Robotics

Mobile Robotics

State estimation, Bayes filter, odometry, particle filter, Kalman filter, SLAM, joint Bayes filter, EKF SLAM, particle SLAM, graph-based SLAM

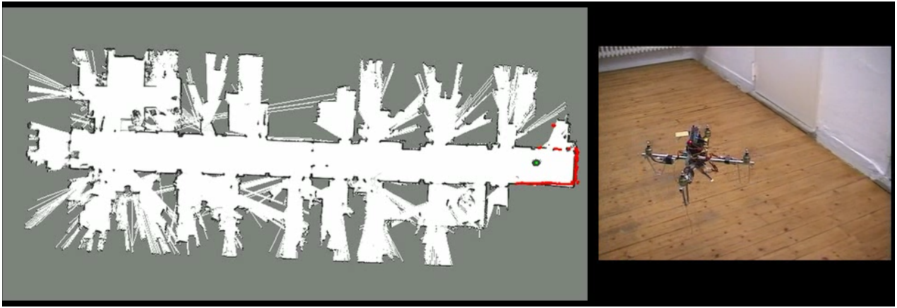
Marc Toussaint
U Stuttgart

`http://www.darpa.mil/grandchallenge05/`

DARPA Grand Challenge 2005

http://www.darpa.mil/grandchallenge/index.asp

DARPA Grand Urban Challenge 2007



http://www.slawomir.de/publications/grzonka09icra/grzonka09icra.pdf

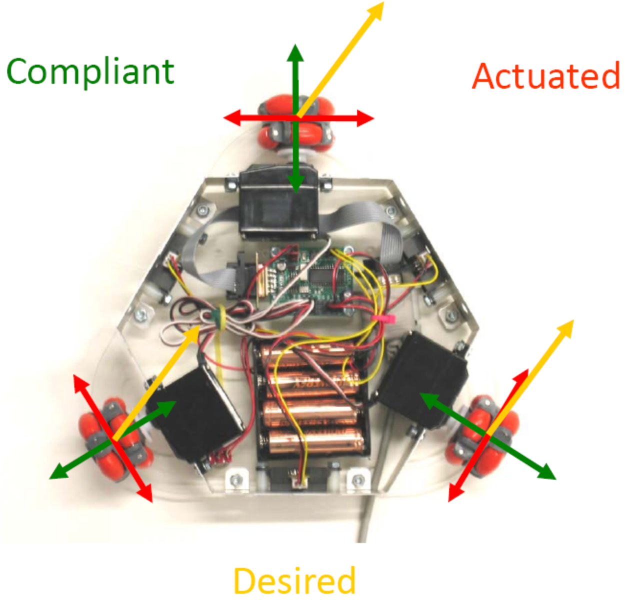
Quadcopter Indoor Localization

http://stair.stanford.edu/multimedia.php

STAIR: STanford Artificial Intelligence Robot

Types of Robot Mobility

Types of Robot Mobility



- Each type of robot mobility corresponds to a

  system equation $x_{t+1} = x_t + \tau f(x_t, u_t)$

  or, if the dynamics are stochastic,

$$P(x_{t+1} \,|\, u_t, x_t) = \mathcal{N}(x_{t+1} \,|\, x_t + \tau f(x_t, u_t), \Sigma)$$

- We considered control, path finding, and trajectory optimization

  For this we always assumed to know the state x_t of the robot (e.g., its posture/position)!

Outline

- PART I:
  A core challenge in mobile robotics is **state estimation**
  $\rightarrow$ Bayesian filtering & smoothing
  particles, Kalman

- PART II:
  Another challenge is to **build a map** while exploring
  $\rightarrow$ SLAM (simultaneous localization and mapping)

PART I: State Estimation Problem

- Our sensory data does not provide sufficient information to determine our location.

- Given the local sensor readings y_t, the current state x_t (location, position) is *uncertain*.

  – which hallway?
  – which door exactly?
  – which heading direction?

State Estimation Problem

- What is the probability of being in front of room 154, given we see what is shown in the image?
- What is the probability given that we were just in front of room 156?
- What is the probability given that we were in front of room 156 and moved 15 meters?

Recall Bayes' theorem

$$P(X|Y) = \frac{P(Y|X)\ P(X)}{P(Y)}$$

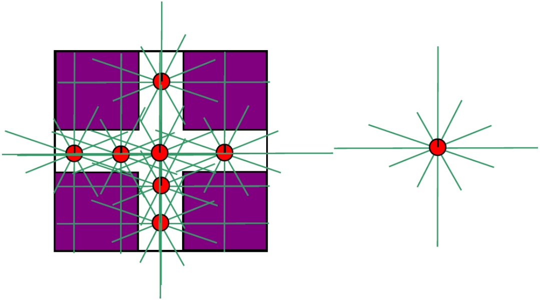
$$\text{posterior} = \frac{\text{likelihood} \cdot \text{prior}}{\text{(normalization)}}$$

- How can we apply this to the State Estimation Problem?

- How can we apply this to the State Estimation Problem?

Using Bayes Rule:

$$P(\text{location} \,|\, \text{sensor}) = \frac{P(\text{sensor} \,|\, \text{location})P(\text{location})}{P(\text{sensor})}$$



Bayes Filter

x_t = state (location) at time t

y_t = sensor readings at time t

$u_{t\text{-}1}$ = control command (action, steering, velocity) at time t-1

- Given the history $y_{0:t}$ and $u_{0:t\text{-}1}$, we want to compute the probability distribution over the state at time t

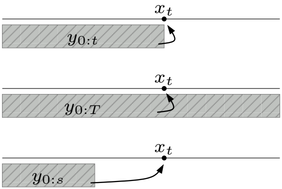
$$p_t(x_t) := P(x_t \mid y_{0:t}, u_{0:t\text{-}1})$$

- Generally:

Filtering: $P(x_t|y_{0:t})$

Smoothing: $P(x_t|y_{0:T})$

Prediction: $P(x_t|y_{0:s})$



Bayes Filter

$$p_t(x_t) := P(x_t \,|\, y_{0:t}, u_{0:t\text{-}1})$$

Bayes Filter

$$\begin{aligned} p_t(x_t) &:= P(x_t \,|\, y_{0:t}, u_{0:t\text{-}1}) \\ &= c_t \; P(y_t \,|\, x_t, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; P(x_t \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \end{aligned}$$

using Bayes rule $P(X|Y,Z) = c \; P(Y|X,Z) \; P(X|Z)$ with some normalization constant c_t

Bayes Filter

$$
\begin{aligned}
p_t(x_t) &:= P(x_t \,|\, y_{0:t}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \,|\, x_t, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; P(x_t \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \,|\, x_t) \; P(x_t \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1})
\end{aligned}
$$

uses conditional independence of the observation on past observations and controls

Bayes Filter

$$
\begin{aligned}
p_t(x_t) &:= P(x_t \,|\, y_{0:t}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \,|\, x_t, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; P(x_t \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \,|\, x_t) \; P(x_t \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \,|\, x_t) \; \int_{x_{t\text{-}1}} P(x_t, x_{t\text{-}1} \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; dx_{t\text{-}1}
\end{aligned}
$$

by definition of the marginal

Bayes Filter

$$
\begin{aligned}
p_t(x_t) &:= P(x_t \,|\, y_{0:t}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \,|\, x_t, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; P(x_t \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \,|\, x_t) \; P(x_t \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \,|\, x_t) \; \int_{x_{t\text{-}1}} P(x_t, x_{t\text{-}1} \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; dx_{t\text{-}1} \\
&= c_t \; P(y_t \,|\, x_t) \; \int_{x_{t\text{-}1}} P(x_t \,|\, x_{t\text{-}1}, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; P(x_{t\text{-}1} \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; dx_{t\text{-}1}
\end{aligned}
$$

by definition of a conditional

Bayes Filter

$$
\begin{aligned}
p_t(x_t) &:= P(x_t \,|\, y_{0:t}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \,|\, x_t, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; P(x_t \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \,|\, x_t) \; P(x_t \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \,|\, x_t) \; \int_{x_{t\text{-}1}} P(x_t, x_{t\text{-}1} \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; dx_{t\text{-}1} \\
&= c_t \; P(y_t \,|\, x_t) \; \int_{x_{t\text{-}1}} P(x_t \,|\, x_{t\text{-}1}, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; P(x_{t\text{-}1} \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; dx_{t\text{-}1} \\
&= c_t \; P(y_t \,|\, x_t) \; \int_{x_{t\text{-}1}} P(x_t \,|\, x_{t\text{-}1}, u_{t\text{-}1}) \; P(x_{t\text{-}1} \,|\, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; dx_{t\text{-}1}
\end{aligned}
$$

given $x_{t\text{-}1}$, x_t depends only on the controls $u_{t\text{-}1}$ (Markov Property)

Bayes Filter

$$
\begin{aligned}
p_t(x_t) &:= P(x_t \mid y_{0:t}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \mid x_t, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; P(x_t \mid y_{0:t\text{-}1}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \mid x_t) \; P(x_t \mid y_{0:t\text{-}1}, u_{0:t\text{-}1}) \\
&= c_t \; P(y_t \mid x_t) \; \int_{x_{t\text{-}1}} P(x_t, x_{t\text{-}1} \mid y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; dx_{t\text{-}1} \\
&= c_t \; P(y_t \mid x_t) \; \int_{x_{t\text{-}1}} P(x_t \mid x_{t\text{-}1}, y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; P(x_{t\text{-}1} \mid y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; dx_{t\text{-}1} \\
&= c_t \; P(y_t \mid x_t) \; \int_{x_{t\text{-}1}} P(x_t \mid x_{t\text{-}1}, u_{t\text{-}1}) \; P(x_{t\text{-}1} \mid y_{0:t\text{-}1}, u_{0:t\text{-}1}) \; dx_{t\text{-}1} \\
&= c_t \; P(y_t \mid x_t) \; \int_{x_{t\text{-}1}} P(x_t \mid u_{t\text{-}1}, x_{t\text{-}1}) \; p_{t\text{-}1}(x_{t\text{-}1}) \; dx_{t\text{-}1}
\end{aligned}
$$

- A Bayes filter updates the posterior belief $p_t(x_t)$ in each time step using the:
  observation model $P(y_t \mid x_t)$
  transition model $P(x_t \mid u_{t\text{-}1}, x_{t\text{-}1})$

Bayes Filter

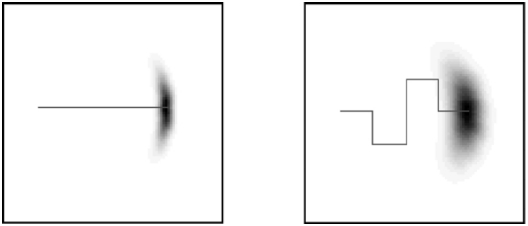
$$p_t(x_t) \;\propto\; \underbrace{P(y_t \,|\, x_t)}_{\text{new information}} \; \underbrace{\int_{x_{t\text{-}1}} P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1}) \; \underbrace{p_{t\text{-}1}(x_{t\text{-}1})}_{\text{old estimate}} \; dx_{t\text{-}1}}_{\text{predictive estimate } \hat{p}_t(x_t)}$$

1. We have a belief $p_{t\text{-}1}(x_{t\text{-}1})$ of our previous position

2. We use the motion model to predict the current position
$$\hat{p}_t(x_t) \propto \int_{x_{t\text{-}1}} P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1}) \, p_{t\text{-}1}(x_{t\text{-}1}) \, dx_{t\text{-}1}$$

3. We integetrate this with the current observation to get a better belief
$$p_t(x_t) \propto P(y_t \,|\, x_t) \, \hat{p}_t(x_t)$$

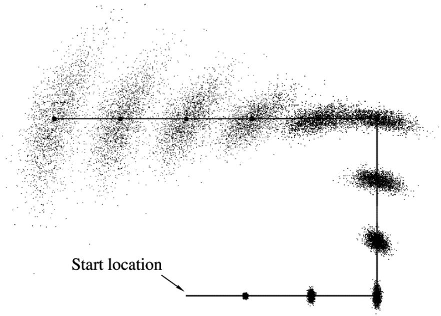
- Typical transition model $P(x_t \mid u_{t\text{-}1}, x_{t\text{-}1})$ in robotics:



(from *Robust Monte Carlo localization for mobile robots* Sebastian Thrun, Dieter Fox, Wolfram Burgard, Frank Dellaert)

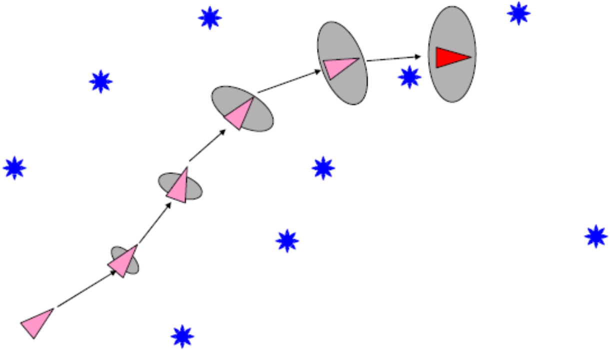
Odometry ("Dead Reckoning")

- The predictive distributions $\hat{p}_t(x_t)$ *without integrating observations* (removing the $P(y_t|x_t)$ part from the Bayesian filter)

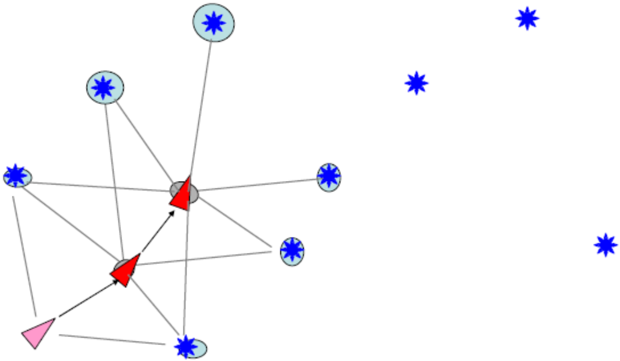


(from *Robust Monte Carlo localization for mobile robots* Sebastian Thrun, Dieter Fox, Wolfram Burgard, Frank Dellaert)

Again, predictive distributions $\hat{p}_t(x_t)$ *without integrating landmark observations*



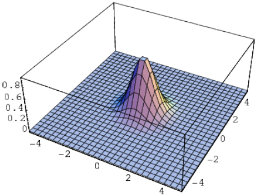
The Bayes-filtered distributions $p_t(x_t)$ integrating landmark observations



Bayesian Filters

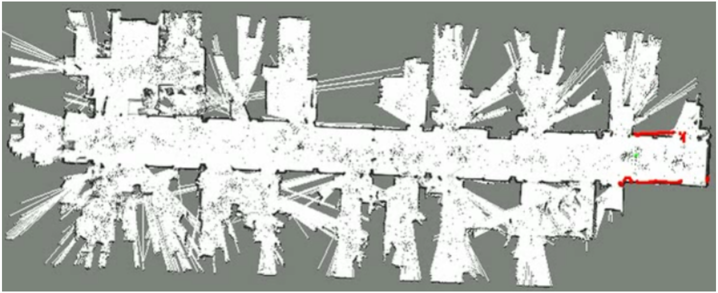
- How to represent the belief $p_t(x_t)$:

- Gaussian



$$\frac{1}{\sigma\sqrt{2\pi}}\ \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

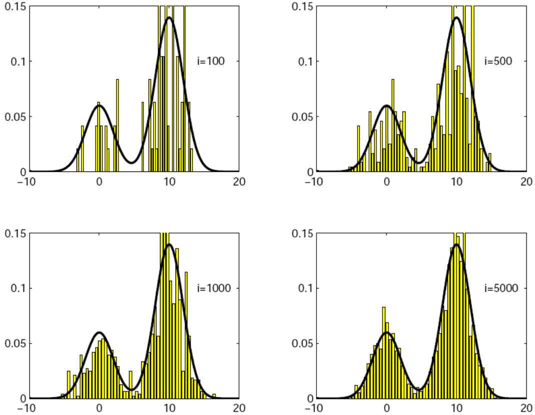
- Particles



Recall: Particle Representation of a Distribution

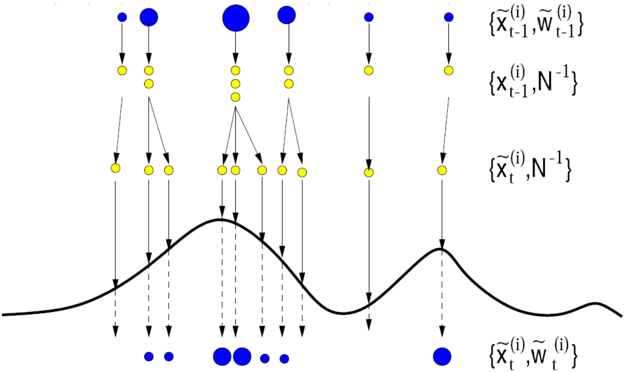
- Weighed set of N particles $\{(x^i, w^i)\}_{i=1}^N$

$$p(x) \approx q(x) := \sum_{i=1}^N w^i \delta(x, x^i)$$



Particle Filter := Bayesian Filtering with Particles

(Bayes Filter: $p_t(x_t) \propto P(y_t \mid x_t) \int_{x_{t\text{-}1}} P(x_t \mid u_{t\text{-}1}, x_{t\text{-}1})\, p_{t\text{-}1}(x_{t\text{-}1})\, dx_{t\text{-}1}$)



1. Start with N particles $\{(x^i_{t\text{-}1}, w^i_{t\text{-}1})\}_{i=1}^N$
2. Resample particles to get N weight-1-particles: $\{\hat{x}^i_{t\text{-}1}\}_{i=1}^N$
3. Use motion model to get new "predictive" particles $\{x^i_t\}_{i=1}^N$
   each $x^i_t \sim P(x_t \mid u_{t\text{-}1}, \hat{x}^i_{t\text{-}1})$
4. Use observation model to assign new weights $w^i_t \propto P(y_t \mid x^i_t)$

- "Particle Filter"

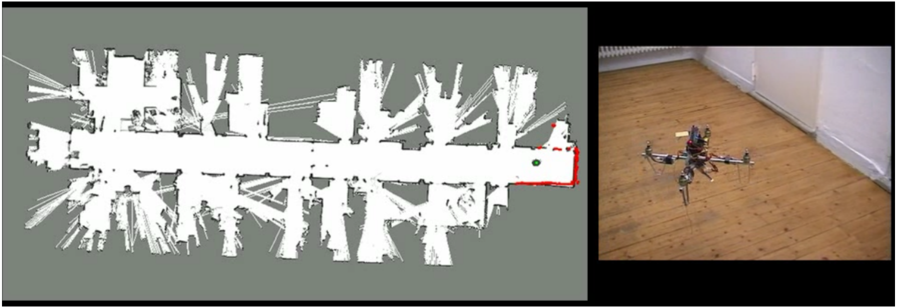
  aka *Monte Carlo Localization* in the mobile robotics community

  Condensation Algorithm in the vision community

- Efficient resampling is important:
  Typically "Residual Resampling":

  Instead of sampling directly $\hat{n}^i \sim$ Multi$(\{Nw_i\})$ set $\hat{n}^i = \lfloor Nw_i \rfloor + \bar{n}_i$ with $\bar{n}_i \sim$ Multi$(\{Nw_i - \lfloor Nw_i \rfloor\})$
  Liu & Chen (1998): Sequential Monte Carlo Methods for Dynamic Systems.
  Douc, Cappé & Moulines: Comparison of Resampling Schemes for Particle Filtering.

Example: Quadcopter Localization



http://www.slawomir.de/publications/grzonka09icra/grzonka09icra.pdf

Quadcopter Indoor Localization

Typical Pitfall in Particle Filtering

- Predicted particles $\{x_t^i\}_{i=1}^N$ have very low observation likelihood $P(y_t \,|\, x_t^i) \approx 0$
  ("particles die over time")

- Classical solution: generate particles also with other than purely forward proposal $P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1})$:

  – Choose a proposal that depends on the new observation y_t, ideally approximating $P(x_t \,|\, y_t, u_{t\text{-}1}, x_{t\text{-}1})$

  – Or mix particles sampled directly from $P(y_t \,|\, x_t)$ and from $P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1})$.
  (*Robust Monte Carlo localization for mobile robots.* Sebastian Thrun, Dieter Fox, Wolfram Burgard, Frank Dellaert)

Kalman filter := Bayesian Filtering with Gaussians

Bayes Filter: $p_t(x_t) \ \propto \ P(y_t \,|\, x_t) \int_{x_{t\text{-}1}} P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1})\, p_{t\text{-}1}(x_{t\text{-}1})\; dx_{t\text{-}1}$

- Can be computed analytically for linear-Gaussian observations and transitions:

  $P(y_t \,|\, x_t) = \mathcal{N}(y_t \,|\, Cx_t + c, W)$
  $P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1}) = \mathcal{N}(x_t \,|\, A(u_{t\text{-}1})\; x_{t\text{-}1} + a(u_{t\text{-}1}), Q)$

Defition:
$\mathcal{N}(x \,|\, a, A) = \frac{1}{|2\pi A|^{1/2}}\ \exp\{-\frac{1}{2}(x \text{-} a)^\top\, A^{\text{-}1}\ (x \text{-} a)\}$
Product:
$\mathcal{N}(x \,|\, a, A)\; \mathcal{N}(x \,|\, b, B) = \mathcal{N}(x \,|\, B(A{+}B)^{\text{-}1} a + A(A{+}B)^{\text{-}1} b, A(A{+}B)^{\text{-}1} B)\ \ \mathcal{N}(a \,|\, b, A + B)$
"Propagation":
$\int_y \mathcal{N}(x \,|\, a + Fy, A)\; \mathcal{N}(y \,|\, b, B)\; dy = \mathcal{N}(x \,|\, a + Fb, A + FBF^\top)$
Transformation:
$\mathcal{N}(Fx + f \,|\, a, A) = \frac{1}{|F|}\ \mathcal{N}(x \,|\ \ F^{\text{-}1}(a - f),\ F^{\text{-}1} A F^{\text{-}\top})$

(more identities: see "Gaussian identities"
`http://ipvs.informatik.uni-stuttgart.de/mlr/marc/notes/gaussians.pdf`)

Kalman filter derivation

$$
\begin{aligned}
p_t(x_t) &= \mathcal{N}(x_t \,|\, s_t, S_t) \\
P(y_t \,|\, x_t) &= \mathcal{N}(y_t \,|\, C x_t + c, W) \\
P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1}) &= \mathcal{N}(x_t \,|\, A x_{t\text{-}1} + a, Q)
\end{aligned}
$$

$$
\begin{aligned}
p_t(x_t) &\propto P(y_t \,|\, x_t) \int_{x_{t\text{-}1}} P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1})\, p_{t\text{-}1}(x_{t\text{-}1})\, dx_{t\text{-}1} \\
&= \mathcal{N}(y_t \,|\, C x_t + c, W) \int_{x_{t\text{-}1}} \mathcal{N}(x_t \,|\, A x_{t\text{-}1} + a, Q)\, \mathcal{N}(x_{t\text{-}1} \,|\, s_{t\text{-}1}, S_{t\text{-}1})\, dx_{t\text{-}1} \\
&= \mathcal{N}(y_t \,|\, C x_t + c, W)\, \mathcal{N}(x_t \,|\, \underbrace{A s_{t\text{-}1} + a}_{=: \hat s_t},\ \underbrace{Q + A S_{t\text{-}1} A^\top}_{=: \hat S_t}) \\
&= \mathcal{N}(C x_t + c \,|\, y_t, W)\, \mathcal{N}(x_t \,|\, \hat s_t, \hat S_t) \\
&= \mathcal{N}[x_t \,|\, C^\top W^{\text{-}1}(y_t - c),\ C^\top W^{\text{-}1} C]\, \mathcal{N}(x_t \,|\, \hat s_t, \hat S_t) \\
&= \mathcal{N}(x_t \,|\, s_t, S_t) \cdot \langle \text{terms indep. of } x_t \rangle \\
S_t &= (C^\top W^{\text{-}1} C + \hat S_t^{\text{-}1})^{\text{-}1} = \hat S_t - \underbrace{\hat S_t C^\top (W + C \hat S_t C^\top)^{\text{-}1}}_{\text{“Kalman gain” } K} C \hat S_t \\
s_t &= S_t [C^\top W^{\text{-}1}(y_t - c) + \hat S_t^{\text{-}1} \hat s_t] = \hat s_t + K(y_t - C \hat s_t - c)
\end{aligned}
$$

The second to last line uses the general Woodbury identity.
The last line uses $S_t C^\top W^{\text{-}1} = K$ and $S_t \hat S_t^{\text{-}1} = \mathbf{I} - KC$

Extended Kalman filter (EKF) and Unscented Transform

Bayes Filter: $p_t(x_t) \;\propto\; P(y_t \,|\, x_t) \int_{x_{t\text{-}1}} P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1})\; p_{t\text{-}1}(x_{t\text{-}1})\; dx_{t\text{-}1}$

- Can be computed analytically for linear-Gaussian observations and transitions:
  $P(y_t \,|\, x_t) = \mathcal{N}(y_t \,|\, C x_t + c, W)$
  $P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1}) = \mathcal{N}(x_t \,|\, A(u_{t\text{-}1}) x_{t\text{-}1} + a(u_{t\text{-}1}), Q)$

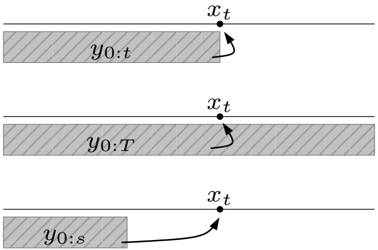
- If $P(y_t \,|\, x_t)$ or $P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1})$ are not linear:
  $P(y_t \,|\, x_t) = \mathcal{N}(y_t \,|\, g(x_t), W)$
  $P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1}) = \mathcal{N}(x_t \,|\, f(x_{t\text{-}1}, u_{t\text{-}1}), Q)$
  – approximate f and g as locally linear (*Extended Kalman Filter*)
  – or sample locally from them and reapproximate as Gaussian (*Unscented Transform*)

Bayes smoothing

Filtering: $P(x_t|y_{0:t})$

Smoothing: $P(x_t|y_{0:T})$

Prediction: $P(x_t|y_{0:s})$



Bayes smoothing

- Let $\mathcal{P} = y_{0:t}$ past observations, $\mathcal{F} = y_{t+1:T}$ future observations

$$\begin{aligned} P(x_t \,|\, \mathcal{P}, \mathcal{F}, u_{0:T}) &\propto P(\mathcal{F} \,|\, x_t, \mathcal{P}, u_{0:T}) \; P(x_t \,|\, \mathcal{P}, u_{0:T}) \\ &= \underbrace{P(\mathcal{F} \,|\, x_t, u_{t:T})}_{=:\beta_t(x_t)} \; \underbrace{P(x_t \,|\, \mathcal{P}, u_{0:t\text{-}1})}_{=:p(x_t)} \end{aligned}$$

Bayesian smoothing fuses a forward filter $p_t(x_t)$ with a backward “filter” $\beta_t(x_t)$

Bayes smoothing

- Let $\mathcal{P} = y_{0:t}$ past observations, $\mathcal{F} = y_{t+1:T}$ future observations

$$
\begin{aligned}
P(x_t \,|\, \mathcal{P}, \mathcal{F}, u_{0:T}) &\propto P(\mathcal{F} \,|\, x_t, \mathcal{P}, u_{0:T})\; P(x_t \,|\, \mathcal{P}, u_{0:T}) \\
&= \underbrace{P(\mathcal{F} \,|\, x_t, u_{t:T})}_{=:\beta_t(x_t)}\; \underbrace{P(x_t \,|\, \mathcal{P}, u_{0:t\text{-}1})}_{=:p(x_t)}
\end{aligned}
$$

Bayesian smoothing fuses a forward filter $p_t(x_t)$ with a backward “filter” $\beta_t(x_t)$

- Backward recursion (derivation analogous to the Bayesian filter)

$$
\begin{aligned}
\beta_t(x_t) &:= P(y_{t+1:T} \,|\, x_t, u_{t:T}) \\
&= \int_{x_{t+1}} \beta_{t+1}(x_{t+1})\; P(y_{t+1} \,|\, x_{t+1})\; P(x_{t+1} \,|\, x_t, u_t)\; dx_{t+1}
\end{aligned}
$$

PART II: Localization and Mapping

- The Bayesian filter requires an observation model $P(y_t \,|\, x_t)$

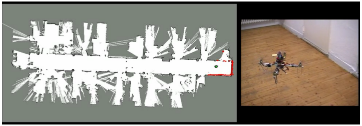
- A **map** is something that provides the observation model:
  A map tells us for each x_t what the sensor readings y_t might look like

Types of maps

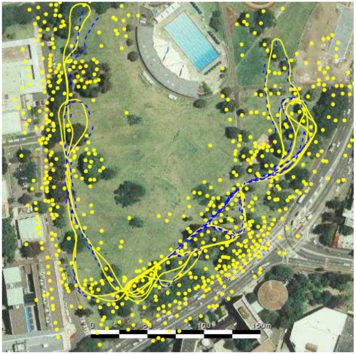
Grid map

K. Murphy (1999): *Bayesian map learning in dynamic environments.*
Grisetti, Tipaldi, Stachniss, Burgard, Nardi: *Fast and Accurate SLAM with Rao-Blackwellized Particle Filters*

Laser scan map



Landmark map



Victoria Park data set
M. Montemerlo, S. Thrun, D. Koller, & B. Wegbreit (2003): *FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges.* IJCAI, 1151–1156.

Simultaneous Localization and Mapping Problem

- Notation:
  x_t = state (location) at time t
  y_t = sensor readings at time t
  $u_{t\text{-}1}$ = control command (action, steering, velocity) at time t-1
  m = the map

- Given the history $y_{0:t}$ and $u_{0:t\text{-}1}$, we want to compute the belief over the pose AND THE MAP m at time t

$$p_t(x_t, m) := P(x_t, m \mid y_{0:t}, u_{0:t\text{-}1})$$

- We assume to know:
  – transition model $P(x_t \mid u_{t\text{-}1}, x_{t\text{-}1})$
  – observation model $P(y_t \mid x_t, m)$

SLAM: classical "chicken or egg problem"

- If we knew the state trajectory $x_{0:t}$ we could efficiently compute the belief over the map

$$P(m \mid x_{0:t}, y_{0:t}, u_{0:t\text{-}1})$$

- If we knew the map we could use a Bayes filter to compute the belief over the state

$$P(x_t \mid m, y_{0:t}, u_{0:t\text{-}1})$$

SLAM: classical "chicken or egg problem"

- If we knew the state trajectory $x_{0:t}$ we could efficiently compute the belief over the map

$$P(m \,|\, x_{0:t}, y_{0:t}, u_{0:t\text{-}1})$$

- If we knew the map we could use a Bayes filter to compute the belief over the state

$$P(x_t \,|\, m, y_{0:t}, u_{0:t\text{-}1})$$

- SLAM requires to tie state estimation and map building together:
  1) Joint inference on x_t and m ($\rightarrow$ Kalman-SLAM)
  2) Tie a state hypothesis (=particle) to a map hypothesis ($\rightarrow$ particle SLAM)
  3) Frame everything as a graph optimization problem ($\rightarrow$ graph SLAM)

Joint Bayesian Filter over x and m

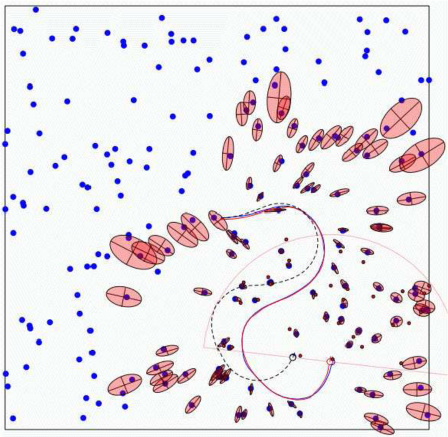
- A (formally) straight-forward approach is the joint Bayesian filter

$$p_t(x_t, m) \;\propto\; P(y_t \,|\, x_t, m) \;\int_{x_{t\text{-}1}} P(x_t \,|\, u_{t\text{-}1}, x_{t\text{-}1})\; p_{t\text{-}1}(x_{t\text{-}1}, m)\; dx_{t\text{-}1}$$

But: How represent a belief over high-dimensional x_t, m?

Map uncertainty

- In the case the map $m = (\theta_1, .., \theta_N)$ is a set of N landmarks, $\theta_j \in \mathbb{R}^2$



- Use Gaussians to represent the uncertainty of landmark positions

(Extended) Kalman Filter SLAM

- Analogous to Localization with Gaussian for the pose belief $p_t(x_t)$
  - But now: joint belief $p_t(x_t, \theta_{1:N})$ is $3 + 2N$-dimensional Gaussian
  - Assumes the map $m = (\theta_1, .., \theta_N)$ is a set of N landmarks, $\theta_j \in \mathbb{R}^2$
  - Exact update equations (under the Gaussian assumption)
  - Conceptually very simple

- Drawbacks:
  - Scaling (full covariance matrix is $O(N^2)$)
  - Sometimes non-robust (uni-modal, “data association problem”)
  - Lacks advantages of Particle Filter
    (multiple hypothesis, more robust to non-linearities)

SLAM with particles

Core idea: Each particle carries its own map belief

SLAM with particles

Core idea: Each particle carries its own map belief

- Use a conditional representation "$p_t(x_t, m) = p_t(x_t)\ p_t(m \,|\, x_t)$"
  (This notation is flaky... the below is more precise)

- As for the Localization Problem use particles to represent the *pose belief* $p_t(x_t)$
  Note: Each particle actually "has a history $x^i_{0:t}$" – a whole trajectory!

- For each particle separately, estimate the *map belief* $p^i_t(m)$ conditioned on the particle history $x^i_{0:t}$.
  The conditional beliefs $p^i_t(m)$ may be factorized over grid points or landmarks of the map

  K. Murphy (1999): *Bayesian map learning in dynamic environments.*
  `http://www.cs.ubc.ca/~murphyk/Papers/map_nips99.pdf`

Map estimation for a *given* particle history

- Given $x_{0:t}$ (e.g. a trajectory of a particle), what is the posterior over the map m?

  $\rightarrow$ simplified Bayes Filter:

  $$p_t(m) := P(m \mid x_{0:t}, y_{0:t}) \propto P(y_t \mid m, x_t)\; p_{t\text{-}1}(m)$$

  (no transtion model: assumption that map is constant)

- In the case of landmarks (FastSLAM):
  $m = (\theta_1, .., \theta_N)$
  θ_j = position of the jth landmark, $j \in \{1, .., N\}$
  n_t = which landmark we observe at time t, $\quad n_t \in \{1, .., N\}$

  We can use a separate (Gaussian) Bayes Filter for each θ_j
  conditioned on $x_{0:t}$, each θ_j is independent from each θ_k:

  $$P(\theta_{1:N} \mid x_{0:t}, y_{0:n}, n_{0:t}) = \prod_j P(\theta_j \mid x_{0:t}, y_{0:n}, n_{0:t})$$

Particle likelihood in SLAM

- Particle likelihood for Localization Problem:
$$w_t^i = P(y_t \,|\, x_t^i)$$
(determins the new importance weight w_t^i

- In SLAM the map is uncertain $\rightarrow$ each particle is weighted with the *expected* likelihood:
$$w_t^i = \int P(y_t \,|\, x_t^i, m)\; p_{t-1}(m)\; dm$$

- In case of landmarks (FastSLAM):
$$w_t^i = \int P(y_t \,|\, x_t^i, \theta_{n_t}, n_t)\; p_{t-1}(\theta_{n_t})\; d\theta_{n_t}$$

- Data association problem (actually we don't know n_t):
For each particle separately choose $n_t^i = \mathrm{argmax}_{n_t}\, w_t^i(n_t)$

Particle-based SLAM summary

- We have a set of N particles $\{(x^i, w^i)\}_{i=1}^N$ to represent the pose belief $p_t(x_t)$

- For each particle we have a separate map belief $p_t^i(m)$; in the case of landmarks, this factorizes in N separate 2D-Gaussians

- Iterate
  1. Resample particles to get N weight-1-particles: $\{\hat{x}_{t\text{-}1}^i\}_{i=1}^N$
  2. Use motion model to get new "predictive" particles $\{x_t^i\}_{i=1}^N$
  3. Update the map belief $p_m^i(m) \propto P(y_t \,|\, m, x_t)\; p_{t\text{-}1}^i(m)$ for each particle
  4. Compute new importance weights $w_t^i \propto \int P(y_t \,|\, x_t^i, m)\; p_{t-1}(m)\; dm$ using the observation model and the map belief

Demo: Visual SLAM

- Map building from a freely moving camera

Demo: Visual SLAM

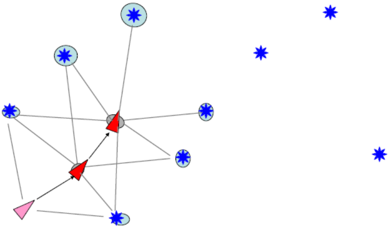
- Map building from a freely moving camera

  – SLAM has become a bit topic in the vision community..
  – features are typically landmarks $\theta_{1:N}$ with SURF/SIFT features
  – PTAM (Parallel Tracking and Mapping) parallelizes computations...

PTAM1 PTAM2
TODO: 11-DTAM-Davidson

G Klein, D Murray: *Parallel Tracking and Mapping for Small AR Workspaces* `http://www.robots.ox.ac.uk/~gk/PTAM/`

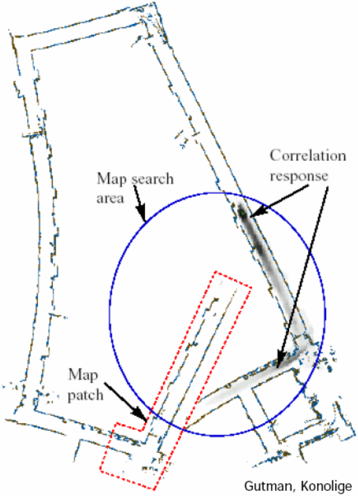
Alternative SLAM approach: Graph-based



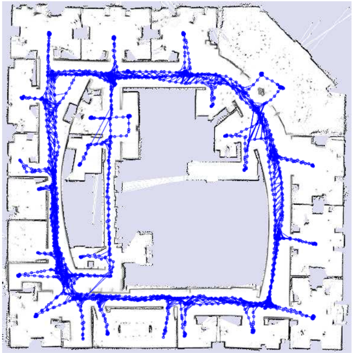
- Represent the previous trajectory as a graph
  – nodes = estimated positions & observations
  – edges = transition & step estimation based on scan matching
- Loop Closing: check if some nodes might coincide $\rightarrow$ new edges
- Classical Optimization:
  The whole graph defines an optimization problem: Find poses that minimize sum of edge & node errors

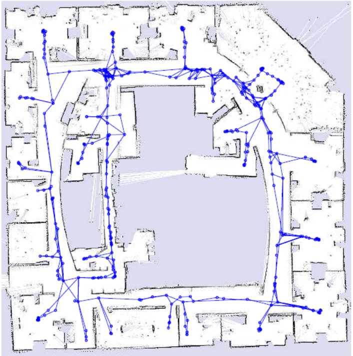
Loop Closing Problem

(Doesn't explicitly exist in Particle Filter methods: If particles cover the belief, then "data association" solves the "loop closing problem")



Graph-based SLAM





Life-long Map Learning for Graph-based SLAM Approaches in Static Environments

Kretzschmar, Grisetti, Stachniss

SLAM code

- Graph-based and grid map methods:
  `http://openslam.org/`

- Visual SLAM
  e.g. `http://ewokrampage.wordpress.com/`