Principles of Safe Autonomy: Lecture 15: Filtering applications and SLAM

Sayan Mitra

Reference: Probabilistic Robotics by Sebastian Thrun, Wolfram Burgard, and Dieter Fox
Slides: From the book’s website
Outline of filtering and state estimation module

- Applications of Particle filter
  - Monte Carlo localization (MCL)
- Kahoot
- Overview of SLAM
Particle Filters

• Represent belief by finite number of parameters (just like histogram filter)
• But, they differ in how the parameters (particles) are generated and populate the state space
• Key idea: represent belief $bel(x_t)$ by a random set of state samples
• Advantages
  • The representation is approximate and nonparametric and therefore can represent a broader set of distributions than e.g., Gaussian
  • Can handle nonlinear transformations
• Related ideas: Monte Carlo filter, Survival of the fittest, Condensation, Bootstrap filter, Filtering: [Rubin, 88], [Gordon et al., 93], [Kitagawa 96], Dynamic Bayesian Networks: [Kanazawa et al., 95]
Particle filtering algorithm

\[ X_t = x_t^{[1]}, x_t^{[2]}, ..., x_t^{[M]} \] particles

Algorithm Particle_filter(\(X_{t-1}, u_t, z_t\)):

\( \tilde{X}_{t-1} = X_t = \emptyset \) for all \( m \) in \([M]\) do:

\begin{align*}
& \text{sample } x_t^{[m]} \sim p(x_t | u_t, x_{t-1}^{[m]}) \\
& w_t^{[m]} = p(z_t | x_t^{[m]}) \\
& X_t = X_t + (x_t^{[m]}, w_t^{[m]})
\end{align*}

end for

for all \( m \) in \([M]\) do:

\begin{align*}
& \text{draw } i \text{ with probability } \propto w_t^{[i]} \\
& \text{add } x_t^{[i]} \text{ to } X_t
\end{align*}

end for

return \( X_t \)

ideally, \( x_t^{[m]} \) is selected with probability prop. to \( p(x_t | z_{1:t}, u_{1:t}) \)

\( \tilde{X}_{t-1} \) is the temporary particle set

\begin{itemize}
  \item // sampling from state transition dist.
  \item // calculates importance factor \( w_t \) or weight
  \item // resampling or importance sampling; these are distributed according to \( \eta p(z_t | x_t^{[m]}) \overline{bel}(x_t) \)
  \item // survival of fittest: moves/adds particles to parts of the state space with higher probability
\end{itemize}
Localization as coordinate transformation

Shaded known:
map (m), control inputs (u),
measurements(z). White nodes
to be determined (x)

maps (m) are described in
global coordinates. Localization
= establish coord transf.,
between m and robot’s local
coordinates

Transformation used for objects
of interest (obstacles, pedestrians) for decision,
planning and control
Monte Carlo Localization

- Represents beliefs by particles
Importance Sampling

suppose we want to compute $E_f [I(x \in A)]$ but we can only sample from density $g$

$E_f [I(x \in A)]$

$= \int f(x)I(x \in A)dx$
$= \int \frac{f(x)}{g(x)}g(x)I(x \in A)dx$, provided $g(x) > 0$
$= \int w(x)g(x)I(x \in A)dx$
$= E_g [w(x)I(x \in A)]$

We need $f(x) > 0 \Rightarrow g(x) > 0$

**Weight samples:** $w = f / g$
Monte Carlo Localization (MCL)

\( X_t = x_t[1], x_t[2], \ldots, x_t[M] \) particles

**Algorithm MCL**\((X_{t-1}, u_t, z_t, m)\):

\( \bar{X}_{t-1} = X_t = \emptyset \)

for all \( m \) in \([M]\) do:

\[
\begin{align*}
x_t[m] & = \text{sample\_motion\_model}(u_t, x_{t-1}[m]) \\
w_t[m] & = \text{measurement\_model}(z_t, x_t[m], m) \\
\bar{X}_t & = \bar{X}_t + (x_t[m], w_t[m])
\end{align*}
\]

end for

for all \( m \) in \([M]\) do:

\[
\begin{align*}
draw \text{ i with probability } & \propto w_t[i] \\
\text{add } x_t[i] \text{ to } X_t
\end{align*}
\]

end for

return \( X_t \)

Plug in motion and measurement models in the particle filter
Particle Filters
Sensor Information: Importance Sampling

\[ Bel(x) \leftarrow \alpha \ p(z \mid x) \ Bel^-(x) \]
\[ w \leftarrow \frac{\alpha \ p(z \mid x) \ Bel^-(x)}{Bel^-(x)} = \alpha \ p(z \mid x) \]
Robot Motion

\[ Bel^{-}(x) \leftarrow \int p(x \mid u, x') Bel(x') \, dx' \]
Sensor Information: Importance Sampling

\[ \begin{align*}
\text{Bel}(x) & \leftarrow \alpha \ p(z \mid x) \ \text{Bel}^{-}(x) \\
\text{w} & \leftarrow \frac{\alpha \ p(z \mid x) \ \text{Bel}^{-}(x)}{\text{Bel}^{-}(x)} = \alpha \ p(z \mid x)
\end{align*} \]
Robot Motion

\[ \text{Bel}^{-}(x) \leftarrow \int p(x \mid u, x') \text{Bel}(x') \, dx' \]
Sample-based Localization (sonar)
Initial Distribution
After Incorporating Ten Ultrasound Scans
After Incorporating 65 Ultrasound Scans
Estimated Path
Using Ceiling Maps for Localization

Sensor: Upward looking camera
Map / model of the world: Ceiling Mosaic (construction is nontrivial)
https://www.cs.cmu.edu/~minerva/tech/mosaic.html
Vision-based Localization

\[ P(z|x) \]

\[ h(x) \]
Under a Light

Measurement $z$:  

$P(z|x)$:
Next to a Light

Measurement $z$:  

$P(z|x)$:
Elsewhere

Measurement $z$: $P(z|x)$:
Global Localization Using Vision
Kahoot

• [https://play.kahoot.it/v2/?quizId=3f040019-06e6-4fbe-9c98-780be526f271](https://play.kahoot.it/v2/?quizId=3f040019-06e6-4fbe-9c98-780be526f271)
Summary: Advantages and Limitations of MCL

Advantages of particle filtering-based localization (MCL)

- Solves global localization
- Can approximate any distributions (non-parametric)
- Increasing M improves accuracy of approximation (clear trade-off)
  - Possible to have adaptive implementations
  - Track the pose of a mobile robot and to

Disadvantages

- Cannot solve global localization failures or kidnapped robot problem
  - Disappearance of diversity: particles other than the most likely positions disappear; only near a single pose “survive”; cannot recover if the pose is wrong
    - Can be resolved by injecting some random particles; how many? from what distribution?
    - Add particles based on some estimate of localization performance
      \[ p(z_t | z_{t-1}) = \frac{1}{M} \sum w_t^{[m]} \]
  - Particle deprivation: if \( p(x_t | x_{t-1}, u_t) \) is very different from \( p(x_t | z_t) \) then many more particles are needed; if the measurement model has no uncertainty---no noise---MCL fails
    - Simple solution trick: use noisy sensors;
Random Samples
Vision-Based Localization

936 Images, 4MB, .6secs/image

Trajectory of the robot:
Kidnapping the Robot
The SLAM Problem

- SLAM stands for simultaneous localization and mapping
- The task of building a map while estimating the pose of the robot relative to this map
- Why is SLAM hard?
  Chicken and egg problem: a map is needed to localize the robot and a pose estimate is needed to build a map
The SLAM Problem

A robot moving though an unknown, static environment

Given:
- The robot’s controls
- Observations of nearby features

Estimate:
- Map of features
- Path of the robot
SLAM Applications

Indoors

Space

Undersea

Underground
Online SLAM

Shaded known: control inputs (u), measurements (z).
White nodes to be determined (x, m)

want to calculate
\( p(x_t, m | z_{1:t}, u_{1:t}) \)
Shaded known: control inputs \((u)\), measurements\((z)\). White nodes to be determined \((x,m)\)

want to calculate
\[ p(x_{1:t}, m|z_{1:t}, u_{1:t}) \]

Continuous unknowns: \(x_{1:t}, m\)
Discrete unknowns: Relationship of detected objects to new objects

\[ p(x_{1:t}, c_t, m|z_{1:t}, u_{1:t}) \]

\(c_t\): correspondence variable
Representations

• Grid maps

[Lu & Milios, 97; Gutmann, 98; Thrun 98; Burgard, 99; Konolige & Gutmann, 00; Thrun, 00; Arras, 99; Haehnel, 01;…]

• Landmark-based

[Leonard et al., 98; Castelanos et al., 99; Dissanayake et al., 2001; Montemerlo et al., 2002;…]
Why is SLAM a hard problem?

**SLAM**: robot path and map are both **unknown**

Robot path error correlates errors in the map
Why is SLAM a hard problem?

- In the real world, the mapping between observations and landmarks is unknown
- Picking wrong data associations can have catastrophic consequences
- Pose error correlates data associations
SLAM:
Simultaneous Localization and Mapping

• Full SLAM: Estimates entire path and map!

\[ p(x_{1:t}, m | z_{1:t}, u_{1:t}) \]

• Online SLAM: Integrations typically done one at a time.

\[
p(x_t, m | z_{1:t}, u_{1:t}) = \int \int \ldots \int p(x_t, m | z_{1:t}, u_{1:t}) \, dx_1 \, dx_2 \ldots dx_{t-1}
\] Estimates most recent pose and map!
Data Association Problem

• A data association is an assignment of observations to landmarks
• In general there are more than $\binom{n}{m}$ (n observations, m landmarks) possible associations
• Also called “assignment problem”
Particle Filters

- Represent belief by random samples
- Estimation of non-Gaussian, nonlinear processes

- Sampling Importance Resampling (SIR) principle
  - Draw the new generation of particles
  - Assign an importance weight to each particle
  - Resampling

- Typical application scenarios are tracking, localization, ...
Localization vs. SLAM

- A particle filter can be used to solve both problems

- Localization: state space \(<x, y, \theta>\)

- SLAM: state space \(<x, y, \theta, map>\)
  - for landmark maps = \(<l_1, l_2, ..., l_m>\)
  - for grid maps = \(<c_{11}, c_{12}, ..., c_{1n}, c_{21}, ..., c_{nm}>\)

- **Problem:** The number of particles needed to represent a posterior grows exponentially with the dimension of the state space!
• Naïve implementation of particle filters to SLAM will be crushed by the curse of dimensionality
Dependencies

- Is there a dependency between the dimensions of the state space?
- If so, can we use the dependency to solve the problem more efficiently?
Dependencies

- Is there a dependency between the dimensions of the state space?
- If so, can we use the dependency to solve the problem more efficiently?

- In the SLAM context
  - The map depends on the poses of the robot.
  - We know how to build a map given the position of the sensor is known.
Conditional Independence

- A and B are conditionally independent given C if \( P(A, B \mid C) = \frac{P(A \mid C)}{P(B \mid C)} \)

- Height and vocabulary are not independent
- But they are conditionally independent given age
Factored Posterior (Landmarks)

\[ p(x_{1:t}, l_{1:m} \mid z_{1:t}, u_{0:t-1}) = \]
\[ p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot p(l_{1:m} \mid x_{1:t}, z_{1:t}) \]

poses  map  observations & movements

SLAM posterior

Robot path posterior

landmark positions

Does this help to solve the problem?

Factorization first introduced by Murphy in 1999
Factored Posterior (Landmarks)

\[
p(x_{1:t}, l_{1:m} | z_{1:t}, u_{0:t-1}) = \frac{p(x_{1:t} | z_{1:t}, u_{0:t-1}) \cdot p(l_{1:m} | x_{1:t}, z_{1:t})}{p(l_{1:m} | z_{1:t}, u_{0:t-1})}
\]

poses → map → observations & movements

Factorization first introduced by Murphy in 1999
Knowledge of the robot's true path renders landmark positions conditionally independent.
Factored Posterior

\[
p(x_{1:t}, l_{1:m} \mid z_{1:t}, u_{0:t-1}) = p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot p(l_{1:m} \mid x_{1:t}, z_{1:t})
\]

\[
= p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot \prod_{i=1}^{M} p(l_i \mid x_{1:t}, z_{1:t})
\]

Robot path posterior (localization problem)

Conditionally independent landmark positions
Rao-Blackwellization

\[ p(x_{1:t}, l_{1:m} \mid z_{1:t}, u_{0:t-1}) = \]

\[ p(x_{1:t} \mid z_{1:t}, u_{0:t-1}) \cdot \prod_{i=1}^{M} p(l_i \mid x_{1:t}, z_{1:t}) \]

- This factorization is also called Rao-Blackwellization
- Given that the second term can be computed efficiently, particle filtering becomes possible!
FastSLAM

- Rao-Blackwellized particle filtering based on landmarks [Montemerlo et al., 2002]
- Each landmark is represented by a 2x2 Extended Kalman Filter (EKF)
- Each particle therefore has to maintain $M$ EKFs
FastSLAM – Action Update

Particle #1

Particle #2

Particle #3

Landmark #1 Filter

Landmark #2 Filter
FastSLAM – Sensor Update

Particle #1

Particle #2

Particle #3

Landmark #1
Filter

Landmark #2
Filter
FastSLAM – Sensor Update

Particle #1

Particle #2

Particle #3
FastSLAM - Video
FastSLAM Complexity

- Update robot particles based on control $u_{t-1}$: $O(N)$
  - Constant time per particle
- Incorporate observation $z_t$ into Kalman filters: $O(N \cdot \log(M))$
  - Log time per particle
- Resample particle set: $O(N \cdot \log(M))$
  - Log time per particle

$N = \text{Number of particles}$
$M = \text{Number of map features}$
Data Association Problem

- Which observation belongs to which landmark?

- A robust SLAM must consider possible data associations
- Potential data associations depend also on the pose of the robot
Multi-Hypothesis Data Association

- Data association is done on a per-particle basis

- Robot pose error is factored out of data association decisions
Per-Particle Data Association

Was the observation generated by the red or the blue landmark?

- Two options for per-particle data association
  - Pick the most probable match
  - Pick a random association weighted by the observation likelihoods
- If the probability is too low, generate a new landmark

\[
P(\text{observation} | \text{red}) = 0.3 \quad P(\text{observation} | \text{blue}) = 0.7
\]
Results – Victoria Park

- 4 km traverse
- < 5 m RMS position error
- 100 particles

Blue = GPS
Yellow = FastSLAM

Dataset courtesy of University of Sydney
Results – Victoria Park

https://www.youtube.com/watch?v=BIOJSNHYSbc

Dataset courtesy of University of Sydney
Results – Data Association

Comparison of FastSLAM and EKF Given Motion Ambiguity

- FastSLAM
- EKF

Robot RMS Position Error (m)

Error Added to Rotational Velocity (std.)
Results – Accuracy

Accuracy of FastSLAM vs. the EKF on Simulated Data

RMS Pose Error (meters)

Number of Particles
FastSLAM with Scan-Matching
FastSLAM with Scan-Matching
FastSLAM with Scan-Matching
Grid-based SLAM

- Can we solve the SLAM problem if no pre-defined landmarks are available?
- Can we use the ideas of FastSLAM to build grid maps?
- As with landmarks, the map depends on the poses of the robot during data acquisition
- If the poses are known, grid-based mapping is easy ("mapping with known poses")
FastSLAM with Scan-Matching
Typical Evolution of $n_{\text{eff}}$

- visiting new areas
- closing the first loop
- visiting known areas
- second loop closure
Intel Lab

- 15 particles
- four times faster than real-time P4, 2.8GHz
- 5cm resolution during scan matching
- 1cm resolution in final map
15 particles

Compared to FastSLAM with Scan-Matching, the particles are propagated closer to the true distribution.
Outdoor Campus Map

- 30 particles
- 250x250m²
- 1.088 miles (odometry)
- 20cm resolution during scan matching
- 30cm resolution in final map
MIT Killian Court

• The “infinite-corridor-dataset” at MIT
More Details on FastSLAM

- M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. FastSLAM: A factored solution to simultaneous localization and mapping, AAAI'02

- D. Haehnel, W. Burgard, D. Fox, and S. Thrun. An efficient FastSLAM algorithm for generating maps of large-scale cyclic environments from raw laser range measurements, IROS'03


- G. Grisetti, C. Stachniss, and W. Burgard. Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling, ICRA'05

- A. Eliazar and R. Parr. DP-SLAM: Fast, robust simultaneous localization and mapping without predetermined landmarks, IJCAI'03
Summary

- Particle filters are an implementation of recursive Bayesian filtering.
- They represent the posterior by a set of weighted samples.
- In the context of localization, the particles are propagated according to the motion model.
- They are then weighted according to the likelihood of the observations.
- In a re-sampling step, new particles are drawn with a probability proportional to the likelihood of the observation.