Particle Filter

Sensor Fusion

Fredrik Gustafsson
fredrik.gustafsson@liu.se

Gustaf Hendeby
gustaf.hendeby@liu.se

Linköping University

Purpose

To explain the basic particle filter and its implementation

- The Bayesian optimal filter revisited.
- The point-mass filter ($\sim$ 1970) requires adaptive grid and scales badly with state dimension and has quadratic complexity in the number of grid points.
- The particle filter (1993) resolves these issues.
- Numerical examples.

Bayes Optimal Filter: summary

General nonlinear state-space model:

$$
\begin{aligned}
x_{k+1} &= f(x_k, u_k, v_k) & \qquad x_k | x_{k-1} &\sim p(x_k | x_{k-1}) \\
y_k &= h(x_k, u_k, e_k) & \qquad y_k | x_k &\sim p(y_k | x_k)
\end{aligned}
$$

General Bayesian recursion (time and measurement updates)

$$
p(x_{k+1} | y_{1:k}) = \int p(x_{k+1} | x_k) p(x_k | y_{1:k}) \, dx_k,
$$

$$
p(x_k | y_{1:k}) = \frac{p(y_k | x_k) p(x_k | y_{1:k-1})}{p(y_k | y_{1:k-1})}.
$$

- Analytic solution available in a few special cases (KF)
- Key idea: for a given trajectory $x_{1:k}$, the recursion can be computed.
- PF: evaluate trajectories on a *random grid* of the state space

Numerical Approximation

Basic idea: (same as PMF) postulate a discrete approximation of the posterior. For the predictive density, we have

$$\hat{p}(x_k|y_{1:k-1}) = \sum_{i=1}^{N} w_{k|k-1}^{(i)} \delta(x_k - x_k^{(i)}).$$

The first moments (mean and covariance) are simple to compute from this approximation:

$$\hat{x}_{k|k-1} = \mathrm{E}(x_k) = \sum_{i=1}^{N} w_{k|k-1}^{(i)} x_k^{(i)},$$

$$P_{k|k-1} = \mathrm{Cov}(x_k) = \sum_{i=1}^{N} w_{k|k-1}^{(i)} (x_k^{(i)} - \hat{x}_{k|k-1})(x_k^{(i)} - \hat{x}_{k|k-1})^T.$$

Also, the MAP estimate can be useful:

$$\hat{x}_{k|k-1}^{\mathrm{map}} = \arg\max_{x_k^{(i)}} \hat{p}(x_k|y_{1:k-1}).$$

Measurement Update

The measurement (same as PMF) follows directly, without any extra approximations

$$\hat{p}(x_k|y_{1:k}) = \sum_{i=1}^{N} \underbrace{\frac{1}{c_k} p(y_k|x_k^{(i)}) w_{k|k-1}^{(i)}}_{w_{k|k}^{(i)}} \delta(x_k - x_k^{(i)})$$

$$c_k = \sum_{i=1}^{N} p(y_k|x_k^{(i)}) w_{k|k-1}^{(i)}.$$

The normalization constant c_k corresponds to assuring that $\sum_{i=1}^{N} w_{k|k}^{(i)} = 1$.

Particle Filter

- Trick to avoid quadratic complexity: sample trajectories, not states
- Time update for the weight of a trajectory:

$$p(x_{1:k+1}^{(i)}|y_{1:k}) = \underbrace{p(x_{k+1}^{(i)}|x_{1:k}^{(i)}, y_{1:k})}_{p(x_{k+1}^{(i)}|x_k^{(i)})} \underbrace{p(x_{1:k}^{(i)}|y_{1:k})}_{w_{k|k}^{(i)}}$$
$$= w_{k|k}^{(i)} p(x_{k+1}^{(i)}|x_k^{(i)}) = w_{k+1|k}^{(i)}.$$

- In contrast to the PMF, there is no sum involved here! This avoids the quadratic ($O(N^2)$) complexity of the PMF.
- The new sample is sampled from the prior in the original PF (SIR, or bootstrap, PF)

$$x_{k+1}^{(i)} \sim p(x_{k+1}|x_k^{(i)}).$$

Basic SIR PF Algorithm

Choose the number of particles N.

Initialization: Generate $x_0^{(i)} \sim p_{x_0}, i = 1, \dots, N$ particles.

Iterate for $k = 1, 2, \dots, t$:

1. *Measurement update:* For $k = 1, 2, \dots,$

$$w_k^{(i)} = w_{k-1}^{(i)} p(y_k | x_k^{(i)}).$$

2. *Normalize:* $w_k^{(i)} := w_k^{(i)} / \sum_j w_k^{(j)}$.
3. *Estimation:* MMSE $\hat{x}_k \approx \sum_{i=1}^N w_k^{(i)} x_k^{(i)}$ or MAP.
4. *Resampling:* Bayesian bootstrap: Take N samples with replacement from the set $\{x_k^{(i)}\}_{i=1}^N$ where the probability to take sample i is $w_k^{(i)}$. Let $w_k^{(i)} = 1/N$.
5. *Prediction:* Generate random process noise samples

$$v_k^{(i)} \sim p_{v_k}, \quad x_{k+1}^{(i)} = f(x_k^{(i)}, v_k^{(i)}).$$

PF Code

Input arguments: NL object m, SIG object z.

Output arguments: SIG object zhat.

```
y = z.y.';
u = z.u.';
xp = m.x0.' + rand(m.px0, Np);  % Initialization
for k = 1:N
   % Time update
   v = rand(m.pv, Np);  % Random process noise
   xp = m.f(k, xp.', u(:,k), m.th).' + v;  % State prediction
   % Measurement update
   yp = m.h(k, xp.', u(:, k).', m.th).';  % Measurement prediction
   w = pdf(m.pe, y(:,k).'-yp);  % Likelihood
   xhat(k,:) = mean(w(:).*xp);  % Estimation
   [xp, w] = resample(xp, w);  % Resampling
   xMC(:, k, :) = xp;  % MC uncertainty repr.
end
```

Example: 1D terrain navigation

Problem: Measured velocity u_k, unknown velocity disturbance v_k, known altitude profile $h(x)$ which is observed with y_k.

Model:

$$x_{k+1} = x_k + u_k + v_k,$$
$$y_k = h(x_k) + e_k,$$

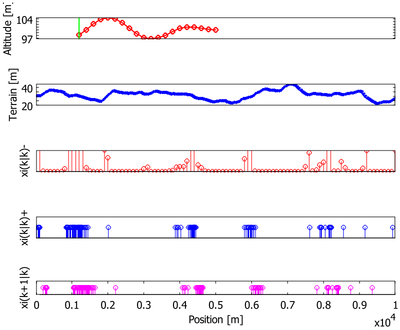
y_1

$h(x)$

Step 1. MUP $p(x_1|y_1)$

Step 4. Resampling $p(x_1|y_1)$

Step 5. TUP $p(x_2|y_1)$



http://youtu.be/thNhOE6tmVO

Example: 1D terrain navigation

Problem: Measured velocity u_k, unknown velocity disturbance v_k, known altitude profile $h(x)$ which is observed with y_k.

Model:

$$x_{k+1} = x_k + u_k + v_k,$$
$$y_k = h(x_k) + e_k,$$

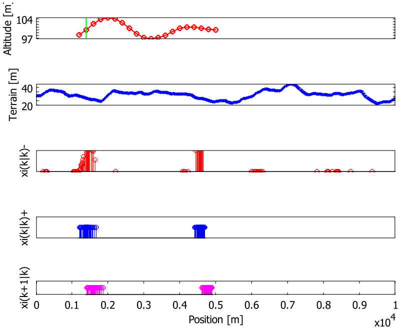
y_1

$h(x)$

Step 1. MUP $p(x_1|y_1)$

Step 4. Resampling $p(x_1|y_1)$

Step 5. TUP $p(x_2|y_1)$



http://youtu.be/thNh0E6tmVO

Example: 1D terrain navigation

Problem: Measured velocity u_k, unknown velocity disturbance v_k, known altitude profile $h(x)$ which is observed with y_k.

Model:

$$x_{k+1} = x_k + u_k + v_k,$$
$$y_k = h(x_k) + e_k,$$

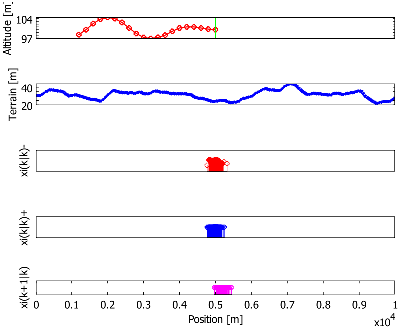
y_1

$h(x)$

Step 1. MUP $p(x_1|y_1)$

Step 4. Resampling $p(x_1|y_1)$

Step 5. TUP $p(x_2|y_1)$



http://youtu.be/thNhOE6tmVO

Example: 1D terrain navigation

Problem: Measured velocity u_k, unknown velocity disturbance v_k, known altitude profile $h(x)$ which is observed with y_k.

Model:

$$x_{k+1} = x_k + u_k + v_k,$$
$$y_k = h(x_k) + e_k,$$

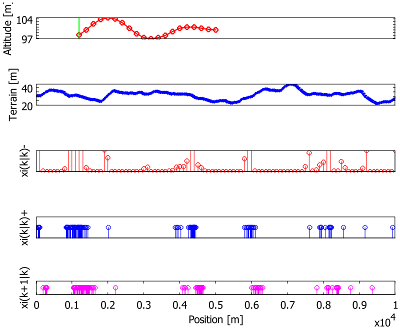
y_1

$h(x)$

Step 1. MUP $p(x_1|y_1)$

Step 4. Resampling $p(x_1|y_1)$

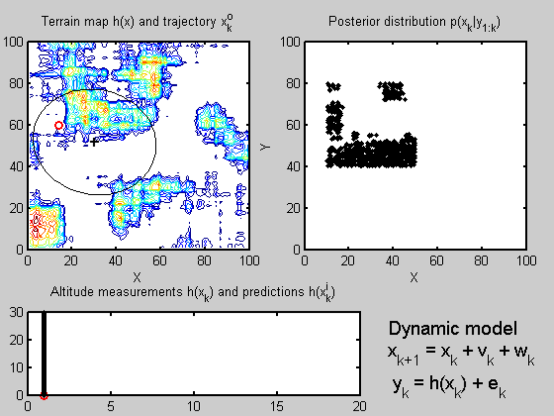
Step 5. TUP $p(x_2|y_1)$



http://youtu.be/thNhOE6tmVO

Example: 2D terrain navigation

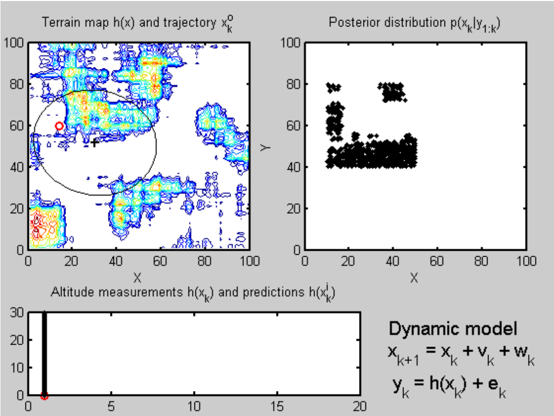
Same assumptions as in 1D: aircraft measures ground altitude as measurement y_k and noisy speed $u_k = v_k + w_k$, terrain elevation map (TAM) provides $h(x_k)$.



http://youtu.be/1uT8U22PmwO

Example: 2D terrain navigation

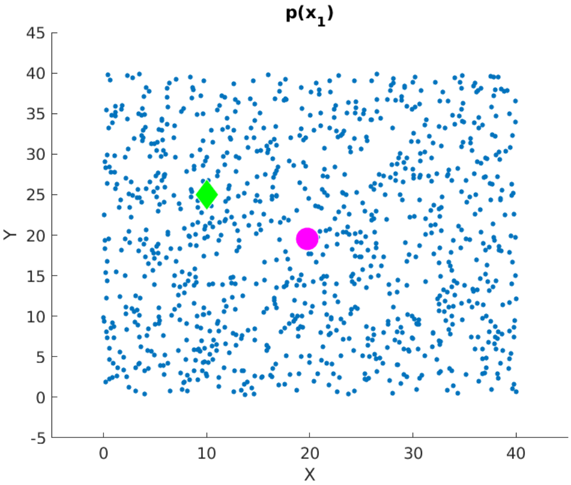
Same assumptions as in 1D: aircraft measures ground altitude as measurement y_k and noisy speed $u_k = v_k + w_k$, terrain elevation map (TAM) provides $h(x_k)$.



http://youtu.be/1uT8U22PmwO

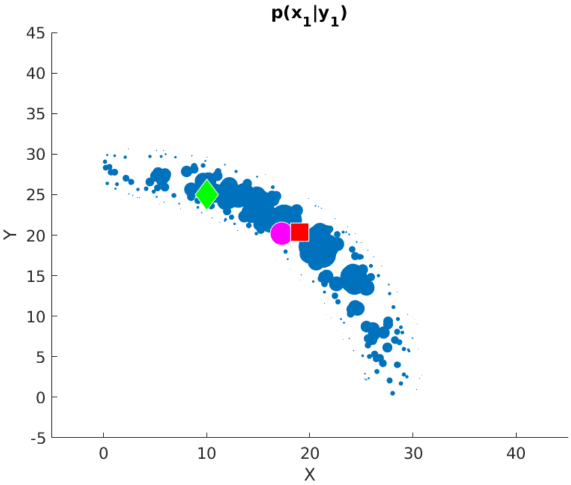
Particle Filter Illustration: radar

- Range bearing measurements
- Constant position motion model (no velocity)
- $R = diag(1, .3)^2$
- $Q = diag(5, 5)$
- Magenta circle: estimate
- Green romb: ground truth
- Red square: measurement



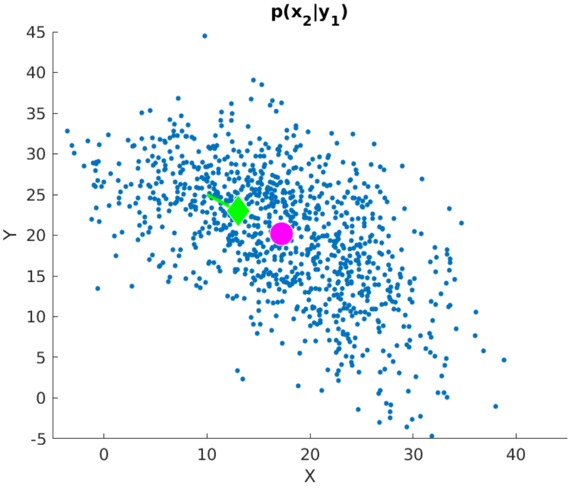
Particle Filter Illustration: radar

- Range bearing measurements
- Constant position motion model (no velocity)
- $R = diag(1, .3)^2$
- $Q = diag(5, 5)$
- Magenta circle: estimate
- Green romb: ground truth
- Red square: measurement



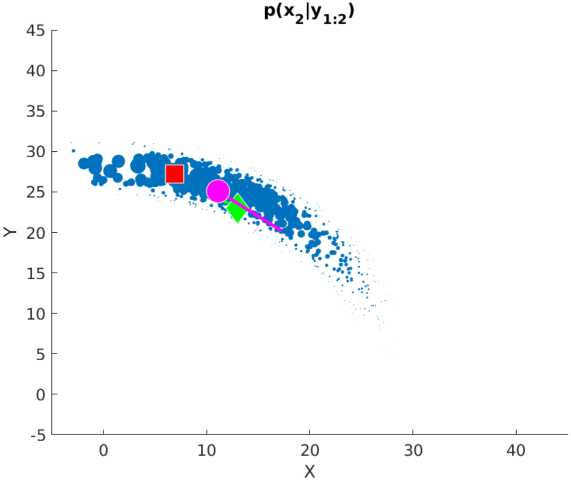
Particle Filter Illustration: radar

- Range bearing measurements
- Constant position motion model (no velocity)
- $R = diag(1, .3)^2$
- $Q = diag(5, 5)$
- Magenta circle: estimate
- Green romb: ground truth
- Red square: measurement



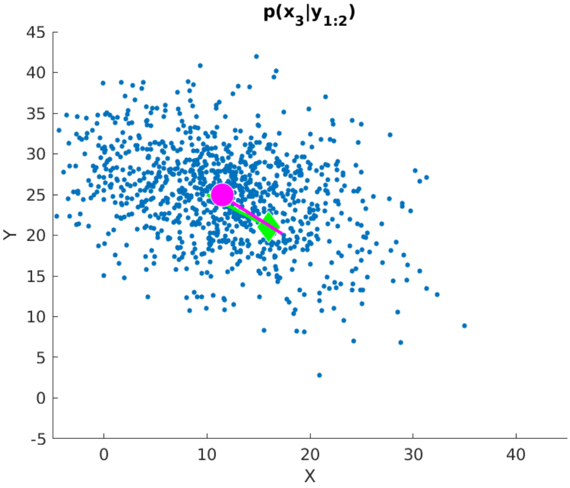
Particle Filter Illustration: radar

- Range bearing measurements
- Constant position motion model (no velocity)
- $R = diag(1, .3)^2$
- $Q = diag(5, 5)$
- Magenta circle: estimate
- Green romb: ground truth
- Red square: measurement



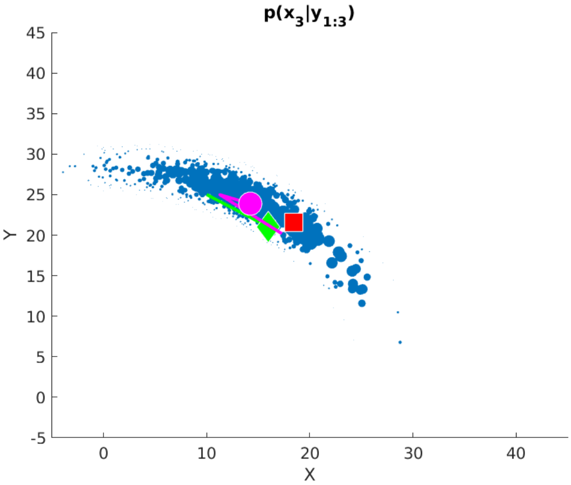
Particle Filter Illustration: radar

- Range bearing measurements
- Constant position motion model (no velocity)
- $R = diag(1, .3)^2$
- $Q = diag(5, 5)$
- Magenta circle: estimate
- Green romb: ground truth
- Red square: measurement



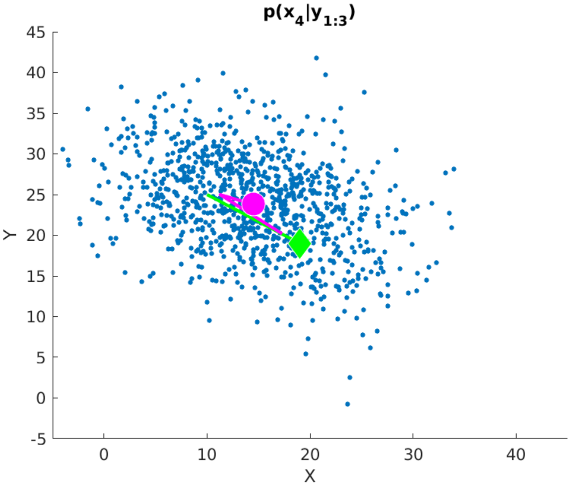
Particle Filter Illustration: radar

- Range bearing measurements
- Constant position motion model (no velocity)
- $R = diag(1, .3)^2$
- $Q = diag(5, 5)$
- Magenta circle: estimate
- Green romb: ground truth
- Red square: measurement



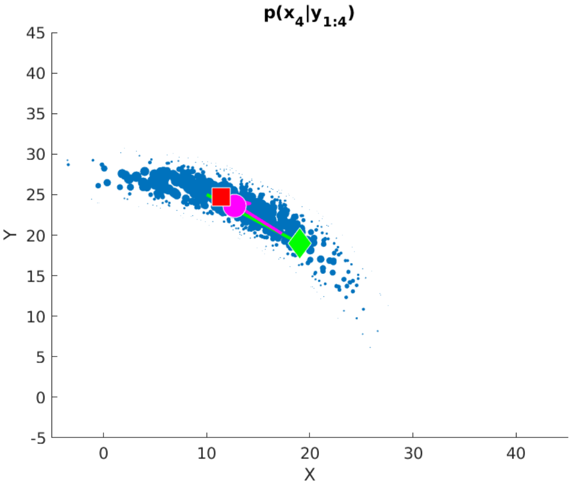
Particle Filter Illustration: radar

- Range bearing measurements
- Constant position motion model (no velocity)
- $R = diag(1, .3)^2$
- $Q = diag(5, 5)$
- Magenta circle: estimate
- Green romb: ground truth
- Red square: measurement



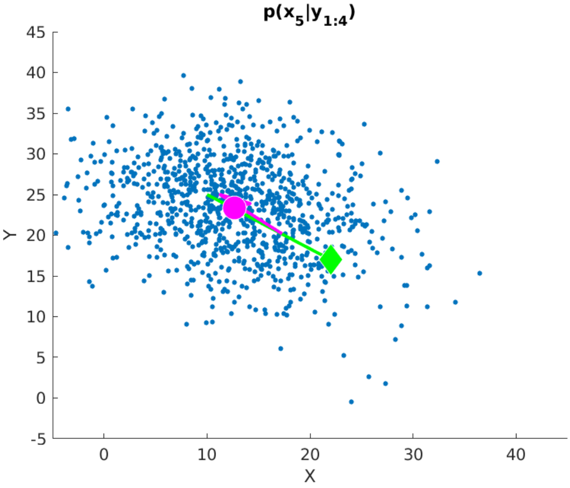
Particle Filter Illustration: radar

- Range bearing measurements
- Constant position motion model (no velocity)
- $R = diag(1, .3)^2$
- $Q = diag(5, 5)$
- Magenta circle: estimate
- Green romb: ground truth
- Red square: measurement



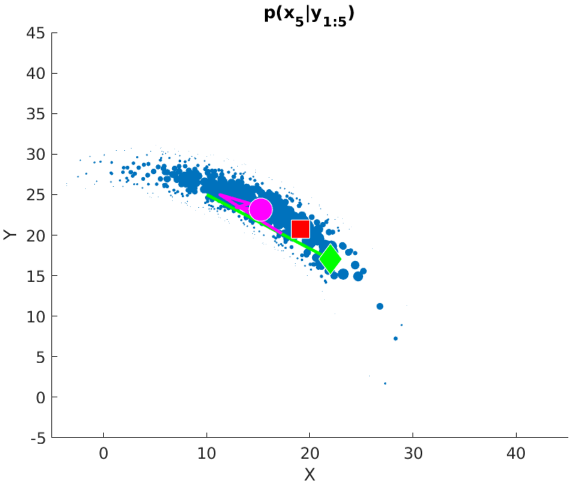
Particle Filter Illustration: radar

- Range bearing measurements
- Constant position motion model (no velocity)
- $R = diag(1, .3)^2$
- $Q = diag(5, 5)$
- Magenta circle: estimate
- Green romb: ground truth
- Red square: measurement



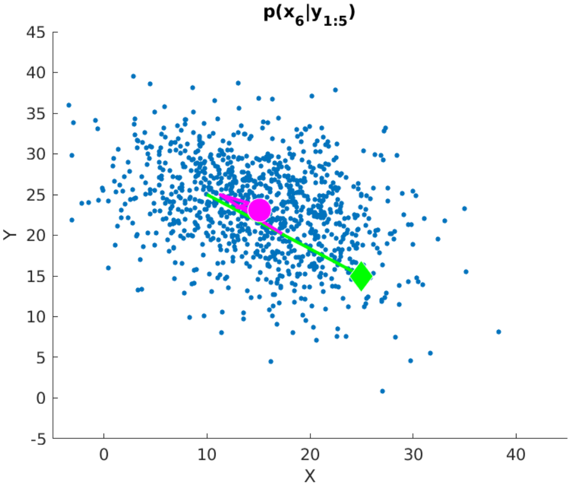
Particle Filter Illustration: radar

- Range bearing measurements
- Constant position motion model (no velocity)
- $R = diag(1, .3)^2$
- $Q = diag(5, 5)$
- Magenta circle: estimate
- Green romb: ground truth
- Red square: measurement



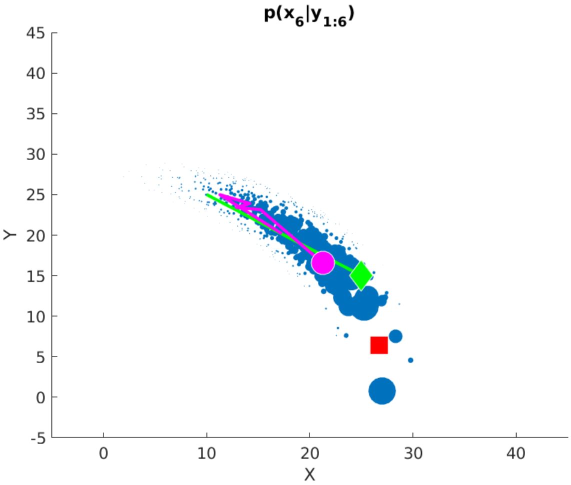
Particle Filter Illustration: radar

- Range bearing measurements
- Constant position motion model (no velocity)
- $R = diag(1, .3)^2$
- $Q = diag(5, 5)$
- Magenta circle: estimate
- Green romb: ground truth
- Red square: measurement



Particle Filter Illustration: radar

- Range bearing measurements
- Constant position motion model (no velocity)
- $R = diag(1, .3)^2$
- $Q = diag(5, 5)$
- Magenta circle: estimate
- Green romb: ground truth
- Red square: measurement



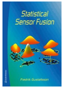
Lecture 7: summary

Basic SIR PF algorithm

Choose N, generate $x_0^{(i)} \sim p_{x_0}$, $i = 1, \dots, N$, particles and iterate for $k = 1, 2, \dots, t$:

1. *Measurement update:* For $k = 1, 2, \dots,$
$$\bar{w}_{k|k}^{(i)} = w_{k|k-1}^{(i)} p(y_k | x_k^{(i)}).$$
2. *Normalize:* $w_{k|k}^{(i)} := \bar{w}_{k|k}^{(i)} / \sum_j \bar{w}_{k|k}^{(j)}$.
3. *Estimation:* MMSE $\hat{x} \approx \sum_{i=1}^{N} w^{(i)} x^{(i)}$ or MAP.
4. *Resampling:* Bayesian bootstrap: Take N samples with replacement from the set $\{x_k^{(i)}\}_{i=1}^{N}$ where the probability to take sample i is $w_{k|k}^{(i)}$ Let $w_{k|k}^{(i)} = 1/N$.
5. *Prediction:* Generate random process noise samples
$$v_k^{(i)} \sim p_{v_k}, \qquad x_{k+1}^{(i)} = f(x_k^{(i)}, v_k^{(i)}) \qquad w_{k+1|k} = w_{k|k}.$$

Main advantages: easy to implement, adaptive random grid, almost linear complexity ($O(N)$) and explores state spaces for $n_x \leq 4$ quite well.



Section 9.3