Efficient Incremental Smoothing
SLAM Tutorial @ ICRA 2016

Michael Kaess

May 20, 2016
The SLAM Problem (t=0)
The SLAM Problem (t=1)

Odometry measurement

Robot

Landmark measurement

Landmark 1

Landmark 2
The SLAM Problem \((t=n-1)\)

Odometry measurement

Landmark measurement

Landmark 1  Landmark 2
The SLAM Problem (t=n)

Odometry measurement

Landmark measurement
Factor Graph Representation of SLAM

Bipartite graph with **variable nodes** and **factor nodes**

Odometry measurement

Landmark measurement
Factor Graph Representation of SLAM

Bipartite graph with **variable nodes** and **factor nodes**

Odometry measurement

Loop closing constraint

Landmark measurement

Landmark position

Robot pose
Variables and Measurements

• Variables:

\[ \Theta = \{ x_0, x_1 \ldots x_n, l_1, l_2 \} \]

Might include other quantities such as lines, planes and calibration parameters

• Measurements:

\[ Z = \{ p, u_1 \ldots u_n, m_1 \ldots m_4 \} \]

\( p \) is a prior to fix the gauge freedom (all other measurements are relative!)
SLAM as a **Sparse Least-Squares Problem**

\[
\arg\max_\Theta \prod_{z \in Z} p(z | \Theta)
\]

Gaussian noise

\[
\arg\min_\Theta \sum_i \| h_i(\Theta) - z \|^2
\]

Sparse measurement Jacobian

\[
\arg\min_\Theta \| A\Theta - b \|^2
\]
Incremental Smoothing and Mapping (iSAM)

Solving a growing system:

- R factor from previous step
- How do we add new measurements?

Key idea:

- Append to existing matrix factorization
- “Repair” using Givens rotations

\[
\begin{bmatrix}
1 & c \\
-s & c
\end{bmatrix} \cdot \begin{bmatrix}
x \\
0
\end{bmatrix} = \begin{bmatrix}
1 \\
1
\end{bmatrix}
\]
Matrix vs. Graph

Measurement Jacobian

Factor Graph

Information Matrix

Markov Random Field

Square Root Inf. Matrix
Matrix vs. Graph

Measurement Jacobian
Factor Graph

Information Matrix
Markov Random Field

Square Root Inf. Matrix
Bayes Tree
Matrix vs. Graph

Matrix factorization

A

ATA

R
Matrix vs. Graph

Matrix factorization

Variable elimination
iSAM2: Bayes Tree

Goal: Convert factor graph to tree structure

Why? Inference in tree structure is easy!

Two stage process:
1. Variable elimination converts factor graph to Bayes net
2. Discovering cliques provides Bayes tree

“iSAM2: Incremental Smoothing and Mapping Using the Bayes Tree” M. Kaess et al., IJRR 2012
Variable Elimination – Example

- Choose ordering: $l_1, l_2, x_1, x_2, x_3$
- Eliminate one node at a time

\[
p(l_1, x_1, x_2) = p(l_1 | x_1, x_2) \cdot p(x_1, x_2)
\]
Variable Elimination – Example

- Choose ordering: $l_1, l_2, x_1, x_2, x_3$
- Eliminate one node at a time

\[
p(l_1|x_1,x_2)\]

\[
p(x_1,x_2)\]

\[
p(l_1,x_1,x_2) = p(l_1|x_1,x_2) p(x_1,x_2)\]
Variable Elimination – Example

- Choose ordering: $l_1, l_2, x_1, x_2, x_3$
- Eliminate one node at a time

$$p(l_2, x_3) = p(l_2 | x_3) \cdot p(x_3)$$
Variable Elimination – Example

• Choose ordering: $l_1, l_2, x_1, x_2, x_3$
• Eliminate one node at a time

$p(x_1, x_2) = p(x_1 | x_2) \cdot p(x_2)$
Variable Elimination – Example

- Choose ordering: $l_1, l_2, x_1, x_2, x_3$
- Eliminate one node at a time

$$p(x_2, x_3) = p(x_2 | x_3) \ p(x_3)$$
Variable Elimination – Example

- Choose ordering: $l_1, l_2, x_1, x_2, x_3$
- Eliminate one node at a time
Bayes Tree Data Structure

Step 1

The Bayes net has a special property: its undirected equivalent is chordal by construction.

Chordal: No cycle greater than 3 that has no shortcut
Bayes Tree Data Structure

Step 1

Step 2: Find cliques in reverse elimination order:
Bayes Tree Data Structure

Step 1

Step 2: Find cliques in reverse elimination order:

$x_2, x_3$
Bayes Tree Data Structure

Step 1

Step 2: Find cliques in reverse elimination order:

\[ x_2, x_3 \]

\[ x_1 : x_2 \]
Bayes Tree Data Structure

Step 1

Step 2: Find cliques in reverse elimination order:

$x_2, x_3$

$x_1 : x_2$

$l_2 : x_3$
Bayes Tree Data Structure

Step 1

Step 2: Find cliques in reverse elimination order:
Bayes Tree Data Structure

Step 1

Step 2: Find cliques in reverse elimination order:

\[
P(x_j|S_j) \propto \exp \left\{ -\frac{1}{2\sigma^2} (x_j + rS_j - d)^2 \right\}
\]
Backsubstitution in the Graph

- Inference is a two step process:
  - Elimination starts at leaves and proceeds to the root
  - Solving starts at root and proceeds to the leaves
iSAM2: Bayes Tree Example

Complexity depends on the size of the largest clique

Manhattan dataset (Olson)
iSAM2: Bayes Tree Example

How to update with new measurements / add variables?

Manhattan dataset (Olson)

[Kaess et al., IJRR 12]
iSAM2: Updating the Bayes Tree

Add new factor between $x_1$ and $x_3$
iSAM2: Updating the Bayes Tree

Add new factor between $x_1$ and $x_3$
iSAM2: Updating the Bayes Tree

Add new factor between $x_1$ and $x_3$
iSAM2: Updating the Bayes Tree

Add new factor between $x_1$ and $x_3$
Incremental Variable Reordering

For a small loop, what constitutes a “good” ordering?

Include loop closing into cut

Loop closing not part of cut

Include loop closing into cut

Loop closing not part of cut
Incremental Variable Reordering

Most recent variable at the end expected to make future updates cheaper

- Force most recent variables to the end
- Find best ordering for remaining variables

Using constrained version of COLAMD algorithm (CCOLAMD)

[Kaess et al., IJRR 12]
Variable Reordering – Constrained COLAMD

Greedy approach
Arbitrary placement of newest variable

Constrained Ordering
Newest variables forced to the end

Number of affected variables:
low                                           high

Much cheaper!
iSAM2: Incremental Update + Variable Ordering

Variable ordering changes incrementally during update
- Not understood in matrix version
- Sparse matrix data structure not suitable

Large savings in computation
Variable Reordering – Fill-in

Incremental ordering still yields good overall ordering

- Only slightly more fill-in than batch COLAMD ordering
- Constrained ordering is worse than naïve/greedy:
  - Suboptimal ordering because of partial constraint, but cheaper to update!
Again good quality and low cost are achievable:
iSAM2: Bayes Tree for Manhattan Sequence
Conclusion

• Exploit temporal structure
• Efficient incremental nonlinear least-squares solution

• Requirements:
  – Sparse graph
  – Good initial estimates