Project acronym: LADIO Project number: 731970 Work package: Deliverable number and name: D3.3: Improve SfM accuracy and precision	Title: Improve SfM accuracy and precision Work Package: WP3 Version: 1 Date: August 31, 2017
	Author: Pierre Gurdjos
Type: [ ] Report [ ] Demonstrator, pilot, prototype [ ] Website, patent filings, videos, etc. [X] Other	Co-Author(s): Fabien Castan, Tomas Pajdla, Michal Polic To: Albert Gauthier, Project Officer
Status: [ ] Draft [ ] To be reviewed [ ] Proposal [X] Final / Released to EC	Confidentiality: [X] PU – Public [ ] CO – Confidential [ ] CL - Classified
Revision: Final	
Contents: Deliverable 3.3. Improve SfM accuracy and precision	

Table of contents

Introduction

The goal of this deliverable is to improve the robustness and accuracy of the SfM (Structure from Motion) pipeline while at the same time improving the performances for large-scale SfM datasets. The openMVG library implements two approaches as SfM pipelines, referred to as the sequential and the global methods. We use the incremental method as it provides better robustness, but this method has an important limitation. It adds images by small groups and needs to basically perform a Bundle Adjustment (BA) at each step. The BA alone is by itself quadratic, so in practice we are limited in the number of images we can support as input.
The ambition of this deliverable was also to provide quality feedback by uncertainty estimation and based on the results in term of quality and performances, we can decide to compute and use this information inside the pipeline to improve the robustness.

Combine Multiple Descriptors

We added the support to mix different types of features/descriptors (SIFT, AKAZE, CCTAG, …) in the SfM pipeline. The idea behind is that we can play with feature extraction parameters to retrieve more features but these features become more and more unstable. The number of outliers increase faster than the number of inliers, decreasing the inliers/outliers ratio which is critical for Ransac.

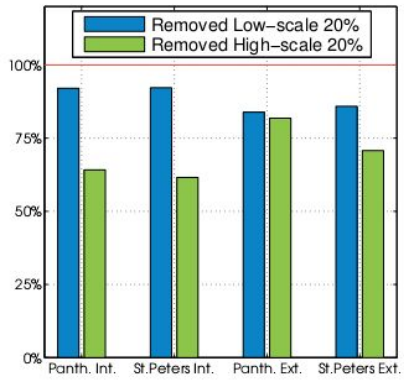


Figure 1 : from [Rajvi:2014]: Effect of removing high/low scale SIFT features

Features extraction methods have different criteria and will attach on different parts of the image. For instance, we have observed that AKAZE features provide more valid matches on skin textures than SIFT. By mixing multiple types of features/descriptors, we can combine the robustness of different feature extraction methods. This idea has been proposed in several papers, as [Ferber:2016] and [Abeles:2013].

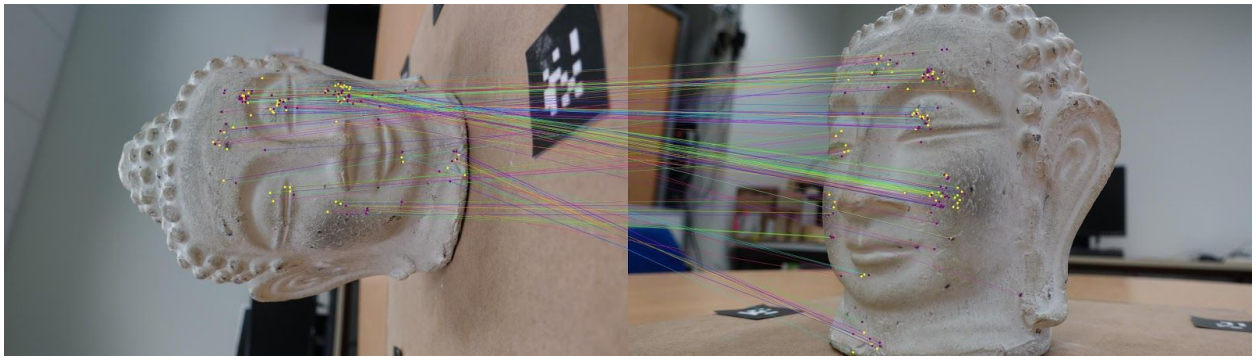


Figure 2 : SIFT features are represented in yellow, AKAZE features in purple and matches use random color lines.

In term of implementation, we perform the photometric feature matching per descriptor type, fuse all these matching candidates and perform the geometric validation globally.

The support of multiple features/descriptors is also important for the genericity of the library. We have done an important refactoring to manipulate features/descriptors more easily and add the notion of feature type in the whole pipeline. Based on this refactoring, we have integrated a GPU implementation of SIFT (PopSift released in POPART EU project) into the SfM pipeline (previously it was only used in the camera localization part).
Having the notion of descriptor types is useful for many use cases. Now, we can perform the reconstruction with natural features and markers but perform localization only with markers to work in real-time for instance.
We have updated the previz software (appCutie) and the Nuke camera tracking plugin (NukeMVG) to add these notions.

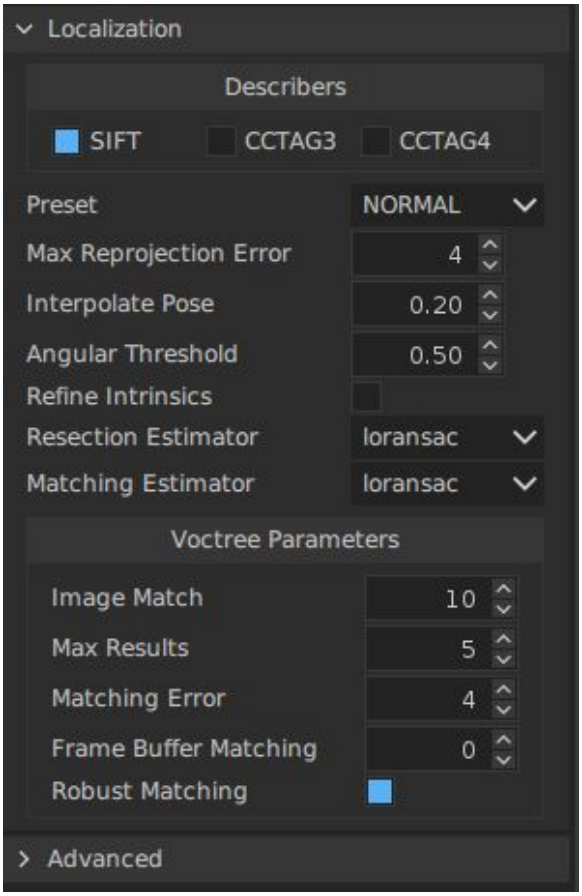


Figure 3 : Screenshot of the newly exposed parameters in our previz tool appCutie

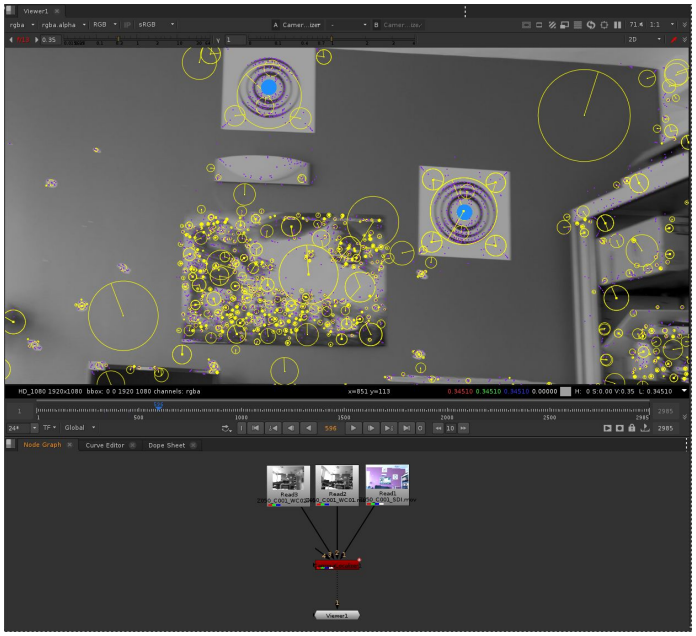


Figure 4 : Screenshot of SIFT natural features (yellow) and CCTag markers (blue) detected in our NukeMVG plugin

This work has been merged and is publically available: https://github.com/alicevision/openMVG

References

[Abeles:2013] Peter Abeles, “Examination of Hybrid Image Feature Trackers”, ISVC (2) 2013: 552-561

[Rajvi:2014] Rajvi Shah, Aditya Deshpande, PJ Narayanan, “Multistage SFM: Revisiting Incremental Structure from Motion”, IEEE WACV 2015, pp. 278-285, 2015.

[Ferber:2016] Marvin Ferber, Mark Sastuba, Steve Grehl, Bernhard Jung, “Combining SURF and SIFT for Challenging Indoor Localization using a Feature Cloud”, IEEE Int’l Conf. on Indoor Positioning and Indoor Navigation, 2016.

Bundle Adjustment Performances

As LADIO uses an incremental SfM pipeline, particular attention must be given how to efficiently apply Bundle Adjustment (BA) algorithms. Existing approaches improving incremental SfM can be divided into at least two groups:

1. Approaches focusing on mathematical improvements to make algorithms as efficient as possible, especially when dealing with large-scale datasets ;
2. Approaches focusing on reducing the numbers of unknowns i.e., the cameras and the landmark points to be estimated in BA using a technique that we call “Local Bundle Adjustment” (LBA).

Parameter ordering (a type-1 approach)

Solving the normal linear equation $H\Delta x = g$ for the parameter-vector increment Δx at each iteration of the Levenberg-Marquardt algorithm used in BA basically requires the inversion (by factorization) of the “regularized Hessian matrix” H, which is symmetric and has dimension $(p+q)\times(p+q)$, where p and q are affine functions of the number m of cameras and the number n of points respectively. Typically, we have $p = 11m$ and $q = 3n$, and when p and q become large, solving the normal equation becomes a critical issue, possibly unfeasible.

In order to deal with large-scale BA data, the so-called “Schur Complement” trick transforms the linear solving of H to another linear solving of a $r \times r$ matrix S for a subset Δy of $r \ll p+q$ parameters followed by a back-substitution of the obtained solution to get the optimal values of the remaining $p+q-r$ parameters. Technically speaking, it consists in replacing the original normal linear (matrix) equation, using Gaussian elimination, by a system of two new linear (matrix) equations, one of which, called “reduced equation”, writes $S\Delta y = u$ so only depending

on Δy, where S is the so-called "Schur Complement matrix" of the $(p+q-r)\times(p+q-r)$ principal submatrix of H obtained by removing the r rows and the same r columns corresponding to Δy.

One must carefully select these r parameters in such a way that the new system is cheaper to solve i.e., its resolution has a true lower complexity. The selection of these r parameters is referred to as *parameter ordering* in Ceres solver.

The established *modus operandi* for parameter ordering is that of leveraging that $p \ll q$ i.e., the number of camera parameters is much smaller than the number of point parameters. Therefore, an efficient ordering can be obtained by partitioning the parameter vector Δx appearing in the original normal equation into two concatenated groups $\Delta x = [\Delta y^T, \Delta z^T]^T$, with $u = [v^T, w^T]^T$ accordingly, by setting as upper subvector Δy all the p camera parameters (so $r = p$) and, as lower subvector Δz, all the q point parameters. This leads to a regularized Hessian matrix of the form

$$H = \begin{bmatrix} B & E \\ E^\top & C \end{bmatrix}$$

where B is a squared block sparse matrix of dimension $p \times p$ and C is a squared block diagonal matrix of dimension $q \times q$ e.g., with n "small" blocks of size 3 on the diagonal when $q = 3n$. It is worthy of mention that

- if the m cameras are calibrated, so $p = 6m$, B is also a block diagonal matrix of dimension $q \times q$ e.g., with m "small" blocks of size 6 on the diagonal, otherwise it is a sparse block matrix,
- if all points are not visible in all views, the block matrix E can be very sparse.
-

Using the Gaussian elimination of Δz, one obtain a new system to solve which consists of two new equations $\left(B - EC^{-1}E\right)\Delta y = v - EC^{-1}w$ and $\Delta z = C^{-1}(w - E^T\Delta y)$, both linear, the former being the reduced one. Constructing the new equations, and especially the Schur Complement matrix $S = B - EC^{-1}E$, does not show up any difficulty as, thanks to the parameter ordering, calculating the inverse of the block diagonal matrix C can be carried out by inverting each of these blocks and so is an extremely cheap.
In CERES Solver, the resolution of the reduced equation $S\Delta y = v - EC^{-1}w$ for Δy is carried out via the Cholesky factorization of S. As
S is typically a fairly sparse matrix, one can also use row and column reordering algorithms to maximize the sparsity of the Cholesky decomposition, and focus their compute effort on the non-zero part of the factorization. At the end, back-substituting Δy to obtain the value of Δz is straightforward.

In Ceres Solver, we re-ordered the whole set of parameters such that the first parameter group to be eliminated are the q parameters of the points (Δz) so the r parameters to be estimated

first coincide with the p parameters of the cameras (Δy). As explained before, the order in which variables are eliminated in a linear solver can have a significant of impact on the efficiency and accuracy of the method. Nevertheless, we point out that Ceres Solver is able to determine itself an automatic way to order the parameters (default option).

We applied two "parameter ordering" methods into the code and tested it on 4 datasets: the "automatic" CERES Solver method vs. the proposed one as described above.

Dataset name	m = # points	n = # images	Time (s) "automatic"	Time (s) "ordering"	Time saving
Lastpant	*375 000*	*1050*	*7000*	*6350*	***9.3%***
Lou	*285 000*	*510*	*1600*	*1480*	***7.5%***
Cirque	*120 000*	*500*	*2800*	*2700*	***3.6%***
Levallois	*120 000*	*540*	*2550*	*2500*	***2%***

Table 1 : results of Parameter Ordering on 4 datasets

Local Bundle Adjustment (type-2 approach)

Let C be set of cameras/views where c_i denote the camera/view number i ($i = 1..m$) and $\beta(c_i)$ the associated parameters. Similarly, let X be set of 3D landmarks where x_j denote the landmark number j ($j = 1..m$) and $\beta(x_j)$ the associated parameters.

We construct the camera graph $G = (V, E)$ where vertices (whose set is denoted by V) are camera numbers and edges (whose set is denoted by E) link camera pairs only if the two cameras have at least s matched measurements of 3D landmarks (e.g., $s = 100$).

Let $V' \subseteq V$ be a subset of views/cameras newly resected and to be added to the SfM pipeline. In the above toy example, we have $V = \{1, 2, 3, 4, 5, 6\}$ and $V' = (8, 9\}$

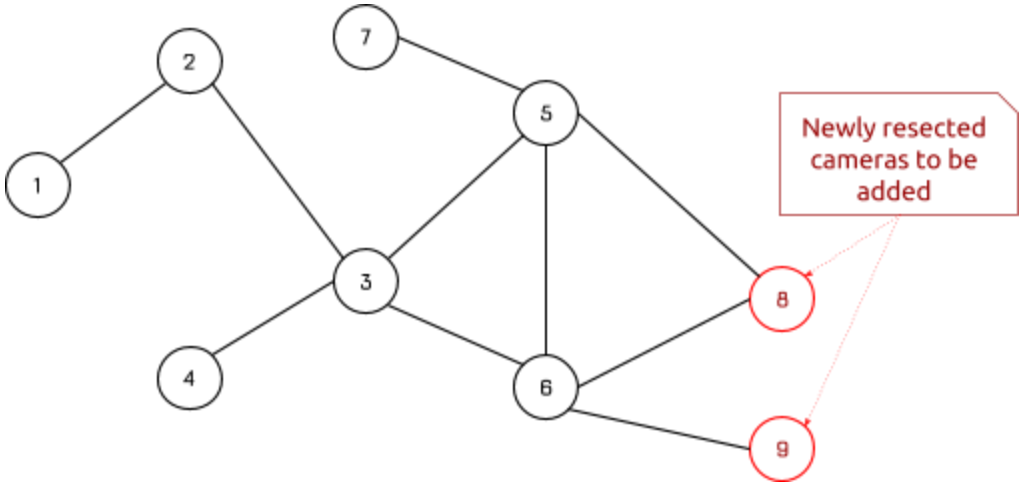


Figure 5 : Graph representation of the reconstruction of the first nine cameras.

In order to define what is Local Bundle Adjustment (LBA), the following definitions of “active regions” and “passive regions” in the camera graph must be given.

The ***active region*** of V' w.r.t. a property P is the union of V' and the subset $V_A \supseteq V \setminus V'$ such that, for each $u \in V_A$,

1) P holds for u and
2) there exists in G at least one path from u to a vertex $v \in V'$ (i.e., u is connected to v).

The ***passive region*** of V' w.r.t. P is the subset $V_P \subseteq V \setminus (V_A \cup V')$ such that, for each $u \in V_P$ there exists at least one edge from u to a vertex $v \in V_A$ (i.e., u is adjacent to v).

As a consequence,

- $V'_A \cap V_P = \varnothing$,
- P holds for all $u \in V_A$,
- P doesn't hold for any $u \in V_P$.

Now, we can define ***Local Bundle Adjustment (LBA)*** as the problem of jointly estimating the camera parameters $\{\beta(v_i)\}_{i \in V' \cup V_A}$ *only* associated to active views and the parameters $\{\beta(x_j)\}_{j \in J}$ of landmarks *only* seen in active views by minimizing the (robust) sum of squared reprojection errors in *both* the active and passive views.

In LADIO, we defined the property P for a vertex/camera u as “*given an integer $D > 1$, there exists a vertex/camera v in the set V' (of newly added/resected cameras) to which the distance from u to is less or equal to D* ”, where the distance refers to the number of edges in a shortest path from u to v. It is basically the solution suggested in the SLAM pipeline of [Mei:2011]. It is also very similar to [Mouragnon:2006] where views are acquired one at a time from camera

poses $c_{t-D}, c_{t-D+1}, \cdots, c_t$ and a camera is at distance D from camera C_t if it is one of the last D-th cameras.

The graph management is done with the *Lemon* graph library already included in OpenMVG and we compute the distance using the a Breadth First Search (BFS) function.

The basic LBA pseudo-code is described below. Remind that while the BA algorithm minimizes all the parameters, the Local BA algorithm minimizes the closest cameras only.

Algorithm :

```
Dmax : a distance limit.

for each resected camera
       if camera's distance <= Dmax
              its extrinsic parameters are refined
       if camera's distance == Dmax + 1
              its extrinsic parameters are set to constant
       else
              its extrinsic parameters are not set into the solver

for each landmark
       if it is observed by a refined camera (camera's distance <= Dmax)
              its coordinates are refined
       else
              its coordinates are not set into the solver

all the intrinsics parameter are refined
```

Estimated parameters determine the “Active Region”, constant parameters determine the “Passive Region” and remaining parameters determine the “Skipped Region”.

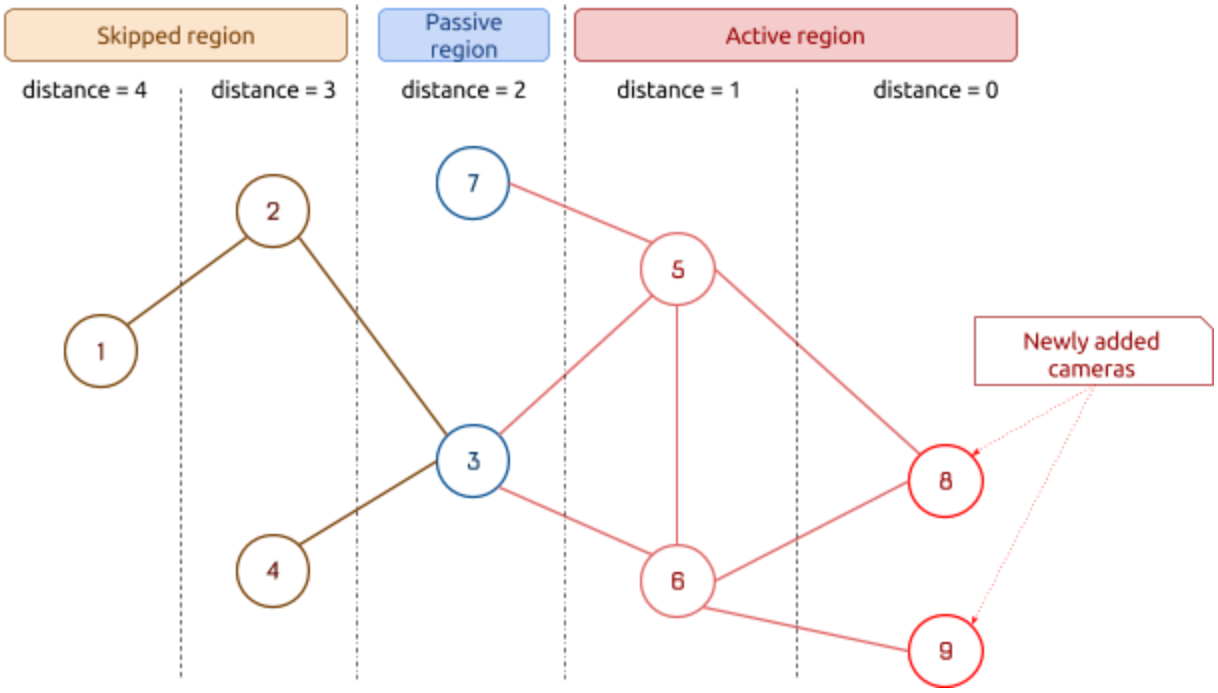
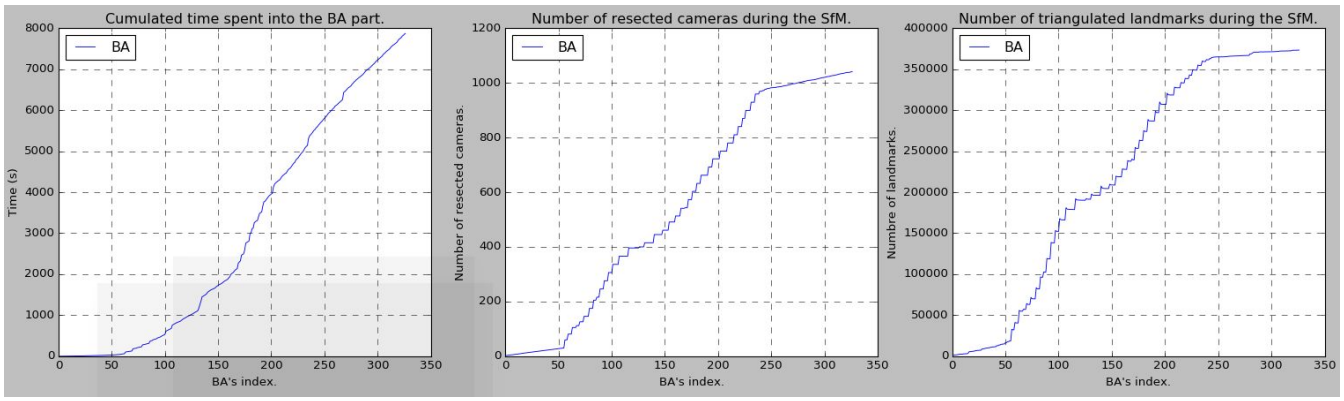


Figure 6 : Regions distribution for $D = 1$.

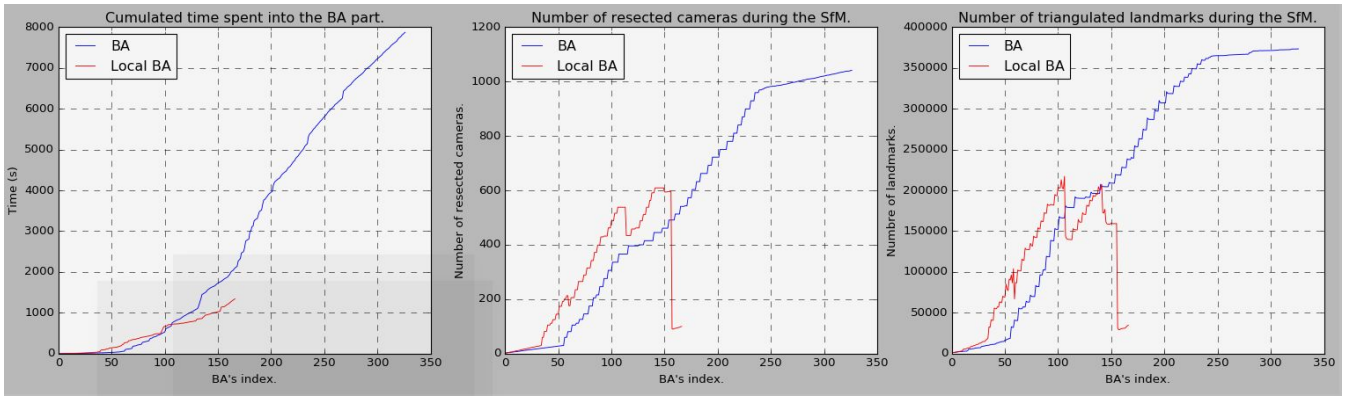
We applied the LBA algorithm to the *Lastpant* dataset (1100 images)
Here are the results obtained with the classical BA method :

- ~1100 resected images
- ~370.000 estimated 3D points
- ~8000s spent in the BA part (~2h15)



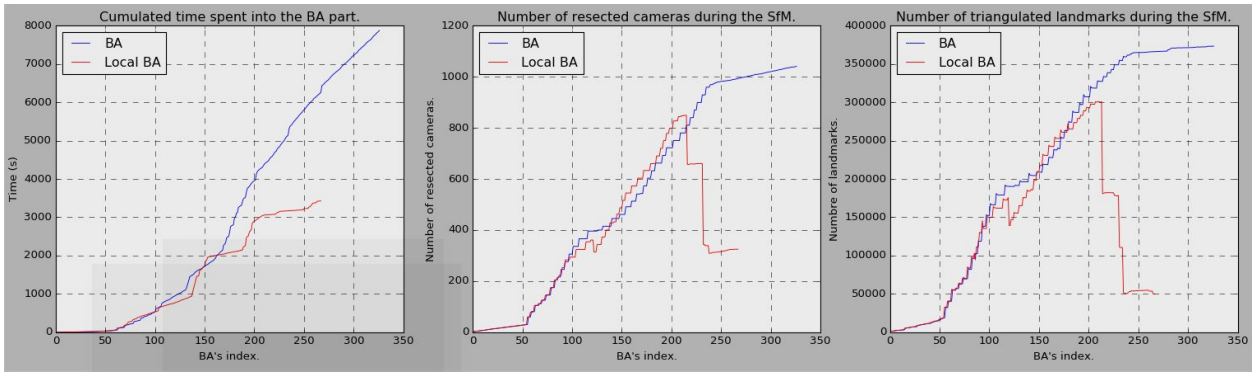
(a) *(b)* *(c)*

Figure 7 : Statistics about 3D reconstruction using the classical BA.



(a) *(b)* *(c)*

Figure 8 : Statistics about 3D reconstruction using the Local BA. with $D = 1$ (red lines) overlapping the results with the classical BA (blue lines).



(a) *(b)* *(c)*

Figure 9 : Statistics about 3D reconstruction using the Local BA. with $D = 3$ (red lines) overlapping the results with the classical BA (blue lines).

The Local BA technique improves a lot the time consumed into the BA part :

- for $D = 1$: time reduced by a factor 7 (~20min)
- for $D = 3$: time reduced by a factor 2 (~55min)

However, badly estimated cameras have a important impact on the reconstruction reducing the number of cameras and points in the reconstruction. Once we found a way to identify those bad cameras, the compromise between time consumed and the 3D reconstruction quality will be very interesting.

This work is not yet merged but is publically available:
https://github.com/alicevision/openMVG/tree/dev_LocalBA

References

[Agarwal:2010] "***Bundle Adjustment in the Large***", S. Agarwal *et al*., ECCV, Sep 2010, Heraklion, Crete.

[Ceres:1] "Solving Non-linear Least Squares / ***LinearSolver***", Ceres Solver documentation (ceres-solver.org/nnls_solving.html#linearsolver)

[Ceres:2] "Solving Non-linear Least Squares / LinearSolver / ***Ordering***", Ceres Solver documentation (ceres-solver.org/nnls_solving.html#ordering)

[Jeong:2012] "***Pushing the Envelope of Modern Methods for Bundle Adjustment***", Y. Jeong *et al*., PAMI Vol. 32, n° 8, August 2012.

[Mei:2011] "***RSLAM: A System for Large-Scale Mapping in Constant-Time Using Stereo***", C. Mei *et al*., IJCV Vol. 94, n° 2, September 2011.

[Mouragnon:2006] "***Real Time Localization and 3D Reconstruction***", E. Mouragnon *et al*., CVPR, N.Y.C., USA, June 2006.

[Triggs:2000] "***Bundle Adjustment — A Modern Synthesis***", B. Triggs *et al*., International Workshop on Vision Algorithms, Sep 2000, Corfu, Greece.

Uncertainty Estimation

In computer vision, large scale Structure from Motion pipelines do not often evaluate the quality of the reconstruction by error propagation from measurements to the estimated parameters. It is a numerically sensitive and computationally challenging process, which is not easy to implement in practice for large scenes. We have developed a new algorithm that increases the numerical precision of the uncertainty propagation. It works with millions of feature points, thousands of cameras and millions of 3D points on a single computer.

The task is to compute the backward transport (from measurements to the parameters) of the uncertainty for nonlinear over-parameterized systems (represented by objective functions) [Hartley 2001]. The reconstruction by SfM can be moved, rotated and scaled without change of objective function which cause the rank-deficiency of the Fisher information matrix [Rao 1973] . To invert it, we had to fix the ambiguity (gauge freedom). We project the space of the parameters to a manifold such that there is a one-to-one mapping from observations to parameters, i.e. there is no ambiguity in parameters. There are infinitely many such manifolds. Thus we investigated which one minimizes the differences in comparison with the normal form of the covariance matrices [Kanatani 2001]. We empirically found out that most similar uncertainties produce the fixation of the three most distant points which we seek by RANSAC. The inversion allowed us to scale the Fisher information matrix and its decomposition to smaller blocks, which increased the precision and speed up the propagation.

We published the Taylor expansion (TE) idea in SCIA [Polic SCIA2017] and extended it by replacing the computation of the M-P pseudoinverse and estimating the inverse of the Shur

complement of the matrix of the point parameters in the paper submitted to 3DV [Polic 3DV2017].

We made a C++ implementation with our publication. The first version corresponding to [Polic SCIA2017] is available on our private github: https://github.com/alicevision/cuc-double
Then we made a new C++ implementation of the new approach, also available on our private github: https://github.com/alicevision/uncertainty. This new implementation provides an integration of openMVG, so we can directly compute cameras uncertainties on the openMVG SfM results.

Precision

To compare all algorithms, we construct the normal form of the covariance matrix (considered as the Ground Truth (GT)) in Maple using 100 significant digits. The standardly used double has 15 significant digits. The previous works end up with too large errors and completely fail (e.g. algorithm 6 for Daliborka dataset) with the growing size of the reconstruction. Our algorithm has the opposite trend and therefore is suitable for computation of the uncertainty matrices for large scale reconstructions.

Visualised subset of datasets

Id	Name	Number of cameras	Number of points	Number of observations
1	Cube	6	15	60
2	Toy	10	60	200
3	Flat	30	100	1033
4	Daliborka	64	200	5205

Compared algorithms

Id	Algorithm
1	SVD of Fisher information matrix using Maple (GT) [Kanatani 2001]
2	TE inversion of scaled Shur complement matrix with three points fix [Polic 3DV2017]
3	SVD of Fisher information matrix using Ceres [Kanatani 2001]
4	TE inversion of scaled Shur complement matrix with trivial camera fix
5	SVD of Shur complement matrix with correction term [Lhuillier 2006]

6	SVD of Fisher information matrix using Matlab [Kanatani 2001]
7	M-P inverse of Shur complement matrix using TE [Polic SCIA2017]

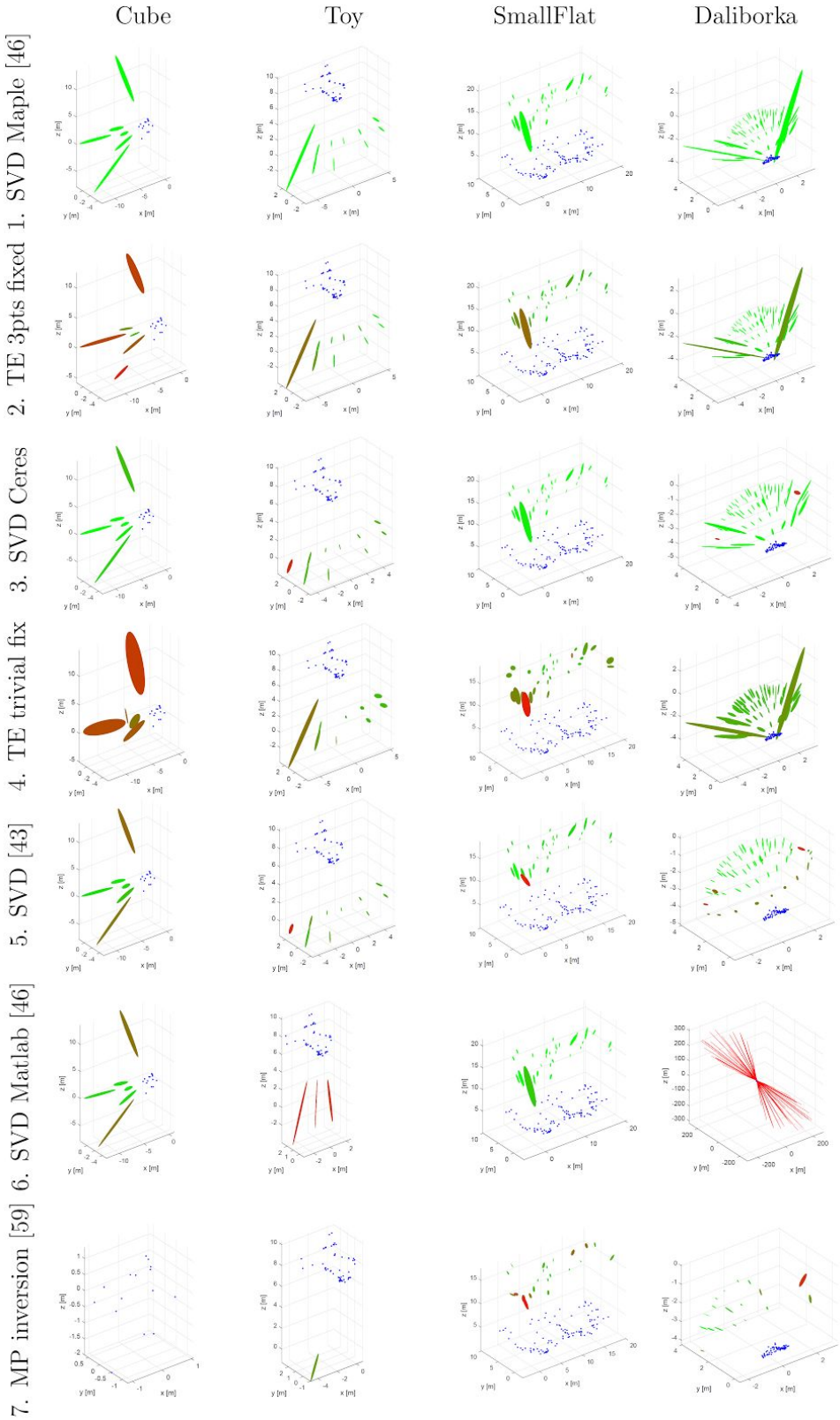


Figure 10 : The precision of compared algorithms

Speed

The covariance matrix using M-P pseudo-inversion of Fisher information matrix for Daliborka dataset (i.e. for 1176 reconstruction parameters) was computed using Ceres solver in 25.9 min and our TE inversion was calculated in 0.35 sec. Further, the first middle sized reconstruction (i.e. 243681 reconstruction parameters) requires in Ceres about 470 GB and the evaluation would take approximately 9 million times the time of Daliborka evaluation in Ceres. The TE inversion was performed in 4.32 sec.

References

[Hartley 2001] Hartley, Richard, and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[Rao 1973] Rao, Calyampudi Radhakrishna, et al. *Linear statistical inference and its applications*. Vol. 2. New York: Wiley, 1973.

[Kanatani 2001] Kanatani, Ken-ichi, and Daniel D. Morris. "Gauges and gauge transformations for uncertainty description of geometric structure with indeterminacy." *IEEE Transactions on Information Theory* 47.5 (2001): 2017-2028.

[Lhuillier 2006] Lhuillier, Maxime, and Mathieu Perriollat. "Uncertainty ellipsoids calculations for complex 3D reconstructions." *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*. IEEE, 2006.

[Polic 3DV2017] Polic, Michal, and Tomas Pajdla. "Camera Uncertainty Computation in Large 3D Reconstruction.."

[Polic SCIA2017] Polic, Michal, and Tomas Pajdla. "Uncertainty Computation in Large 3D Reconstruction." *Scandinavian Conference on Image Analysis*. Springer, Cham, 2017.