

Stanford CS223B Computer Vision, Winter 2007

Lecture 12

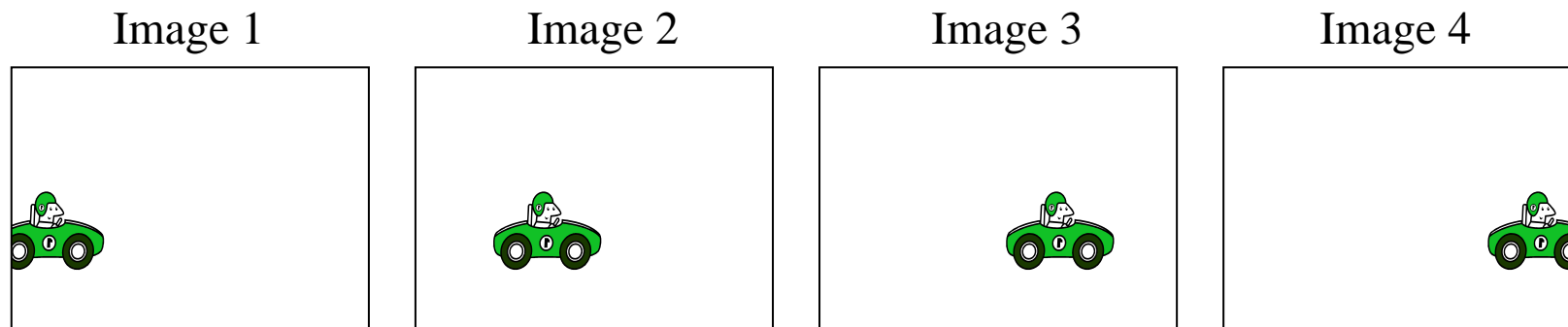
Tracking Motion

Professors Sebastian Thrun and Jana Kořecká
CAs: Vaibhav Vaish and David Stavens

Overview

- q The Tracking Problem
- q Bayes Filters
- q Particle Filters
- q Kalman Filters
- q Using Kalman Filters

The Tracking Problem



- q Can we estimate the position of the object?
- q Can we estimate its velocity?
- q Can we predict future positions?

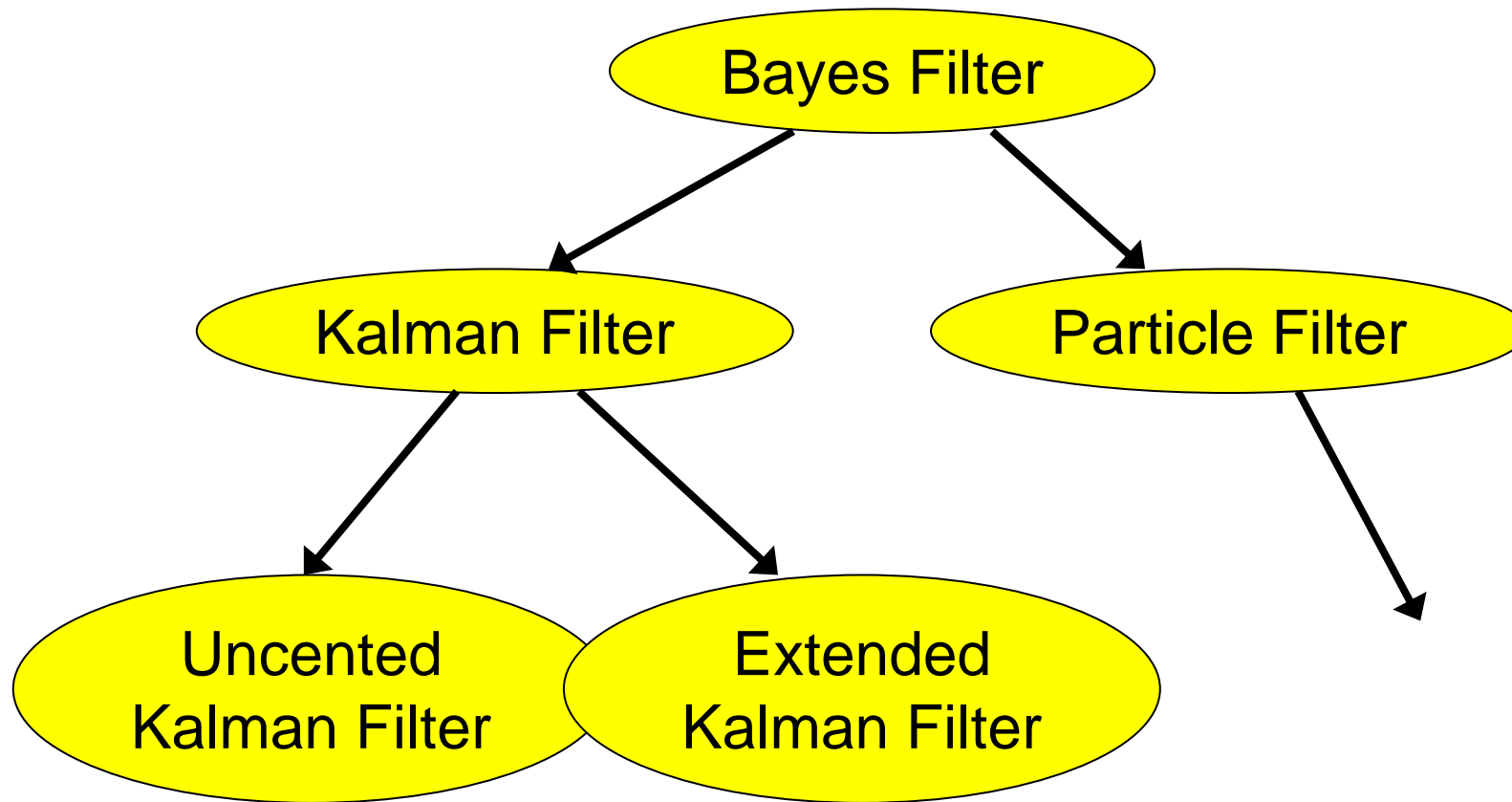
The Tracking Problem

- q Given Sequence of Images
- q Find center of moving object
- q Camera might be moving or stationary

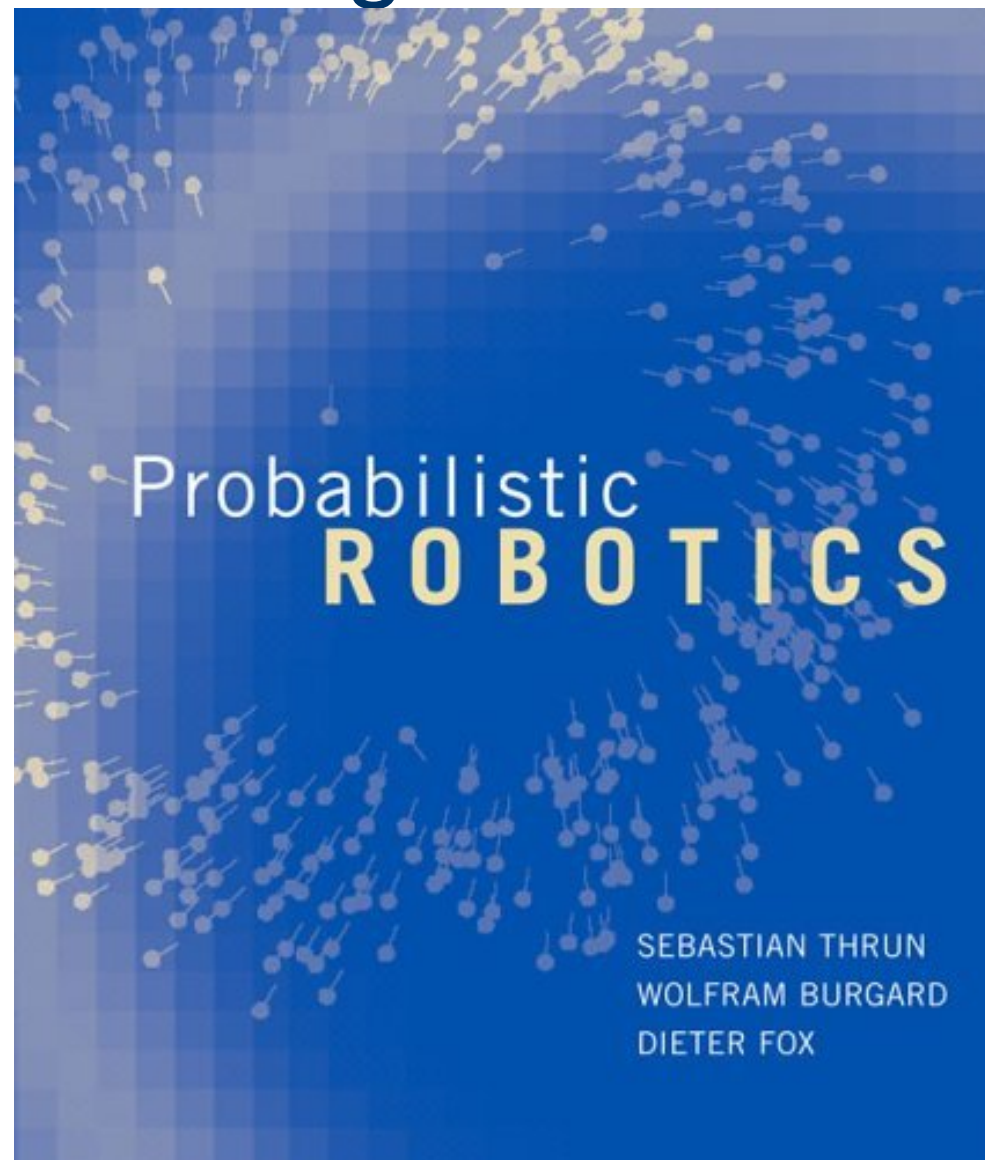
- q We assume: We can find object in individual images.
- q The Problem: Track across multiple images.

- q Is a fundamental problem in computer vision

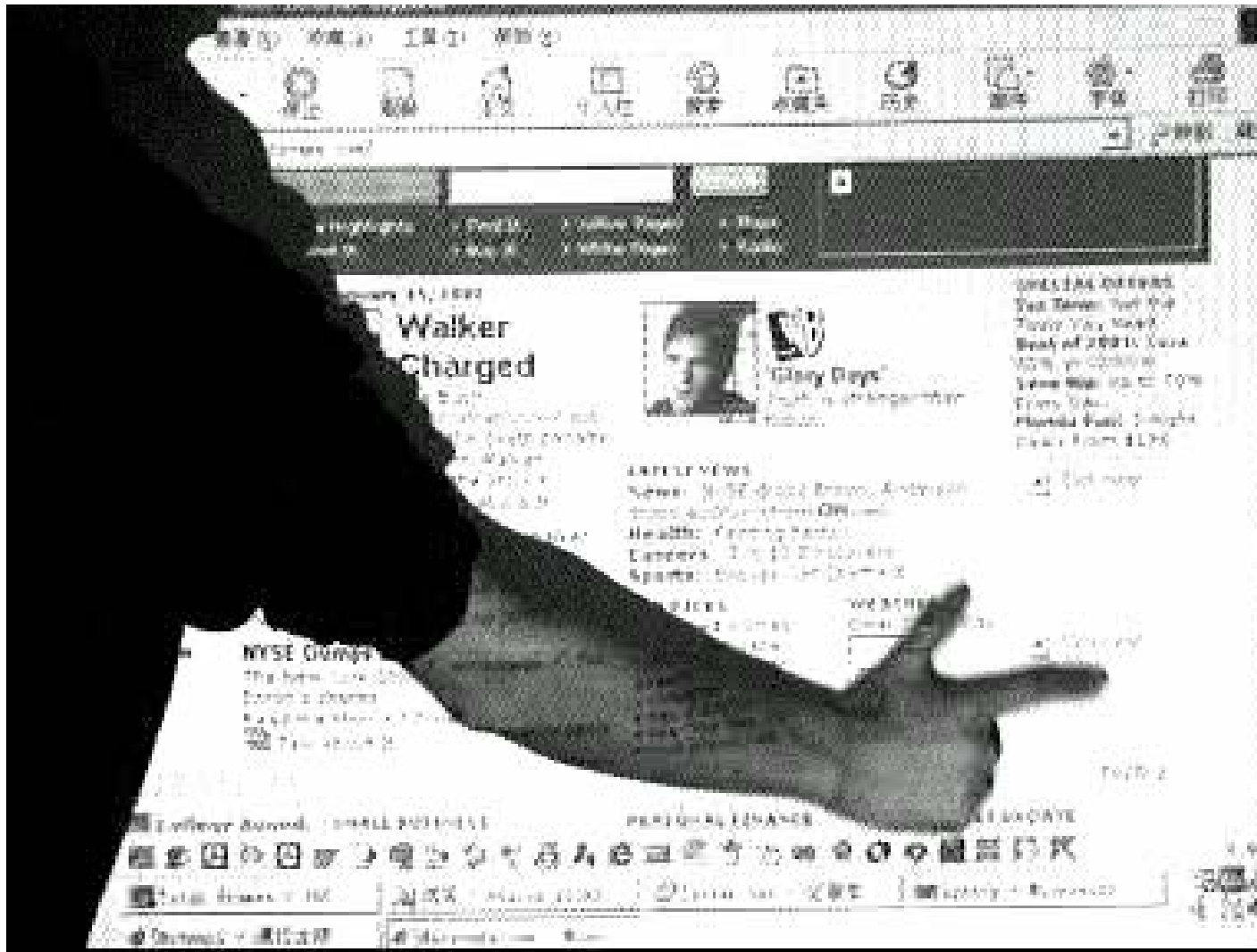
Methods



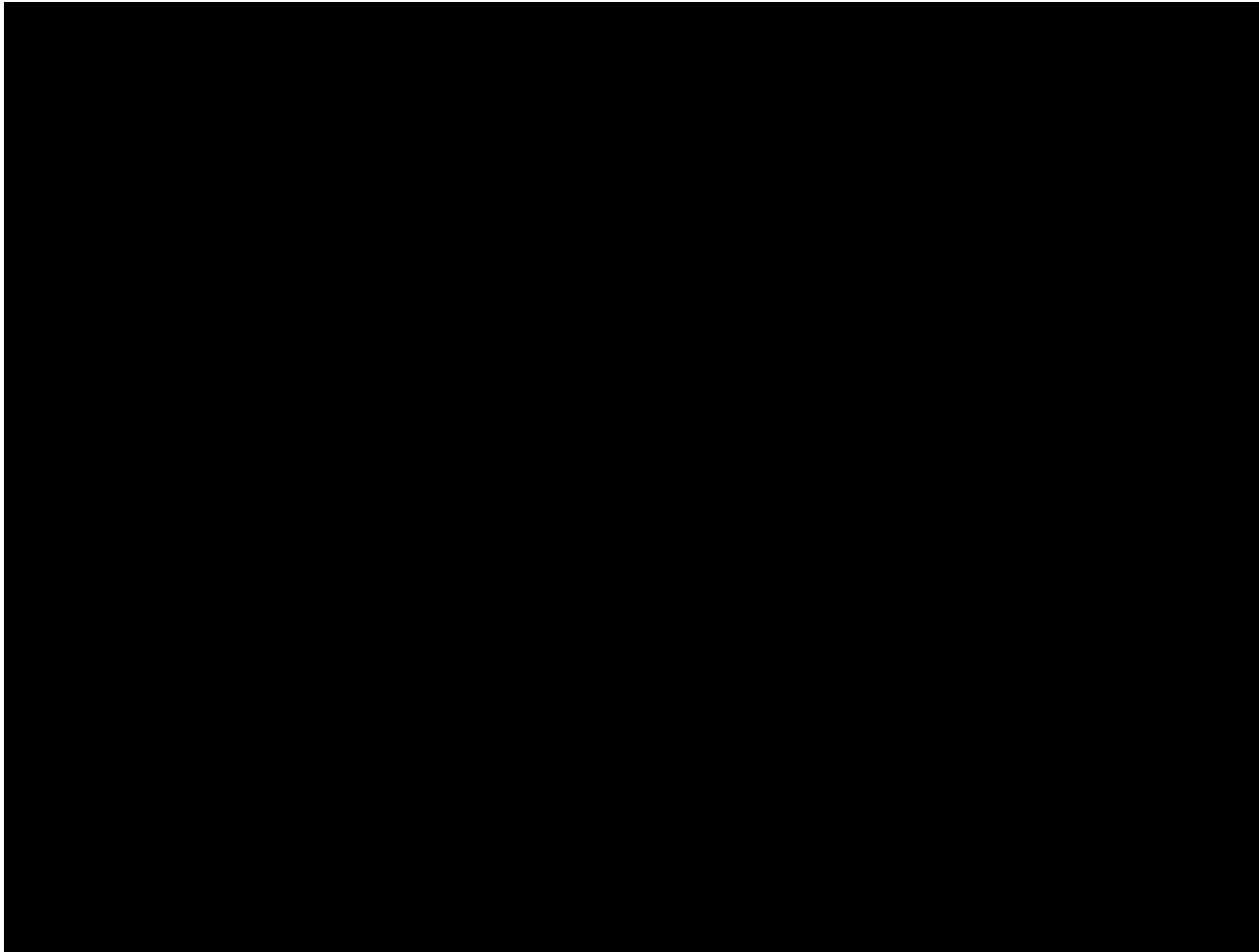
Further Reading...



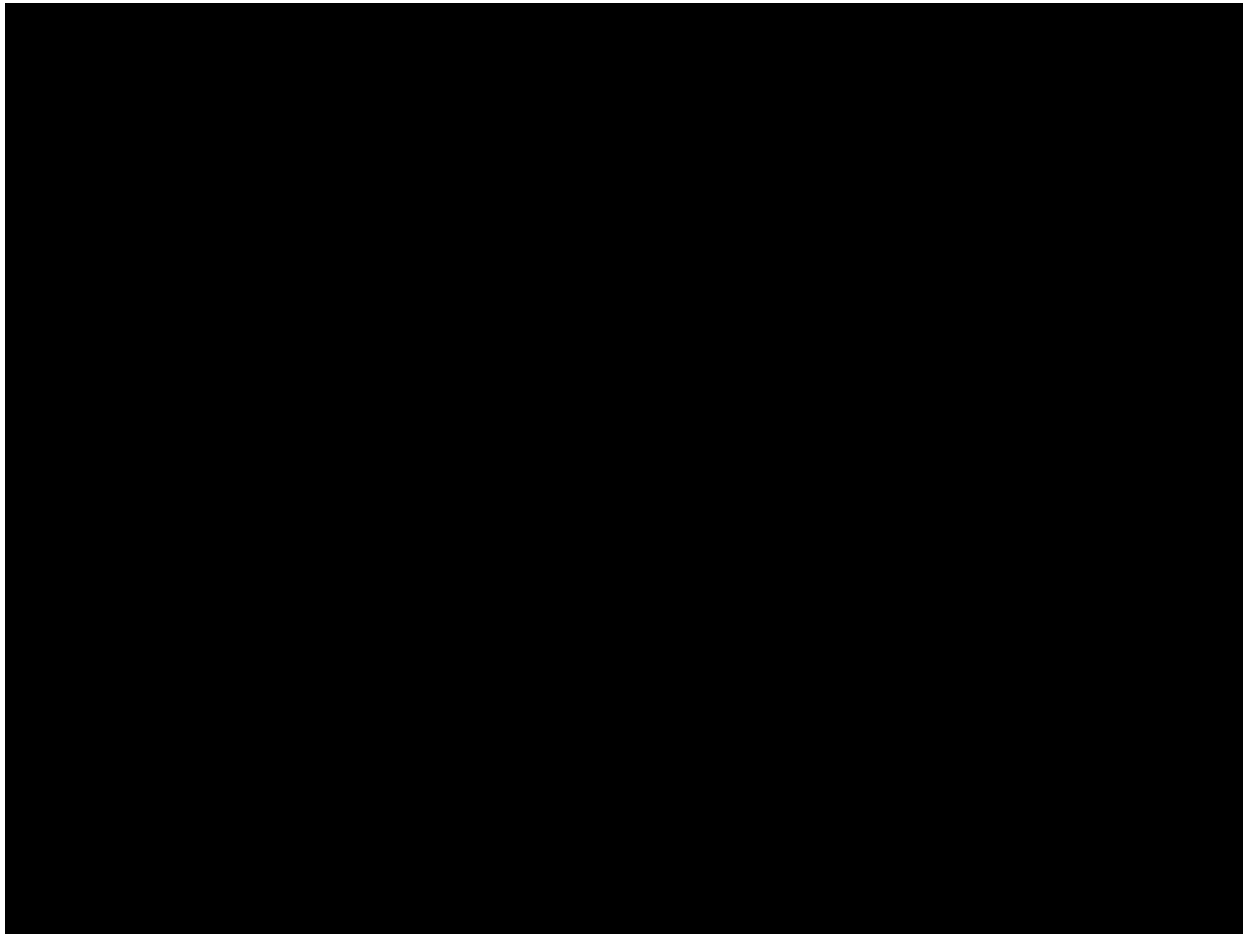
Example: Moving Object



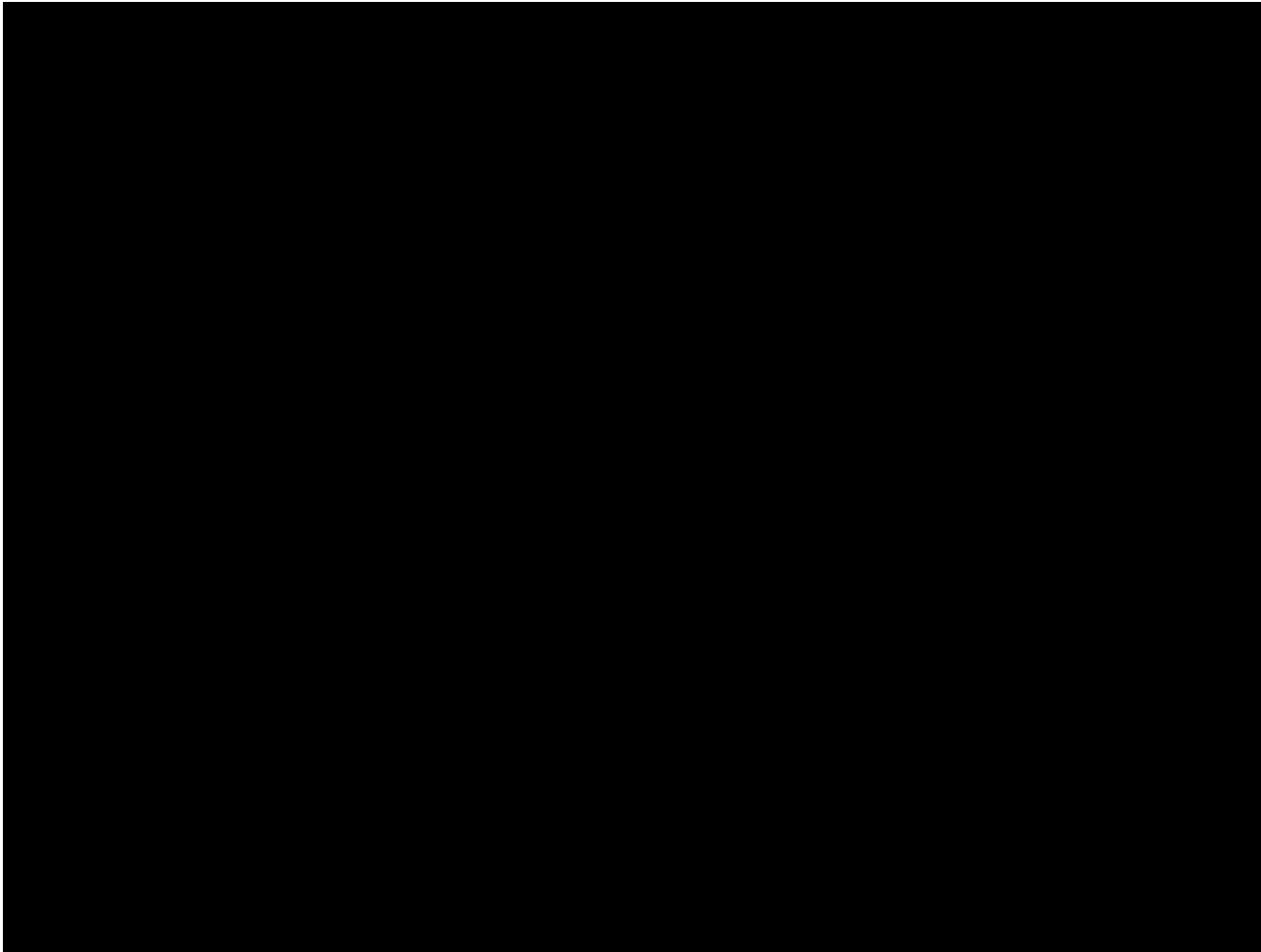
Kalman Filter Tracking



Particle Filter Tracking



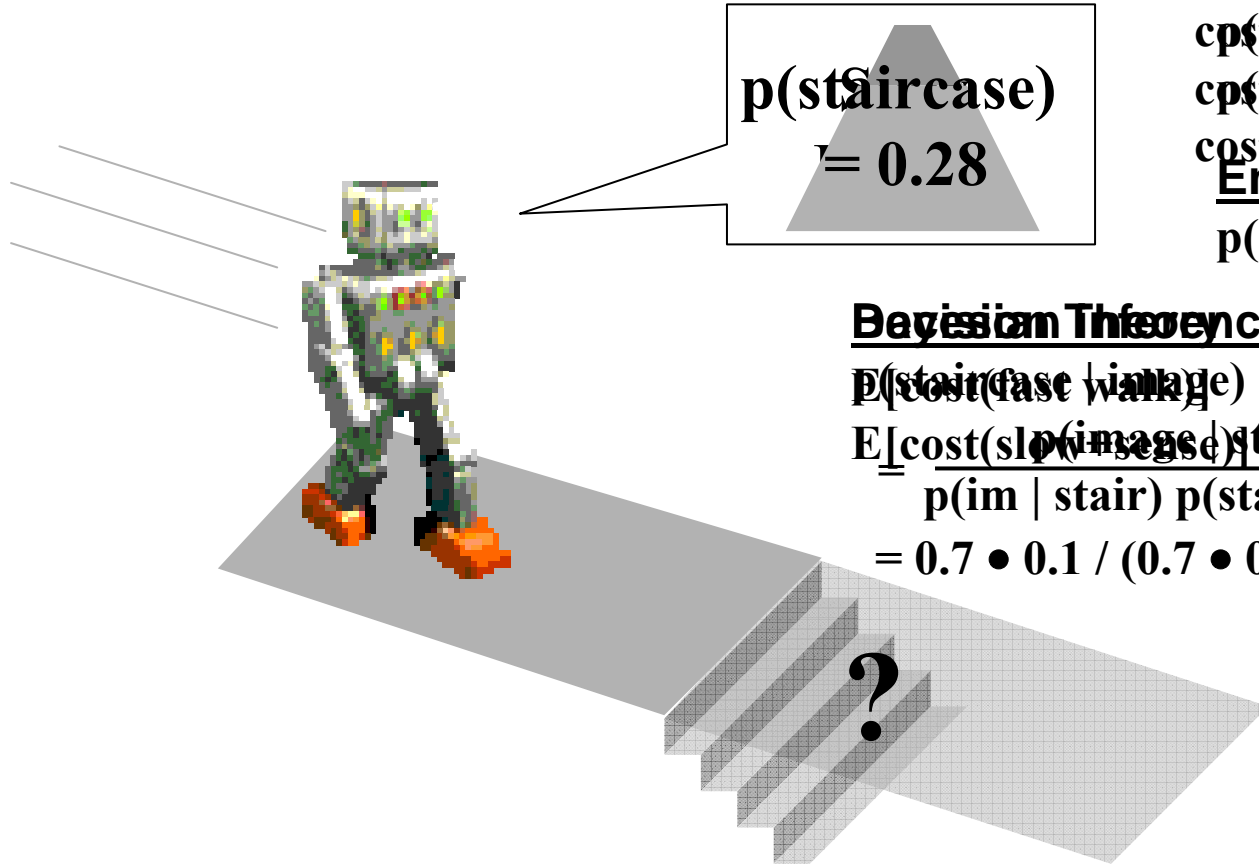
Mixture of KF / PF (Unscented PF)



Overview

- q The Tracking Problem
- q Bayes Filters
- q Particle Filters
- q Kalman Filters
- q Using Kalman Filters

Example of Bayesian Inference



$p(\text{staircase}) = 0.28$

Cost model

$\text{cost}(\text{fast} + \text{walk} | \text{staircase}) = \$1,000$

$\text{cost}(\text{fast} + \text{walk} | \text{no staircase}) = \0

$\text{cost}(\text{slow} + \text{sense}) = \1

Environment prior

$p(\text{staircase}) = 0.1$

Bayesian Inference

$E[\text{cost}(\text{fast} + \text{walk})] = \$1,000 \cdot 0.28 = \$280$

$$E[\text{cost}(\text{slow} + \text{sense}) | \text{staircase}] = \frac{p(\text{im} | \text{stair}) p(\text{stair})}{p(\text{im} | \text{stair}) p(\text{stair}) + p(\text{im} | \text{no stair}) p(\text{no stair})}$$

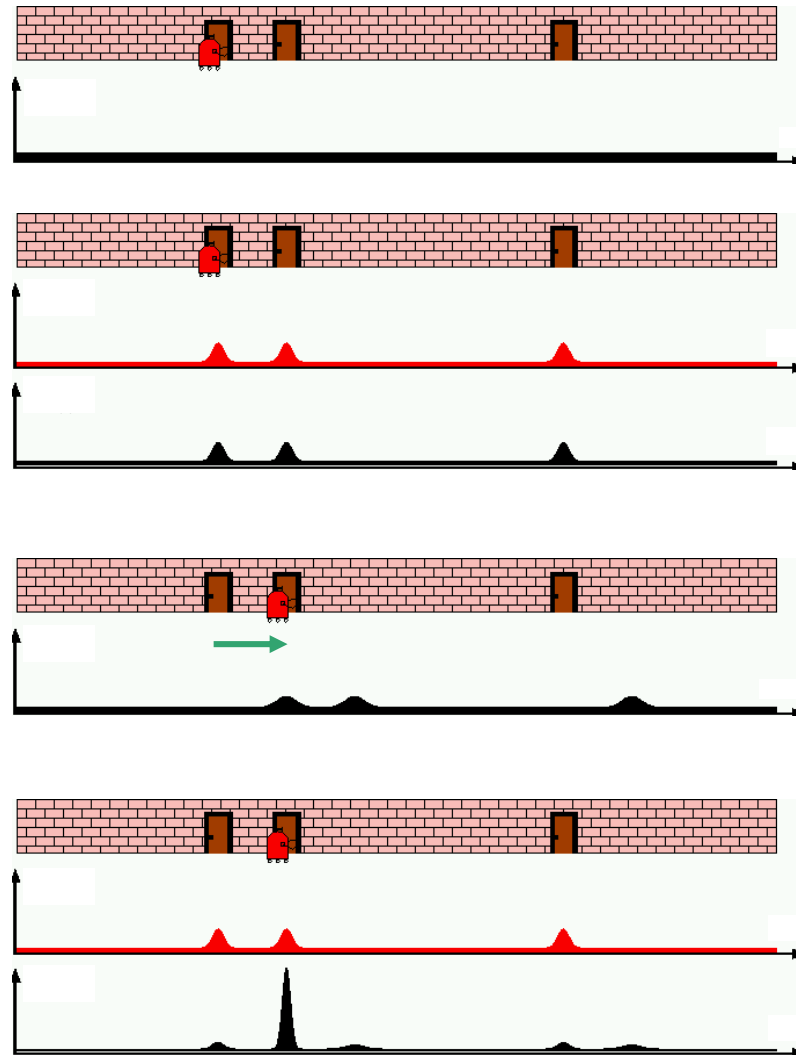
$$= 0.7 \cdot 0.1 / (0.7 \cdot 0.1 + 0.2 \cdot 0.9) = 0.28$$

Bayes Filter Definition

- q Environment state x_t
- q Measurement z_t

- q Can we calculate $p(x_t | z_1, z_2, \dots, z_t)$?

Bayes Filters Illustrated



Bayes Filters: Essential Steps

- q Belief: $\text{Bel}(x_t)$
- q Measurement update: $\text{Bel}(x_t) \stackrel{\circ}{\leftarrow} \text{Bel}(x_t) p(z_t|x_t)$
- q Time update: $\text{Bel}(x_{t+1}) \leftarrow \text{Bel}(x_t) \otimes p(x_{t+1}|u_t, x_t)$

Bayes Filters

x = state
 t = time
 z = observation
 u = action
 η = constant

$$Bel(x_t) = p(x_t | z_{0..t}, u_{0..t})$$

$$= p(x_t | z_t, u_{t-1}, z_{t-1}, \dots, u_0)$$

Bayes

$$= \eta p(z_t | x_t, u_{t-1}, z_{t-1}, \dots, z_0) p(x_t | u_{t-1}, z_{t-1}, \dots, z_0)$$

Markov

$$= \eta p(z_t | x_t) p(x_t | u_{t-1}, z_{t-1}, \dots, z_0)$$

$$= \eta p(z_t | x_t) \int p(x_t | x_{t-1}, u_{t-1}, \dots, z_0) p(x_{t-1} | u_{t-1}, \dots, z_0) dx_{t-1}$$

Markov

$$= \eta p(z_t | x_t) \int p(x_t | x_{t-1}, u_{t-1}) p(x_{t-1} | z_{t-1}, u_{t-2}, \dots, u_0) dx_{t-1}$$

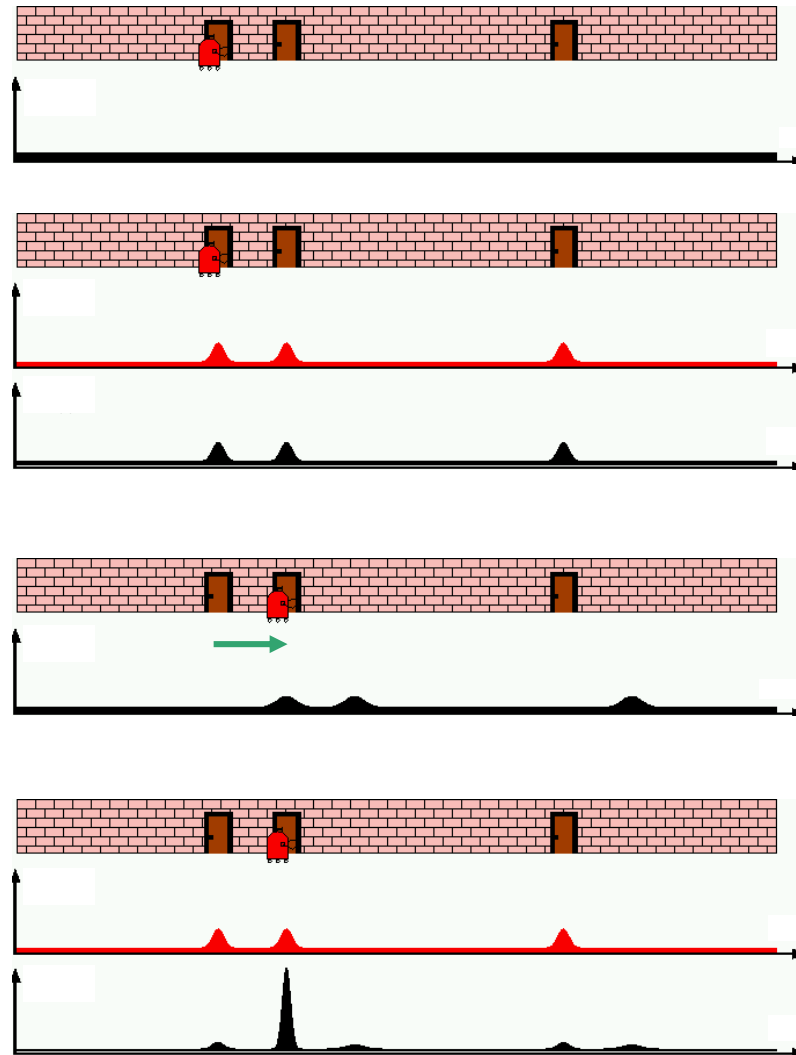
$$= \eta p(z_t | x_t) \int p(x_t | x_{t-1}, u_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

Bayes Filters

x = state
 t = time
 z = observation
 u = action

$$Bel(x_t) = \eta p(z_t | x_t) \int p(x_t | x_{t-1}, u_{t-1}) Bel(x_{t-1}) dx_{t-1}$$

Bayes Filters Illustrated



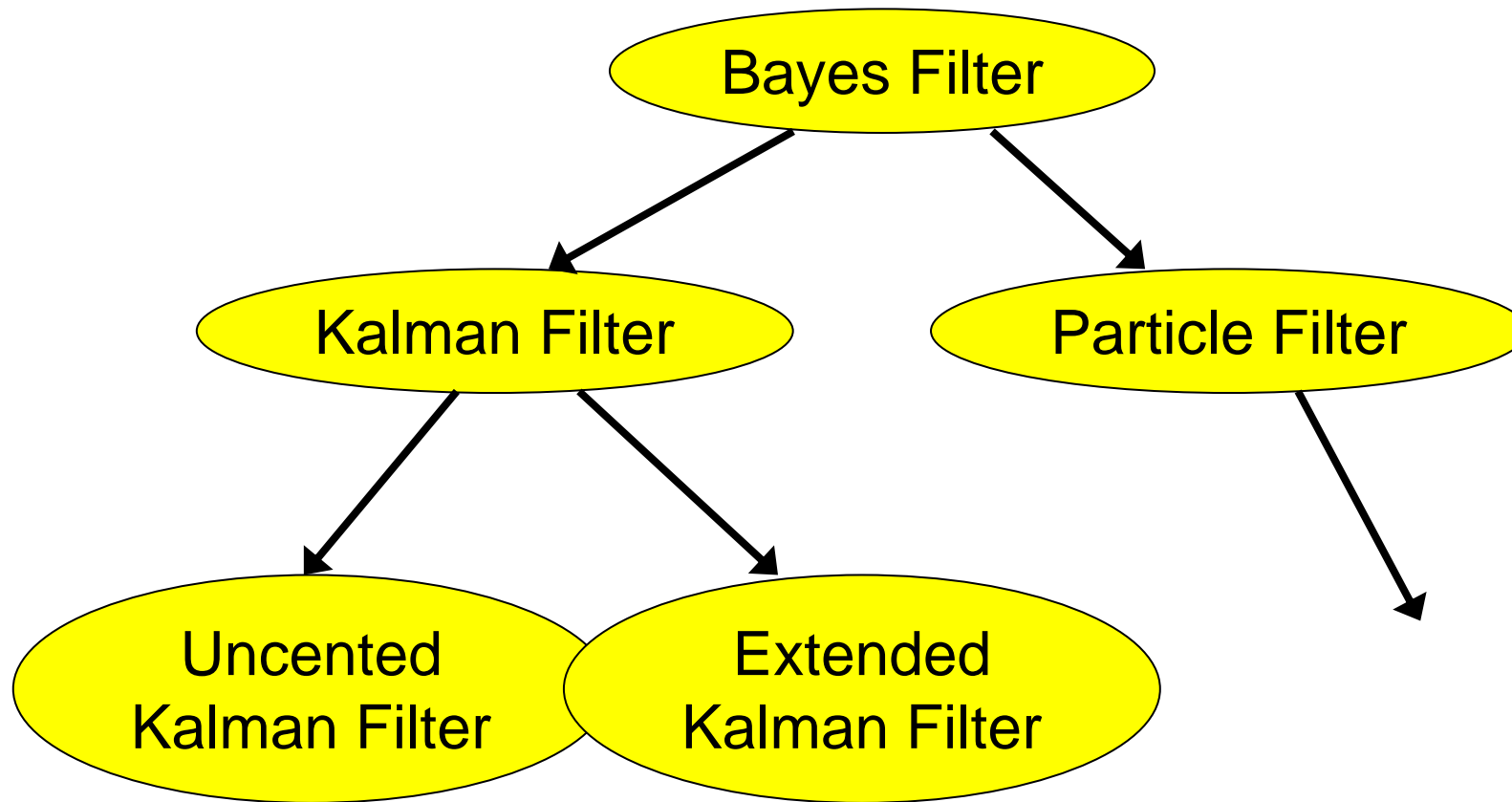
Bayes Filters

- q Initial Estimate of State

- q Iterate

- Receive measurement, update your belief (uncertainty shrinks)
- Predict, update your belief (uncertainty grows)

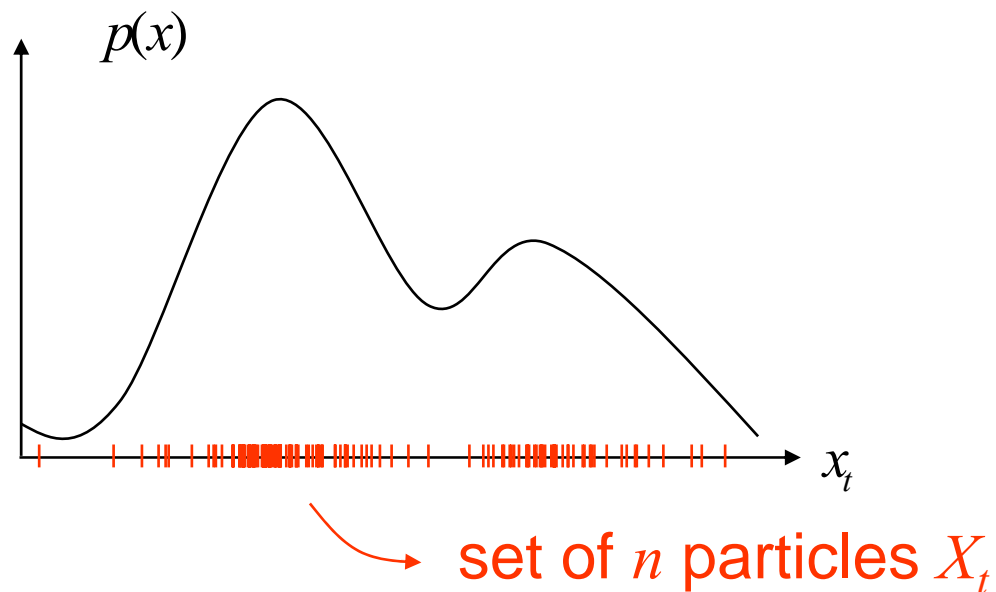
Methods



Overview

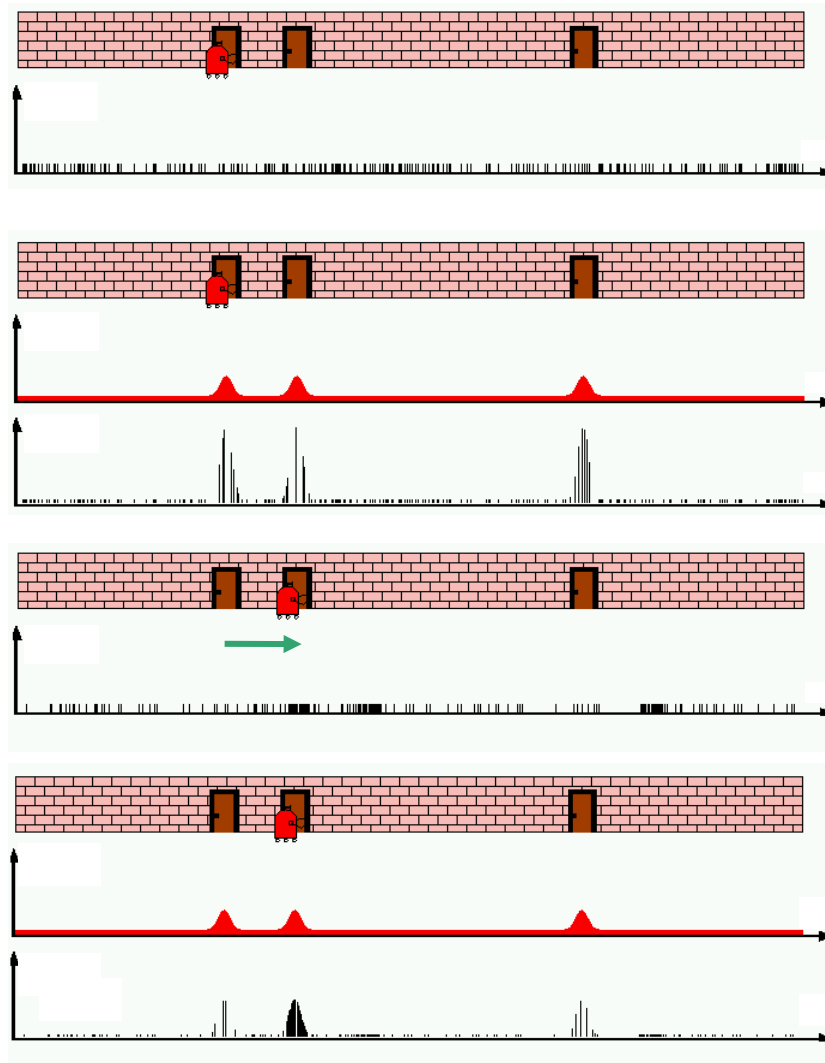
- q The Tracking Problem
- q Bayes Filters
- q Particle Filters
- q Kalman Filters
- q Using Kalman Filters

Particle Filters: Basic Idea



$$p(x_t \in X_t) \approx p(x_t | z_{1:t}) \quad (\text{equality for } n \uparrow \infty)$$

Particle Filter Explained



Basic Particle Filter Algorithm

Initialization:

$$X_0 \leftarrow n \text{ particles } x_0^{[i]} \sim p(x_0)$$

particleFilters(X_{t-1}) {

for $i=1$ to n

$$x_t^{[i]} \sim p(x_t | x_{t-1}^{[i]})$$

(prediction)

$$w_t^{[i]} = p(z_t | x_t^{[i]})$$

(importance weights)

endfor

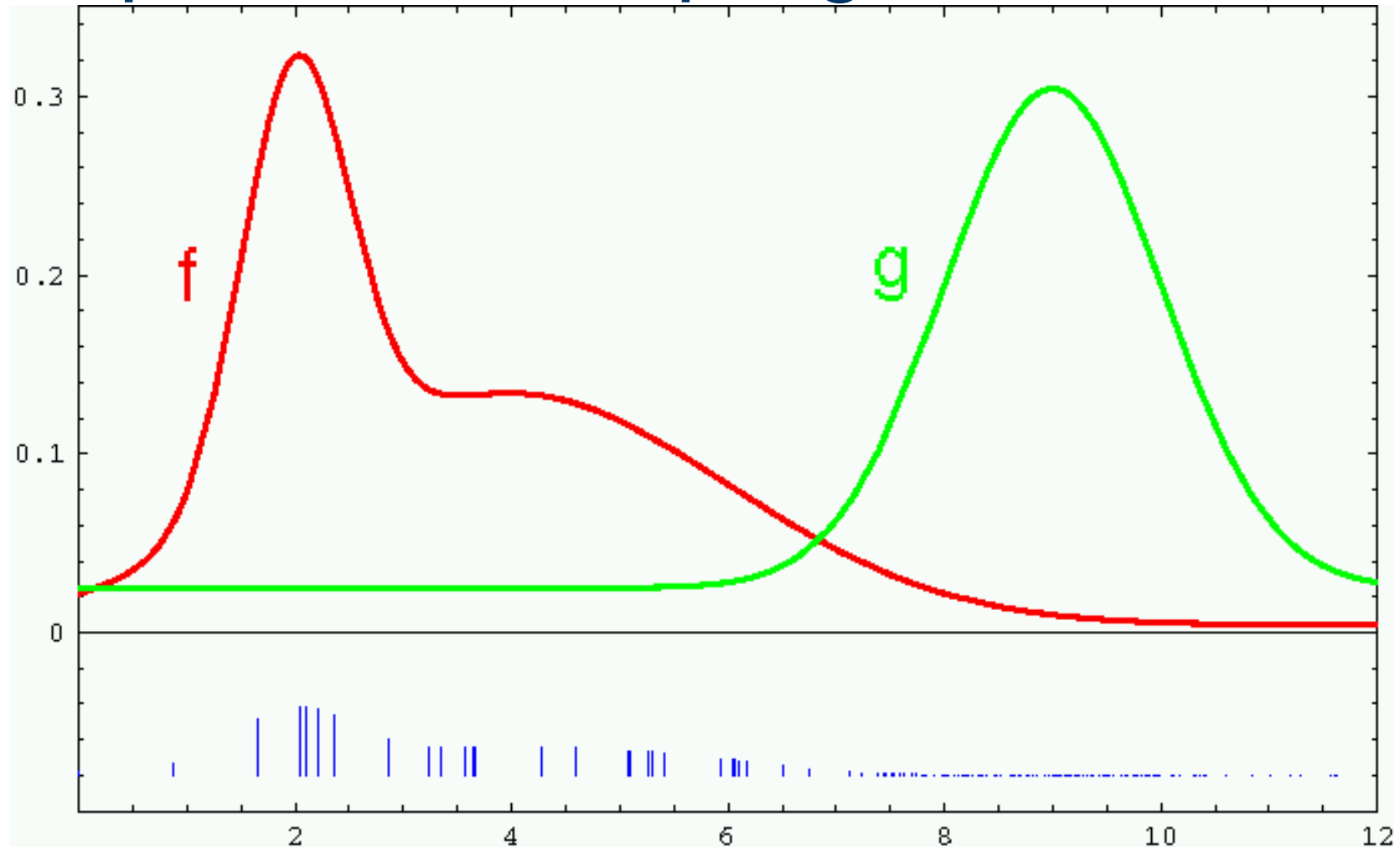
for $i=1$ to n

} include $x_t^{[i]}$ in X_t with probability $\propto w_t^{[i]}$ (resampling)

$$p(x_t | z_{1..t}, u_{1..t}) = \eta p(z_t | x_t) \int p(x_t | u_t, x_{t-1}) p(x_{t-1} | z_{1..t-1}, u_{1..t-1}) dx_{t-1}$$

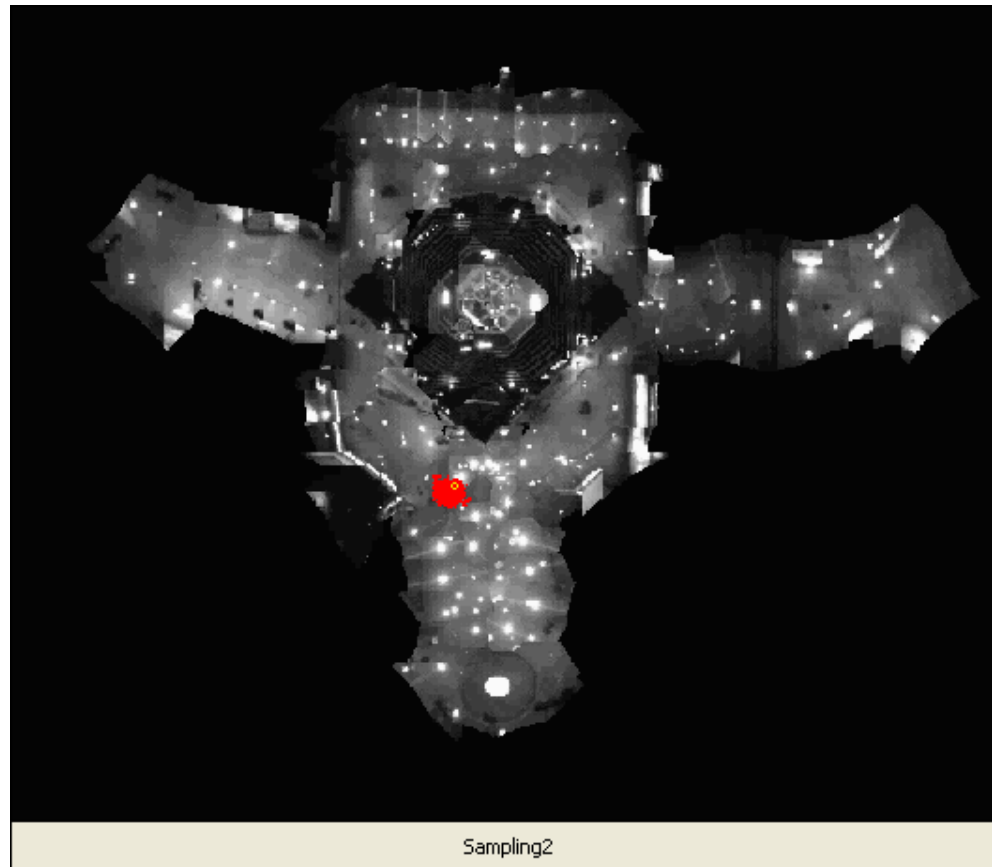
$$p(x_t \in X_t) \approx p(x_t | z_{1..t}, u_{1..t})$$

Importance Sampling



Weight samples: $w = f/g$

Particle Filter



By Frank Dellaert

Case Study:

Track moving objects from Helicopter

1. Harris Corners
2. Optical Flow (with clustering)
3. Motion likelihood function
4. Particle Filter
5. Centroid Extraction

David Stavens, Andrew Lookingbill, David Lieb, CS223b Winter 2004

1. Harris Corner Extraction



David Stavens, Andrew Lookingbill, David Lieb, CS223b Winter 2004

2. Optical Flow + Motion Detection



David Stavens, Andrew Lookingbill, David Lieb, CS223b Winter 2004

3. Motion Likelihood Function



David Stavens, Andrew Lookingbill, David Lieb, CS223b Winter 2004

4. Particle Filters



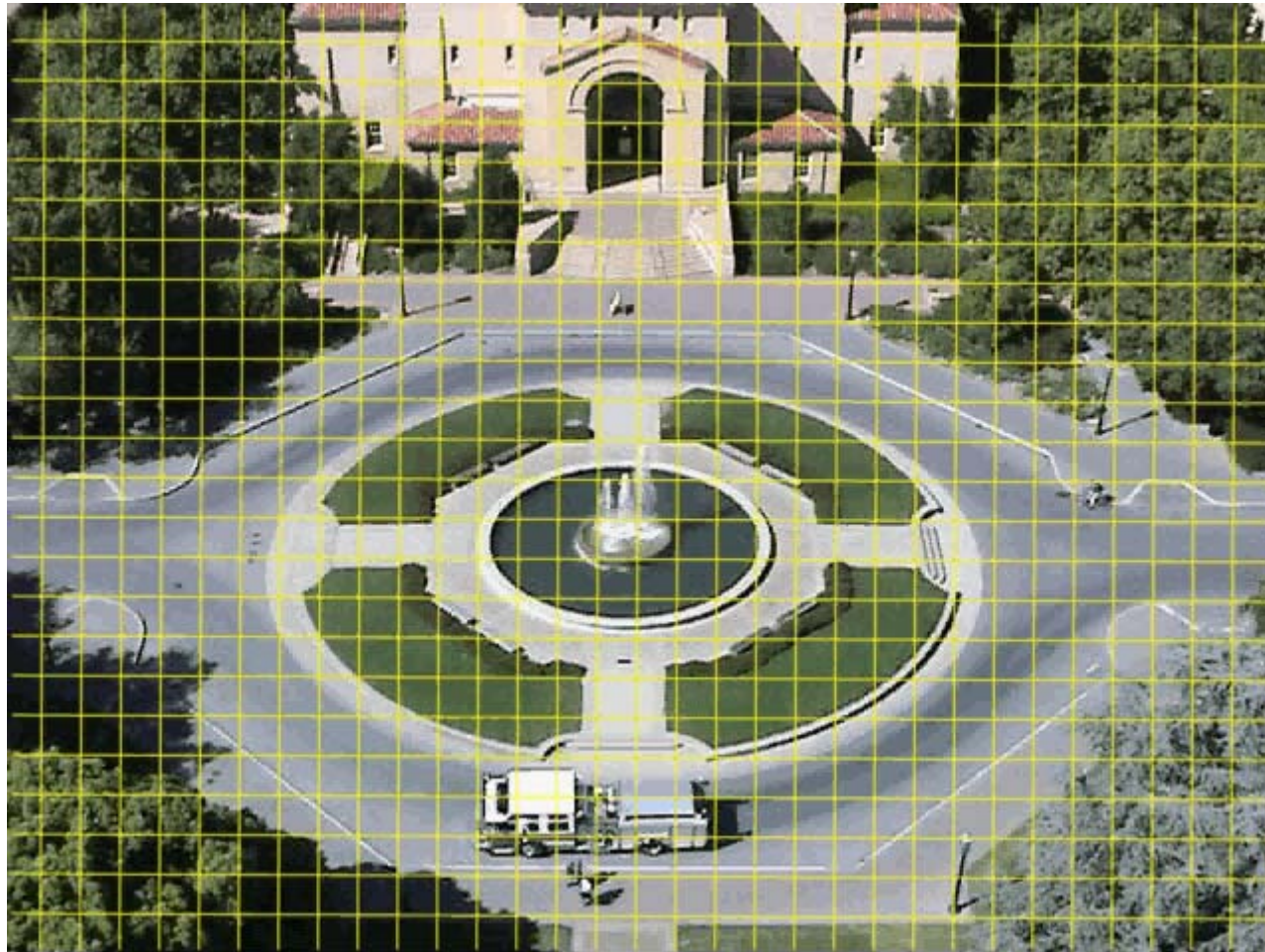
David Stavens, Andrew Lookingbill, David Lieb, CS223b Winter 2004

5. Extract Centroid



David Stavens, Andrew Lookingbill, David Lieb, CS223b Winter 2004

More Particle Filter Tracking



David Stavens, Andrew Lookingbill, David Lieb, CS223b Winter 2004

Some Robotics Examples

- q Tracking Hands, People
- q Mobile Robot localization
- q People localization
- q Car localization
- q Mapping

Examples Particle Filter



Siu Chi Chan McGill University

Another Example



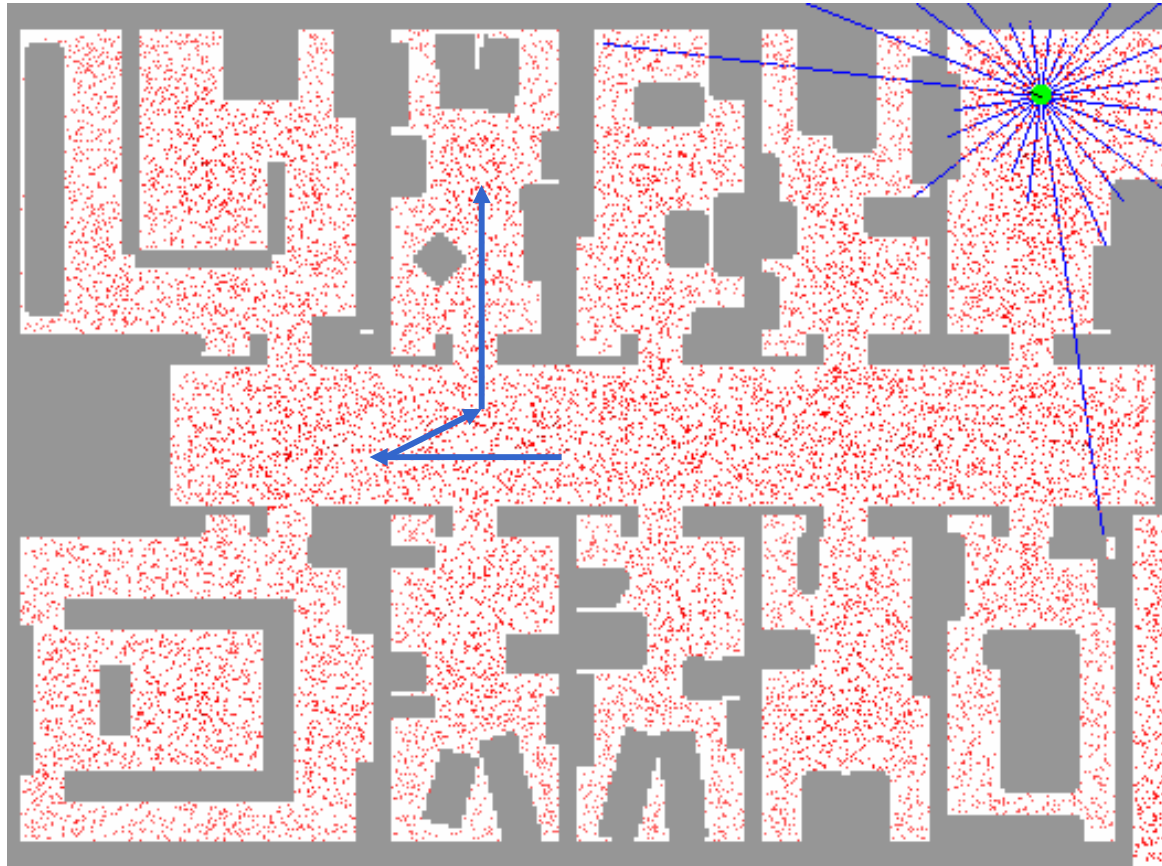
Mike Isard and Andrew Blake

Tracking Fast moving Objects



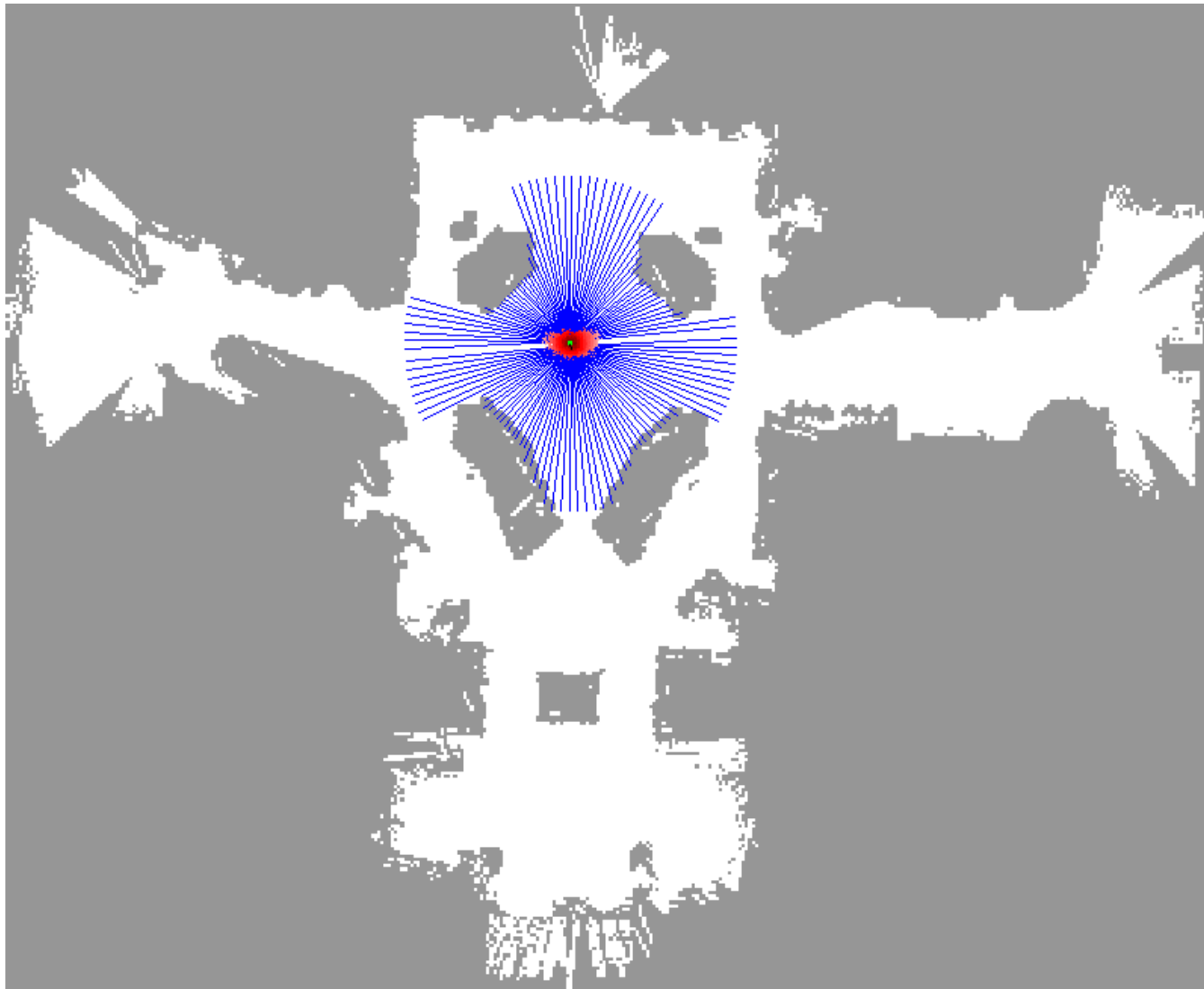
Particle Filters: Illustration

With: Wolfram Burgard, Dieter Fox, Frank Dellaert

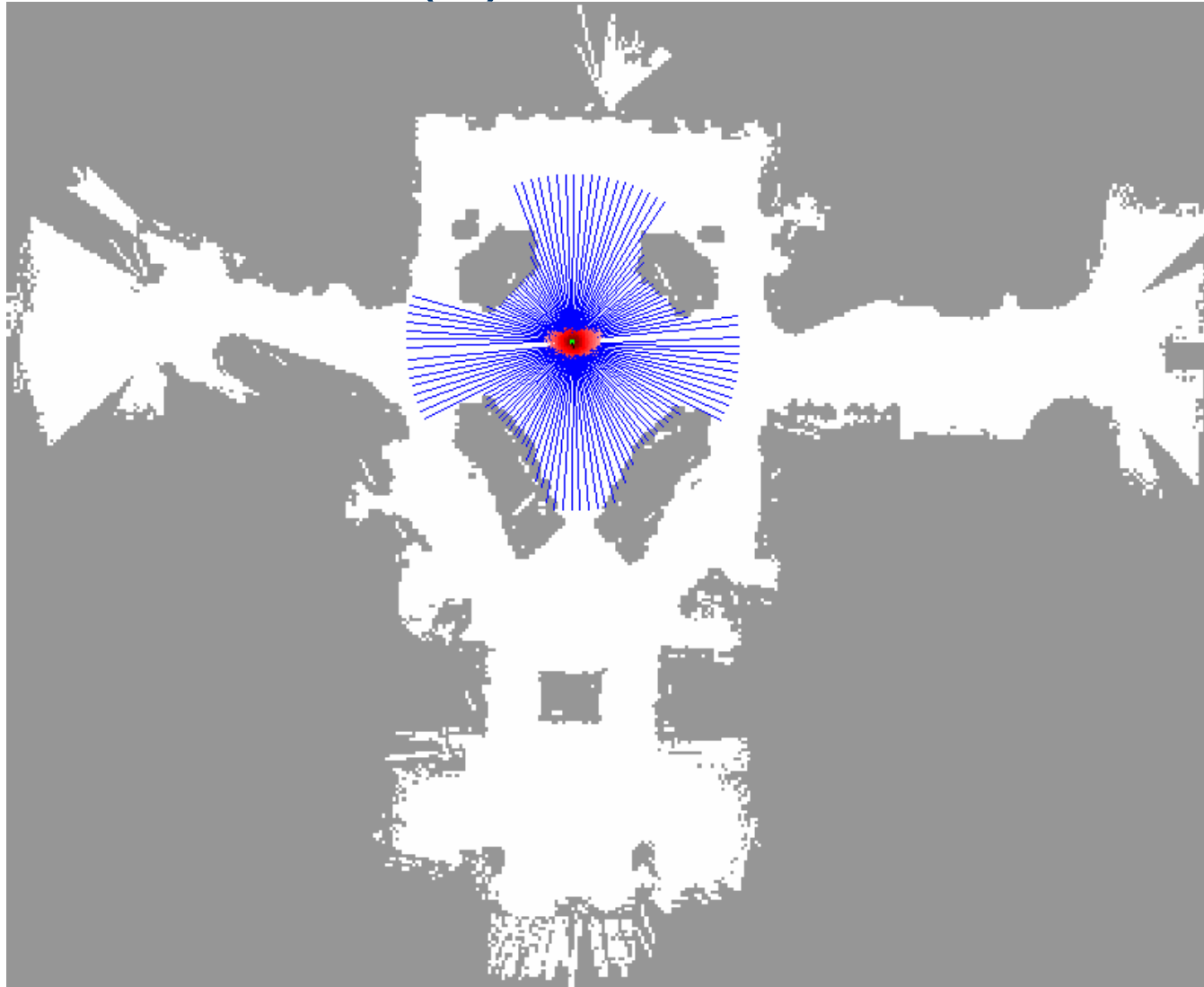


$$p(x_t \in X_t) \approx p(x_t | z_{1..t}, u_{1..t})$$

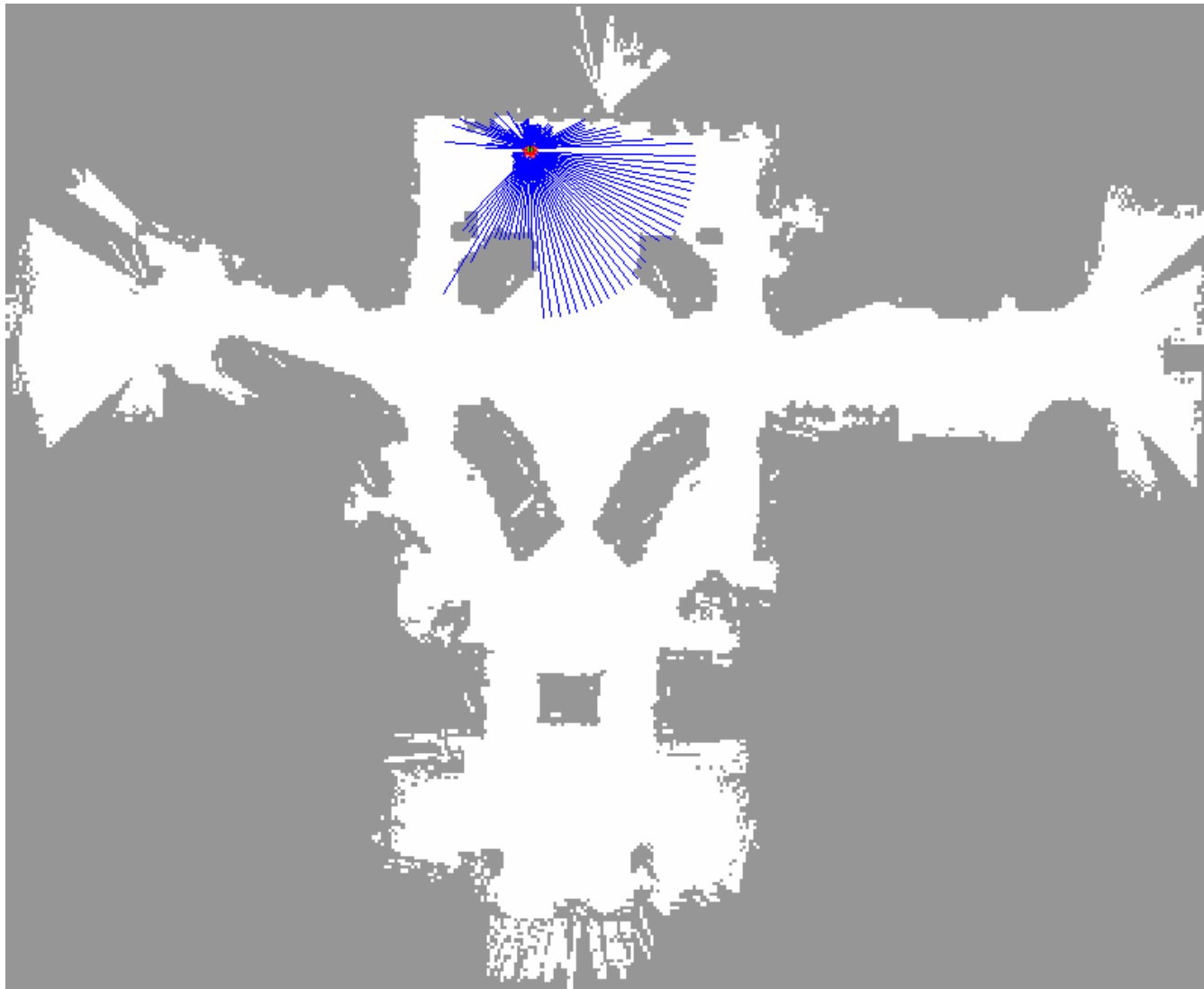
Particle Filters (1)



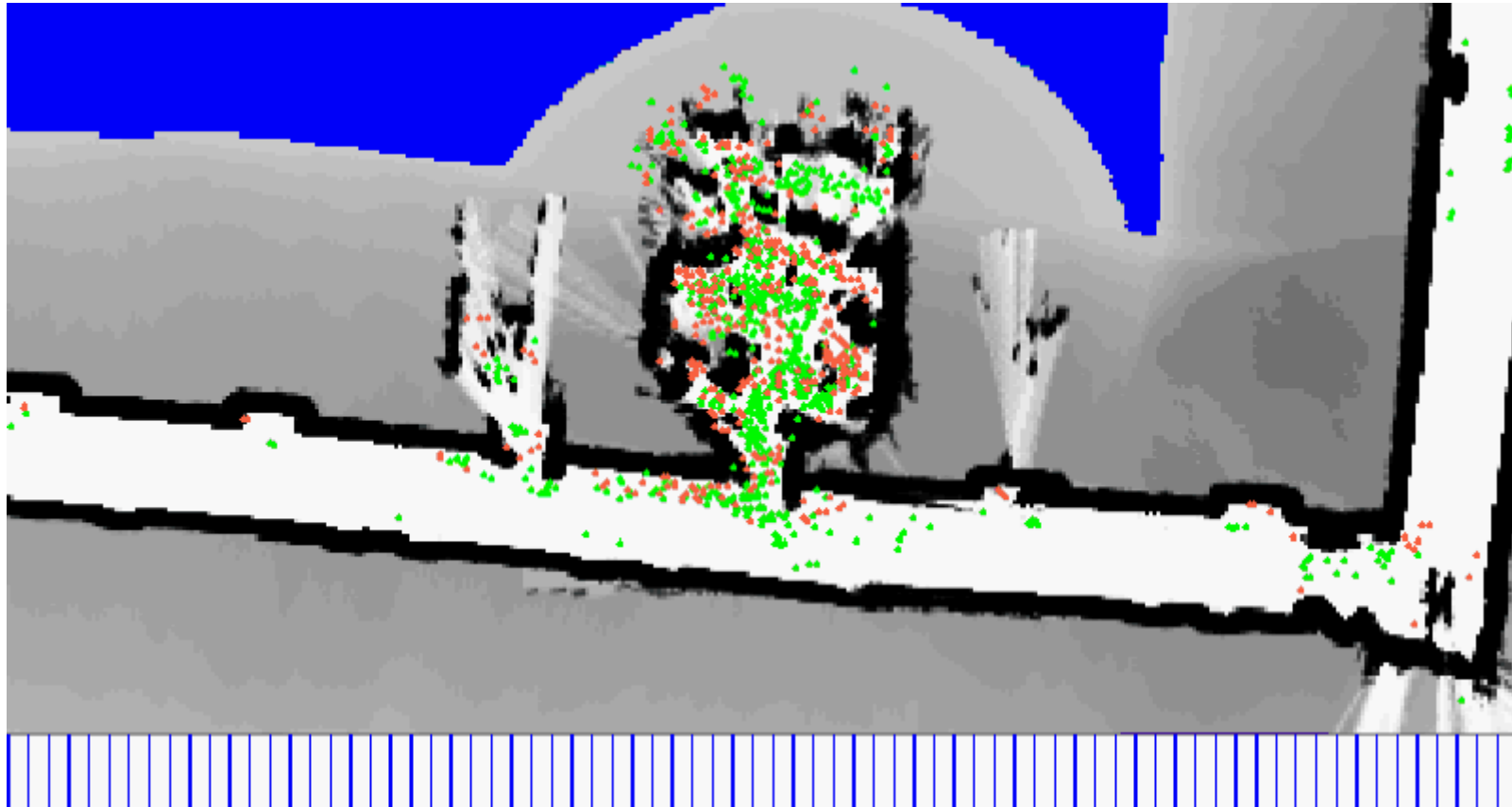
Particle Filters (2)



Particles = Robustness



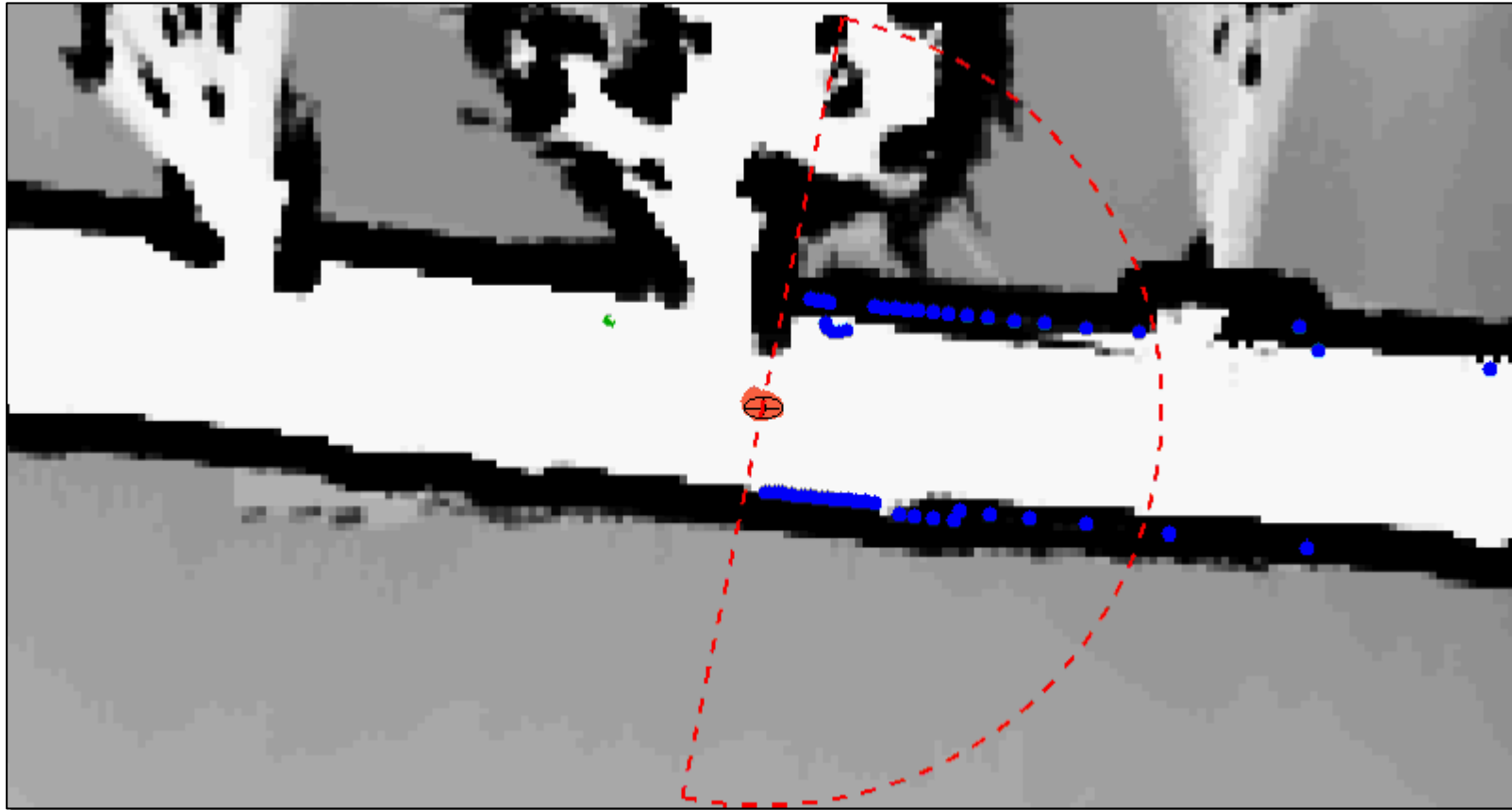
Tracking People from Moving Platform



- robot location (particles)
- people location (particles)
- laser measurements (wall)

With Michael Montemerlo

Tracking People from Moving Platform



- robot location (particles)
- people location (particles)
- laser measurements (wall)

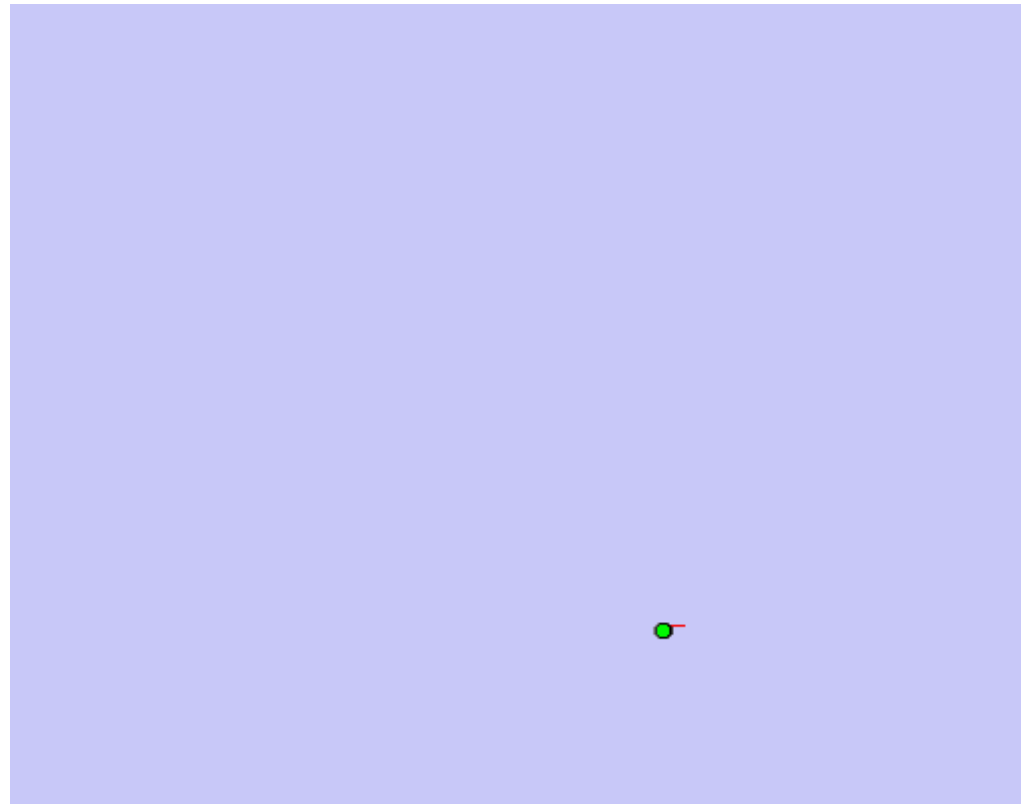
With Michael Montemerlo

Particle Filters for Tracking Cars



With Anya Petrovskaya

Mapping Environments (SFM)



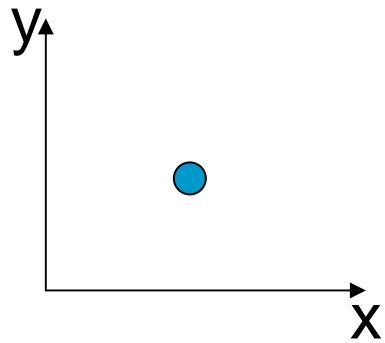
With Dirk Haehnel

Overview

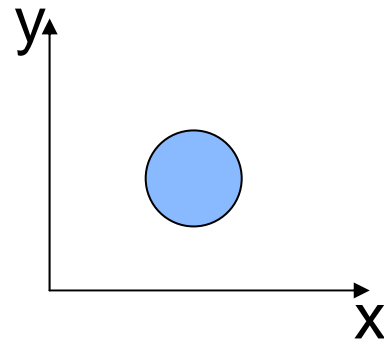
- q The Tracking Problem
- q Bayes Filters
- q Particle Filters
- q Kalman Filters
- q Using Kalman Filters

Tracking with KFs: Gaussians!

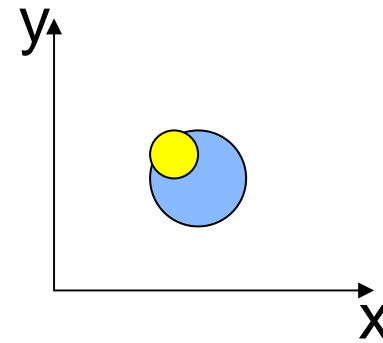
initial estimate



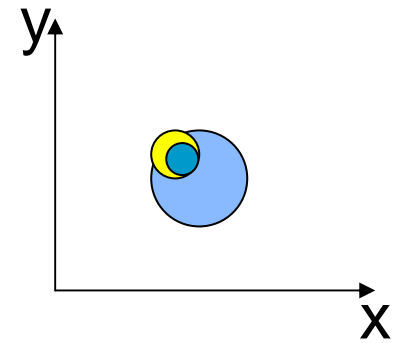
prediction



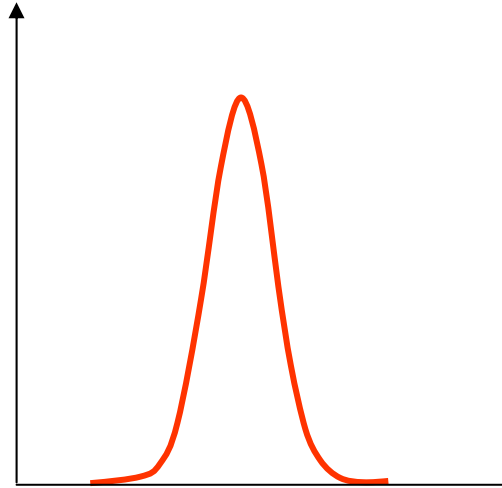
measurement



update



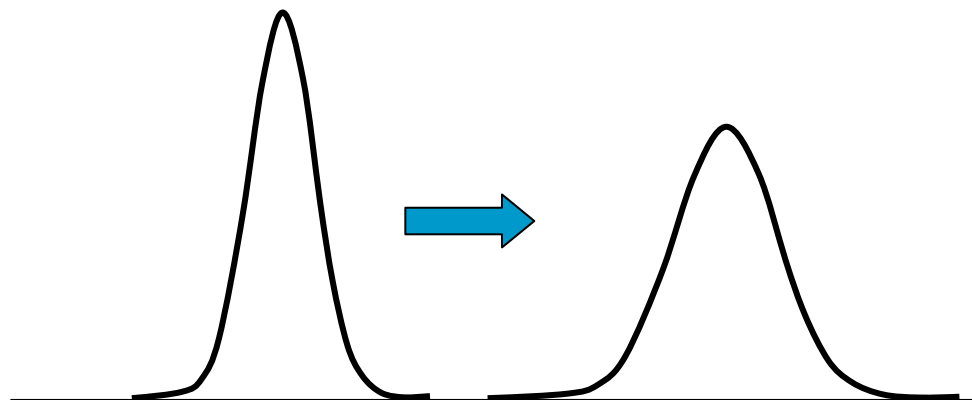
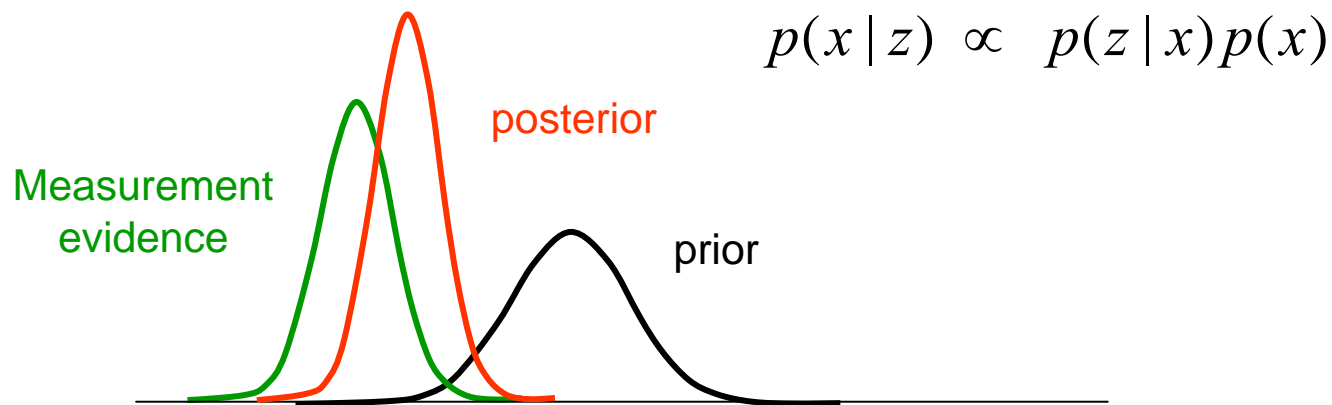
Kalman Filters



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

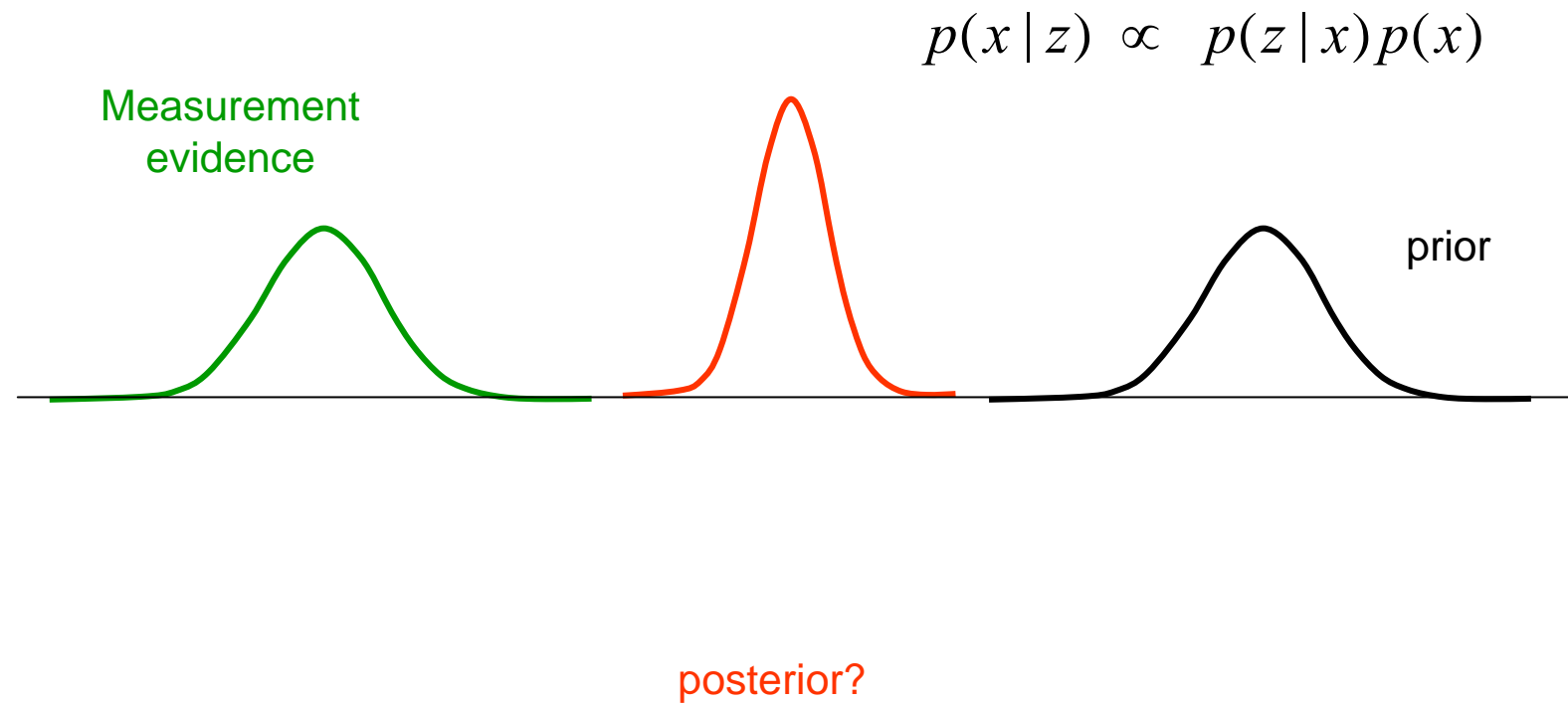
$$p(x) \propto \exp\left\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right\}$$

Kalman Filters



$$p(x') = \int p(x'|x) p(x) dx$$

A Quiz

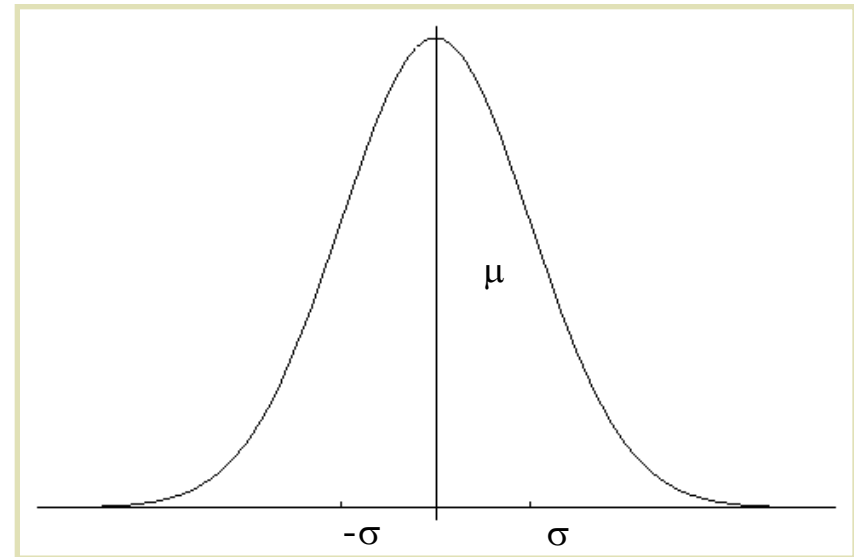


Gaussians

Univariate

$p(x) \sim N(\mu, \sigma^2)$:

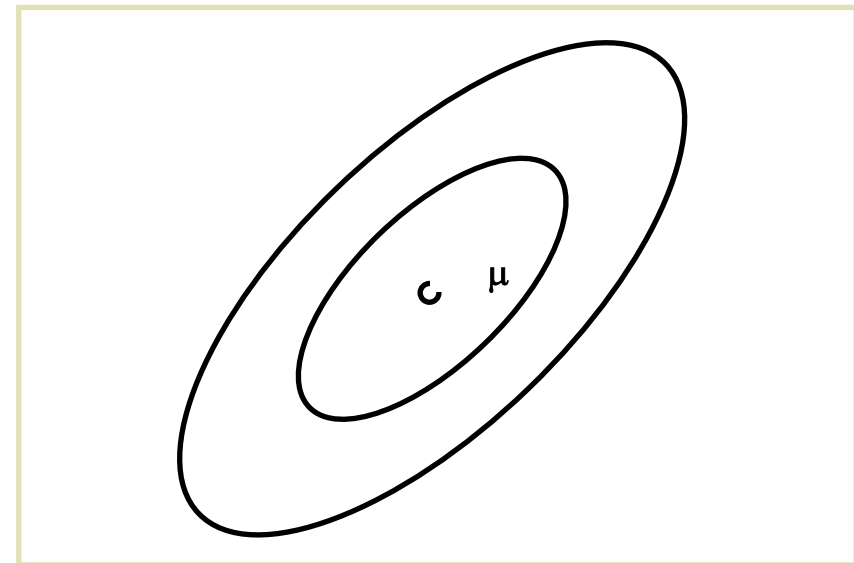
$$p(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2} \frac{(x-\mu)^2}{\sigma^2}}$$



Multivariate

$p(\mathbf{x}) \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2} |\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2} (\mathbf{x}-\boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})}$$



Properties of Univariate Gaussians

$$\left. \begin{array}{l} X \sim N(\mu, \sigma^2) \\ Y = aX + b \end{array} \right\} \Rightarrow Y \sim N(a\mu + b, a^2\sigma^2)$$

$$\left. \begin{array}{l} X_1 \sim N(\mu_1, \sigma_1^2) \\ X_2 \sim N(\mu_2, \sigma_2^2) \end{array} \right\} \Rightarrow p(X_1) \cdot p(X_2) \sim N\left(\frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2} \mu_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \mu_2, \frac{1}{\sigma_1^{-2} + \sigma_2^{-2}}\right)$$

Measurement Update Derived

$$\left. \begin{array}{l} X_1 \sim N(\mu_1, \sigma_1^2) \\ X_2 \sim N(\mu_2, \sigma_2^2) \end{array} \right\} \Rightarrow p(X_1) \cdot p(X_2) = \text{const.} \cdot \exp \left\{ -\frac{1}{2} \frac{(x - \mu_1)^2}{\sigma_1^2} - \frac{1}{2} \frac{(x - \mu_2)^2}{\sigma_2^2} \right\}$$

$$\frac{\partial}{\partial x} \left\{ -\frac{1}{2} \frac{(x - \mu_1)^2}{\sigma_1^2} - \frac{1}{2} \frac{(x - \mu_2)^2}{\sigma_2^2} \right\} = \frac{x - \mu_1}{\sigma_1^2} + \frac{x - \mu_2}{\sigma_2^2} = 0 \quad (\text{for new } \mu)$$

$$(\mu - \mu_1)\sigma_2^2 + (\mu - \mu_2)\sigma_1^2 = 0$$

$$\mu(\sigma_1^2 + \sigma_2^2) = \mu_1\sigma_2^2 + \mu_2\sigma_1^2$$

$$\mu = \frac{\mu_1\sigma_2^2 + \mu_2\sigma_1^2}{\sigma_1^2 + \sigma_2^2}$$

$$\frac{\partial^2}{\partial x^2} \left\{ -\frac{1}{2} \frac{(x - \mu_1)^2}{\sigma_1^2} - \frac{1}{2} \frac{(x - \mu_2)^2}{\sigma_2^2} \right\} = \sigma_1^{-2} + \sigma_2^{-2}$$

$$\sigma = \frac{1}{\sigma_1^{-2} + \sigma_2^{-2}}$$

Properties Multivariate Gaussians

Essentially the same as in the 1-D case, but with more general notation

$$\left. \begin{array}{l} X \sim N(\mu, \Sigma) \\ Y = AX + B \end{array} \right\} \Rightarrow Y \sim N(A\mu + B, A\Sigma A^T)$$

$$\left. \begin{array}{l} X_1 \sim N(\mu_1, \Sigma_1) \\ X_2 \sim N(\mu_2, \Sigma_2) \end{array} \right\} \Rightarrow p(X_1) \cdot p(X_2) \sim N\left((\Sigma_1 + \Sigma_2)^{-1} \Sigma_2 \mu_1 + (\Sigma_1 + \Sigma_2)^{-1} \Sigma_1 \mu_2, (\Sigma_1^{-1} + \Sigma_2^{-1})^{-1}\right)$$

- We stay in the “Gaussian world” as long as we start with Gaussians and perform only linear transformations.

Linear Kalman Filter

Estimates the state x of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t$$

with a measurement

$$z_t = C_t x_t + \delta_t$$

Components of a Kalman Filter

- A_t Matrix ($n \times n$) that describes how the state evolves from t to $t+1$ without controls or noise.
- B_t Matrix ($n \times i$) that describes how the control u_t changes the state from t to $t+1$.
- C_t Matrix ($k \times n$) that describes how to map the state x_t to an observation z_t .
- ε_t Random variables representing the process and measurement noise that are assumed to be independent and normally distributed with covariance R_t and Q_t respectively.
- δ_t

Kalman Filter Algorithm

1. Algorithm **Kalman_filter**($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$):

2. Prediction:

3. $\mu_t = A_t \mu_{t-1} + B_t u_t$

4. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$

5. Correction:

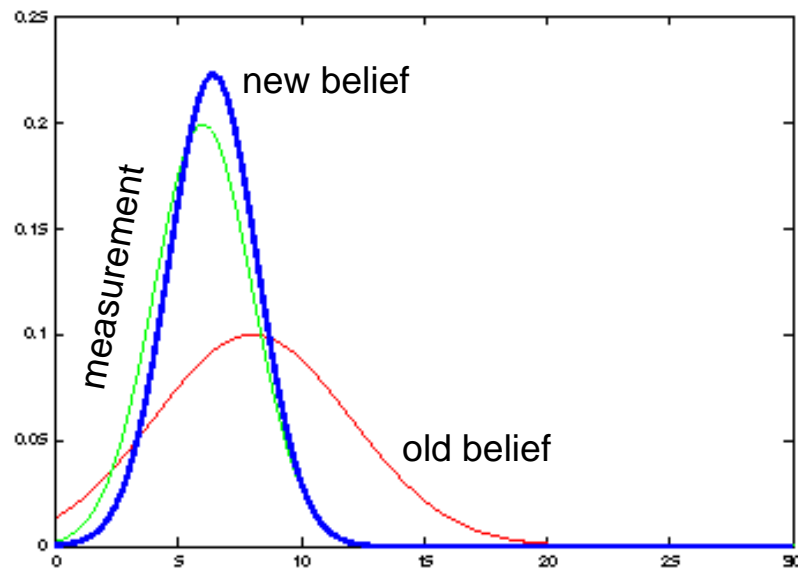
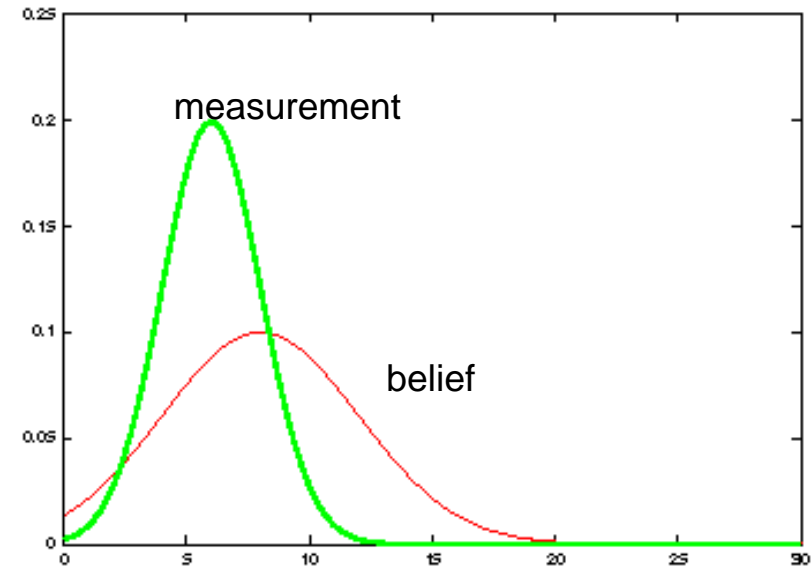
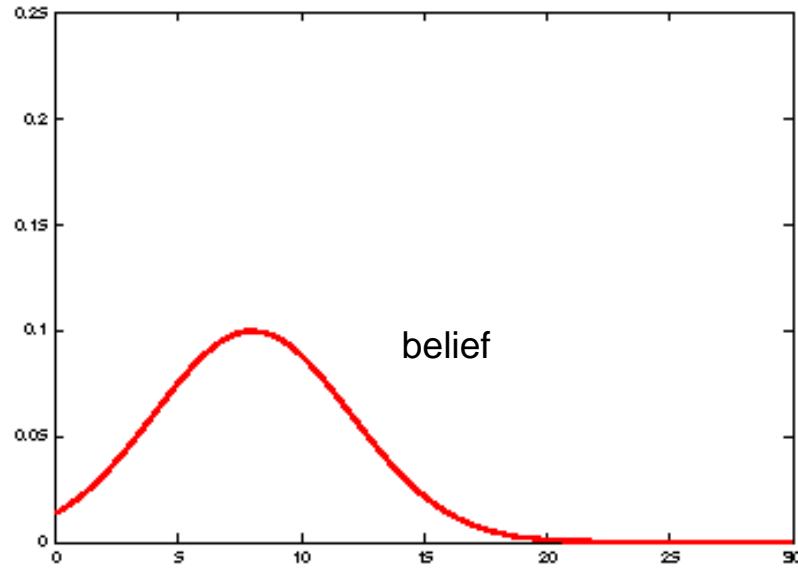
6. $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$

7. $\mu_t = \mu_t + K_t (z_t - C_t \mu_t)$

8. $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$

9. Return μ_t, Σ_t

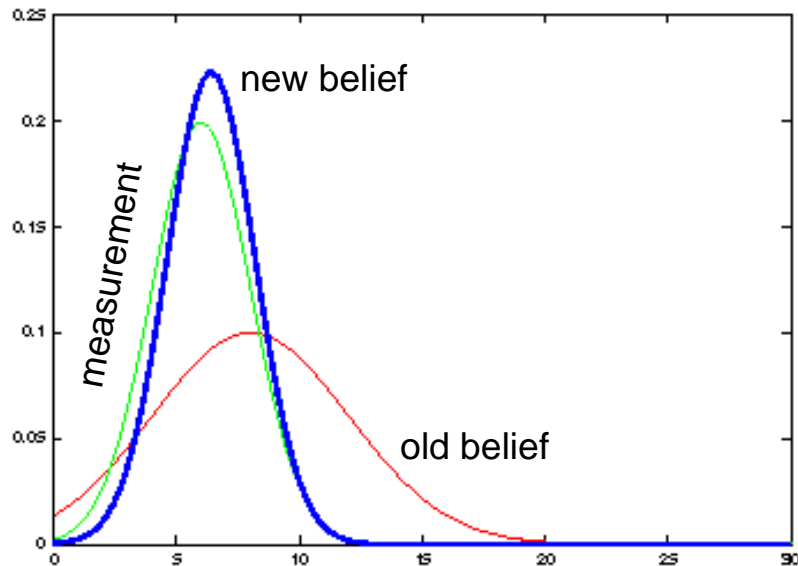
Kalman Filter Updates in 1D



Kalman Filter Updates in 1D

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2 \end{cases} \quad \text{with} \quad K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obs,t}^2}$$

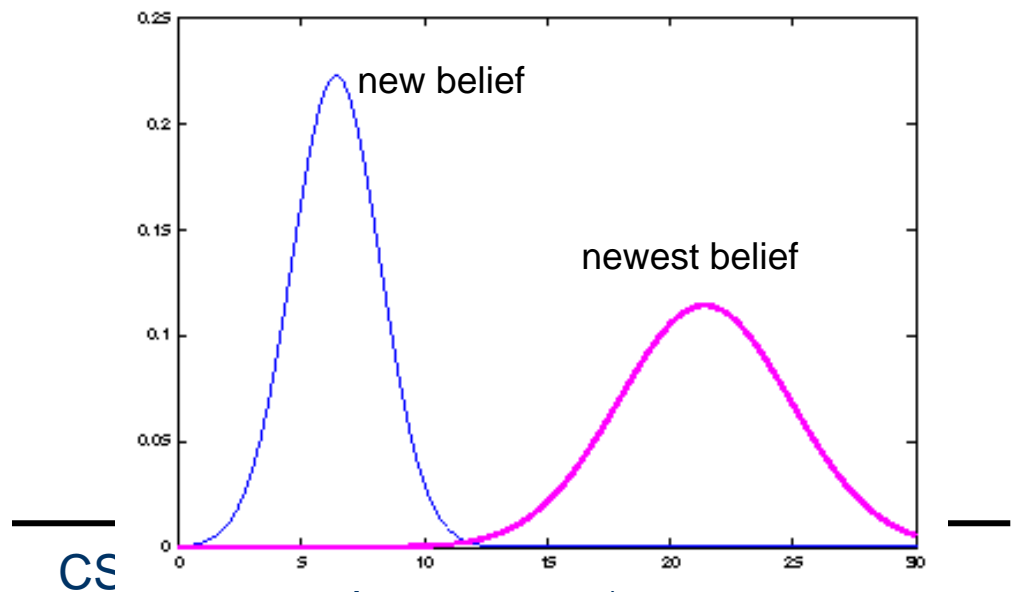
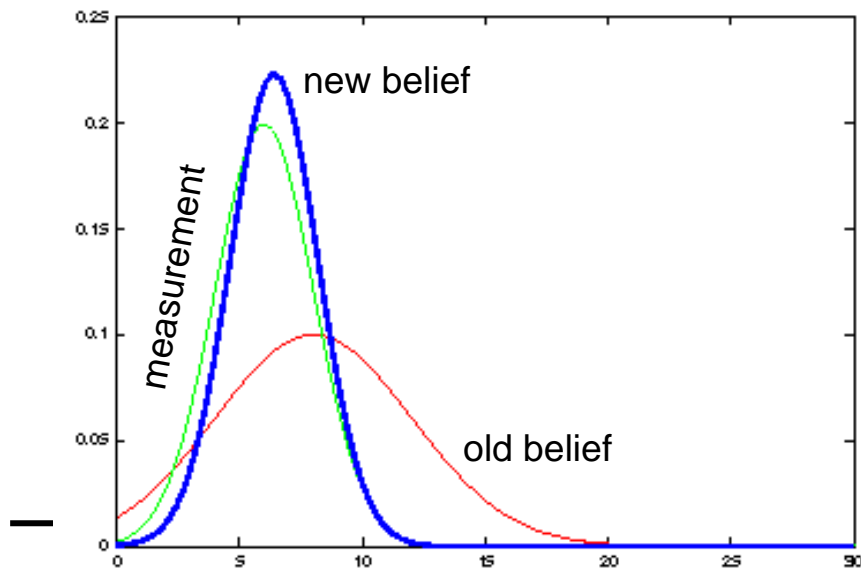
$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t\bar{\mu}_t) \\ \Sigma_t = (I - K_tC_t)\bar{\Sigma}_t \end{cases} \quad \text{with} \quad K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$



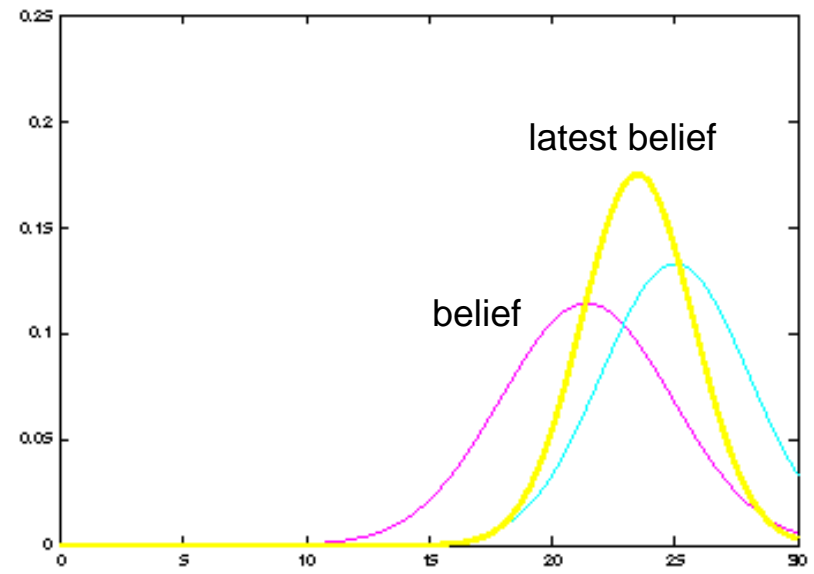
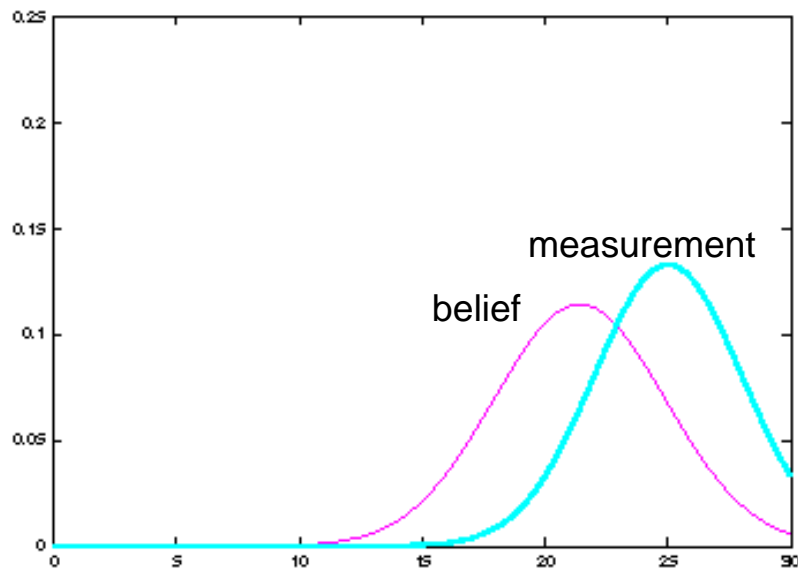
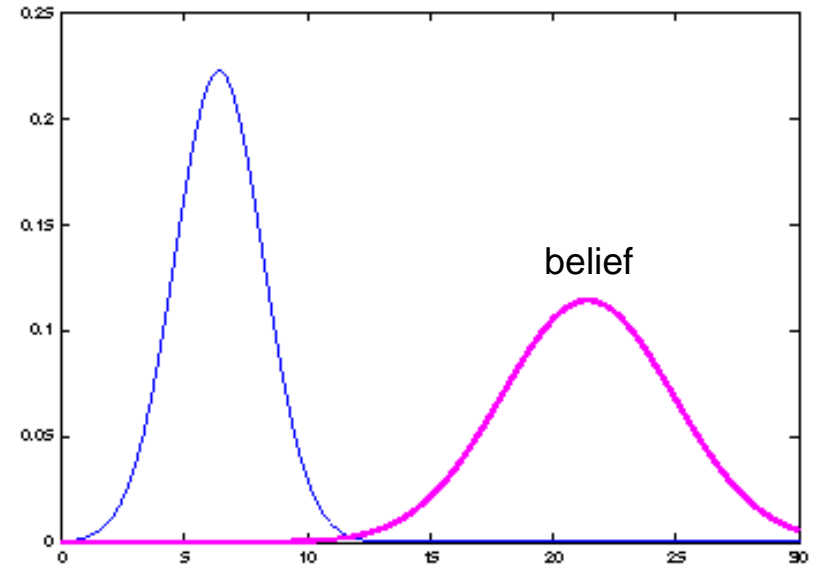
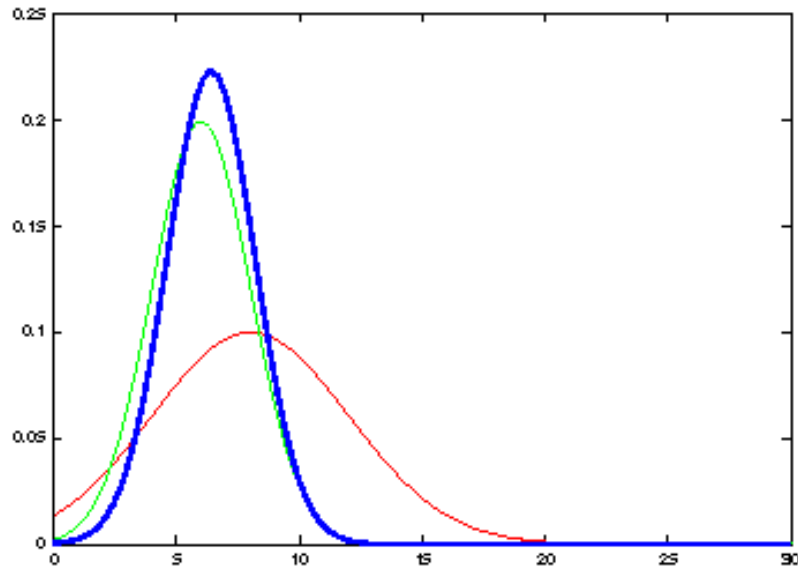
Kalman Filter Updates in 1D

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t \mu_{t-1} + b_t u_t \\ \bar{\sigma}_t^2 = a_t^2 \sigma_{t-1}^2 + \sigma_{act,t}^2 \end{cases}$$

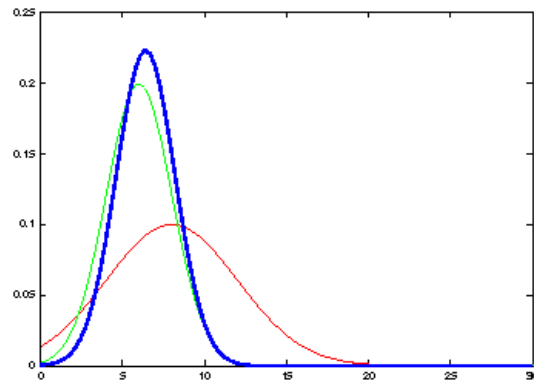
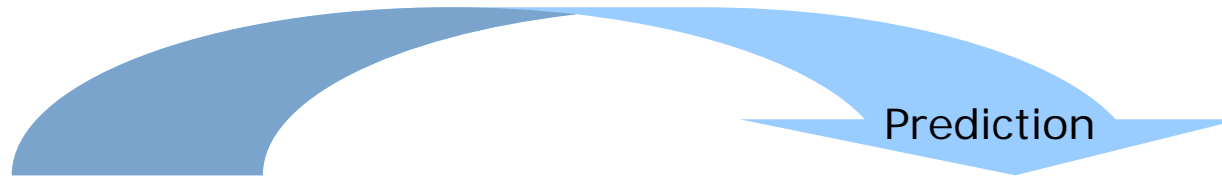
$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \end{cases}$$



Kalman Filter Updates

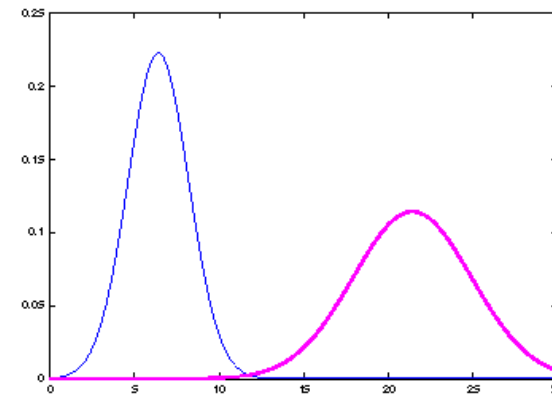


The Prediction-Correction-Cycle

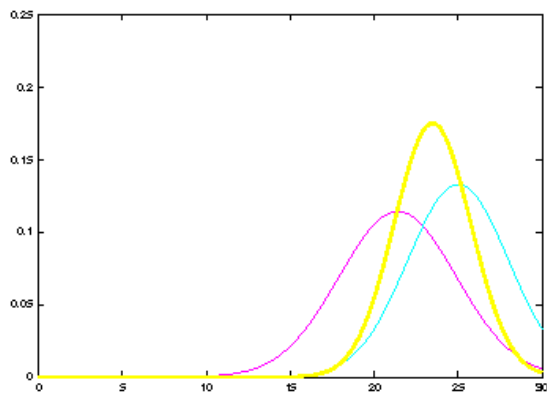


$$\overline{bel}(x_t) = \begin{cases} \overline{\mu}_t = a_t \mu_{t-1} + b_t u_t \\ \overline{\sigma}_t^2 = a_t^2 \sigma_{t-1}^2 + \sigma_{act,t}^2 \end{cases}$$

$$\overline{bel}(x_t) = \begin{cases} \overline{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \overline{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \end{cases}$$

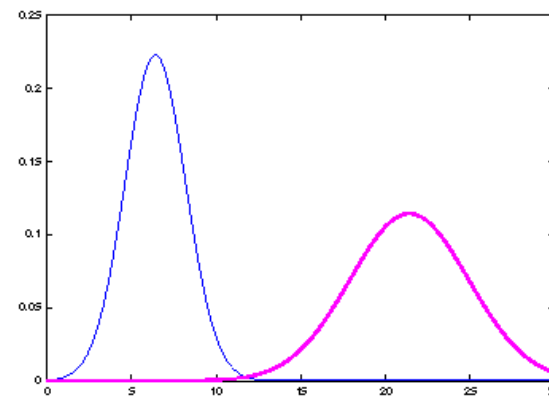


The Prediction-Correction-Cycle



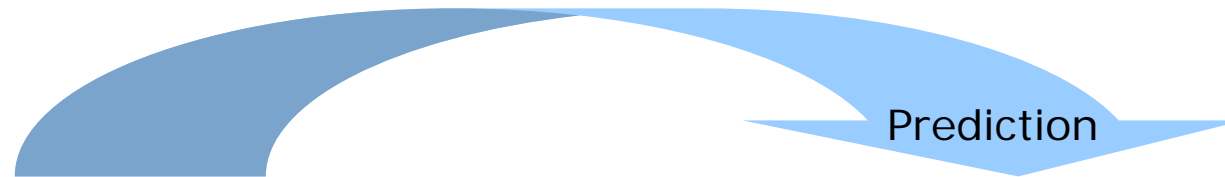
$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2 \end{cases}, K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obs,t}^2}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t\bar{\mu}_t) \\ \Sigma_t = (I - K_tC_t)\bar{\Sigma}_t \end{cases}, K_t = \bar{\Sigma}_tC_t^T(C_t\bar{\Sigma}_tC_t^T + Q_t)^{-1}$$



Correction

The Prediction-Correction-Cycle



$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2 \end{cases}, K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obs,t}^2}$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t \mu_{t-1} + b_t u_t \\ \bar{\sigma}_t^2 = a_t^2 \sigma_{t-1}^2 + \sigma_{act,t}^2 \end{cases}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t) \\ \Sigma_t = (I - K_t C_t) \bar{\Sigma}_t \end{cases}, K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \end{cases}$$



Kalman Filter Summary

- q *Highly efficient*: Polynomial in measurement dimensionality k and state dimensionality n :
 $O(k^{2.376} + n^2)$
- q *Optimal for linear Gaussian systems!*
- q Most robotics systems are *nonlinear!*

Overview

- q The Tracking Problem
- q Bayes Filters
- q Particle Filters
- q Kalman Filters
- q Using Kalman Filters

Let's Apply KFs to Tracking Problem

Image 1

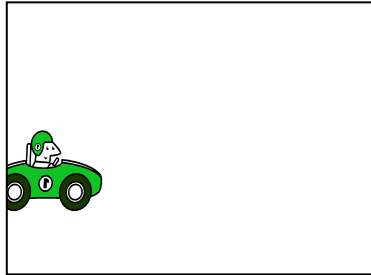


Image 2

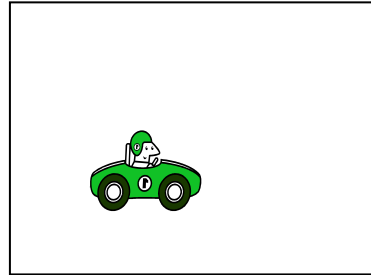


Image 3

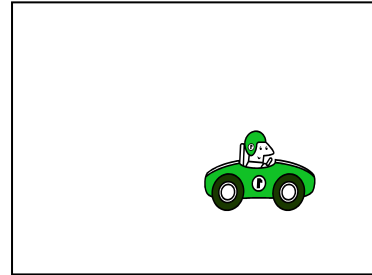
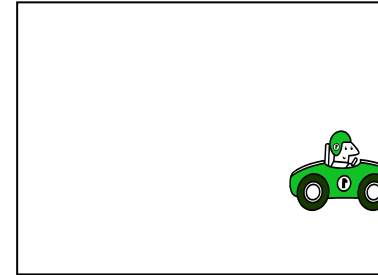


Image 4



Kalman Filter with 2-Dim Linear Model

Linear Change (Motion)

$$x' = Ax + B + N(0, R)$$

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$B = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad R = \begin{pmatrix} \rho^2 & 0 \\ 0 & \rho^2 \end{pmatrix}$$

What is C, D, R?

Linear Measurement Model

$$z = Cx + D + N(0, Q)$$

$$C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$D = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad Q = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{pmatrix}$$

What is A, B, Q?

Kalman Filter Algorithm

1. Algorithm **Kalman_filter**($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$):

2. Prediction:

3. $\mu_t = A_t \mu_{t-1} + B_t u_t$

4. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad B = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad R = \begin{pmatrix} \rho^2 & 0 \\ 0 & \rho^2 \end{pmatrix}$$

5. Correction:

6. $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$

7. $\mu_t = \mu_t + K_t (z_t - C_t \mu_t - D)$

8. $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$

9. Return μ_t, Σ_t

$$C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad D = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad Q = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \sigma^2 \end{pmatrix}$$

Kalman Filter in Detail

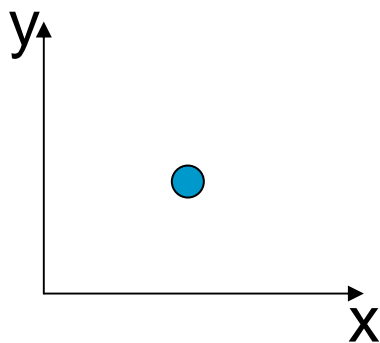
q Measurements $z = x + N(0, R)$

q Change $x' = x + N(0, Q)$

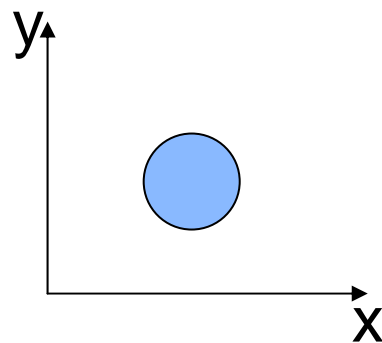
q Prediction $\Sigma' = \Sigma + Q$ $\mu' = \mu$

q Measurement Update $\Sigma'' = (\Sigma'^{-1} + R^{-1})^{-1}$
 $\mu'' = \mu' + \Sigma'' R^{-1} (z - \mu')$

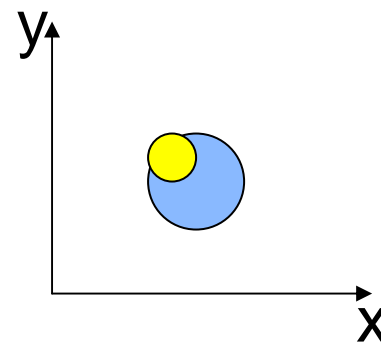
initial position



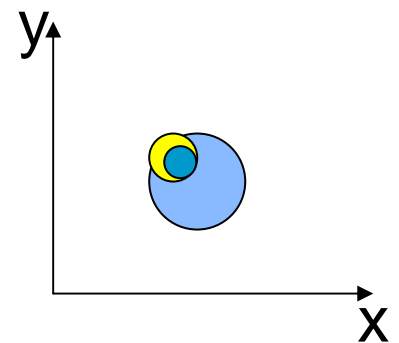
prediction



measurement



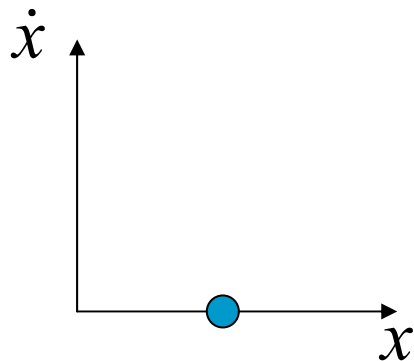
update



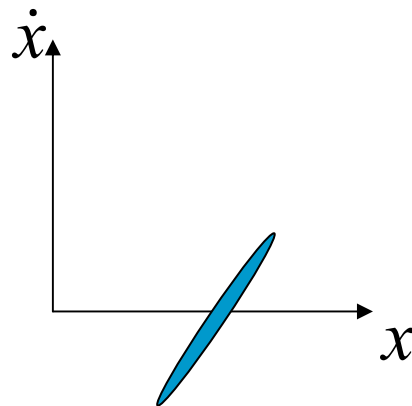
Can We Do Better?

Kalman, Better!

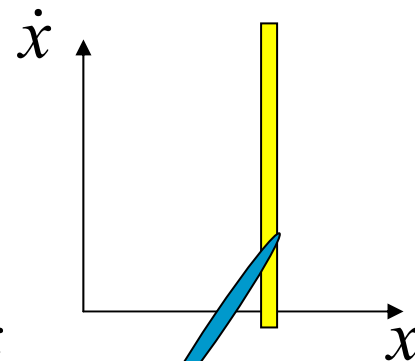
initial position



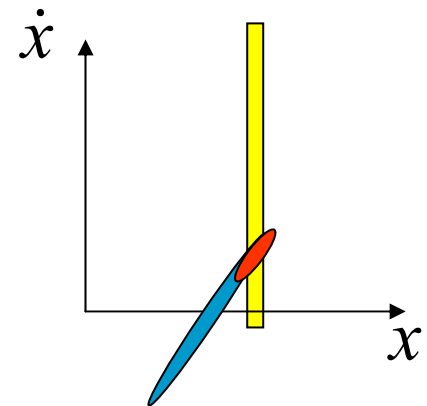
prediction



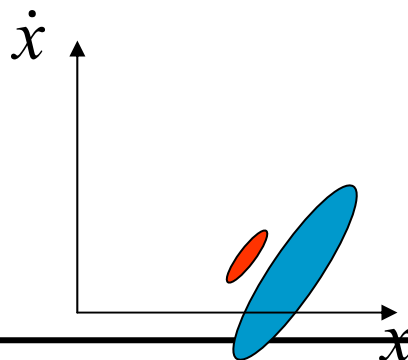
measurement



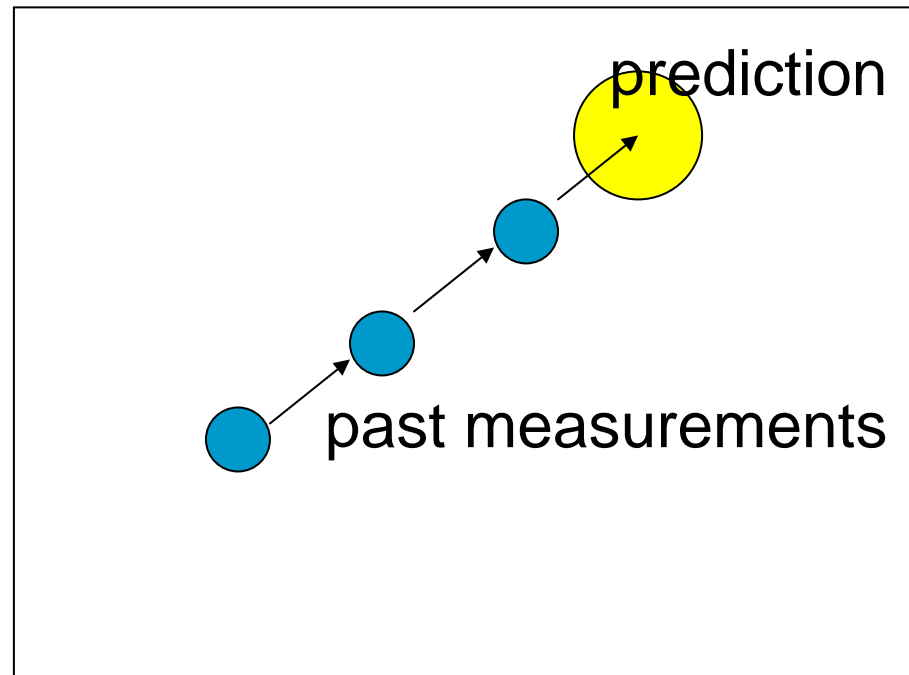
update



next prediction



We Can Estimate Velocity!



Kalman Filter For 2D Tracking

□ Linear Measurement model (now with 4 state variables)

$$\begin{pmatrix} x_{obs} \\ y_{obs} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{pmatrix} + N(0, R) \quad R = \begin{pmatrix} r^2 & 0 \\ 0 & r^2 \end{pmatrix}$$

□ Linear Change

$$\begin{pmatrix} x' \\ y' \\ \dot{x}' \\ \dot{y}' \end{pmatrix} = \begin{pmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{pmatrix} + N(0, Q) \quad Q = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & q^2 & 0 \\ 0 & 0 & 0 & q^2 \end{pmatrix}$$

Putting It Together Again

q Measurements $\vec{z} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \vec{x} + N(0, R)$

q Change $\vec{x}' = \begin{pmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \vec{x} + N(0, Q)$

q Prediction $\Sigma' = \Sigma + CQC^T$
 $\mu' = C\mu$

q Measurement Update $\Sigma' = (\Sigma^{-1} + A^T R^{-1} A)^{-1}$
 $\mu' = \mu + \Sigma' A^T R^{-1} (z - A\mu)$

Summary Kalman Filter

- q Estimates state of a system
 - Position
 - Velocity
 - Many other continuous state variables possible
- q KF maintains
 - Mean vector for the state
 - Covariance matrix of state uncertainty
- q Implements
 - Time update = prediction
 - Measurement update
- q Standard Kalman filter is linear-Gaussian
 - Linear system dynamics, linear sensor model
 - Additive Gaussian noise (independent)
 - Nonlinear extensions: extended KF, unscented KF: linearize
- q More info:
 - CS226
 - Probabilistic Robotics (Thrun/Burgard/Fox, MIT Press)

Summary ~~Kalman~~ Particle Filter

- q Estimates state of a system
 - Position
 - Velocity
 - Many other continuous *and discrete* state variables possible
- q KF maintains
 - ~~Mean vector for the state~~
 - ~~Covariance matrix of state uncertainty~~
 - set of particles (example states)
- q Implements
 - Time update = prediction = *predictive sampling*
 - Measurement update = *resampling, importance weights*
- q ~~Standard Kalman filter is linear-Gaussian~~
 - ~~Linear system dynamics, linear sensor model~~
 - ~~Additive Gaussian noise (independent)~~
 - ~~Nonlinear extensions: extended KF, unscented KF: linearize~~
 - *fully nonlinear*
- q More info:
 - CS226
 - Probabilistic Robotics (Thrun/Burgard/Fox, MIT Press)
 - *easy to implement*

Summary

- q The Tracking Problem
- q Bayes Filters
- q Particle Filters
- q Kalman Filters
- q Using Kalman Filters