Concurrent Filtering and Smoothing

Michael Kaess\textsuperscript{1}, Stephen Williams\textsuperscript{2}, Vadim Indelman\textsuperscript{2}, Richard Roberts\textsuperscript{2}, John Leonard\textsuperscript{1}, Frank Dellaert\textsuperscript{2}

1) CSAIL, Massachusetts Institute of Technology (MIT)
2) College of Computing, Georgia Institute of Technology
Navigation

- Tracking position, velocity and orientation of an observer using IMU, GPS, LBL (acoustic), wind speed, camera...

- How to solve? Depends on who you ask!

Mars Rover
[JPL]

Aircraft
[wikipedia]

Underwater
[MIT]
Navigation Community: Filtering

- Established, well-tested solution in Aerospace etc.
- Estimate current state
- Objective:

\[
\hat{x}_t = \arg \max_{x_t} p(x_t | Z)
\]

- Update:

\[
p(x_{t+1} | Z) = \int_{x_t} p(x_{t+1}, x_t | Z)
\]
Robots Community: Smoothing

- Full SLAM (Simultaneous Localization and Mapping)
- Estimate all states, current and past
- Objective:

$$\hat{X}_t = \arg \max_{X_t} p(X_t|Z)$$

- Update:

$$p(X_{t+1}|Z) = p(x_{t+1}, X_t|Z)$$

Map of Intel Labs
### Concurrent Filtering and Smoothing

**Filtering vs. Smoothing**

<table>
<thead>
<tr>
<th>Filtering</th>
<th>Smoothing</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
</tr>
</tbody>
</table>

- Constant high frame rate
- Only current state is tracked

- Allows loop closure
- No constant time guarantee

Can we combine the advantages of both methods?
Filtering and Smoothing

• Can we combine the advantages of both methods?

• Goals:
  – Perform smoothing in a separate, asynchronous process
  – Maintain real-time performance of the filtering process
    • Minimize calculations of any required synchronization
  – Produce the optimal Bayesian solution
Related Work

• **Combining filtering and loop closing** [Eustice, Singh, and Leonard 2006]
  – Uses an augmented state filter to allow loop closures
  – Maintains real-time performance
  – Produces only an approximate solution

• **Parallel tracking and mapping** [Klein and Murray 2007] [Newcombe et al. 2011]
  – Performs Bundle Adjustment (BA) in a separate process
  – Relocalizes after BA instead of fusing results
  – Does not incorporate additional navigation sensors

• **Dual-layer estimator** [Mourikis and Roumeliotis 2008]
  – Combines an EKF with BA
  – EKF must be rolled back to incorporate the BA update in a consistent manner
Filtering and Smoothing (Cont.)

- In this work:
  - We perform smoothing and filtering in parallel
    - High-rate measurements are processed by the filter
    - Loop closures are added directly to the smoother
  - Smoothing and filtering are considered two components of a single optimization problem
    - Ensures the optimal Bayesian estimate is obtained
  - The problem is represented using a Bayes tree
    - Intuitive graphical model
    - Exploits sparsity
    - Allows incremental inference
Bayes Tree

- The maximum a posteriori (MAP) estimate is given by

\[ \hat{X} = \arg \max_X (p(X)) \]

- Applying the chain rule yields a factorization:

\[ p(X) = \prod_i p(X_i | S_i) \]

  - where \( X_i, S_i \subset X \)

- Different factorizations exist, depending on the order in which variables are chosen

- Given a particular factorization, a unique Bayes tree can be constructed
  - Each node represents a conditional distribution
  - Each node is conditioned only on its ancestors in the tree
  - Solving for \( \hat{X} \) involves applying the chain rule, starting from the root
• The maximum a posteriori (MAP) estimate is given by

\[ \hat{X} = \arg \max (p(X)) \]

• Applying the chain rule yields a factorization:

\[ p(x_2, x_3) \]

• Different factorizations exist, depending on the order in which variables are chosen.

• Given a particular factorization:
  – Each node represents a conditional distribution.
  – Each node is conditioned only on its ancestors in the tree.
  – Solving for involves applying the chain rule, starting from the root.

Bayes Tree

The maximum a posteriori (MAP) estimate is given by

\[ \hat{X} = \arg \max (p(X)) \]
Bayes Tree – Gaussian Distribution

- Bayes tree corresponds to the square root information matrix \( \mathcal{I} = R^T R \)

\[
R = \begin{bmatrix}
  \times & \times & \times & \times & \times & \times \\
  \times & \times & \times & \times & \times \\
  \times & \times & \times & \times \\
  \times & \times \\
  \times \\
  \times \\
\end{bmatrix}
\]

\( x_6 \quad x_5 \quad x_1 \quad x_4 \quad x_3 \quad x_2 \)

- **Factorization/Elimination** (i.e. calculation of R) corresponds to constructing the Bayes tree
  - Performed from bottom upwards

- Solving for \( \hat{X} \):
  - Performed by **back-substitution**, from root of Bayes tree downwards
**Efficient Bayes Tree Updates** [Kaess et al IJRR 12]

- **Key Insights**
  - **Incorporating new measurements can be done efficiently**
  - Affects only variables involved in the measurement model and their ancestors
    - Only affects variables in the path to the top of the tree
    - Branches remain unaffected
Efficient Bayes Tree Updates [Kaess et al IJRR 12]

- **Key Insights**
  - Many variable orderings exist
  - Ordering affects:
    - Tree structure - different factorization of \( p(X) = \prod_i p(X_i|S_i) \)
    - Number of variables in each node / Computational complexity
    - Does not affect the solution
Parallelizing Filtering and Smoothing

• Factorization based on suitable variable ordering:

\[ p(X) = p(X_R | X_s) p(X_s) p(X_t | X_s) \]

• Corresponding Bayes tree:

• Allows concurrent updates to filter and smoother!
Parallelizing Filtering and Smoothing

- Filter and smoother are kept periodically synchronized
  - Information flows between the smoother and the filter via the separator

\[
p(X_s) \quad \text{Separator}
\]
\[
p(X_R | X_s) \quad \text{Smoothen}
\]
\[
p(X_t | X_s) \quad \text{Filter}
\]

\[
X_R : X_s
\]

\[
X_t : X_s
\]
Parallelizing Filtering and Smoothing

- Loop closure measurements are added to the smoother
  - Smoother processes new measurements in a background process
  - Upon completion, the separator (root of the Bayes tree) is updated
  - Updates from the root are propagated to the filter in a fast process
- Involves only a small number of variables
Parallelizing Filtering and Smoothing

- **High-rate measurements are added to the filter**
  - Filter processes measurements in real time
    - Separator (root) is updated as part of this process
  - The separator accumulates updates
  - When the smoother is available:
    - Propagate these updates to the smoother
    - Performed in a background process

How to move variables from the filter to the smoother?

Back-substitution

Elimination
Moving Variables to the Smoother

- As time progresses, the filter maintains a sparse set of variables
- These variables form a chain of nodes on the filter side of the Bayes tree
- Whenever the smoother is available:
  - The current separator (root) node is moved to the smoother
  - A new separator is formed from the chain on the filter side
  - This is achieved by choosing a new variable ordering for these two nodes

![Diagram showing the process of moving variables to the smoother](image-url)
Evaluation: Simulation

- Simulated flight of an aerial vehicle
  - Velocity: 40 m/s
  - Constant height: 200 m above mean ground level
  - Ground elevation: ±50 m

- Synthetic measurements of different sensors
  - IMU at 100 Hz
  - Stereo camera at 0.5 Hz
    - Produces relative pose measurements
  - Sparse loop closures found in camera data
    - 22 loop closure events identified
    - Provided directly to the smoother
Evaluation: Positional Error

- Compared methods:
  - Filter alone
  - Concurrent Filter and Smoother (Our approach)
  - Incremental Batch

![Error plots](image)
Evaluation: Rotational Error

- Compared methods:
  - Filter alone
  - Concurrent Filter and Smoother (Our approach)
  - Incremental Batch
Evaluation: Velocity Error

- Compared methods:
  - **Filter alone**
  - **Concurrent Filter and Smoother** (Our approach)
  - **Incremental Batch**
Timing: Filter Update

- Filter updates in constant time
- Confirmed by evaluation:

![Bar chart showing filter update times and counts.]

# of filter updates

Filter Update Time (ms)
Timing: Synchronization

- Synchronization only affects top of tree
  - Performed after smoother is done

- Depends on size of root clique

- Evaluation shows very fast synchronization
Future Work

• Relinearization
  – Filter produces linear conditionals
  – “Lifting” of linear constraints (Konolige, TRO 2008)

• Bound complexity of smoother
  – Fixed-lag
  – Sparsification (SLAM)
Summary

• We combined filtering and smoothing
  – Parallel formulation in Bayes tree
  – Constant time filtering
  – Loop closing capability

• Tomorrow:
  “Factor Graph Based Incremental Smoothing in Inertial Navigation Systems”
  V. Indelman, S. Williams, M. Kaess, F. Dellaert