A Generalized Projective Reconstruction Theorem and Depth Constraints for Projective Factorization

Behrooz Nasihatkon · **Richard Hartley** ·
Jochen Trumpf

Received: 13 July 2013 / Accepted: 23 January 2015 / Published online: 6 February 2015

Abstract This paper presents a generalized version of the classic projective reconstruction theorem which helps to choose or assess depth constraints for projective depth estimation algorithms. The theorem shows that projective reconstruction is possible under a much weaker constraint than requiring all estimated projective depths to be nonzero. This result enables us to present classes of depth constraints under which any reconstruction of cameras and points projecting into given image points is projectively equivalent to the true camera-point configuration. It also completely specifies the possible wrong configurations allowed by other constraints. We demonstrate the application of the theorem by analysing several constraints used in the literature, as well as presenting new constraints with desirable properties. We mention some of the implications of our results on iterative depth estimation algorithms and projective reconstruction via rank minimization. Our theory is verified by running experiments on both synthetic and real data.

Communicated by Long Quan.

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

B. Nasihatkon · R. Hartley · J. Trumpf
Australian National University, Canberra, Australia

B. Nasihatkon · R. Hartley
NICTA, Canberra, Australia

B. Nasihatkon (✉)
Chalmers University of Technology, Gothenburg, Sweden
e-mail: behrooz.nasihatkon@chalmers.se

Keywords Multiple view geometry · Projective reconstruction · Projective reconstruction theorem · Projective factorization · Projective depths · Constraints on projective depths

1 Introduction

This paper generalizes the classic theorem of projective reconstruction. The main purpose is to provide a theoretical basis for the choice and verification of constraints on projective depths for factorization-based projective reconstruction algorithms. The basic idea behind factorization-based projective reconstruction is to find an estimation $\hat{\Lambda} = [\hat{\lambda}_{ij}]$ of the projective depth matrix such that when the image data matrix $[\mathbf{x}_{ij}]$ is weighted by the elements of $\hat{\Lambda}$, it can be factored as a product of a $3m \times 4$ matrix $\hat{\mathtt{P}}$, representing the stack of m camera matrices, and a $4 \times n$ matrix $\hat{\mathtt{X}}$, representing the horizontal arrangement of n points. In other words, given the projected image data $\{\mathbf{x}_{ij}\}$ one tries to solve the following equation

$$\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\,\hat{\mathtt{X}}, \tag{1}$$

where the operator $\odot$ multiplies each element $\hat{\lambda}_{ij}$ of the depth matrix $\hat{\Lambda}$ by its corresponding image point $\mathbf{x}_{ij}$, that is $\hat{\Lambda} \odot [\mathbf{x}_{ij}] = [\hat{\lambda}_{ij}\mathbf{x}_{ij}]$. While the true depths, camera matrices and points provide a solution to (1), not every solution to (1) gives a configuration projectively equivalent to the true camera-point setup. Therefore, without putting extra constraints on the depth matrix the above problem can lead to false solutions.

The main source of the false solutions in the factorization-based methods is the possibility of having zero elements in $\hat{\Lambda}$. One can simply see that setting $\hat{\Lambda}$, $\hat{\mathtt{P}}$ and $\hat{\mathtt{X}}$ all equal to zero provides a solution to (1). Another trivial solution, as noted by Oliensis and Hartley (2007), occurs when $\hat{\Lambda}$ has all but four zero columns. In general, it has been noticed that false solutions to (1) can happen when some rows or some columns of the depth matrix are zero. There has been no research, however, specifying all possible false solutions to the factorization equation (1) and the constraints which can prevent them from happening. In this paper, in addition to the cases where the estimated depth matrix $\hat{\Lambda}$ has some zero rows or some zero columns, we present a less trivial class of false solutions where the depth matrix has a cross-shaped form (see Figs. 1 and 2). We shall further show that all the possible false solutions to factorization based projective reconstruction are confined to the above cases. Therefore, with a depth constraint which allows at least one correct solution, prevents zero rows and zero columns in the depth matrix and avoids cross-shaped configurations, any solution to the factorization problem (1) will lead to a correct projective reconstruction.

The main concern of this paper is the classification of the false solutions of (1), and the constraints which can avoid them. Therefore, we do not thoroughly deal with the question of how to solve (1). However, we have to be realistic in choosing proper constraints. The constraints have to possess some desirable properties to make possible the design of efficient and effective algorithms for solving (1). As a trivial example it is essential for many iterative algorithms that the constraint space is closed. As nearly all factorization-based algorithms are solved iteratively, this can guarantee that the algorithm does not converge to something that violates the constraints.

A major class of desirable constraints for projective factorization problems are linear equality constraints. The corresponding affine constraint space is both closed and convex, and usually leads to less complex factorization-based algorithms. We shall show that the linear equality constraints that are used so far in factorization-based reconstruction allow for cross-shaped depth matrices and hence cannot completely rule out false solutions. We shall further introduce *step-like constraints*, a class of linear equality constraints in the form of fixing certain elements of the depth matrix, which provably avoid all the degenerate cases in the factorization problem (see Fig. 3). The element-wise nature of these constraints makes the implementation of the associated factorization-based algorithms very simple.

Another desirable property for the constraint space, which is mutually exclusive with being an affine subspace, is compactness. The importance of a compact constraint space is that certain convergence properties can be proved for a large class of iterative descent algorithms when the sequence of solutions lie inside a compact set. One can think of many compact constraints, however, the important issue is that the constraint needs to be efficiently implementable with a factorization algorithm. Two examples of such constraints are presented in Heyden et al. (1999) and Mahamud et al. (2001), in which, respectively, all rows and all columns of the depth matrix are forced to have a fixed (weighted) l^2-norm. In each case, every iteration of the factorization algorithm requires solving a number of eigenvalue problems. Mahamud et al. (2001) prove the convergence of their algorithm to local

$$\begin{bmatrix} & & a & & & \\ & & b & & & \\ c & d & x & e & f & g \\ & & h & & & \end{bmatrix} \quad \begin{bmatrix} a & b & c & x & d & e \\ & & & f & & \\ & & & g & & \\ & & & h & & \end{bmatrix} \quad \begin{bmatrix} a & & & & & \\ b & & & & & \\ c & & & & & \\ x & d & e & f & g & h \end{bmatrix}$$

Fig. 1 Examples of 4×6 cross-shaped matrices. In cross-shaped matrices all elements of the matrix are zero, except those belonging to a special *row r* or a special *column c* of the matrix. The elements of the *r*th *row* and the *c*th *column* are all nonzero, except possibly the central element located at position (r, c). In the above examples, the *blank* parts of the matrices are zero. The elements $a, b, \ldots, h$ are all nonzero, while x can have any value (zero or nonzero)

$$\begin{array}{r|cccccc}
 & \overbrace{\quad}^{=\bar{\mathbf{C}}_1} & \multicolumn{5}{c}{\overbrace{\qquad\qquad\qquad\qquad}^{\hat{\mathbf{X}}_j=(\mathtt{I}-\bar{\mathbf{C}}_1\bar{\mathbf{C}}_1^T)\mathbf{X}_j}} \\
 & \hat{\mathbf{X}}_1 & \hat{\mathbf{X}}_2 & \hat{\mathbf{X}}_3 & \hat{\mathbf{X}}_4 & \hat{\mathbf{X}}_5 & \hat{\mathbf{X}}_6 \\
\hline
\mathtt{P}_1 = \hat{\mathtt{P}}_1 & 0 & \lambda_{12} & \lambda_{13} & \lambda_{14} & \lambda_{15} & \lambda_{16} \\
\mathtt{P}_2\mathbf{X}_1\bar{\mathbf{C}}_1^{\,T} = \hat{\mathtt{P}}_2 & \lambda_{21} & 0 & 0 & 0 & 0 & 0 \\
\mathtt{P}_3\mathbf{X}_1\bar{\mathbf{C}}_1^{\,T} = \hat{\mathtt{P}}_3 & \lambda_{31} & 0 & 0 & 0 & 0 & 0 \\
\mathtt{P}_4\mathbf{X}_1\bar{\mathbf{C}}_1^{\,T} = \hat{\mathtt{P}}_4 & \lambda_{41} & 0 & 0 & 0 & 0 & 0 \\
\mathtt{P}_5\mathbf{X}_1\bar{\mathbf{C}}_1^{\,T} = \hat{\mathtt{P}}_5 & \lambda_{51} & 0 & 0 & 0 & 0 & 0 \\
 & \multicolumn{6}{c}{\underbrace{\qquad\qquad\qquad\qquad\qquad\qquad}_{\hat{\Lambda}}}
\end{array}$$

Fig. 2 An example of a degenerate solution with a cross-shaped depth matrix $\hat{\Lambda}$. Here, $(\{\mathtt{P}_i\}, \{\mathbf{X}_{ij}\}, \{\lambda_{ij}\})$ is the configuration of true camera matrices, 3D points and projective depths. Given this true configuration, a solution $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_{ij}\}, \{\hat{\lambda}_{ij}\})$ has been constructed as shown in the figure, where $\bar{\mathbf{C}}_1$ is the normalized camera centre of $\mathtt{P}_1$ (a unit vector in the null space of $\mathtt{P}_1$). One can easily check that $\hat{\mathtt{P}}_i\hat{\mathtt{X}}_j = \hat{\lambda}_{ij}\mathbf{x}_{ij}$ where $\mathbf{x}_{ij} = \frac{1}{\lambda_{ij}}\mathtt{P}_i\mathbf{X}_j$. Therefore, this degenerate solution satisfies the projective equation $\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\,\hat{\mathtt{X}}$. See Sect. 5 for more details

$$\underset{\textbf{(a)}}{\begin{bmatrix} 1 & 1 & & & & \\ & 1 & 1 & 1 & & \\ & & & 1 & 1 & \\ & & & & 1 & 1 \end{bmatrix}} \quad \underset{\textbf{(b)}}{\begin{bmatrix} 1 & 1 & 1 & & & \\ & & 1 & & & \\ & & 1 & & & \\ & & 1 & 1 & 1 & 1 \end{bmatrix}} \quad \underset{\textbf{(c)}}{\begin{bmatrix} 1 & & & & & \\ 1 & & & & & \\ 1 & & & & & \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}}$$

Fig. 3 Examples of 4 × 6 step-like mask matrices. *Blank* parts of the matrices indicate zero values. A step-like matrix contains a chain of ones, starting from its *upper left* corner and ending at its *lower right* corner, made by making *rightward* and *downward* moves only. An exclusive step-like mask is one which is not cross-shaped. In the above, **a** and **b** are samples of an exclusive step-like mask while **c** is a nonexclusive one. Associated with an $m \times n$ step-like mask $\mathtt{M}$, one can put a constraint on an $m \times n$ depth matrix $\hat{\Lambda}$ in the form of fixing the elements of $\hat{\Lambda}$ to 1 (or some nonzero values) at the sites where $\mathtt{M}$ has ones. For an exclusive step-like mask, this type of constraint rules out all the wrong solutions to the factorization-based problems

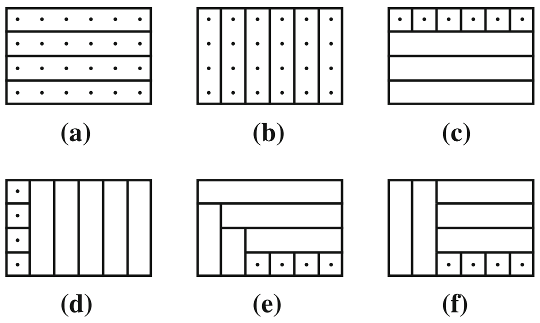

Fig. 4 Examples of tiling a 4×6 depth matrix with *row* and *column* vectors. The associated constraint is to force every tile of the depth matrix to have a unit (or a fixed) norm. This gives a compact constraint space. If the tiling is done according to (**a**), every *row* of the constrained depth matrix has unit norm. Similarly, tiling according to (**b**) requires *columns* with unit norms. Constraints associated with **a** and **b**, respectively, allow zero *columns* and zero *rows* in the depth matrix, along with cross-shaped configurations. The associated constraints for **c**–**f** do not allow any zero *rows* or zero *columns*, however, they all allow cross-shaped structures. For each of the cases **a**–**f**, the *dots* indicate possible locations where the cross-shaped structures can be centred. Clearly, for **a** and **b** the *cross* can be centred anywhere, whereas for **c**–**f** they can only be centred at 1×1 tiles

minima using the General Convergence Theorem (Zangwill 1969; Luenberger 1984). However, these constraints allow zero columns or zero rows in the depth matrix, as well as cross-shaped structures. In this paper, we combine the constraints used in Heyden et al. (1999) and Mahamud et al. (2001), in the sense of *tiling* the matrix with row and column vectors and requiring each tile to have a unit (or fixed) norm (see Fig. 4). With a proper tiling, convergence to configurations with zero rows and zero columns is ruled out. Such tilings still allow for cross-shaped structures, however, as shown in Fig. 4, the number of possible cross-shaped structures is limited.

1.1 Previous Attempts

It is clear from what has been mentioned above that this paper focuses on those algorithms that try to estimate the camera matrices and points directly, as opposed to the so-called tensor-based approaches in which the camera parameters are estimated from the fundamental matrices, trifocal tensors, or quadrifocal tensors, estimated, respectively, from image data of pairs, triples or quadruples of views (Hartley and Zisserman 2004). The advantage of the former class of algorithms is that they make uniform use of all image data to build a reconstruction, and thus do not give a solution that is biased towards particular views. It is also evident that we are only dealing with the cases where the estimation of projective depths is involved at some stage of the algorithm. This excludes approaches like Bundle Adjustment (Triggs et al. 2000). Here, we list some of the attempts made to solve the factorization problem.

1.1.1 Sturm–Triggs Factorization

The link between projective depth estimation and projective reconstruction of cameras and points was noted by Sturm and Triggs (1996), whereby it is shown that given the true projective depths, camera matrices and points can be found from the factorization of the data matrix weighted by the depths. However, to estimate the projective depths Sturm and Triggs make use of fundamental matrices estimated from pairwise image correspondences. Several papers have proposed that the Sturm–Triggs method can be extended to iteratively estimate the depth matrix $\hat{\Lambda}$ and camera-point configurations $\hat{\mathtt{P}}$ and $\hat{\mathtt{X}}$ (Triggs 1996; Ueshiba and Tomita 1998; Heyden et al. 1999; Mahamud et al. 2001; Hartley and Zisserman 2004). It has been noted that without constraining or normalizing the depths, such algorithms can converge to false solutions. Especially, Oliensis and Hartley (2007) show that the basic iterative generalization of the Sturm–Triggs factorization algorithm can converge to trivial false solutions, and that in the presence of the slightest amount of noise, it generally does not converge to a correct solution.

1.1.2 Unit Row Norm Constraint

Heyden et al. (1999) estimate the camera-point configuration and the projective depths alternatingly, under the constraint that every row of the depth matrix has unit l^2-norm. They also suggest a normalization step which scales each column of the depth matrix to make the first row of the matrix have all unit elements. However, they do not use this normalization step in their experiments, reporting better convergence properties in its absence. It is clear that by just requiring rows to have unit norm, we allow zero columns in the depth matrix as well as cross-shaped configurations. If all rows except the first are required to have unit norm, and at the same time (and not in a separate normalization step) the first row is constrained to have all unit elements, then having zero columns is not possible, but still a cross-shaped depth matrix is allowed. We refer the reader to Sect. 9 for experiments on this constraint.

1.1.3 Unit Column Norm Constraint

Mahamud et al. (2001) propose an algorithms which is in some ways similar to that of Heyden et al. (1999). Again, the depths and camera-point configuration are alternatingly estimated, but under the constraint that each column of the weighted data matrix has a unit l^2-norm. The convergence to a local minimum is proved, while no theoretical guarantee is given for not converging to a wrong solution. In fact, the above constraint can allow zero rows in the depth matrix in addition to cross-shaped depth matrices.

1.1.4 Fixed Row and Column Norms

Triggs (1996) suggests that the process of estimating depths and camera-point structure in the Sturm–Triggs algorithm can be done in an alternating and iterative fashion. He also suggests a depth balancing stage after the depth estimation phase, in which it is sought to rescale rows and columns of the depth matrix such that all rows have the same Euclidean length and similarly all columns have a common length. The same balancing scheme has been suggested by Hartley and Zisserman (2004). The normalization step is in the form of rescaling the rows to have similar norm and then doing the same to the columns. At each iteration, this can either be done once each, or in a repeated iterative fashion. If an l^p-norm is used for this procedure, alternatingly balancing rows and columns is the same as applying Sinkhorn's algorithm Sinkhorn (1964, 1967) to a matrix whose elements are $|\hat{\lambda}_{ij}|^p$ and thereby forcing all rows of the depth matrix to eventually have the same norm, and similarly all columns to have the same norm. We will show that forcing the matrix to have equal nonzero column norms and equal nonzero row norms will prevent false solutions to the factorization-based algorithm. However, the direct implementation of this constraint is difficult. Implementing it as a balancing stage after every iteration can destroy the property of having descent moves in the algorithm. Oliensis and Hartley (2007) report that the normalization step can lead to bad convergence properties.

1.1.5 CIESTA

Oliensis and Hartley (2007) prove that if the basic iterative factorization is done without putting any constraint on the depth matrix (except possibly retaining a global scale), it can converge to trivial false solutions. More interestingly, they show that in the presence of noise it generally always converges to a wrong solution. They also argue that many variants of the algorithm, including (Mahamud et al. 2001) and (Hartley and Zisserman 2004) either are likely to converge to false solutions or can exhibit undesirable convergence behavior. They propose a new algorithm, called CIESTA, which minimizes a regularized target function. Although some convergence properties have been proved for CIESTA, the solution is biased as it favors projective depths that are close to 1. For this choice, even when there is no noise present, the correct solution does not generally coincide with the global minimum of the CIESTA target function. We do not deal with such approaches in this paper.

1.1.6 Fixing Elements of a Row and a Column

Ueshiba and Tomita (1998) suggest estimating the projective depths through a conjugate gradient optimization process seeking to make the final singular values of the weighted image data matrix small, thus making it close to a rank-four matrix. To avoid having multiple solutions due to the ambiguity associated with the projective depths, the algorithm constrains the depth matrix to have all elements of the r-th row and the c-th column equal to one for some choice of r and c, that is $\hat{\lambda}_{ij} = 1$ when $i = r$ or $j = c$. This constraint can lead to cross-shaped configurations, although there is only one possible location for the centre of cross, namely (r, c).

1.1.7 Transportation Polytope Constraint

Dai et al. (2010, 2013) seek to estimate the depths by putting a rank constraint on the data matrix weighted by the depth matrix. The weighted data matrix is restricted to have rank four or less. In addition, the depth matrix is constrained to have fixed row and column sums. In addition, this approach also enforces the constraint $\hat{\lambda}_{ij} \geq 0$, that is the projective depths are all nonnegative.[1] In Angst et al. (2011) it has been noted that the corresponding constraint space is known as the Transportation Polytope. In Dai et al. (2010, 2013) the problem is formulated as a rank minimization problem and is solved by using the trace norm as a convex surrogate of the rank function. The relaxed optimization problem can be recast as a semi-definite program. One drawback of this approach is the use of inequality constraints, preventing it from taking advantage of the fast rank minimization techniques for large scale data such as Lin et al. (2010); Yang and Yuan (2013). The same idea as Dai et al. (2010) is used in Angst et al. (2011), however, a generalized trace norm target function is exploited to approximate the rank. While the authors mention the transportation polytope constraint space, for implementation just a single constraint is used that fixes the total scale of the whole depth matrix. As this constraint is prone to giving degenerate trivial solutions, the authors add inequality constraints whenever necessary. We shall show that the transportation polytope constraint avoids false solutions to the factorization methods if the marginal values to which rows and columns must sum up are chosen properly.

1.1.8 Fixed Row and Column Sums

As noted before, the inequality constraint used in (Dai et al. 2010, 2013) can prevent the design of fast algorithms. This might be the reason why, when it comes to introducing scalable algorithms in (Dai et al. 2013), the inequality constraint has been neglected. We will show that neglecting the inequality constraint and just constraining row and columns to have specified sums always allows for cross-shaped structures and

[1] Actually, in Dai et al. (2010, 2013) the constraint is given as imposing strictly positive depths: $\hat{\lambda}_{ij} > 0$, giving a non-closed constraint space. However, what can be implemented in practice using semi-definite programming or other iterative methods is non-strict inequalities $\hat{\lambda}_{ij} \geq 0$.

thus for false solutions. However, as argued in Sect. 6.2.1, it is difficult to converge to such structures under these constraints starting from a sensible initial solution (see Fig. 8). This belief is supported by our experiments in Sect. 9.

In what comes next, we first define the problem in precise mathematical terms and discuss in more detail why it is not fully solved by the previous results in multiple view geometry (Sects. 2 and 3). Then, we present a more general version of the Theorem of Projective Reconstruction working under a much weaker set of assumptions than the depths being all nonzero (Sect. 4). By giving a counterexample we then demonstrate that the assumptions made for the proof of our theorem are minimal in a certain sense (Sect. 5). Afterwards, in Sect. 6, we present a class of constraints, called *reconstruction friendly constraints*, under which false solutions to the projective factorization problem are avoided. Then we show how the results developed here can be used for the assessment of depth constraints by analysing some of the constraints used in the literature. We also give examples demonstrating how our results can be exploited for the design of new constraints with desirable properties. Especially, we present the step-like mask constraints as examples of reconstruction friendly constraints in the form of linear equations. In Sect. 7 we study the implications of our results to the rank minimization approach to projective reconstruction. Then, in Sect. 8, we consider the case where the output of the algorithm is in the form of a convergent sequence of solutions. The paper is finished by running experimental tests on synthetic and real data (Sect. 9).

2 Motivation

Consider a set of projection matrices $\mathtt{P}_1, \mathtt{P}_2, \ldots, \mathtt{P}_m \in \mathbb{R}^{3\times 4}$, a set of points $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n \in \mathbb{R}^4$. Each point $\mathbf{X}_j$ is projected through each camera matrix $\mathtt{P}_i$ to produce the set of image points $\mathbf{x}_{ij} \in \mathbb{R}^3$ according to

$$\lambda_{ij}\mathbf{x}_{ij} = \mathtt{P}_i\mathbf{X}_j$$

where $\lambda_{ij} \neq 0$ are nonzero scalars known as *projective depths*. The projective depths λ_{ij}, $i = 1, \ldots, m$, $j = 1, \ldots, n$, can be arranged as an $m \times n$ array to form the *depth matrix* $\Lambda = [\lambda_{ij}]$. Similarly, the image data $\{\mathbf{x}_{ij}\}$ can be arranged as a $3m \times n$ matrix $[\mathbf{x}_{ij}]$ called here the *data matrix*. The above equation can be written in the matrix form

$$\Lambda \odot [\mathbf{x}_{ij}] = \mathtt{P}\,\mathtt{X}, \tag{2}$$

where $\mathtt{P} = \mathrm{stack}(\mathtt{P}_1, \mathtt{P}_2, \cdots, \mathtt{P}_m)$ is the vertical concatenation of the camera matrices, $\mathtt{X} = [\mathbf{X}_1\mathbf{X}_2\cdots\mathbf{X}_n]$ and $\Lambda \odot [\mathbf{x}_{ij}] = [\lambda_{ij}\mathbf{x}_{ij}]$, that is the operator $\odot$ multiplies each element λ_{ij} of Λ by the corresponding 3×1 block $\mathbf{x}_{ij}$ of the data matrix $[\mathbf{x}_{ij}]$. We stress that in this paper the projection matrices and points are not considered as projective quantities, unless explicitly stated. No equation, therefore, implies equality up to scale. From (2) it is obvious that having the true depth matrix Λ, the weighted data matrix $\Lambda \odot [\mathbf{x}_{ij}] = [\lambda_{ij}\mathbf{x}_{ij}]$ can be factored as the product of a $3m \times 4$ matrix $\mathtt{P}$ and a $4 \times n$ matrix $\mathtt{X}$. Equivalently, the matrix $\Lambda \odot [\mathbf{x}_{ij}]$ has rank 4 or less. This is where the underlying idea of factorization-based algorithms comes from. These algorithms try to find the camera-matrix configuration from the given image data $\{\mathbf{x}_{ij}\}$ by finding a depth matrix $\hat{\Lambda}$ for which $\hat{\Lambda} \odot [\mathbf{x}_{ij}]$ has rank 4 (or less), and thus, can be factored as the product of $3m \times 4$ and $4 \times n$ matrices $\hat{\mathtt{P}}$ and $\hat{\mathtt{X}}$:

$$\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\,\hat{\mathtt{X}}. \tag{3}$$

One hopes that by solving the above problem, dividing $\hat{\mathtt{P}}$ into blocks $\hat{\mathtt{P}}_i \in \mathbb{R}^{3\times 4}$ as $\hat{\mathtt{P}} = \mathrm{stack}(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_2, \cdots, \hat{\mathtt{P}}_m)$ and letting $\hat{\mathbf{X}}_j$ be the j-th column of $\hat{\mathtt{X}}$, the camera-point configuration $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ is equal to the true configuration $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ up to a projective ambiguity.[2] However, it is obvious that given the data matrix $[\mathbf{x}_{ij}]$ not every solution to (3) gives the true configuration up to projectivity. A simple reason is the existence of trivial solutions, such as $\hat{\Lambda} = 0$, $\hat{\mathtt{P}} = 0$, $\hat{\mathtt{X}} = 0$, or when $\hat{\Lambda}$ has all but four of its columns equal to zero. In the latter case it is obvious that $\hat{\Lambda} \odot [\mathbf{x}_{ij}]$ can be factored as (3) as it has at most rank 4. This is why we see that in almost all projective depth estimation algorithms the depth matrix is restricted to some constraint space. This can be done either by optimizing some target function subject to an explicit constraint, or as a normalization or balancing stage after every iteration. While the constraints are sometimes said to guarantee the uniqueness of the solution to (3), their main purpose is to prevent the depth estimation procedure from collapsing to the so-called trivial solutions where some columns (or rows) of the depth matrix converge to zero (see Oliensis and Hartley 2007 for more detail). It is not obvious, however, (and we shall prove it false) if possible false solutions to (3) are restricted to these trivial cases. Therefore, factorization-based algorithms lack a proper theoretical basis for finding possible false solutions allowed by given constraints or to determine what constraints on the depth matrix make every solution to (3) projectively equivalent to the ground truth.

The main theoretical basis for the analysis of projective reconstruction is the Projective Reconstruction Theorem (Hartley and Zisserman 2004). It says that, under certain generic conditions, all configurations of camera matrices and 3D points yielding a common set of 2D image points are equal up to a projective ambiguity. This theorem is derived from a geometric perspective and therefore presumes assumptions

[2] Throughout the paper we follow the convention that estimated depths, camera matrices and points are denoted using the hatted quantities such as $\hat{\lambda}_{ij}$, $\hat{\mathtt{P}}_i$ and $\hat{\mathbf{X}}_j$, while the ground truth is shown using unhatted symbols like λ_{ij}, $\mathtt{P}_i$ and $\mathbf{X}_j$.

like the estimated camera matrices $\hat{\mathtt{P}}_i$ having full row rank and all the estimated projective depths $\hat{\lambda}_{ij}$ being nonzero. While these are useful enough for the so-called tensor-based reconstruction approaches, they are not a good fit for the analysis of algebraic algorithms, especially factorization-based depth estimation algorithms. Obviously, these geometric assumptions can be reasonably assumed for the true set of depths $\{\lambda_{ij}\}$ and the true camera-point configuration $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$. However, for most of the factorization-based algorithms, at least in the case of large-scale problems, it is hard to impose these constraints on the *estimated* depths $\{\hat{\lambda}_{ij}\}$ and camera-point configuration $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ a priori.

Actually, the basic assumption for the proof of the classic Projective Reconstruction Theorem (Hartley and Zisserman 2004) is that the estimated depths $\hat{\lambda}_{ij}$ are all nonzero. Other geometric assumptions like full-row-rank estimated camera matrices $\hat{\mathtt{P}}_i$ follow from this assumption under reasonable conditions. Therefore, one might like to enforce $\hat{\lambda}_{ij} \neq 0$ as a constraint for any algorithm for solving (3), and make use of this theorem to show that the algorithm avoids false solutions. However, this type of constraint space cannot be easily applied in most of the iterative algorithms. Since this constraint space is not closed, it is possible that the procedure may converge to a solution outside the constraint space, even if in all iterations the solution lies inside the constraint space. In this case, some of the projective depths can converge to zero, resulting in a degenerate solution. Making use of the scale ambiguity of the projective depths, the constraint space can be made closed by using $|\hat{\lambda}_{ij}| \geq \delta$ for some positive number δ rather than $\hat{\lambda}_{ij} \neq 0$. However, this non-connected constraint space again cannot be easily handled by many of the iteration based algorithms. Actually, in practice, when there is no missing data, it is usually the case that all true depths λ_{ij} are positive, as all the 3D points are in front of the cameras. In this case, we can have a convex constraint space by forcing all depths to be positive, that is $\hat{\lambda}_{ij} > 0$. Obviously, due to the scale ambiguity, the constraint space can be made closed by using $\hat{\lambda}_{ij} \geq \delta$ instead, for some $\delta > 0$. This gives a set of linear inequalities.

One problem with the inequality constraints is that they are hard to implement for fast and efficient factorization-based algorithms, especially for large-scale problems. Thus, we seek even simpler constraints making the optimization-based techniques more efficient and easier to solve. For example, linear equality constraints, which are easier to handle and for which usually much faster algorithms exist compared to inequality constraints. This can be seen, for example, in state-of-the-art algorithms designed for the convex relaxation of large scale rank minimization problems which work with linear equality constraints (Lin et al. 2010; Yang and Yuan 2013). We observed the use of linear equality constraints in papers like Ueshiba and Tomita (1998) (by fixing special elements of the depth matrix $\hat{\Lambda}$) and also Dai et al. (2010, 2013) (by fixing the row and column sums of $\hat{\Lambda}$) when it comes to large scale problems. We also observed other examples of constraints like requiring rows of $\hat{\Lambda}$ (Heyden et al. 1999), or columns of $\hat{\Lambda} \odot [\mathbf{x}_{ij}]$ (Mahamud et al. 2001) to have a unit l^2-norm, which allowed for efficient factorization-based algorithms. However, as these constraints, per se, are unable to guarantee all depths to be nonzero or strictly positive, we cannot take advantage of the classic theorem of projective reconstruction to analyse their effectiveness. This shows the need to finding weaker conditions under which projective reconstruction succeeds. The new conditions must allow the verification of the constraints that fit the factorization-based algorithms. We will introduce such a theorem in Sect. 4, after providing the required background in the next section.

3 Background

3.1 Notations

We denote matrices by typewriter letters ($\mathtt{X}$), vectors by bold letters ($\mathbf{x}$ or $\mathbf{X}$), sets by upper-case normal letters (X), scalars by lower-case normal letters x, and mappings (functions) by calligraphic letters ($\mathcal{X}$). Mappings $\mathcal{C}$, $\mathcal{R}$ and $\mathcal{N}$ are respectively used to represent the column space, row space and null space of a matrix. For matrices $\mathtt{A}_1, \mathtt{A}_2, \ldots, \mathtt{A}_m$ sharing the same number of columns, $\text{stack}(\mathtt{A}_1, \mathtt{A}_2, \ldots, \mathtt{A}_m)$ denotes their vertical concatenation. The reader must keep in mind that all expressions here are in algebraic affine geometry sense. The equality sign "=", here, means algebraically equal and not equal up to scale.

3.2 Projective Equivalence and the Depth Matrix

For a set of 3×4 projection matrices $\mathtt{P}_1, \mathtt{P}_2, \ldots, \mathtt{P}_m$, a set of points $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n$ in $\mathbb{R}^4$, and a set of image data $\mathbf{x}_{ij} \in \mathbb{R}^3$ formed according to the projection relation

$$\lambda_{ij}\mathbf{x}_{ij} = \mathtt{P}_i\mathbf{X}_j$$

with nonzero projective depths $\lambda_{ij} \neq 0$, the problem of projective reconstruction is to find the projection matrices $\mathtt{P}_i$ and the points $\mathbf{X}_j$ up to a projective ambiguity given the image points $\mathbf{x}_{ij}$. The next definitions make clear what is meant by projective ambiguity and projective equivalence. Readers can refer to Hartley and Zisserman (2004) for more details.

Definition 1 Two sets of projection matrices $\{\mathtt{P}_i\}$ and $\{\hat{\mathtt{P}}_i\}$, with $\mathtt{P}_i, \hat{\mathtt{P}}_i \in \mathbb{R}^{3\times 4}$ for $i = 1, 2, \ldots, m$ are projectively equivalent if there exist *nonzero* scalars $\tau_1, \tau_2, \ldots, \tau_m$ and a 4×4 invertible matrix $\mathtt{H}$ such that

$$\hat{\mathtt{P}}_i = \tau_i\, \mathtt{P}_i\, \mathtt{H}, \quad i = 1, 2, \ldots, m. \tag{4}$$

Definition 2 Two configurations of m projections and n points, namely $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ where $\mathtt{P}_i, \hat{\mathtt{P}}_i \in \mathbb{R}^{3\times4}$ for $i = 1, 2, \ldots, m$ and $\mathbf{X}_j, \hat{\mathbf{X}}_j \in \mathbb{R}^4$ for $j = 1, 2, \ldots, n$, are projectively equivalent if there exist an invertible 4×4 matrix $\mathtt{H}$ and *nonzero* scalars $\tau_1, \tau_2, \ldots, \tau_m$ and $\nu_1, \nu_2, \ldots, \nu_n$ such that

$$\hat{\mathtt{P}}_i = \tau_i \, \mathtt{P}_i \, \mathtt{H}, \qquad i = 1, 2, \ldots, m, \tag{5}$$

$$\hat{\mathbf{X}}_j = \nu_j \, \mathtt{H}^{-1} \mathbf{X}_j, \qquad j = 1, 2, \ldots, n. \tag{6}$$

We need to see the implications of projective equivalence of $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ on the depth matrices $\Lambda = [\lambda_{ij}]$ and $\hat{\Lambda} = [\hat{\lambda}_{ij}]$. First, we define the concept of *diagonal equivalence*[3] for matrices:

Definition 3 Two $m \times n$ matrices Λ and $\hat{\Lambda}$ are diagonally equivalent if there exist *nonzero* scalars $\tau_1, \tau_2, \ldots, \tau_m$ and $\nu_1, \nu_2, \ldots, \nu_n$ such that

$$\hat{\Lambda} = \operatorname{diag}(\boldsymbol{\tau}) \, \Lambda \, \operatorname{diag}(\boldsymbol{\nu}) \tag{7}$$

where $\boldsymbol{\tau} = [\tau_1, \tau_2, \ldots, \tau_m]^T$, $\boldsymbol{\nu} = [\nu_1, \nu_2, \ldots, \nu_n]^T$ and $\operatorname{diag}(\cdot)$ arranges the entries of a vector on the diagonal of a diagonal matrix.

The concepts of projective equivalence of projections and points and diagonal equivalence of depth matrices are related by the following lemma whose proof is given in Appendix 1.

Lemma 1 *Consider two configurations of $m \geq 2$ projection matrices and $n \geq 4$ points $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$, with $\mathtt{P}_i, \hat{\mathtt{P}}_i \in \mathbb{R}^{3\times4}$ and $\mathbf{X}_j, \hat{\mathbf{X}}_j \in \mathbb{R}^4$, such that*

(i) $\mathtt{P}_i \mathbf{X}_j \neq \mathbf{0}$ *for all i, j,*
(ii) $\operatorname{span}(\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n) = \mathbb{R}^4$, *and*
(iii) $\mathtt{P} = \operatorname{stack}(\mathtt{P}_1, \mathtt{P}_2, \ldots, \mathtt{P}_m)$ *has full column rank.*

Also, consider two $m \times n$ matrices $\Lambda = [\lambda_{ij}]$ and $\hat{\Lambda} = [\hat{\lambda}_{ij}]$. If the relations

$$\lambda_{ij} \mathbf{x}_{ij} = \mathtt{P}_i \mathbf{X}_j$$
$$\hat{\lambda}_{ij} \mathbf{x}_{ij} = \hat{\mathtt{P}}_i \hat{\mathbf{X}}_j$$

hold for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$, then $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent if and only if the matrices Λ and $\hat{\Lambda}$ are diagonally equivalent.

3.3 The Fundamental Matrix

Another important entity used in this paper is the *fundamental matrix*. For two cameras, the fundamental matrix gives a bilinear relation between pairs of corresponding image points. There are many different ways to define the fundamental matrix (Hartley and Zisserman 2004). Here, we choose the following definition

Definition 4 For two 3×4 matrices $\mathtt{Q}$ and $\mathtt{R}$, the corresponding fundamental matrix is represented by the function value $\mathscr{F}(\mathtt{Q}, \mathtt{R})$, where $\mathscr{F}\colon \mathbb{R}^{3\times4} \times \mathbb{R}^{3\times4} \to \mathbb{R}^{3\times3}$ is defined as

$$[\mathscr{F}(\mathtt{Q}, \mathtt{R})]_{ki} = (-1)^{i+k} \det \begin{bmatrix} \mathtt{Q}_{-i} \\ \mathtt{R}_{-k} \end{bmatrix} \tag{8}$$

where $\mathtt{Q}_{-i} \in \mathbb{R}^{2\times4}$ is formed by removing the i-th row of $\mathtt{Q}$ and $\mathtt{R}_{-k}$ is defined similarly.

For more details of this definition we refer the reader to (Hartley and Zisserman 2004, Sect.17.1). Notice that the fundamental matrix is the output of the function $\mathscr{F}$ applied to $\mathtt{Q}$ and $\mathtt{R}$ and not the mapping $\mathscr{F}$ itself. One of the advantages of using the above definition for the fundamental matrix is that it is not restricted to the case of proper full-rank camera matrices. It can be defined for any pair of 3×4 matrices. Also, the reader must keep in mind that, like other entities in this paper, the fundamental matrix here is treated as a member of $\mathbb{R}^{3\times3}$, not as an up-to-scale equivalence class of matrices. Basically, the above definition says that the elements of the fundamental matrix of two matrices $\mathtt{Q}, \mathtt{R} \in \mathbb{R}^{3\times4}$ are the determinant of 4×4 submatrices $\operatorname{stack}(\mathtt{Q}, \mathtt{R})$ made by choosing two rows from $\mathtt{Q}$ and two rows from $\mathtt{R}$. This gives the following lemma

Lemma 2 *For two 3×4 matrices $\mathtt{Q}$ and $\mathtt{R}$, the fundamental matrix $\mathscr{F}(\mathtt{Q}, \mathtt{R})$ is nonzero if and only if there exists an* invertible 4×4 *submatrix of* $\operatorname{stack}(\mathtt{Q}, \mathtt{R})$ *made by choosing two rows from $\mathtt{Q}$ and two rows from $\mathtt{R}$.*

The next two lemmas about the fundamental matrix will be used later in this paper.

Lemma 3 (Hartley and Zisserman 2004) *Consider two pairs of camera matrices $\mathtt{Q}, \mathtt{R}$ and $\hat{\mathtt{Q}}, \hat{\mathtt{R}}$ such that $\mathtt{Q}$ and $\mathtt{R}$ both have full row rank and also have distinct null spaces, that is $\mathscr{N}(\mathtt{Q}) \neq \mathscr{N}(\mathtt{R})$. Then $(\mathtt{Q}, \mathtt{R})$ and $(\hat{\mathtt{Q}}, \hat{\mathtt{R}})$ are projectively equivalent according to Definition 1 if and only if $\mathscr{F}(\mathtt{Q}, \mathtt{R})$ and $\mathscr{F}(\hat{\mathtt{Q}}, \hat{\mathtt{R}})$ are equal up to a* nonzero *scaling factor.*

Notice that, unlike $(\mathtt{Q}, \mathtt{R})$, no assumptions are made in the above about $(\hat{\mathtt{Q}}, \hat{\mathtt{R}})$.

Lemma 4 (Hartley and Zisserman 2004) *Consider two full-row-rank matrices $\mathtt{Q}$ and $\mathtt{R}$ such that $\mathscr{N}(\mathtt{Q}) \neq \mathscr{N}(\mathtt{R})$. If for a matrix $\mathtt{F} \in \mathbb{R}^{3\times3}$ the relation*

$$\mathtt{Q}^T \mathtt{F} \mathtt{R} + \mathtt{R}^T \mathtt{F}^T \mathtt{Q} = \mathtt{0}_{4\times4}$$

holds (or equivalently $\mathbf{X}^T(\mathtt{Q}^T \mathtt{F} \mathtt{R})\mathbf{X} = 0$ holds for all $\mathbf{X} \in \mathbb{R}^4$), then $\mathtt{F}$ is equal to $\mathscr{F}(\mathtt{Q}, \mathtt{R})$ up to a scaling factor.

[3] This term has been borrowed from Sinkhorn (1967).

3.4 Cross-shaped Matrices

The concept of *cross-shaped* matrices is essential for the statement of our main theorem and the characterization of false solutions to the projective factorization problem.

Definition 5 A matrix $\mathtt{A} = [a_{ij}]$ is said to be *cross-shaped*, if it has a row r and a column c for which

$$\begin{cases} a_{ij} = 0 & i \neq r, \quad j \neq c, \\ a_{ij} \neq 0 & i = r, \quad j \neq c, \\ a_{ij} \neq 0 & i \neq r, \quad j = c. \end{cases} \tag{9}$$

The pair of indices (r, c) is called the *centre* of a cross-shaped matrix and a_{rc} is called its *central element*, which can be either zero or nonzero according to (9). A cross-shaped matrix can be *zero-centred* or *nonzero-centred* depending on whether the central element a_{rc} is zero or nonzero.

A cross-shaped matrix has all of its elements equal to zero except the elements of a certain row r and a certain column c. The r-th row and the c-th column have all nonzero elements, except at their junction where the element can be zero or nonzero. Examples of cross-shaped matrices are depicted in Fig. 1. Notice that any permutation to rows and columns of a cross-shaped matrix results in another cross-shaped matrix.

Lemma 5 (i) *Any two $m \times n$ nonzero-centred cross-shaped matrices with a common centre (r, c) are diagonally equivalent.* (ii) *any two $m \times n$ zero-centred cross-shaped matrices with a common centre (r, c) are diagonally equivalent.*

Proof Consider two $m \times n$ cross-shaped matrices $\mathtt{A} = [a_{ij}]$ and $\mathtt{B} = [b_{ij}]$ with a common centre (r, c). According to Definition 3, to prove diagonal equivalence we need to show that $\mathtt{B} = \mathrm{diag}(\boldsymbol{\tau})\,\mathtt{A}\,\mathrm{diag}(\boldsymbol{\nu})$ for some vectors $\boldsymbol{\tau}$ and $\boldsymbol{\nu}$ with all nonzero entries. If $\mathtt{A}$ and $\mathtt{B}$ are both zero-centred, that is $a_{rc} = b_{rc} = 0$, then we choose the vectors $\boldsymbol{\tau} = (\tau_1, \tau_2, \ldots, \tau_m)^T$ and $\boldsymbol{\nu} = (\nu_1, \nu_2, \ldots, \nu_n)^T$, such that $\tau_r = \nu_c = 1$, $\tau_i = b_{ic}/a_{ic}$ for $i \neq r$, and $\nu_j = b_{rj}/a_{rj}$ for $j \neq c$. If $\mathtt{A}$ and $\mathtt{B}$ are both nonzero-centred, that is $a_{rc} \neq 0$ and $b_{rc} \neq 0$, then the vectors $\boldsymbol{\tau} = (\tau_1, \tau_2, \ldots, \tau_m)^T$ and $\boldsymbol{\nu} = (\nu_1, \nu_2, \ldots, \nu_n)^T$ are chosen such that $\tau_i = b_{ic}/a_{ic}$ for $i = 1, \ldots, m$, $\nu_c = 1$, and $\nu_j = b_{rj}/(a_{rj}\tau_r)$ for $j \neq c$. In either cases, one can easily check that $\boldsymbol{\tau}$ and $\boldsymbol{\nu}$ have all-nonzero entries and $\mathtt{B} = \mathrm{diag}(\boldsymbol{\tau})\,\mathtt{A}\,\mathrm{diag}(\boldsymbol{\nu})$. □

Now, we have the required tools to state our main theorem on projective reconstruction.

4 A General Projective Reconstruction Theorem

In this section we give a projective reconstruction theorem which is more general than the classic theorem in the sense that it does not assume, a priori, that the estimated depths $\hat{\lambda}_{ij}$ are all nonzero. This provides significantly more flexibility in the choice of depth constraints for depth estimation algorithms.

Our general projective reconstruction theorem is then:

Theorem 1 *Consider a set of $m \geq 2$ camera matrices $\{\mathtt{P}_i\}$ and $n \geq 8$ points $\{\mathbf{X}_j\}$ which are generic in the sense of conditions (G1–G4) which will be introduced later, and project into a set of image points $\{\mathbf{x}_{ij}\}$ according to*

$$\lambda_{ij}\mathbf{x}_{ij} = \mathtt{P}_i\mathbf{X}_j, \tag{10}$$

for nonzero depths $\lambda_{ij} \neq 0$ for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. Now, consider any other configuration of m camera matrices $\{\hat{\mathtt{P}}_i\}$, n points $\{\hat{\mathbf{X}}_j\}$ and mn depths $\{\hat{\lambda}_{ij}\}$ related to the same image data $\{\mathbf{x}_{ij}\}$ by

$$\hat{\lambda}_{ij}\mathbf{x}_{ij} = \hat{\mathtt{P}}_i\hat{\mathbf{X}}_j. \tag{11}$$

If the depth matrix $\hat{\Lambda} = [\hat{\lambda}_{ij}]$ satisfies the following conditions

(D1) $\hat{\Lambda}$ has no zero columns,
(D2) $\hat{\Lambda}$ has no zero rows, and
(D3) $\hat{\Lambda}$ is not a cross-shaped matrix,

then the camera-point configuration $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ is projectively equivalent to $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$.

Furthermore, we shall show in Sect. 5 that if any of the depth assumptions (D1), (D2) or (D3) is removed, it allows the existence of a configuration $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$, satisfying the relations $\hat{\lambda}_{ij}\mathbf{x}_{ij} = \hat{\mathtt{P}}_i\hat{\mathbf{X}}_j$ and projectively non-equivalent to $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$.

Loosely speaking, by true camera matrices $\mathtt{P}_i$ and points $\mathbf{X}_j$ being generic, we mean that the camera matrices have full row rank and the points and camera centres are in general position. In Sect. 4.1 we will be more specific about the required genericity conditions and mention four generic properties (G1–G4) under which Theorem 1 is true. To understand the paper, it is essential to notice that the genericity assumptions only apply to the true configuration $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$. No assumption is made about the estimated (hatted) quantities $\hat{\mathtt{P}}_i$ and $\hat{\mathbf{X}}_j$ except the relation $\hat{\lambda}_{ij}\mathbf{x}_{ij} = \hat{\mathtt{P}}_i\hat{\mathbf{X}}_j$. We do not a priori rule out the possibility that $\hat{\mathtt{P}}_i$-s or $\hat{\mathbf{X}}_j$-s belong to some non-generic set. Referring to $\hat{\mathtt{P}}_i$-s as camera matrices carries no implications about them whatsoever other than that they are 3×4 real matrices. They can be rank-deficient or even zero unless the opposite is proven.

At a first glance, Theorem 1 might seem contradictory, as it says that only some small subset of the elements of $\hat{\Lambda} = [\hat{\lambda}_{ij}]$ being nonzero is sufficient for $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ being projectively equivalent. On the other hand, from Lemma 1 we know that if $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent, then $\hat{\Lambda}$ must be diagonally equivalent to Λ and hence have all nonzero elements. The matter is that one has to distinguish between the implications of depth assumptions (D1–D3) in their own rights and their implications combined with the relations $\hat{\lambda}_{ij}\mathbf{x}_{ij} = \hat{\mathtt{P}}_i\hat{\mathbf{X}}_j$. Theorem 1, therefore, implies that if a special subset of depths $\{\hat{\lambda}_{ij}\}$ are known to be nonzero, then all of them are. This provides a sound theoretical base for choosing and analysing depth constraints for factorization-based projective reconstruction.

4.1 The Generic Camera-Point Setup

It is known that projective reconstruction from image data can be problematic if the (true) camera matrices and points belong to special degenerate setups (Hartley and Zisserman 2004; Hartley and Kahl 2007). The Projective Reconstruction Theorem is then said to be generically true, meaning that is can be proved under some generic assumptions about how the ground truth is configured. Here, we list the generic assumptions made about the true setup of cameras and points for the proof of our theorem.

We assume that there exist $m \geq 2$ camera matrices $\mathtt{P}_1, \mathtt{P}_2, \ldots, \mathtt{P}_m \in \mathbb{R}^{3\times4}$ and $n \geq 8$ points $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n$ in $\mathbb{R}^4$. They are generically configured in the following sense:

(G1) All camera matrices $\mathtt{P}_1, \mathtt{P}_2, \ldots, \mathtt{P}_m \in \mathbb{R}^{3\times4}$ have full row rank.

(G2) Taking any two views i and k, and two nonzero vectors $\mathbf{C}_i \in \mathcal{N}(\mathtt{P}_i)$ and $\mathbf{C}_k \in \mathcal{N}(\mathtt{P}_k)$, any four vectors among $\mathbf{C}_i, \mathbf{C}_k, \mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n$, are linearly independent.

(G3) For any view i, and a nonzero vector $\mathbf{C}_i \in \mathcal{N}(\mathtt{P}_i)$, no n points among $\mathbf{C}_i, \mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n$ lie on a twisted cubic[4] (or any of the degenerate critical sets resulting in a resection ambiguity, see (Hartley and Zisserman 2004, Sect. 22.1 and Hartley and Kahl 2007).

(G4) For any two views i and k, and two nonzero vectors $\mathbf{C}_i \in \mathcal{N}(\mathtt{P}_i)$ and $\mathbf{C}_k \in \mathcal{N}(\mathtt{P}_k)$, the points $\{\mathbf{C}_i, \mathbf{C}_k\} \cup \{\mathbf{X}_j\}_{j=1,\ldots,n}$ do not all lie on any (proper or degenerate) ruled quadric surface[4] (see Hartley and Zisserman 2004, Sect. 22.2 and Hartley and Kahl 2007).

Notice that condition (G1) makes the choice of $\mathbf{C}_i$ and $\mathbf{C}_k$ in conditions (G2–G4) unique up to scale. It implies that that any nonzero $\mathbf{C}_i \in \mathcal{N}(\mathtt{P}_i)$ represents the camera centre of $\mathtt{P}_i$. Condition (G2) has many implications when combined with (G1). Here, we list the ones needed in the paper:

(G2-1) For all i and j we have $\mathtt{P}_i\mathbf{X}_j \neq \mathbf{0}$ (as for any nonzero $\mathbf{C}_i \in \mathcal{N}(\mathtt{P}_i)$, $\mathbf{C}_i$ and $\mathbf{X}_j$ are linearly independent). Geometrically, $\mathbf{X}_j$ does not coincide with the camera centre of $\mathtt{P}_i$.

(G2-2) For any two views i, k we have $\mathcal{N}(\mathtt{P}_i) \neq \mathcal{N}(\mathtt{P}_k)$, and hence, no pair of cameras share a common camera centre.

(G2-3) For any two views i, k, $\mathrm{stack}(\mathtt{P}_i, \mathtt{P}_k)$ has full row rank, and thus, so does $\mathtt{P} = \mathrm{stack}(\mathtt{P}_1, \mathtt{P}_2, \ldots, \mathtt{P}_m)$.

(G2-4) For any two views i, k, and any point $\mathbf{X}_j$, the three nonzero vectors $\mathbf{C}_i$, $\mathbf{C}_k$ and $\mathbf{X}_j$ are linearly independent and therefore, $\mathbf{X}_j$ does not lie on the projective line[4] joining the camera centres of $\mathtt{P}_i$ and $\mathtt{P}_k$.

(G2-5) For any view i, any three vectors among $\mathtt{P}_i\mathbf{X}_1, \mathtt{P}_i\mathbf{X}_2, \ldots, \mathtt{P}_i\mathbf{X}_n$ are linearly independent (as $\mathbf{C}_i \notin \mathrm{span}(\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}_3)$ for any three distinct vectors $\mathbf{Y}_1, \mathbf{Y}_2, \mathbf{Y}_3 \in \{\mathbf{X}_j\}$ and any nonzero vector $\mathbf{C}_i \in \mathcal{N}(\mathtt{P}_i)$).

Notice that conditions (G3) and (G4) are generic for $n \geq 8$, because of the facts that 6 points in general position completely specify a twisted cubic and 9 points in general position determine a quadric surface (Semple and Kneebone 1952). One might find tighter generic conditions under which our projective reconstruction theorem is still true. However, we avoid doing this as it unnecessarily complicates the paper.

4.2 Overview of the Proof

Here, we state the general outline of the proof. Each part of the proof will then be demonstrated in a separate subsection. The complete proof is rather long and intricate. The reader may therefore wish to skip to Sect. 5.

Proof (Sketch of the Proof for Theorem 1) Under the theorem's assumptions, we shall show the following:

- There exist at least two views k and l for which the corresponding fundamental matrix $\mathcal{F}(\hat{\mathtt{P}}_k, \hat{\mathtt{P}}_l)$ is nonzero (Sect. 4.3).
- If $\mathcal{F}(\hat{\mathtt{P}}_k, \hat{\mathtt{P}}_l) \neq 0$ then the two configurations $(\mathtt{P}_k, \mathtt{P}_l, \{\mathbf{X}_j\})$ and $(\hat{\mathtt{P}}_k, \hat{\mathtt{P}}_l, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent (Sect. 4.4).
- If for two views k and l, $(\mathtt{P}_k, \mathtt{P}_l, \{\mathbf{X}_j\})$ and $(\hat{\mathtt{P}}_k, \hat{\mathtt{P}}_l, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent, then $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent (Sect. 4.5).

This completes the proof. □

Before stating the different parts of the proof, it is worth mentioning that for proving Theorem 1 one may simply assume that the set of true depths λ_{ij} are all equal to one. This can be seen by a simple change of variables $\mathbf{x}'_{ij} = \lambda_{ij}\mathbf{x}_{ij}$, $\lambda'_{ij} = 1$ and $\hat{\lambda}'_{ij} = \hat{\lambda}_{ij}/\lambda_{ij}$, implying $\lambda'_{ij}\mathbf{x}'_{ij} = \mathbf{x}'_{ij} = \mathtt{P}_i\mathbf{X}_j$

[4] For simplicity of notation, we are being a bit sloppy here about the projective entities like projective lines, quadric surfaces and twisted cubics. The reader must understand that when talking about a point $\mathbf{X} \in \mathbb{R}^4$ lying on a projective entity, what we really mean is that the projective point in $\mathbb{P}^3$ represented by $\mathbf{X}$ in homogeneous coordinates lies on them.

and $\hat{\lambda}'_{ij}\mathbf{x}'_{ij} = \hat{\mathtt{P}}_i\hat{\mathbf{X}}_j$. Notice that $\hat{\lambda}'_{ij} = \hat{\lambda}_{ij}/\lambda_{ij}$ is zero if and only if $\hat{\lambda}_{ij}$ is zero. Therefore, (D1–D3) are true for the $\hat{\lambda}'_{ij}$-s if and only if they hold for the $\hat{\lambda}_{ij}$-s. This change of variables requires $\lambda_{ij} \neq 0$ which was among the assumptions of the theorem (and even if it was not explicitly mentioned, it would be required as a consequence of $\mathtt{P}_i\mathbf{X}_j \neq \mathbf{0}$ from (G2-1) and the relations $\lambda_{ij}\mathbf{x}_{ij} = \mathtt{P}_i\mathbf{X}_j$). Throughout the proof of Theorem 1, we assume $\lambda_{ij} = 1$. With this assumption, the Eqs. (10) and (11) are combined into

$$\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j = \hat{\lambda}_{ij}\mathtt{P}_i\mathbf{X}_j. \qquad (12)$$

Theorem 1 is proved as a conjunction of several lemmas. Therefore, to avoid redundancy, we assume the following throughout the rest of this section:

There exist $m \geq 2$ camera matrices $\mathtt{P}_1, \mathtt{P}_2, \ldots, \mathtt{P}_m \in \mathbb{R}^{3\times4}$ and $n \geq 8$ points $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n \in \mathbb{R}^4$ (called the *true* sets of camera matrices and points, or the *ground truth*), and an *estimated* setup of m camera matrices and n points $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$, related by (12) for a set of scalars $\{\hat{\lambda}_{ij}\}$.

4.3 The Existence of a Nonzero Fundamental Matrix

The following lemma is the key to the proof of Theorem 1:

Lemma 6 *If the genericity assumptions (G1–G4) hold for* $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$*, and depth assumptions (D1–D3) hold for* $\{\hat{\lambda}_{ij}\}$*, there exist two views k and l such that the fundamental matrix* $\mathscr{F}(\hat{\mathtt{P}}_k, \hat{\mathtt{P}}_l)$ *is nonzero.*

Using Lemma 2, one can say that what is claimed in Lemma 6 is equivalent to the existence of an invertible 4×4 submatrix of $\mathrm{stack}(\hat{\mathtt{P}}_k, \hat{\mathtt{P}}_l)$ for some views k and l, made by choosing two rows from $\hat{\mathtt{P}}_k$ and two rows from $\hat{\mathtt{P}}_l$. This lemma is essential as the case of zero fundamental matrices for all pairs of views happens in the cross-shaped degenerate solutions. We will see later in Sect. 5 that a cross-shaped depth matrix $\hat{\Lambda}$ happens when for one special view r we have $\mathrm{rank}(\hat{\mathtt{P}}_r) = 3$ and for the rest of the views $i \neq r$ we have $\mathrm{rank}(\hat{\mathtt{P}}_i) = 1$. One can easily see from Lemma 2 that in this case all pairwise fundamental matrices are zero.

Lemma 6 is the hardest step in the proof of Theorem 1. We prove this lemma as a consequence of a series of lemmas. Figure 5 can help the reader to keep track of the inference process. The reader might notice that there are different ways of proving some of the lemmas here. Part of this is because the genericity conditions (G1–G4) are not tight. First, we state a lemma giving some simple facts about the second configuration of cameras, points and depths $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\}, \{\hat{\lambda}_{ij}\})$.

Lemma 7 *Under (G1, G2) and (D1, D2) The following hold*

(i) *For all j we have $\hat{\mathbf{X}}_j \neq \mathbf{0}$, and for all i we have $\hat{\mathtt{P}}_i \neq 0$,*
(ii) *$\hat{\lambda}_{ij} = 0$ if and only if $\hat{\mathbf{X}}_j \in \mathscr{N}(\hat{\mathtt{P}}_i)$, where $\mathscr{N}(\hat{\mathtt{P}}_i)$ is the null space of $\hat{\mathtt{P}}_i$.*

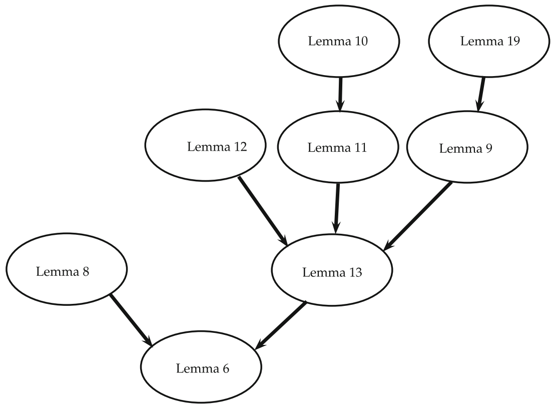

Fig. 5 The inference graph for the proof of Lemma 6. Lemma 7 has been omitted due to its frequent use

(iii) $\mathrm{rank}(\hat{\mathtt{P}}_i) \geq \min(3, n_i)$*, where n_i is the number of nonzero elements among* $\hat{\lambda}_{i1}, \hat{\lambda}_{i2}, \ldots, \hat{\lambda}_{in}$*,*
(iv) *If* $\mathrm{rank}(\mathrm{stack}(\hat{\mathtt{P}}_i, \hat{\mathtt{P}}_k)) = \mathrm{rank}(\hat{\mathtt{P}}_i) = 3$ *for two* distinct *views i, k, then for all j, $\hat{\lambda}_{ij} = 0$ implies $\hat{\lambda}_{kj} = 0$.*
(v) *If* $\mathrm{rank}(\hat{\mathtt{P}}_i) = 3$*, all the points $\hat{\mathbf{X}}_j$ for which $\hat{\lambda}_{ij} = 0$ are equal up to a nonzero scaling factor.*

Proof To see (i), notice that for any i and j if we have $\hat{\lambda}_{ij} \neq 0$, then from $\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j = \hat{\lambda}_{ij}\mathtt{P}_i\mathbf{X}_j$ and $\mathtt{P}_i\mathbf{X}_j \neq \mathbf{0}$ (G2-1) we conclude that $\hat{\mathbf{X}}_j \neq \mathbf{0}$ and $\hat{\mathtt{P}}_i \neq 0$. Then (i) follows from the fact that at each row and each column of $\hat{\Lambda} = [\hat{\lambda}_{ij}]$ there exists at least one nonzero element due to (D1, D2).

(ii) is obvious by $\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j = \hat{\lambda}_{ij}\mathtt{P}_i\mathbf{X}_j$ from (12) and the fact that $\mathtt{P}_i\mathbf{X}_j \neq \mathbf{0}$ from (G2-1).

To prove (iii), notice that if $\hat{\lambda}_{ij}$ is nonzero for some i and j, from $\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j = \hat{\lambda}_{ij}\mathtt{P}_i\mathbf{X}_j$ we conclude that $\mathtt{P}_i\mathbf{X}_j \in \mathscr{C}(\hat{\mathtt{P}}_i)$, where $\mathscr{C}(\hat{\mathtt{P}}_i)$ denotes the column space of $\hat{\mathtt{P}}_i$. Now, if there are n_i nonzero $\hat{\lambda}_{ij}$-s for view i, which (by a possible relabeling) we assume they are $\hat{\lambda}_{i1}, \hat{\lambda}_{i2}, \ldots, \hat{\lambda}_{in_i}$, then $\mathrm{span}(\mathtt{P}_i\mathbf{X}_1, \mathtt{P}_i\mathbf{X}_2, \ldots, \mathtt{P}_i\mathbf{X}_{n_i}) \subseteq \mathscr{C}(\hat{\mathtt{P}}_i)$. By (G2-5) then we have $\min(3, n_i) = \dim\left(\mathrm{span}(\mathtt{P}_i\mathbf{X}_1, \mathtt{P}_i\mathbf{X}_2, \ldots, \mathtt{P}_i\mathbf{X}_{n_i})\right) \leq \dim(\mathscr{C}(\hat{\mathtt{P}}_i)) = \mathrm{rank}(\hat{\mathtt{P}}_i)$.

To see (iv), notice that as $\mathrm{rank}(\hat{\mathtt{P}}_i) = 3$, if the matrix $\mathrm{stack}(\hat{\mathtt{P}}_i, \hat{\mathtt{P}}_k)$ has a rank of less than 4, the row space of $\hat{\mathtt{P}}_i$ includes that of $\hat{\mathtt{P}}_k$, that is $\mathscr{R}(\hat{\mathtt{P}}_k) \subseteq \mathscr{R}(\hat{\mathtt{P}}_i)$, and thus $\mathscr{N}(\hat{\mathtt{P}}_i) \subseteq \mathscr{N}(\hat{\mathtt{P}}_k)$. Hence, from part (ii) of the lemma we have $\hat{\lambda}_{ij} = 0 \Leftrightarrow \hat{\mathbf{X}}_j \in \mathscr{N}(\hat{\mathtt{P}}_i) \Rightarrow \hat{\mathbf{X}}_j \in \mathscr{N}(\hat{\mathtt{P}}_k) \Leftrightarrow \hat{\lambda}_{kj} = 0$.

(v) simply follows from parts (i) and (ii) of this lemma and the fact that a $\hat{\mathtt{P}}_i$ of rank 3 has a 1D null space. □

We make extensive use of Lemma 7 in what comes next. The reader might want to keep sight of it while reading the proofs.

Lemma 8 *Consider two 3×4 matrices $\mathtt{Q}$ and $\mathtt{R}$ such that* $\mathrm{rank}(\mathtt{Q}) \geq 2$ *and* $\mathrm{rank}(\mathtt{R}) \geq 2$*. Then* $\mathscr{F}(\mathtt{Q}, \mathtt{R}) \neq 0$ *if and only if* $\mathrm{stack}(\mathtt{Q}, \mathtt{R})$ *has rank 4.*

Proof Assume $\operatorname{stack}(\mathtt{Q}, \mathtt{R})$ has rank 4. If $\mathtt{R}$ and $\mathtt{Q}$ have both rank 3, then $\operatorname{stack}(\mathtt{Q}, \mathtt{R})$ having rank 4 means $\mathscr{N}(\mathtt{R}) \neq \mathscr{N}(\mathtt{Q})$. Geometrically, it means that $\mathtt{R}$ and $\mathtt{Q}$ are two rank-3 camera matrices with different camera centres. It is well known that in this case the fundamental matrix $\mathscr{F}(\mathtt{Q}, \mathtt{R})$ is nonzero (Hartley and Zisserman 2004).

If $\mathtt{R}$ has rank 2, it has two rows $\mathbf{r}_i^T$ and $\mathbf{r}_j^T$ spanning its row space, that is $\operatorname{span}(\mathbf{r}_i, \mathbf{r}_j) = \mathscr{R}(\mathtt{R})$. Further, as $\operatorname{stack}(\mathtt{Q}, \mathtt{R})$ has rank 4, there exist at least two rows $\mathbf{q}_k^T$ and $\mathbf{q}_l^T$ of $\mathtt{Q}$ such that $\dim(\operatorname{span}(\mathbf{r}_i, \mathbf{r}_j, \mathbf{q}_k, \mathbf{q}_l)) = 4$. The two rows $\mathbf{q}_k$ and $\mathbf{q}_l$ can be chosen by taking the set $\{\mathbf{r}_i, \mathbf{r}_j\}$, adding rows of $\mathtt{Q}$, one by one, to this set, and choose the two rows whose addition leads to a jump in the dimension the span of the vectors in the set. As, the 4×4 matrix $\operatorname{stack}(\mathbf{r}_i^T, \mathbf{r}_j^T, \mathbf{q}_k^T, \mathbf{q}_l^T)$ has rank 4, Lemma 2 suggests that $\mathscr{F}(\mathtt{Q}, \mathtt{R}) \neq 0$.

The other direction of the lemma is proved immediately from Lemma 2. □

Lemma 8 shows that to prove the main Lemma 6, it is sufficient to find two camera matrices both of rank 2 or more, whose vertical concatenation gives a matrix of rank 4. We will show in Lemma 13 that this is possible. But, to get there we need two extra lemmas. The next lemma relies on the Camera Resectioning Lemma discussed in Appendix 1.

Lemma 9 *Under (G1–G3), if for two distinct views k and l, there are at least $n-1$ indices j among the point indices $1, 2, \ldots, n$, for which the vector $(\hat{\lambda}_{kj}, \hat{\lambda}_{lj})$ is nonzero, we cannot have $\mathscr{R}(\hat{\mathtt{P}}_l) \subseteq \mathscr{R}(\hat{\mathtt{P}}_k)$, where $\mathscr{R}$ denotes the row space of a matrix.*

Proof To get a contradiction, assume $\mathscr{R}(\hat{\mathtt{P}}_l) \subseteq \mathscr{R}(\hat{\mathtt{P}}_k)$. Then there must exist a 3×3 matrix $\mathtt{H}$ such that $\hat{\mathtt{P}}_l = \mathtt{H}\hat{\mathtt{P}}_k$. Therefore, for all j we have $\hat{\mathtt{P}}_l\hat{\mathbf{X}}_j = \mathtt{H}\hat{\mathtt{P}}_k\hat{\mathbf{X}}_j$ and by the relation $\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j = \hat{\lambda}_{ij}\mathtt{P}_i\mathbf{X}_j$ we get $\hat{\lambda}_{lj}\mathtt{P}_l\mathbf{X}_j = \hat{\lambda}_{kj}\mathtt{H}\mathtt{P}_k\mathbf{X}_j$ for all j. Now, we can apply Lemma 19 on Camera Resectioning (see Appendix 1) as $(\hat{\lambda}_{kj}, \hat{\lambda}_{lj})$ is nonzero for at least $n-1$ indices j and (G1–G3) hold.[5] By applying Lemma 19 we get

$$\mathtt{H}\mathtt{P}_k = a\,\mathtt{P}_l. \tag{13}$$

for some scalar a. Now notice that $\mathtt{H} \neq 0$, as otherwise from $\hat{\mathtt{P}}_l = \mathtt{H}\hat{\mathtt{P}}_k$ we have $\hat{\mathtt{P}}_l = 0$, which is not possible due to Lemma 7i. As $\mathtt{H} \neq 0$ and $\mathtt{P}_k$ has full row rank according to (G1), then the scalar a in (13) cannot be zero. Therefore, we have

$$\mathtt{P}_l = \frac{1}{a}\mathtt{H}\mathtt{P}_k \tag{14}$$

[5] According to (G3) the $n-1$ points $\mathbf{X}_j$ corresponding to nonzero zero vectors $(\hat{\lambda}_{kj}, \hat{\lambda}_{lj})$ and the camera centre of $\mathtt{P}_l$ do not all lie on a twisted cubic. This is a generic property as $n-1 \geq 6$ (see Sect. 4.1). Notice that here the matrices $\mathtt{P}_l$ and $\mathtt{H}\mathtt{P}_k$ respectively act as $\mathtt{Q}$ and $\hat{\mathtt{Q}}$ in Lemma 19. The genericity conditions (G1–G3) provide the conditions (C1, C2) in Lemma 19.

meaning $\mathscr{R}(\mathtt{P}_l) \subseteq \mathscr{R}(\mathtt{P}_k)$. This possibility is excluded by (G1, G2-2) and hence we get a contradiction. This completes the proof. □

Lemma 10 *If (D1, D2) and (G1, G2) hold, then for at least one view i we have $\operatorname{rank}(\hat{\mathtt{P}}_i) \geq 2$.*

Proof To get a contradiction, assume that no matrix $\hat{\mathtt{P}}_i$ has rank 2 or more. As $\hat{\mathtt{P}}_i$-s are nonzero (Lemma 7i), we conclude that all $\hat{\mathtt{P}}_i$-s have rank 1. By (D2) and Lemma 7iii then each row of $\hat{\Lambda}$ must have exactly one nonzero element. Moreover, according to (D1), all columns of $\hat{\Lambda}$ have at least one nonzero element. These two facts imply that $m \geq n$ and that (by relabeling of the views) the rows of $\hat{\Lambda}$ can be permuted such that its top $n \times n$ block is a diagonal matrix $\mathtt{D}_{n\times n}$ with all nonzero diagonal elements, that is

$$\hat{\Lambda} = \begin{bmatrix} \mathtt{D}_{n\times n} \\ \mathtt{A} \end{bmatrix} \tag{15}$$

where $\mathtt{D}_{n\times n} = \operatorname{diag}(\hat{\lambda}_{11}, \hat{\lambda}_{22}, \ldots, \hat{\lambda}_{nn})$ and $\hat{\lambda}_{jj} \neq 0$ for all $j = 1, \ldots, n$. Using the relations $\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j = \hat{\lambda}_{ij}\mathtt{P}_i\mathbf{X}_j$, the above gives

$$\begin{bmatrix} \hat{\mathtt{P}}_1 \\ \hat{\mathtt{P}}_2 \\ \vdots \\ \hat{\mathtt{P}}_n \end{bmatrix} \begin{bmatrix} \hat{\mathbf{X}}_1 \, \hat{\mathbf{X}}_2 \, \ldots \, \hat{\mathbf{X}}_n \end{bmatrix} = \begin{bmatrix} \mathbf{v}_1 & & & \\ & \mathbf{v}_2 & & \\ & & \ddots & \\ & & & \mathbf{v}_n \end{bmatrix} \tag{16}$$

where the $3m \times n$ matrix on the right hand side is block-diagonal with nonzero diagonal blocks $\mathbf{v}_j = \hat{\lambda}_{jj}\mathtt{P}_j\mathbf{X}_j \neq 0$ (as $\hat{\lambda}_{jj} \neq 0$ and $\mathtt{P}_j\mathbf{X}_j \neq 0$ due to (G2-1)). This suggests that on the right hand side there is a matrix of rank n. On the other hand, the left hand side of (16) has rank 4 or less as $[\hat{\mathbf{X}}_1 \, \hat{\mathbf{X}}_2 \ldots \hat{\mathbf{X}}_n]$ is $4 \times n$. This is a contradiction since $n \geq 8$. □

Lemma 11 *If (D1, D2) and (G1, G2) hold, then for at least one view i we have $\operatorname{rank}(\hat{\mathtt{P}}_i) = 3$.*

Proof To get a contradiction, we assume that $\operatorname{rank}(\hat{\mathtt{P}}_i) \leq 2$ for all i. Consider an arbitrary view l. As $\operatorname{rank}(\hat{\mathtt{P}}_l) \leq 2$, by Lemma 7iii, we know that among $\hat{\lambda}_{l1}, \hat{\lambda}_{l2}, \ldots, \hat{\lambda}_{ln}$ at most two are nonzero. By relabeling the points $\{\mathbf{X}_j\}$ and accordingly $\{\hat{\mathbf{X}}_j\}$ if necessary, we can assume that $\hat{\lambda}_{l3} = \hat{\lambda}_{l4} = \cdots = \hat{\lambda}_{ln} = 0$. Now, by (D1), we know that the third column of $\hat{\Lambda}$ is not zero and therefore, there must be a view k for which $\hat{\lambda}_{k3} \neq 0$. Again, there are at most two nonzero projective depths among $\hat{\lambda}_{k1}, \ldots, \hat{\lambda}_{kn}$, and thus, at most one nonzero depths among $\hat{\lambda}_{k4}, \ldots, \hat{\lambda}_{kn}$. By relabeling the points $\mathbf{X}_4, \ldots, \mathbf{X}_n$ and accordingly $\hat{\mathbf{X}}_4, \ldots, \hat{\mathbf{X}}_n$, we can assume that $\hat{\lambda}_{k5} = \hat{\lambda}_{k6} = \cdots = \hat{\lambda}_{kn} = 0$. Notice that this relabeling retains $\hat{\lambda}_{l3} = \hat{\lambda}_{l4} = \cdots = \hat{\lambda}_{ln} = 0$.

Now, as $n \geq 8$, we can consider the points $\hat{\mathbf{X}}_5$, $\hat{\mathbf{X}}_6$ and $\hat{\mathbf{X}}_7$. If these points are equal up to scale, then by Lemma 7ii, for each view i, the depths $\hat{\lambda}_{i5}$, $\hat{\lambda}_{i6}$ and $\hat{\lambda}_{i7}$ are either all zero or all nonzero. In this case, by (D1), there must be a view i for which $\hat{\lambda}_{i5}$, $\hat{\lambda}_{i6}$ and $\hat{\lambda}_{i7}$ are all nonzero. But this means that $\text{rank}(\hat{\mathtt{P}}_i) = 3$ by Lemma 7iii, contradicting our assumption $\text{rank}(\hat{\mathtt{P}}_i) \leq 2$ for all i.

Thus, $\hat{\mathbf{X}}_5$, $\hat{\mathbf{X}}_6$ and $\hat{\mathbf{X}}_7$ are not equal up to scale, and therefore, the dimension of $\text{span}(\hat{\mathbf{X}}_5, \hat{\mathbf{X}}_6, \hat{\mathbf{X}}_7)$ is at least 2. As $\hat{\lambda}_{k3} \neq 0$ and $\hat{\lambda}_{k5} = \hat{\lambda}_{k6} = \hat{\lambda}_{k7} = 0$, by Lemma 7ii we have $\hat{\mathbf{X}}_3 \notin \mathcal{N}(\hat{\mathtt{P}}_k)$ and $\text{span}(\hat{\mathbf{X}}_5, \hat{\mathbf{X}}_6, \hat{\mathbf{X}}_7) \subseteq \mathcal{N}(\hat{\mathtt{P}}_k)$. This means that $\dim(\text{span}(\hat{\mathbf{X}}_3, \hat{\mathbf{X}}_5, \hat{\mathbf{X}}_6, \hat{\mathbf{X}}_7))$ is at least 3. Now, since $\hat{\lambda}_{l3} = \hat{\lambda}_{l5} = \hat{\lambda}_{l6} = \hat{\lambda}_{l7} = 0$, by Lemma 7ii, we can say $\text{span}(\hat{\mathbf{X}}_3, \hat{\mathbf{X}}_5, \hat{\mathbf{X}}_6, \hat{\mathbf{X}}_7) \subseteq \mathcal{N}(\hat{\mathtt{P}}_l)$. Since $\text{span}(\hat{\mathbf{X}}_3, \hat{\mathbf{X}}_5, \hat{\mathbf{X}}_6, \hat{\mathbf{X}}_7)$ is either 3D or 4D, this means that $\text{rank}(\hat{\mathtt{P}}_l) \leq 1$. As we chose l to be any arbitrary view, this means that $\text{rank}(\hat{\mathtt{P}}_i) \leq 1$ for all i. But according to Lemma 10 this cannot happen, and we get a contradiction. □

Lemma 12 *Assume that (D1, D2) and (G1, G2) hold, and denote by n_i the number of nonzero elements of the i-th row of $\hat{\Lambda}$. If for some view r we have $n_r \geq n - 1$ and $n_i = 1$ for all $i \neq r$, then the matrix $\hat{\Lambda}$ has to be* cross-shaped.

Proof As $m \geq 2$, there exist at least another view k other than r. Assume the (only) nonzero element on the k-th row of $\hat{\Lambda}$ is $\hat{\lambda}_{kc}$. We will show that for any view l other than r and k (if there is any) the only nonzero element in the l-th row of $\hat{\Lambda}$ has to be $\hat{\lambda}_{lc}$.

Consider a view l other than r and k. As $n \geq 8$, and there is exactly one nonzero element in the k-th row of $\hat{\Lambda}$, one nonzero element in the l-th row of $\hat{\Lambda}$, and at most one zero element in the r-th row of $\hat{\Lambda}$, one can find three distinct indices j_1, j_2, j_3 such that $\hat{\lambda}_{rj_1} \neq 0, \hat{\lambda}_{rj_2} \neq 0, \hat{\lambda}_{rj_3} \neq 0$, $\hat{\lambda}_{kj_1} = \hat{\lambda}_{kj_2} = \hat{\lambda}_{kj_3} = 0$ and $\hat{\lambda}_{lj_1} = \hat{\lambda}_{lj_2} = \hat{\lambda}_{lj_3} = 0$. We have

$$\begin{aligned}\hat{\mathtt{P}}_r\, \text{span}(\hat{\mathbf{X}}_{j_1}, \hat{\mathbf{X}}_{j_2}, \hat{\mathbf{X}}_{j_3}) &= \text{span}\Big(\hat{\mathtt{P}}_r\hat{\mathbf{X}}_{j_1}, \hat{\mathtt{P}}_r\hat{\mathbf{X}}_{j_2}, \hat{\mathtt{P}}_r\hat{\mathbf{X}}_{j_3}\Big) \\ &= \text{span}\Big(\mathtt{P}_r\mathbf{X}_{j_1}, \mathtt{P}_r\mathbf{X}_{j_2}, \mathtt{P}_r\mathbf{X}_{j_3}\Big). \quad (17)\end{aligned}$$

where the product $\hat{\mathtt{P}}_r\, \text{span}(\hat{\mathbf{X}}_{j_1}, \hat{\mathbf{X}}_{j_2}, \hat{\mathbf{X}}_{j_3})$ represents the set created by multiplying $\hat{\mathtt{P}}_r$ by each element of the subspace $\text{span}(\hat{\mathbf{X}}_{j_1}, \hat{\mathbf{X}}_{j_2}, \hat{\mathbf{X}}_{j_3})$. The last equality in (17) comes from (12) and the fact that $\hat{\lambda}_{rj_1}$, $\hat{\lambda}_{rj_2}$ and $\hat{\lambda}_{rj_3}$ are nonzero. According to (G2-5), $\text{span}(\mathtt{P}_r\mathbf{X}_{j_1}, \mathtt{P}_r\mathbf{X}_{j_2}, \mathtt{P}_r\mathbf{X}_{j_3})$ is 3D, and therefore, (17) suggests that $\text{span}(\hat{\mathbf{X}}_{j_1}, \hat{\mathbf{X}}_{j_2}, \hat{\mathbf{X}}_{j_3})$ has to be also 3D. From $\hat{\lambda}_{kj_1} = \hat{\lambda}_{kj_2} = \hat{\lambda}_{kj_3} = 0$ and $\hat{\lambda}_{lj_1} = \hat{\lambda}_{lj_2} = \hat{\lambda}_{lj_3} = 0$ respectively we conclude that $\text{span}(\hat{\mathbf{X}}_{j_1}, \hat{\mathbf{X}}_{j_2}, \hat{\mathbf{X}}_{j_3}) \in \mathcal{N}(\hat{\mathtt{P}}_k)$ and $\text{span}(\hat{\mathbf{X}}_{j_1}, \hat{\mathbf{X}}_{j_2}, \hat{\mathbf{X}}_{j_3}) \in \mathcal{N}(\hat{\mathtt{P}}_l)$ (Lemma 7ii). As $\hat{\mathtt{P}}_k$ and $\hat{\mathtt{P}}_l$ are both nonzero (Lemma 7i), and hence, of rank one or more, and their null-spaces include a the 3D subspace $\text{span}(\hat{\mathbf{X}}_{j_1}, \hat{\mathbf{X}}_{j_2}, \hat{\mathbf{X}}_{j_3})$, it follows that $\mathcal{N}(\hat{\mathtt{P}}_k) = \mathcal{N}(\hat{\mathtt{P}}_l) = \text{span}(\hat{\mathbf{X}}_{j_1}, \hat{\mathbf{X}}_{j_2}, \hat{\mathbf{X}}_{j_3})$. This means that for any j, $\hat{\lambda}_{kj}$ and $\hat{\lambda}_{lj}$ are either both nonzero or both zero. As $\hat{\lambda}_{kc} \neq 0$, we must have $\hat{\lambda}_{lc} \neq 0$. Since this is true for any view l other than k and r, we can say that for all views $i \neq r$, the (only) nonzero element is in the c-th column of $\hat{\lambda}_{ic}$.

By the assumption of the lemma, the r-th row of $\hat{\Lambda}$ can have either no zero element or one zero element. If it does have one zero element, it has to be $\hat{\lambda}_{rc}$, as otherwise, if $\hat{\lambda}_{rc'} = 0$ for some $c' \neq c$, the c'-th column of $\hat{\Lambda}$ would be zero, violating (D1). Now, we have the case where all elements of $\hat{\Lambda}$ are zero except those in the r-th row or the c-th column, and among the elements in the r-th row or the c-th column, all are nonzero except possibly $\hat{\lambda}_{rc}$. This means that $\hat{\Lambda}$ is cross-shaped. □

Lemma 13 *Under (D1–D3), (G1–G3) there exist two views i and k such that $\text{rank}(\hat{\mathtt{P}}_i) \geq 2$, $\text{rank}(\hat{\mathtt{P}}_k) \geq 2$ and $\text{stack}(\hat{\mathtt{P}}_i, \hat{\mathtt{P}}_k)$ has rank 4.*

Proof Lemma 11 says that under our assumptions, there exists at least one estimated camera matrix $\hat{\mathtt{P}}_i$ of rank 3. With a possible re-indexing of the views, we can assume that $\text{rank}(\hat{\mathtt{P}}_1) = 3$. Now we consider two cases. The first case is when among $\hat{\lambda}_{11}, \hat{\lambda}_{12}, \ldots, \hat{\lambda}_{1n}$ there exists at most one zero element. In this case there must be at least another view k with two or more nonzero elements in the corresponding row of $\hat{\Lambda}$, as otherwise, according to Lemma 12, $\hat{\Lambda}$ would be cross-shaped, violating (D3). By Lemma 7iii then we have $\text{rank}(\hat{\mathtt{P}}_k) \geq 2$. Because at least for $n - 1$ point indices j we have $\hat{\lambda}_{1j} \neq 0$, and thus $(\hat{\lambda}_{1j},\ \hat{\lambda}_{kj})^T \neq \mathbf{0}$, from Lemma 9 we know that the row space of $\hat{\mathtt{P}}_k$ cannot be a subset of the row space of $\hat{\mathtt{P}}_1$. Therefore, as $\text{rank}(\hat{\mathtt{P}}_1) = 3$ we have $\text{rank}\begin{bmatrix}\hat{\mathtt{P}}_1\\ \hat{\mathtt{P}}_k\end{bmatrix} = 4$. This along with the fact that $\text{rank}(\hat{\mathtt{P}}_1) = 3 \geq 2$ and $\text{rank}(\hat{\mathtt{P}}_k) \geq 2$ completes the proof for this case.

The only case left is when there are at least two zero elements among $\hat{\lambda}_{11}, \hat{\lambda}_{12}, \ldots, \hat{\lambda}_{1n}$. By a possible re-indexing we can assume that $\hat{\lambda}_{11} = \hat{\lambda}_{12} = 0$. From Lemma 7(v) it follows that $\hat{\mathbf{X}}_1$ and $\hat{\mathbf{X}}_2$ must be equal up to scale. According to (D1), there must be at least one view k for which $\hat{\lambda}_{k1} \neq 0$. As $\hat{\mathbf{X}}_1$ and $\hat{\mathbf{X}}_2$ are nonzero (Lemma 7i) and equal up to scale, $\hat{\lambda}_{k1} \neq 0$ implies $\hat{\lambda}_{k2} \neq 0$ (by Lemma 7ii). This means that $\text{rank}(\hat{\mathtt{P}}_k) \geq 2$ (Lemma 7iii). As we have $\text{rank}(\hat{\mathtt{P}}_1) = 3$, $\hat{\lambda}_{11} = 0$ and $\hat{\lambda}_{k1} \neq 0$, by Lemma 7iv we get $\text{rank}\begin{bmatrix}\hat{\mathtt{P}}_1\\ \hat{\mathtt{P}}_k\end{bmatrix} = 4$. This completes the proof as we also have $\text{rank}(\hat{\mathtt{P}}_1) \geq 2$ and $\text{rank}(\hat{\mathtt{P}}_k) \geq 2$. □

Lemma 6 now follows directly from Lemmas 13 and 8.

4.4 Projective Equivalence for Two Views

The main result of this section is the following lemma:

Lemma 14 *Under (G1, G2, G4) and (D1), If the fundamental matrix $\mathcal{F}(\hat{\mathtt{P}}_k, \hat{\mathtt{P}}_l)$ is nonzero for two views k and l, then the two configurations $(\mathtt{P}_k, \mathtt{P}_l, \{\mathbf{X}_j\})$ and $(\hat{\mathtt{P}}_k, \hat{\mathtt{P}}_l, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent.*

Proof For simplicity, we take $k = 1$ and $l = 2$. The other cases follow by relabeling the views. For each j we have $\hat{\mathtt{P}}_1\hat{\mathbf{X}}_j = \hat{\lambda}_{1j}\mathtt{P}_1\mathbf{X}_j$ and $\hat{\mathtt{P}}_2\hat{\mathbf{X}}_j = \hat{\lambda}_{2j}\mathtt{P}_2\mathbf{X}_j$, or equivalently

$$\begin{bmatrix} \hat{\mathtt{P}}_1, & \mathtt{P}_1\mathbf{X}_j & \mathbf{0} \\ \hat{\mathtt{P}}_2, & \mathbf{0} & \mathtt{P}_2\mathbf{X}_j \end{bmatrix} \begin{pmatrix} -\hat{\mathbf{X}}_j \\ \hat{\lambda}_{1j} \\ \hat{\lambda}_{2j} \end{pmatrix} = 0, \quad j = 1, 2, \ldots, n. \quad (18)$$

As, $\hat{\mathbf{X}}_j \neq 0$ (Lemma 7i) the 6×6 matrix on the left hand side of (18) has a nontrivial null space and hence a vanishing determinant. Define the function $\mathscr{S}\colon \mathbb{R}^4 \to \mathbb{R}$ as

$$\mathscr{S}(\mathbf{X}) \overset{\text{def}}{=} \det \begin{bmatrix} \hat{\mathtt{P}}_1, & \mathtt{P}_1\mathbf{X} & \mathbf{0} \\ \hat{\mathtt{P}}_2, & \mathbf{0} & \mathtt{P}_2\mathbf{X} \end{bmatrix}. \quad (19)$$

Using the properties of the determinant and Definition 4 of the fundamental matrix, the above can be written as (Hartley and Zisserman, 2004, Sect.17.1):

$$\mathscr{S}(\mathbf{X}) = \mathbf{X}^T\mathtt{P}_1^T\,\hat{\mathtt{F}}_{12}\,\mathtt{P}_2\mathbf{X} = \mathbf{X}^T\,\mathtt{S}\,\mathbf{X} \quad (20)$$

where $\hat{\mathtt{F}}_{12} \overset{\text{def}}{=} \mathscr{F}(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_2)$ is the fundamental matrix of $\hat{\mathtt{P}}_1$ and $\hat{\mathtt{P}}_2$ as defined in Definition 4, and $\mathtt{S} \overset{\text{def}}{=} \mathtt{P}_1^T\,\hat{\mathtt{F}}_{12}\,\mathtt{P}_2$. We shall show that $\mathscr{S}(\mathbf{X})$ has to be identically zero. To see this, assume that $\mathscr{S}(\mathbf{X})$ is not identically zero. Then the equation

$$\mathscr{S}(\mathbf{X}) = \mathbf{X}^T\,\mathtt{S}\,\mathbf{X} = 0 \quad (21)$$

defines a quadric surface. From (18) we know $\mathscr{S}(\mathbf{X}_j) = 0$ for all $j = 1, 2, \ldots, n$ and therefore all the points $\{\mathbf{X}_j\}$ lie on this quadric surface. Also, for any pair of nonzero vectors $\mathbf{C}_1 \in \mathcal{N}(\mathtt{P}_1)$ and $\mathbf{C}_2 \in \mathcal{N}(\mathtt{P}_2)$ (camera centres) one can easily check that $\mathscr{S}(\mathbf{C}_1) = \mathscr{S}(\mathbf{C}_2) = 0$ and therefore, $\mathbf{C}_1$ and $\mathbf{C}_2$ also lie on the quadric surface.

As the fundamental matrix $\hat{\mathtt{F}}_{12} \overset{\text{def}}{=} \mathscr{F}(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_2)$ is rank deficient (Hartley and Zisserman 2004), we can have a nonzero vector $\mathbf{v} \in \mathcal{N}(\hat{\mathtt{F}}_{12})$. Since $\mathtt{P}_2$ has full row rank by (G1), we can write $\mathbf{v} = \mathtt{P}_2\mathbf{Y}$ for some $\mathbf{Y} \in \mathbb{R}^4$. Then, by taking a nonzero vector $\mathbf{C}_2 \in \mathcal{N}(\mathtt{P}_2)$, one can easily check that for any two scalars α and β we have $\mathscr{S}(\alpha\mathbf{Y} + \beta\mathbf{C}_2) = 0$. This, plus the fact that $\mathbf{Y}$ and $\mathbf{C}_2$ are linearly independent (as $\mathtt{P}_2\mathbf{C}_2 = \mathbf{0} \neq \mathbf{v} = \mathtt{P}_2\mathbf{Y}$), implies that the quadric surface $\mathscr{S}(\mathbf{X}) = 0$ contains a projective line and hence is *ruled*.

Now, we have the case that the nonzero vectors $\mathbf{C}_1 \in \mathcal{N}(\mathtt{P}_1)$ and $\mathbf{C}_2 \in \mathcal{N}(\mathtt{P}_2)$ (camera centres) plus the points $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n$ all lie on a (proper or degenerate) ruled quadric surface represented by (21). This contradicts the genericity condition (G4). This only leaves the possibility that $\mathscr{S}(\mathbf{X})$ is identically zero or equivalently, $\mathtt{S} + \mathtt{S}^T = \mathtt{0}$, that is

$$\mathtt{P}_1^T\,\hat{\mathtt{F}}_{12}\,\mathtt{P}_2 + \mathtt{P}_2^T\,\hat{\mathtt{F}}_{12}^T\,\mathtt{P}_1 = \mathtt{0} \quad (22)$$

Therefore, according to Lemma 4 (whose conditions hold by (G1) and (G2-2)) the matrix $\hat{\mathtt{F}}_{12} = \mathscr{F}(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_2)$ is a multiple of $\mathscr{F}(\mathtt{P}_1, \mathtt{P}_2)$. As we have assumed that $\mathscr{F}(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_2) \neq \mathtt{0}$, and having (G1) and (G2-2), by Lemma 3 we know that $(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_2)$ is projectively equivalent to $(\mathtt{P}_1, \mathtt{P}_2)$. As (G2-4) holds, using the Triangulation Lemma 18 (see Appendix 1) we can prove that $(\mathtt{P}_1, \mathtt{P}_2, \{\mathbf{X}_j\})$ and $(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_2, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent.[6] □

4.5 Projective Equivalence for All Views

Lemma 15 *Under (G1–G4) and (D1, D2), if for two views k and l the two configurations $(\mathtt{P}_k, \mathtt{P}_l, \{\mathbf{X}_j\})$ and $(\hat{\mathtt{P}}_k, \hat{\mathtt{P}}_l, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent, then for the whole camera matrices and points, the configurations $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent.*

Proof For convenience, take $k = 1$ and $l = 2$ (the other cases follow by relabeling the views). First of all, notice that as $(\mathtt{P}_1, \mathtt{P}_2, \{\mathbf{X}_j\})$ and $(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_2, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent, we have

$$\hat{\mathtt{P}}_1 = \tau_1\mathtt{P}_1\mathtt{H}, \quad \hat{\mathtt{P}}_2 = \tau_2\mathtt{P}_2\mathtt{H}, \quad (23)$$

$$\hat{\mathbf{X}}_j = \nu_j\mathtt{H}^{-1}\mathbf{X}_j, \quad j = 1, 2, \ldots, n, \quad (24)$$

for an invertible matrix $\mathtt{H}$ and nonzero scalars τ_1, τ_2 and $\nu_1, \ldots, \nu_n$. From (G2) and (24), we can say that for any four distinct point indices $j_1, \ldots, j_4$, the points $\hat{\mathbf{X}}_{j_1}$, $\hat{\mathbf{X}}_{j_2}$, $\hat{\mathbf{X}}_{j_3}$ and $\hat{\mathbf{X}}_{j_4}$ span a 4-dimensional space. Therefore, for each view i at most 3 depth scalars $\hat{\lambda}_{ij}$ can be zero, as otherwise, if we have $\hat{\lambda}_{ij_1} = \hat{\lambda}_{ij_2} = \hat{\lambda}_{ij_3} = \hat{\lambda}_{ij_4} = 0$ it means that $\hat{\mathbf{X}}_{j_1}, \hat{\mathbf{X}}_{j_2}, \hat{\mathbf{X}}_{j_3}, \hat{\mathbf{X}}_{j_4} \in \mathcal{N}(\hat{\mathtt{P}}_i)$ (Lemma 7ii). This, however, implies $\hat{\mathtt{P}}_i = \mathtt{0}$ contradicting Lemma 7i.

Now, since we know that for each view i we have at most 3 zero depths $\hat{\lambda}_{ij}$, from $n \geq 8$, we know that there are more than 3 nonzero depths $\hat{\lambda}_{ij}$ at each row i. Therefore, according to Lemma 7iii, we can say that $\text{rank}(\hat{\mathtt{P}}_i) = 3$ for all i.

Now, notice that as $(\mathtt{P}_1, \mathtt{P}_2, \{\mathbf{X}_j\})$ and $(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_2, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent, from Lemma 1 (whose conditions hold by (G1, G2) and their consequences (G2-1) and (G2-3)) we have $\hat{\lambda}_{1j} \neq 0$ and $\hat{\lambda}_{2j} \neq 0$ for all $j = 1, 2, \ldots, n$. Now, for any view $k \geq 3$, consider the pair of matrices $(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_k)$. We have $\text{rank}(\hat{\mathtt{P}}_k) = \text{rank}(\hat{\mathtt{P}}_1) = 3$ and moreover, the vector $(\hat{\lambda}_{1j}, \hat{\lambda}_{kj})$ is nonzero for all j. Therefore, by Lemma 9 we get $\text{rank}\left(\text{stack}(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_k)\right) = 4$. After that, by Lemma 13 it follows that the fundamental matrix $\mathscr{F}(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_k)$ is nonzero. Then by Lemma 14 we can say that $(\mathtt{P}_1, \mathtt{P}_k, \{\mathbf{X}_j\})$ and $(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_k, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent. Therefore,

$$\hat{\mathtt{P}}_1 = \tau_1'\mathtt{P}_1\mathtt{G}, \quad \hat{\mathtt{P}}_k = \tau_k'\mathtt{P}_k\mathtt{G}, \quad (25)$$

$$\hat{\mathbf{X}}_j = \nu_j'\mathtt{G}^{-1}\mathbf{X}_j, \quad j = 1, 2, \ldots, n, \quad (26)$$

for an invertible matrix $\mathtt{G}$ and nonzero scalars τ_1', τ_k' and $\nu_1', \nu_2', \ldots, \nu_n'$. From (24) and (26) we have

$$\mathtt{G}\mathtt{H}^{-1}\mathbf{X}_j = \frac{\nu_j'}{\nu_j}\mathbf{X}_j = \alpha_j\mathbf{X}_j \quad (27)$$

[6] This can be done similarly to (Hartley and Zisserman, 2004, Theorem 10.1), so we do not repeat it here.

where $\alpha_j \overset{\text{def}}{=} \nu'_j/\nu_j$. This says that the points $\mathbf{X}_1, \ldots, \mathbf{X}_n$ are eigenvectors of $\mathtt{G}\mathtt{H}^{-1}$. Consider $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_5$. They are all eigenvectors of the 4×4 matrix $\mathtt{G}\mathtt{H}^{-1}$, each associated with a nonzero eigenvalue $\alpha_j = \nu'_j/\nu_j$, and according to (G2), no four eigenvectors among them are linearly dependent. This can only happen when $\mathtt{G}\mathtt{H}^{-1} \in \mathbb{R}^{4\times 4}$ has a 4D eigenspace, and therefore, all eigenvalues of $\mathtt{G}\mathtt{H}^{-1}$, including α_j-s, are equal to a common nonzero scalar α. Thus, we must have $\mathtt{G}\mathtt{H}^{-1} = \alpha\mathtt{I}$ or $\mathtt{G} = \alpha\mathtt{H}$. This, plus (23) and (25) gives $\tau_1 = \alpha\tau'_1$. By using $\alpha = \alpha_j = \nu'_j/\nu_j$ and $\tau_1 = \alpha\tau'_1$, and defining $\tau_k \overset{\text{def}}{=} \alpha\tau'_k$ we have

$$\hat{\mathtt{P}}_1 = \tau_1\mathtt{P}_1\mathtt{H}, \quad \hat{\mathtt{P}}_k = \tau_k\mathtt{P}_k\mathtt{H}, \tag{28}$$

$$\hat{\mathbf{X}}_j = \nu_j\mathtt{H}^{-1}\mathbf{X}_j, \quad j = 1, 2, \ldots, n, \tag{29}$$

Since the above is true for all $k = 3, \ldots, n$, and also for $k = 2$ by (23), we conclude that the two configurations $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent. □

5 Minimality of (D1–D3) and Cross-shaped Configurations

From depth assumptions (D1–D3) we see that in order to get the projective reconstruction working we require that none of the rows or columns of the depth matrix $\hat{\Lambda} = [\hat{\lambda}_{ij}]$ are zero and that $\hat{\Lambda}$ is not cross-shaped. One might wonder whether projective reconstruction is possible under weaker constraints on the estimated depth matrix. For example, what happens if we just require that the matrix has no zero rows and no zero columns.

In this section we shall show that, in some specific sense, (D1–D3) is a minimal assumption for projective reconstruction. However, by this we do not mean that it is the weakest possible constraint that guarantees the uniqueness of projective reconstruction up to projectivity. But, it is minimal in the sense that if any of (D1), (D2) or (D3) is removed, and no extra conditions are added, the resulting constraints cannot rule out false solutions to projective reconstruction. This shows that the false solutions to the factorization problem $\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\,\hat{\mathtt{X}}$ are not limited to the trivial cases of having depth matrices with some zero rows or columns.

It is trivial to demonstrate degenerate solutions created by violating (D1). For example, we can set $\hat{\lambda}_{1k} = \hat{\lambda}_{2k} = \ldots = \hat{\lambda}_{mk} = 0$ and $\hat{\mathbf{X}}_k = \mathbf{0}$, as it satisfies $\hat{\mathtt{P}}_i\hat{\mathbf{X}}_k = \hat{\lambda}_{ik}\mathbf{x}_{ik}$. For the rest of variables we can have $\hat{\mathtt{P}}_i = \mathtt{P}_i$ for all i and $\hat{\mathbf{X}}_j = \mathbf{X}_j$ and $\hat{\lambda}_{ij} = \lambda_{ij}$ for all $j \neq k$. Similarly, if we relax (D2) by allowing the l-th row of $\hat{\Lambda}$ to be nonzero, we can have a configuration in which $\hat{\mathtt{P}}_l = \mathtt{0}$.

The more difficult job is to show that the relaxation of (D3) allows a projectively non-equivalent setup. Relaxing this condition means that $\hat{\Lambda}$ is *cross-shaped*. We show that in this case for any configuration of the true camera matrices $\mathtt{P}_i$, points $\mathbf{X}_j$ and depths λ_{ij}, we can find a non-equivalent setup $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\}, \{\hat{\lambda}_{ij}\})$ satisfying the projection equations.

Consider m arbitrary 3×4 projection matrices $\mathtt{P}_1, \mathtt{P}_2, \ldots, \mathtt{P}_m$ and an arbitrary set of points $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n \in \mathbb{R}^4$ (with m and n arbitrary), giving the image points $\mathbf{x}_{ij}$ through the relation $\lambda_{ij}\mathbf{x}_{ij} = \mathtt{P}_i\mathbf{X}_j$ for nonzero scalars $\hat{\lambda}_{ij}$. Now, for any arbitrary view r and point index c we can take

$$\hat{\lambda}_{ic} = \lambda_{ic}, \quad i = 1, 2, \ldots, m, \tag{30}$$

$$\hat{\lambda}_{rj} = \lambda_{rj}, \quad j = 1, 2, \ldots, n, \tag{31}$$

$$\hat{\lambda}_{ij} = 0, \quad i \neq r,\ j \neq c. \tag{32}$$

$$\hat{\mathtt{P}}_r = \mathtt{P}_r, \tag{33}$$

$$\hat{\mathtt{P}}_i = \mathtt{P}_i\mathbf{X}_c\bar{\mathbf{C}}_r^T \quad i \neq r \tag{34}$$

$$\hat{\mathbf{X}}_c = \left(\mathtt{I} - \bar{\mathbf{C}}_r\bar{\mathbf{C}}_r^T\right)\mathbf{X}_c + \bar{\mathbf{C}}_r, \tag{35}$$

$$\hat{\mathbf{X}}_j = \left(\mathtt{I} - \bar{\mathbf{C}}_r\bar{\mathbf{C}}_r^T\right)\mathbf{X}_j \quad j \neq c. \tag{36}$$

where $\bar{\mathbf{C}}_r$ is the normalized camera centre of $\mathtt{P}_r$ (a unit vector in the null-space of $\mathtt{P}_r$). Notice that the matrix $\mathtt{I} - \bar{\mathbf{C}}_r\bar{\mathbf{C}}_r^T$ is the orthogonal projection onto the row space of $\mathtt{P}_r$. Now, it can be easily checked that

$$\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j = \mathtt{P}_i\mathbf{X}_j = \lambda_{ij}\mathbf{x}_{ij} = \hat{\lambda}_{ij}\mathbf{x}_{ij} \quad \text{if } i = r \text{ or } j = c \tag{37}$$

$$\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j \ = 0 = 0 \cdot \mathbf{x}_{ij} = \hat{\lambda}_{ij}\mathbf{x}_{ij} \quad \text{if } i \neq r \text{ and } j \neq c \tag{38}$$

Notice that to derive (37) one has to check three cases separately: first $i = r, j = c$, second $i = r, j \neq c$, and third $i \neq r, j = c$. You can see that with this choice we have $\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j = \hat{\lambda}_{ij}\mathbf{x}_{ij}$ for all i and j. It is obvious that $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ is not projectively equivalent to $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$, as, for example, for any $i \neq r$ we have $\text{rank}(\hat{\mathtt{P}}_i) = 1$ regardless of the value of $\mathtt{P}_i$. From (30–32) it follows that

$$\hat{\Lambda} = \begin{bmatrix} \mathtt{0} & \mathbf{1}_{r-1} & \mathtt{0} \\ \mathbf{1}_{c-1}^T & 1 & \mathbf{1}_{n-c}^T \\ \mathtt{0} & \mathbf{1}_{m-r} & \mathtt{0} \end{bmatrix} \circ \Lambda \tag{39}$$

where the zero matrices denoted by $\mathtt{0}$ are of compatible size and $\circ$ denotes the Hadamard (element-wise) product. This shows that $\hat{\Lambda} = [\hat{\lambda}_{ij}]$ is a *nonzero-centred* cross-shaped matrix centred at (r, c), according to Definition 5.

One can observe that instead of (35) we can give any arbitrary value to $\hat{\mathbf{X}}_c$, provided that it is not perpendicular to $\mathbf{C}_r$, and still get a setup with a cross-shaped depth matrix. Especially, we leave it to the reader to check that by taking $\hat{\mathbf{X}}_c$ equal to $\bar{\mathbf{C}}_r$ instead of $(\mathtt{I} - \bar{\mathbf{C}}_r\bar{\mathbf{C}}_r^T)\mathbf{X}_c + \bar{\mathbf{C}}_r$ in (35), we have a setup in which the depth matrix $\hat{\Lambda}$ is arranged as (30–32) with the exception that the central element $\hat{\lambda}_{rc}$ is zero, that is

$$\hat{\Lambda} = \begin{bmatrix} \mathtt{0} & \mathbf{1}_{r-1} & \mathtt{0} \\ \mathbf{1}_{c-1}^T & 0 & \mathbf{1}_{n-c}^T \\ \mathtt{0} & \mathbf{1}_{m-r} & \mathtt{0} \end{bmatrix} \circ \Lambda. \tag{40}$$

This means that $\hat{\Lambda}$ is a zero-centred cross-shaped matrix. Fig. 2 illustrates such a solution for the case of $r = 1, c = 1$.

Obviously for any pair of vectors $\boldsymbol{\tau} \in \mathbb{R}^m$ and $\boldsymbol{\nu} \in \mathbb{R}^n$ with all nonzero entries, we can find a new configuration with $\hat{\Lambda}' = \operatorname{diag}(\boldsymbol{\tau})\,\hat{\Lambda}\,\operatorname{diag}(\boldsymbol{\nu})$, $\hat{\mathtt{P}}'_i = \tau_i \hat{\mathtt{P}}_i$ and $\hat{\mathbf{X}}'_j = \nu_j \hat{\mathbf{X}}_j$, satisfying $\hat{\mathtt{P}}'_i \hat{\mathbf{X}}'_j = \hat{\lambda}'_{ij} \mathbf{x}_{ij}$ (as $(\tau_i \hat{\mathtt{P}}_i)(\nu_j \hat{\mathbf{X}}_j) = (\tau_i \nu_j \hat{\lambda}_{ij})\,\mathbf{x}_{ij}$). Notice that, according to the above discussion, both configurations (39) and (40) can be obtained for any configuration of m views and n points, and for any choice of r and c. We also know from Lemma 5 that any $m \times n$ cross-shaped matrix is diagonally equivalent to either (39) or (40) for some choice of r and c. Putting all these together we get the following lemma.

Lemma 16 *Consider any configuration of m camera matrices and n points* $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ *giving the image points* $\{\mathbf{x}_{ij}\}$ *through the relations* $\lambda_{ij}\mathbf{x}_{ij} = \mathtt{P}_i \mathbf{X}_j$ *with nonzero scalars* $\lambda_{ij} \neq 0$. *Then for* any *cross-shaped matrix* $\hat{\Lambda} = [\hat{\lambda}_{ij}]$, *there exists a configuration* $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$, *such that the relation* $\hat{\lambda}_{ij}\mathbf{x}_{ij} = \hat{\mathtt{P}}_i \hat{\mathbf{X}}_j$ *holds for all* $i = 1, \ldots, m$ *and* $j = 1, \ldots, n$.

This lemma is used in the next session as a useful test for the assessment of depth constraints. It says that if a constraint allows any cross-shaped structure for the depth matrix, then it allows for a false solution.

6 The Constraint Space

In this section we will have a closer look at the depth constraints used in factorization-based projective reconstruction. Consider a set of $m \geq 2$ projection matrices $\mathtt{P}_1, \ldots, \mathtt{P}_m \in \mathbb{R}^{3\times 4}$ and a set of $n \geq 8$ points $\mathbf{X}_1, \ldots, \mathbf{X}_n \in \mathbb{R}^4$, generically configured in the sense of (G1-G4) and projecting into a set of image points $\mathbf{x}_{ij} \in \mathbb{R}^3$ according to $\lambda_{ij}\mathbf{x}_{ij} = \mathtt{P}_i \mathbf{X}_j$. Given a constraint space $C \subseteq \mathbb{R}^{m\times n}$ we want to assess the solutions to the problem

$$\text{find}_{\hat{\Lambda},\ \hat{\mathtt{P}}_{3m\times 4},\ \hat{\mathtt{X}}_{4\times n}} \quad \text{s.t.} \quad \hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\,\hat{\mathtt{X}}, \quad \hat{\Lambda} \in C \tag{41}$$

in terms of whether $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ is projectively equivalent to $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$, where $\hat{\mathtt{P}} = \operatorname{stack}(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_2, \cdots, \hat{\mathtt{P}}_m)$, $\hat{\mathtt{X}} = [\hat{\mathbf{X}}_1 \hat{\mathbf{X}}_2 \cdots \hat{\mathbf{X}}_n]$ and $\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\,\hat{\mathtt{X}}$ represents all the relations $\hat{\lambda}_{ij}\mathbf{x}_{ij} = \hat{\mathtt{P}}_i \hat{\mathbf{X}}_j$ in matrix form, as described for (2) and (3). By $\hat{\mathtt{P}}_{3m\times 4}$ and $\hat{\mathtt{X}}_{4\times n}$ we respectively mean $\hat{\mathtt{P}} \in \mathbb{R}^{3m\times 4}$ and $\hat{\mathtt{X}} \in \mathbb{R}^{4\times n}$.

Notice that, it is not sufficient that every $\hat{\Lambda}$ in C satisfies depth assumptions (D1–D3). The constraint space must also be *inclusive*, that is, it must make possible the existence of $\{\hat{\mathtt{P}}_i\}$ and $\{\hat{\mathbf{X}}_j\}$ for which $\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\,\hat{\mathtt{X}}$ holds for all i and j. In other words, it must guarantee that (41) has at least one solution. One can check that for any $\hat{\Lambda}$ diagonally equivalent to the true depth matrix Λ, there exists a setup $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$, defined by $\hat{\mathtt{P}}_i = \tau_i \mathtt{P}_i$, $\hat{\mathbf{X}}_j = \nu_j \mathbf{X}_j$, which is projectively equivalent to $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and satisfies the relation $\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\,\hat{\mathtt{X}}$. Therefore, for (41) to have at least one solution, it is sufficient that the constraint space C allows at least one $\hat{\Lambda}$ which is diagonally equivalent to Λ. Actually, this requirement is also necessary, since, according to Lemma 1, if there exists a setup $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ projectively equivalent to $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ which satisfies the relations $\hat{\lambda}_{ij}\mathbf{x}_{ij} = \hat{\mathtt{P}}_i \hat{\mathbf{X}}_j$, then $\hat{\Lambda}$ must be diagonally equivalent to Λ. As we do not know the true depths Λ beforehand, we would like the constraint $\hat{\Lambda} \in C$ to work for any initial value of depths Λ. Hence, we need it to allow at least one diagonally equivalent matrix for every depth matrix Λ whose entries are all nonzero. If we have some prior knowledge about the true depth matrix Λ in the form of $\Lambda \in P$ for some set $P \subseteq \mathbb{R}^{m\times n}$, the constraint is only required to allow at least one diagonally equivalent matrix for every depth matrix Λ in P. For example, in many applications it is known a priori that the true depths λ_{ij} are all positive. In such cases P is the set of $m \times n$ matrices with all positive elements. The concept of inclusiveness, therefore, can be defined formally as follows:

Definition 6 Given a set $P \subseteq \mathbb{R}^{m\times n}$ representing our prior knowledge about the possible values of the true depth matrix ($\Lambda \in P$), the constraint space $C \subseteq \mathbb{R}^{m\times n}$ is called *inclusive* if for every $m \times n$ matrix $\Lambda \in P$, there exists at least one matrix $\hat{\Lambda} \in C$ which is diagonally equivalent to Λ.

Definition 7 The constraint space $C \subseteq \mathbb{R}^{m\times n}$ is called *uniquely inclusive* if for every $m \times n$ matrix $\Lambda \in P$, there exists *exactly* one matrix $\hat{\Lambda} \in C$ which is diagonally equivalent to Λ.

In this paper whenever we use the term inclusive without specifying P, we mean the general case of P being the set of all $m\times n$ matrices with no zero element. We will only consider one other case where P is the set of all $m \times n$ matrices with all positive elements.

In addition to inclusiveness as a necessary property for a constraint, it is desirable for a constraint to *exclude* false solutions. This property can be defined as follows:

Definition 8 For $m\geq 2$ and $n\geq 8$, a constraint space $C \subseteq \mathbb{R}^{m\times n}$ is called *exclusive*[7] if every $\hat{\Lambda} \in C$ satisfies (D1-D3).

Now, we can present a class of constraints under which solving problem (41) leads to projective reconstruction:

Definition 9 Given integers $m \geq 2$ and $n \geq 8$, and a set $P \subseteq \mathbb{R}^{m\times n}$ representing our prior knowledge about the true depth

[7] In fact, the term exclusive might not be a precise term here, as (D1–D3) holding for all $\hat{\Lambda} \in C$ is just a sufficient condition for a constraint to exclude false solutions. While, according to Lemma 16, (D3) holding for all $\hat{\Lambda} \in C$ is necessary for ruling out false solutions, (D1) and (D2) holding for all members of C is not necessary for this purpose. This is because there might exist some $\hat{\Lambda} \in C$ for which (D1) or (D2) do not hold, but it is excluded by $\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\hat{\mathtt{X}}$. This is why in Sect. 5 we said that (D1–D3) are minimal *in a certain sense*.

matrix, we call the constraint space $C \subseteq \mathbb{R}^{m\times n}$ (uniquely) *reconstruction friendly* if it is both *exclusive* and (uniquely) *inclusive* with respect to P.

We will apply the same terms (inclusive, exclusive, reconstruction friendly) to the constraints themselves (as relations), and what we mean is that the corresponding constraint space has the property. The following proposition follows from the discussion above and Theorem 1.

Proposition 1 *Consider a setup of $m \geq 2$ camera matrices and $n \geq 8$ points $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ generically configured in the sense of (G1–G4), and projecting into the image points $\{\mathbf{x}_{ij}\}$ according to $\lambda_{ij}\mathbf{x}_{ij} = \mathtt{P}_i\mathbf{X}_j$ with nonzero scalars λ_{ij}. If C is a reconstruction friendly constraint space, then problem (41) has at least one solution and for any solution $(\hat{\Lambda}, \hat{\mathtt{P}}, \hat{\mathtt{X}})$, the configuration $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ is projectively equivalent to $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$, where the matrices $\hat{\mathtt{P}}_i \in \mathbb{R}^{3\times 4}$ and the points $\hat{\mathbf{X}}_j \in \mathbb{R}^4$ come from $\hat{\mathtt{P}} = \mathrm{stack}(\hat{\mathtt{P}}_1, \hat{\mathtt{P}}_2, \cdots, \hat{\mathtt{P}}_m)$ and $\hat{\mathtt{X}} = [\hat{\mathbf{X}}_1\hat{\mathbf{X}}_2\cdots\hat{\mathbf{X}}_n]$. If C is uniquely reconstruction friendly, then there is a unique depth matrix $\hat{\Lambda}$ as the solution to (41).*

Notice, that the uniqueness is with respect to $\hat{\Lambda}$, however a certain solution $\hat{\Lambda}$ gives a class of camera matrices and points, namely $(\hat{\mathtt{P}}\mathtt{H}, \mathtt{H}^{-1}\hat{\mathbf{X}})$ where $\mathtt{H}$ is an arbitrary 4×4 invertible matrix.

Being reconstruction friendly is a desirable property for a constraint. However, this does not mean that other constraints are not useful. There can be other ways of avoiding false solutions, including choosing a proper initial solution for iterative factorization algorithms or trying different initial solutions or different forms of a certain class of constraints. What is important for reconstruction *unfriendly* constraints is to be aware of possible false solutions and being able to determine whether the algorithm has fallen into any of them.

Besides giving correct solutions to (41), there are other desirable properties for a constraint space. We are specifically talking about the properties making the constraint usable with practical algorithms. For example, when dealing with iterative algorithms that converge to the final solution, it is essential that the constraint space C is closed. This is because for a non-closed constraint space, even if the sequence of solutions throughout all iterations satisfy all the constraints, they may converge to something outside C.

In the next subsections, to demonstrate how the theory we developed can be applied to the analysis of depth constraints, we examine some of the depth constraints used in the literature on factorization-based algorithms. It turned out that all of the constraints we could find in the literature either have a *compact* constraint space or are in the form of *linear equalities*. We consider each of these classes in a separate subsection. For each class, in addition to reviewing the constraints in the literature, we introduce a new class of constraints with extra desirable properties. This gives the reader an idea as to how our theory can be exploited for the design of new constraints. In particular, in Sect. 6.2.3, we introduce a class of linear equality constraints which are reconstruction friendly.

6.1 Compact Constraint Spaces

6.1.1 The Transportation Polytope Constraint

We consider the constraint used in Dai et al. (2010, 2013), which requires $\hat{\Lambda}$ to have prescribed row and column sums and to have all nonnegative elements. This can be represented as

$$\hat{\Lambda}\mathbf{1}_n = \mathbf{u}, \quad \hat{\Lambda}^T\mathbf{1}_m = \mathbf{v}, \tag{42}$$

$$\hat{\Lambda} \succeq 0, \tag{43}$$

where the vectors $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{v} \in \mathbb{R}^n$ are such that $u_i > 0$ for all i, $v_j > 0$ for all j and $\sum_{i=1}^m u_i = \sum_{j=1}^n v_j$. The relation $\succeq$ means element-wise greater or equal. Notice that although (42) introduces $m+n$ constraints, only $m+n-1$ of them are linearly independent. In Angst et al. (2011) it has been noted that the corresponding constraint space is known as the Transportation Polytope. Thanks to a generalization of the well-known Sinkhorn's Theorem (Sinkhorn 1964) for rectangular matrices (Sinkhorn 1967), one can say that for every $m\times n$ matrix Λ with all positive elements and any two vectors $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{v} \in \mathbb{R}^n$ with all positive entries, there exists a matrix $\hat{\Lambda}$ which is diagonally equivalent to Λ and satisfies the row and column sums constraint (42). Therefore, (42) is inclusive if the true depth matrix Λ is known to have all positive values, that is the set P representing the prior knowledge in Definition 9 is equal to the set of all $m\times n$ matrices with all positive elements. It is also obvious that the constraint (42) enforces all rows and all columns of $\hat{\Lambda}$ to be nonzero. Hence, every matrix in the constraint space satisfies (D1, D2). To see if the constraint is exclusive it only remains to examine whether or not constraints (42) and (43) allow for any cross-shaped depth matrix.

Assume that $\hat{\Lambda}$ is a cross-shaped matrix centred at (r, c), as in Fig. 6. Then the elements of $\hat{\Lambda}$ are uniquely determined by (42) as follows: $\hat{\lambda}_{ic} = u_i$ for all $i \neq r$, $\hat{\lambda}_{rj} = v_j$ for all $j \neq c$ and $\hat{\lambda}_{rc} = u_r - \sum_{j\neq c} v_j = v_c - \sum_{i\neq r} u_j$ (the latter equality is true due to $\sum_{i=1}^m u_i = \sum_{j=1}^n v_j$). This has been illustrated in Fig. 6. It is easy to check at all elements of $\hat{\Lambda}$ are nonnegative except possibly $\hat{\lambda}_{rc}$. Therefore, to satisfy (43), we must have $u_r - \sum_{j\neq c} v_j \geq 0$. Therefore, if for any choice of r and c, $u_r - \sum_{j\neq c} v_j \geq 0$ is satisfied, then the constraints (42) and (43) allow for a cross-shaped structure and hence, according to Lemma 16, allow a false solution to (41). Otherwise, (42) and (43) together give a reconstruction friendly constraint space, and hence, do not allow any false solution according to Proposition 1.

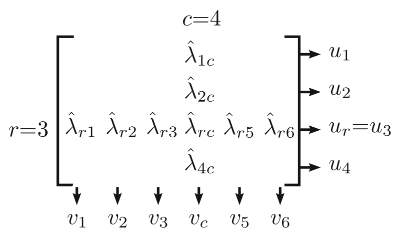
$c{=}4$
$\hat\lambda_{1c}$
$\hat\lambda_{2c}$
$r{=}3$
$\hat\lambda_{r1}$ $\hat\lambda_{r2}$ $\hat\lambda_{r3}$ $\hat\lambda_{rc}$ $\hat\lambda_{r5}$ $\hat\lambda_{r6}$
$\hat\lambda_{4c}$
u_1
u_2
$u_r{=}u_3$
u_4
v_1 v_2 v_3 v_c v_5 v_6

Fig. 6 A 4×6 cross-shaped depth matrix $\hat\Lambda$ centred at (r, c) with $r = 3, c = 4$. The *blank* parts of the matrix indicate zero elements. The only way for the *rows* and *columns* of the matrix to sum up to the marginal values $\{u_i\}$ and $\{v_j\}$ is to have $\hat\lambda_{ic} = u_i$ for $i \neq r$, $\hat\lambda_{rj} = v_j$ for $j \neq c$, and $\hat\lambda_{rc} = u_r - \sum_{j\neq c} v_j = v_c - \sum_{i\neq r} u_j$.

As a major example, if we take $\mathbf{u} = n\mathbf{1}_m$ and $\mathbf{v} = m\mathbf{1}_n$ as chosen in (Dai et al. 2010, 2013), for any choice of r and c we have $u_r - \sum_{j\neq c} v_j = m + n - mn$. This is always smaller than zero by our assumption of having two or more views ($m \geq 2$) and 8 or more points ($n \geq 8$). Therefore, with the choice of $\mathbf{u} = n\mathbf{1}_m$ and $\mathbf{v} = m\mathbf{1}_n$, (42) and (43) give a reconstruction friendly constraint space. The disadvantage of this constraint is that it includes inequalities. This makes it difficult to implement fast and efficient algorithms for large scale problems.

6.1.2 Fixing the Norms of Rows and Columns

As suggested by Triggs (1996) and Hartley and Zisserman (2004), after each iteration of a factorization-based algorithm, one can alternatingly scale row and columns of $\hat\Lambda$ to have prescribed norms. Here, we analyse this case for the cases where the norms are l^p-norms for some real number $p \geq 1$ (being real implies $p < \infty$). Consider the matrix $\hat\Gamma \overset{\text{def}}{=} [|\hat\lambda_{ij}|^p]$, whose ij-th element is equal to $|\hat\lambda_{ij}|^p$. If all $\hat\lambda_{ij}$-s are nonzero, all elements of $\hat\Gamma$ are positive, and hence, alternatingly scaling row and columns of $\hat\Lambda$ to have prescribed l^p-norms is equivalent to alternatingly scaling rows and columns of $\hat\Gamma$ to have prescribed sums, that is applying the Sinkhorn's algorithm to $\hat\Gamma$ (Sinkhorn 1964, 1967), making $\hat\Gamma$ converge to a matrix with the desired marginal sums and hence making $\hat\Lambda$ converge to a matrix with given row and column l^p-norms. Therefore, applying this iterative procedure after every iteration of a factorization-based algorithms keeps $\hat\Lambda$ in the following constraint space

$$\sum_{j=1}^{n} |\hat\lambda_{ij}|^p = u_i, \quad i = 1, \ldots, m \tag{44}$$

$$\sum_{i=1}^{m} |\hat\lambda_{ij}|^p = v_j, \quad j = 1, \ldots, n \tag{45}$$

for vectors $\mathbf{u} = [u_1, \ldots, u_m]^T$ and $\mathbf{v} = [v_1, \ldots, v_n]^T$ with all positive elements. Notice that $\mathbf{u}$ and $\mathbf{v}$ must be taken such that $\sum_{i=1}^m u_i = \sum_{j=1}^n v_j$. The above constrains $\hat\Gamma = [|\hat\lambda_{ij}|^p]$ as

$$\hat\Gamma\mathbf{1}_n = \mathbf{u}, \quad \hat\Gamma^T\mathbf{1}_m = \mathbf{v}. \tag{46}$$

Moreover, $\hat\Gamma \succeq 0$ is automatically satisfied by the definition of $\hat\Gamma$. For the true depths λ_{ij}, take $\Gamma \overset{\text{def}}{=} [|\lambda_{ij}|^p]$ and notice that it has all positive elements as λ_{ij}-s are all nonzero. Thus, by applying the generalization of the Sinkhorn's theorem to rectangular matrices (Sinkhorn 1967) we can say that there exists vectors $\boldsymbol{\tau} = [\tau_1, \tau_2, \ldots, \tau_m]^T$, $\boldsymbol{\nu} = [\nu_1, \nu_2, \ldots, \nu_n]^T$ with all positive entries such that $\hat\Gamma = \operatorname{diag}(\boldsymbol{\tau})\,\Gamma\operatorname{diag}(\boldsymbol{\nu})$ satisfies (46). Thus, for $\boldsymbol{\tau}' = [\tau_1^{1/p}, \tau_2^{1/p}, \ldots, \tau_m^{1/p}]^T$, $\boldsymbol{\nu}' = [\nu_1^{1/p}, \nu_2^{1/p}, \ldots, \nu_n^{1/p}]^T$, the matrix $\hat\Lambda = \operatorname{diag}(\boldsymbol{\tau}')\,\Lambda\operatorname{diag}(\boldsymbol{\nu}')$ satisfies (44) and (45). Therefore, (44) and (45) together give an inclusive constraint space. To check for (D1–D3), notice that $\hat\Gamma$ and $\hat\Lambda$ have a common zero pattern. Therefore, (D1–D3) are satisfied for $\hat\Lambda$ if and only if they are satisfied for $\hat\Gamma$. By considering (46) and $\hat\Gamma \succeq 0$, with the same discussion as the previous subsection we can say that (44) and (45) form a reconstruction friendly constraint if and only if $u_r - \sum_{j\neq c} v_j \geq 0$ for all r and c. Specifically, if one requires rows to have common norms and also columns to have common norms, as suggested by Triggs (1996) and Hartley and Zisserman (2004), then we have $\mathbf{u} = \alpha n\mathbf{1}_m$ and $\mathbf{v} = \alpha m\mathbf{1}_n$ for some nonzero scaling factor α. A similar argument as in the previous subsection shows that with this choice of $\mathbf{u}$ and $\mathbf{v}$, fixing l^p-norms of rows and columns results in a reconstruction friendly constraint space.

The problem with (46) as a constraint is that even simple target functions are hard to optimize subject to it. Implementing this constraint as a balancing stage after every iteration of a factorization-based algorithm can prevent us from having a descent move at every iteration.

6.1.3 Fixed Row or Column Norms

Heyden et al. (1999) uses the constraint of fixing the l^2-norms of the rows of the depth matrix. This constraint can be written as

$$\sum_{j=1}^{n} |\hat\lambda_{ij}|^2 = u_i, \quad i = 1, \ldots, m \tag{47}$$

for fixed positive numbers u_i. Indeed, this constraint is inclusive as for every matrix Λ with all nonzero rows one can scale the rows to obtain a matrix $\hat\Lambda = \operatorname{diag}(\boldsymbol{\tau})\Lambda$ with prescribed row norms. Every matrix $\hat\Lambda$ satisfying this constraint has nonzero rows. However, the constraint allows for zero columns and cross-shaped solutions. A similar situation holds for Mahamud et al. (2001) where the columns of the depth matrix are required to have a unit weighted l^2-norm.

The disadvantage of these constraints is allowing for zero columns (or zero rows in the second case) and cross-shaped structures. The advantage is that they can be efficiently implemented with iterative factorization-based algorithms, by solving a number of eigenvalue problems at every iteration (Mahamud et al. 2001). The compactness of the constraint space contributes to the proof of special convergence properties for special factorization-based algorithms (Mahamud et al. 2001).

6.1.4 *Fixing Norms of Tiles*

In this subsection we show how the fixed row and fixed column constraints can be somehow combined to make more desirable constraints. This is done by *tiling* the depth matrix $\hat{\Lambda}$ with row and column vectors, and requiring each tile to have a unit norm (or a fixed norm in general). Examples of tiling can be seen in Fig. 4 at p. 2.

The process of tiling is done as follow: It starts by putting a single tile (row vector or column vector) in the matrix. We then keep adding tiles such that the tiled area stays rectangular. At every stage either a horizontal tile (row vector) is vertically concatenated or a vertical tile (column vector) is horizontally concatenated to the tiled area, with the constraint that the tiled region remains rectangular. The process is continued until the whole $\hat{\Lambda}$ is tiled. This process is illustrated in Fig. 7. By tiling the matrix in this way, the corresponding constraint will be inclusive. We do not prove this formally here, instead, we show how the proof is constructed by giving an example in Fig. 7.

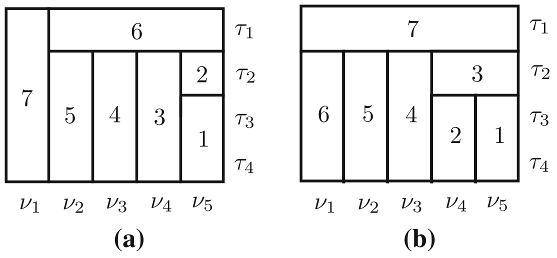

Fig. 7 Examples of the procedure of tiling a 4×5 depth matrix. The numbers show the order in which the tiles are placed. In these examples, we start by placing a 2×1 tile on the *left bottom* of the matrix. The tiles are added such that the tiled region at any time remains a *rectangle*. Having an $m' \times n'$ *rectangular* area tiled already, we either concatenate an $m' \times 1$ vertical block to its left, or a $1 \times n'$ block to its top. The claim is that with this procedure the constraint of every tile having a unit (or a fixed positive) norm is inclusive. This can be shown as follows: We start by taking $\hat{\Lambda} = \Lambda$, and keep updating $\hat{\Lambda}$ by scaling one of its *rows* or one of its *columns* at a time until it satisfies all the constraints, that is all of its tiles have a unit norm. For matrix **a**, the updates can be done as follows: choose arbitrary nonzero values for τ_3 and τ_4 and apply them to the matrix (multiply them respectively by the 3rd and 4th *row* of $\hat{\Lambda}$). Now, choose ν_5 such that tile 1 has a unit norm and apply it. Then choose τ_2 and apply it such that tile 2 has a unit norm. Now, choose and apply ν_4, ν_3 and ν_2 such that tiles 3, 4, 5 have a unit norm, and finally choose and apply τ_1 and then ν_1 to respectively make tiles 6 and 7 have a unit norm. The procedure for **b** is similar, but the order of finding τ_i-s and ν_j-s is as follows: $\tau_3, \tau_4, \nu_5, \nu_4, \tau_2, \nu_3, \nu_2, \nu_1, \tau_1$

Figure 4 at p. 2 shows six examples of tiling a 4×6 depth matrix. Looking at Fig. 4a one can see that for an $m \times n$ matrix, if the tiling begins by placing a $1 \times n$ block, all other tiles have to be also $1 \times n$ and the constraint is reduced to the case of requiring fixed row norms, a special case of which was discussed in the previous subsection. Similarly, if the first tile is $m \times 1$, the constraint amounts to fixing the norms of columns of the depth matrix Fig. 4b. But the case of interest here is when the first tile is a 1×1 block, like Fig. 4c–f. In this case, the constraint rules out having zero rows or zero columns in the depth matrix. It does not rule out cross-shaped structures, but it constrains the central position of the cross to the location of 1×1 tiles (see Fig. 4c–f).

If the norms used for the constraints are weighted l^2-norms with properly chosen weights, an efficient factorization algorithm can be implemented. For more details see Sect. 9. Similar convergence properties as in Mahamud et al. (2001) can be proved for these constraints given a proper algorithm.

6.2 Linear Equality Constraints

6.2.1 *Fixing Sums of Rows and Columns*

In this subsection, we consider constraining $\hat{\Lambda}$ to have prescribed row and column sums, that is

$$\hat{\Lambda}\mathbf{1}_n = \mathbf{u}, \quad \hat{\Lambda}^T\mathbf{1}_m = \mathbf{v}, \tag{48}$$

for two m- and n-dimensional vectors $\mathbf{u}$ and $\mathbf{v}$ with all nonzero entries for which $\sum_{i=1}^{m} u_i = \sum_{j=1}^{n} v_j$. This is similar to the transportation polytope constraint introduced in Sect. 6.1.1, with the only difference that it does not require $\hat{\Lambda} \succeq 0$. Thus, it has the advantage of allowing for more efficient algorithms compared to the case where inequality constraints are also present. We can see this in Dai et al. (2013), where the inequality constraint $\hat{\Lambda} \succeq 0$ has been disregarded when proposing fast and scalable algorithms.

With a similar argument as was made in Sect. 6.1.1, one can say that (48) gives an inclusive constraint space when the true depth matrix Λ is known to have all positive elements and $\mathbf{u}$ and $\mathbf{v}$ are chosen to have all positive entries. The constraint also enforces all rows and columns of $\hat{\Lambda}$ to be nonzero.

However, as noted in Sect. 6.1.1, a cross-shaped matrix with any arbitrary centre (r, c) whose elements are chosen as $\hat{\lambda}_{ic} = u_i$ for all $i \neq r$, $\hat{\lambda}_{rj} = v_j$ for all $j \neq c$ and $\hat{\lambda}_{rc} = u_r - \sum_{j \neq c} v_j = v_c - \sum_{i \neq r} u_j$, satisfies (48). Therefore, by Lemma 16 we can say that it always allows for cross-shaped solutions.

The bad thing about this type of constraint is that there is no limitation as to where the cross-shaped structure can be centred. But the good news is that, according to our experiments (Sect. 9), it can be hard for an iterative algorithm to converge to a cross-shaped solution with the choice of $\mathbf{u} = n\mathbf{1}_m$ and $\mathbf{v} = m\mathbf{1}_n$. This could be explained as follows: As noted in

$$\underset{\textbf{(a)}}{\begin{bmatrix} 1&1&1&1&1&1\\ 1&1&1&1&1&1\\ 1&1&1&1&1&1\\ 1&1&1&1&1&1 \end{bmatrix}} \quad \Rightarrow \quad \underset{\textbf{(b)}}{\begin{bmatrix} &&&6&&\\ 4&4&4&-14&4&4\\ &&&6&&\\ &&&6&& \end{bmatrix}}$$

Fig. 8 Examples of 4×6 matrices, both satisfying $\hat{\Lambda}\mathbf{1}_n = n\mathbf{1}_m$ and $\hat{\Lambda}^T\mathbf{1}_m = m\mathbf{1}_n$. **a** is a typical initial state for iterative factorization-based algorithm, **b** is the only cross-shape structure centred at (2,4) allowed by the constraint. If the true depths are all positive, it can be harder for an algorithm to converge from (**a**) to (**b**), compared to converging to a correct solution with all positive elements

Sect. 6.1.1, if any cross-shaped structure occurs, the central element will have to be equal to $m + n - mn$. Under our assumptions ($m \geq 2, n \geq 8$), this is a negative number and its absolute value grows linearly both with respect to m and n. This can make it hard for the algorithm to converge to a cross-shaped structure starting from an initial solution like a matrix of all ones. This has been depicted in Fig. 8 for a 4×6 matrix, where the central element of the cross has to be -14. For a fairly small configuration of 20-views and 8-points this value is -132. This suggests that as the dimension of the depth matrix grows, it is made harder for the algorithm to converge to a cross-shaped solution.

6.2.2 Fixing Elements of One Row and One Column

Here, we consider the constraint of having all elements of a specific row and a specific column of the depth matrix equal to one, as used in Ueshiba and Tomita (1998). This means requiring $\hat{\lambda}_{rj} = 1$ for all j, and $\hat{\lambda}_{ic} = 1$ for all i. This can be represented as

$$\mathtt{M} \circ \hat{\Lambda} = \mathtt{M}. \tag{49}$$

where $\circ$ represents the Hadamard (element-wise) product and $\mathtt{M}$ is a mask matrix, having all elements of a specific row r and a specific column c equal to 1, and the rest of its elements equal to zero. This means that the mask matrix $\mathtt{M}$ is a cross-shaped matrix centred at (r, c). We leave it to the reader to check that this is an inclusive constraint, and also every matrix in the constraint space satisfies depth assumptions (D1) and (D2). However, one can easily check that, as $\mathtt{M}$ itself is a cross-shaped matrix, the constraint (49) allows for cross-shaped depth matrices. Therefore, by using the above constraint problem (41) can admit false solutions.

One advantage of this type of constraint is its element-wise nature. This can make the formulation of iterative factorization algorithms much easier compared to other types of constraints. The other advantage is that there is only a single possibility about where the cross in centred, which is the centre of cross in $\mathtt{M}$. Therefore, the occurrence of a cross-shaped solution can be easily verified. In the case where a cross-shaped solution happens, one can try rerunning the algorithm with a different mask $\mathtt{M}$ whose cross is centred elsewhere.

6.2.3 Step-like Mask Constraint: A Linear Reconstruction Friendly Equality Constraint

This section demonstrates a group of linear equality constraints which are reconstruction friendly. Like the previous subsection, the linear equalities are in the form of fixing elements of the depth matrix at certain sites. Therefore, it enjoys all the benefits of elementwise constraints.

To present the constraint, we first define the concept of a step-like mask. Consider an $m \times n$ matrix $\mathtt{M}$. To make a step-like mask, we have a travel starting from the upper-left corner of the matrix (location $1, 1$) and ending at its lower-right corner (location m, n). The travel from $(1, 1)$ to (m, n) is done by taking $m + n - 2$ moves, such that at each move we either go one step to the right or go one step down. In total, we will make $m - 1$ downward moves and $n - 1$ moves to the right. Therefore, the travel can be made in $(m+n-2)!/((m-1)!\,(n-1)!)$ ways. After doing a travel, we make the associated step-like mask by setting to 1 all $(m+n-1)$ elements of $\mathtt{M}$ corresponding to the locations that we have visited and setting to zero the rest of the elements. Examples of step-like masks are shown in Fig. 3 at p. 2 for $m = 4$ and $n = 6$.

Notice that a step-like mask has $m + n - 1$ nonzero elements which are arranged such that the matrix has no zero rows and no zero columns. An exclusive step-like mask is defined to be a step-like mask which is not cross-shaped (see Fig. 3). With an $m \times n$ step-like mask we can put linear equality constraints on a depth matrix $\hat{\Lambda}$ as follows

$$\mathtt{M} \circ \hat{\Lambda} = \mathtt{M}. \tag{50}$$

where $\circ$ represents the Hadamard (element-wise) product. In other words, it enforces the matrix $\hat{\Lambda}$ to have unit elements at the sites where $\mathtt{M}$ has ones.

One can show that with an exclusive step-like mask $\mathtt{M}$, the constraint (50) is uniquely reconstruction friendly. As the constraints enforce $\hat{\Lambda}$ to be nonzero at the sites where $\mathtt{M}$ has ones, it is easy to see that if $\hat{\Lambda}$ satisfies (50), it satisfies (D1–D3) and hence the constraint space is exclusive. Therefore, we just have to show that for each matrix Λ with all nonzero elements, there exists exactly one diagonally equivalent matrix $\hat{\Lambda}$ satisfying (50). The proof is quite simple, but we do not provide it here. Instead, we explain the idea of the proof by giving an example for a special case in Fig. 9.

One can think of many ways to extend the step-like constraints. For example, one can fix the desired elements of $\hat{\Lambda}$ to arbitrary nonzero values instead of ones. The reader can also check that if $\mathtt{M}$ is obtained by applying any row

$$\begin{array}{c|cccc} & \nu_1 & \nu_2 & \nu_3 & \nu_4 \\ \hline \tau_1 & \underline{\lambda_{11}} & \underline{\lambda_{12}} & \lambda_{13} & \lambda_{14} \\ \tau_2 & \lambda_{21} & \underline{\lambda_{22}} & \lambda_{23} & \lambda_{24} \\ \tau_3 & \lambda_{31} & \underline{\lambda_{32}} & \underline{\lambda_{33}} & \underline{\lambda_{34}} \end{array} \qquad \mathtt{M} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

Fig. 9 An example of a 3×4 depth matrix Λ (*left*) and an exclusive step-like mask $\mathtt{M} = [m_{ij}]$ (*right*). The elements λ_{ij} of Λ are *underlined* at the sites where $m_{ij} = 1$, which is where $\hat{\lambda}_{ij}$-s are constrained to be equal to 1. The aim is to show that there exists a unique $\hat{\Lambda}$ in the form of $\hat{\Lambda} = \mathrm{diag}(\boldsymbol{\tau})\, \Lambda\, \mathrm{diag}(\boldsymbol{\nu})$ whose elements are 1 at the sites where $\mathtt{M}$ has ones. Equivalently $\mathtt{M} \circ \hat{\Lambda} = \mathtt{M}$. This can be done as follows: Start by taking $\hat{\Lambda} = \Lambda$, and keep updating $\hat{\Lambda}$ by scaling its *rows* and *columns*, one at a time, until it satisfies the constraint $\mathtt{M} \circ \hat{\Lambda} = \mathtt{M}$. For the above matrix, we start by assigning an arbitrary nonzero value to τ_1 and multiplying τ_1 by the first *row* of $\hat{\Lambda}$. Then we choose ν_1 and ν_2 and multiply them by the corresponding *columns* of $\hat{\Lambda}$ such that $\hat{\lambda}_{11} = 1$ and $\hat{\lambda}_{12} = 1$. Now, we choose τ_2 and τ_3 and multiply them by the corresponding *rows* of $\hat{\Lambda}$ such that we have $\hat{\lambda}_{22} = 1$ and $\hat{\lambda}_{32} = 1$. Finally, we choose ν_3 and ν_4 and multiply them by the corresponding *columns* of $\hat{\Lambda}$ to have $\hat{\lambda}_{33} = 1$ and $\hat{\lambda}_{34} = 1$. Notice that in this process, except τ_1 which is chosen arbitrarily, there is only one choice for each of the entries $\tau_2, \tau_3, \nu_1, \nu_2, \nu_3, \nu_4$ for each choice of τ_1. Because, given any pair of vectors $(\boldsymbol{\tau}, \boldsymbol{\nu})$, all pairs of vectors $(\alpha\boldsymbol{\tau}, \alpha^{-1}\boldsymbol{\nu})$ for all $\alpha \neq 0$ have the same effect, this means that given the matrices Λ and $\mathtt{M}$, the choice of $\hat{\Lambda} = \mathrm{diag}(\boldsymbol{\tau})\, \Lambda\, \mathrm{diag}(\boldsymbol{\nu})$ is unique

and column permutation to an exclusive step-like mask, then the constraint (50) will still be reconstruction friendly. One important extension is to remove some of the constraints by turning to 0 some of the elements of the mask matrix $\mathtt{M}$. Potential elements of a step-like matrix $\mathtt{M}$ for the removal (switching to zero) are the stair edges, which are the elements whose left and lower elements (or right and upper elements) are both 1 (see Fig. 10). We call the new matrices *edgeless* step-like masks. As switching some elements of $\mathtt{M}$ to zero amounts to removing some linear equations from the set of constraints, an edgeless step-like mask still gives an inclusive constraint. If the edge elements for the removal are chosen carefully from an exclusive step-like mask, the corresponding constraint $\mathtt{M} \circ \hat{\Lambda} = \mathtt{M}$ can still be exclusive, not allowing for the violation of (D1–D3). Figure 10a, b illustrates examples of exclusive edgeless step-like masks. The corresponding constraint $\mathtt{M} \circ \hat{\Lambda} = \mathtt{M}$ for such a mask is reconstruction friendly, however it is not *uniquely* reconstruction friendly. Our experiments show that, using the same algorithm, an edgeless mask results in a faster convergence than its corresponding edged mask. One explanation is that, in this case, the removal of each constraint, in addition to increasing the dimension of the search space, increases the dimension of the solution space[8] by one. This can allow an iterative algorithm to find a shorter path from the initial estimate of $\hat{\Lambda}$ to a correct solution.

[8] namely $\{\hat{\Lambda} \mid \hat{\Lambda} = \mathrm{diag}(\boldsymbol{\tau})\, \Lambda \mathrm{diag}(\boldsymbol{\nu}),\ \mathtt{M} \circ \hat{\Lambda} = \mathtt{M}\}$.

$$\underset{\textbf{(a)}}{\begin{bmatrix} 1 & 0 & & & & \\ & 1 & 1 & 0 & & \\ & & & 1 & 0 & \\ & & & & 1 & 1 \end{bmatrix}} \quad \underset{\textbf{(b)}}{\begin{bmatrix} 1 & 1 & 0 & & & \\ & & 1 & & & \\ & & 1 & & & \\ & & 0 & 1 & 1 & 1 \end{bmatrix}} \quad \underset{\textbf{(c)}}{\begin{bmatrix} 1 & & & & & \\ 1 & & & & & \\ 1 & & & & & \\ 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}}$$

Fig. 10 Examples of 4×6 edgeless step-like mask matrices obtained by removing (making zero) some of the stair edges of matrices in Fig. 3. The *blank* parts of the matrices are zero. The elements explicitly shown by 0 are the removed edges (those that are 1 on the original step-like matrix). **a** and **b** are examples of an exclusive edgeless step-like matrix, resulting in a reconstruction friendly constraint

7 Projective Reconstruction via Rank Minimization

Recall from the last section that in the factorization-based projective reconstruction the following problem is sought to be solved

$$\text{find}_{\hat{\Lambda},\ \hat{\mathtt{P}}_{3m\times 4},\ \hat{\mathtt{X}}_{4\times n}} \quad \text{s.t.}\ \ \hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\,\hat{\mathtt{X}}, \quad \hat{\Lambda} \in C \tag{51}$$

which is a restatement of (41). Rank minimization is one of the approaches to factorization-based projective reconstruction, in which, in lieu of (51), the following problem is solved:

$$\min_{\hat{\Lambda}}\ \mathrm{rank}\big(\hat{\Lambda} \odot [\mathbf{x}_{ij}]\big) \quad \text{s.t.}\ \ \hat{\Lambda} \in C. \tag{52}$$

Two other closely related problems are

$$\text{find}\ \hat{\Lambda} \quad \text{s.t.}\ \mathrm{rank}\big(\hat{\Lambda} \odot [\mathbf{x}_{ij}]\big) \leq 4, \quad \hat{\Lambda} \in C, \tag{53}$$

$$\text{find}\ \hat{\Lambda} \quad \text{s.t.}\ \mathrm{rank}\big(\hat{\Lambda} \odot [\mathbf{x}_{ij}]\big) = 4, \quad \hat{\Lambda} \in C. \tag{54}$$

If any solution $\hat{\Lambda}$ is found for any of the above problems such that $\mathrm{rank}(\hat{\Lambda} \odot [\mathbf{x}_{ij}]) \leq 4$, the camera matrices and points can be estimated from the factorization of $\hat{\Lambda} \odot [\mathbf{x}_{ij}]$. We shall show that if C is reconstruction friendly, any solution to any of the above problems leads to projective reconstruction. First, it is easy to see that (53) is in fact equivalent to problem (51):

Lemma 17 *Given any set of 3D points $\mathbf{x}_{ij}$ for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$, the problems* (53) *and* (51) *are equivalent in terms of finding $\hat{\Lambda}$.*

Here, by being equivalent we mean that any solution $\hat{\Lambda}$ to one problem is a solution to the other. Obviously, this implies that if there exists no solution to one of the problems, then there cannot exist any solution to the other. The proof, which is left to the reader, uses the fact that any $3m \times n$ matrix whose rank is 4 or less, can be factored as the product of a $3m \times 4$ matrix $\hat{\mathtt{P}}$ by a $4 \times n$ matrix $\hat{\mathtt{X}}$. Notice that to prove the above lemma we need not make any assumption about C or how the points $\mathbf{x}_{ij}$ are created. The two other problems (52) and (54) are not in general equivalent to (51). However, if C is reconstruction friendly, one can show that all the four problems (52)–eq:projspsrankspsfind and (51) are equivalent:

Proposition 2 *Consider a setup of $m \geq 2$ camera matrices and $n \geq 8$ points $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ generically configured in the sense of (G1–G4), and projecting into the image points $\{\mathbf{x}_{ij}\}$ according to $\lambda_{ij}\mathbf{x}_{ij} = \mathtt{P}_i\mathbf{X}_j$ with nonzero scalars λ_{ij}. If $C \subseteq \mathbb{R}^{m\times n}$ is a reconstruction friendly constraint space, then given the image points $\mathbf{x}_{ij}$, the problems (52), (53) and (54) are all equivalent to (51) in terms of finding $\hat{\Lambda}$.*

Proof As (53) and (51) are equivalent, the proof will be complete by showing

– (54) $\subseteq$ (53),
– (51) $\subseteq$ (54),
– (52) $\subseteq$ (53),
– (54) $\subseteq$ (52),

where (P1) $\subseteq$ (P2) means that any solution to (P1) is a solution to (P2). The first part, that is (54) $\subseteq$ (53), is obvious. To show (51) $\subseteq$ (54), assume that $(\hat{\Lambda}, \hat{\mathtt{P}}, \hat{\mathtt{X}})$ is a solution to (51). By Proposition 1 and the definition of projective equivalence we can conclude that $\hat{\mathtt{P}} = \mathrm{diag}(\boldsymbol{\tau} \otimes \mathbf{1}_3)\, \mathtt{PH}$ and $\hat{\mathtt{X}} = \mathtt{H}^{-1}\mathtt{X}\,\mathrm{diag}(\boldsymbol{\nu})$ for some invertible matrix $\mathtt{H}$ and vectors $\boldsymbol{\tau}$ and $\boldsymbol{\nu}$ with all nonzero entries, where $\mathtt{P} = \mathrm{stack}(\mathtt{P}_1, \ldots, \mathtt{P}_m)$, $\mathtt{X} = [\mathbf{X}_1, \ldots, \mathbf{X}_n]$ and $\otimes$ denotes the Kronecker product. This gives

$$\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\hat{\mathtt{X}} = \mathrm{diag}(\boldsymbol{\tau} \otimes \mathbf{1}_3)\, \mathtt{PX}\,\mathrm{diag}(\boldsymbol{\nu}) \tag{55}$$

From (G1,G2) it follows that $\mathtt{P}$ and $\mathtt{X}$ respectively have full column and full row rank, and hence, $\mathtt{PX}$ is of rank 4. Given this, plus the fact that $\boldsymbol{\tau}$ and $\boldsymbol{\nu}$ have all nonzero entries, (55) implies that $\mathrm{rank}(\hat{\Lambda} \odot [\mathbf{x}_{ij}]) = 4$, meaning that $\hat{\Lambda}$ is a solution to (54).

To see (52) $\subseteq$ (53), notice that according to Proposition 1, (51) has at least one solution. This means that the equivalent problem (53) has also one solution $\hat{\Lambda}' \subseteq C$ for which $\mathrm{rank}(\hat{\Lambda}' \odot [\mathbf{x}_{ij}]) \leq 4$. For any solution $\hat{\Lambda} \subseteq C$ to (52) we have $\mathrm{rank}(\hat{\Lambda} \odot [\mathbf{x}_{ij}]) \leq \mathrm{rank}(\hat{\Lambda}' \odot [\mathbf{x}_{ij}]) \leq 4$. This means that $\hat{\Lambda}$ is also a solution to (53).

Finally, to show (54) $\subseteq$ (52), notice that from (53) $\equiv$ (51) and (51) $\subseteq$ (54) we already know that (53) $\subseteq$ (54). It follows that $\mathrm{rank}(\hat{\Lambda} \odot [\mathbf{x}_{ij}]) \geq 4$ for all $\hat{\Lambda} \in C$. Thus, any solution to (54) minimizes $\mathrm{rank}(\hat{\Lambda} \odot [\mathbf{x}_{ij}])$, and hence, is also a solution to (52). □

As a corollary, we can say that with the conditions of Proposition 2, all the problems (52), (53) and (54) have at least one solution. This is because Proposition 1 suggests this fact about (51). It is also worth to mention that, with some extra effort, a stronger variant of Proposition 2 can be proved in which the constraint $\hat{\Lambda} \in C$ is only required to exclude (D1) and (D2), rather than (D1–D3) altogether. In other words, the problems (51–54) are still equivalent if only (D1) and (D2) are ruled out by the constraint.

8 Iterative Projective Reconstruction Algorithms

Most of the projective factorization-based problems are solved iteratively. The output of such algorithms is not in the form of a deterministic final solution, but rather is a sequence $(\{\hat{\mathtt{P}}_i^{(t)}\}, \{\hat{\mathbf{X}}_j^{(t)}\}, \{\hat{\lambda}_{ij}^{(t)}\})$ which one hopes to converge to a sensible solution. There are many questions such as whether this sequence converges, and if it does, whether it converges to a correct solution. Answering such algorithm-specific questions, however, is beyond the scope of this paper. However, a more basic question that needs answering is that, given a constraint space C, if the sequence $\{\hat{\Lambda}^{(t)}\} \subseteq C$ converges to some $\hat{\Lambda}$, and moreover, the sequence $\{\hat{\Lambda}^{(t)} \odot [\mathbf{x}_{ij}] - \hat{\mathtt{P}}^{(t)}\,\hat{\mathtt{X}}^{(t)}\}$ converges to zero, then whether $\hat{\Lambda}$ is a solution to the factorization problem (41), that is $\hat{\Lambda} \in C$ and $\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\,\hat{\mathtt{X}}$ for some $\hat{\mathtt{P}} \in \mathbb{R}^{3m\times 4}$ and $\hat{\mathtt{X}} \in \mathbb{R}^{4\times n}$. It is easy to check that C being *closed* is sufficient for this to happen:

Proposition 3 *Consider a set of image points $\{\mathbf{x}_{ij}\}$, $i = 1, \ldots, m$ and $j = 1, \ldots, n$, and a* closed *constraint space $C \subseteq \mathbb{R}^{m\times n}$. If there exists a sequence of depth matrices $\{\hat{\Lambda}^{(t)}\} \subseteq C$ converging to a matrix $\hat{\Lambda}$, and for each $\hat{\Lambda}^{(t)}$ there exist $\hat{\mathtt{P}}^{(t)} \in \mathbb{R}^{3m\times 4}$ and $\hat{\mathtt{X}}^{(t)} \in \mathbb{R}^{4\times n}$ such that $\hat{\Lambda}^{(t)} \odot [\mathbf{x}_{ij}] - \hat{\mathtt{P}}^{(t)}\,\hat{\mathtt{X}}^{(t)} \to 0$ as $t \to \infty$, then there exist $\hat{\mathtt{P}} \in \mathbb{R}^{3m\times 4}$ and $\hat{\mathtt{X}} \in \mathbb{R}^{4\times n}$ such that $(\hat{\Lambda}, \hat{\mathtt{P}}, \hat{\mathtt{X}})$ is a solution to the factorization problem*

$$\mathrm{find}_{\hat{\Lambda},\ \hat{\mathtt{P}}_{3m\times 4},\ \hat{\mathtt{X}}_{4\times n}} \quad \text{s.t.}\ \ \hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\,\hat{\mathtt{X}}, \quad \hat{\Lambda} \in C \tag{56}$$

Proof Let $\mathtt{A}^{(t)} = \hat{\mathtt{P}}^{(t)}\,\hat{\mathtt{X}}^{(t)}$. As the mapping $\Lambda' \mapsto \Lambda' \odot [\mathbf{x}_{ij}]$ is continuous, $\hat{\Lambda}^{(t)} \odot [\mathbf{x}_{ij}] - \mathtt{A}^{(t)} \to 0$ and $\hat{\Lambda}^{(t)} \to \hat{\Lambda}$ give $\mathtt{A}^{(t)} \to \hat{\Lambda} \odot [\mathbf{x}_{ij}] \overset{\mathrm{def}}{=} \mathtt{A}$. Also, $\mathrm{rank}(\mathtt{A}) \leq 4$ because $\mathrm{rank}(\mathtt{A}^{(t)}) \leq 4$ and the space of $3m \times n$ real matrices with rank 4 or less is closed. Thus, $\mathtt{A}$ can be factored as $\mathtt{A} = \hat{\mathtt{P}}\hat{\mathtt{X}}$ for some $\hat{\mathtt{P}} \in \mathbb{R}^{3m\times 4}$ and $\hat{\mathtt{X}} \in \mathbb{R}^{4\times n}$, giving $\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \mathtt{A} = \hat{\mathtt{P}}\hat{\mathtt{X}}$. Moreover, as C is closed and $\{\hat{\Lambda}^{(t)}\} \subseteq C$ we have $\hat{\Lambda} \in C$. This completes the proof. □

According to the above, as long as the constraint space C is closed, all the results obtained in the previous section about the solutions to the factorization problem (41), can be safely used for iterative algorithms when the sequence of depths $\{\hat{\Lambda}^{(t)}\}$ is convergent and $\hat{\Lambda}^{(t)} \odot [\mathbf{x}_{ij}] - \hat{\mathtt{P}}^{(t)}\,\hat{\mathtt{X}}^{(t)}$ converges to zero.

9 Experimental Results

9.1 Constraints and Algorithms

The results of this paper are not bound to any particular algorithm and this paper is not concerned with convergence properties or how to find global minima. The aim of this section is, therefore, the verification of our theory by implementing a basic iterative factorization procedure and showing the algorithm's behaviour for different choices of the depth constraints, in terms of finding the correct solutions.

Given the image data matrix $[\mathbf{x}_{ij}]$ and a constraint space C, we estimate the depths through the following optimization problem:

$$\min_{\hat{\Lambda},\hat{\mathtt{P}},\hat{\mathtt{X}}} \left\| \hat{\Lambda} \odot [\mathbf{x}_{ij}] - \hat{\mathtt{P}}\hat{\mathtt{X}} \right\|_F \quad \text{subject to} \quad \hat{\Lambda} \in C, \tag{57}$$

where $\hat{\Lambda} \in \mathbb{R}^{m\times n}$, $\hat{\mathtt{X}} \in \mathbb{R}^{m\times 4}$ and $\hat{\mathtt{P}} \in \mathbb{R}^{4\times n}$ for a configuration of m views and n points. Clearly, when the data is noise-free (that is $\mathbf{x}_{ij}$ exactly equals $\mathtt{P}_i\mathbf{X}_j/\lambda_{ij}$ for all i, j), and the constraint space C is inclusive, the above problem has global minima with zero target value, including the correct solutions. If the constraint space is also exclusive, and therefore is *reconstruction friendly*, the global minima contain only the correct solutions for which $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathtt{X}}_j\})$ are projectively equivalent to the true configuration $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$.

To make a clear comparison, among many different possible choices for depth constraints, we choose only four, each representing one class of constraints discussed before. A schema of these four constraints is depicted in Fig. 11. The first two constraints are linear equality ones and the next two are examples of compact constraint spaces. The first constraint, abbreviated as ES-MASK is a masked constraint which fixes some elements of $\hat{\Lambda}$ according to $\mathtt{M} \circ \hat{\Lambda} = \mathtt{M}$ for a mask $\mathtt{M}$. ES-MASK uses a specific exclusive edgeless step-like mask. In the case of a fat depth matrix ($n \geq m$), this

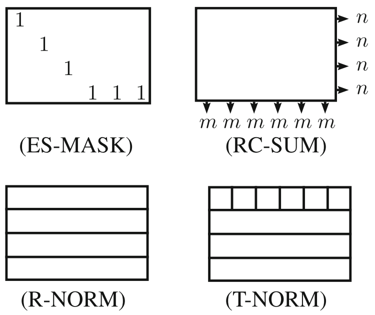

Fig. 11 Four constraints implemented for the experiments. ES-MASK is a masked constraint with an edgeless step-like mask $\mathtt{M}$. The constraint fixes some elements of $\hat{\Lambda}$ according to $\mathtt{M} \circ \hat{\Lambda} = \mathtt{M}$. RC-SUM fixes *row* and *column* sums according to $\hat{\Lambda}\mathbf{1}_n = n\mathbf{1}_m$, $\hat{\Lambda}^T\mathbf{1}_m = m\mathbf{1}_n$. R-NORM fixes a weighted l^2-norm of each *rows* of $\hat{\Lambda}$, and T-NORM fixes a weighted l^2-norm of tiles of $\hat{\Lambda}$

mask is the horizontal concatenation of an $m \times m$ identity matrix and an $m \times (n-m)$ matrix whose last row consists of ones and its rest of elements are zero (see Fig. 11). A similar choice can be made for tall matrices. We choose the edgeless step-like mask as our experiments show that it converges more quickly than the edged version (see Sect. 6.2.3 for a discussion). The second-constraint, RC-SUM, makes the rows of $\hat{\Lambda}$ sum up to n and its columns sum up to m, that is $\hat{\Lambda}\mathbf{1}_n = n\mathbf{1}_m$, $\hat{\Lambda}^T\mathbf{1}_m = m\mathbf{1}_n$ (Sect. 6.2.1). The third constraint, R-NORM, requires rows of the depth matrix to have a unit norm (Sect. 6.1.3). The final constraint, T-norm, is requiring tiles of the depth matrix to have a unit norm (Sect. 6.1.4), where the tiling is done according to Fig. 11. The last two constraints can be considered as examples of tiled constraints (see Sect. 6.1.4). The norm used for these two constraints are weighted l^2-norms with special weights as follows: For an $m' \times n'$ tile ($m' = 1$ or $n' = 1$) in the depth matrix, the constraint is that the corresponding $3m' \times n'$ block in $\hat{\Lambda} \odot [\mathbf{x}_{ij}]$ has a unit Frobenius norm, which amounts to a unit weighted l^2-norm for the corresponding $m' \times n'$ block of $\hat{\Lambda}$.

The optimization problem (57) is hard to solve. Here, we try to solve it by alternatingly minimizing over different sets of variables. With linear equality constraints, we consider two algorithms for the minimization problem (57):

(A1) Alternate between minimizing with respect to $\hat{\Lambda}$ subject to the constraint $\hat{\Lambda} \in C$, and minimizing with respect to $(\hat{\mathtt{X}}, \hat{\mathtt{P}})$.
(A2) Alternate between minimizing with respect to $(\hat{\Lambda}, \hat{\mathtt{P}})$, and minimizing with respect to $(\hat{\Lambda}, \hat{\mathtt{X}})$, subject to $\hat{\Lambda} \in C$ in both cases.

Notice that, the minimization with respect to $\hat{\Lambda}$, $(\hat{\Lambda}, \hat{\mathtt{P}})$ or $(\hat{\Lambda}, \hat{\mathtt{X}})$ is nothing but minimizing a positive definite quadratic form with respect to a linear equality constraint, which has a closed-form solution. Minimizing with respect to $(\hat{\mathtt{X}}, \hat{\mathtt{P}})$ can be done by a rank-4 SVD thresholding of $\hat{\Lambda} \odot [\mathbf{x}_{ij}]$ and factorizing the rank-4 matrix as $\hat{\mathtt{P}}\hat{\mathtt{X}}$. While each iteration of (A2) is usually more complicated and time-consuming compared to (A1), our experiments show that, generally, (A2) results in faster convergence. Another advantage of (A2) is that it can be readily adapted for the when there is missing data (see Hartley and Schaffalizky 2003) for the case of affine camera model). In our experiments, we use (A2) for optimizing with respect to ES-MASK as it converges more quickly. For RC-SUM in most of the cases (A1) and (A2) both give the correct result. However, it appears that (A2) is relatively more prone to converge to a cross-shaped solution, which is allowed by this constraint. Therefore, for RC-SUM we use (A1). We will provide a brief comparison between (A1) and (A2) in the next subsection.

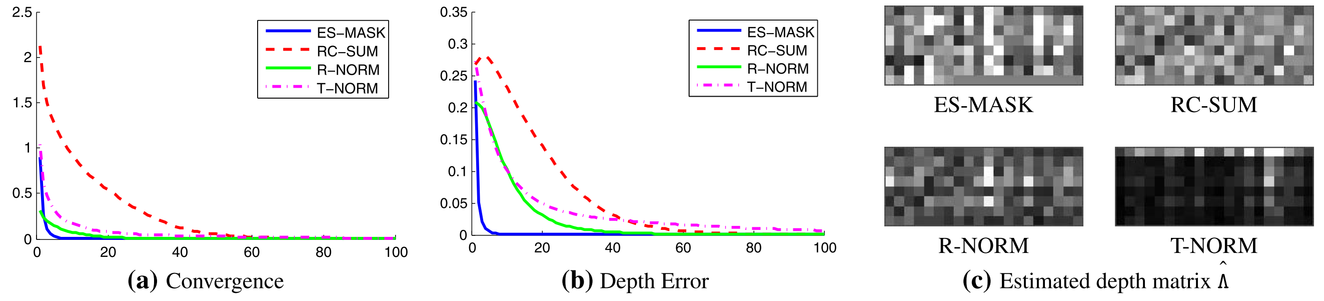

Fig. 12 An example where all algorithms converge to a correct solution. **a** Shows all the four cases have converged to a global minimum, **b** shows that all the four cases have obtained the true depths up to diagonal equivalence, and **c** confirms this by showing that the depth matrix $\hat{\Lambda}$ satisfies (D1–D3). In **c** the *gray*-level of the image at different locations represents the absolute value of the corresponding element in $\hat{\Lambda}$

The last two constraints are both examples of tiling constraints. Our method for optimizing (57) is to alternatingly minimize with respect to $\hat{\Lambda}$ and then with respect to $(\hat{\mathtt{X}}, \hat{\mathtt{P}})$. The latter is done by a rank-4 SVD thresholding of $\hat{\Lambda} \odot [\mathbf{x}_{ij}]$ and factorization. For the former step, we fix $\hat{\mathtt{P}}\hat{\mathtt{X}}$ and minimize $\left\| \hat{\Lambda} \odot [\mathbf{x}_{ij}] - \hat{\mathtt{P}}\hat{\mathtt{X}} \right\|_F$ subject to the constraint that for each $m' \times n'$ tile of $\hat{\Lambda}$, the corresponding $3m' \times n'$ block of $\hat{\Lambda} \odot [\mathbf{x}_{ij}]$ has unit Frobenius norm. This means that, each tile of $\hat{\Lambda}$ can be optimized separately. Showing by $\hat{\boldsymbol{\lambda}}$, the vector of elements of $\hat{\Lambda}$ belonging to a certain tile, the corresponding optimization problem for this tile is in the form of $\min_{\hat{\boldsymbol{\lambda}}} \|\mathtt{A}\hat{\boldsymbol{\lambda}} - \mathbf{b}\|_2$ with respect to $\|\mathtt{A}\hat{\boldsymbol{\lambda}}\|_2 = 1$ for some matrix $\mathtt{A}$ and some vector $\mathbf{b}$. This problem has a closed-form solution.

9.2 Synthetic Data

We take a configuration of 8 views and 20 points. The elements of the matrices $\mathtt{P}_i$ and points $\mathbf{X}_j$ are sampled according to a standard normal distribution. The depths are taken to be $\lambda_{ij} = 3 + \eta_{ij}$, where the η_{ij}-s are sampled from a standard normal distribution. This way we can get a fairly wide range of depths. Negative depths are not allowed, and if they happen, we repeat the sampling. This is mainly because of the fact that for the RC-SUM constraint, the inclusiveness is only proved for positive depths. The image data is calculated according to $\mathbf{x}_{ij} = \mathtt{P}_i \mathbf{X}_j / \lambda_{ij}$, with no added error. Notice that here, unlike in the case of real data in the next subsection, we do not require the last element of the $\mathbf{X}_j$-s and the $\mathbf{x}_{ij}$-s to be 1, and consider the projective factorization problem in its general algebraic form.

In each case, we plot the convergence graph, which is the value of the target function $\left\| \hat{\Lambda} \odot [\mathbf{x}_{ij}] - \hat{\mathtt{P}}\hat{\mathtt{X}} \right\|_F$ throughout iterations, followed by a graph of depth error. To deal with diagonal ambiguity of the depth matrix, the depth error is calculated as $\left\| \Lambda - \operatorname{diag}(\boldsymbol{\tau})\, \hat{\Lambda}\, \operatorname{diag}(\boldsymbol{\nu}) \right\|$, where $\boldsymbol{\tau}$ and $\boldsymbol{\nu}$ are set such that $\operatorname{diag}(\boldsymbol{\tau})\, \hat{\Lambda}\, \operatorname{diag}(\boldsymbol{\nu})$ has the same row norms and column norms as Λ. This can be done using Sinkhorn's algorithm as described in Sect. 6.1.2. Finally, for each constraint we depict the estimated depth matrix $\hat{\Lambda}$ as a grayscale image whose intensity values show the absolute values of the elements of $\hat{\Lambda}$.

In the first example, we set the initial value of $\hat{\Lambda}$ to $\mathtt{1}_{m\times n}$ which is a matrix of all ones. The results for one run of the algorithm are shown in Fig. 12. Figure 12a shows that the algorithm has converged to a global minimum for all four constraints. Figure 12b shows that in all four cases the algorithm has converged to a correct solution. Figure 12c confirms this by showing that in no case the algorithm has converged to a cross-shaped solution or a solution with zero rows or zero columns.

In the second test, we set the initial value of $\hat{\Lambda}$ to be 1 at the first row and 10th column, and 0.02 elsewhere. This makes the initial $\hat{\Lambda}$ close to a cross-shaped matrix. The result is shown in Fig. 13. According to Fig. 13a, in all cases the target error has converged to zero, meaning that a solution is found for the factorization problem $\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\hat{\mathtt{X}}$. Figure 13b, shows that for the constraint ES-MASK and RC-SUM, the algorithm gives a correct solution, however, for R-NORM and T-NORM, it has converged to a wrong solution. Figure 13c supports this by showing that the algorithm has converged to a cross-shaped solution for R-NORM and T-NORM. Although, the constraint RC-SUM allows for cross-shaped configurations, according to our discussion in Sect. 6.2.1, it is unlikely for the algorithm to converge to a cross-shaped solution if the initial solution has all positive numbers (see Fig. 8). However, our experiments show that if we start from a configuration close to the cross-shaped solution of the constraint RC-SUM (with a negative element at the centre of the cross), the algorithm will converge to a cross-shaped configuration.

Next, we provide a comparison between the algorithms (A1) and (A2), introduced in Sect. 9.1, for the linear equality constraints ES-MASK and RC-SUM. We run 100 trials for each algorithm with the same random setup as the previous experiment. Each algorithm runs until a cost of less than 10^{-6} is obtained or 20000 iterations are reached. Table 1

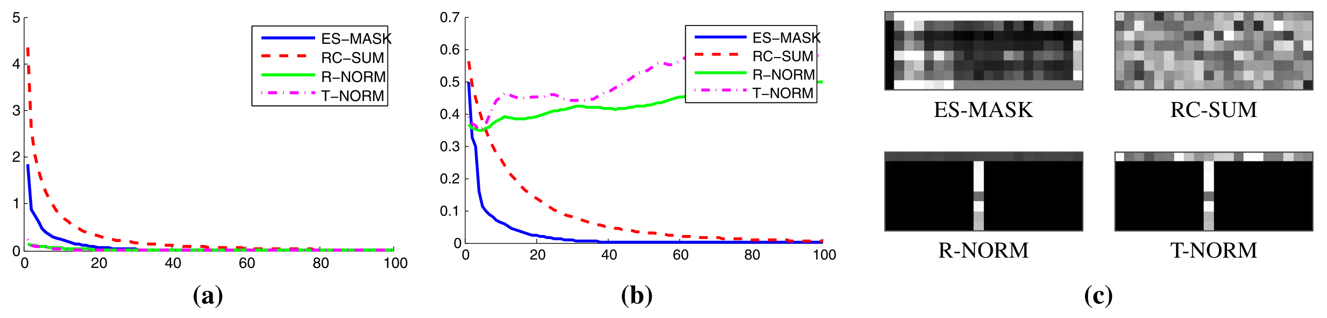

Fig. 13 **a** The target error in all cases has converged to zero, **b** the depth error has converged to zero only for ES-MASK and RC-SUM, meaning that only ES-MASK and RC-SUM have converged to a correct solution, **c** confirms this by showing that R-NORM and T-NORM have converged to cross-shaped solutions

Table 1 Comparison of the algorithms (A1) and (A2) for each of the linear equality constraints ES-MASK and RC-SUM

Algorithm	ES-MASK		RC-SUM	
	A1	A2	A1	A2
Iterations (med)	2256	42	421	34
Iter. time (mean)	0.2 ms	2.9 ms	0.5 ms	3.2 ms
Total time (med)	523 ms	138 ms	254 ms	117 ms

The results are obtained over 100 trials per algorithm. Each algorithm stops when it achieves a cost of less than 10^{-6}, or reaches 20,000 iterations. The first row of the table is the median of the number of iterations taken by each algorithm. The second row is the average time of a single iteration. The last row is the median of the total time of each trial

summarizes the results. In general, we can say (A2) spends more time at each iteration, however, it converges in significantly fewer iterations than (A1). Overall, (A2) has a faster convergence.

9.3 Real Data

We use the Model House data set provided by the Visual Geometry Group at Oxford University.[9] As our theory does not deal with the case of missing data, from the data matrix we choose a block of 8 views and 19 points for which there is no missing data. Here, the true depths are not available. Thus, to see if the algorithm has converged to a correct solution, we use a special version of the reprojection error. The basic reprojection error is $\sum_{ij} \|\mathbf{x}_{ij} - \alpha_{ij}\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j\|$ where for each i and j, α_{ij} is chosen such that the third entry of the vector $\alpha_{ij}\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j$ is equal to the third entry of $\mathbf{x}_{ij}$, which is 1 in this case. However, as this can cause fluctuations in the convergence graph at the points where the last element of $\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j$ is close to zero, we instead choose each α_{ij} such that it minimizes $\|\mathbf{x}_{ij} - \alpha_{ij}\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j\|$.

Figure 14 shows one run of the algorithm for each of the four constraints starting from $\hat{\Lambda} = \mathbf{1}_{m\times n}$. It can be seen that for all the constraints the algorithm has converged to a solution with a very small error. Figure 14b shows that all of them have converged to something close to a correct solution. This is affirmed by Fig. 14c, showing that all solutions satisfy depth conditions (D1–D3). Comparing Fig. 14c with Fig. 12c one can see that the depths in Fig. 14c are more uniform. One reason is that the true depths in this experiment are relatively close together compared to the case of synthetic data. Except, T-NORM, all the other constraints tend to somewhat preserve this uniformity, especially when the initial solution is a uniform choice like $\mathbf{1}_{m\times n}$.

In the second test we start from an initial $\hat{\Lambda}$ which is close to a cross-shaped matrix, as chosen in the second test for the synthetic data. The result is shown in Fig. 15. Figure 15a shows that the RC-SUM has not converged to a solution with a small target error, but the other 3 constraints seem to have.[10] Therefore, we cannot say anything about RC-SUM. Figure 15b shows that R-NORM and T-NORM did not converge to a correct solution. Figure 15c confirms this by showing that R-NORM and T-NORM have converged to (something close to) a cross-shaped solution.

10 Conclusion and Future Work

We proved a more general version of the Projective Reconstruction Theorem, which is well suited to the choice and analysis of depth constraints for factorization-based projective reconstruction algorithms. We also demonstrated how our theoretical results can be used for the analysis of existing depth constraints used for the factorization-based algorithms and also for the design of new types of depth constraints.

The main result of our paper is that the false solutions to the factorization problem $\hat{\Lambda} \odot [\mathbf{x}_{ij}] = \hat{\mathtt{P}}\,\hat{\mathtt{X}}$, are restricted to the cases where $\hat{\Lambda}$ has zero rows or zero columns and also

[9] http://www.robots.ox.ac.uk/~vgg/data/data-mview.html.

[10] Notice that the scale of the vertical axis in Fig. 15a is different from that of Fig. 14a.

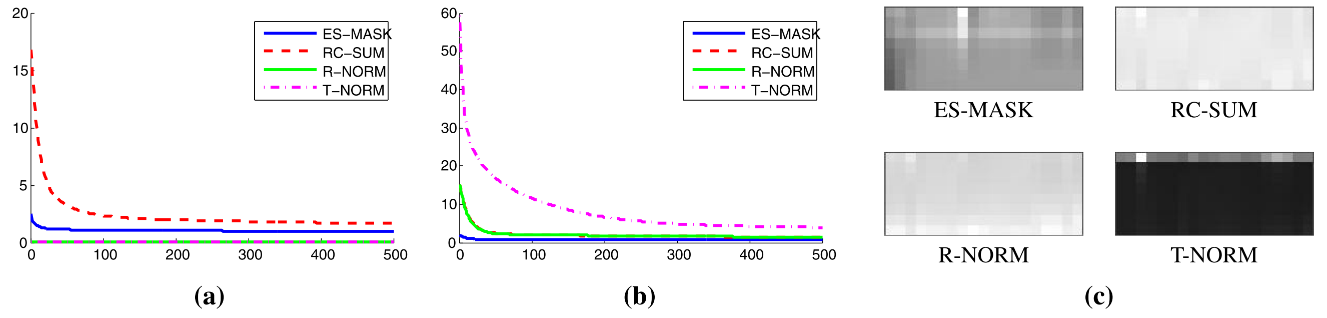

Fig. 14 An example where all algorithms converge to a solution with a very small target value which is also close to a correct solution. In **c**, one can observe a bright strip on the top of the corresponding image of T-NORM. The reason is that T-NORM forces each elements of the *top row* of $\hat{\Lambda}$ to have a unit (weighted l^2) norm, while for the other *rows*, the whole *row* is required to have a unit norm. See Fig. 11(T-NORM)

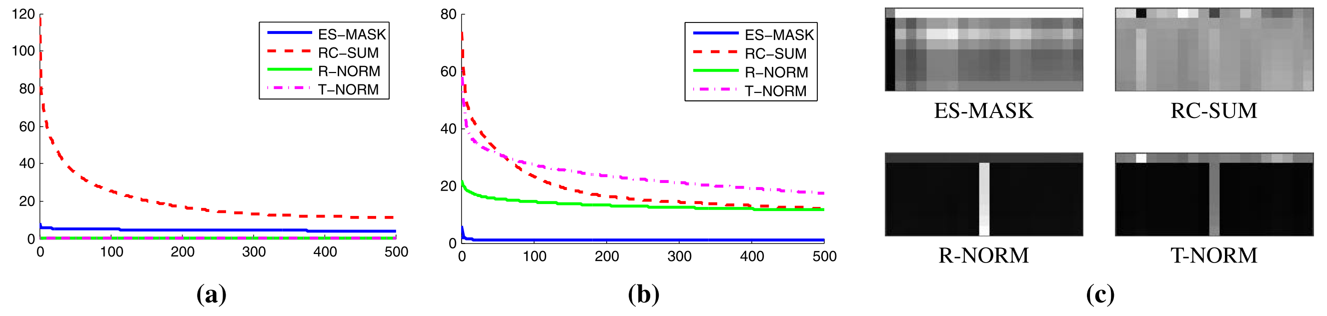

Fig. 15 An example where the algorithms are started from an initial solution which is close to a *cross-shaped* matrix. **a** shows that RC-SUM has not converged to a solution with a small target error. R-NORM and T-NORM have converged to something with a small target value, but did not get close to a correct solution. This is obvious from (**b**) and (**c**)

when it has a cross-shaped structure. Any solution which rules out these cases is a correct solution.

We presented a class of linear equality constraints which are able to rule out all the degenerate false solutions. Our experiments also showed that choosing a good initial solution can result in finding the correct depths, even with some of the constraints that do not rule out all the false solutions.

Indeed, this paper is just a first step on this matter. Hence, many practical issues have been disregarded. For example, here it has been assumed that all points are visible in all views. A very important extension to this work is therefore considering the case of incomplete image data. Another assumption here was that the image data is not contaminated with noise. The case of noisy data is another major issue which must be addressed in future work.

Another important generalization of this work is the extension to higher dimensional projections, for example projections from $\mathbb{P}^r$ to $\mathbb{P}^s$, or more generally, when for the i-th view the projection is $\mathbb{P}^r \to \mathbb{P}^{s_i}$. This extension is important because it has applications in problems like projective motion segmentation and non-rigid reconstruction.

Yet another follow-up is the study of the convergence of specific factorization-based algorithms for each of the constraints and the design of constraints with desirable convergence properties. For example, we know that certain convergence properties can be proved for certain algorithms with compact constraint spaces. However, guaranteed convergence to a global minimum is still an unsolved problem. Another interesting problem to solve is to find compact constraints which are reconstruction friendly, allow for efficient factorization-based algorithms, and give a descent move at every iteration of the algorithm.

Appendix 1: The Triangulation Problem

Triangulation is the process of determining the location of a 3D point given its images in two or more cameras with known camera matrices. The following lemma states that the solution to triangulation is unique in generic cases:

Lemma 18 (Triangulation) *Consider two full-row-rank camera matrices* $\mathtt{P}_1, \mathtt{P}_2 \in \mathbb{R}^{3\times4}$*, two points* $\mathbf{X}, \mathbf{Y} \in \mathbb{R}^4$*, and scalars* $\hat{\lambda}_1$ *and* $\hat{\lambda}_2$ *such that the vector* $(\hat{\lambda}_1, \hat{\lambda}_2)$ *is nonzero, for which the relations*

$$\mathtt{P}_1\mathbf{Y} = \hat{\lambda}_1\mathtt{P}_1\mathbf{X} \quad (58)$$

$$\mathtt{P}_2\mathbf{Y} = \hat{\lambda}_2\mathtt{P}_2\mathbf{X} \quad (59)$$

hold. Take nonzero vectors $\mathbf{C}_1 \in \mathcal{N}(\mathtt{P}_1)$ and $\mathbf{C}_2 \in \mathcal{N}(\mathtt{P}_2)$. If the three vectors $\mathbf{C}_1$, $\mathbf{C}_2$ and $\mathbf{X}$ are linearly independent, then $\mathbf{Y}$ is equal to $\mathbf{X}$ up to a nonzero scaling factor.

Notice that the condition of $\mathbf{C}_1$, $\mathbf{C}_2$ and $\mathbf{X}$ being linearly independent means that the two camera centres are distinct and $\mathbf{X}$ does not lie on the projective line joining them (see footnote 4). A geometric proof of this is given in (Hartley and Zisserman, 2004, Theorem 10.1). Here, we give an algebraic proof as one might argue that Hartley and Zisserman (2004) has used projective equality relations which cannot be fully translated to our affine space equations since we do not assume that $\hat{\lambda}_1$ and $\hat{\lambda}_2$ are both nonzero in (58) and (59).

Proof Since $\mathtt{P}_1$ and $\mathtt{P}_2$ have full row rank they have a 1D null space. Thus, relations (58) and (59) respectively imply

$$\mathbf{Y} = \alpha_1\mathbf{C}_1 + \hat{\lambda}_1\mathbf{X}, \tag{60}$$

$$\mathbf{Y} = \alpha_2\mathbf{C}_2 + \hat{\lambda}_2\mathbf{X}, \tag{61}$$

for some scalars α_1 and α_2. These give $\alpha_1\mathbf{C}_1 + \hat{\lambda}_1\mathbf{X} = \alpha_2\mathbf{C}_2 + \hat{\lambda}_2\mathbf{X}$ or

$$\alpha_1\mathbf{C}_1 - \alpha_2\mathbf{C}_2 + (\hat{\lambda}_1 - \hat{\lambda}_2)\mathbf{X} = 0 \tag{62}$$

As the three vectors $\mathbf{C}_1$, $\mathbf{C}_2$ and $\mathbf{X}$ are linearly independent, (62) implies that $\alpha_1 = 0$, $\alpha_2 = 0$ and $\hat{\lambda}_1 = \hat{\lambda}_2$. Define $\nu \overset{\text{def}}{=} \hat{\lambda}_1 = \hat{\lambda}_2$. Then, from (60) we have $\mathbf{Y} = \nu\mathbf{X}$. Moreover, ν must be nonzero as the lemma assumes that the vector $(\hat{\lambda}_1, \hat{\lambda}_2) = (\nu, \nu)$ is nonzero. □

Appendix 2: The Camera Resectioning Problem

Camera resectioning is the task of computing camera parameters given the 3D points and their images. It can be shown that with sufficient 3D points in general locations, the camera matrix can be uniquely determined up to scale (Hartley and Zisserman 2004). Here, we consider a slightly revised version of this problem, which fits our case where the estimated projective depths are not assumed to be all nonzero and the second (estimated) set of camera matrices need not be assumed to have full rank.

Lemma 19 (Resectioning) *Consider a 3×4 matrix $\mathtt{Q}$ of rank 3 and a set of points $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_p$ such that for a nonzero vector $\mathbf{C} \in \mathcal{N}(\mathtt{Q})$ we have*

(C1) *Any four vectors among $\mathbf{C}, \mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_p$ are linearly independent, and*

(C2) *the set of points $\{\mathbf{C}, \mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_n\}$ do not lie on a twisted cubic (see footnote 4) or any of the degenerate critical sets resulting in a resection ambiguity (set out in Hartley and Zisserman 2004, Sect.22.1).*

Now, for any $\hat{\mathtt{Q}} \in \mathbb{R}^{3\times 4}$ if we have

$$\alpha_j\mathtt{Q}\mathbf{X}_j = \beta_j\hat{\mathtt{Q}}\mathbf{X}_j \tag{63}$$

for all $j = 1, 2, \ldots, p$ where scalars α_j and β_j are such that the vector (α_j, β_j) is nonzero for all j, then $\hat{\mathtt{Q}} = a\mathtt{Q}$ for some scalar a.

Proof First, since 6 points in general position completely specify a twisted cubic (Semple and Kneebone 1952), (C2) implies that $p + 1 \geq 7$, or $p \geq 6$.

If $\hat{\mathtt{Q}} = \mathtt{0}$, then $\hat{\mathtt{Q}} = a\mathtt{Q}$ with $a = 0$, proving the claim of the lemma. Thus, in what follows we only consider the case of $\hat{\mathtt{Q}} \neq \mathtt{0}$.

By (C1), for all j we have $\mathtt{Q}\mathbf{X}_j \neq \mathbf{0}$. Therefore, $\beta_j \neq 0$, as otherwise if $\beta_j = 0$ from $(\alpha_j, \beta_j)^T \neq \mathbf{0}$ we would have $\alpha \neq 0$ and therefore $\mathbf{0} = \beta_j\hat{\mathtt{Q}}\mathbf{X}_j = \alpha_j\mathtt{Q}\mathbf{X}_j \neq \mathbf{0}$, which is a contradiction. From $\beta_j \neq 0$ and (63) it follows that if $\alpha_j = 0$ for some j, then $\mathbf{X}_j \in \mathcal{N}(\hat{\mathtt{Q}})$. Now, if for 4 indices j we have $\alpha_j = 0$, from (C1) it follows that $\hat{\mathtt{Q}}$ has a 4D null space, or equivalently $\hat{\mathtt{Q}} = \mathtt{0}$. Since we excluded this case, we conclude that there are less than 4 zero-valued α_j-s. As $p \geq 6$, it follows that there are at least three nonzero α_j-s, namely α_{j_1}, α_{j_2} and α_{j_3}. Since β_j-s are all nonzero, $\alpha_j \neq 0$ along with (63) implies that $\mathtt{Q}\mathbf{X}_j$ is in $\mathscr{C}(\hat{\mathtt{Q}})$, the column space of $\hat{\mathtt{Q}}$. Therefore, we have $\mathrm{span}(\mathtt{Q}\mathbf{X}_{j_1}, \mathtt{Q}\mathbf{X}_{j_2}, \mathtt{Q}\mathbf{X}_{j_3}) \subseteq \mathscr{C}(\hat{\mathtt{Q}})$. From (C1) we know that $\mathrm{span}(\mathbf{X}_{j_1}, \mathbf{X}_{j_2}, \mathbf{X}_{j_3})$ is 3-dimensional and does not contain the null space of $\mathtt{Q}$. Therefore, $\mathrm{span}(\mathtt{Q}\mathbf{X}_{j_1}, \mathtt{Q}\mathbf{X}_{j_2}, \mathtt{Q}\mathbf{X}_{j_3})$ is also 3-dimensional. From $\mathrm{span}(\mathtt{Q}\mathbf{X}_{j_1}, \mathtt{Q}\mathbf{X}_{j_2}, \mathtt{Q}\mathbf{X}_{j_3}) \subseteq \mathscr{C}(\hat{\mathtt{Q}})$ then we conclude that $\hat{\mathtt{Q}}$ has full row rank.

As $\mathrm{rank}(\hat{\mathtt{Q}}) = 3$, we can consider it as a proper camera matrix in multiple view geometry, talking about its camera centre represented by its null space. Therefore, for two camera matrices $\mathtt{Q}$ and $\hat{\mathtt{Q}}$ and all the points $\mathbf{X}_j$ for which $\alpha_j \neq 0$ we can apply the results of the classic camera resectioning problem: It is known that for two (up to scale) distinct camera matrices $\mathtt{Q}$ and $\hat{\mathtt{Q}}$ to see the points $\mathbf{X}_j$ equally up to a possible nonzero scaling factor, the points $\mathbf{X}_j$ and the camera centres must lie on a common twisted cubic (or possibly some other specific degenerate sets, see (Hartley and Zisserman 2004; Buchanan 1988)).

Notice that, as $\mathrm{rank}(\hat{\mathtt{Q}}) = 3$, (C1) implies that among the points $\mathbf{X}_j$ at most one lies on the null-space of $\hat{\mathtt{Q}}$ and therefore, by (63) we can say that at most one α_j can be zero. By possibly relabeling the points we assume that $\alpha_1, \ldots, \alpha_{p-1}$ are all nonzero.

Now to get a contradiction, assume that there is a resection ambiguity. We consider two cases namely $\alpha_p \neq 0$ and $\alpha_p = 0$. If $\alpha_p \neq 0$ then by $\alpha_j\mathtt{Q}\mathbf{X}_j = \beta_j\hat{\mathtt{Q}}\mathbf{X}_j$ we know that $\mathbf{X}_1, \ldots, \mathbf{X}_p$ are viewed equally up to scale by both $\mathtt{Q}$ and $\hat{\mathtt{Q}}$ and thus $\mathbf{X}_1, \ldots, \mathbf{X}_6$ along with the camera centre of $\mathtt{Q}$ must lie on a twisted cubic (or other degenerate sets leading to a resection ambiguity), which is impossible due to (C2). If

$\alpha_6 = 0$, implying $\mathbf{X}_6 \in \mathcal{N}(\hat{\mathtt{Q}})$, then again the camera center of $\mathtt{Q}$, $\mathbf{X}_1, \ldots, \mathbf{X}_5$ and $\mathbf{X}_6$ (this time as the camera centre of $\hat{\mathtt{Q}}$) must lie on a twisted cubic (or the degenerate sets), contradicting with (C2). Hence there can be no resection ambiguity and $\mathtt{Q}$ and $\hat{\mathtt{Q}}$ must be equal up to a scaling factor. □

Appendix 3: Proof of Lemma 1

Proof of Lemma 1 We need to prove that under assumptions of Lemma 1 and the relations

$$\lambda_{ij}\mathbf{x}_{ij} = \mathtt{P}_i\mathbf{X}_j \tag{64}$$
$$\hat{\lambda}_{ij}\mathbf{x}_{ij} = \hat{\mathtt{P}}_i\hat{\mathbf{X}}_j \tag{65}$$

$(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent if and only if the matrices Λ and $\hat{\Lambda}$ are diagonally equivalent.

First, assume that $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent. Then, there exist nonzero scalars $\tau_1, \tau_2, \ldots, \tau_m$ and $\nu_1, \nu_2, \ldots, \nu_n$ and an invertible matrix $\mathtt{H}$ such that (5) and (6) hold. Therefore we have

$$\hat{\lambda}_{ij}\mathtt{P}_i\mathbf{X}_j = \hat{\lambda}_{ij}\lambda_{ij}\mathbf{x}_{ij} = \lambda_{ij}\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j$$
$$= \lambda_{ij}\nu_j\tau_i\,\mathtt{P}_i\,\mathtt{H}\,\mathtt{H}^{-1}\,\mathbf{X}_j = \lambda_{ij}\nu_j\tau_i\,\mathtt{P}_i\,\mathbf{X}_j.$$

where the first, second and third equations above hold respectively from (64), (65) and (5) and (6) together. By condition (i) in the lemma, that is $\mathtt{P}_i\,\mathbf{X}_j \neq \mathbf{0}$, we have $\hat{\lambda}_{ij} = \lambda_{ij}\nu_j\tau_i$ for all i and j. This is equivalent to (7) and hence Λ and $\hat{\Lambda}$ are diagonally equivalent.

To prove the other direction, assume that Λ and $\hat{\Lambda}$ are diagonally equivalent. Then from (7) we have $\hat{\lambda}_{ij} = \lambda_{ij}\nu_j\tau_i$. This along with (64) and (65) gives

$$\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j = \hat{\lambda}_{ij}\mathbf{x}_{ij} = \lambda_{ij}\nu_j\tau_i\mathbf{x}_{ij} = \tau_i\nu_j\mathtt{P}_i\mathbf{X}_j = (\tau_i\mathtt{P}_i)(\nu_j\mathbf{X}_j) \tag{66}$$

for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. Let $\mathtt{Q}_i = \tau_i\mathtt{P}_i$ and $\mathbf{Y}_j = \nu_j\mathbf{X}_j$, so we have $\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j = \mathtt{Q}_i\mathbf{Y}_j$. Denote by $\mathtt{Q}$ and $\hat{\mathtt{P}}$ the vertical concatenations of $\mathtt{Q}_i$-s and $\hat{\mathtt{P}}_i$-s respectively and denote by $\mathtt{Y}$ and $\hat{\mathtt{X}}$ respectively the horizontal concatenations of $\mathbf{Y}_j$-s and $\hat{\mathbf{X}}_j$-s. From $\hat{\mathtt{P}}_i\hat{\mathbf{X}}_j = \mathtt{Q}_i\mathbf{Y}_j$ we have

$$\hat{\mathtt{P}}\hat{\mathtt{X}} = \mathtt{Q}\mathtt{Y} \overset{\text{def}}{=} \mathtt{A}. \tag{67}$$

From conditions (ii) and (iii) in the lemma along with the fact that τ_i and ν_j are nonzero, we can conclude that $\mathtt{Q}$ has full column rank and $\mathtt{Y}$ has full row rank. Therefore, $\mathtt{A} \overset{\text{def}}{=} \mathtt{QY}$ has rank 4 and the $3m \times 4$ and $4 \times n$ matrices $\hat{\mathtt{P}}$ and $\hat{\mathtt{X}}$ must be full-column- and full-row-rank matrices respectively. As $\mathtt{QY}$ and $\hat{\mathtt{P}}\hat{\mathtt{X}}$ are two rank-r factorizations of $\mathtt{A}$, having $\hat{\mathtt{P}} = \mathtt{Q}\,\mathtt{H}$ and $\hat{\mathtt{X}} = \mathtt{H}^{-1}\mathtt{Y}$ for some invertible matrix $\mathtt{H}$ is the only possibility[11]. This is the same thing as

$$\hat{\mathtt{P}}_i = \mathtt{Q}_i\mathtt{H} = \tau_i\mathtt{P}_i\mathtt{H} \tag{68}$$
$$\hat{\mathbf{X}}_j = \mathtt{H}^{-1}\mathbf{Y}_j = \nu_j\mathtt{H}^{-1}\mathbf{X}_j \tag{69}$$

Thus, $(\{\mathtt{P}_i\}, \{\mathbf{X}_j\})$ and $(\{\hat{\mathtt{P}}_i\}, \{\hat{\mathbf{X}}_j\})$ are projectively equivalent. □

[11] The proof is quite simple: The column space of $\mathtt{Q}$, $\hat{\mathtt{P}}$ and $\mathtt{A}$ must be equal and therefore we have $\hat{\mathtt{P}} = \mathtt{QH}$ for some invertible 4×4 matrix $\mathtt{H}$. Similarly, we can argue that $\hat{\mathtt{X}} = \mathtt{GY}$ for some invertible $\mathtt{G}$. Therefore, we have $\mathtt{Q}\,\mathtt{Y} = \mathtt{Q}\,\mathtt{H}\,\mathtt{G}\,\mathtt{Y}$. As $\mathtt{Q}$ has full column rank and $\mathtt{Y}$ has full row rank, the above implies $\mathtt{H}\,\mathtt{G} = \mathtt{I}$ and hence, $\mathtt{G} = \mathtt{H}^{-1}$.

References

Angst, R., Zach, C., & Pollefeys, M. (2011). The generalized trace-norm and its application to structure-from-motion problems. In: *Proceedings of the IEEE International Conference on Computer Vision* (ICCV) 2011, (pp. 2502–2509). doi:10.1109/ICCV.2011.6126536.

Buchanan, T. (1988). The twisted cubic and camera calibration. *Computer Vision, Graphics, and Image Processing*, *42*(1), 130–132.

Dai, Y., Li, H., & He, M. (2010). Element-wise factorization for n-view projective reconstruction. In: *Proceedings of the 11th European conference on Computer vision: Part IV*, ECCV'10, (pp. 396–409). Berlin: Springer.

Dai, Y., Li, H., & He, M. (2013). Projective multi-view structure and motion from element-wise factorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence 35*(9), 2238–2251. doi:10.1109/TPAMI.2013.20.

Hartley, R., & Kahl, F. (2007). Critical configurations for projective reconstruction from multiple views. *International Journal of Computer Vision*, *71*(1), 5–47. doi:10.1007/s11263-005-4796-1.

Hartley, R., & Schaffalizky, F. (2003). PowerFactorization: 3D reconstruction with missing or uncertain data. In: Proceedings of the Australia-Japan Advanced Workshop on Computer Vision.

Hartley, R. I., & Zisserman, A. (2004). *Multiple View Geometry in Computer Vision* (2nd ed.). Cambridge, MA: Cambridge University Press. ISBN: 0521540518.

Heyden, A., Berthilsson, R., & Sparr, G. (1999). An iterative factorization method for projective structure and motion from image sequences. *Image and Vision Computing*, *17*(13), 981–991.

Lin, Z., Chen, M., & Wu, L. (2010). The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. Analysis math.OC (Technical Report UILU-ENG-09-2215):-09-2215.

Luenberger, D. G. (1984). *Linear and Nonlinear Programming*. Boston, MA: Addison-Wesley Publishing Company.

Mahamud, S., Hebert, M., Omori, Y., & Ponce, J. (2001). Provably-convergent iterative methods for projective structure from motion. In: *Proceedings of the Computer Vision and Pattern Recognition* (CVPR), (pp. 1018–1025). IEEE

Oliensis, J., & Hartley, R. (2007). Iterative extensions of the Sturm/Triggs algorithm: Convergence and nonconvergence. *Pattern Analysis and Machine Intelligence*, *29*(12), 2217–2233. doi:10.1109/TPAMI.2007.1132.

Semple, J., & Kneebone, G. (1952). *Algebraic Projective Geometry*., Oxford Classic Texts in the Physical Sciences Series Oxford: Clarendon Press.

Sinkhorn, R. (1964). A relationship between arbitrary positive matrices and doubly stochastic matrices. *The Annals of Mathematical Statistics*, *35*(2), 876–879.

Sinkhorn, R. (1967). Diagonal equivalence to matrices with prescribed row and column sums. *The American Mathematical Monthly*, *74*(4), 402–405.

Sturm, P.F., & Triggs, B. (1996). A factorization based algorithm for multi-image projective structure and motion. In: *Proceedings of the 4th European Conference on Computer Vision*, ECCV '96, (vol. 2, pp 709–720). London: Springer.

Triggs, B. (1996). Factorization methods for projective structure and motion. In: *Proceedings of the 1996 Conference on Computer Vision and Pattern Recognition* (CVPR '96), (pp. 845). IEEE Computer Society, Washington, DC.

Triggs, B., McLauchlan, P. F., Hartley, R. I., & Fitzgibbon, A. W. (2000). Bundle adjustment - a modern synthesis. In: *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, ICCV '99, (pp 298–372). London: Springer.

Ueshiba, T., & Tomita, F. (1998). A factorization method for projective and euclidean reconstruction from multiple perspective views via iterative depth estimation. *Computer*, *I*, 296–310.

Yang, J., & Yuan, X. (2013). Linearized augmented Lagrangian and alternating direction methods for nuclear norm minimization. *Mathematics of Computation*, *82*(281), 301–329. doi:10.1090/S0025-5718-2012-02598-1.

Zangwill, W. (1969). *Nonlinear Programming: A Unified Approach.*, Prentice-Hall International Series in Management Englewood Cliffs, NJ: Prentice-Hall.