Introduction to Mobile Robotics

SLAM – Landmark-based FastSLAM

Wolfram Burgard, Diego Tipaldi

Partial slide courtesy of Mike Montemerlo
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-or-egg problem:
  - A map is needed to localize the robot
  - 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
Map Representations

Typical models are:

- Feature maps
- Grid maps (occupancy or reflection probability maps)

today
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
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
  - Resample

- 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 $< x, y, \theta, \text{map}>$
  - 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!
Dependencies

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

- Is there a dependency between certain 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.
Factored Posterior (Landmarks)

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

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

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

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

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

SLAM posterior

Robot path posterior

landmark positions

Does this help to solve the problem?

Factorization first introduced by Murphy in 1999
Rao-Blackwellization

- Factorization to exploit dependencies between variables:

\[ p(a, b) = p(b | a) p(a) \]

- If \( p(b | a) \) can be computed in closed form, represent only \( p(a) \) with samples and compute \( p(b | a) \) for every sample.

- It comes from the Rao-Blackwell theorem.
Revisit the Graphical Model

Courtesy: Thrun, Burgard, Fox
Revisit the Graphical Model

Courtesy: Thrun, Burgard, Fox
Landmarks are Conditionally Independent Given the Poses

Landmark variables are all disconnected (i.e. independent) given the robot’s path
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 for SLAM

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

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

- 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

<table>
<thead>
<tr>
<th>Particle #1</th>
<th>x, y, θ</th>
<th>Landmark 1</th>
<th>Landmark 2</th>
<th>…</th>
<th>Landmark M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Particle #2</td>
<td>x, y, θ</td>
<td>Landmark 1</td>
<td>Landmark 2</td>
<td>…</td>
<td>Landmark M</td>
</tr>
<tr>
<td>…</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Particle N</td>
<td>x, y, θ</td>
<td>Landmark 1</td>
<td>Landmark 2</td>
<td>…</td>
<td>Landmark M</td>
</tr>
</tbody>
</table>
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 – Sensor Update

Particle #1

Particle #2

Particle #3

Update map of particle #1

Update map of particle #2

Update map of particle #3
FastSLAM - Video
FastSLAM Complexity – Naive

- Update robot particles based on the control: $O(N)$
- Incorporate an observation into the Kalman filters: $O(N)$
- Resample particle set: $O(NM)$

\[
\mathcal{O}(NM)
\]

$N =$ Number of particles
$M =$ Number of map features
A Better Data Structure for FastSLAM

j ≤ 4 ?

T

j ≤ 2 ?

T

j ≤ 1 ?

T

μ₁,Σ₁

μ₂,Σ₂

F

μ₃,Σ₃

μ₄,Σ₄

j ≤ 3 ?

T

μ₅,Σ₅

μ₆,Σ₆

F

μ₇,Σ₇

μ₈,Σ₈

j ≤ 6 ?

F

j ≤ 5 ?

T

μ₉,Σ₉

μ₁₀,Σ₁₀

F

μ₁₁,Σ₁₁

F

j ≤ 7 ?

Courtesy: M. Montemerlo
A Better Data Structure for FastSLAM
FastSLAM Complexity

- Update robot particles based on the control $O(N)$
- Incorporate an observation into the Kalman filters $O(N \log M)$
- Resample particle set $O(N \log M)$

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

- Which observation belongs to which landmark?

- A robust SLAM solution 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 brown landmark?

\[
P(\text{observation}|\text{red}) = 0.3 \quad P(\text{observation}|\text{brown}) = 0.7
\]

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

Dataset courtesy of University of Sydney
Results – Data Association

Comparison of FastSLAM and EKF Given Motion Ambiguity

Robot RMS Position Error (m)

Error Added to Rotational Velocity (std.)
FastSLAM Summary

- FastSLAM factors the SLAM posterior into low-dimensional estimation problems
  - Scales to problems with over 1 million features
- FastSLAM factors robot pose uncertainty out of the data association problem
  - Robust to significant ambiguity in data association
  - Allows data association decisions to be delayed until unambiguous evidence is collected
- Advantages compared to the classical EKF approach (especially with non-linearities)
- Complexity of $O(N \log M)$