Incremental Light Bundle Adjustment

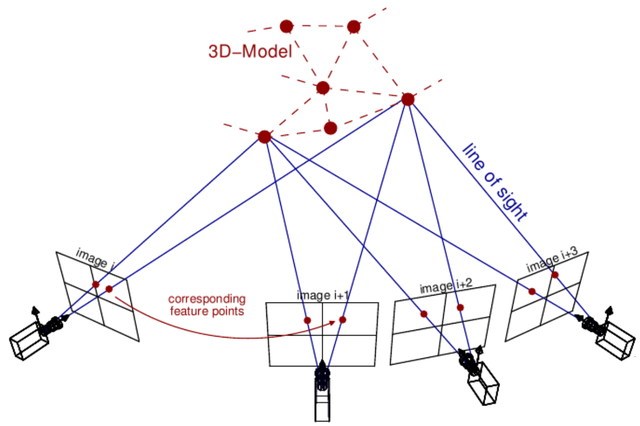
Vadim Indelman, Richard Roberts, Chris Beall, Frank Dellaert

College of Computing, Georgia Institute of Technology

September 2012

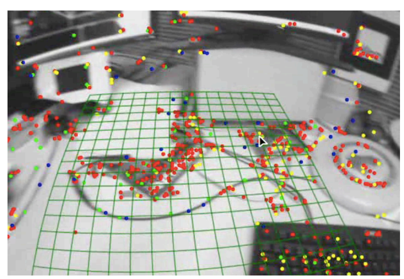
Introduction

- Bundle Adjustment: reconstruct camera poses and structure

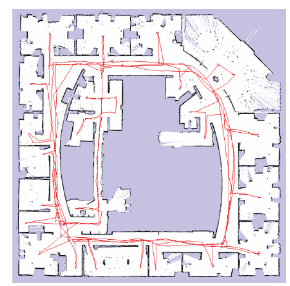


- Applied in a variety of applications:

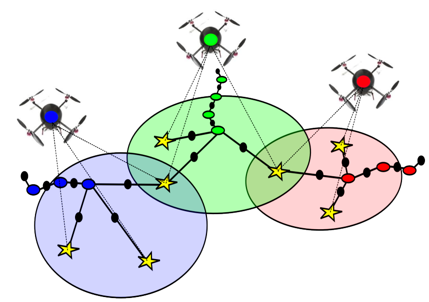
Structure from motion
[Snavely et al., 2006]



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]

$$J_{BA}\left(\hat{\mathbf{x}},\hat{\mathbf{L}}\right) \doteq \sum_{i=1}^{N}\sum_{j=1}^{M}\left\|\mathbf{p}_i^j - \mathbf{Proj}\left(\hat{\mathbf{x}}_i,\hat{\mathbf{L}}_j\right)\right\|_{\Sigma}^2$$

- 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

$$J_{SLB}(\hat{\mathbf{x}}, \hat{\mathbf{p}}) \doteq \sum_{i=1}^{N} \sum_{j=1}^{M} \left\| \mathbf{p}_i^j - \hat{\mathbf{p}}_i^j \right\|_{\Sigma}^2 - 2\boldsymbol{\lambda}^T \mathbf{h}(\hat{\mathbf{x}}, \hat{\mathbf{p}})$$

  - All multi-view constraints for a given sequence of view:

$$\mathbf{h} \doteq \begin{bmatrix} h_1 & \ldots & h_{N_h} \end{bmatrix}^T$$

$\mathbf{x}_i$ - i-th camera pose
$\mathbf{x}$ - all camera poses
$\mathbf{p}_i^j$ - observation of j-th 3D point in i-th image
$\mathbf{p}$ - all image observations

  - 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 $\mathbf{h}_i$:
  - MAP estimate is equivalent to a non-linear least-squares optimization
- Cost function: $$J_{LBA}(\hat{\mathbf{x}}) \doteq \sum_{i=1}^{N_h} \|h_i(\hat{\mathbf{x}}, \mathbf{p})\|^2_{\Sigma_i}$$
  - Σ_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 Σ_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]:

$$g_{2v}\left(x_k, x_l\right) = \mathbf{q}_k \cdot \left(\mathbf{t}_{k\to l} \times \mathbf{q}_l\right)$$

$$g_{2v}\left(x_l, x_m\right) = \mathbf{q}_l \cdot \left(\mathbf{t}_{l\to m} \times \mathbf{q}_m\right)$$

Epipolar constraints

$$g_{3v}\left(x_k, x_l, x_m\right) = \left(\mathbf{q}_l \times \mathbf{q}_k\right) \cdot \left(\mathbf{q}_m \times \mathbf{t}_{l\to m}\right) - \left(\mathbf{q}_k \times \mathbf{t}_{k\to l}\right) \cdot \left(\mathbf{q}_m \times \mathbf{q}_l\right)$$

Scale consistency

$$\boxed{\mathbf{q}_i \doteq R_i^T K_i^{-1} \mathbf{p}_i}$$

- 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:

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

$$h_i \in \{g_{2v}, g_{3v}\}$$

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

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

$$J_{SLB}(\hat{\mathbf{x}}, \hat{\mathbf{p}}) \doteq \sum_{i=1}^{N} \sum_{j=1}^{M} \left\| \mathbf{p}_i^j - \hat{\mathbf{p}}_i^j \right\|_{\Sigma}^2 - 2\boldsymbol{\lambda}^T \mathbf{h}(\hat{\mathbf{x}}, \hat{\mathbf{p}})$$

- 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: $\hat{\mathcal{X}} = \arg\max_{\mathcal{X}} p\left(\mathcal{X}|Z\right)$

- Factorization of the joint probability function $p\left(\mathcal{X}|Z\right)$

$$p\left(\mathcal{X}|Z\right) \propto \prod_i f_i\left(\mathcal{X}_i\right)$$

  - Each factor f_i represents a single term in the cost function
  - $\mathcal{X}_i$ is a subset of variables related by the *i*th measurement\process model

- Example:

$$p(\mathcal{X}) = p(x_0) \prod_j p(x_j|x_{j-1}) \prod_k p\left(z_k|x_{j_k}\right)$$

iLBA - Factor Graph Formulation

- MAP estimate is given by: $\hat{\mathcal{X}} = \arg\max_{\mathcal{X}} p(\mathcal{X}|Z)$
- Factorization of the joint probability function $p(\mathcal{X}|Z)$

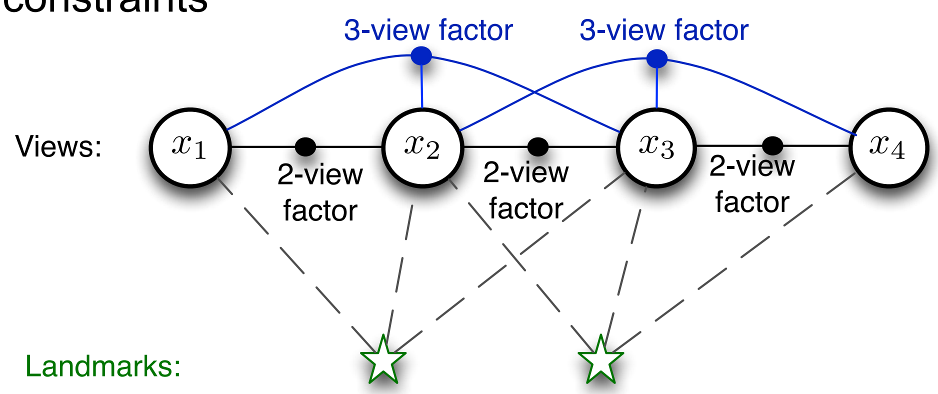
$$p(\mathcal{X}|Z) \propto \prod_i f_i(\mathcal{X}_i)$$

  – Each factor f_i represents a single term in the cost function
  – $\mathcal{X}_i$ is a subset of variables related by the *i*th measurement\process model

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

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

$$h_i \in \{g_{2v}, g_{3v}\}$$



Indelman et al., Incremental Light Bundle Adjustment

Georgia Institute of Technology
Robotics@GT & Intelligent Machines

Incremental Inference in iLBA

- Consider the non-linear optimization problem:

$$J_{LBA}(\hat{\mathbf{x}}) \doteq \sum_{i=1}^{N_h} \|h_i(\hat{\mathbf{x}}, \mathbf{p})\|^2_{\Sigma_i} \quad \Longleftrightarrow \quad \hat{\mathcal{X}} = \arg\max_{\mathcal{X}} \left(p(\mathcal{X}|Z)\right) = \arg\max_{\mathcal{X}} \prod_i f_i(\mathcal{X}_i)$$

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

- Non-linear optimization involves repeated linearization

$$\boldsymbol{\Delta}^* = \arg\min_{\boldsymbol{\Delta}} \left(A\boldsymbol{\Delta} - \mathbf{b}\right)$$

A - **sparse** Jacobian matrix
$\mathbf{b}$ - right hand side vector
$\boldsymbol{\Delta}$ - delta vector

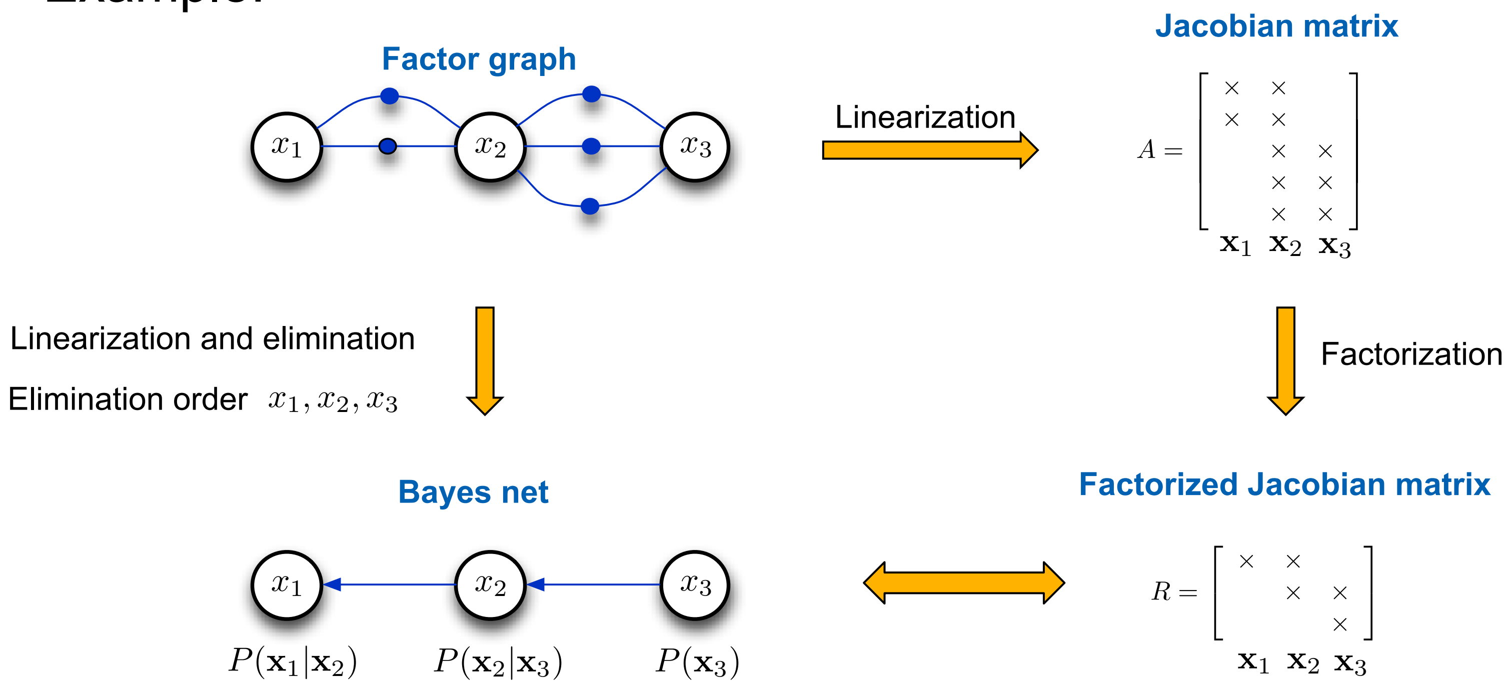
- Solution involves factorization of A (e.g. QR)

In our case - $\boldsymbol{\Delta}$ contains corrections to camera poses

- 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)

Incremental Inference in iLBA (Cont.)

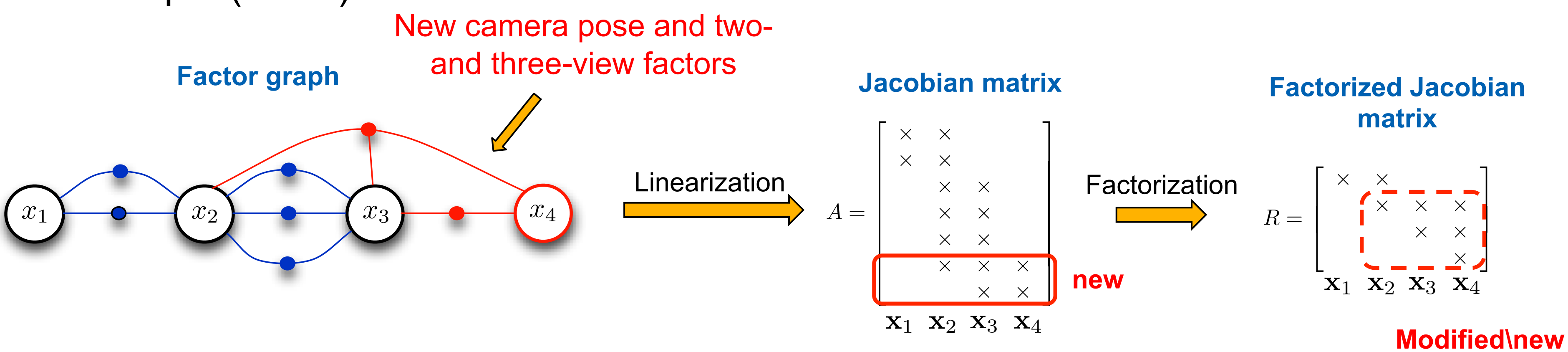
- Example:



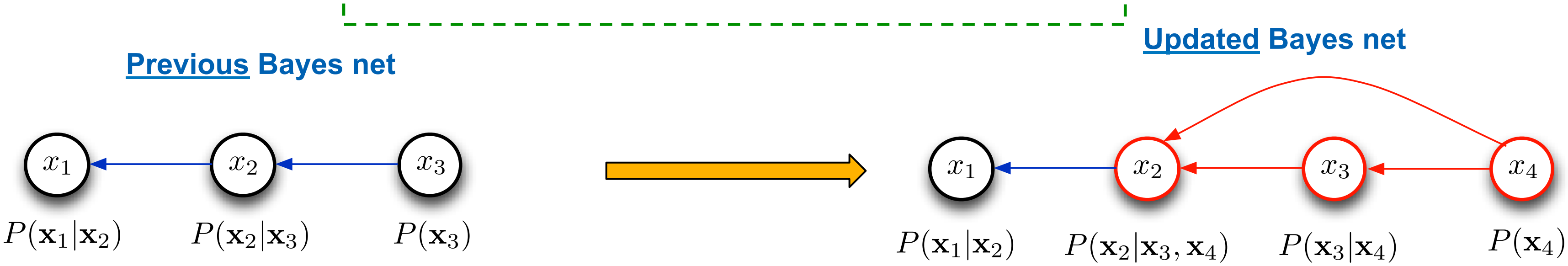
- Linearization and factorization of the Jacobian A is equivalent to converting the factor graph into a Bayes net using a chosen elimination order [Pearl, 1998]

Incremental Inference in iLBA (Cont.)

- Adding new measurements and\or new camera poses involves updating **only part** of the Bayes net
- Example (Cont.):



Bayes net does not change for $\mathbf{x}_1$; 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:

$$J_{LBA}(\hat{\mathbf{x}}) \doteq \sum_{i=1}^{N_h} \|h_i(\hat{\mathbf{x}}, \mathbf{p})\|^2_{\Sigma_i} \qquad\qquad \boldsymbol{\Delta}^* = \arg\min_{\boldsymbol{\Delta}} (A\boldsymbol{\Delta} - \mathbf{b})$$

  - Back-substitution (calculation of $\boldsymbol{\Delta}$) 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

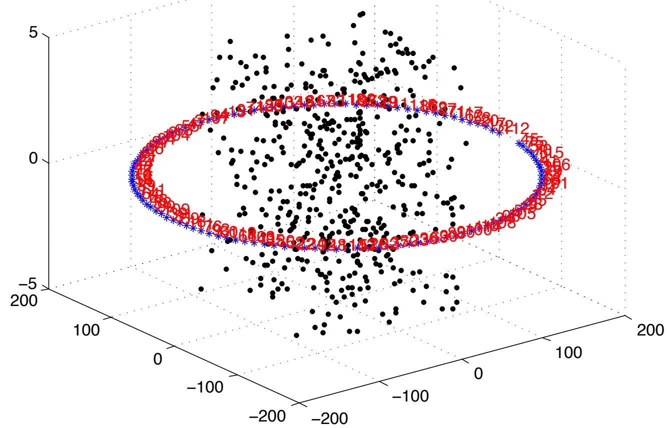
Dataset	# Images	# 3D Points	# Observations
Cubicle	33	11,066	36,277
Straight	14	4,227	14,019
Circle (Synthetic)	120	500	58,564

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

Cubicle

Straight

Circle (synthetic)



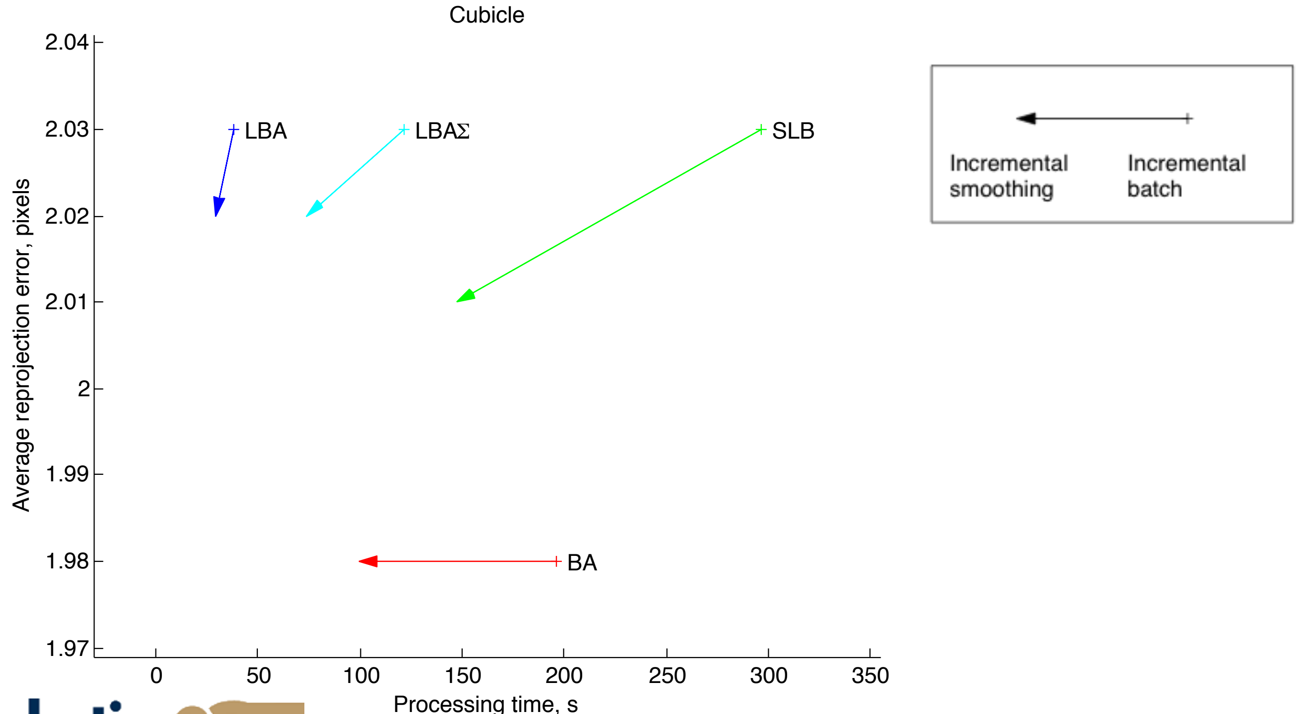
Results (Cont.)

Notation	Method	Cost function
LBA	Light bundle adjustment with the covariance Σ_i calculated once	$J_{LBA}(\hat{\mathbf{x}}) \doteq \sum_{i=1}^{N_h} \|h_i(\hat{\mathbf{x}}, \mathbf{p})\|^2_{\Sigma_i}$
$LBA\Sigma$	Light BA with the covariance Σ_i re-calculated at each linearization	
SLB	Structure-less bundle adjustment with image observations corrections	$J_{SLB}(\hat{\mathbf{x}}, \hat{\mathbf{p}}) \doteq \sum_{i=1}^{N} \sum_{j=1}^{M} \left\| \mathbf{p}_i^j - \hat{\mathbf{p}}_i^j \right\|^2_{\Sigma} - 2\boldsymbol{\lambda}^T \mathbf{h}(\hat{\mathbf{x}}, \hat{\mathbf{p}})$
BA	Bundle adjustment	$J_{BA}\left(\hat{\mathbf{x}}, \hat{\mathbf{L}}\right) \doteq \sum_{i=1}^{N} \sum_{j=1}^{M} \left\| \mathbf{p}_i^j - \mathbf{Proj}\left(\hat{\mathbf{x}}_i, \hat{\mathbf{L}}_j\right) \right\|^2_{\Sigma}$

- **Incremental smoothing** vs **incremental batch** results will be shown for each method

Results (Cont.)

Notation	Method
LBA	Light BA with the covariance Σ_i calculated once
$LBA\Sigma$	Light BA with the covariance Σ_i re-calculated upon each linearization
SLB	Structure-less BA with image observations corrections
BA	Bundle adjustment



Results (Cont.)

Notation	Method
LBA	Light BA with the covariance Σ_i calculated once
$LBA\Sigma$	Light BA with the covariance Σ_i re-calculated upon each linearization
SLB	Structure-less BA with image observations corrections
BA	Bundle adjustment

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

Re-projection errors

Dataset	BA	iLBA	iLBAΣ	SLB	N, M, #Obsrv
Cubicle	1.981 (μ) 1.6301 (σ)	2.1017 (μ) 1.8364 (σ)	2.0253 (μ) 1.742 (σ)	1.9193 (μ) 1.6294 (σ)	33, 11066, 36277
Straight	0.519 (μ) 0.4852 (σ)	0.5434 (μ) 0.5127 (σ)	0.5407 (μ) 0.5098 (σ)	0.5232 (μ) 0.4870 (σ)	14, 4227, 14019
Circle (synthetic)	0.6186 (μ) 0.3220 (σ)	0.6244 (μ) 0.3253 (σ)	0.6235 (μ) 0.3246 (σ)	0.6209 (μ) 0.3235 (σ)	120, 500, 58564

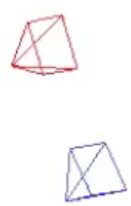
Computational cost [sec]

Dataset	BA	Structure-less BA			
		iLBA	iLBAΣ	SLB	structure recon.
Cubicle	99.5	29.1	73.7	147.3	18.4
Straight	12.1	3.0	10.0	10.5	6.9
Circle (synthetic)	>2hr	131.8	301.6	>2hr	3.8

Extended Cubicle dataset

# Images	148
# 3D Points	31,910
# Observations	164,358

	iLBA	iSLB	iBA
Run time - Optimization	20 min	76 min	122 min
Run time - Structure rec.	2 min		



Outdoor dataset

# Images	**308**
# 3D Points	**74,070**
# Observations	**316,696**

	iLBA	iSLB	iBA
Run time - Optimization	1:56 hr	6:35 hr	5:40 hr
Run time - Structure rec.	2 min		

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