Bayesian Approaches to Localization, Mapping, and SLAM

Robotics Institute 16-735
http://voronoi.sbp.ri.cmu.edu/~motion

Howie Choset
http://voronoi.sbp.ri.cmu.edu/~choset
Quick Probability Review: Bayes Rule

\[ p(a \mid b) = \frac{p(b \mid a) \ p(a)}{p(b)} \]

\[ p(a \mid b, c) = \frac{p(b \mid a, c) \ p(a \mid c)}{p(b \mid c)} \]
QPR: Law of Total Probability

\[ p(a) = \sum_i p(a \land b_i) \]

Discrete

\[ = \sum_i p(a | b_i)p(b_i) \]

Continuous

\[ p(a) = \int p(a | b)p(b)db \]

it follows that:

\[ p(a | b) = \int p(a | b, c)p(c | b)dc \]
QPR: Markov Assumption

Future is Independent of Past Given Current State

“Assume Static World”
The Problem

• What is the world around me (mapping)
  – sense from various positions
  – integrate measurements to produce map
  – assumes perfect knowledge of position

• Where am I in the world (localization)
  – sense
  – relate sensor readings to a world model
  – compute location relative to model
  – assumes a perfect world model

• Together, these are SLAM (Simultaneous Localization and Mapping)
Localization

Tracking: Known initial position
Global Localization: Unknown initial position
Re-Localization: Incorrect known position
(kidnapped robot problem)

SLAM

Mapping while tracking locally and globally

Challenges
- Sensor processing
- Position estimation
- Control Scheme
- Exploration Scheme
- Cycle Closure
- Autonomy
- Tractability
- Scalability
Representations for Robot Localization

Discrete approaches (’95)
- Topological representation (’95)
  - uncertainty handling (POMDPs)
  - occas. global localization, recovery
- Grid-based, metric representation (’96)
  - global localization, recovery

Kalman filters (late-80s?)
- Gaussians
- approximately linear models
- position tracking

Particle filters (’99)
- sample-based representation
- global localization, recovery

Multi-hypothesis (’00)
- multiple Kalman filters
- global localization, recovery

AI

Robotics

RI 16-735, Howie Choset
The Basic Idea

Robot can be anywhere

[Simmons/Koenig 95]
[Kaelbling et al 96]
[Burgard et al 96]
Notes

• Perfect Sensing
  – No false positives/neg.
  – No error

• Data association
Notation for Localization

The posterior

\[ P(x(k) \mid u(0 : k - 1), y(1 : k)) \]

- At every step \( k \)
- Probability over all configurations
- Given
  - Sensor readings \( y \) from 1 to \( k \)
  - Control inputs \( u \) from 0 to \( k-1 \)
  - Interleaved:
    \[ u(0), y(1), \ldots, u(k - 1), y(k) \]

Map \( m \) (should be in condition statements too)

Velocities, force, odometry, something more complicated

RI 16-735, Howie Choset
Predict and Update, combined

\[ P(x(k) \mid u(0 : k - 1), y(1 : k)) \]

\[ = \eta(k) \frac{P(y(k) \mid x(k))}{\sum_{x(k-1) \in X} \left( P(x(k) \mid u(k-1), x(k-1)) \frac{P(x(k-1) \mid u(0 : k - 2), y(1 : k - 1))}{P(x(k-1) \mid u(0 : k - 2), y(1 : k - 1))} \right)} \]

Motion model: commanded motion moved from robot \( x(k-1) \) to \( x(k) \)

Sensor model: robot perceives \( y(k) \) given a map and that it is at \( x(k) \)

Features

- Generalizes beyond Gaussians
- Recursive Nature

Issues

- Realization of sensor and motion models
- Representations of distributions

RI 16-735, Howie Choset
Prediction Step

• Occurs when an odometry measurement (like a control) or when a control is invoked…. Something with $u(k-1)$

• Suppose $u(0: k-2)$ and $y(1: k-1)$ known and

Current belief is $P(x(k-1) \mid u(0 : k - 2), y(1 : k - 1))$

• Obtain

$$P(x(k) \mid u(0 : k - 1), y(1 : k - 1))$$

  – Integrate/sum over all possible $x(k-1)$
  – Multiply each

$$P(x(k-1) \mid u(0 : k - 2), y(1 : k - 1))$$

by

$$P(x(k) \mid u(k - 1), x(k - 1))$$

$$= \sum_{x(k-1) \in X} \left( P(x(k) \mid u(k - 1), x(k - 1)) \right)$$

$$P(x(k - 1) \mid u(0 : k - 2), y(1 : k - 1))$$
Update Step

• Whenever a sensory experience occurs… something with $y(k)$

• Suppose $P(x(k) \mid u(0 : k - 1), y(1 : k - 1))$ is known and we just had sensor $y(k)$

• For each state $x(k)$

  Multiply $P(x(k) \mid u(0 : k - 1), y(1 : k - 1))$ by $P(y(k) \mid x(k))$

  $$P(x(k) \mid u(0 : k - 1), y(1 : k))$$

  $$= P(y(k) \mid x(k)) P(x(k) \mid u(0 : k - 1), y(1 : k - 1))$$
That pesky normalization factor

• Bayes rule gives us

\[ \eta(k) = P(y(k) \mid u(0 : k - 1), y(1 : k - 1))^{-1} \]

• This is hard to compute:
  – What is the dependency of \( y(k) \) on previous controls and sensor readings without knowing your position or map of the world?

\[ \eta(k) = \left[ \sum_{x(k) \in X} P(y(k) \mid x(k)) P(x(k) \mid u(0 : k - 1), y(1 : k - 1)) \right]^{-1} \]

• We know these terms
Summary

\[ P(x(k) \mid u(0: k-1), y(1: k)) \]

\[ = \eta(k) P(y(k) \mid x(k)) \sum_{x(k-1) \in X} \left( P(x(k) \mid u(k-1), x(k-1)) P(x(k-1) \mid u(0: k-2), y(1: k-1)) \right) \]

**prediction:**

\[ P(x(k) \mid u(0: k-1), y(1: k-1)) \]

\[ = \sum_{x(k-1) \in X} \left( P(x(k) \mid u(k-1), x(k-1)) \right) \]

\[ P(x(k-1) \mid u(0: k-2), y(1: k-1)) \]

**update:**

\[ \eta(k) \]

\[ = \left[ \sum_{x(k) \in X} P(y(k) \mid x(k)) P(x(k) \mid u(0: k-1), y(1: k-1)) \right]^{-1} \]

\[ P(x(k) \mid u(0: k-1), y(1: k)) \]

\[ = \eta(k) P(y(k) \mid x(k)) P(x(k) \mid u(0: k-1), y(1: k-1)). \]
Issues to be resolved

• Initial distribution $P(0)$
  – Gaussian if you have a good idea
  – Uniform if you have no idea
  – Whatever you want if you have some idea

• How to represent distributions: prior & posterior, sensor & motion models

• How to compute conditional probabilities
  \[ P(x(k) \mid u(k - 1), x(k - 1)) \quad P(y(k) \mid x(k)) \]

• Where does this all come from? (we will do that first)
The derivation: \( P(x(k) \mid u(0 : k - 1), y(1 : k)) \)

- Consider odometry and sensor information separately

- Let's start with new sensor reading comes in – a new \( y(k) \)
  - Assume \( y(1:k-1) \) and \( u(0:k-1) \) as known
  - Apply Bayes rule

\[
P(x(k) \mid u(0 : k - 1), y(1 : k)) \\
= \eta P(y(k) \mid x(k)) P(x(k) \mid u(0 : k - 1), y(1 : k - 1))
\]

Once state is known, then all previous controls and measurements are independent of current reading

Denominator is a normalizer which is the same for all of \( x(k) \)
Incorporate motions

• We have

\[
P(x(k) \mid u(0 : k - 1), y(1 : k)) \\
= \eta(k) \ P(y(k) \mid x(k)) \ P(x(k) \mid u(0 : k - 1), y(1 : k - 1))
\]

• Use law of total probability on right-most term

\[
P(x(k) \mid u(0 : k - 1), y(1 : k - 1)) \\
= \sum_{x(k-1) \in X} P(x(k) \mid u(k - 1), x(k - 1)) \\
P(x(k - 1) \mid u(0 : k - 2), y(1 : k - 1))
\]

assume that \(x(k)\) is independent of sensor readings \(y(1:k-1)\) and controls \(u(1:k-2)\) that got the robot to state \(x(k-1)\) given we know the robot is at state \(x(k-1)\)

assume controls at \(k-1\) take robot from \(x(k-1)\) to \(x(k)\), which we don’t know \(x(k)\) \(x(k-1)\) is independent of \(u(k-1)\)

RI 16-735, Howie Choset
Incorporate motions

• We have

\[
P(x(k) \mid u(0 : k - 1), y(1 : k)) = \eta(k) P(y(k) \mid x(k)) \sum_{x(k-1) \in X} \left( P(x(k) \mid u(k - 1), x(k - 1)) \right)
\]

\[
P(x(k - 1) \mid u(0 : k - 2), y(1 : k - 1))
\]
Representations of Distributions

• Kalman Filters

• Discrete Approximations

• Particle Filters
Extended Kalman Filters

\[ P(x(k) \mid u(0 : k - 1), y(1 : k)) \] as a Gaussian

**The Good**
- Computationally efficient
- Easy to implement

**The Bad**
- Linear updates
- Unimodal
Discretizations

- Topological structures

- Grids

Spatial 10-30cm
Angular 2-10 degrees
Algorithm to Update Posterior $P(x)$

Start with $u(0:k-1)$ and $y(1:k)$

1: $P(x) \leftarrow P(x(0))$
2: for $i \leftarrow 1$ to $k$ do
3:      for all states $x \in X$ do
4:         $P'(x) \leftarrow \sum_{x' \in X} P(x \mid u(i-1), x') \cdot P(x')$
5:      end for
6:      $\eta \leftarrow 0$
7:      for all states $x \in X$ do
8:         $P(x) \leftarrow P(y(i) \mid x) \cdot P'(x)$
9:      $\eta \leftarrow \eta + P(x)$
10: end for
11: for all states $x \in X$ do
12:    $P(x) \leftarrow P(x)/\eta$
13: end for

k loops
Integrate $u(i-1)$ and $y(i-1)$ in each loop
Incorporate motion $u(i-1)$ with motion model
Sensor model
Normalization Constant

Complexity $O(n^2)$
Bypass with convolution details we will skip

RI 16-735, Howie Choset
Convolution Mumbo Jumbo

- To efficiently update the belief upon robot motions, one typically assumes a bounded Gaussian model for the motion uncertainty.
- This reduces the update cost from $O(n^2)$ to $O(n)$, where $n$ is the number of states.
- The update can also be realized by shifting the data in the grid according to the measured motion.
- In a second step, the grid is then convolved using a separable Gaussian Kernel.
- Two-dimensional example:

```
1/16 1/8 1/16
1/8 1/4 1/8
1/16 1/8 1/16
```

\[\Rightarrow\]

```
1/4
1/2
1/4
```

\[+\]

```
1/4 1/2 1/4
```

- Fewer arithmetic operations
- Easier to implement
Probabilistic Action model

Continuous probability density $Bel(st)$ after moving

Darker area has higher probability.

Thrun et. al.

RI 16-735, Howie Choset
Probabilistic Sensor Model

Probabilistic sensor model for laser range finders

RI 16-735, Howie Choset
One of Wolfram et al’s Experiments

Known map

A, after 5 scans;
B, after 18 scans,
C, after 24 scans

5 scans
18 scans
24 scans

RI 16-735, Howie Choset
What do you do with this info?

- Mean, continuous but may not be meaningful

- Mode, max operator, not continuous but corresponds to a robot position

- Medians of x and y, may not correspond to a robot position too but robust to outliers
Particle Filters

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

- Monte Carlo filter, Survival of the fittest, Condensation, Bootstrap filter, Particle filter

- **Filtering:** [Rubin, 88], [Gordon et al., 93], [Kitagawa 96]
- **Computer vision:** [Isard and Blake 96, 98]
- **Dynamic Bayesian Networks:** [Kanazawa et al., 95]
Basic Idea

• Maintain a set of $N$ samples of states, $x$, and weights, $w$, in a set called $M$.

• When a new measurement, $y(k)$ comes in, the weight of particle $(x,w)$ is computed as $p(y(k)|x)$ – observation given a state

• Resample $N$ samples (with replacement) from $M$ according to weights $w$
Particle Filter Algorithm and Recursive Localization

\[
Bel(x_t) = \eta p(y_t | x_t) \int p(x_t | x_{t-1}, u_{t-1}) Bel(x_{t-1}) \, dx_{t-1}
\]

- draw \( x^i_{t-1} \) from \( Bel(x_{t-1}) \)
- draw \( x^i_t \) from \( p(x_t | x^i_{t-1}, u_{t-1}) \)

Importance factor for \( x^i_t \):

\[
w^i_t = \frac{\text{target distribution}}{\text{proposal distribution}}
\]

\[
= \frac{\eta \ p(y_t | x_t) \ p(x_t | x_{t-1}, u_{t-1}) \ Bel \ (x_{t-1})}{p(x_t | x_{t-1}, u_{t-1}) \ Bel \ (x_{t-1})}
\]

\[\propto p(y_t | x_t)\]
Particle Filters
Sensor Information: Importance Sampling

\[
Bel(x) \leftarrow \alpha \ p(y \mid x) \ Bel^{-}(x)
\]

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

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

\[ Bel(x) \leftarrow \alpha p(y \mid x) Bel^{-}(x) \]

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

\[ Bel^{-}(x) \leftarrow \int p(x \mid u, x') Bel(x') \, dx' \]
Motion Model Reminder

Or what if robot keeps moving and there are no observations

RI 16-735, Howie Choset
Proximity Sensor Model Reminder

Laser sensor

Sonar sensor

RI 16-735, Howie Choset
Particle Filter Algorithm

1. Algorithm \texttt{particle\_filter}( \mathcal{M}_{t-1}, u_{t-1}, y_t ):

2. \( M_t = \emptyset, \quad \eta = 0 \)

3. \textbf{For} \quad i = 1 \ldots n \quad \textit{Generate new samples}

4. Sample index \( j(i) \) from the discrete distribution given by \( M_{t-1} \)

5. Sample \( x_t^i \) from \( p(x_t | x_{t-1}, u_{t-1}) \) using \( x_t^{j(i)} \) and \( u_{t-1} \)

6. \( w_t^i = p(y_t | x_t^i) \) \quad \textit{Compute importance weight}

7. \( \eta = \eta + w_t^i \) \quad \textit{Update normalization factor}

8. \( M_t = M_t \cup \{< x_t^i, w_t^i >\} \) \quad \textit{Insert}

9. \textbf{For} \quad i = 1 \ldots n

10. \( w_t^i = w_t^i / \eta \) \quad \textit{Normalize weights}

11. \textbf{RESAMPLE!!!}
Resampling

• **Given**: Set $M$ of weighted samples.

• **Wanted**: Random sample, where the probability of drawing $x_i$ is given by $w_i$.

• Typically done $N$ times with replacement to generate new sample set $M'$. 
Resampling Algorithm

1. Algorithm **systematic_resampling**\((M, n)\):

2. \(M' = \emptyset, c_1 = w^1\)

3. For \(i = 2 \ldots n\)
   - **Generate cdf**
   - \(c_i = c_{i-1} + w^i\)

4. \(u_1 \sim U\[0, n^{-1}\], i = 1\)
   - **Initialize threshold**

5. For \(j = 1 \ldots n\)
   - **Draw samples …**
   - While \(u_j > c_i\) \(i = i + 1\)

6. \(M' = M' \cup \{< x^i, n^{-1} >\}\)
   - **Insert**

7. \(u_{j+1} = u_j + n^{-1}\)
   - **Increment threshold**

8. **Return** \(M'\)
Resampling, an analogy Wolfram likes

- Roulette wheel
- Binary search, $n \log n$

- Stochastic universal sampling
- Systematic resampling
- Linear time complexity
- Easy to implement, low variance

RI 16-735, Howie Choset
Initial Distribution

RI 16-735, Howie Choset
After Incorporating Ten Ultrasound Scans
After Incorporating 65 Ultrasound Scans
Limitations

• The approach described so far is able to
  – track the pose of a mobile robot and to
  – globally localize the robot.

• How can we deal with localization errors (i.e., the kidnapped robot problem)?
Approaches

• Randomly insert samples (the robot can be teleported at any point in time).
• Insert random samples proportional to the average likelihood of the particles (the robot has been teleported with higher probability when the likelihood of its observations drops).
Summary

- Recursive Bayes Filters are a robust tool for estimating the pose of a mobile robot.
- Different implementations have been used such as discrete filters (histograms), particle filters, or Kalman filters.
- Particle filters represent the posterior by a set of weighted samples.
Change gears to MAPPI NG
Occupancy Grids [Elfes]

- In the mid 80’s Elfes starting implementing cheap ultrasonic transducers on an autonomous robot
- Because of intrinsic limitations in any sonar, it is important to compose a coherent world-model using information gained from multiple reading
Occupancy Grids Defined

- The grid stores the probability that $C_i = \text{cell}(x,y)$ is occupied $O(C_i) = P[s(C_i) = \text{OCC}](C_i)$

- Phases of Creating a Grid:
  - Collect reading generating $O(C_i)$
  - Update Occ. Grid creating a map
  - Match and Combine maps from multiple locations

Binary variable

Cell $m_l$ is occupied $P(m_l | x(1:k), y(1:k))$

Given sensor observations $y(1:k) = y(1), \ldots, y(k)$

Given robot locations $x(1:k) = x(1), \ldots, x(k)$
Bayes Rule Rules!

- Seek to find $m$ to maximize $P(m \mid x(1:k), y(1:k))$

\[
P(m \mid x(1:k), y(1:k)) = \frac{P(y(k) \mid m, x(1:k), y(1:k-1))P(m \mid x(1:k), y(1:k-1))}{P(y(k) \mid x(1:k), y(1:k-1))}
\]

Assume that current readings is independent of all previous states and readings given we know the map

\[
P(m \mid x(1:k), y(1:k)) = \frac{P(y(k) \mid m, x(k))P(m \mid x(1:k), y(1:k-1))}{P(y(k) \mid x(1:k), y(1:k-1))}
\]

Bayes rule on $P(y(k) \mid m, x(k))$

\[
P(m \mid x(1:k), y(1:k)) = \frac{P(m \mid x(k), y(k))P(y(k) \mid x(k))P(m \mid x(1:k-1), y(1:k-1))}{P(m)P(y(k) \mid x(1:k), y(1:k-1))}
\]
A cell is occupied or not

• The m

\[
P(m \mid x(1 : k), y(1 : k))
= \frac{P(m \mid x(k), y(k)) P(y(k) \mid x(k)) P(m \mid x(1 : k - 1), y(1 : k - 1))}{P(m) P(y(k) \mid x(1 : k), y(1 : k - 1))}
\]

• Or not the m

\[
P(\neg m \mid x(1 : k), y(1 : k))
= \frac{P(\neg m \mid x(k), y(k)) P(y(k) \mid x(k)) P(\neg m \mid x(1 : k - 1), y(1 : k - 1))}{P(\neg m) P(y(k) \mid x(1 : k), y(1 : k - 1))}
\]

\[
\frac{P(m \mid x(1 : k), y(1 : k))}{1 - P(m \mid x(1 : k), y(1 : k))} = \frac{P(m \mid x(k), y(k))}{1 - P(m \mid x(k), y(k))} \quad 1 - P(m) \quad \frac{P(m \mid x(1 : k - 1), y(1 : k - 1))}{1 - P(m \mid x(1 : k - 1), y(1 : k - 1))}
\]

\[
P(\neg A) = 1 - P(A)
\]
The Odds

\[ \text{Odds}(x) = \frac{P(x)}{1 - P(x)}\]

\[
\begin{align*}
\frac{P(m \mid x(k), y(k))}{1 - P(m \mid x(k), y(k))} & \cdot \frac{1 - P(m)}{P(m)} \\
& \cdot \frac{P(m \mid x(1 : k - 1), y(1 : k - 1))}{1 - P(m \mid x(1 : k - 1), y(1 : k - 1))} \\
& = \text{Odds}(m \mid x(k), y(k)) \cdot \text{Odds}(m \mid x(1 : k - 1), y(1 : k - 1)) \\
& \cdot \text{Odds}(m)
\end{align*}
\]

\[ \log \text{Odds}(m \mid x(1 : k), y(1 : k)) \]

\[ \quad = \log \frac{\text{Odds}(m \mid x(k), y(k))}{\text{Odds}(m)} - \log \text{Odds}(m) \]

\[ + \log \text{Odds}(m \mid x(1 : k - 1), y(1 : k - 1)) \]

RECURSION

RI 16-735, Howie Choset
Recover Probability

\[
P(x) = \frac{\text{Odds}(x)}{1 + \text{Odds}(x)} = \left[1 + \frac{1}{\text{Odds}(x)}\right]^{-1}
\]

\[
P(m \mid x(1 : k), y(1 : k)) = \left[1 + \frac{\text{Odds}(m)}{\text{Odds}(m \mid x(k), y(k)) \cdot \text{Odds}(m \mid x(1 : k - 1), y(1 : k - 1))}\right]^{-1}
\]

\[
= \left[1 + \frac{1 - P(m \mid x(k), y(k))}{P(m \mid x(k), y(k))} \cdot \frac{P(m)}{1 - P(m)} \cdot \frac{1 - P(m \mid x(1 : k - 1), y(1 : k - 1))}{P(m \mid x(1 : k - 1), y(1 : k - 1))}\right]^{-1}
\]

Given a sequence of measurements \(y(1:k)\), known positions \(x(1:k)\), and an initial distribution \(P_0(m)\)

**Determine**

\[
P_m = P(m \mid x(1 : k), y(1 : k))
\]

\[
P_m \leftarrow P_0(m)
\]

for \(i \leftarrow 1 \) to \(k\) do

\[
P_m \leftarrow \left[1 + \frac{1 - P(m \mid x(i), y(i))}{P(m \mid x(i), y(i))} \cdot \frac{P(m)}{1 - P(m)} \cdot \frac{1 - P_m}{P_m}\right]^{-1}
\]

end for
Actual Computation of $P(m \mid x(k), y(k))$

- Big Assumption: All Cells are Independent

$$P(m) = \prod_{i} P(m_i)$$

- Now, we can update just a cell

$$P(m_{l} \mid x(k), y(k)) = P(m_{d,\theta}(x(k)) \mid y(k), x(k))$$

$P(m_{d,\theta}(x(k)) \mid y(k), x(k)) = P(m_{d,\theta}(x(k)))$

$$+ \begin{cases} 
- s(y(k), \theta) & d < y(k) - d_1 \\
-s(y(k), \theta) + \frac{s(y(k), \theta)}{d_1} (d - y(k) + d_1) & d < y(k) + d_1 \\
s(y(k), \theta) & d < y(k) + d_2 \\
s(y(k), \theta) - \frac{s(y(k), \theta)}{d_3 - d_2} (d - y(k) - d_2) & d < y(k) + d_3 \\
0 & \text{otherwise.}
\end{cases}$$

 Depends on current cell, distance to cell and angle to central axis
More details on $s$

\[
P(m_{d,\theta}(x(k)) \mid y(k), x(k)) = P(m_{d,\theta}(x(k)))
\]

\[
+ \begin{cases} 
  -s(y(k), \theta) & d < y(k) - d_1 \\
  -s(y(k), \theta) + \frac{s(y(k), \theta)}{d_1} (d - y(k) + d_1) & d < y(k) + d_1 \\
  s(y(k), \theta) & d < y(k) + d_2 \\
  s(y(k), \theta) - \frac{s(y(k), \theta)}{d_3 - d_2} (d - y(k) - d_2) & d < y(k) + d_3 \\
  0 & \text{otherwise.}
\end{cases}
\]

Deviation from occupancy probability from the prior given a reading and angle

\[
s(y(k), \theta) = g(y(k)) \mathcal{N}(0, \sigma_\theta)
\]
Break it down

• $d_1, d_2, d_3$ specify the intervals

• Between the arc and current location, lower probability

\[ d < \hat{y}(k) - d_1 \quad P(m_1) - s(y(k), \theta) \]

• Cells close to the arc, ie. Whose distances are close to readings

\[ \hat{y}(k) - d_1 \leq d < y(k) + d_1 \quad \text{Some linear function} \]

• Immediately behind the cell (obstacles have thickness)

\[ y(k) + d_1 \quad < d < \quad y(k) + d_2 \quad P(m_1) + s(y(k), \theta) \]

• No news is no news \( P(m_{d,\theta}(x(k)) \mid y(k), x(k)) \) is prior beyond
Example \( P(m_{d,\theta}(x(k)) \mid y(k), x(k)) \)

\[ y(k) = 2m, \text{ angle} = 0, s(2m,0) = .16 \]
Example

\[ P(m_{d,\theta}(x(k)) \mid y(k), x(k)) \]

\[
\begin{align*}
\text{y(k) = 2m} & \quad \text{y(k) = 2.5m}
\end{align*}
\]
A Wolfram Mapping Experiment
with a B21r with 24 sonars

18 scans, note each scan looks a bit uncertain but result starts to look like parallel walls

RI 16-735, Howie Choset
Are we independent?

• Is this a bad assumption?
SLAM!

- A recursive process.

\[
P(x(1 : k), m | u(0 : k - 1), y(1 : k)) = \alpha \int \left( P(x(k) | u(k - 1), x(k - 1), x(1 : k - 1)) P(x(1 : k - 1), m | u(0 : k - 2), y(1 : k - 1)) \right) dx(1 : k - 1)
\]

Motion model

Sensor model

Posterior, hard to calculate
“Scan Matching”

At time $k - 1$ the robot is given

1. An estimate $\hat{x}(k - 1)$ of state

2. A map estimate $\hat{m}(\hat{x}(1 : k - 1), y(1 : k - 1))$

The robot then moves and takes measurement $y(k)$

And robot chooses state estimate which maximizes

$$\hat{x}(k) = \arg\max_{x(k)} \left\{ P(y(k) \mid x(k), \hat{m}(\hat{x}(1 : k - 1), y(1 : k - 1))) \right\}.$$

And then the map is updated with the new sensor reading
Another Wolfram Experiment

28m x 28m, .19m/s, 491m

RI 16-735, Howie Choset
Another Wolfram Experiment

before

after

28m x 28m, .19m/s, 491m

RI 16-735, Howie Choset
Tech Museum, San Jose

CAD map

occupancy grid map

RI 16-735, Howie Choset
Issues

• Greedy maximization step (unimodal)
• Computational burden (post-processing)
• Inconsistency (closing the loop, global map?)

Solutions [still maintain one map, but update at loop closing]
• Grid-based technique (Konolodige et. al)
• Particle Filtering (Thrun et. al., Murphy et. al.)
• Topological/Hybrid approaches (Kuipers et. al, Leonard et al, Choset et a.)
Probabilistic SLAM
Rao-Blackwell Particle Filtering

If we know the map, then it is a localization problem
If we know the landmarks, then it is a mapping problem

Some intuition: if we know \( x(1:k) \) (not \( x(0) \)), then we know the “relative map” but Not its global coordinates

The promise: once path \( (x(1:k)) \) is known, then map can be determined analytically

Find the path, then find the map
Mapping with Rao-Blackwellized Particle Filters

• **Observation:**
  Given the true trajectory of the robot, all measurements are independent.

• **Idea:**
  – Use a particle filter to represent potential trajectories of the robot (multiple hypotheses). Each particle is a path (maintain posterior of paths)
  
  – For each particle we can compute the map of the environment (mapping with known poses).

  – Each particle survives with a probability that is proportional to the likelihood of the observation given that particle and its map.

[Murphy et al., 99]
RBPF with Grid Maps

map of particle 1

map of particle 2

map of particle 3

3 particles
Some derivation

\[ P(x(1:k), m \mid u(0:k-1), y(1:k)) \]

\[ P(x(1:k), m \mid u(0:k-1), y(1:k)) = P(m \mid x(1:k), y(1:k), u(0:k-1)) \]
\[ \cdot P(x(1:k) \mid y(1:k), u(0:k-1)). \]

\[ P(m \mid x(1:k), y(1:k), u(0:k-1)) = P(m \mid x(1:k), y(1:k)) \]
\[ \text{m is independent of } u(0:k-1) \text{ given } x(1:k) \]

\[ P(x(1:k), m \mid u(0:k-1), y(1:k)) = \frac{P(m \mid x(1:k), y(1:k))}{P(x(1:k) \mid y(1:k), u(0:k-1))} \cdot \]

We can compute \hspace{1cm} Use particle filtering

Computing prob map (local map) given trajectory for each particle

RI 16-735, Howie Choset
Methodology

• $M$ be a set of particles where each particle starts at $[0,0,0]^T$

• Let $h^{(j)}(1:k)$ be the $j$th path or particle

• Once the path is known, we can compute most likely map

$$m^{(j)}(1:k - 1) = \arg\max_m P(m | h^{(j)}(1:k), y(1:k - 1))$$

Hands start waving….. Just a threshold here

• Once a new $u(i-1)$ is received (we move), do same thing as in localization, i.e., sample from $P(x | x_j, u(i - 1))$.

Not an issue, but in book
– Note, really sampling from $P(x | x_j, u(i - 1), m^{(j)}(1:k - 1))$
– Ignore the map for efficiency purposes, so drop the $m$

• Get our $y(k)$’s to determine weights, and away we go (use same sensor model as in localization)
Rao-Blackwell Particle Filtering

**Input:** Sequence of measurements $y(1:k)$ and movements $u(0:k-1)$ and set $\mathcal{M}$ of $N$ samples $(x_j, \omega_j)$

**Output:** Posterior $P(x(1:k), m | u(0:k-1), y(1:k))$ represented by $\mathcal{M}$ about the path of the robot at time and the map

\[
\begin{align*}
\text{for } j &\leftarrow 1 \text{ to } N \text{ do} \\
x_j &\leftarrow (0, 0, 0) \\
\text{end for} \\
\text{for } i &\leftarrow 1 \text{ to } k \text{ do} \\
\text{for } j &\leftarrow 1 \text{ to } N \text{ do} \\
&\text{compute a new state } x \text{ by sampling according to } P(x | u(i-1), x_j). \\
x_j &\leftarrow x \\
\text{end for} \\
\eta &\leftarrow 0 \\
\text{for } j &\leftarrow 1 \text{ to } N \text{ do} \\
&\text{compute } w_j = P(y(i) | x_j, m^{(j)}(1:i-1))) \\
&\eta = \eta + w_j \\
\text{end for} \\
\text{for } j &\leftarrow 1 \text{ to } N \text{ do} \\
&w_j = \eta^{-1} \cdot w_j \\
\text{end for} \\
\mathcal{M} &\leftarrow \text{resample}(\mathcal{M}) \\
\text{end for}
\end{align*}
\]

\[
P(x(1:k), m | u(0:k-1), y(1:k)) \\
= P(m | x(1:k), y(1:k)) \cdot P(x(1:k) | y(1:k), u(0:k-1)).
\]
Wolfram Experiment
Most Recent Implementations

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

Courtesy by Giorgio Grisetti & Cyrill Stachniss
Maps, space vs. time

Maintain a map for each particle

OR

Compute the map each time from scratch

Subject of research
    Montermerlou and Thrun look for tree-like structures that capture commonality among particles.

    Hahnel, Burgard, and Thrun use recent map and subsample sensory experiences
How many particles?

- What does one mean?
- What does an infinite number mean?