Chapter 7
Bundle Adjustment & Nonlinear Optimization

Multiple View Geometry
Summer 2025

Prof. Daniel Cremers
Chair of Computer Vision and Artificial Intelligence
School of Computation, Information and Technology
Technical University of Munich

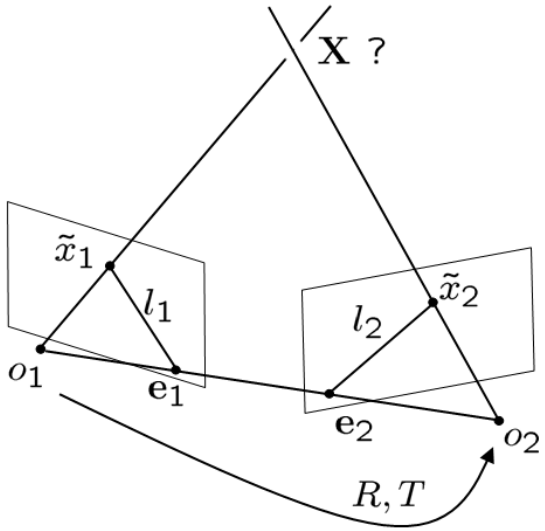
Overview

Optimality in Noisy Real World Conditions

In the previous chapters we discussed linear approaches to solve the structure and motion problem. In particular, the eight-point algorithm provides closed-form solutions to estimate the camera parameters and the 3D structure, based on singular value decomposition.

However, if we have noisy data $\tilde{\boldsymbol{x}}_1, \tilde{\boldsymbol{x}}_2$ (correspondences not exact or even incorrect), then we have no guarantee

- that R and T are as close as possible to the true solution.
- that we will get a consistent reconstruction.



Statistical Approaches to Cope with Noise

The linear approaches are elegant because optimal solutions to respective problems can be computed in closed form. However, they often fail when dealing with noisy and imprecise point locations. Since measurement noise is not explicitly considered or modeled, such spectral methods often provide suboptimal performance in noisy real-world conditions.

In order to take noise and statistical fluctuation into account, one can revert to a Bayesian formulation and determine the most likely camera transformation R, T and 'true' 2D coordinates $\boldsymbol{x}$ given the measured coordinates $\tilde{\boldsymbol{x}}$, by performing a maximum aposteriori estimate:

$$\arg \max_{\boldsymbol{x},R,T} \mathcal{P}(\boldsymbol{x}, R, T \mid \tilde{\boldsymbol{x}}) = \arg \max_{\boldsymbol{x},R,T} \mathcal{P}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}, R, T)\, \mathcal{P}(\boldsymbol{x}, R, T)$$

This approach will however involve modeling probability densities $\mathcal{P}$ on the fairly complicated space $SO(3) \times \mathbb{S}^2$ of rotation and translation parameters, as $R \in SO(3)$ and $T \in \mathbb{S}^2$ (3D translation with unit length).

Bundle Adjustment and Nonlinear Optimization

Under the assumption that the observed 2D point coordinates $\tilde{\boldsymbol{x}}$ are corrupted by zero-mean Gaussian noise, maximum likelihood estimation leads to bundle adjustment:

$$E(R, T, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_N) = \sum_{j=1}^{N} \left|\tilde{\boldsymbol{x}}_1^j - \pi(\boldsymbol{X}_j)\right|^2 + \left|\tilde{\boldsymbol{x}}_2^j - \pi(R, T, \boldsymbol{X}_j)\right|^2$$

It aims at minimizing the reprojection error between the observed 2D coordinates $\tilde{\boldsymbol{x}}_i^j$ and the projected 3D coordinate $\boldsymbol{X}_j$ (w.r.t. camera 1). Here $\pi(R, T, \boldsymbol{X}_j)$ denotes the perspective projection of $\boldsymbol{X}_j$ after rotation and translation.

For the general case of m images, we get:

$$E(\{R_i, T_i\}_{i=1..m}, \{\boldsymbol{X}_j\}_{j=1..N}) = \sum_{i=1}^{m} \sum_{j=1}^{N} \theta_{ij} \left|\tilde{\boldsymbol{x}}_i^j - \pi(R_i, T_i, \boldsymbol{X}_j)\right|^2,$$

with $T_1 = 0$ and $R_1 = \mathbf{1}$. $\theta_{ij} = 1$ if point j is visible in image i, $\theta_{ij} = 0$ else. The above problems are non-convex.

Different Parameterizations of the Problem

The same optimization problem can be parameterized differently. For example, we can introduce $\boldsymbol{x}_i^j$ to denote the true 2D coordinate associated with the measured coordinate $\tilde{\boldsymbol{x}}_i^j$:

$$E(\{\boldsymbol{x}_1^j, \lambda_1^j\}_{j=1..N}, R, T) = \sum_{j=1}^{N} \|\boldsymbol{x}_1^j - \tilde{\boldsymbol{x}}_1^j\|^2 + \|\tilde{\boldsymbol{x}}_2^j - \pi(R\lambda_1^j \boldsymbol{x}_1^j + T)\|^2.$$

Alternatively, we can perform a constrained optimization by minimizing a cost function (similarity to measurements):

$$E(\{\boldsymbol{x}_i^j\}_{j=1..N}, R, T) = \sum_{j=1}^{N} \sum_{i=1}^{2} \|\boldsymbol{x}_i^j - \tilde{\boldsymbol{x}}_i^j\|^2$$

subject to (consistent geometry):

$$\boldsymbol{x}_2^{j\top} \widehat{T} R \boldsymbol{x}_1^j = 0, \quad \boldsymbol{x}_1^{j\top} e_3 = 1, \quad \boldsymbol{x}_2^{j\top} e_3 = 1, \quad j = 1, \ldots, N.$$

On the History of Bundle Adjustment

Bundle adjustment emerged in the field of photogrammetry and geodesy. The aim is to jointly estimate the 3D coordinates of points and the camera parameters – typically the rigid body motion, but sometimes also intrinsic calibration parameters or radial distortion. Different models of the noise in the observed 2D points lead to different cost functions, zero-mean Gaussian noise being the most common assumption.

The approach is called bundle adjustment (Bündelausgleich) because it aims at adjusting the bundles of light rays emitted from the 3D points. The nonconvex optimization problems are typically solved with nonlinear least squares estimation.

A good overview can be found in:
Triggs, McLauchlan, Hartley, Fitzgibbon, “Bundle Adjustment – A Modern Synthesis”, ICCV Workshop 1999.

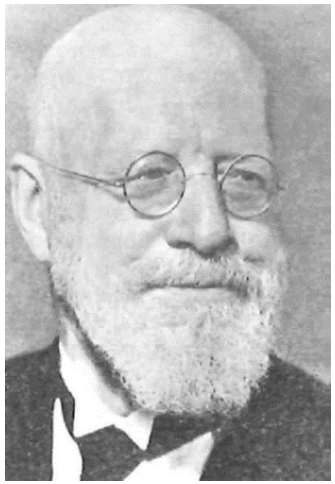
The fields of geodesy and photogrammetry have a long tradition with many pioneers. The following slides lists some of the most influential ones.

Pioneers of Photogrammetry and Bundle Adjustment

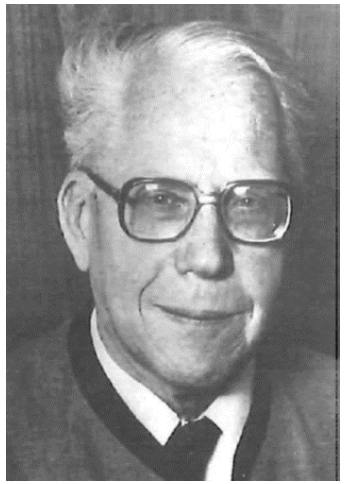
- **Carl Maximilian von Bauernfeind**, founder of TU Munich, was a geodesist. He strongly believed in bringing together engineers and applied mathematicians.
- **Eduard Dolezal** founded the International Society for Photogrammetry and Remote Sensing (Vienna 1910).
- **Sebastian Finsterwalder** pioneered analytical photogrammetry – orientation estimations from feature points. The reconstruction of 3D maps from camera images often took years of calculations.
- The mathematician **Karl Rinner** advocated the use of vector algebra and projective geometry.
- **Otto von Gruber** developed linearization techniques for nonlinear least squares problems.
- **Helmut H. Schmid** developed nonlinear least-squares estimation of 3D point and camera parameters. He moved to the US in 1945 and together with **Duane C. Brown** they deployed these methods on the largest computers of their time at the Ballistic Research Laboratories (1952-55).

Pioneers of Photogrammetry and Bundle Adjustment

Eduard Dolezal
(1862-1955)



Sebastian Finsterwalder
(1862-1951)



Karl Rinner
(1912-1991)

Otto von Gruber
(1884-1942)

Helmut H. Schmid
(1914-1998)

Duane C. Brown
(1929-1994)

Nonlinear Programming

Nonlinear programming denotes the process of iteratively solving a nonlinear optimization problem, i.e. a problem involving the maximization or minimization of an objective function over a set of real variables under a set of equality or inequality constraints.

There are numerous methods and techniques. Good overviews of respective methods can be found for example in Bersekas (1999) “Nonlinear Programming”, Nocedal & Wright (1999), “Numerical Optimization” or Luenberger & Ye (2008), “Linear and nonlinear programming”.

Depending on the cost function, different algorithms are employed. In the following, we will discuss (nonlinear) least squares estimation and several popular iterative techniques for nonlinear optimization:

- the gradient descent,
- Newton methods,
- the Gauss-Newton algorithm,
- the Levenberg-Marquardt algorithm.

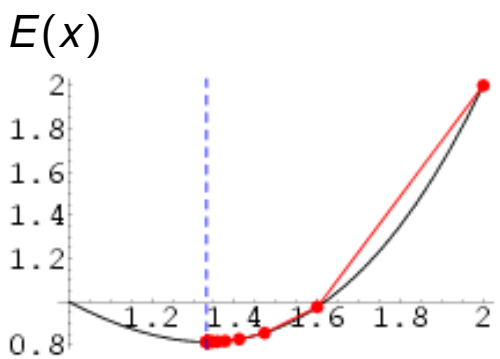
Gradient Descent

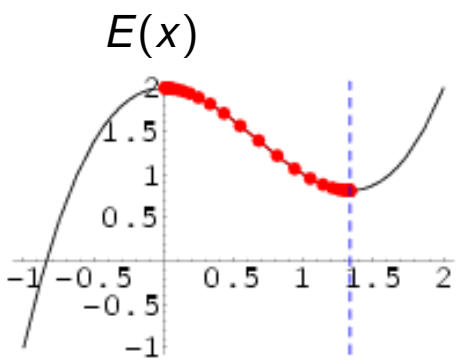
Gradient descent or steepest descent is a first-order optimization method. It aims at computing a local minimum of a (generally) non-convex cost function by iteratively stepping in the direction in which the energy decreases most. This is given by the negative energy gradient.

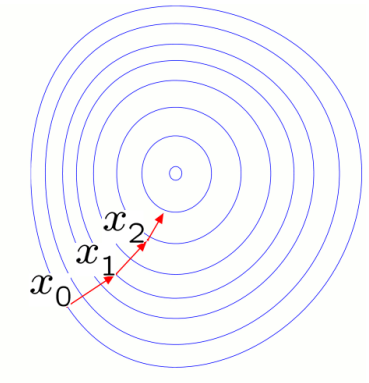
To minimize a real-valued cost $E : \mathbb{R}^n \to \mathbb{R}$, the gradient flow for $E(x)$ is defined by the differential equation:

$$\begin{cases} x(0) = x_0 \\ \frac{dx}{dt} = -\frac{dE}{dx}(x) \end{cases}$$

Discretization: $x_{k+1} = x_k - \epsilon \frac{dE}{dx}(x_k), \qquad k = 0, 1, 2, \ldots.$







Gradient Descent

Under certain conditions on $E(x)$, the gradient descent iteration

$$x_{k+1} = x_k - \epsilon \frac{dE}{dx}(x_k), \qquad k = 0, 1, 2, \ldots$$

converges to a local minimum. For the case of convex E, this will also be the global minimum. The step size ϵ can be chosen differently in each iteration.

Gradient descent is a popular and broadly applicable method. It is typically not the fastest solution to compute minimizers because the asymptotic convergence rate is often inferior to that of more specialized algorithms. First-order methods with optimal convergence rates were pioneered by Yuri Nesterov.

In particular, highly anisotropic cost functions (with strongly different curvatures in different directions) require many iterations and trajectories tend to zig-zag. Locally optimal step sizes in each iteration can be computed by line search. For specific cost functions, alternative techniques such as the conjugate gradient method, Newton methods, or the BFGS method are preferable.

Linear Least Squares Estimation

Ordinary least squares or linear least squares is a method to for estimating a set of parameters $x \in \mathbb{R}^d$ in a linear regression model. Assume for each input vector $b_i \in \mathbb{R}^d, i \in \{1, .., n\}$, we observe a scalar response $a_i \in \mathbb{R}$. Assume there is a linear relationship of the form

$$a_i = b_i^\top x + \eta_i$$

with an unknown vector $x \in \mathbb{R}^d$ and zero-mean Gaussian noise $\eta \sim \mathcal{N}(0, \boldsymbol{\Sigma})$ with a diagonal covariance matrix of the form $\boldsymbol{\Sigma} = \sigma^2 \boldsymbol{I_n}$. Maximum likelihood estimation of x leads to the ordinary least squares problem:

$$\min_x \sum_i (a_i - x^\top b_i)^2 = (a - \boldsymbol{B}x)^\top (a - \boldsymbol{B}x).$$

Linear least squares estimation was introduced by Legendre (1805) and Gauss (1795/1809). When asking for which noise distribution the optimal estimator was the arithmetic mean, Gauss invented the normal distribution.

Linear Least Squares Estimation

For general $\boldsymbol{\Sigma}$, we get the generalized least squares problem:

$$\min_x (a - \boldsymbol{B}x)^\top \boldsymbol{\Sigma}^{-1} (a - \boldsymbol{B}x).$$

This is a quadratic cost function with positive definite $\boldsymbol{\Sigma}^{-1}$. It has the closed-form solution:

$$\begin{aligned} \hat{x} &= \arg\min_x (a - \boldsymbol{B}x)^\top \boldsymbol{\Sigma}^{-1} (a - \boldsymbol{B}x) \\ &= (\boldsymbol{B}^\top \boldsymbol{\Sigma}^{-1} \boldsymbol{B})^{-1} \boldsymbol{B}^\top \boldsymbol{\Sigma}^{-1} a. \end{aligned}$$

If there is no correlation among the observed variances, then the matrix $\boldsymbol{\Sigma}$ is diagonal. This case is referred to as weighted least squares:

$$\min_x \sum_i w_i (a_i - x^\top b_i)^2, \qquad \text{with } w_i = \sigma_i^{-2}.$$

For the case of unknown matrix $\boldsymbol{\Sigma}$, there exist iterative estimation algorithms such as feasible generalized least squares or iteratively reweighted least squares.

Iteratively Reweighted Least Squares

The method of iteratively reweighted least squares aims at minimizing generally non-convex optimization problems of the form

$$\min_x \sum_i w_i(x)|a_i - f_i(x)|^2,$$

with some known weighting function $w_i(x)$. A solution is obtained by iterating the following problem:

$$x_{t+1} = \arg\min_x \sum_i w_i(x_t)\,|a_i - f_i(x)|^2$$

For the case that f_i is linear, i.e. $f_i(x) = x^\top b_i$, each subproblem

$$x_{t+1} = \arg\min_x \sum_i w_i(x_t)\,|a_i - x^\top b_i|^2$$

is simply a weighted least squares problem that can be solved in closed form. Nevertheless, this iterative approach will generally not converge to a global minimum of the original (nonconvex) problem.

Nonlinear Least Squares Estimation

Nonlinear least squares estimation aims at fitting observations (a_i, b_i) with a nonlinear model of the form $a_i \approx f(b_i, x)$ for some function f parameterized with an unknown vector $x \in \mathbb{R}^d$. Minimizing the sum of squares error

$$\min_x \sum_i r_i(x)^2, \qquad \text{with } r_i(x) = a_i - f(b_i, x),$$

is generally a non-convex optimization problem.

The optimality condition is given by

$$\sum_i r_i \frac{\partial r_i}{\partial x_j} = 0, \quad \forall j \in \{1, .., d\}.$$

Typically one cannot directly solve these equation. Yet, there exist iterative algorithms for computing approximate solutions, including Newton methods, the Gauss-Newton algorithm and the Levenberg-Marquardt algorithm.

Newton Methods for Optimization

Newton methods are second order methods: In contrast to first-order methods like gradient descent, they also make use of second derivatives. Geometrically, Newton method iteratively approximate the cost function $E(x)$ quadratically and takes a step to the minimizer of this approximation.

Let x_t be the estimated solution after t iterations. Then the Taylor approximation of $E(x)$ in the vicinity of this estimate is:

$$E(x) \approx E(x_t) + g^\top (x - x_t) + \frac{1}{2}(x - x_t)^\top \boldsymbol{H}(x - x_t),$$

The first and second derivative are denoted by the Jacobian $g = dE/dx(x_t)$ and the Hessian matrix $d^2E/dx^2(x_t)$. For this second-order approximation, the optimality condition is:

$$\frac{dE}{dx} = g + \boldsymbol{H}(x - x_t) = 0 \qquad (*)$$

Setting the next iterate to the minimizer x leads to:

$$x_{t+1} = x_t - \boldsymbol{H}^{-1} g.$$

Newton Methods for Optimization

In practice, one often choses a more conservative step size $\gamma \in (0, 1)$:

$$x_{t+1} = x_t - \gamma \boldsymbol{H}^{-1} g.$$

When applicable, second-order methods are often faster than first-order methods, at least when measured in number of iterations. In particular, there exists a local neighborhood around each optimum where the Newton method converges quadratically for $\gamma = 1$ (if the Hessian is invertible and Lipschitz continuous).

For large optimization problems, computing and inverting the Hessian may be challenging. Moreover, since this problem is often not parallelizable, some second order methods do not profit from GPU acceleration. In such cases, one can aim to iteratively solve the extremality condition $(*)$.

In case that $\boldsymbol{H}$ is not positive definite, there exist quasi-Newton methods which aim at approximating $\boldsymbol{H}$ or $\boldsymbol{H}^{-1}$ with a positive definite matrix.

The Gauss-Newton Algorithm

The Gauss-Newton algorithm is a method to solve non-linear least-squares problems of the form:

$$\min_x \sum_i r_i(x)^2.$$

It can be derived as an approximation to the Newton method. The latter iterates:

$$x_{t+1} = x_t - \boldsymbol{H}^{-1} g.$$

with the gradient g:

$$g_j = 2 \sum_i r_i \frac{\partial r_i}{\partial x_j},$$

and the Hessian $\boldsymbol{H}$:

$$H_{jk} = 2 \sum_i \left(\frac{\partial r_i}{\partial x_j} \frac{\partial r_i}{\partial x_k} + r_i \frac{\partial^2 r_i}{\partial x_j \partial x_k} \right).$$

Dropping the second order term leads to the approximation:

$$H_{jk} \approx 2 \sum_i J_{ij} J_{ik}, \qquad \text{with } J_{ij} = \frac{\partial r_i}{\partial x_j}.$$

The Gauss-Newton Algorithm

The approximation

$$\boldsymbol{H} \approx 2\boldsymbol{J}^\top \boldsymbol{J}, \qquad \text{with the Jacobian } \boldsymbol{J} = \frac{dr}{dx},$$

together with $g = 2\boldsymbol{J}^\top r$, leads to the Gauss-Newton algorithm:

$$x_{t+1} = x_t + \Delta, \qquad \text{with } \Delta = -(\boldsymbol{J}^\top \boldsymbol{J})^{-1} \boldsymbol{J}^\top r$$

In contrast to the Newton algorithm, the Gauss-Newton algorithm does not require the computation of second derivatives. Moreover, the above approximation of the Hessian is by construction positive definite.

This approximation of the Hessian is valid if

$$\left| r_i \frac{\partial^2 r_i}{\partial x_j \partial x_k} \right| \ll \left| \frac{\partial r_i}{\partial x_j} \frac{\partial r_i}{\partial x_k} \right|,$$

This is the case if the residuum r_i is small or if it is close to linear (in which case the second derivatives are small).

The Levenberg-Marquardt Algorithm

The Newton algorithm

$$x_{t+1} = x_t - \boldsymbol{H}^{-1} g,$$

can be modified (damped):

$$x_{t+1} = x_t - \left(\boldsymbol{H} + \lambda \boldsymbol{I_n}\right)^{-1} g,$$

to create a hybrid between the Newton method ($\lambda = 0$) and a gradient descent with step size $1/\lambda$ (for $\lambda \to \infty$) .

In the same manner, Levenberg (1944) suggested to damp the Gauss-Newton algorithm for nonlinear least squares:

$$x_{t+1} = x_t + \Delta, \qquad \text{with } \Delta = -\left(\boldsymbol{J}^\top \boldsymbol{J} + \lambda \boldsymbol{I_n}\right)^{-1} \boldsymbol{J}^\top r.$$

Marquardt (1963) suggested a more adaptive component-wise damping of the form:

$$\Delta = -\left(\boldsymbol{J}^\top \boldsymbol{J} + \lambda \mathit{diag}(\boldsymbol{J}^\top \boldsymbol{J})\right)^{-1} \boldsymbol{J}^\top r,$$

which avoids slow convergence in directions of small gradient.

Bundle Adjustment: Derivation of the Normal Equation

Bundle adjustment can be expressed as a nonlinear least squares problem of the form

$$F(x) = \frac{1}{2}\|r(x)\|_2^2 = \frac{1}{2}\sum_i \|r_i(x)\|_2^2\,,$$

with the vector $x = (x_p, x_\ell)$ containing the poses and landmarks. Denoting the updates Δx, the linearization

$$r(x) \approx r^0 + J\Delta x,$$

together with a damping term, leads to the quadratic problem

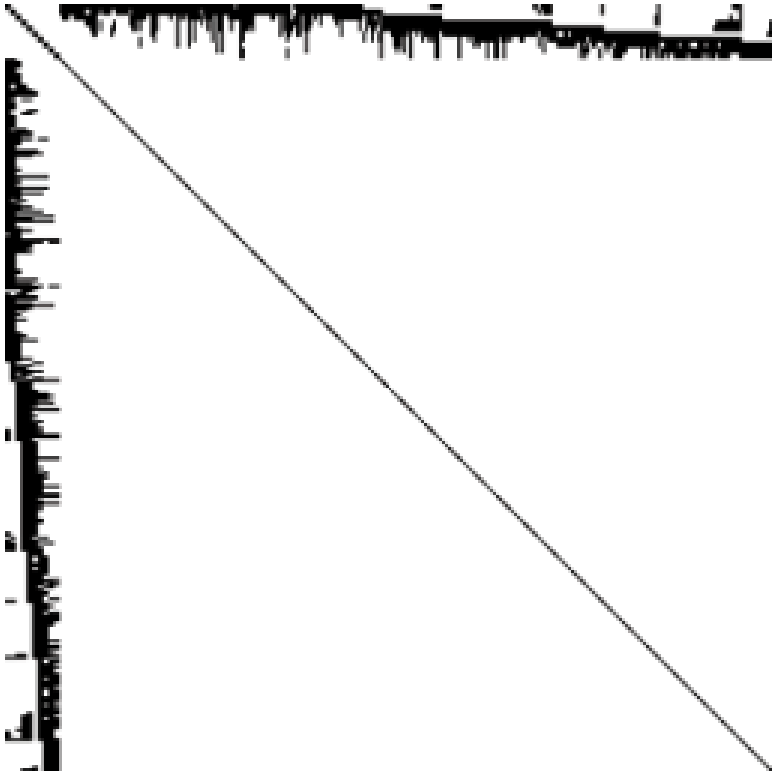
$$\min_{\Delta x_p, \Delta x_\ell} \frac{1}{2}\left(\left\| r^0 + \begin{pmatrix} J_p & J_\ell \end{pmatrix} \begin{pmatrix} \Delta x_p \\ \Delta x_\ell \end{pmatrix} \right\|_2^2 + \lambda \left\| \begin{pmatrix} D_p & D_\ell \end{pmatrix} \begin{pmatrix} \Delta x_p \\ \Delta x_\ell \end{pmatrix} \right\|_2^2\right).$$

The optimality condition is given by the normal equation:

$$\begin{pmatrix} U & W \\ W^\top & V \end{pmatrix} \begin{pmatrix} \Delta x_p \\ \Delta x_\ell \end{pmatrix} = \begin{pmatrix} b_p \\ b_\ell \end{pmatrix}.$$

The System Matrix

Solving the normal equation requires the inversion of the system matrix containing the matrices U, V and W, where V is typically block-diagonal and easy to invert.



Graphical visualization of the system matrix, where the black areas denote the non-zero elements. Clearly the large matrix V on the bottom right has a block diagonal structure.

(source: Lourakis)

The Schur Complement Trick

One can simplify the solution of the normal equation by means of the Schur complement trick. Left-multiplication of the normal equation with the invertible matrix

$$\begin{pmatrix} I & -WV^{-1} \\ 0 & I \end{pmatrix},$$

leads to

$$\begin{pmatrix} S & 0 \\ W^\top & V \end{pmatrix} \begin{pmatrix} \Delta x_p \\ \Delta x_\ell \end{pmatrix} = \begin{pmatrix} b_p - WV^{-1} b_\ell \\ b_\ell \end{pmatrix},$$

where $S = U - WV^{-1}W^\top$ is called the Schur complement. It is symmetric, positive definite and block structured.
We can therefore first estimate the typically small number of pose parameter updates Δx_p by solving the so-called reduced camera system:

$$S\Delta x_p = b_p - WV^{-1} b_\ell.$$

and subsequently plug these into the second equation to compute the update of the depth parameters Δx_ℓ.

Power Bundle Adjustment for Large Scale Reconstruction

The reduced camera system is commonly solved by an iterative solver such as the conjugate gradient method. In the paper Weber, Demmel, Chan, Cremers, CVPR 2023, we instead propose a solution that is faster, more accurate and more memory-efficient. To this end, we approximate the inverse of the Schur complement by a matrix power series.

Let us recall the geometric series

$$\sum_{i=0}^{\infty} r^i = \frac{1}{1-r}, \qquad \forall\, r \in (-1,1).$$

It generalizes to matrices $M \in \mathbb{R}^{n\times n}$. If $\|M\| \leq 1$ then:

$$(1-M)^{-1} = \sum_{i=0}^{m} M^i + R, \text{ with the residual } R = \sum_{i=m+1}^{\infty} M^i,$$

where

$$\|R\| \leq \frac{\|M\|^{m+1}}{1-\|M\|} \underset{m\to\infty}{\longrightarrow} 0.$$

Power Bundle Adjustment for Large Scale Reconstruction

We approximate the inverse Schur complement with a matrix power series. To this end, we rewrite the Schur matrix as:

$$S = U - WV^{-1}W^\top = U\left(I - U^{-1}WV^{-1}W^\top\right).$$

Hence the inverse is given by

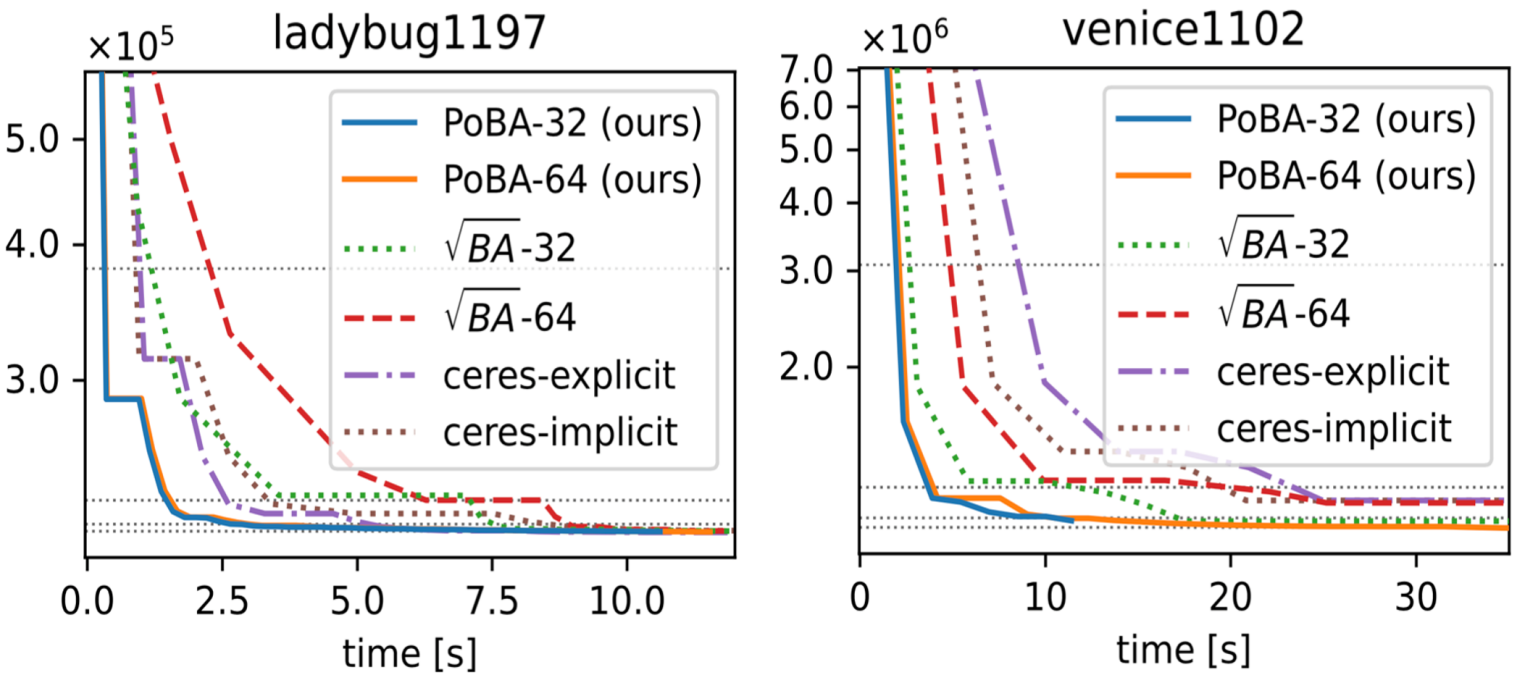
$$S^{-1} = (I - U^{-1}WV^{-1}W^\top)^{-1}\,U^{-1}.$$

Let μ be an eigenvalue of $U^{-1}WV^{-1}W^\top$. Then $\mu \in [0, 1)$. Therefore we can approximate:

$$S^{-1} \approx \sum_{i=0}^{m} (U^{-1}WV^{-1}W^\top)^i\,U^{-1}$$

This approximation of S^{-1} merely requires matrix multiplications. As a result, the updates for poses (in the reduced camera system) and landmarks can be computed much faster and in a more memory efficient way.

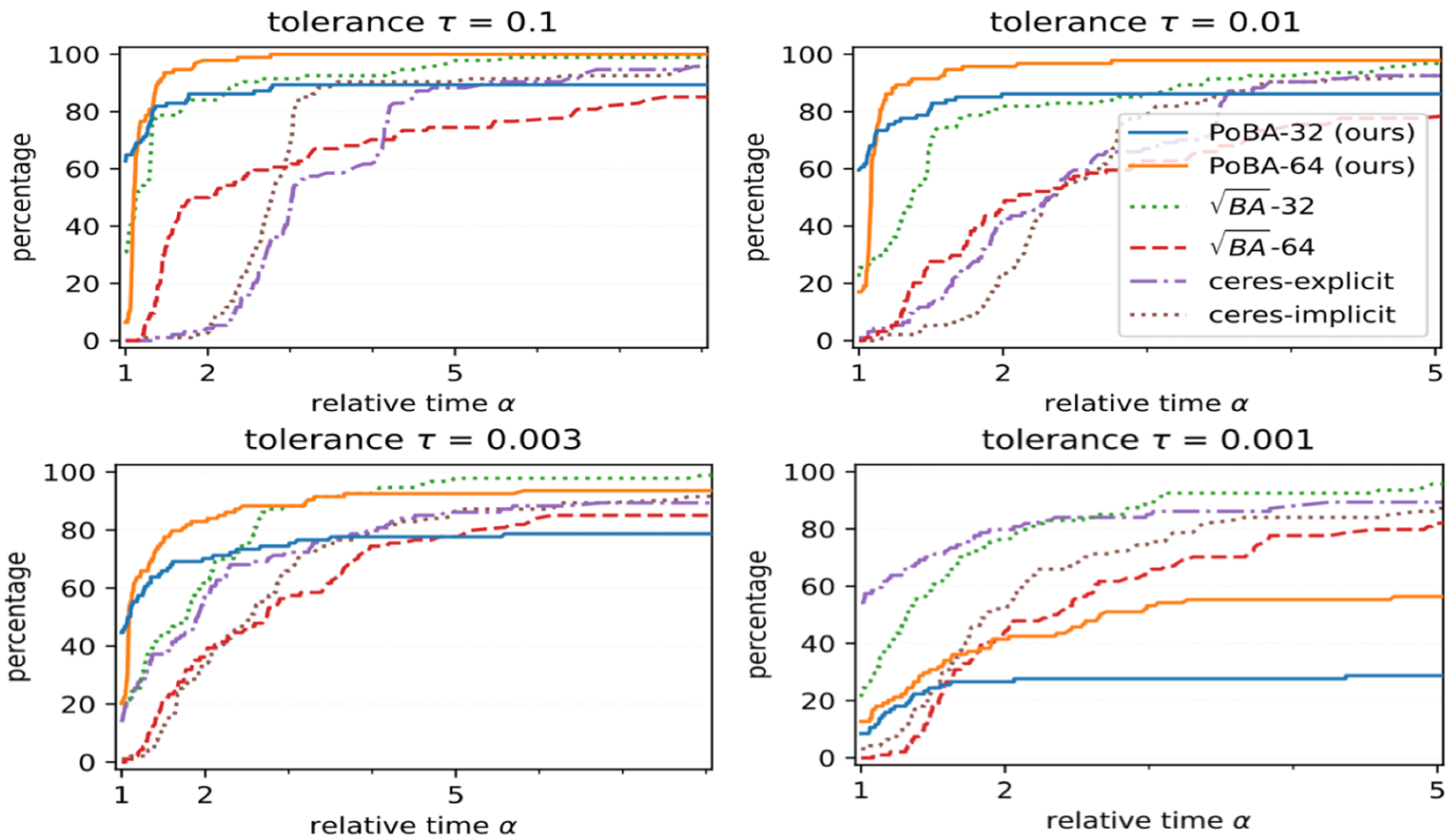
Power Bundle Adjustment for Large Scale Reconstruction



Convergence plot of the bundle adjustment cost as a function of time for the two problems “Ladybug” and “Venice”: The power bundle adjustment method is significantly faster than existing methods, both for a 32-bit and for a 64-bit implementation.

Weber, Demmel, Chan, Cremers, “Power BA”, CVPR 2023

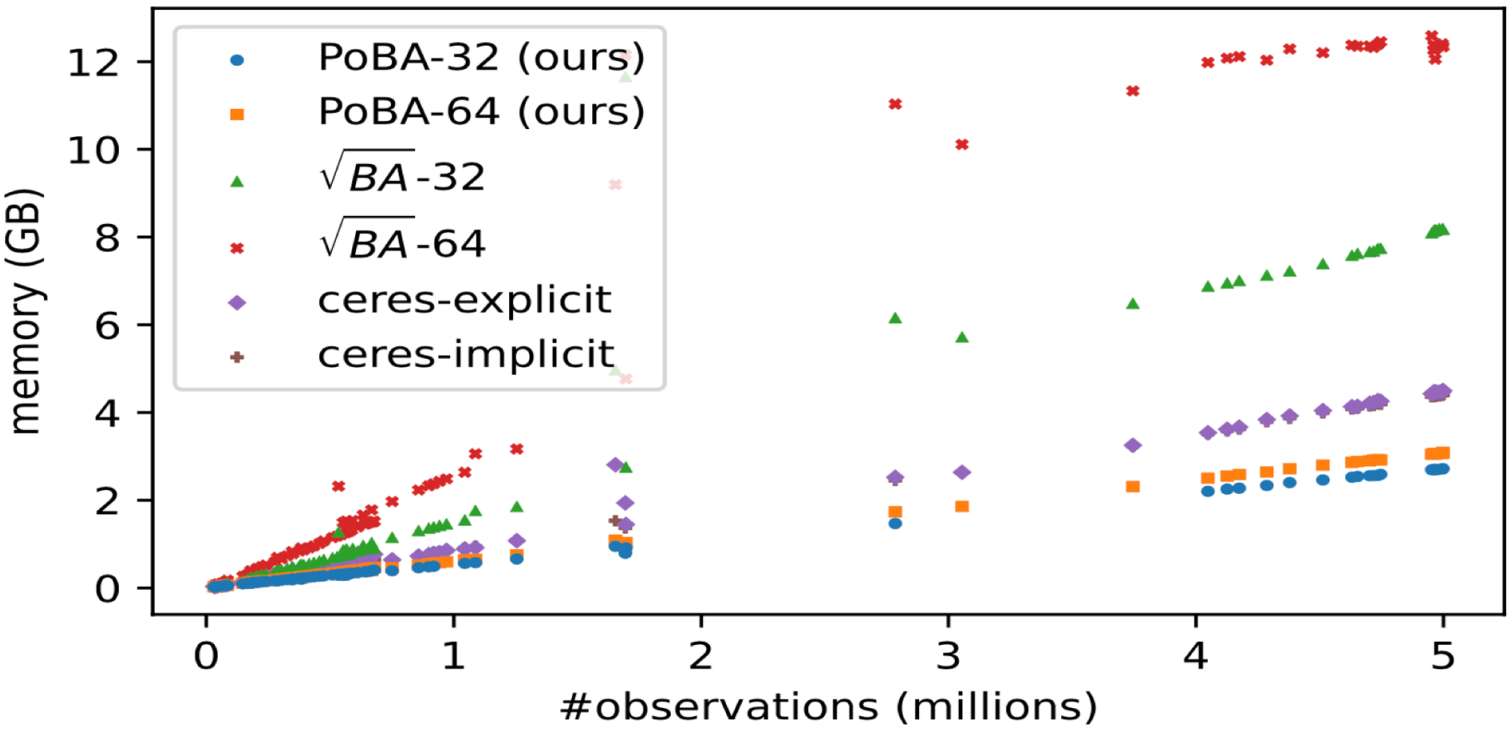
Power Bundle Adjustment for Large Scale Reconstruction



Percentage of problems solved by a certain runtime for various tolerance levels: For low and medium precision, Power BA is both faster and more accurate, for high precision ($\tau = 0.003$) it is faster, but not more accurate. For very small tolerance ($\tau = 0.001$), the baseline methods are more suitable.

Weber, Demmel, Chan, Cremers, "Power BA", CVPR 2023

Power Bundle Adjustment for Large Scale Reconstruction



Memory footprint of various bundle adjustment solvers as a function of the number of observations: Power Bundle Adjustment is up to five times more memory efficient than existing methods.

Weber, Demmel, Chan, Cremers, “Power BA”, CVPR 2023

Summary

Bundle adjustment was pioneered for over a century as a technique for structure and motion estimation in noisy real-world conditions. It aims at estimating the locations of N 3D points $\boldsymbol{X}_j$ and camera motions (R_i, T_i), given noisy 2D projections $\tilde{\boldsymbol{x}}_i^j$ in m images.

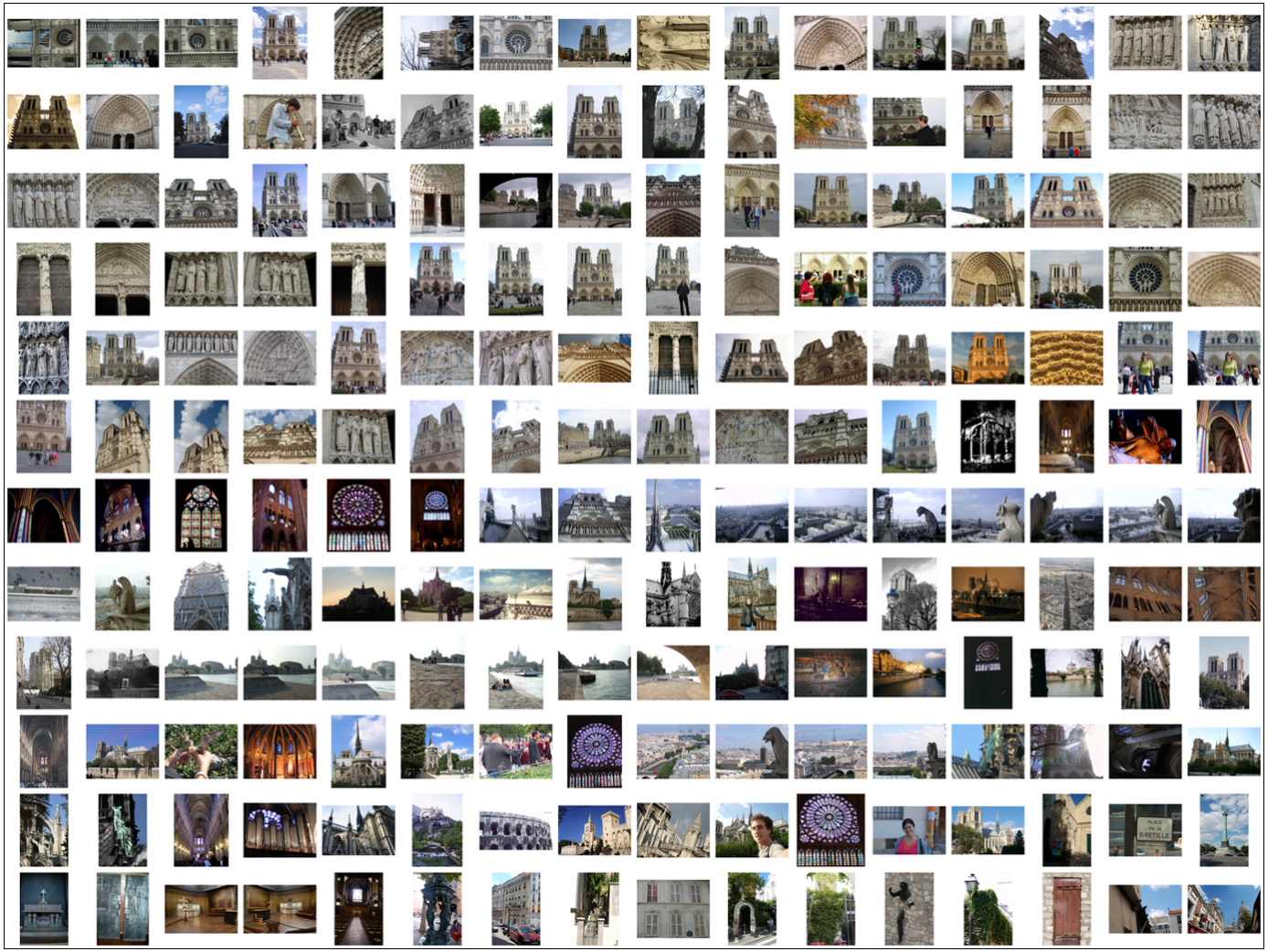
The assumption of zero-mean Gaussian noise on the 2D observations leads to the weighted nonlinear least squares problem:

$$E\left(\{R_i, T_i\}_{i=1..m}, \{\boldsymbol{X}_j\}_{j=1..N}\right) = \sum_{i=1}^{m} \sum_{j=1}^{N} \theta_{ij} \left|\tilde{\boldsymbol{x}}_i^j - \pi(R_i, T_i, \boldsymbol{X}_j)\right|^2,$$

with $\theta_{ij} = 1$ if point j is visible in image i, $\theta_{ij} = 0$ else.

Solutions of this nonconvex problem can be computed by various iterative algorithms, most importantly the Gauss-Newton algorithm or its damped version, the Levenberg-Marquardt algorithm. For reasonable tolerance levels, Power Bundle Adjustment is faster, more accurate and more memory-efficient than baseline methods.

Example I: From Internet Photo Collections...



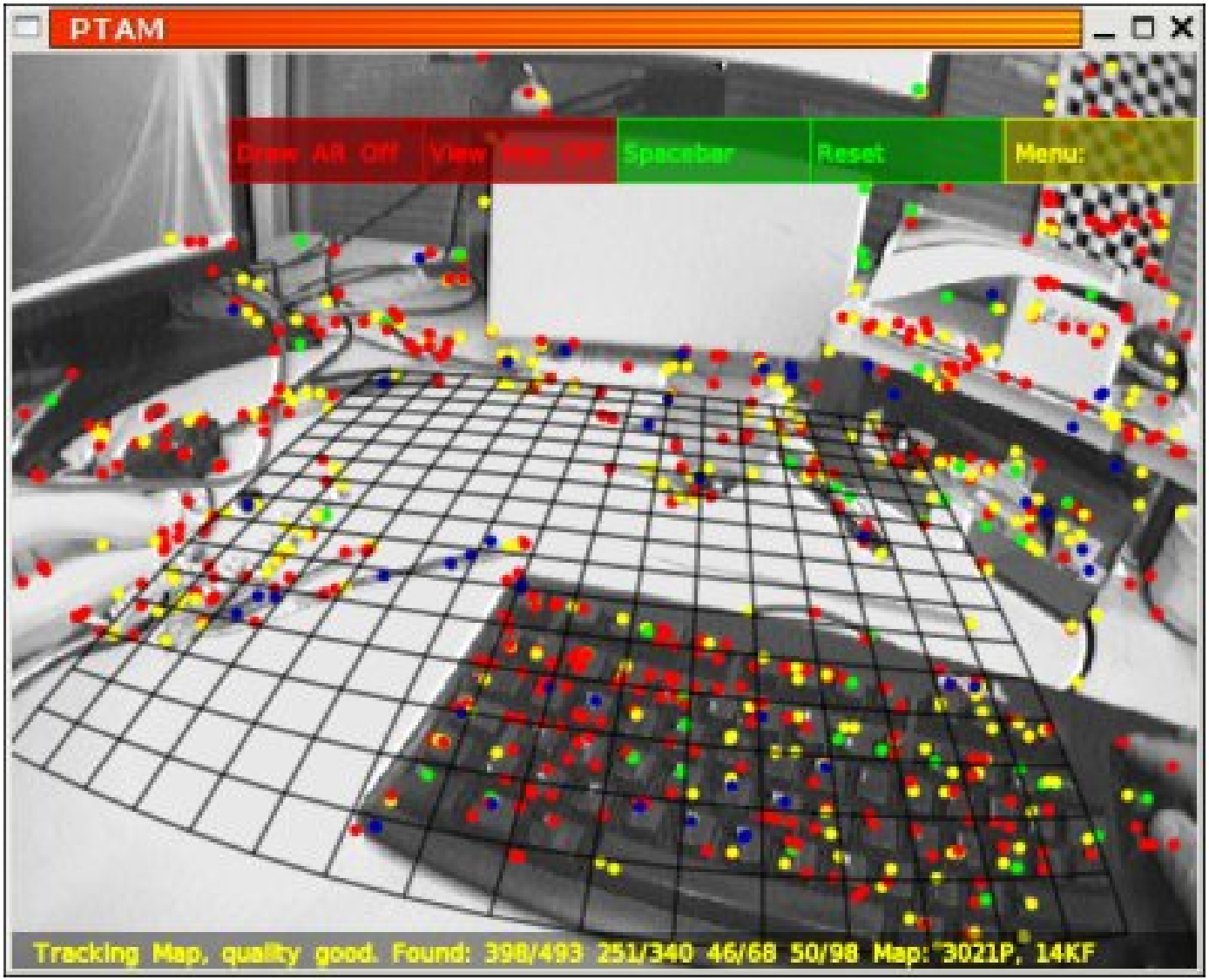
Flickr images for search term “Notre Dame”

Snavely, Seitz, Szeliski, “Modeling the world from Internet photo collections,” IJCV 2008.

...to Sparse Reconstructions

Snavely, Seitz, Szeliski, “Modeling the world from Internet photo collections,” IJCV 2008.

Example II: Realtime Structure and Motion



Klein & Murray, “Parallel Tracking and Mapping (PTAM) for Small AR Workspaces,” ISMAR 2007.