Factor Grouping for Efficient Bundle Adjustment

Tin Chon Chan
Technische Universität München
Chair of Computer Vision & AI
Master Thesis
Advisor: Nikolaus Demmel, Simon Weber
Supervisor: Prof. Dr. Daniel Cremers
Outline

- Motivation & Background
- Schur complement trick
- Factor grouping
- Square Root Bundle Adjustment
- Experiments & Analysis
Motivation

• 3D reconstruction from a set of unordered images looking at a scene
• Jointly estimate structure and camera parameters
• Bottleneck of SfM system
Background

\[ r_{ij} = u_{ij} - \pi(R_i X_j + t_i; c_i) \]

- \( r_{ij} \): residual
- \( u_{ij} \): pixel location
- \( \pi \): projection function
- \( R_i \): camera rotation
- \( t_i \): camera translation
- \( c_i \): camera intrinsic

\[ E(x_p, x_l) = \sum_{i,j} \| r_{ij} \|^2 = \| r(x_p, x_l) \|^2 \]
Background

Solve by Levenberg–Marquardt

\[
\min_x E(x) = \|r(x)\|^2
\]

\[
\min_{\Delta x} E_{\text{lin}}(\Delta x) = \|r + Jx\|^2 = r^T r + 2r^T J \Delta x + \Delta x^T H \Delta x
\]

Linearize in every LM iteration

\[J_p\]

\[J_i\]

Jacobian \(J\)

Hessian \(H\)

Tin Chon Chan | Factor Grouping for Efficient Bundle Adjustment | Informatik IX
Schur complement trick

\[
\begin{pmatrix}
H_{pp} & H_{pl} \\
H_{lp} & H_{ll}
\end{pmatrix}
\begin{pmatrix}
-\Delta x_p \\
-\Delta x_l
\end{pmatrix}
=
\begin{pmatrix}
b_p \\
b_l
\end{pmatrix}
\]

Reduced Camera System (RCS)

\[
\hat{H}_{pp}(-\Delta x_p) = \tilde{b}_p
\]

\[
\hat{H}_{pp} = H_{pp} - H_{pl}H_{ll}^{-1}H_{lp}
\]

\[
\tilde{b}_p = b_p - H_{pl}H_{ll}^{-1}b_l
\]
Schur complement trick

Hessian $H$

10 cameras
982 landmarks
Schur complement trick

Solve by Preconditioned Conjugate Gradient

\[ \hat{H}_{pp} \left( -\Delta x_p \right) = \tilde{b}_p \]

\[ \hat{H}_{pp} = H_{pp} - H_{pl}H_{ll}^{-1}H_{lp} \]

Explicit SC

\[ \hat{H}_{pp} \nu \]

Implicit SC

\[ H_{pp} \nu - H_{pl}H_{ll}^{-1}H_{lp} \nu \]

Diagonal blocks of \( \hat{H}_{pp} \) are used as preconditioner
Factor grouping

Schur complement
Factor grouping

Time complexity of matrix-vector multiplication

\#cameras = P, \#landmarks = L

Explicit: $O(P^2)$  
Implicit: $O(PL)$

(assume all cameras see all landmarks)

Explicit is better than Implicit only if $P < L$
Factor grouping

Explicit:

Implicit:

\[ P < L \]
\[ P = 2, L = 3 \]

Explicit: \( O(P^2) \)  \quad \text{Implicit: } O(PL) \]
Factor grouping

Frequent Pattern Tree

group landmarks having a common set of camera observations
Square Root Bundle Adjustment

\[ \min_{\Delta x_p, \Delta x_l} E_{lin}(\Delta x_p, \Delta x_l) = \left\| r + (J_p \ J_l) \left( \begin{array}{c} \Delta x_p \\ \Delta x_l \end{array} \right) \right\|^2 \]

Nullspace Marginalization

QR decomposition

\[ J_l = (Q_1 \ Q_2) \begin{pmatrix} R_1 \\ 0 \end{pmatrix} \]

\[ \min_{\Delta x_p} \left\| Q_2^T r + Q_2^T J_p \Delta x_p \right\|^2 \]

Project \( Q_2 \)
Square Root Bundle Adjustment ($\sqrt{BA}$)

$$\min_{\Delta x_p, \Delta x_l} E_{\text{lin}}(\Delta x_p, \Delta x_l) = \left\| r + (J_p \ J_l) \begin{pmatrix} \Delta x_p \\ \Delta x_l \end{pmatrix} \right\|^2$$

Nullspace Marginalization

Schur complement

QR decomposition

$$J_l = (Q_1 \quad Q_2) \begin{pmatrix} R_1 \\ 0 \end{pmatrix}$$

Project $Q_2$

$$\min_{\Delta x_p} \left\| Q_2^T r + Q_2^T J_p \Delta x_p \right\|^2 \quad \iff \quad \hat{H}_{pp}(-\Delta x_p) = \tilde{b}_p$$

Tin Chon Chan | Factor Grouping for Efficient Bundle Adjustment | Informatik IX
Square Root Bundle Adjustment ($\sqrt{BA}$)

$\sqrt{BA}$ landmark block

Marginalization

Preconditioned Conjugate Gradient

$\sqrt{BA}$

$\left( Q_2^T J_p \right)^T \left( Q_2^T J_p \right) v$

Schur complement

$\widehat{H}_{pp} v$
Square Root Bundle Adjustment (\(\sqrt{\text{BA}}\))

\(\sqrt{\text{BA}}\)

Marginalization

Implicit \(\sqrt{\text{BA}}\)

For every multiplication:
1. Copy to \(\sqrt{\text{BA}}\) landmark
2. Perform marginalization

Factor grouping + Implicit \(\sqrt{\text{BA}}\) → Factor \(\sqrt{\text{BA}}\)
Experiments & Analysis
Explicit and Implicit SC

Matrix-Vector Multiplication
Compute Preconditioner
Prepare RCS

Tin Chon Chan | Factor Grouping for Efficient Bundle Adjustment | Informatik IX
Explicit and Implicit SC

Explicit SC
- **Expensive** to prepare RCS ($\tilde{H}_{pp}$)
- **Cheap** to compute preconditioner
- **Cheap** multiplication

Implicit SC
- **Cheap** to prepare RCS
- **Expensive** to compute preconditioner
- **Expensive** multiplication

### Time complexity

<table>
<thead>
<tr>
<th></th>
<th>Explicit SC</th>
<th>Implicit SC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prepare RCS</td>
<td>$O(LP^2)$</td>
<td>$O(LP)$</td>
</tr>
<tr>
<td>Preconditioner</td>
<td>$O(P)$</td>
<td>$O(LP)$</td>
</tr>
<tr>
<td>Multiplication</td>
<td>$O(P^2)$</td>
<td>$O(LP)$</td>
</tr>
</tbody>
</table>

#cameras = $P$, #landmarks = $L$

$L \gg P$
Factor grouping

Matrix-Vector Multiplication
Compute Preconditioner
Prepare RCS

Tin Chon Chan | Factor Grouping for Efficient Bundle Adjustment | Informatik IX
Factor grouping

Factor SC
- Relatively cheap to prepare RCS ($\tilde{H}_{pp}$)
- Relatively cheap to compute preconditioner
- Relatively cheap multiplication

Explicit SC performed worse when given many cameras (large-scale or dense problems)

- Less cameras → Mitigate the quadratic complexity
- Improve overall multiplication runtime
Factor grouping

- sparse
- dense

- small
- large

- explicit-64
- implicit-64
- factor-64

△ fastest for that problem

Speed compared to the fastest
Square Root Bundle Adjustment (\(\sqrt{BA}\))

easier to solve, double precision
(Bundle Adjustment in the Large)

harder to solve, single precision
(1DSfM)

Tin Chon Chan | Factor Grouping for Efficient Bundle Adjustment | Informatik IX
Square Root Bundle Adjustment ($\sqrt{BA}$)

Lower accuracy

Higher accuracy

Tin Chon Chan | Factor Grouping for Efficient Bundle Adjustment | Informatik IX
\[ \sqrt{BA} \text{ and Implicit } \sqrt{BA} \]

**\[ \sqrt{BA} \]**

Recall from memory for every multiplication

**Implicit \[ \sqrt{BA} \]**

Deliciated memory block

Recall from memory for every multiplication

Recompute for every multiplication
Implicit $\sqrt{BA}$

- Write result to closer memory location (linearization)
- Recall and consume less amount of memory
- Marginalize on same memory block

Better memory access pattern
(spatial and temporal locality)
Overall performance

\[ \sqrt{IBA-64} \quad \sqrt{BA-64} \quad \sqrt{FBA-64} \quad \text{explicit-64} \quad \text{implicit-64} \quad \text{factor-64} \quad \text{power-64} \]

\[ \text{Power Bundle Adjustment for Large-Scale 3D Reconstruction} \]

\[ \text{fastest for that problem} \]
\[ \times \text{solution didn’t reach the accuracy} \]
Conclusion

- Implemented 5 solvers (highly parallelized and vectorized)
- Analyzed the performance of 7 solvers
- Analyzed time complexity and memory access pattern
- Recomputing intermediate result can be faster than recalling from memory

Recommendations

- **Implicit SC** for large-scale and dense problems
- **Factor SC** for small- and medium-scale problems
- **Implicit $\sqrt{BA}$** for solving in single precision
- **Power BA** for solving double precision and values speed over accuracy
Reference


