In proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Minneapolis, USA, June 18-23, 2007.

Algorithms for Batch Matrix Factorization with Application to Structure-from-Motion

Jean-Philippe Tardif* Adrien Bartoli† Martin Trudeau*
Nicolas Guilbert‡ Sébastien Roy*

Abstract

Matrix factorization is a key component for solving several computer vision problems. It is particularly challenging in the presence of missing or erroneous data, which often arise in Structure-from-Motion. We propose batch algorithms for matrix factorization. They are based on closure *and* basis *constraints, that are used either on the cameras or the structure, leading to four possible algorithms. The constraints are robustly computed from complete measurement sub-matrices with e.g. random data sampling. The cameras and 3D structure are then recovered through Linear Least Squares. Prior information about the scene such as identical camera positions or orientations, smooth camera trajectory, known 3D points and coplanarity of some 3D points can be directly incorporated. We demonstrate our algorithms on challenging image sequences with tracking error and more than 95% missing data.*

1. Introduction

Matrix factorization is an essential tool for solving several computer vision problems including Structure-from-Motion [20, 23], plane-based pose estimation [21], non rigid 3D reconstruction [6] and motion segmentation [25]. When no data are missing or corrupted by outliers, an efficient algorithm based on Singular Value Decomposition (SVD) can be used. However, missing and erroneous data are certainly unavoidable in many real-life situations, and they make factorization much more difficult. Furthermore, the SVD-based algorithm makes it difficult to enforce constraints specific to the formulation or provided by prior information about the problem.

We propose algorithms for batch matrix factorization with special attention given to the Structure-from-Motion (SfM) problem. *Closure* or *basis constraints* on one of the two factors are computed from complete measurement sub-matrices. For example, *Camera Closure Constraints* (CCC) can be computed from their left kernel [24]. We investigate three variations: *Camera Basis Constraints* (CBC), *Structure Closure Constraints* (SCC) and *Structure Basis Constraints* (SBC), that are respectively left basis, right kernel and right basis of the measurement matrix. Our experiments and comparison with other state-of-the-art batch factorization algorithms show that basis constraints usually give better results than closures, for both affine and perspective camera models. We also show that structure constraints can be used first to compute the 3D structure instead of the cameras, which allows to enforce directly constraints such as known 3D points or known planar surface. In the case of affine SBC, an approximation to the reprojection error is minimized.

Organization. The next section reviews previous work on factorization and Structure-from-Motion and points out the main differences with ours. In §3, we give our notation and formally introduce the factorization problem with his specialization to the affine Structure-from-Motion problem. Camera Closure Constraints (CCC) are first reviewed in §3.1, then follows our contribution in §4. The details are provided for the affine camera model and we discuss how the theory applies to the perspective model in §5. Robustness is discussed in §6, experiments are described and analyzed in §7, followed by conclusion in §8.

2. Previous Work

Structure-from-Motion can be formulated most generally as a bilinear (modulo homogeneous scale) inverse problem. The seminal work on affine factorization [23] however showed that it could be relaxed to a bilinear problem for the affine camera model. For perspective cameras, it can be formulated similarly when the homogeneous feature point coordinates are rescaled by their projective depth, which must be estimated *a priori* [14, 20].

The algorithms for matrix factorization despite missing data can be divided into three main categories: iterative, batch and hierarchical. In the first one, factorization is performed by minimizing directly the factorization error either

*Université de Montréal, Canada {tardifj,trudeaum,roys}@iro.umontreal.ca
†CNRS - LASMEA, France, adrien.bartoli@univ-bpclermont.fr
‡Lund University, Sweden, nicolas@maths.lth.se

by non-linear methods [7] or alternation [5, 13, 17]. However, the convergence to the global minimum is not guaranteed because they can get stuck in a local minimum. Although good performances have been reported when initializing the algorithms with a random solution [7, 12, 17], an initialization as close as possible to the global minimum is recommended. Hierarchical approaches proceed by factorizing overlapping sub-blocks of the measurement matrix [8, 16]. The solutions are then merged in a hierarchical manner, and care must be taken in choosing the merging scheme. This allows to deal with very large factorization problems.

Batch algorithms provide a solution for initializing iterative algorithms with a low computational cost. They usually minimize an approximation to the reprojection error to simplify the optimization and avoid local minima [9, 14, 24]. In the absence of noise, they find the global minimum. The approximation is done by computing the two factors through two linear steps. Constraints on one of the two factors are computed to span the whole solution space. Once the first factor is estimated, the second one can be easily computed. Non-linear and batch algorithms are usually considered complementary solutions. In [19], essential matrices are used between pairwise views to estimate the motion of all the cameras without the structure. This is similar to our solution in philosophy although a very different approach is taken.

It has been shown that the reconstruction problem is considerably simplified when observing a reference plane in all the images of the sequence [11, 22]. The structure constraints we propose handle this situation naturally. The solution must still proceed in two steps, but has the benefit that the objects **do not** need to be visible in all of the images. Finally, robustness is enforced one constraint at a time, rather than globally [1, 2, 12], thereby allowing the use of RANSAC-type algorithms.

3. Notation and Preliminaries

Notation. Matrices are in *sans-serif*, *e.g.* M, and 'joint matrices' in calligraphic characters, *e.g.* $\mathcal{M}$. Vectors are always in bold, *e.g.* $\mathbf{v}$. The matrix operator $\odot$ is the Hadamard element-wise product. Finally, $\mathbb{P}$ is the projective space.

Problem statement. The factorization of a matrix M with missing data is formulated as the problem of finding a weighted approximation of M with the closest rank r matrix (AB) such that:

$$\min_{\mathsf{A},\mathsf{B}} \|\mathsf{W}_{(n\times m)} \odot \left(\mathsf{M}_{(n\times m)} - \mathsf{A}_{(n\times r)}\mathsf{B}_{(r\times m)}\right)\|,$$

where M is called the *measurement matrix* composed of points $\mathbf{m}_p^j$, and W is a weighting matrix with zeros for missing elements in M. In some problems, constraints on elements of A and B must be enforced, *e.g.* affine SfM. In SfM, B is called the Joint Structure Matrix (JSM) and represents the 3D points $\mathbf{q}^j \in \mathbb{P}^3$, $\mathsf{A} = (\mathcal{P} \quad \mathbf{t})$ is the Joint Projection Matrix (JPM) and consists of the stacked camera projection matrices.

Affine SfM can be formulated as a rank-3 or a rank-4 factorization of a measurement matrix $\mathcal{M}$, depending on whether the input matrix has missing data or not (with the exception of [5], where predictions are made for the missing data). In the rank-3 case, the projection can be expressed with:

$$\mathbf{m}^j_{p(2\times 1)} = \mathsf{P}_{p(2\times 3)}\mathbf{q}^j_{(3\times 1)} + \mathbf{t}_{p(2\times 1)} \tag{1}$$

where an optimal choice for the *joint translation vector* $\mathbf{t}$ can be computed as the column means of $\mathcal{M}$. It can be eliminated from (1), giving the centered measurement matrix $(\mathcal{M} - \mathbf{t}\mathbf{1}^\mathsf{T})$. The factorization of this matrix computed using SVD is an optimal solution in terms of the reprojection error [23]. We have rank-4 when data are missing, so the joint translation vector cannot be computed *a priori*. Bilinear matrix factorization [17] provides a solution as long as the last row of the JSM is constrained to unity:

$$\mathcal{M}_{(2n\times m)} = \begin{pmatrix}\mathcal{P}_{(2n\times 3)} & \mathbf{t}\end{pmatrix}\begin{pmatrix}\mathcal{Q}_{(3\times m)} \\ \mathbf{1}^\mathsf{T}\end{pmatrix}. \tag{2}$$

3.1. Camera Closure Constraints (CCC)

We review the *Closure Constraints* [9, 10, 24] for affine cameras and give a generic formulation for estimating matching tensors between many views.

Deriving the constraints. Let $\hat{\mathcal{M}}$ be a sub-block of $\mathcal{M}$ without missing data. Selecting a subset of views is done by multiplying to the left by some row-amputated block-diagonal matrix Π with (2×2) identity blocks. Selecting a subset of features is done similarly by multiplying to the right by Γ, an identity matrix amputated of some of its columns, yielding:

$$\hat{\mathcal{M}}_{(2\hat{n}\times\hat{m})} \stackrel{\text{def}}{=} \Pi\mathcal{M}\Gamma.$$

In this case, the measurements can be expressed as:

$$\hat{\mathcal{M}} = \Pi\mathcal{M}\Gamma = \Pi\mathcal{P}\mathcal{Q}\Gamma + \Pi\mathbf{t}\mathbf{1}^\mathsf{T}\Gamma = \hat{\mathcal{P}}\hat{\mathcal{Q}} + \hat{\mathbf{t}}\mathbf{1}^\mathsf{T}. \tag{3}$$

We define $\boldsymbol{\mu}_m$ as:

$$\boldsymbol{\mu}_m \stackrel{\text{def}}{=} \frac{1}{m}\mathbf{1}_{(m\times 1)},$$

which computes the column means of an $(n \times m)$ matrix by multiplying to the right. We define the centered measurement matrix as:

$$\bar{\mathcal{M}} \stackrel{\text{def}}{=} \hat{\mathcal{M}} - \hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}\mathbf{1}^\mathsf{T}_{(1\times\hat{m})}, \tag{4}$$

which does not equal $(\hat{\mathcal{P}}\hat{\mathcal{Q}})$ in general, as the row means of the sub-blocks are not necessarily those of the complete

matrix. The SVD $\bar{\mathcal{M}} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$ can be used to compute optimal rank-3 factors given by the leading 3 columns of U and 3 rows of $\Sigma\mathsf{V}^\mathsf{T}$. The first factor gives the a solution for the partial Joint Projection Matrix while the remaining columns of U form a basis for the best approximation to the left kernel of $\bar{\mathcal{M}}$, which we call *centered matching tensor*, denoted $\bar{\mathcal{N}}$. We have:

$$\bar{\mathcal{N}}^\mathsf{T}\bar{\mathcal{M}} = \mathbf{0},$$

and from (4):

$$\bar{\mathcal{N}}^\mathsf{T}\begin{pmatrix}\hat{\mathcal{M}} & -\hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}\mathbf{1}^\mathsf{T}\end{pmatrix} = \mathbf{0},$$

that rewrites as:

$$\underbrace{\begin{pmatrix}\bar{\mathcal{N}}^\mathsf{T} & -\bar{\mathcal{N}}^\mathsf{T}\hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}\end{pmatrix}}_{\mathcal{N}^\mathsf{T}}\begin{pmatrix}\hat{\mathcal{M}} \\ \mathbf{1}^\mathsf{T}\end{pmatrix} = \mathbf{0}, \tag{5}$$

where appears the non-centered matching tensor $\mathcal{N}$ we are seeking. Note that directly computing a tensor from $\left(\hat{\mathcal{M}}^\mathsf{T}\ \mathbf{1}\right)^\mathsf{T}$ would not be optimal in terms of reprojection error. Tensor $\mathcal{N}$ corresponds to the classical affine matching tensor. The affine fundamental matrix has 4 degrees of freedom, and the non-centered left kernel obtained from a measurement matrix with four rows has 5 components up to scale. A matching tensor computed from a matrix with six rows is of size (3×6) and has orthonormal rows, which leaves 12 degrees of freedom like the affine trifocal tensor.

Estimating the Joint Projection Matrix. The unity constraint on the last row of the Joint Structure Matrix can be expressed with extra rows in $\mathcal{M}$ and $\mathcal{P}$:

$$\begin{pmatrix}\mathcal{M} \\ \mathbf{1}^\mathsf{T}\end{pmatrix} = \begin{pmatrix}\mathcal{P} & \mathbf{t} \\ \mathbf{0}_{(1\times 3)} & 1\end{pmatrix}\begin{pmatrix}\mathcal{Q} \\ \mathbf{1}^\mathsf{T}\end{pmatrix}. \tag{6}$$

Let:

$$\mathsf{D} \stackrel{\text{def}}{=} \Pi^\mathsf{T}\bar{\mathcal{N}};$$

multiplying (6) by $\begin{pmatrix}\Pi & \mathbf{0} \\ \mathbf{0} & 1\end{pmatrix}$ to the left and Γ to the right and substituting in (5), we obtain:

$$\begin{pmatrix}\mathsf{D}^\mathsf{T} & -\bar{\mathcal{N}}^\mathsf{T}\hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}\end{pmatrix}\begin{pmatrix}\mathcal{P} & \mathbf{t} \\ \mathbf{0} & 1\end{pmatrix} = \mathbf{0},$$

$$\boxed{\mathsf{D}^\mathsf{T}\begin{pmatrix}\mathcal{P} & \mathbf{t}\end{pmatrix} = \begin{pmatrix}\mathbf{0}_{(1\times 3)} & \bar{\mathcal{N}}^\mathsf{T}\hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}\end{pmatrix}.}$$

Stacking every such constraint computed from different sub-blocks of $\mathcal{M}$ in a single matrix equation, we get:

$$\begin{pmatrix}\mathsf{D}_1^\mathsf{T} \\ \vdots \\ \mathsf{D}_l^\mathsf{T}\end{pmatrix}\begin{pmatrix}\mathcal{P} & \mathbf{t}\end{pmatrix} = \begin{pmatrix}\mathbf{0} & \bar{\mathcal{N}}_1^\mathsf{T}\hat{\mathcal{M}}_1\boldsymbol{\mu}_{\hat{m}} \\ \vdots & \vdots \\ \mathbf{0} & \bar{\mathcal{N}}_l^\mathsf{T}\hat{\mathcal{M}}_l\boldsymbol{\mu}_{\hat{m}}\end{pmatrix}.$$

The design matrix, denoted $\mathcal{D} \stackrel{\text{def}}{=} (\mathsf{D}_1, \ldots, \mathsf{D}_l)^\mathsf{T}$, is highly sparse. This is exploited when computing the Linear Least Square (LLS) solution. As explained in [9], choosing the projection matrix of some of the cameras fixes the gauge of the system and ensures a full-rank design matrix. The error minimized by this method is difficult to interpret because it is expressed in terms of matching tensors. Once the JPM is estimated, the structure can be computed by affine triangulation.

4. Batch Matrix Factorization for Affine SfM and Generalization of CCC

4.1. Summary of the Algorithms

The algorithms we propose follow the typical steps of batch algorithms: 1) Measurement sub-matrices without missing data are found. 2) Each of these matrices is used to compute a constraint, either on the partial Joint Projection Matrix or partial Joint Structure Matrix. 3) The constraints are combined to estimate one of the factors. 4) The second factor is estimated by camera resectioning or triangulation. 5) Finally, non-linear or alternation methods refine the solution.

4.2. Camera Basis Constraints (CBC)

We show how basis constraints can be used instead of matching tensors. The partial JPM $\mathcal{P}$ can be computed alone, *i.e.* without the joint translation vector. This is done by aligning bases of the projection matrices of partial reconstructions performed on measurement sub-matrices. Once $\mathcal{P}$ is recovered, the joint translation vector and the structure can be computed together to minimize the reprojection error.

Consider a centered sub-matrix $\bar{\mathcal{M}}$ of $\mathcal{M}$ and compute its SVD $\bar{\mathcal{M}} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$. The 3 leading columns of U, denoted $\bar{\mathsf{U}}$, form a basis of $\hat{\mathcal{P}}$, that is, there is a 3×3 invertible matrix Z (the aligning transformation) such that:

$$\hat{\mathcal{P}} = \bar{\mathsf{U}}\mathsf{Z}, \tag{7}$$

leading to the Camera Basis Constraint:

$$\boxed{\Pi\mathcal{P} = \bar{\mathsf{U}}\mathsf{Z}.} \tag{8}$$

Because of the remark following (4), this is not trivial. To demonstrate this, we note that multiplying an $(n \times m)$ matrix by:

$$\boldsymbol{\vartheta}_m \stackrel{\text{def}}{=} \mathrm{I} - \frac{1}{m}1_{(m\times m)} = \mathrm{I} - \boldsymbol{\mu}_m\mathbf{1}_{(1\times m)},$$

to the right subtracts the row mean from each of its entries. Consequently:

$$\bar{\mathcal{M}} = \Pi\mathcal{M}\Gamma\boldsymbol{\vartheta}_{\hat{m}} = \Pi\mathcal{P}\mathcal{Q}\Gamma\boldsymbol{\vartheta}_{\hat{m}} + \Pi\mathbf{t}\mathbf{1}^\mathsf{T}\Gamma\boldsymbol{\vartheta}_{\hat{m}} = \Pi\mathcal{P}\mathcal{Q}\Gamma\boldsymbol{\vartheta}_{\hat{m}},$$

since $\Pi \mathbf{t} \mathbf{1}^\mathsf{T} \Gamma \boldsymbol{\vartheta}_{\hat{m}} = 0$. Hence, the columns of $\bar{\mathcal{M}}$ are linear combinations of those of $\Pi\mathcal{P}$, and since $\bar{\mathcal{M}}$ has rank 3, the same is true of $\bar{\mathsf{U}}$.

4.2.1 Solving for the Joint Projection Matrix

Computing many CBC's for different sub-blocks of the measurement matrix amounts to performing partial affine reconstructions. Solving for the Joint Projection Matrix is done by: 1) aligning basis constraints to recover the reduced JPM and 2) recovering the joint translation vector and the structure.

Let l be the number of CBC's. Aligning bases together involves minimizing:

$$\sum_{k=1}^{l} \|\bar{\mathsf{U}}_k \mathsf{Z}_k - \Pi_k \mathcal{P}\|^2 = \sum_{k=1}^{l} \left\| \begin{pmatrix} \Pi_k & -\bar{\mathsf{U}}_k \end{pmatrix} \begin{pmatrix} \mathcal{P} \\ \mathsf{Z}_k \end{pmatrix} \right\|^2,$$

which is a simple LLS system, that can be rewritten as:

$$\left\| \underbrace{\begin{pmatrix} \Pi_1 & -\bar{\mathsf{U}}_1 & & \mathbf{0} \\ \vdots & & \ddots & \\ \Pi_l & \mathbf{0} & & -\bar{\mathsf{U}}_l \end{pmatrix}}_{\mathcal{D}} \underbrace{\begin{pmatrix} \mathcal{P} \\ \mathsf{Z}_1 \\ \vdots \\ \mathsf{Z}_l \end{pmatrix}}_{\mathsf{X}} \right\|^2. \tag{9}$$

Although the size of the design matrix $\mathcal{D}$ is quadratic in the number of bases, the equation system can be minimized efficiently. Indeed it is extremely sparse, and we do not need to compute the aligning matrices Z_k. The minimized error is the alignment of the optimal cameras of partial reconstructions. Although this error is algebraic, as when expressing the error in terms of matching tensor constraints, our experiments suggest it is more stable, *c.f.* §7.

Once the partial JPM $\mathcal{P}$ is estimated, we propose two approaches for estimating the translations and the structure. The best solution comes from doing both at the same time, by using an LLS formulation. It minimizes the reprojection error. This is because the orientation and intrinsics of the camera are already estimated up to a (3×3) invertible transformation G, which has no effect on the minimized error:

$$\|\mathcal{M} - \mathcal{P}\mathcal{Q} - \mathbf{1}^\mathsf{T}\mathbf{t}\| = \|\mathcal{M} - \mathcal{P}\mathsf{G}\mathsf{G}^{-1}\mathcal{Q} - \mathbf{1}^\mathsf{T}\mathbf{t}\|. \tag{10}$$

However, for a long sequence, this equation system uses a lot of memory and is rather long to minimize. In this case, it is more efficient to estimate only the joint translation vector and then perform individual triangulation for each feature track. To this end, we combine the computed basis $\bar{\mathsf{U}}$ with the translation $\bar{\mathbf{t}} = \hat{\mathcal{M}}\boldsymbol{\mu}_{\hat{m}}$ of the cameras of the partial reconstructions. Thus, the camera alignment can also be performed with:

$$\Pi \begin{pmatrix} \mathcal{P} & \mathbf{t} \end{pmatrix} = \begin{pmatrix} \hat{\mathcal{P}} & \hat{\mathbf{t}} \end{pmatrix} = \begin{pmatrix} \bar{\mathsf{U}} & \bar{\mathbf{t}} \end{pmatrix} \begin{pmatrix} \mathsf{Z} & \mathbf{v} \\ \mathbf{0}_{(1\times 3)} & 1 \end{pmatrix},$$

where matrices $\hat{\mathsf{U}}$ and Z are those from (8). The joint translation vector can be estimated by minimizing:

$$\left\| \mathcal{D} \begin{pmatrix} \mathbf{t} \\ \mathbf{v}_1 \\ \vdots \\ \mathbf{v}_l \end{pmatrix} - \begin{pmatrix} \bar{\mathbf{t}}_1 \\ \vdots \\ \bar{\mathbf{t}}_l \end{pmatrix} \right\|^2, \tag{11}$$

where $\mathcal{D}$ is the design matrix of (9).

In [14], bases for cameras were computed from $\hat{\mathcal{M}}$, not $\bar{\mathcal{M}}$ like we do. The partial reconstruction corresponding to these cameras is not optimal. Furthermore, a method for estimating $\hat{\mathbf{t}}$ from the bases independently of the structure is not given. This is essential when dealing with very large sequences or data corrupted by outliers (*c.f.* §6).

4.3. Structure Closure Constraints (SCC)

Structure Closure Constraint is the analogue of the CCC applied to the Joint Structure Matrix. From (2), a matching tensor is in the column space of $\mathcal{Q}$ and $\mathbf{1}^\mathsf{T}$. It can be computed using SVD by minimizing:

$$\boxed{\|\hat{\mathcal{M}}\mathcal{N}\|^2, \text{ subject to } \mathbf{1}^\mathsf{T}\mathcal{N} = 0.}$$

Note that unlike the CCC, the SCC is not computed from a centered input matrix. This is because the last row of the JSM must be $\mathbf{1}^\mathsf{T}$. A more formal explanation is given in §4.4. By accumulating many constraints, we can form a design matrix $\mathcal{D}$ with:

$$\mathsf{D}_i \stackrel{\text{def}}{=} \Gamma\mathcal{N}_i.$$

By construction $\mathcal{D}$ is rank deficient because each of its rows vanishes on $\mathbf{1}^\mathsf{T}$. Hence, the right singular vector corresponding to the smallest singular value, equal to zero, is $\mathbf{1}/\|\mathbf{1}\|$. The estimate for $\mathcal{Q}^\mathsf{T}$ is given by the next three right singular vectors of $\mathcal{D}$.

4.4. Structure Basis Constraints (SBC)

As with Camera Closures, using Structure Closures implies minimizing a purely algebraic function difficult to interpret. Structure Bases can be used to estimate partial reconstructions which are then aligned together in a single computation. From an SVD of $\bar{\mathcal{M}} = \mathsf{U}\Sigma\mathsf{V}^\mathsf{T}$, the three leading columns of V estimate the structure, up to an affine transformation, in the partial reconstruction corresponding to $\hat{\mathcal{M}}$. However we prefer using $\bar{\mathsf{V}}$ as the three leading columns of $\mathsf{V}\Sigma^\mathsf{T}$, as explained below. It can be aligned with the structure through:

$$\hat{\mathcal{Q}}^\mathsf{T} = \mathsf{Z}_{(3\times 4)} \begin{pmatrix} \bar{\mathsf{V}} & \mathbf{1} \end{pmatrix}^\mathsf{T},$$

leading to the Structure Basis Constraint:

$$\boxed{\Gamma^\mathsf{T}\mathcal{Q}^\mathsf{T} = \mathsf{Z}\bar{\mathsf{V}}^\mathsf{T}.}$$

Note that unlike a CBC, an SBC cannot be aligned with a (3×3) matrix. This is because the row space of $\bar{\mathsf{V}}$ is that of:

$$\Pi\mathcal{P}\mathcal{Q}\Gamma\boldsymbol{\vartheta}_{\hat{m}} = \Pi\mathcal{P}\begin{pmatrix}\hat{\mathsf{Q}}^1\boldsymbol{\vartheta}_{\hat{m}}\\ \hat{\mathsf{Q}}^2\boldsymbol{\vartheta}_{\hat{m}}\\ \hat{\mathsf{Q}}^3\boldsymbol{\vartheta}_{\hat{m}}\\ \mathbf{1}^\mathsf{T}\boldsymbol{\vartheta}_{\hat{m}}\end{pmatrix} = \Pi\mathcal{P}\begin{pmatrix}\hat{\mathsf{Q}}^1 - \hat{\mathsf{Q}}^1\boldsymbol{\mu}_{\hat{m}}\mathbf{1}_{(1\times m)}\\ \hat{\mathsf{Q}}^2 - \hat{\mathsf{Q}}^2\boldsymbol{\mu}_{\hat{m}}\mathbf{1}_{(1\times m)}\\ \hat{\mathsf{Q}}^3 - \hat{\mathsf{Q}}^3\boldsymbol{\mu}_{\hat{m}}\mathbf{1}_{(1\times m)}\\ \mathbf{0}^\mathsf{T}\end{pmatrix},$$

where the $\hat{\mathsf{Q}}^i$'s are the rows of $\hat{\mathsf{Q}}$, that is, the row space of $\bar{\mathsf{V}}$ is the one generated by the $\hat{\mathsf{Q}}^i$'s and $\mathbf{1}^\mathsf{T}$, but not necessarily only by the $\hat{\mathsf{Q}}^i$'s. Partial reconstructions can be aligned together by solving an equation system similar to (9).

Choosing the bases. Consider two partial reconstructions $\bar{\mathsf{V}}$ and $\bar{\mathsf{V}}'$. Aligning them together amounts to finding:

$$\arg\min_{\mathsf{Z}} \|\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}}\|^2. \tag{12}$$

We show that when $\bar{\mathsf{V}}$ is chosen so that the corresponding projection matrix $\hat{\mathcal{P}}$ is orthonormal, the 3D error approximates the reprojection error [3]. Projecting the residual given by (12) into the images corresponding to the block, we obtain:

$$\begin{aligned}\|\hat{\mathcal{P}}\bar{\mathsf{V}}' + \mathbf{1}^\mathsf{T}\hat{\mathbf{t}} - (\hat{\mathcal{P}}\mathsf{Z}\bar{\mathsf{V}} + \mathbf{1}^\mathsf{T}\hat{\mathbf{t}})\|^2 &= \|\hat{\mathcal{P}}\bar{\mathsf{V}}' - \hat{\mathcal{P}}\mathsf{Z}\bar{\mathsf{V}}\|^2\\ &= \|\hat{\mathcal{P}}(\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}})\|^2.\end{aligned}$$

Thanks to the orthonormal property $\hat{\mathcal{P}}^\mathsf{T} = \hat{\mathcal{P}}^\dagger$, our error function simplifies to:

$$\mathrm{tr}\left((\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}})^\mathsf{T}\underbrace{\hat{\mathcal{P}}^\mathsf{T}\hat{\mathcal{P}}}_{\mathrm{I}_{(3\times 3)}}(\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}})\right) = \|\bar{\mathsf{V}}' - \mathsf{Z}\bar{\mathsf{V}}\|^2.$$

The minimization is only exact if both $\bar{\mathsf{V}}$ and $\bar{\mathsf{V}}'$ have been estimated without error in their respective partial reconstruction. Hence, under noise, it only approximates the reprojection error.

4.5. Enforcing Constraints

In many situations, prior knowledge about the configuration of the scene structure is available. Examples are two cameras with identical position and/or orientation, smooth camera path, known 3D points and planar structure. In our algorithms, a certain number of constraints can be enforced. This is done when estimating the first factor. As a consequence, one can only force constraints on either cameras or structure.

Most of our equation systems are homogeneous, like (9). In order to simplify the constrained optimization, a well known trick is to select a gauge, *i.e.* to give a value to some rows of X, and solve the resulting regression problem. A valid gauge fixes the degrees of freedom of the affine ambiguity, which makes the original design matrices rank deficient. Once this is done, each column $\mathbf{X}^i$ of the solution matrix X can be individually estimated by regression under linear constraints, which is an instance of convex quadratic programming [4]. Aligning two cameras together can be done by choosing:

$$\begin{aligned}\begin{pmatrix}\mathsf{P}_a & \mathbf{t}_a\end{pmatrix} = \begin{pmatrix}\mathsf{P}_b & \mathbf{t}_b\end{pmatrix} &= \begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\end{pmatrix}\\ \begin{pmatrix}\mathsf{P}_c & \mathbf{t}_c\end{pmatrix} &= \begin{pmatrix}0 & 1 & 1 & 0\\ * & * & * & *\end{pmatrix}\end{aligned}$$

Others can also be aligned through equality constraints $\mathsf{P}_e - \mathsf{P}_f = 0$, or variable elimination.

Enforcing known 3D points can be done similarly with structure constraints. The gauge is fixed with at least four non-coplanar points. Three planar surfaces can also be enforced by forcing groups of points to have their (X, Y, Z) coordinates to either $(0, *, *)$, $(*, 0, *)$ or $(*, *, 0)$, where $*$ means the coordinate is not fixed[1]. For more than three planes, their absolute equation has to be known. Observe that the gauge is at least partially fixed by the points located on the planes. Consequently, caution must be taken to ensure that these planes are really orthogonal up to an affine transformation. Prior information from points and planar structures can also be incorporated as long as the constraints are compatible, *i.e.* the affine ambiguity is fixed by the same matrix.

5. The Perspective Camera Model

In projective SfM, the point coordinates are multiplied by their projective depth λ_p^j and the projection is performed by (3×4) matrices defined up to scale:

$$\lambda_p^j\begin{pmatrix}\mathbf{m}_p^j\\ 1\end{pmatrix} = \begin{pmatrix}\mathsf{P}_{p(3\times 3)} & \mathbf{t}_{p(3\times 1)}\end{pmatrix}\mathbf{q}_{(4\times 1)}^j$$

where $\mathbf{m}_p^j \in \mathbb{R}^2$ is an interest point tracked throughout the sequence and $\mathbf{q}^j \in \mathbb{P}^3$. We assume that the projective depths are estimated *a priori*. In our experiments, we used the algorithm presented in [14]. Rank-4 factorization is then performed without any restriction on $\mathbf{q}^j$, unlike for the affine case. Closures and rank-4 bases are purely algebraic and are computed from $\hat{\mathcal{M}}_{(3\hat{n}\times\hat{m})} = \Pi\mathcal{M}\Gamma$ instead of $\bar{\mathcal{M}}$. This is akin to what was presented in [14]. The main difference is the way system (9) is minimized. They used Matlab's EIGS method to estimate four orthonormal solution vectors. We used a solution based on fixing one of the bases $\bar{\mathsf{U}}_i$ to identity and solve the resulting sparse regression problem. Finally, we performed a QR factorization on the estimated JPM to orthonormalize its columns. It was suggested in [14] that gluing via points cannot be used with the perspective camera model. We did not encounter this limitation.

[1] Details will be provided in a technical report.

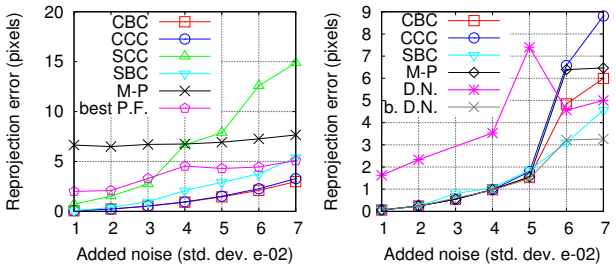

Figure 1. Comparison of the algorithms for the simulated sequence. M-P stands for Martinec-Pajdla [14], P.F. for Powerfactorization [17], D.N. for Damped Newton [7] and b. for the best solution out of the 15 trials. **Left)** Affine model. **Right)** Perspective model (SCC not shown).

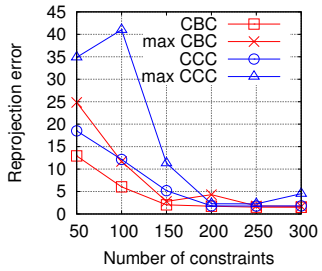

Figure 2. Comparison for the number of constraints between CCC and CBC, for our simulated sequence.

6. Robustness

To deal with outliers, we use random sampling to compute each Closure/Basis Constraint. Once the camera path or 3D points have been computed, we rely again on random sampling to perform robust triangulation or resection. We do not reject points directly from the computed matching tensors, which can leave certain outliers behind. We avoid performing the optimization using a robust (non-convex) cost function, that would typically have a lot of local minima, or using alternation with re-weighting.

For CBC, we rely on (11) to compute the position of the cameras, because the estimation of the structure and the translation vectors in a single step (*i.e.* using (10)) would be computationally expensive in a random sampling strategy.

7. Experiments and Analysis

We compared our three algorithms to Guilbert *et al.* [9], Martinec-Pajdla [14], Hartley-Schaffalitzky [17] and Buchanan-Fitzgibbon [7] methods on simulated and real sequences. Powerfactorization and Damped Newton were used respectively for the affine and perspective camera model. They were limited to 1000 iterations, which was sufficient to attain convergence from a random solution in most cases. The tracks were sorted in order of appearance in the sequence and we assumed that the resulting measurement matrix was approximately band-diagonal as in figure 4. Heuristics were used to find complete sub-blocks, making sure that constraints are approximately equally distributed among the cameras or the 3D points, depending on the constraint type.

Simulation. The simulated sequence consisted of 50 cameras and around 900 3D points with a lot of occlusion, resulting in around 96% missing data. In figure 1, the algorithms are compared for both camera models at different levels of Gaussian noise. Our results strongly suggest that Camera Basis performs best, especially under affine projection. Under perspective projection, Closure and Structure Basis gave similar results with a slight advantage to SBC in the presence of very high noise. The advantage vanished when projective depths were exact (not shown here). Hence, SBC seems to be the most robust to erroneous projective depths. The CBC method performed slightly better than Martinec-Padjla's, a likely result of our balancing of the constraints. Structure Closures performed rather poorly. This is not surprising, at least for the affine case, since it is the only one of our algorithms whose constraint is not optimal. Powerfactorization and Damped Newton provided good performance in the best case, but on average, they did not converge to satisfying solutions.

Most of the computation time was spent finding complete sub-blocks from the measurement matrix. The time for solving the two factors was almost negligible. Hence, a good algorithm performs well even with a small number of constraints. We compared CCC and CBC on this criteria with the simulated sequence as well as with the *Teddy Bear* sequence (*c.f.* figure 2 and 3(a)). The number of closures had to be nearly twice as large as that of bases to obtain a comparable average reprojection error. The maximal reprojection error also suggests more stability for bases.

Real sequences. We compared batch algorithms and Powerfactorization on five real sequences, four affine and one perspective, (*c.f.* table 1). To take them out of the comparison, we removed the outliers in each of the sequence using the robust CBC-based algorithm (see test below). For the *Dinosaur* sequence, the algorithm of Guilbert *et al.* and Martinec-Pajdla obtained a better mean reprojection error than what they reported in their paper, respectively 5.4 and 2.57 pixels. This is probably because we used more constraints (around 350). Computation time on an AMD Athlon 64 3500+, in Matlab, for the camera estimation (not including sub-block search) were 0.01 and 0.29 seconds for CCC and CBC (for which five different gauges were tested) and 0.34 seconds for Martinec-Pajdla. On the larger *Teddy bear* sequence, computation time were respectively 0.49, 0.73, 0.91 seconds (not including the translations since it was much longer minimizing (10) than (11)). Structure Constraint based algorithms were slower because their equation systems are larger. Iterative

Sequence (# Img., # 3D pts, # 2D pts, miss. data)	Mean (max) reprojection error in pixels					
	CCC [9]	CBC	SCC	SBC [3]	P.F. [17]	M-P [14]
Dinosaur (36,2683,11832,96.9%)	0.56 (5.49)	**0.49 (4.62)**	0.65 (7.27)	0.66 (7.12)	1.75 (73.1)	0.56 (6.99)
Book (95, 254, 10253, 89%)	0.54 (6.86)	**0.50 (5.59)**	0.55 (5.25)	0.56 (5.66)	0.54 (5.96)	2.56 (41.1)
Building (194, 779, 17233, 97%)	**0.86 (14.4)**	0.95 (22.8)	1.24 (17.5)	1.21 (21.5)	(3.45) 256.8	1.28 (39.2)
Teddy Bear (196, 2480, 93589, 95%)	**0.65 (8.14)**	**0.65 (8.14)**	4.67 (174.5)	1.13 (35.3)	1.91 (38.8)	4.453 (96.97)
Desk (66, 2483, 26771, 95.9%)	0.99 (43.36)	**0.87 (19.5)**	3.93 (132.66)	1.44 (45.18)	—-	**0.83 (24.4)**

Table 1. Comparison between batch methods and Powerfactorization for four real sequences. All were reconstructed using the affine camera model except for the *Desk* sequence. The best solutions are in bold.

algorithms achieved convergence after a few hundred iterations, resulting in minutes, if not hours, of computation. When initialized using a batch method, only a few iterations were necessary.

The *Desk* sequence (*c.f.* figure 4, 5 and 6(c)) was reconstructed using the perspective algorithms and the one based on affine CBC. In the former case, outliers were removed before factorization using fundamental matrices. All batch algorithms but the one based on SCC provided satisfying results. The reconstruction shown in figure 6(c) is the result of refining the solution with Mahamud *et al.* method followed by self-calibration. Projective and Euclidean bundle adjustment improved the reconstruction only slightly. We also achieved reconstruction by initializing a Euclidean bundle adjustment with the robust algorithm based on affine CBC. After convergence, the recovered focal length was similar to the one recovered using projective reconstruction.

Outliers issue. The performance of CCC and CBC for handling outliers was also tested similarly to [18]. Up to 7 % outliers where added to the *Dinosaur* sequence. This gives up to $7n$ % unusable n-view constraints (we used up to 4). The constraints were computed from robustly selected sub-blocks. Thus, as the number of outliers increased, the number of points used to compute the constraints decreased. We tested: 1) the percentage of recovered valid outliers, 2) the percentage of removed points that were in fact inliers (false positive) and 3) the average reprojection error of the reconstruction using the original data set. Our results are shown in figure 3(b) and 3(c).

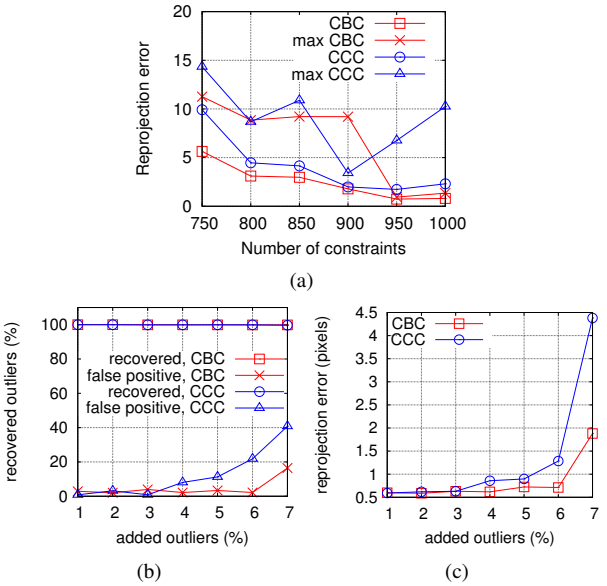

Figure 3. Comparison between CCC and CBC with real data. **a)** number of constraints for the *Teddy Bear* sequence. For added outliers in the *Dinosaur* sequence: **b)** percentage of outliers recovered and percentage of false positive, **c)** average reprojection error of the original data.

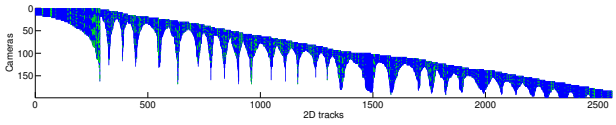

Figure 4. Tracks (blue/dark) and detected outliers (green/light) from the *Desk* sequence.

8. Conclusion

We presented algorithms for efficient batch matrix factorization. Constraints on measurement sub-matrices are combined to estimate one of the two factors. We extended Trigg's Camera Closure Constraints to Structure Closure Constraints and proposed Camera and Structure Basis. Experimental results showed that Basis Constraints fared better than state-of-the-art methods on most of our tests, with simulated and real sequences and both for affine and perspective camera models. Future work will focus on the measurement sub-matrix search and selection mechanism, and on experiments for enforcing *a priori* on the structure.

Acknowledgments. To A. Buchanan and D. Martinec for providing their source code, to A. Fitzgibbon and A. Zisserman for the Dinosaur sequence and to K. McHenry and G. Petit for the Teddy bear sequence.

References

[1] H. Aanaes, R. Fisker, K. Astrom, J. M. Carstensen, Robust Factorization. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 9, pp. 1215-1225, Sept., 2002.

Figure 5. Five out of 66 images of the *Desk* sequence. Reconstruction shown in 6(c).

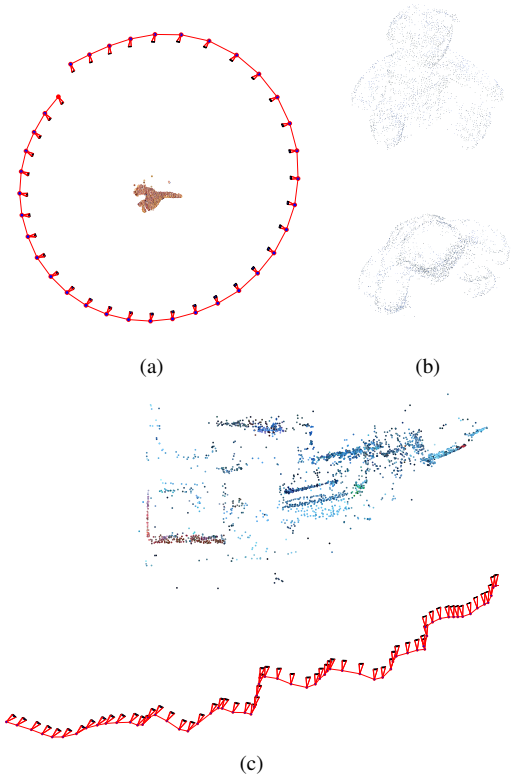

Figure 6. Reconstruction results. **a)** Top view of the *Dinosaur* sequence without using a closing constraint. **b)** Side and top view of the *Teddy Bear* model. **c)** Top view of the *Desk* sequence. All but the *Desk* sequence were obtained solely with an algorithm based on CBC.

[2] P. Anandan and M. Irani, Factorization with Uncertainty. International Journal of Computer Vision (IJCV), 49(2-3): 101-116, September 2002.

[3] A. Bartoli, H. Martinsson, F. Gaspard, J.-M. Lavest, On Aligning Sets of Points Reconstructed from Uncalibrated Affine Cameras. SCIA 2005.

[4] S. Boyd and L. Vandenberghe, Convex Optimization. Cambridge University Press, 2004.

[5] S. Brandt. Closed-form solutions for affine reconstruction under missing data. ECCV 2002.

[6] C. Bregler, A. Hertzmann, and H. Biermann, Recovering non-rigid 3D shape from image streams. CVPR 2000.

[7] A. M. Buchanan, A. W. Fitzgibbon, Damped Newton Algorithms for Matrix Factorization with Missing Data. CVPR 2005.

[8] A.W. Fitzgibbon, A. Zisserman, Automatic Camera Recovery for Closed or Open Image Sequences. ECCV 1998

[9] N. Guilbert, A. Bartoli and A. Heyden, Affine Approximation for Direct Batch Recovery of Euclidean Motion From Sparse Data. International Journal of Computer Vision, Vol. 69, No. 3, pp. 317-333, September 2006.

[10] F. Kahl, A. Heyden, Affine Structure and Motion from Points, Lines and Conics. International Journal of Computer Vision, 33(3):163-180, 1999.

[11] R. Kaucic, R. Hartley, and N. Dano, Plane-based Projective Reconstruction. ICCV 2001.

[12] Q. Ke, T. Kanade, Robust L-1 Norm Factorization in the Presence of Outliers and Missing Data by Alternative Convex Programming. CVPR 2005.

[13] S. Mahamud, M. Hebert, Y. Omori, J. Ponce, Provably-Convergent Iterative Methods for Projective Structure from Motion. CVPR 2001.

[14] D. Martinec and T. Pajdla, 3D Reconstruction by Fitting Low-rank Matrices with Missing Data. CVPR 2005.

[15] D. Martinec and T. Pajdla, 3D Reconstruction by Gluing Pair-wise Euclidean Reconstructions, or "How to Achieve a Good Reconstruction from Bad Images". 3DPVT 2006.

[16] D. Nistèr, Reconstruction from uncalibrated sequences with a hierarchy of trifocal tensors. ECCV 2003.

[17] R. Hartley and F. Schaffalitzky, PowerFactorization: an approach to affine reconstruction with missing and uncertain data. In Australia-Japan Advanced Workshop on Computer Vision, 2003.

[18] K. Sim, R. Hartley. Removing Outliers Using The L-∞ Norm. CVPR 2006.

[19] K. Sim, R. Hartley, Recovering Camera Motion Using L-∞ Minimization. CVPR 2006.

[20] P. Sturm and B. Triggs, A factorization based algorithm for multi-image projective structure and motion. ECCV 1996.

[21] P. Sturm, Algorithms for Plane-Based Pose Estimation. CVPR 2000.

[22] C. Rother and S. Carlsson, Linear Multi View reconstruction and Camera Recovery using a Reference Plane. IJCV 2002, 49(2/3):117-141.

[23] C. Tomasi, T. Kanade, Shape and motion from image streams under orthography: a factorization method. International Journal of Computer Vision, 9(2):137-154, November 1992.

[24] B. Triggs, Linear projective reconstruction from matching tensors. Image and Vision Computing, 15(8):617625, 1997.

[25] R. Vidal and R. Hartley, Motion Segmentation with Missing Data using PowerFactorization and GPCA. CVPR 2004.