \rightarrow X_t^n = (m_t^n, \tilde{x}_t^{n,1}, \ldots)$

Prediction function

1. **set** $m_t^n := 0$.
2. **for** $i = 1$ **to** $m_{t-1}^{n'}$:
   (a) **generate** r **distributed as** $U[0,1)$.
   (b) **if** $r < \lambda_r$ **set** $m_t^n := m_t^n + 1$ **and** $\tilde{x}^{n,m_t^n} := f(\tilde{x}_{t-1}^{n',i})$
3. **generate** r **distributed as** $U[0,1)$.
4. **if** $r < \lambda_i$ **set** $m_t^n := m_t^n + 1$ **and set** $\tilde{x}^{n,m_t^n} := g(t)$

Figure 6: **The multi-object prediction algorithm**

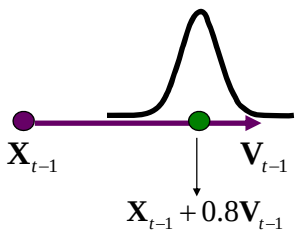
Initialization function

Prediction function

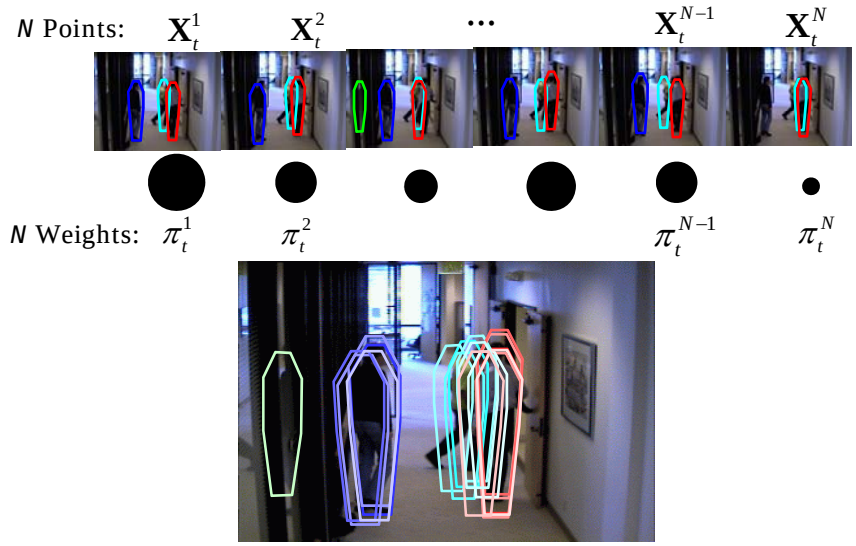
- Motion evolution: damped constant velocity
- Shape evolution: 1st order auto-regressive process model (ARP)

$$f(\phi, (\mathcal{X}, \mathcal{V}, \mathcal{S})^T) = (\phi, (\mathcal{X}', \mathcal{V}', \mathcal{S}')^T)$$

$$\mathcal{X}' = \mathcal{X} + \lambda_v \mathcal{V} + b_X \omega_X$$
$$\mathcal{V}' = \lambda_v \mathcal{V} + b_X \omega_X$$
$$\mathcal{S}' = A_S(\mathcal{S} - \overline{\mathcal{S}}) + B_S \omega_S$$
$$\overline{\mathcal{S}} = (\mu_1, \ldots, \mu_8)$$
$$B_S = \mathrm{diag}(\rho_1, \ldots, \rho_8)$$
$$A_S = \mathrm{diag}(a_1, \ldots, a_8)$$



Particles



Estimate $\hat{X}_t$

- Denote $\mathbf{M}_t = \{\Phi_1, \ldots, \Phi_M\}$ the set of existing unique identifiers

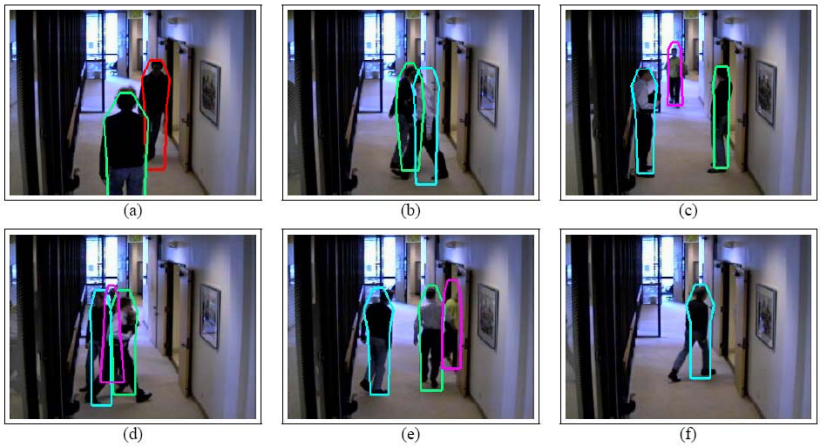
Total probability that object Φ_i exists

(particle,target)

for $i = 1$ **to** M_t
(a) **compute** $\mathcal{M}_t^{\Phi_i} = \{(n,j) \mid \phi_t^{n,j} = \Phi_i\}$.
(b) **compute** $\Pi_t^{\Phi_i} = \sum_{(n,j)\in\mathcal{M}_t^{\Phi_i}} \pi_t^n$.
(c) **if** $\Pi_t^{\Phi_i} > \lambda_d$ **estimate** $\hat{x}_t^{\Phi_i} = \sum_{(n,j)\in\mathcal{M}_t^{\Phi_i}} \pi_t^n s_t^{(n,j)} / \Pi_t^{\Phi_i}$.

Figure 7: **The estimation algorithm**

Results



- N=1000 particles
- initialization samples always generated

Results

- Single foreground model cannot distinguish between overlapping objects – causes id switches

Parameters

symbol	meaning	value
λ_r	object survival probability	0.99
λ_i	new object arrival probability	0.02
λ_d	object display threshold	0.8
δ_e	minimum physical separation between distinct objects (m)	0.5
δ_B	background likelihood additional covariance factor (grey-levels^2)	100
τ_B	background likelihood cutoff (grey-levels^{-6})	2.0×10^{-14}
τ_F	foreground likelihood cutoff (grey-levels^{-6})	3.0×10^{-13}
b_X	translation process noise (m)	0.11

	w_f	w_w	w_s	w_h	h	θ	α_w	α_s
mean μ_i	0.20m	0.22m	0.25m	0.08m	1.80m	0.75	0.60	0.83
steady-state standard deviation σ_i	0.03m	0.04m	0.04m	0.02m	0.05m	0.25	0.02	0.02
process noise ρ_i	0.003m	0.002m	0.002m	0.002m	0.003m	0.05	0.001	0.001

Summary

- Particle filters were shown to produce good approximations under relatively weak assumptions
  - can deal with nonlinearities
  - can deal with non-Gaussian noise
  - Multiple hypotheses
  - can be implemented in O(N)
  - Relatively “simple”
  - **Adaptive focus on more probable regions of the state-space**

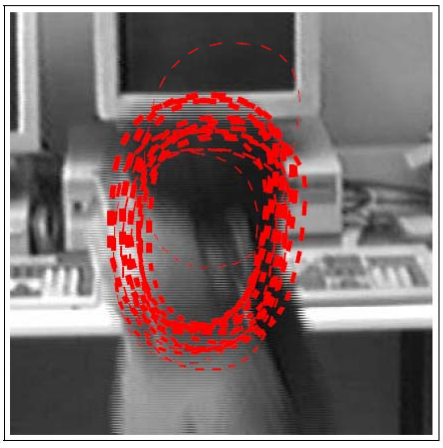
In practice

1. State (object) model
2. System (evolution) model
3. Measurement (likelihood) model
4. Initial (prior) state
5. State estimate (given the pdf)

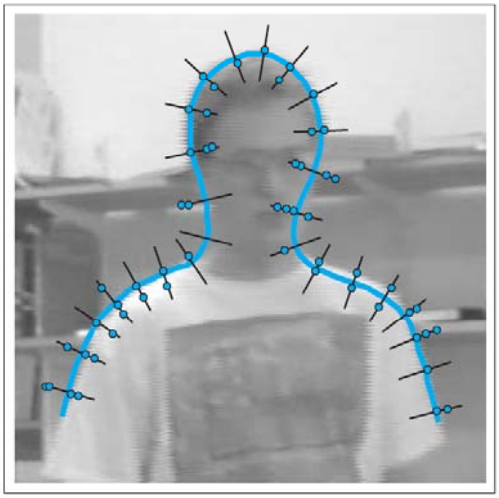
6. PF specifics
   1. Proposal density
   2. Resampling method

- Configurations for specific problems can be found in literature

Isard&Blake *CONDENSATION– conditional density propagation for visual tracking* IJCV 98



Isard&Blake *CONDENSATION– conditional density propagation for visual tracking* IJCV 98



Isard&Blake *CONDENSATION– conditional density propagation for visual tracking* IJCV 98

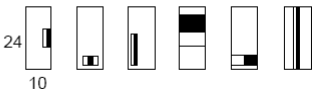
"girl dancing vigorously to a Scottish reel" – 100 particles

"bush blowing in the wind" – 1200 particles

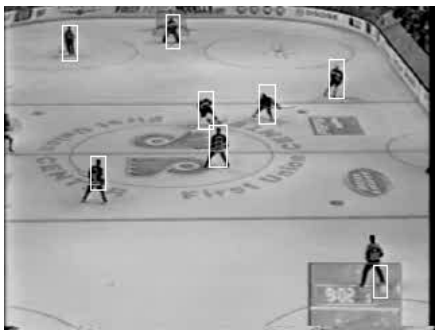
Okuma el al. *Boosted Particle Filter* ECCV 2004

- Goal: track hockey players
- Idea: AdaBoost + PF

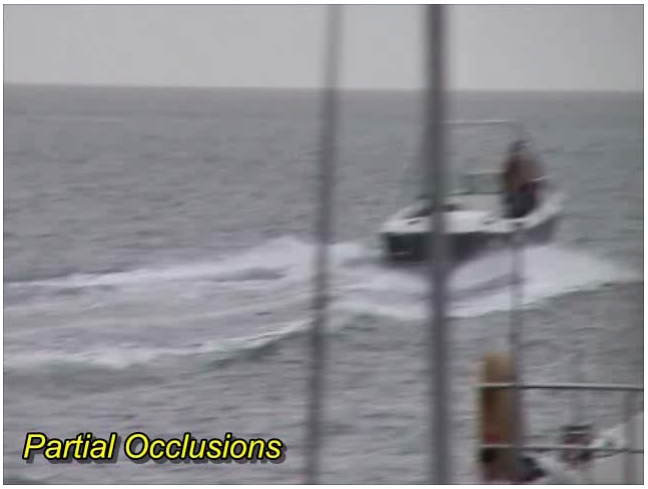
Key (Haar) features:



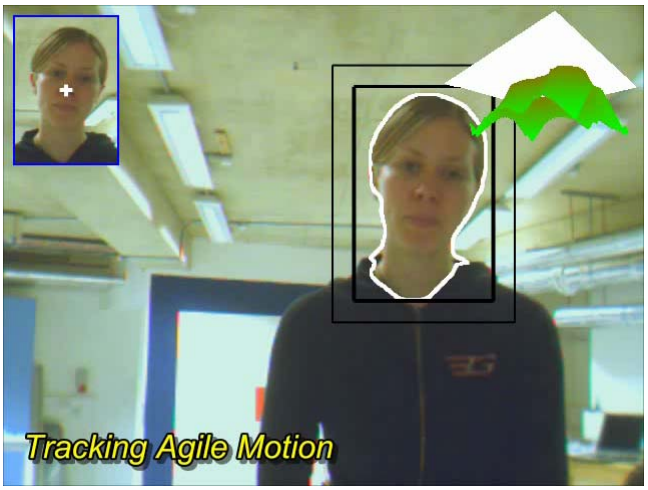
Okuma el al. *Boosted Particle Filter* ECCV 2004



Bibby&Reid *Tracking using Pixel-Wise Posteriors* (ECCV08)



Bibby&Reid *Tracking using Pixel-Wise Posteriors* (ECCV08)



Thank you!

References

- *Beyond the Kalman filter/* Ristic,Arulamplam,Gordon
  - Online tutorial: *A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking/* Arulampalam et al 2002
- *Stochastic models, estimation and control/* Peter S. Maybeck
- *An Introduction to the Kalman Filter/* Greg Welch, Gary Bishop
- *Particle filters an overview/* Matthias Muhlich

Sequential derivation 1

- Suppose at time k-1, $\{x_{0:k-1}^i, w_{k-1}^i\}_{i=1}^N$ characterize $p(x_{0:k-1} \mid z_{1:k-1})$
- We receive new measurement z_k and need to approximate $p(x_{0:k} \mid z_{1:k})$ using new set of samples
- We choose q such that

$$q(x_{0:k} \mid z_{1:k}) = q(x_k \mid x_{0:k-1}, z_{1:k}) q(x_{0:k-1} \mid z_{1:k-1})$$

And we can generate new particles

$$x_k^i \sim q(x_k \mid x_{0:k-1}^i, z_{1:k})$$

Sequential derivation 2

- For the weight update equation, it can be shown that

$$\begin{aligned} p(x_{0:k} \mid z_{1:k}) &= \frac{p(z_k \mid x_k) p(x_k \mid x_{k-1})}{p(z_k \mid z_{1:k-1})} p(x_{0:k-1} \mid z_{1:k-1}) \\ &\propto p(z_k \mid x_k) p(x_k \mid x_{k-1}) p(x_{0:k-1} \mid z_{1:k-1}) \end{aligned}$$

And so

$$\begin{aligned} w_k^i &= \frac{p(x_{0:k} \mid z_{1:k})}{q(x_{0:k} \mid z_{1:k})} = \frac{p(z_k \mid x_k) p(x_k \mid x_{k-1}) p(x_{0:k-1} \mid z_{1:k-1})}{q(x_k \mid x_{0:k-1}, z_{1:k}) q(x_{0:k-1} \mid z_{1:k-1})} \\ &= w_{k-1}^i \frac{p(z_k \mid x_k^i) p(x_k^i \mid x_{k-1}^i)}{q(x_k^i \mid x_{0:k-1}^i, z_{1:k})} \end{aligned}$$

Sequential derivation 3

- Further, if $q(x_k \mid x_{0:k-1}, z_{1:k}) = q(x_k \mid x_{k-1}, z_k)$
- Then the weights update rule becomes

$$w_k^i = w_{k-1}^i \frac{p(z_k \mid x_k^i)\, p(x_k^i \mid x_{k-1}^i)}{q(x_k^i \mid x_{k-1}^i, z_k)} \qquad (3)$$

(and need not store entire particle paths and full history of observations)

- Finally, the (filtered) posterior density is approximated by $p(x_k \mid z_{1:k}) \approx \sum_{i=1}^{N} w_k^i \delta(x_k - x_k^i)$

Choice of importance density

- Choose q to minimize variance of weights
- Optimal choice: $q(x_k \mid x_{k-1}^i, z_k)_{opt} = p(x_k \mid x_{k-1}^i, z_k)$
  $\Rightarrow w_k^i \propto w_{k-1}^i p(z_k \mid x_{k-1}^i)$
  - Usually cannot sample from q_{opt} or solve for w_k^i (in some specific cases is works)
- Most commonly used (suboptimal) alternative: $q(x_k \mid x_{k-1}^i, z_k)_{opt} = p(x_k \mid x_{k-1}^i)$
  $\Rightarrow w_k^i \propto w_{k-1}^i p(z_k \mid x_k^i)$
  - i.e. the transitional prior

Generic PF

- Resampling reduces degeneracy, but new problems arise…

1. Limits parallelization
2. *Sample impoverishment*: particles with high weights are selected many times which leads to loss of diversity
   - if process noise is small – all particles tend to collapse to single point within few interations
   - Methods exist to counter this as well…