Incremental Light Bundle Adjustment

Vadim Indelman, Richard Roberts, Chris Beall, Frank Dellaert

College of Computing, Georgia Institute of Technology
Introduction

- Bundle Adjustment: reconstruct camera poses and structure

- Applied in a variety of applications:

  - Structure from motion [Snavely et al., 2006]
  - Augmented Reality [Klein et al., 2007]
  - Full SLAM Map of Intel Labs
  - Distributed SAM [Cunningham et al., 2010]

Top image from: http://www.tnt.uni-hannover.de/project/motionestimation
Bundle Adjustment (BA)

- A large sparse optimization problem
  - Minimization of re-projection errors between all views and observed 3D points
  - Efficient solvers exist that exploit the sparse nature of typical SfM\SLAM problems
    - SBA [Lourakis et al., 2009]
    - SSBA [Konolige, 2010]
    - iSAM2 [Kaess et al., 2012]

\[
J_{BA}\left(\hat{x}, \hat{L}\right) = \sum_{i=1}^{N} \sum_{j=1}^{M} \left\| p_i^j - \text{Proj}\left(\hat{x}_i, \hat{L}_j\right) \right\|_\Sigma^2
\]

- Assuming N cameras\images observing M 3D points
  - Number of variables to optimize: \(6N + 3M\)
  - Need to initialize both camera poses and 3D points (structure)
“Structure-Less” BA

- Camera poses are optimized without iterative structure estimation
- Cost function is based on multi-view constraints
  - Instead of minimizing re-projections errors as in conventional BA
  - 3D points are algebraically eliminated
  - Much less variables to optimize over [Rodríguez et al., 2011]!
- If required, all or some of the 3D points can be reconstructed
  - Based on the optimized camera poses

- Several structure-less BA methods have been recently developed
  - [Steffen et al., 2010], [Rodríguez et al., 2011], [Indelman, 2012]
- All methods perform **batch optimization**
Incremental Light Bundle Adjustment (iLBA)

In this work:

- We combine two key-ideas
  - **Structure-less BA:**
    - Significantly less variables to optimize over than in BA
    - Three-view constraints are used to allow consistent estimates also when camera centers are co-linear
  - **Incremental inference over graphical models:**
    - Only part of the camera poses are re-calculated
      - These cameras are systematically identified
      - Calculations from previous steps are re-used
    - Sparsity is fully exploited
    - Developed in robotics community [Kaess et al., 2012]
Structure-Less BA (SLB)

- Re-projection errors are approximated by the difference between measured and “fitted” image observations [Steffen et al., 2010], [Indelman, 2012]
  - Subject to satisfying applicable multi-view constraints
    \[ J_{SLB}(\hat{x}, \hat{p}) = \sum_{i=1}^{N} \sum_{j=1}^{M} \left\| \mathbf{p}_{i}^{j} - \hat{\mathbf{p}}_{i}^{j} \right\|^2 - 2\lambda^T h(\hat{x}, \hat{p}) \]
  - All multi-view constraints for a given sequence of view:
    \[ h \doteq \left[ h_1 ~ \ldots ~ h_{N_h} \right]^T \]
    - \( h_k \): k-th multi-view constraint
    - \( N_h \): Number of all applicable multi-view constraints for a given sequence

- Number of actual optimized variables is larger than in BA!
Light Bundle Adjustment (LBA)

- To substantially reduce computational complexity:
  - Do not make corrections to the image observations [Rodríguez et al., 2011]
- Assuming a Gaussian distribution of multi-view constraints $h_i$:
  - MAP estimate is equivalent to a non-linear least-squares optimization

Cost function:

$$J_{LBA}(\hat{x}) = \sum_{i=1}^{N_h} \|h_i(\hat{x}, p)\|^2_{\Sigma_i}$$

- $\Sigma_i$: An equivalent covariance $\Sigma_i = A_i \Sigma A_i^T$
- $A_i$: Jacobian with respect to the image observations (re-calculated each re-linearization)
- In practice: Calculate $\Sigma_i$ only once

Number of optimized variables: $6N$
LBA Using Three-View Constraints

- Algebraic elimination of a 3D point that is observed by 3 views \( k, l \) and \( m \) leads to [Indelman et al., 2012]:

\[
\begin{align*}
g_{2v}(x_k, x_l) &= q_k \cdot (t_{k \rightarrow l} \times q_l) \\
g_{2v}(x_l, x_m) &= q_l \cdot (t_{l \rightarrow m} \times q_m) \\
g_{3v}(x_k, x_l, x_m) &= (q_l \times q_k) \cdot (q_m \times t_{l \rightarrow m}) - (q_k \times t_{k \rightarrow l}) \cdot (q_m \times q_l)
\end{align*}
\]

- Necessary and sufficient conditions
- Consistent motion estimation also when camera centers are co-linear
  - In contrast to using only epipolar constraints [Rodríguez et al., 2011]
  - In robotics: reduce position errors along motion heading in straight trajectories

- LBA cost function with three-view constraints:

\[
J_{LBA}(\hat{x}) = \sum_{i=1}^{N_h} \|h_i(\hat{x}, p)\|^2_{\Sigma_i} \\
h_i \in \{g_{2v}, g_{3v}\}
\]
Incremental LBA (iLBA)

- Previous structure-less BA approaches: **batch** optimization
  - [Steffen et al., 2010], [Rodríguez et al., 2011], [Indelman, 2012]
  - Involves updating **all** camera poses each time a new image is added

\[
J_{LBA}(\hat{x}) = \sum_{i=1}^{N_h} \| h_i(\hat{x}, p) \|_{\Sigma_i}^2 \\
J_{SLB}(\hat{x}, \hat{p}) = \sum_{i=1}^{N} \sum_{j=1}^{M} \| p_i^j - \hat{p}_i^j \|_{\Sigma_i}^2 - 2\lambda^T h(\hat{x}, \hat{p})
\]

- However:
  - Short-track features: encode valuable information for camera poses of **only the recent past images**
  - Observing feature points for many frames and loop closures: will typically involve optimizing more camera poses
iLBA - Concept

- Each time a new image is received:
  - Adaptively identify which camera poses should be updated
  - Only part of the previous camera poses are recalculated
  - Calculations from previous steps are re-used
  - Exact solution

- Incremental inference [Kaess et al., 2012]
  - Formulate the optimization problem using a factor graph [Kschischang et al., 2001]
  - Incremental optimization by converting to Bayes net and a directed junction tree (Bayes tree)
iLBA - Factor Graph Formulation

- MAP estimate is given by:
  \[
  \hat{\mathbf{x}} = \arg \max_{\mathbf{x}} p(\mathbf{x}|Z)
  \]

- Factorization of the joint probability function \( p(\mathbf{x}|Z) \)
  \[
  p(\mathbf{x}|Z) \propto \prod_i f_i(\mathbf{x}_i)
  \]
  - Each factor \( f_i \) represents a single term in the cost function
  - \( \mathbf{x}_i \) is a subset of variables related by the \( i \)th measurement\process model

- Example:
  \[
  p(\mathbf{x}) = p(x_0) \prod_j p(x_j|x_{j-1}) \prod_k p(z_k|x_{j_k})
  \]
iLBA - Factor Graph Formulation

- MAP estimate is given by:
  \[ \hat{\mathbf{X}} = \arg \max_{\mathbf{X}} p(\mathbf{X}|Z) \]

- Factorization of the joint probability function \( p(\mathbf{X}|Z) \)
  \[ p(\mathbf{X}|Z) \propto \prod_{i} f_{i}(\mathbf{X}_{i}) \]
  - Each factor \( f_{i} \) represents a single term in the cost function
  - \( \mathbf{X}_{i} \) is a subset of variables related by the \( i \)th measurement/process model

- In our case:
  - The variables are the camera poses: \( \mathbf{X} \equiv \mathbf{x} \)
  - The factors represent two- and three-view constraints

\[ f_{i}(\mathbf{X}_{i}) \equiv \exp \left( -\frac{1}{2} \| h_{i}(\mathbf{x}, \mathbf{p}) \|^{2}_{\Sigma_{i}} \right) \]
\[ h_{i} \in \{ g_{2v}, g_{3v} \} \]
Incremental Inference in iLBA

- Consider the non-linear optimization problem:

\[
J_{LBA}(\hat{x}) = \sum_{i=1}^{N_h} \| h_i(\hat{x}, p) \|^2_{\Sigma_i}
\]

\[
\hat{x} = \arg \max_{\mathcal{X}} (p(\mathcal{X}|Z)) = \arg \max_{\mathcal{X}} \prod_i f_i(\mathcal{X}_i)
\]

\[
f_i(\mathcal{X}_i) = \exp \left( -\frac{1}{2} \| h_i(x, p) \|^2_{\Sigma_i} \right)
\]

- Non-linear optimization involves repeated linearization

\[
\Delta^* = \arg \min_{\Delta} (A\Delta - b)
\]

- Solution involves factorization of \( A \) (e.g. QR)

- In our case - \( \Delta \) contains corrections to camera poses

- \( A \) - sparse Jacobian matrix
- \( b \) - right hand side vector
- \( \Delta \) - delta vector
Incremental Inference in iLBA

- Consider the non-linear optimization problem:

\[ J_{LBA}(\hat{x}) = \sum_{i=1}^{N_h} \| h_i(\hat{x}, p) \|_{\Sigma_i}^2 \]

\[ \hat{x} = \arg \max_{\chi} p(\chi | Z) = \arg \max_{\chi} \prod_i f_i(\chi_i) \]

\[ f_i(\chi_i) = \exp \left( -\frac{1}{2} \| h_i(x, p) \|_{\Sigma_i}^2 \right) \]

- Non-linear optimization involves repeated linearization

\[ \Delta^* = \arg \min_{\Delta} (A\Delta - b) \]

- Solution involves factorization of \( A \) (e.g. QR)

- When adding a new camera pose, calculations can be re-used
  - Factorization can be updated (and not re-calculated)
  - Only some of the variables should be re-linearized and solved for

- The above is realized by converting the factor graph into a Bayes net (and then to a directed junction tree)

\( A \) - sparse Jacobian matrix
\( b \) - right hand side vector
\( \Delta \) - delta vector
Incremental Inference in iLBA (Cont.)

- Example:

Factor graph

\[ x_1 \rightarrow x_2 \rightarrow x_3 \]

Linearization

Jacobian matrix

\[
A = \begin{bmatrix}
\times & \times \\
\times & \times \\
\times & \times \\
\times & \times \\
x_1 & x_2 & x_3
\end{bmatrix}
\]

Factorization

Factorized Jacobian matrix

\[
R = \begin{bmatrix}
\times & \times \\
\times & \times \\
\times & \\
x_1 & x_2 & x_3
\end{bmatrix}
\]
Incremental Inference in iLBA (Cont.)

- Example:

  ![Factor graph](image)

  ![Jacobian matrix](image)

  Linearization and elimination
  Elimination order $x_1, x_2, x_3$

  ![Bayes net](image)

  ![Factorized Jacobian matrix](image)

  Linearization and factorization of the Jacobian $A$ is equivalent to converting the factor graph into a Bayes net using a chosen elimination order [Pearl, 1998]
Incremental Inference in iLBA (Cont.)

- Adding new measurements and/or new camera poses involves updating only part of the Bayes net.

Example (Cont.):

New camera pose and two- and three-view factors

Bayes net does not change for $x_1$; calculations can be reused.
Incremental Inference in iLBA (Cont.)

- How to identify what should be re-calculated?
  - Bayes net is converted to Bayes tree (a directed junction tree) [Kaess et al., 2012]

- The “big” picture:

\[ J_{LBA}(\hat{x}) = \sum_{i=1}^{N_h} \| h_i(\hat{x}, p) \|_{\Sigma_i}^2 \]

\[ \Delta^* = \arg \min_{\Delta} (A\Delta - b) \]

  - Back-substitution (calculation of \( \Delta \)) is performed only for part of the variables (=camera poses)
  - Re-linearization is performed only when needed and only for part of the variables

- Overall - Allows an efficient sparse incremental non-linear optimization
Results

<table>
<thead>
<tr>
<th>Dataset</th>
<th># Images</th>
<th># 3D Points</th>
<th># Observations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cubicle</td>
<td>33</td>
<td>11,066</td>
<td>36,277</td>
</tr>
<tr>
<td>Straight</td>
<td>14</td>
<td>4,227</td>
<td>14,019</td>
</tr>
<tr>
<td>Circle (Synthetic)</td>
<td>120</td>
<td>500</td>
<td>58,564</td>
</tr>
</tbody>
</table>

- Image correspondences and camera calibration were obtained by first running bundler [http://phototour.cs.washington.edu/bundler/](http://phototour.cs.washington.edu/bundler/)
- Bundler’s data was **not** used elsewhere
### Results (Cont.)

<table>
<thead>
<tr>
<th>Notation</th>
<th>Method</th>
<th>Cost function</th>
</tr>
</thead>
<tbody>
<tr>
<td>$LBA$</td>
<td>Light bundle adjustment with the covariance $\Sigma_i$ calculated once</td>
<td>$J_{LBA}(\hat{x}) = \sum_{i=1}^{N_h} | h_i(\hat{x}, p) |_{\Sigma_i}^2$</td>
</tr>
<tr>
<td>$LBA\Sigma$</td>
<td>Light BA with the covariance $\Sigma_i$ re-calculated at each linearization</td>
<td></td>
</tr>
<tr>
<td>$SLB$</td>
<td>Structure-less bundle adjustment with image observations corrections</td>
<td>$J_{SLB}(\hat{x}, \hat{p}) = \sum_{i=1}^{N} \sum_{j=1}^{M} \left( | p_i^j - \hat{p}<em>i^j |</em>{\Sigma_i}^2 - 2\lambda^T h(\hat{x}, \hat{p}) \right)$</td>
</tr>
<tr>
<td>$BA$</td>
<td>Bundle adjustment</td>
<td>$J_{BA}(\hat{x}, \hat{L}) = \sum_{i=1}^{N} \sum_{j=1}^{M} \left( | p_i^j - \text{Proj} \left( \hat{x}_i, \hat{L}<em>j \right) |</em>{\Sigma_i}^2 \right)$</td>
</tr>
</tbody>
</table>

- Incremental smoothing vs incremental batch results will be shown for each method
### Results (Cont.)

<table>
<thead>
<tr>
<th>Notation</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>(LBA)</td>
<td>Light BA with the covariance (\Sigma_{i}) calculated once</td>
</tr>
<tr>
<td>(LBA\Sigma)</td>
<td>Light BA with the covariance (\Sigma_{i}) re-calculated upon each linearization</td>
</tr>
<tr>
<td>(SLB)</td>
<td>Structure-less BA with image observations corrections</td>
</tr>
<tr>
<td>(BA)</td>
<td>Bundle adjustment</td>
</tr>
</tbody>
</table>

![Graph showing average reprojection error vs. processing time](image)
## Results (Cont.)

<table>
<thead>
<tr>
<th>Notation</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LBA</strong></td>
<td>Light BA with the covariance $\Sigma_i$ calculated once</td>
</tr>
<tr>
<td><strong>LBA$\Sigma$</strong></td>
<td>Light BA with the covariance $\Sigma_i$ re-calculated upon each linearization</td>
</tr>
<tr>
<td><strong>SLB</strong></td>
<td>Structure-less BA with image observations corrections</td>
</tr>
<tr>
<td><strong>BA</strong></td>
<td>Bundle adjustment</td>
</tr>
</tbody>
</table>

- Additional results using **incremental smoothing** (for all methods):

<table>
<thead>
<tr>
<th>Dataset</th>
<th>BA</th>
<th>iLBA</th>
<th>iLBA$\Sigma$</th>
<th>SLB</th>
<th>$N$, $M$, #Obsrv</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cubicle</td>
<td>1.981 ($\mu$)</td>
<td>2.1017 ($\mu$)</td>
<td>2.0253 ($\mu$)</td>
<td>1.9193 ($\mu$)</td>
<td>33, 11066, 36277</td>
</tr>
<tr>
<td></td>
<td>1.6301 ($\sigma$)</td>
<td>1.8364 ($\sigma$)</td>
<td>1.742 ($\sigma$)</td>
<td>1.6294 ($\sigma$)</td>
<td></td>
</tr>
<tr>
<td>Straight</td>
<td>0.519 ($\mu$)</td>
<td>0.5434 ($\mu$)</td>
<td>0.5407 ($\mu$)</td>
<td>0.5232 ($\mu$)</td>
<td>14, 4227, 14019</td>
</tr>
<tr>
<td></td>
<td>0.4852 ($\sigma$)</td>
<td>0.5127 ($\sigma$)</td>
<td>0.5098 ($\sigma$)</td>
<td>0.4870 ($\sigma$)</td>
<td></td>
</tr>
<tr>
<td>Circle (synthetic)</td>
<td>0.6186 ($\mu$)</td>
<td>0.6244 ($\mu$)</td>
<td>0.6235 ($\mu$)</td>
<td>0.6209 ($\mu$)</td>
<td>120, 500, 58564</td>
</tr>
<tr>
<td></td>
<td>0.3220 ($\sigma$)</td>
<td>0.3253 ($\sigma$)</td>
<td>0.3246 ($\sigma$)</td>
<td>0.3235 ($\sigma$)</td>
<td></td>
</tr>
</tbody>
</table>

### Re-projection errors

### Computational cost [sec]
## Extended Cubicle dataset

<table>
<thead>
<tr>
<th># Images</th>
<th>148</th>
</tr>
</thead>
<tbody>
<tr>
<td># 3D Points</td>
<td>31,910</td>
</tr>
<tr>
<td># Observations</td>
<td>164,358</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>iLBA</th>
<th>iSLB</th>
<th>iBA</th>
</tr>
</thead>
<tbody>
<tr>
<td>Run time - Optimization</td>
<td>20 min</td>
<td>76 min</td>
<td>122 min</td>
</tr>
<tr>
<td>Run time - Structure rec.</td>
<td>2 min</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Outdoor dataset

<table>
<thead>
<tr>
<th></th>
<th>iLBA</th>
<th>iSLB</th>
<th>iBA</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Run time - Optimization</strong></td>
<td>1:56 hr</td>
<td>6:35 hr</td>
<td>5:40 hr</td>
</tr>
<tr>
<td><strong>Run time - Structure rec.</strong></td>
<td>2 min</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Summary

- We presented an incremental structure-less BA method: **iLBA**
  - **Reduced number of variables**: 3D points are algebraically eliminated
  - **Incremental inference**: only part of the camera poses are re-calculated each time a new image is added
  - Can handle degenerate configurations (co-linear camera centers)
  - Structure can be reconstructed, but only if required
Summary

- We presented an incremental structure-less BA method: **iLBA**
  - Structure-Less BA + incremental inference
    - Reduced number of variables - 3D points are not part of the iterative optimization
    - Only part of the camera poses are re-calculated each time a new image is added
  - Can handle degenerate configurations (co-linear camera centers)
  - Structure can be reconstructed, but only if required