Bundle adjustment gone public

Manolis I.A. Lourakis

lourakis@ics.forth.gr

`http://www.ics.forth.gr/~lourakis/sba`

Institute of Computer Science
Foundation for Research and Technology - Hellas
Vassilika Vouton, P.O. Box 1385, GR 711 10
Heraklion, Crete, GREECE

SaM

- Structure and Motion Estimation (SaM) is the problem of using 2D measurements arising from a set of images of the same scene in order to recover information related to the 3D geometry of the imaged scene as well as the locations and optical characteristics of the employed camera(s)

- It is an archetypal problem with a wide spectrum of applications:
  - organizing community photo collections
  - visual odometry
  - augmented reality & virtual telepresence
  - video post production
  - video metrology
  - image-based 3D graphics
  - 3D motion capture
  - object grasping & manipulation
  - ...

Bundle Adjustment

- Bundle Adjustment (BA) is a key ingredient of SaM, almost always used as its last step
- It is an optimization problem over the 3D structure and viewing parameters (camera pose, intrinsic calibration, & radial distortion parameters), which are simultaneously refined for minimizing reprojection error
- BA is the ML estimator assuming zero-mean Gaussian image noise
- BA boils down to a very large nonlinear least squares problem, typically solved with the Levenberg-Marquardt (LM) algorithm
- Std LM involves the repetitive solution of linear systems, each with $O(N^3)$ time and $O(N^2)$ storage complexity, resp.
- Example: for 54 cameras and 5207 3D points, $N = 15945$
- This is prohibitively large for practical problems!

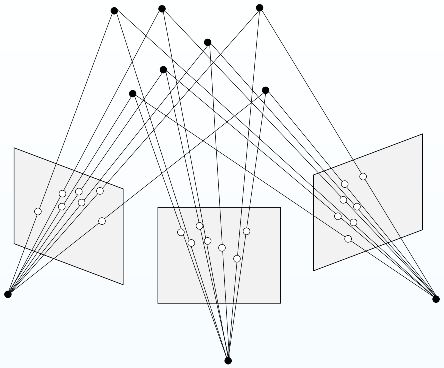
What this talk is about

- Fortunately, there is a way out
- The linear systems that LM needs to solve for BA have a sparse block structure
- This is because the projection of a point on a certain camera does not depend on the parameters of any other point or camera
- Sparse BA is one of the driving forces behind the success of recent SaM systems
- This talk concerns
  - a scheme for dealing with BA that exploits sparseness to yield significant computational savings
  - an ANSI C software library (called `sba`) that implements this scheme and which has been made publicly available under the GNU GPL

Levenberg-Marquardt algorithm overview

- Let $f : \mathcal{R}^m \rightarrow \mathcal{R}^n$. Given an initial estimate $\mathbf{p}_0 \in \mathcal{R}^m$ and a measurement vector $\mathbf{x} \in \mathcal{R}^n$, LM seeks to find $\mathbf{p}^+$ minimizing $\epsilon^T\epsilon$, $\epsilon = \mathbf{x} - f(\mathbf{p})$
- Note that this is a (nonlinear) least squares problem since $\epsilon^T\epsilon = ||\mathbf{x} - f(\mathbf{p})||^2$, $||.||$ being the L2 norm
- The minimizer can be found by the Gauss-Newton method, which iteratively linearizes f at $\mathbf{p}$ and determines incremental update steps $\delta_{\mathbf{p}}$ by solving the *normal equations* $\mathbf{J}^T\mathbf{J}\delta_{\mathbf{p}} = \mathbf{J}^T\epsilon$, $\mathbf{J}$ being the Jacobian of f at $\mathbf{p}$ and $\mathbf{J}^T\mathbf{J}$ the approximate Hessian of $||\epsilon||^2$
- To ensure convergence, LM uses *damping*, i.e. adaptively alters the diagonal elements of $\mathbf{J}^T\mathbf{J}$ and solves the *augmented normal equations* $(\mathbf{J}^T\mathbf{J} + \mu\mathbf{I})\delta_{\mathbf{p}} = \mathbf{J}^T\epsilon,\ \mu > 0$

Bundle Adjustment



- Assume n 3D points are seen in m views. Illustration with $n = 7, m = 3$
- Let $\mathbf{x}_{ij}$ be the projection of the i-th point on image j, $\mathbf{a}_j$ the vector of parameters for camera j and $\mathbf{b}_i$ the vector of parameters for point i
- BA minimizes the *reprojection error* over all point and camera parameters: $\min_{\mathbf{a}_j, \mathbf{b}_i} \displaystyle\sum_{i=1}^{n} \sum_{j=1}^{m} v_{ij} d(\mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i),\ \mathbf{x}_{ij})^2$, $\mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i)$ being the predicted projection of point i on image j, $d(.,.)$ the Euclidean distance between image points and $v_{ij} = 1$ iff point i is visible in image j
- This is a large problem: if κ, λ are the dimensions of the $\mathbf{a}_j$ & $\mathbf{b}_i$, the total number of parameters involved in BA is $m\kappa + n\lambda$

BA as a nonlinear least squares problem

- A parameter vector $\mathbf{P}$ is defined by partitioning parameters as $\mathbf{P} = ({\mathbf{a}_1}^T, \ldots, {\mathbf{a}_m}^T, \ldots, {\mathbf{b}_1}^T, \ldots, {\mathbf{b}_n}^T)^T$
- A measurement vector $\mathbf{X}$ is defined as $({\mathbf{x}_{11}}^T, \ldots, {\mathbf{x}_{1m}}^T,\ {\mathbf{x}_{21}}^T, \ldots, {\mathbf{x}_{2m}}^T,\ \ldots,\ {\mathbf{x}_{n1}}^T, \ldots, {\mathbf{x}_{nm}}^T)^T$
- For each parameter vector, an estimated measurement $\hat{\mathbf{X}}$ is $({\hat{\mathbf{x}}_{11}}^T, \ldots, {\hat{\mathbf{x}}_{1m}}^T,\ {\hat{\mathbf{x}}_{21}}^T, \ldots, {\hat{\mathbf{x}}_{2m}}^T,\ \ldots,\ {\hat{\mathbf{x}}_{n1}}^T, \ldots, {\hat{\mathbf{x}}_{nm}}^T)^T$ and the corresponding error $({\epsilon_{11}}^T, \ldots, {\epsilon_{1m}}^T,\ {\epsilon_{21}}^T, \ldots, {\epsilon_{2m}}^T,\ \ldots,\ {\epsilon_{n1}}^T, \ldots, {\epsilon_{nm}}^T)^T$, where $\hat{\mathbf{x}}_{ij} \equiv \mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i)$ and $\epsilon_{ij} \equiv \mathbf{x}_{ij} - \hat{\mathbf{x}}_{ij}\ \forall\ i, j$
- With the above definitions, BA corresponds to minimizing $\sum_{i=1}^{n} \sum_{j=1}^{m} ||\epsilon_{ij}||^2 = ||\mathbf{X} - \hat{\mathbf{X}}||^2$ over $\mathbf{P}$, which is a nonlinear least squares problem

Jacobian block structure

- The Jacobian $\mathbf{J} = \frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{P}}$ has a block structure $[\,\mathbf{A}|\mathbf{B}\,]$, where $\mathbf{A} = \left[\frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{a}}\right]$ and $\mathbf{B} = \left[\frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{b}}\right]$

- The LM updating vector δ is partitioned as $({\delta_{\mathbf{a}}}^T,\ {\delta_{\mathbf{b}}}^T)^T$

- The normal equations become

$$\left[\begin{array}{c|c} \mathbf{A}^T\mathbf{A} & \mathbf{A}^T\mathbf{B} \\ \hline \mathbf{B}^T\mathbf{A} & \mathbf{B}^T\mathbf{B} \end{array}\right] \left(\begin{array}{c} \delta_{\mathbf{a}} \\ \hline \delta_{\mathbf{b}} \end{array}\right) = \left(\begin{array}{c} \mathbf{A}^T\epsilon \\ \hline \mathbf{B}^T\epsilon \end{array}\right)$$

- The lhs matrix above is sparse due to $\mathbf{A}$ and $\mathbf{B}$ being sparse: $\frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_k} = \mathbf{0},\ \forall\, j \neq k$ and $\frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_k} = \mathbf{0},\ \forall\, i \neq k$

- This is the so-called *primary structure* of BA

A simple case with $n = 4$, $m = 3$

- Assume all points are seen in all images
- The measurement vector $\mathbf{x} = (\mathbf{x}_{11}{}^T, \mathbf{x}_{12}{}^T, \mathbf{x}_{13}{}^T, \mathbf{x}_{21}{}^T, \mathbf{x}_{22}{}^T, \mathbf{x}_{23}{}^T, \mathbf{x}_{31}{}^T, \mathbf{x}_{32}{}^T, \mathbf{x}_{33}{}^T, \mathbf{x}_{41}{}^T, \mathbf{x}_{42}{}^T, \mathbf{x}_{43}{}^T)^T$
- The parameter vector $\mathbf{P} = (\mathbf{a}_1{}^T, \mathbf{a}_2{}^T, \mathbf{a}_3{}^T, \mathbf{b}_1{}^T, \mathbf{b}_2{}^T, \mathbf{b}_3{}^T, \mathbf{b}_4{}^T)^T$
- The LM updating vector $\delta = (\delta_{\mathbf{a}_1}{}^T, \delta_{\mathbf{a}_2}{}^T, \delta_{\mathbf{a}_3}{}^T, \delta_{\mathbf{b}_1}{}^T, \delta_{\mathbf{b}_2}{}^T, \delta_{\mathbf{b}_3}{}^T, \delta_{\mathbf{b}_4}{}^T)^T$
- Let $\mathbf{A}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{a}_j}$ and $\mathbf{B}_{ij} = \frac{\partial \hat{\mathbf{x}}_{ij}}{\partial \mathbf{b}_i}$

A simple case with $n = 4$, $m = 3$ (cont'd)

- The Jacobian $\mathbf{J}$ in block form:

$$
\frac{\partial \hat{\mathbf{X}}}{\partial \mathbf{P}} =
\begin{array}{c}
\\
\mathbf{x}_{11}\\ \mathbf{x}_{12}\\ \mathbf{x}_{13}\\ \mathbf{x}_{21}\\ \mathbf{x}_{22}\\ \mathbf{x}_{23}\\ \mathbf{x}_{31}\\ \mathbf{x}_{32}\\ \mathbf{x}_{33}\\ \mathbf{x}_{41}\\ \mathbf{x}_{42}\\ \mathbf{x}_{43}
\end{array}
\begin{array}{c}
\begin{array}{ccccccc}
\mathbf{a_1}^T & \mathbf{a_2}^T & \mathbf{a_3}^T & \mathbf{b_1}^T & \mathbf{b_2}^T & \mathbf{b_3}^T & \mathbf{b_4}^T
\end{array}\\
\begin{pmatrix}
\mathbf{A}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{11} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & \mathbf{A}_{12} & \mathbf{0} & \mathbf{B}_{12} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & \mathbf{0} & \mathbf{A}_{13} & \mathbf{B}_{13} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\
\mathbf{A}_{21} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{21} & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & \mathbf{A}_{22} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{22} & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & \mathbf{0} & \mathbf{A}_{23} & \mathbf{0} & \mathbf{B}_{23} & \mathbf{0} & \mathbf{0}\\
\mathbf{A}_{31} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{31} & \mathbf{0}\\
\mathbf{0} & \mathbf{A}_{32} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{32} & \mathbf{0}\\
\mathbf{0} & \mathbf{0} & \mathbf{A}_{33} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{33} & \mathbf{0}\\
\mathbf{A}_{41} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{41}\\
\mathbf{0} & \mathbf{A}_{42} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{42}\\
\mathbf{0} & \mathbf{0} & \mathbf{A}_{43} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{B}_{43}
\end{pmatrix}
\end{array}
\tag{1}
$$

A simple case with $n = 4$, $m = 3$ (cont'd)

- Approximate Hessian in block form:

$$
\mathbf{J}^T\mathbf{J} = \begin{array}{c c}
 & \begin{array}{ccccccc} \mathbf{a}_1{}^T & \mathbf{a}_2{}^T & \mathbf{a}_3{}^T & \mathbf{b}_1{}^T & \mathbf{b}_2{}^T & \mathbf{b}_3{}^T & \mathbf{b}_4{}^T \end{array} \\
\begin{array}{c} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \\ \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \\ \mathbf{b}_4 \end{array} &
\begin{pmatrix}
\mathbf{U}_1 & 0 & 0 & \mathbf{W}_{11} & \mathbf{W}_{21} & \mathbf{W}_{31} & \mathbf{W}_{41} \\
0 & \mathbf{U}_2 & 0 & \mathbf{W}_{12} & \mathbf{W}_{22} & \mathbf{W}_{32} & \mathbf{W}_{42} \\
0 & 0 & \mathbf{U}_3 & \mathbf{W}_{13} & \mathbf{W}_{23} & \mathbf{W}_{33} & \mathbf{W}_{43} \\
\mathbf{W}_{11}^T & \mathbf{W}_{12}^T & \mathbf{W}_{13}^T & \mathbf{V}_1 & 0 & 0 & 0 \\
\mathbf{W}_{21}^T & \mathbf{W}_{22}^T & \mathbf{W}_{23}^T & 0 & \mathbf{V}_2 & 0 & 0 \\
\mathbf{W}_{31}^T & \mathbf{W}_{32}^T & \mathbf{W}_{33}^T & 0 & 0 & \mathbf{V}_3 & 0 \\
\mathbf{W}_{41}^T & \mathbf{W}_{42}^T & \mathbf{W}_{43}^T & 0 & 0 & 0 & \mathbf{V}_4
\end{pmatrix}
\end{array}
\equiv \begin{pmatrix} \mathbf{U} & \mathbf{W} \\ \mathbf{W}^T & \mathbf{V} \end{pmatrix},
$$

(2)

where

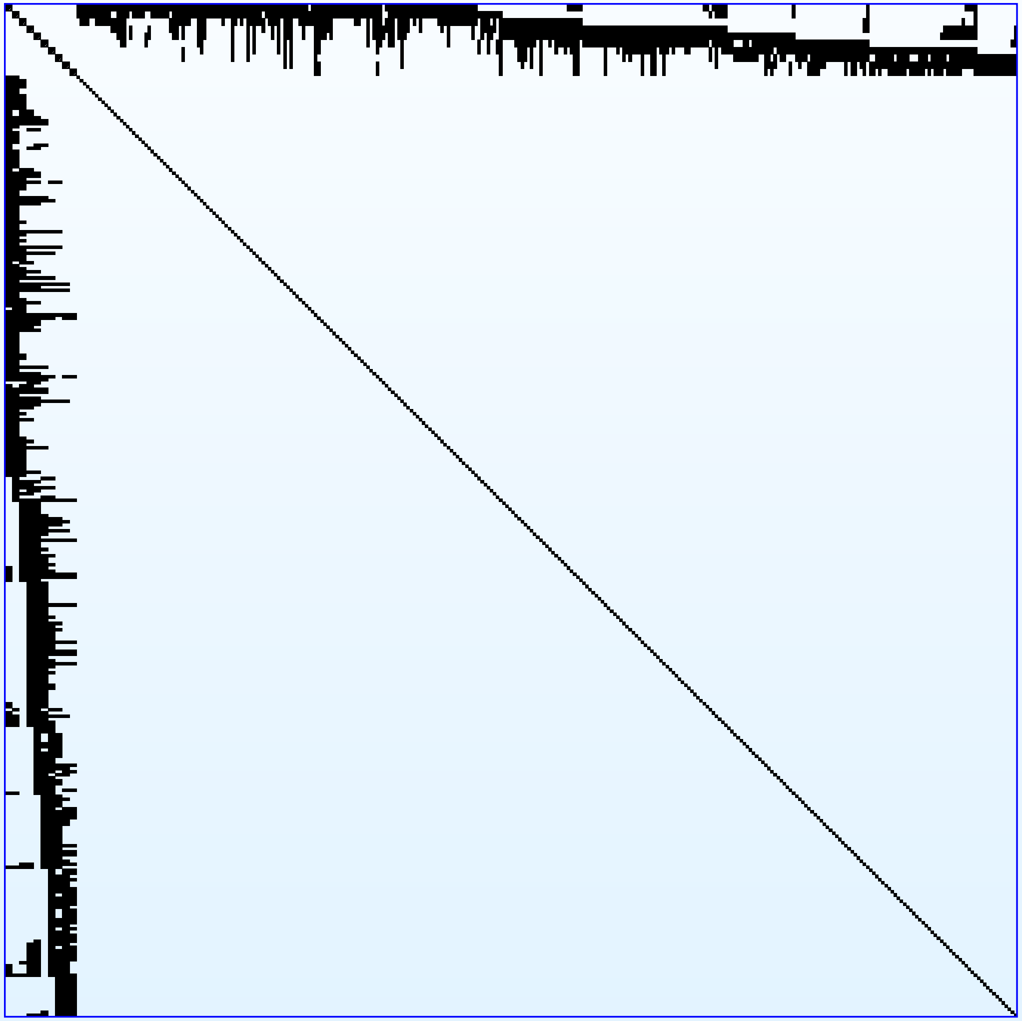
$$
\mathbf{U}_j \equiv \sum_{i=1}^{4} \mathbf{A}_{ij}^T \mathbf{A}_{ij},
$$
$$
\mathbf{V}_i \equiv \sum_{j=1}^{3} \mathbf{B}_{ij}^T \mathbf{B}_{ij},
$$
$$
\mathbf{W}_{ij} = \mathbf{A}_{ij}^T \mathbf{B}_{ij}
$$

- The above generalize directly to arbitrary n and m
- $\mathbf{U}$ and $\mathbf{V}$ are block diagonal, $\mathbf{W}$ arbitrarily sparse

$\mathbf{J}^T\mathbf{J}$ sparsity pattern for a real problem

- Oxford's "basement" sequence

Note the arrowhead pattern



- Matrix is 992×992, black pixels denote nonzero elements

Solving the augmented normal equations

- The augmented normal equations $(\mathbf{J}^T\mathbf{J} + \mu\mathbf{I})\delta_{\mathbf{p}} = \mathbf{J}^T\epsilon$ take the form

(3) $$\begin{pmatrix} \mathbf{U}^* & \mathbf{W} \\ \mathbf{W}^T & \mathbf{V}^* \end{pmatrix} \begin{pmatrix} \delta_{\mathbf{a}} \\ \delta_{\mathbf{b}} \end{pmatrix} = \begin{pmatrix} \epsilon_{\mathbf{a}} \\ \epsilon_{\mathbf{b}} \end{pmatrix}$$

- Performing block Gaussian elimination in the lhs matrix, $\delta_{\mathbf{a}}$ is determined with Cholesky from $\mathbf{V}^*$'s Schur complement:

(4) $$(\mathbf{U}^* - \mathbf{W}\ \mathbf{V}^{*\,-1}\ \mathbf{W}^T)\ \delta_{\mathbf{a}} = \epsilon_{\mathbf{a}} - \mathbf{W}\ \mathbf{V}^{*\,-1}\ \epsilon_{\mathbf{b}}$$

This is not alternation!

- Note that $\mathbf{V}^{*\,-1} = \begin{pmatrix} \mathbf{V}_1^{*\,-1} & \mathbf{0} & \cdots \\ \mathbf{0} & \mathbf{V}_2^{*\,-1} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}$
- Why solve for $\delta_{\mathbf{a}}$ first? Typically $m << n$
- $\delta_{\mathbf{b}}$ can be computed by back substitution into

(5) $$\mathbf{V}^*\ \delta_{\mathbf{b}} = \epsilon_{\mathbf{b}} - \mathbf{W}^T\ \delta_{\mathbf{a}}$$

The reduced camera matrix

- The lhs matrix $\mathbf{S} \equiv \mathbf{U}^* - \mathbf{W}\ {\mathbf{V}^*}^{-1}\ \mathbf{W}^T$ is referred to as the *reduced camera matrix*
- Since not all scene points appear in all cameras, $\mathbf{S}$ is sparse. This is known as *secondary structure*
- The secondary structure depends on the observed point tracks and is hard to predict. This not crucial up to a few hundred cameras
- Two classes of applications for very large datasets
  - visual mapping: extended areas are traversed, limited image overlap (sparse $\mathbf{S}$)
  - centered-object: a large number of overlapping images taken in a small area (dense $\mathbf{S}$)

Courtesy of Agarwal et al., ECCV'10

Dealing with the RCM

- Store as dense, decompose with ordinary linear algebra
  - M. Lourakis, A. Argyros: SBA: A Software Package For Generic Sparse Bundle Adjustment. ACM Trans. Math. Softw. 36(1): (2009)
  - C. Engels, H. Stewenius, D. Nister: Bundle Adjustment Rules. Photogrammetric Computer Vision (PCV), 2006.
- Store as sparse, factorize with sparse direct solvers
  - K. Konolige: Sparse Sparse Bundle Adjustment. BMVC 2010: 1-11
- Store as sparse, use conjugate gradient methods
  memory efficient, iterative, precoditioners necessary!
  - S. Agarwal, N. Snavely, S.M. Seitz, R. Szeliski: Bundle Adjustment in the Large. ECCV (2) 2010: 29-42
  - M. Byrod, K. Astrom: Conjugate Gradient Bundle Adjustment. ECCV (2) 2010: 114-127
- Avoid storing altogether
  - C. Wu, S. Agarwal, B. Curless, S.M. Seitz: Multicore Bundle Adjustment. CVPR 2011: 30 57-3064
  - M. Lourakis: Sparse Non-linear Least Squares Optimization for Geometric Vision. ECCV (2) 2010: 43-56

Reducing the cost of BA

- BA is not a cheap operation, thus for certain applications it may take unacceptably long to complete
- A large body of work is devoted to reducing BA's size or frequency of invocation
- Divide-and-conquer approaches
  - K. Ni, D. Steedly, F. Dellaert: Out-of-Core Bundle Adjustment for Large-Scale 3D Reconstruction. ICCV 2007: 1-8
  - H.-Y. Shum, Z. Zhang, Q. Ke: Efficient Bundle Adjustment with Virtual Key Frames: A Hierarchical Approach to Multi-Frame Structure from Motion. CVPR 1999: 2538-2543
- BA in a sliding time window (local BA)
  - E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, P. Sayd: Real Time Localization and 3D Reconstruction. CVPR (1) 2006: 363-370
- Solve the RCM fewer times: Dog-leg in place of LM
  - M. Lourakis, A. Argyros: Is Levenberg-Marquardt the Most Efficient Optimization Algorithm for Implementing Bundle Adjustment?. ICCV 2005: 1526-1531

`sba` historical overview

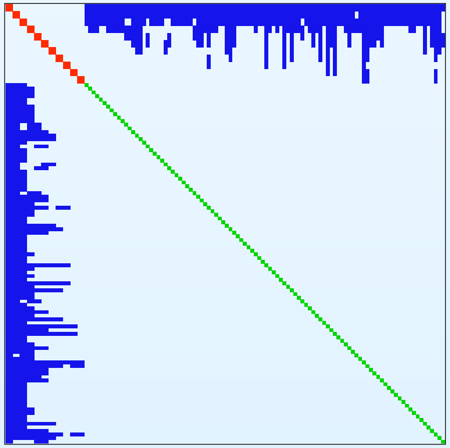
- *"Every good work of software starts by scratching a developer's personal itch"*. Eric S. Raymond, open source proponent
- Started as an internal project in 2004, as a component for camera tracking
- First public version released in September 2004; new versions once or twice a year
- The `sba` library implements the presented scheme in C, forming $\mathbf{S}$ as a dense matrix and relying on LAPACK for linear algebra
- Has spawned two side free source projects
  - LM for dense least squares: `http://www.ics.forth.gr/~lourakis/levmar`
  - LM for arbitrarily sparse least squares: `http://www.ics.forth.gr/~lourakis/sparseLM`

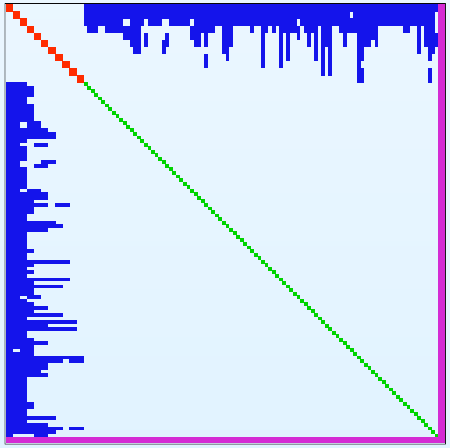
sba **features**

- Supports generic BA by accepting user-defined projection functions; includes code for Euclidean BA as example
- This lends it the versatility to support various BA flavors with the same optimization engine: e.g. BA including/excluding camera extrinsics and/or distortion parameters, projective BA, non-pinhole cameras, etc
- Provides efficient mechanisms for approximating the Jacobians via finite differences and checking the correctness of user supplied ones
- Provides support for intersectioning/resectioning problems (constant camera poses/scene structure, resp.)
- Usable in MATLAB though a MEX interface
- Supports robust cost functions and projection covariances
- Highly optimized yet portable
- Can take advantage of multicore architectures through the PLASMA library (`http://icl.cs.utk.edu/plasma/`)

sba weaknesses

- Storing the full RCM makes it inefficient for very large datasets, e.g. long, weakly connected image sequences (i.e., visual mapping). However, local BA is a viable alternative in some of these cases
- A few interesting practical situations violate its underlying assumption regarding the problem's sparsity pattern, rendering it inapplicable. E.g., fixed but unknown intrinsics shared by all cameras
- Example: Hessians corresponding to BA for motion and structure (left) and BA for motion, structure and shared intrinsics (right)





`sba` usage

- `sba` is considered to be the standard BA implementation
- Best suited to processing small to medium-sized datasets
- Has over 280 citations according to Google Scholar, well above 100K page loads for its webpage
- Was used as the optimization engine of *Bundler*, the SaM system used in the *Photo Tourism* system (`http://phototour.cs.washington.edu`) that was the predecessor of Microsoft's *Photosynth*
- Other sample applications include multiview reconstruction, camera & camera networks calibration, camera tracking, visual SLAM, catadioptric imaging, geocoding, face modeling, autonomous UAVs, remote sensing, and more
- Released under a dual licensing scheme: GPL + proprietary license, thereby creating an income from commercial applications

User feedback

- Compilation problems are by far the most common topic of user inquiries
- Students often ask to have (part of) their homework done
- Most users send thank-you notes
- Others do not want to spend time perusing the documentation but prefer to ask direct questions
- A few knowledgeable users report bugs or suggest extensions
- Even fewer contribute code snippets
- In a couple of cases, users have reported that `sba` has inspired them to do further research

Lessons learnt

- The devil is indeed in the details!
- An efficient implementation should be cache-oblivious so as to minimize data movement across a computer's memory hierarchy
- Releasing code in public motivates oneself to write & maintain better code. Seeing it being used by others can be very rewarding
- Good documentation is important
- Need to set up tools for easy communication and sharing of code, knowledge, experiences and problems among the user community (cf. SourceForge facility)
- The vision community is in need of wider adoption of the free/open source culture
- Should have been given a better name!

Further reading

- B. Triggs, P.F. McLauchlan, R.I. Hartley, A.W. Fitzgibbon: Bundle Adjustment - A Modern Synthesis. Workshop on Vision Algorithms 1999: 298-372
- Y. Jeong, D. Nister, D. Steedly, R. Szeliski, I.-S. Kweon: Pushing the Envelope of Modern Methods for Bundle Adjustment. CVPR 2010: 1474-1481
- M. Lourakis, A. Argyros: SBA: A Software Package For Generic Sparse Bundle Adjustment. ACM Trans. Math. Softw. 36(1): (2009)
- C. Engels, H. Stewenius, D. Nister: Bundle Adjustment Rules. Photogrammetric Computer Vision (PCV), 2006
- K. Konolige: Sparse Sparse Bundle Adjustment. BMVC 2010: 1-11
- S. Agarwal, N. Snavely, S.M. Seitz, R. Szeliski: Bundle Adjustment in the Large. ECCV (2) 2010: 29-42
- M. Byrod, K. Astrom: Conjugate Gradient Bundle Adjustment. ECCV (2) 2010: 114-127
- C. Wu, S. Agarwal, B. Curless, S.M. Seitz: Multicore Bundle Adjustment. CVPR 2011: 30 57-3064
- M. Lourakis: Sparse Non-linear Least Squares Optimization for Geometric Vision.

  ECCV (2) 2010: 43-56

Conclusions

- Presented the mathematical theory behind an LM-based sparse bundle adjustment algorithm
- Described `sba`, a freely available C/C++ software package for generic sparse BA
- `sba` can be obtained from `http://www.ics.forth.gr/~lourakis/sba`
- More details on `sba` can be found in
  Lourakis & Argyros, "SBA: A Software Package For Generic Sparse Bundle Adjustment. ACM Trans. Math. Softw.", 36(1): 2009
  and in the (slightly outdated)
  Lourakis & Argyros, "The Design and Implementation of a Generic Sparse Bundle Adjustment Software Package Based on the Levenberg-Marquardt Algorithm", Tech. Rep. FORTH-ICS TR-340-2004, Aug. 2004
- Making your next project publicly available is worth considering!

The end

Any questions?