Multi-View Correspondence by Enforcement of Rigidity Constraints

R. Oliveira *, J. Xavier, J. P. Costeira

Instituto de Sistemas e Robótica - Instituto Superior Técnico.
Av. Rovisco Pais, 1049-001 Lisboa Codex, PORTUGAL

Abstract

Establishing the correct correspondence between features in an image set remains a challenging problem amongst computer vision researchers. In fact, the combinatorial nature of feature matching effectively hinders the solution of large scale problems, which have direct applications in important areas such as 3D reconstruction and tracking.

The solution is obtained by imposing a geometric constraint - rigidity - that selects the matching solution resulting in a rank-4 observation matrix. Since this is a global criterion, issues usually associated to local matching algorithms (such as the aperture problem) do not present an obstacle in this case. The use of a geometric constraint of this type assumes that all feature points are visible in every image, so as to obtain a complete observation matrix.

The rank of the observation matrix is a function of the matching solutions associated to each image and as such a simultaneous solution for all frames has to be found. For each frame, correspondence is modeled through a *permutation matrix*, which also allows for the rejection of wrong candidates. Although each image is matched individually, an iterative algorithm is used to integrate correspondence information associated to all remaining images. Each individual matching process results in a linear problem: the reduced computational complexity allows the solution of large problems in an acceptable time interval.

Although the algorithm has intrinsically been designed for calibrated systems, some instances of the uncalibrated case can also be solved provided a convenient bootstrap is available.

Keywords: multi-view correspondence, feature matching, factorization,
constrained optimization

1 Introduction

Establishing correspondences between the features of a pair of images has proven to be an essential task in computer vision. In this article an approach to this problem is presented that focuses on an important geometrical property of many scenes - *rigidity*. Geometrical properties are appealing because they establish a direct relation to one of the foremost applications of this kind of method: 3D reconstruction. In fact, many successful reconstruction algorithms rely on previously computed correspondences to determine the 3D structure of a scene. Clear examples of this are classical factorization algorithms such as [24] and more recent methods as [7], [9] and [23].

The difficulty of the correspondence problem is associated to its combinatorial nature: even when matching a diminute number of features, the number of possible solutions excludes any approach degenerating in an exhaustive search. A further step is taken by simultaneously matching features among multiple frames - several issues arise that still raise the complexity of the problem. Clearly, matching images pairwise in a cascade is not sufficient, since a matching error would propagate to all subsequent images. As a consequence, specific algorithms have to be devised for the multi-view case. Furthermore, the matching of multiple frames allows the inclusion of several constraints that cannot be applied in pairwise matching.

Several models have been proposed in order to obtain a solution with an acceptable computational cost. Most of these solutions involve an optimization approach and vary essentially in the way the matching problem is mapped in order to obtain a cost function that is easily optimized. In [11] and [21], the n-frame correspondence problem is formulated as a maximum-flow problem and is solved through graph cut algorithms. In both cases, essentially photometric cost functions are mapped to the edge capacities of graphs. A different approach involving graphs with application to matching has been presented in [22], where a n-frame matching algorithm is proposed which represents image points (in every frame) and correspondences as, respectively, vertex and edges in a digraph. No graph cut is performed to obtain the solution, as in other graph-based algorithms; instead, a greedy algorithm is used to optimize a gain function (for example, correlation). A similar framework is used in [6], where each vertex is an image region to be matched while edges represent correspondences and are weighted according to region similarity.

* Corresponding Author

Email addresses: rco@isr.ist.utl.pt (R. Oliveira), jxavier@isr.ist.utl.pt (J. Xavier), jpc@isr.ist.utl.pt (J. P. Costeira).

URLs: www.isr.ist.utl.pt/~rco (R. Oliveira), www.isr.ist.utl.pt/~jxavier (J. Xavier), www.isr.ist.utl.pt/~jpc (J. P. Costeira).

A natural way to associate a cost function to the correspondence problem is to exploit a constant global characteristic of an important class of 3D scenes: rigidity. The use of rigidity presents the advantage of leading to intrinsically *global* algorithms; moreover, it naturally overcomes the aperture problem, since features are not characterized by their specific local properties (i.e. texture) but by their *position* (i.e. coordinates), thereby allowing matching over gradient-oriented areas, such as straight lines. This geometric constraint can be translated into a rank constraint [14] on the matrix containing the coordinates of the extracted features (the measurement matrix). Actually, it can be shown that when features in different viewpoints are correctly aligned (and only then) this matrix is highly rank-deficient - [13], [17].

Rank-deficiency has also been exploited in [16] (to find missing data in a measurement matrix) and in [8] (to determine optical flow), although in the latter work the rank constraint applies on flow vectors rather than on image coordinates.

Earlier approaches exploiting rigidity have been made: in [15] an algorithm is devised that checks if corresponding points from two images could be the projection of a rigid scene (no correspondence calculation is made), while in [10] it is used for object detection. In [13] the authors use a cost function based on the determinant of the measurement matrix to match features in a pair of images. This approach, although theoretically sound, has two main shortcomings: it is unable to handle the multi-image case and the cost function is intrinsically non-linear, presenting a high computational burden. The use of a non-linear cost function also presents the additional drawback of introducing local minima in the optimization process, thereby allowing non-optimal solutions to be retrieved. In [17] and [18] the authors presented new algorithms based on an alternative cost function, which will detect rank-deficiency based on the sum of the non-dominant singular values of the measurement matrix. This cost function allows the rigidity constraint to be applied to a *multi-frame system*. To obtain a low computational complexity a rank constraint is imposed iteratively by matching each image individually with the remaining frames. We formalize our criterion in such a way as to allow a solution based on the execution of a set of *linear programs*. This guarantees that the algorithm is computationally feasible even for large-scale problems. This aspect is particularly important since it allows the use of the method for real life problems, which are usually high-dimensional. It should be noted that although the cost function is globally nonlinear, each iteration of the algorithm (tackling the problem associated to a single image) solves a linear problem thus achieving a global solution in each iteration.This advantage also reflects one of the drawbacks - the iterative nature of the algorithm - that does not guarantee convergence.

2 Problem Formulation

In this text a framework is presented that is based on the approach described in [17]. The *n-frame correspondence* problem is herein formulated and solved as a global optimization procedure in the factorization context. In particular, a rank constraint is imposed on the matrix containing the feature coordinates - matrices of this type have been shown to be rank-deficient, provided the camera model is affine. In fact, the rank of these matrices is *at most* four, if degenerate motion in the scene is disregarded. Under the factorization framework, obtaining a rank-4 observation matrix can be interpreted as a regression of the matrix data onto the 4D hyperplane so that the Frobenius norm of the error is minimized. This type of subspace constraint on image coordinates has been established explicitly for the paraperspective camera in [19] and the orthographic camera in [24] - however, this constraint holds for any affine camera. Note that not all matching candidates can satisfy this constraint - some do not have a valid match. Under this approach, these candidates will be named 'outliers' since no solution would place them in the proximity of the hyperplane.

It should be emphasized that the alignment of each frame is by itself a combinatorial problem. The multiple frame correspondence problem (i.e. the alignment of *all* images in W) is consequently an extremely complex task. Pairwise correspondence between feature points is modeled using a permutation matrix that determines, for each feature, the corresponding match in the other image while rejecting any extra points it may have (Fig. 1). Note that unicity, one of the most important assumptions of correspondence, is naturally imposed by this formulation.

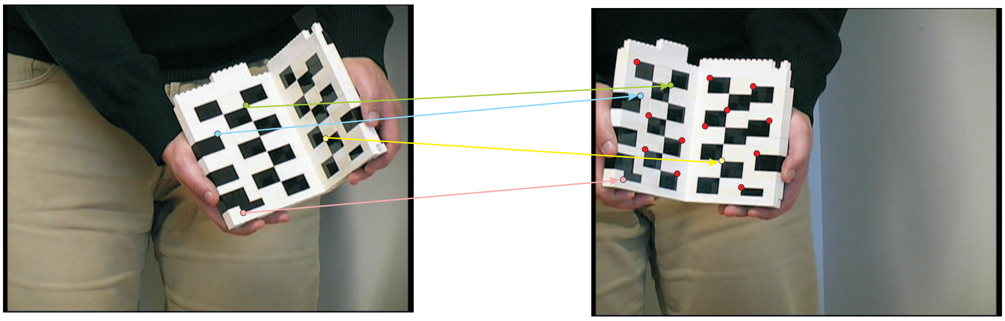


Fig. 1. The Matching Problem: rejected points are presented in red, feature points with valid matches respectively in green, yellow, blue and pink. For every feature, *all* points in the images on the right are potential matches. A correct correspondence has to be found while rejecting outliers.

As will be detailed in subsequent sections, the calculation of the correct match can be formulated as the search for the feature alignment that results in a rank-deficient measurement matrix. This is (indirectly) obtained through the minimization of the non-dominant singular values of the measurement matrix. In practice, this minimization is performed by using an iterative optimization procedure where each frame is matched individually, in a technique similar to *cyclic coordinate descent.* Each iteration results in a linear cost function, thus effectively solving a non-linear problem as a sequence of linear programs.

2.1 Feature representation

Observations on each frame are represented as a set of image coordinates containing the orthogonal projection of 3D feature points in the scene. It should be stressed that in this text the expression *feature point* is applicable to a generic image point and should thus not be limited to corner points. Assuming p_f visible feature points in frame f, we represent the u and v image coordinates in the u^f and v^f vectors. We assume that each set of p_f feature points is corrupted by a certain number of outliers, except for w_1 which contains only the points that are to be matched (i. e. inliers). The observation matrix corresponding to frame f is thus represented by

$$w_f = \begin{bmatrix} u_1^f & \cdots & u_{p_f}^f \\ v_1^f & \cdots & v_{p_f}^f \end{bmatrix} \tag{1}$$

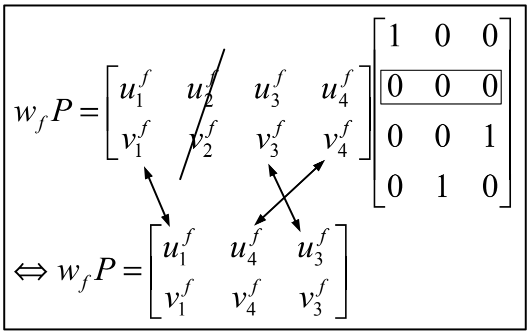
Observations corresponding to several frames can be vertically stacked in order to create a measurement matrix W_f that incorporates the projections of feature points up to scene f. However, the outliers in each frame have to be rejected beforehand; moreover, the remaining points have to be aligned so that corresponding features share the same column in W_f, as in Tomasi and Kanade [24]. Matrix P_k simultaneously aligns the feature points and rejects the outliers in the corresponding matrix w_k. W_f can consequently be written as

$$W_f = \begin{bmatrix} w_1 & \\ w_2 & P_2 \\ \vdots & \vdots \\ w_f & P_f \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} u_1^1 \cdots u_{p_0}^1 \\ v_1^1 \cdots v_{p_0}^1 \end{bmatrix} & I_{[p_0 \times p_0]} \\ \\ \begin{bmatrix} u_1^2 \cdots u_{p_2}^2 \\ v_1^2 \cdots v_{p_2}^2 \end{bmatrix} & P_{2[p_2 \times p_0]} \\ \\ \vdots & \vdots \\ \begin{bmatrix} u_1^f \cdots u_{p_f}^f \\ v_1^f \cdots v_{p_f}^f \end{bmatrix} & P_{f[p_f \times p_0]} \end{bmatrix} \tag{2}$$

In (2), each $P_k, k = 2, ..., f$, is a *rowwise partial permutation matrix*, that is defined by the conditions in (3). In the remainder of the text, for the sake of simplicity, these matrices will be referred to simply as *partial permutation matrices.*

$$\begin{aligned} & P_{k_{ij}} \in \{0,1\}, \forall i = 1...p_k, \forall j = 1...p_0 \\ & \sum_i P_{k_{ij}} = 1, \forall j = 1...p_0 \\ & \sum_j P_{k_{ij}} \in \{0,1\}, \forall i = 1...p_k \end{aligned} \tag{3}$$

In each of the frames, the optimal P_k allows a correct alignment of w_k to be obtained, as depicted in Figure 2.



$w_f P = \begin{bmatrix} u_1^f & u_2^f & u_3^f & u_4^f \\ v_1^f & v_2^f & v_3^f & v_4^f \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$
$\Leftrightarrow w_f P = \begin{bmatrix} u_1^f & u_4^f & u_3^f \\ v_1^f & v_4^f & v_3^f \end{bmatrix}$

Fig. 2. Rowwise Partial Permutation Matrix

In (2) p_0 identifies the number of features that will be matched, i. e. not discarded. p_0 is equivalent to the number of features in w_1, which contains no outliers. Note that it is assumed that $p_0 \leq p_k, \forall k \geq 2$ - for this reason,

P_k is usually a rectangular matrix, since a null row is added for each outlier. Furthermore, it is assumed that these p_0 features are visible in every frame.

2.2 Enforcing rank constraints

It has been shown in [24] that when considering rotation and translation the stacked measurement matrix W_f can be modeled as the product of two rank-4 matrices M and S_h:

$$W_f = MS_h = \left[R_{[2f\times 3]} | T_{[2f\times 1]} \right] \left[\frac{S_{[3\times p]}}{1_{[1\times p]}} \right] \tag{4}$$

In the previous expression, M contains the information necessary to reconstruct the camera position T and orientation R in each frame, while S_h represents the 3D coordinates S of the feature points in homogeneous coordinates. From (4) it can easily be verified that W_f is at most rank-4 (both the rows and columns of W_f lie on 4D linear subspaces), as it is the product of the rank-4 matrices M and S_h. This result is known as the rank theorem. Similar results can be derived for other affine camera models - see for example [19].

Our objective is thus to *find the set of permutation matrices $P_2, ..., P_f$ such that W_f is a rank-4 matrix.* Note that in the presence of bounded noise the rank-4 condition may be unattainable - in this case, we require that W_f be *approximately* rank-4, as shown in [13]. By searching for an *approximately rank-4* matrix we mean that the singular values associated to the null space of W_f should be as small as possible; however, for simplicity, in the remainder of this text we will simply refer to this constraint as the rank-4 or the rank constraint. In order to deal with situations associated to degenerate feature point distributions we assume that the total rank of W_f is known.

Work on factorization algorithms such as the ones referred previously is based on the assumption that a matching solution between the image points has already been found, so that image coordinates corresponding to the same feature point occupy the same column. In the presence of incorrect matches, the resulting W_f is (generally) of higher rank.

When considering f frames, the problem is thus to find the set of partial permutation matrices $P_2, ..., P_f$ that chooses and orders features in $w_2, ..., w_f$ so as to generate a rank-4 W_f. Note that the afore-mentioned rank constraint, as it acts on W_f as a whole, requires that all previously matched permutation matrices be recalculated each time a new frame is processed. To avoid the complexity of solving simultaneously for all P_k, we choose to solve the problem for each partial permutation matrix in sequence, while keeping the remaining matrices constant. In practice, we use a cyclic coordinate descent (CCD) algorithm to solve an optimization problem in $\mathcal{P}^2 \times \mathcal{P}^3 \times ... \times \mathcal{P}^f$, where $\mathcal{P}^k$ represents the space of all partial permutation matrices of dimension $p_k \times p_0$. This process is depicted in Figure 3.

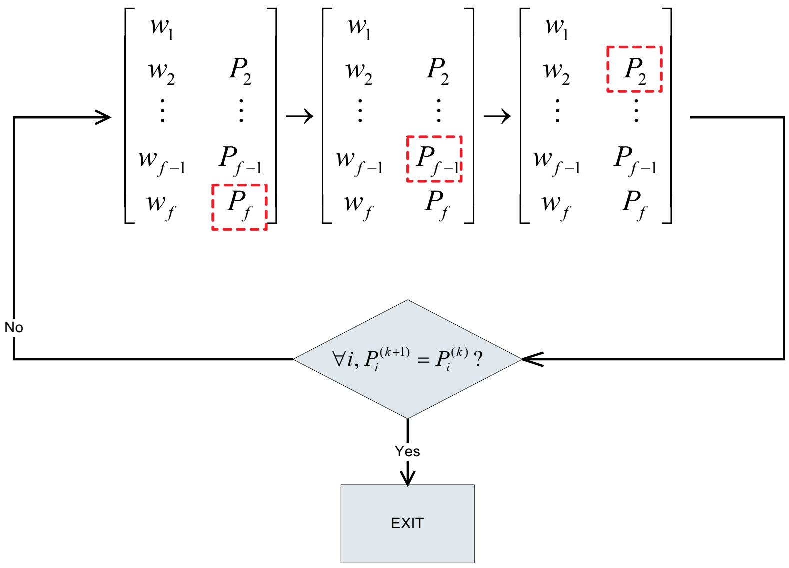


Fig. 3. Iterative enforcement of rank. Red squares identify the permutation matrix which is being aligned at each step.

The algorithm will continue to cycle over all partial permutation matrices until convergence is achieved. Each step of the CCD algorithm corresponds thus to the solution of a matching problem where the features of a certain image are matched against the current correspondence estimates of the remaining images. In other words, the two rows containing the u and v coordinates of each image are aligned so as to be contained in the (approximate) 4D subspace defined by the rows of the remaining images. Note that although each frame is matched individually, the outcome of this single matching procedure will influence *all* remaining frames whose correspondence will be recalculated afterward, already taking into account the new result.

Since the original (non iterative) cost function is, in fact, not convex, it is theoretically possible that the CCD algorithm can get stuck in local minima, thereby returning a suboptimal solution. In practice, however, it was observed that in most situations the algorithm will deliver a very close approximation to the correct result.

2.3 Derivation of the cost function

At the k^{th} step of the cyclic coordinate descent algorithm, we solve for P_{f-k+1} individually. In this section we develop a cost function that solves the correspondence problem for the $f-k+1^{th}$ frame while assuming all other frames are correctly matched.

We consider the SVD decomposition of W_f

$$W_f = \begin{bmatrix} w_1 & \\ w_2 & P_2 \\ \vdots & \vdots \\ w_f & P_f \end{bmatrix} = Q\Sigma V^T \tag{5}$$

and define Z as

$$Z = W_f W_f^T = \begin{bmatrix} w_1 w_1^T & w_1 P_2^T w_2^T & \cdots & w_1 P_f^T w_f^T \\ w_2 P_2 w_1^T & w_2 P_2 P_2^T w_2^T & \cdots & w_2 P_2 P_f^T w_f^T \\ \vdots & \vdots & & \vdots \\ w_f P_f w_1^T & w_f P_f P_2^T w_2^T & \cdots & w_f P_f P_f^T w_f^T \end{bmatrix} \tag{6}$$

As can be seen in (6), Z is a function of all permutation matrices. However, all of them will be considered as constant except for matrix P_{f-k+1} when solving the problem associated to this iteration. The aim of our cost function is to find the matching solution for frame $f-k+1$ that approximates a rank-4 W_f at the k^{th} step of the cyclic coordinate descent algorithm. The approach to finding the matching solution that turns W_f into a rank-4 matrix is to note that when a matrix is rank-deficient, a precise number of its singular values (and eigenvalues!) are zero - corresponding to the difference between the number of rows/columns (whichever is smaller) and the actual rank of the matrix. It can be shown that Z is also rank-deficient and as such can be equivalently used to solve for the desired permutation matrix instead of W_f.

The minimization of the sum of eigenvalues (or, equivalently, the trace), as defined in (7), of a positive semidefinite matrix is a well known method to minimize rank (see, for example [5] for a brief description). For a positive semidefinite matrix such as Z, which possesses nonnegative eigenvalues, this is equivalent to the minimization of the l_1-norm of the vector of eigenvalues $\Lambda(Z)$.

$$\begin{aligned} trace(Z) &= \sum_{i=1}^{2f+2} \lambda_i(Z) \\ \|\Lambda(Z)\|_1 &= \sum_{i=1}^{2f+2} |\lambda_i(Z)| \end{aligned} \tag{7}$$

This minimization tends to create a sparse eigenvalue vector (i.e. with many zero entries), thus effectively minimizing rank. This method is advantageous since it results in a convex optimization problem, as opposed to the original (non-convex) rank minimization problem. In this specific case a rank-4 matrix is desired. Consequently, a slightly different approach will be used: the four largest eigenvalues of Z will not be considered, i.e. only the eigenvalues $\lambda_i, i > 4$ of Z will be minimized. This is effectively equivalent to minimizing the residue in terms of the rank-4 constraint.

The eigenvalues of Z can be obtained, by definition, from the following expression, where q_i represents the i^th column of Q, i. e. the i^{th} eigenvector of $W_f W_f^T$:

$$\lambda_i = q_i^T Z(P_2, ..., P_{f-k+1}, ..., P_f) q_i \tag{8}$$

In (8), each $P_n \in \mathcal{P}^n$. As before, $\mathcal{P}^n$ represents the space of all partial permutation matrices of dimension $p_n \times p_0$.

When processing frame f, this implies the minimization of the sum of the $2f-2$ smallest eigenvalues, since W_f has $2f+2$ rows when the f^{th} frame is processed (the practical need for two additional rows in W_f will be made clear in the section dedicated to initialization).

It should be noted when minimizing the eigenvalues that the value of each λ_i depends itself on the structure of P_{f-k+1}. Our matching problem must thus be formalized as the search for the optimal partial permutation matrix P^*_{f-k+1} such that:

$$\begin{aligned} P^*_{f-k+1} = \underset{P_{f-k+1}}{\arg\min} \left(\sum_{i>4} \lambda_i(\hat{P}_2, ..., P_{f-k+1}, ..., \hat{P}_f) \right) = \\ \underset{P_{f-k+1}}{\arg\min} \left(\sum_{i>4} q_i^T Z(\hat{P}_2, ..., P_{f-k+1}, ..., \hat{P}_f) q_i \right) \end{aligned} \tag{9}$$

The partial permutation matrices that are not being optimized are represented with a *'hat'*, to emphasize that their values correspond to estimates that are being held constant at the current iteration. Note that there is an interdependency between the values of the permutation matrices and the eigenvectors of Z, $q_i, i = 5, ..., 2f+2$. When working with a calibrated system, these eigenvectors are known, since they are strictly dependent on camera position and not on the structure of the scene. In fact, in can be shown that each of the non-dominant eigenvectors is a base vector for the null space of the column space of W_f. It is known from factorization literature that the column space of an observation matrix defines camera movement (see (4) - consequently, if the column space of W_f is known, so are the non-dominant eigenvectors of Z.

In the case of uncalibrated systems, the eigenvectors are unknown unless the permutation matrices are already available, thus effectively adding a new set of unknowns. This case consequently requires a different approach in order to allow the solution of the matching problem. The discussion of this problem will be deferred to later sections of this text; for now, these eigenvectors will be assumed as known, so that attention can be concentrated on obtaining the optimal permutation matrix for step k of the CCD algorithm.

3 Methodology

3.1 Constructing a linear cost function

In the following paragraphs an optimization procedure for the cost function derived in the previous section is presented; in particular, it is shown that the cost function can be written as a linear problem and hence efficiently minimized.

When optimizing in P_n only, each term (corresponding to a single eigenvalue) in the sum of the cost function (9) is represented as a set of terms dependent of P_n. By revisiting the structure of Z in (6), terms involving P_n are either of the form $w_i P_i P_n^T w_n^T$ (and thus linear when considering P_i constant) or of the form $w_n P_n P_n^T w_n^T$, which is apparently non-linear in the elements of P_n, the optimization variables for the present iteration. However, since P_n is a partial permutation matrix, this last term can be written as:

$$P_n P_n^T = \begin{bmatrix} \sum_{i=1}^{p_0} P_{n1i}^2 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \sum_{i=1}^{p_0} P_{np_ni}^2 \end{bmatrix} \tag{10}$$

Note that the diagonal of the matrix in (10) contains exactly p_0 unit entries, which are also the only nonzero entries in the diagonal. Taking into account that the elements of P_n are either 0 or 1, the j^{th} term of the diagonal of the matrix in (10) can be simplified to $\sum_{i=1}^{p_0} P_{nji}$. In practice, this leads to a minimization procedure which is linear in P_n. By arranging the elements of P_n conveniently, a linear formulation for the cost function can thus be found.

Given a matrix M, the *vec* operator stacks its columns in order to form a vector. Rearranging P_{f-k+1} as $x = vec(P_{f-k+1})$, a modified cost function can be written, whose optimization yields the solution to the linear problem in (11). Note that the optimization is constrained in order to force x to be the result of the vectorization of a valid permutation matrix - the relevant constraints are detailed in the Appendix.

$$\begin{aligned}
&x* = \arg\min_{x} c.x \\
&s.t. \\
&x = vec\left(P_{f-k+1}\right), P_{f-k+1} \in \mathcal{P}^{f-k+1}
\end{aligned} \tag{11}$$

The coefficient vector c of the linear program can be determined by developing

$$\sum_{i>4} q_i^T Z(\hat{P}_2, ..., P_{f-k+1}, ..., \hat{P}_f) q_i \tag{12}$$

in order to the elements of P_{f-k+1}. Under these conditions, c is given by (13), assuming W_c^{f-k+1} as the submatrix of W_f not containing w_{f-k+1} and q_i^a, q_i^b as the parts of q_i associated, respectively, to W_c^{f-k+1} and w_{f-k+1}:

$$\begin{aligned}
&c = \sum_{i=5}^{2f-2} c_i, \\
&c_i = 2\left[\left({q_i^a}^T W_c^{f-k+1}\right) \otimes \left({q_i^b}^T w_{f-k+1}\right)\right] + \\
&+1_{[1\times p_0]} \otimes \left[\left({w_{f-k+1}}^T q_i^b\right)^T \bullet \left({w_{f-k+1}}^T q_i^b\right)^T\right]
\end{aligned} \tag{13}$$

In (13), $\otimes$ and $\bullet$ represent the Kronecker and Schur products, respectively. The details of the calculation of c_i are given in the Appendix.

The formulation presented in (11) still remains an integer minimization problem and as such has no efficient solution. However, it is known that the minimum of a linear function over a compact convex set $\mathcal{C}$ is located at an extreme point of $\mathcal{C}$. Consequently, the constraint set of a minimization problem with linear objective function can be relaxed into its convex-hull, provided that all the points in the original set are extreme points of the new set. In the present case, it can be shown that the convex-hull of the set of partial permutation matrices $\mathcal{P}^k$ is the set of rowwise substochastic matrices $\mathcal{S}^k$. This set can be defined by the second and third equations in (3) and by the condition

$$P_{kij} \geq 0, \forall i = 1...p_k, \forall j = 1...p_0 \tag{14}$$

The resulting (continuous) problem is thus equivalent to the original, but for this class of problems there exist several efficient algorithms that can provide

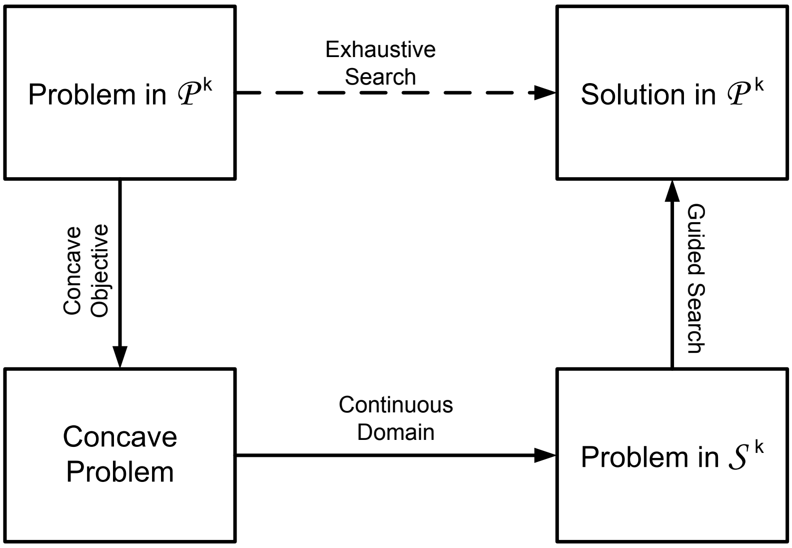

Problem in $\mathcal{P}^k$
Exhaustive Search
Solution in $\mathcal{P}^k$
Concave Objective
Guided Search
Concave Problem
Continuous Domain
Problem in $\mathcal{S}^k$

Fig. 4. Relaxation Process

an adequate solution, such as the well known simplex algorithm. The process is depicted in Figure 4.

This method of solving the integer optimization problem has originally been proposed in [13].

It should be noted that in terms of computational complexity the use of the simplex algorithm allows each iteration to be solved in polynomial time (for practical purposes it can be considered that the simplex algorithm terminates in polynomial time - worst case exponential time will not be considered here as its occurrence is extremely uncommon). The obvious consequence of this is that the iterative process has polynomial complexity.

3.2 Dealing with unknown eigenvectors

As seen in an earlier section, the use of this algorithm with an uncalibrated system leads to a situation where two sets of unknowns (the permutation matrices and the eigenvectors) have to be simultaneously determined to solve the problem. Note that this is a very specific situation, since usually even for systems with unknown camera positions a few correct matches from very prominent features can be extracted from the images and used to (approximately) calculate the column space of W_f, in what is usually called *poor calibration.*

The uncalibrated problem can be tackled only under a feature tracking framework. This assumes a short baseline and evenly spaced images, both in the temporal and spatial sense. Under this framework, an iterative method is proposed, which at step k of the cyclic coordinate descent alternates the optimization on P_{f-k+1} and the calculation of the set $\{q_5, ..., q_{2f+2}\}$ of non-dominant eigenvectors of Z. At each iteration of the algorithm a new P_{f-k+1} is calculated by minimizing the cost function using the current estimate of the eigenvectors. New eigenvectors are subsequently extracted from Z corrected according to the newly found P_{f-k+1}. The algorithm will continue to iterate alternatively in P_{f-k+1} and the eigenvectors until convergence is achieved.

A sufficiently accurate initial estimate is required for $\{q_5, ..., q_{2f+2}\}$ in order to start the iterative process. This estimate will usually be available directly from the previous step of the CCD algorithm, except when a new frame is first matched (i.e. when the frame has just been inserted in the algorithm). In this case, benefiting from the small baseline, an estimate for the correct alignment of the new features is obtained, by assuming a smooth movement in the scene. From this estimate initialization values for the eigenvectors are calculated, so that the CCD algorithm for the new frame can be bootstrapped.

An alternative approach to the technique described in the previous paragraphs would be to use a trial-and-error method, initializing each frame multiple times until a reasonable estimate is obtained. Note that even using a diminute number of features this can be a computationally intensive technique.

It should be pointed out that using either method an inaccurate estimate of the eigenvectors will lead to a situation where the algorithm might not converge to the correct solution. It is thus highly desirable that a reasonable estimate is obtained - this is a limitation of the present implementation of the method under an uncalibrated setup.

3.3 Initialization of the Algorithm

When processing the first frame of an image set, W_f contains only w_1 and w_2, having thus only 4 rows. Under these circumstances, W_f would be rank-4 whatever the value of P_2. To be able to identify rank-deficiency, at least one further row should be present. This problem can be tackled using two different approaches: A correspondence algorithm such as [13] can be used to provide an alignment between two frames - in fact, this would correspond to having a w_1 with four instead of two rows (hence the dimension of W_f being $2f + 2$ instead of $2f$). W_f would then have six rows (four from w_1 and two from w_2), thus being rank-4 only for the correct P_2. An alternative approach would be to add a row of ones to w_1, as shown in (15).

$$w_1 = \begin{bmatrix} 1 & \cdots & 1 \\ u_1^1 & \cdots & u_{p_0}^1 \\ v_1^1 & \cdots & v_{p_0}^1 \end{bmatrix} \tag{15}$$

Note that the addition of a row of ones to w_1 does not alter the row space of W_f. To this end, observe that S_h in (4) is in homogeneous coordinates thus including a row of ones. Since the rows of S_h define the row space of W_f a row of ones is, by definition, part of the row space of W_f. This approach, unlike the former, does not require the previous match of two of the frames in the sequence; however, since only two frames are in fact present when w_2 is matched, points on the epipolar line have the same matching cost, thus becoming indistinguishable to the algorithm. As a consequence, severe mismatches may happen; to avoid this, a simple photometric term may be added that can disambiguate most situations.

4 Summary of the algorithm

Based on the previous chapters, we present in this section an outline of the steps necessary to obtain feature correspondences in a set of images. The second part of the algorithm is only applicable if the eigenvectors are unknown.

4.1 Iterative rank enforcement

(1) Extract the set of observations w_f corresponding to a new frame. This is the current frame ($k = 1$).
(2) Given w_1, w_2, ..., w_f, run 4.2 (for uncalibrated systems) or 4.3 (for calibrated systems) in order to obtain the partial permutation matrix P_{f-k+1} corresponding to the current frame.
(3) Correct the eigenvectors and repeat (2) with $k = k + 1$, until $k = f - 1$.
(4) If any change in the set of permutation matrices is recorded go to (2), and repeat the algorithm with k reset to 1.

4.2 Uncalibrated - simultaneous determination of match and eigenvectors

(1) If w_{f-k+1} has been previously matched in a previous CCD step, go to (3). If not, given the provisional match between w_{f-k-1} and w_{f-k} extrapolate,

assuming a smooth movement, an estimate for the position of the features in w_{f-k+1}.

(2) Using the estimate in (1), determine an initial value for the non-dominant eigenvectors of Z.
(3) With the initial value of the eigenvectors, solve the integer optimization problem for P_{f-k+1} by running 4.3.
(4) If P_{f-k+1} has converged, stop.
(5) Given P_{f-k+1}, update the eigenvectors of $Z(P_{f-k+1})$.
(6) Return to 1.

4.3 Determination of match

(1) Given $w_1, ..., w_{f-k+1}, ..., w_f$ and the eigenvectors, build a linear cost function for P_{f-k+1} as detailed in 2.3 and 3.1.
(2) Solve the integer optimization problem for P_{f-k+1} by using relaxation as referred in 3.1.

5 Experiments

We describe in this section a set of experiments in order to validate the algorithm that has been presented. Experiments with synthetic data provide a proof-of-concept solution and an analysis of the degradation of matching precision with increasing noise. Further experiments with datasets consisting of real images demonstrate the algorithms' ability to function under less than optimal conditions (i.e., with noise and deviations to the theoretical model). Aspects such as robustness to outliers and applications to feature tracking and reconstruction are also discussed in the following sections.

5.1 Sphere Dataset: proof of concept

The synthetic 'Sphere' dataset consists of 100 orthographic images of a spherical structure represented by eight meridians. The meridians are defined by a total of 1216 equally distributed points, which will be used as matching candidates. Of these, 16 points (two on each meridian) have been singled out as the features to match. Unlike real images, coordinates are in this case real numbers and thus unrelated to pixels; in fact, matching candidates are very densely packed and several of them are present within one unit interval (note that matching candidates are at most 0.3 units apart and most of them are significantly closer). Due to its synthetic nature, a precise ground-truth is available.

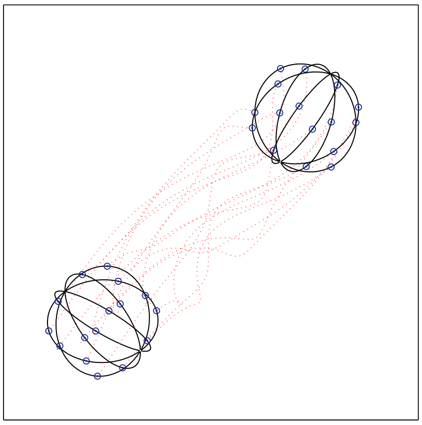


Fig. 5. Superimposed first and last images of the 'Sphere' dataset, with feature trajectories

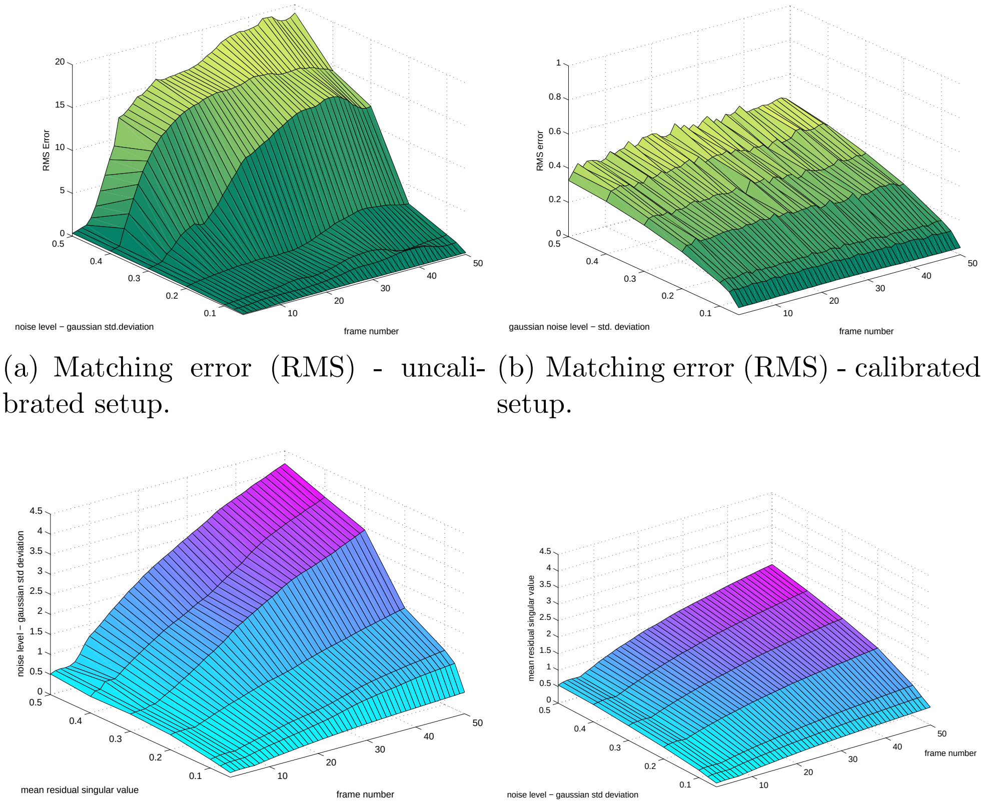
In the first experiment, features in each of the 100 frames of the dataset are matched in an uncalibrated system. Correspondences for the first two images have in this case been fixed beforehand (see the section related to initialization). No noise has been introduced in this experiment, since its sole objective is to demonstrate the ability of the algorithm to tackle an uncalibrated set of images.

Features have been marked in blue and their corresponding trajectories in red. Results can be seen in Fig. 5.1. As could be expected under theoretically ideal conditions, zero error was achieved for all correspondences in every frame. Furthermore, the algorithm also rejects outliers that are densely packed beside each correct feature.

Although the method uses a standard non-commercial algorithm to solve the linear program, the solution for each iteration is obtained in 0.75 seconds for an LP with 19456 variables. Construction of the linear program takes only a few hundreds of seconds. Logically the use of commercial algorithms could further speed up the process.

5.2 Sphere dataset: performance degradation with noise

In this section the previous experiment is repeated with increasing levels of noise corrupting the observations (i.e. each of the w_f). Recall that the sphere data set is *synthetic*, that is, the image coordinates in the dataset are uncorrelated to a physical measure as pixels. Results must thus be interpreted in view of object size, a sphere of diameter 80, and candidate spacing (at most 0.3). The noise added to each of the observations is gaussian with standard deviation between 0.05 and 0.1. Note that an error level of this magnitude, for candidates that are this close apart is significant. Also, noise effects are



(a) Matching error (RMS) - uncalibrated setup.

(b) Matching error (RMS) - calibrated setup.

(c) mean value of residual singular values - uncalibrated setup.

(d) mean value of residual singular values - calibrated setup.

Fig. 6. Performance degradation with increasing noise - ’Sphere’ dataset.

cumulative as the algorithm advances through the sequence, because noisy measurements are integrated into W_f. Also note the in the uncalibrated case wrong matches will have a repercussion on subsequent initialization vectors. To isolate this factor, the experiment has additionally been run under calibrated conditions, i.e. with known initialization vectors.

In the uncalibrated case, it can be seen that for a variance smaller than 0.1 the sequence is processed with virtually null error and no error integration is noticeable. For larger noise levels error integration is significant in the second half of the sequence and a degradation of performance occurs rather quickly. This has to do with features ’jumping’ between intersecting meridians. If this experiment were scaled to a 512x512 image, this would amount to a 2 pixel error standard deviation at the breaking point (i.e. where the algorithm no longer provides a satisfactory match - for a gaussian noise level of 0.2 standard deviation).

The calibrated experiment, on the other hand, shows no error integration, as well as significantly higher overall performance - note that the scale in the corresponding RMS error graphic is $1/20^th$ of the one in the uncalibrated case. It can thus be concluded that the lower results of the uncalibrated experiment

'Hotel'	
min	**max**
0,10 pxl	1,51 pxl

Table 1
Disparities between corresponding pixels for the 'Hotel' sequence.

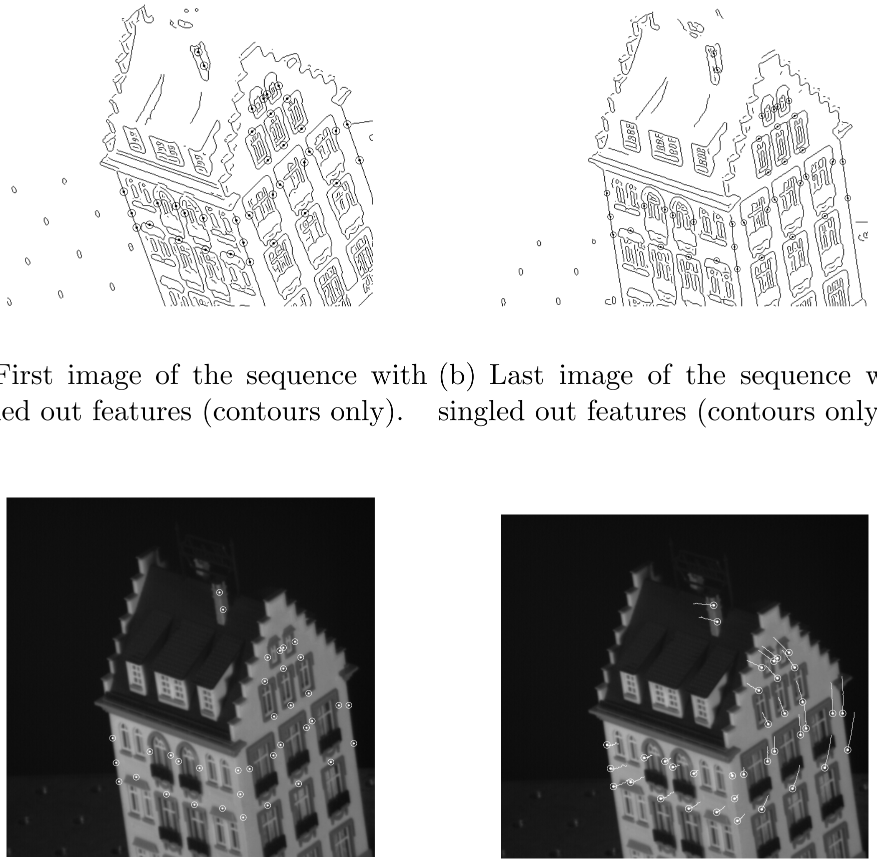
are related to a sensitivity of the initialization vectors to higher levels of noise. Apart from this aspect, the algorithm has demonstrated to be reasonably robust to noise.

5.3 Hotel dataset: feature tracking in an uncalibrated image sequence

In this experiment the algorithm's capacity under a feature matching context is tested. An image sequence originally obtained from CMU's Image database (*http://vasc.ri.cmu.edu/idb/*) is used. Its 30 images of a toy house can be considered orthographic for all practical purposes. Matching candidates are extracted as the points composing the contours of each image(typically 11000 points per image). The 37 features to be tracked have all been placed on straight lines and would thus suffer from the aperture problem if matched with a local method. Initialization is provided by the user: two initial matches are selected manually for bootstrap. Results are presented in Fig. 7. Additionally, a reconstruction has been made using Tomasi and Kanade's algorithm that shows the 3D shape of the house when viewed from above (Fig. 5.3). Although ground-truth is not available for this sequence it can be verified by visual inspection that the returned correspondences have a high degree of precision.

The matching solution to a frame will typically result in a linear program with hundreds of thousands of variables (in this case more than 400000). The number of variables is the result of the product of the number of features in the matched frames (37) with the number of candidates in the frame that is to be matched (around 11000). A linear program of this type takes less than a second to build and about 4 min. to solve (i. e., 4 min. per iteration). In order to speed up the process, *a priori* knowledge is used, by assuming that there is a limited disparity between consecutive images. Under this assumption, matching solutions that would result in a very large movement of the features can be ruled out - in practice, this is equivalent to forcing some of the entries of the partial permutation matrix to 0. In this way, an important reduction of dimensionality is achieved. It has been verified that this reduction does not affect the final result, but only the time required to obtain it.

In this experiment, such a procedure was applied by assuming that consecutive frames have a limited disparity and thus limiting the number of candidates for a given feature. Using this method each linear problem will take no more



(a) First image of the sequence with singled out features (contours only).

(b) Last image of the sequence with singled out features (contours only).

(c) Last image of the sequence (grayscale).

(d) Feature trajectories.

Fig. 7. Results for the 'Hotel' dataset.

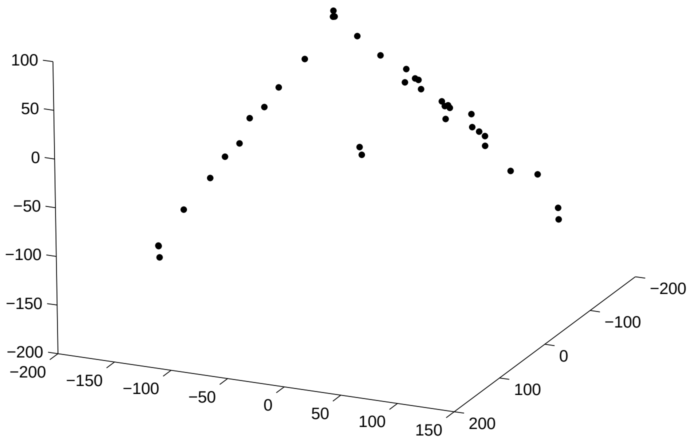


Fig. 8. 3D reconstruction of the 'Hotel' dataset, viewed from above.

than a few seconds to solve, depending on the number of candidates allowed.

It can easily be seen that the number of rows of W_f grows linearly with the number of frames. Although the number of variables in *each* linear program does not suffer an increase the minimum number of cyclic coordinate descent iterations will increase significantly, thereby significantly slowing down the algorithm. In practice, this problem can be avoided by assuming that after a certain number of frames have been processed the first n matching solutions can be considered correct and will thus not be iterated upon. This approximation has led to a significant decrease in processing time, while not affecting the final result's precision.

5.4 Hotel dataset: resistance to outliers

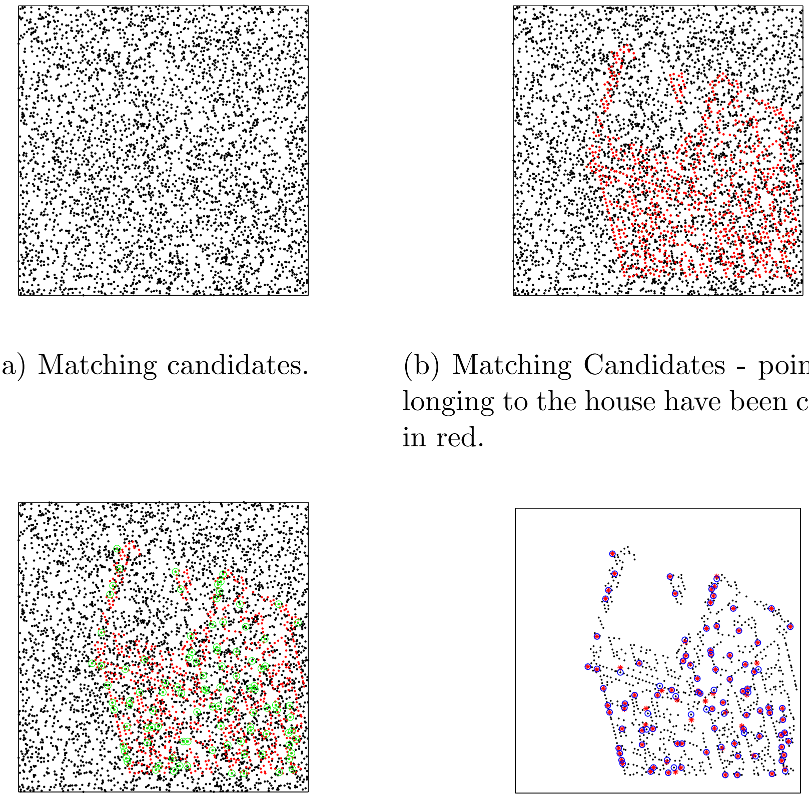
In this experiment a single frame of the hotel dataset is matched in a highly cluttered environment. To the correct matches a massive number of uniformly distributed outliers is added to the image. Nevertheless, the method proposed in this paper is able to single out the features with a very low degree of error. In fact, when comparing with the matching solution provided by [13], almost 90% of the features present less than five pixels of error, as can be seen in the histogram of Fig. 5.4.

Solution to this problem (with 410000 variables) was obtained in a matter of 4min. Note that unlike in the previous experiment, no *a priori* knowledge was used here.

5.5 Grid dataset: calibrated sequence

The 'Grid' dataset comprises a set of images of a LEGO grid with two perpendicular walls. As in the hotel sequence, the matching candidates are the points defining the contour of the image. Among these, 99 features are selected, all of them on straight edges. Note that this image is not perfectly orthographic and as such presents a further challenge to the algorithm. Unlike previous experiences, this dataset has been calibrated by solving correspondence for a few corners with an implementation of the Lucas-Kanade algorithm (note that this algorithm has to rely on *corners* and as such would not be able to match features such as the ones that were chosen). Images of the dataset with tracked features superimposed can be seen in Fig. 11.

The solution of each linear problem takes 9min. and 50 sec. Using *a priori* knowledge about the maximum disparity, as in the first 'Hotel' experiment, processing time for each LP was reduced to a few seconds.



(a) Matching candidates.

(b) Matching Candidates - points belonging to the house have been colored in red.

(c) Matching candidates with correct matches (in green) - house (in red) is superimposed for reference only.

(d) Matching solution (in red) and correct matches (blue circles).

Fig. 9. Results for the outlier robustness test.

6 Future work

In this version of the algorithm, adequate initialization for the present frame is achieved using the result of the previous matching process and extrapolating the new initialization vector by assuming a smooth movement. Better results can be achieved by using more sophisticated ways to extrapolate future movement and thus calculate initialization vectors that are valid over a larger set of disparities.

The fact that the initial set of features has to be visible over the whole sequence is an issue that imposes restrictions on the choice of image points to match. Handling occlusions is at the moment still an open issue.

The algorithm can be made to run radically faster by subdividing the simplex problems so that only a subset of the image points are matched at a time. As the simplex algorithm is typically solved in polynomial time, processing n

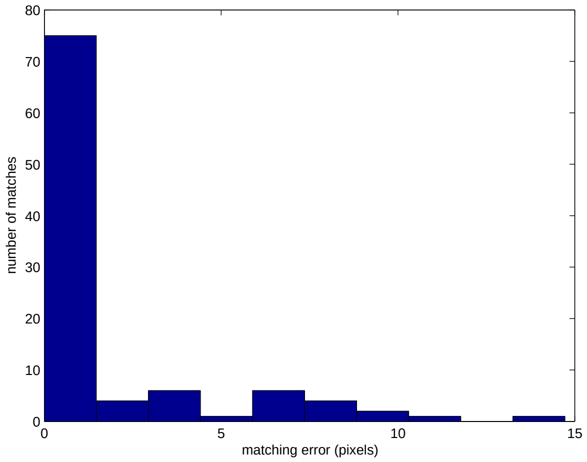
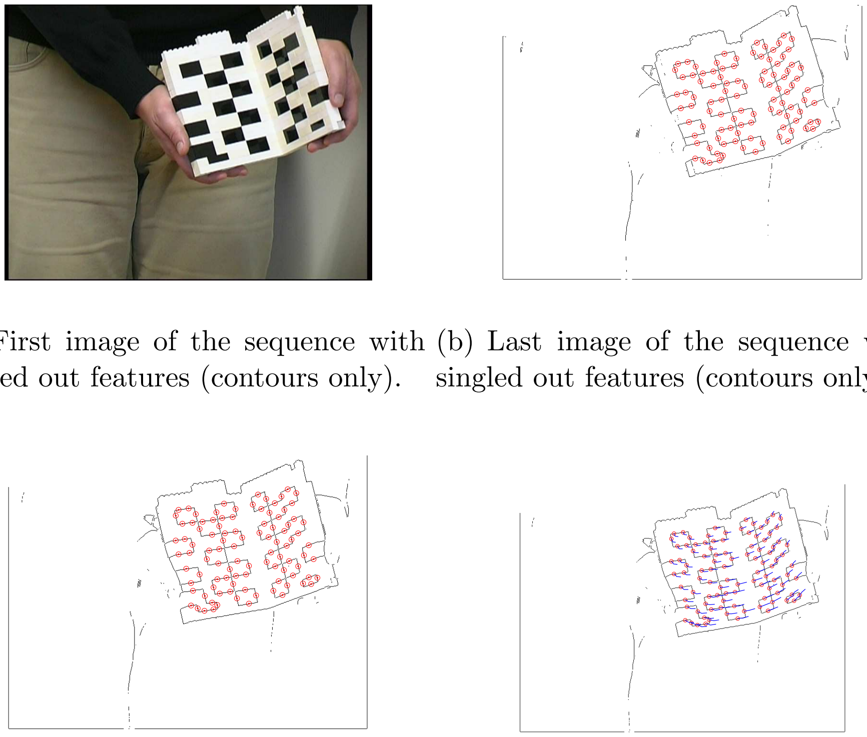


Fig. 10. Histogram of the number of matching errors.



(a) First image of the sequence with singled out features (contours only).

(b) Last image of the sequence with singled out features (contours only).

(c) Last image of the sequence (grayscale).

(d) Feature trajectories.

Fig. 11. Results for the 'Grid' dataset.

d-dimensional problems is usually much faster than processing a single $(n \times d)$-dimensional problem.

Our algorithm is presently constrained to use images obtained from affine cameras. The extension to projective cameras might be obtained using Heydens work in [7]. However, an analysis on the convergence properties of the altered method will have to be performed in order to ensure that performance is not affected.

7 Discussion and conclusions

We have presented a method that is capable of matching image points extracted from feature points presenting inadequate texture to be matched by photometric methods. This algorithm is able to calculate the match of several images, thus optimizing the result in the whole image sequence. Iterative enforcement of rank was shown to be an effective way to enforce a geometric constraint on the scene while maintaining a low computational cost.

Our algorithm is able to cope with a high percentage of outliers without any significant decrease in performance. We have run experiments with a high number of points, demonstrating that our method is computationally feasible. Reconstruction performance visually demonstrates the capabilities of this method.

A Calculation of c

This section of the appendix details the calculation of the coefficient vector c of the linear program, resulting from the matching problem at step k of the CCD algorithm.

Calculations can simplified, without loss of generality, by reordering W_f so that the frame that is to be aligned - w_{f-k+1} - occupies the last pair of rows. The submatrix of the remaining $2f$ rows W_c^{f-k+1} will be considered constant - its elements can thus be disregarded, since they do not depend on P_{f-k+1}. This allows a considerable simplification of the calculations and, since only constant terms are eliminated, does not affect the value of P^*_{f-k+1}. In the following expressions, the $f-k+1$ index will for simplicity be dropped, since it is implicit at all moments that calculations refer to the k^{th} step of the CCD algorithm.

$$Z_0 = \begin{bmatrix} 0_{[2f\times 2f]} & 1_{[2f\times 2]} \\ 1_{[2\times 2f]} & 1_{[2\times 2]} \end{bmatrix} \bullet \begin{bmatrix} W_c W_c^T & W_c P^T w^T \\ wPW_c^T & wPP^T w^T \end{bmatrix} \tag{A.1}$$

The i^{th} term of the cost function in (9) (which corresponds to the minimization of the i^{th} eigenvalue of Z) can then be written as:

$$q_i^T Z_0 q_i = \begin{bmatrix} q_i^1 \\ \vdots \\ q_i^{2f+2} \end{bmatrix}^T \begin{bmatrix} 0_{[2f \times 2f]} & W_c P^T w^T \\ wPW_c^T & wPP^T w^T \end{bmatrix} \begin{bmatrix} q_i^1 \\ \vdots \\ q_i^{2f+2} \end{bmatrix} \tag{A.2}$$

In order to present this problem as a linear program, q_i is divided in the following manner :

$$q_i = \left[{q^a_{i_{[2f\times 1]}}}^T \; {q^b_{i_{[2\times 1]}}}^T \right]^T \tag{A.3}$$

Using (A.3) we can develop (A.2) as follows:

$$\begin{aligned} q_i^T Z_0 q_i = {} & 2 \sum_{m=1}^{p} \sum_{n=1}^{p_0} \left(\sum_{l=1}^{2} (q_i^b)_l^T w_{lm} \right) \left(\sum_{j=1}^{2f} (W_c)^T_{jn} (q_i^a)_j \right) P_{mn} \\ & + \sum_{m=1}^{p} \sum_{n=1}^{p_0} \left(\sum_{l=1}^{2} (w_{lm})^T (q_i^b)_l \right)^2 P_{mn} \end{aligned} \tag{A.4}$$

Note that in the second term we take advantage of the fact that PP^T is a diagonal matrix. We can express (A.4) as a function of $x = vec(P)$. Note that this calculation need be performed $2f-2$ times, corresponding to the number of non-dominant eigenvalues that have to be minimized in order to obtain a rank-4 Z. The complete c is given by the sum of all c_i as in (A.4).

$$c = \sum_{i>4}^{2f-2} 2 \left[\left((q_i^a)^T W_c \right) \otimes \left((q_i^b)^T w \right) \right] + 1_{[1 \times p_0]} \otimes \left[\left(w^T q_i^b \right)^T \bullet \left(w^T q_i^b \right)^T \right] \tag{A.5}$$

B The constraints of the linear program

The linear program presented in (11) is a *constrained* optimization problem - in fact, these constraints exist to guarantee that x actually is the result of the vectorization of a valid partial permutation matrix. Consequently, the constraints of the linear problem can be derived directly from the expressions that define this kind of matrix, as presented in last three equations of (3). By reformulating these constraints according to the vectorized x, the following equations are obtained:

$$\begin{aligned}
A_1 &= \begin{bmatrix} 1_{[1\times p_0]} \otimes I_{[p\times p]} \\ -1_{[1\times p_0]} \otimes I_{[p\times p]} \end{bmatrix} & b_1 &= \begin{bmatrix} 1_{[p\times 1]} \\ -1_{[p\times 1]} \end{bmatrix} \\
A_2 &= I_{[p_0\times p_0]} \otimes 1_{[1\times p]} & b_2 &= 1_{[p_0\times 1]} \\
A_3 &= \begin{bmatrix} 1_{[1\times p_0 p]} \\ 1_{[1\times p_0 p]} \end{bmatrix} & b_3 &= \begin{bmatrix} p_0 \\ -p_0 \end{bmatrix}
\end{aligned} \tag{B.1}$$

Each of the submatrices presented above enforces one of the constraints in (3), except for the first one, since the problem is actually solved in the continuous domain. The complete constraint can then be formulated as in (B.2).

$$\begin{aligned}
& Ax \leq b, \\
& x \geq 0 \\
& A = \begin{bmatrix} A_1 \\ A_2 \\ A_3 \end{bmatrix} \; b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}
\end{aligned} \tag{B.2}$$

References

[1] G.Golub and C. van Loan. *Matrix Computations.* John Hopkins University Press, 1996.

[2] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press 2000.

[3] H. Lütkepohl. *Handbook of Matrices*, John Wiley & Sons 1996.

[4] G. Nemhauser and L. Wolsey. *Integer and Combinatorial Optimization*, John Wiley & Sons 1999.

[5] M. Fazel, H. Hindi and S. Boyd. A Rank Minimization Heuristic with Application to Minimum Order System Approximation. In *Proc. ACC*, June 2001.

[6] V. Ferrari, T. Tuytelaars and L. van Gool. Wide-Baseline Multiple-View Correspondences. In *Proc. ICCV*, October 2003.

[7] A. Heyden, R. Berthilsson and G. Sparr. An iterative factorization method for projective structure and motion from image sequences. *Image and Vision Computing*(17), 13(1), pp. 981-991, November 1999.

[8] M. Irani. Multi-Frame Optical Flow Estimation Using Subspace Constraints. In *Proc. ICCV*, September 1999.

[9] M. Irani and P. Anandan. Factorization with Uncertainty. In *Proc. ECCV*, June 2000.

[10] M. Irani and P. Anandan. A Unified Approach to Moving Object Detection in 2D and 3D Scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 20(6), 1998.

[11] V. Kolmogorov and R. Zabih. Multi-camera Scene Reconstruction via Graph Cuts. In *Proc. ECCV*, May 2002.

[12] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of the* 7^{th} *International Joint Conference on AI*, 1981.

[13] J. Maciel and J. Costeira. A Global Solution to Sparse Correspondence Problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 25(2), February 2003.

[14] Y.Ma, Kun Huang et al. Rank-Conditions on the Multiple View Matrix. *International Journal of Computer Vision* 59(2), 115137, 2004.

[15] D.McReynolds and D. Lowe. Rigidity Cheking of 3D Point Correspondences under Perspective Projection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 18(12), December 1996.

[16] D. Martinec and T. Pajdla. 3D Reconstruction by Fitting Low-Rank Matrices with Data. In *Proc. CVPR*, June 2005.

[17] R. Oliveira, J. Costeira and J. Xavier. Contour Point Tracking by Enforcement of Rigidity Constraints. In *Proc. 3DIM*, June 2005.

[18] R. Oliveira, J. Costeira and J. Xavier. Optimal Point Correspondence of Contour Points Through the Use of Rank Constraints. In *Proc. CVPR*, June 2005.

[19] C. J. Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. In *Proc. ECCV*, pp. 97-108, August 1994.

[20] C. Rother. Linear Multi-View Reconstruction of Points, Lines, Planes and Cameras using a Reference Plane. In *Proc. ICCV*, October 2003.

[21] S. Roy and I. Cox. A Maximum-Flow Formulation of the N-Camera Stereo Correspondence Problem. In *Proc. ICCV*, January 1998.

[22] K. Shafique and M. Shah. A Non-Iterative Greedy Algorithm for Multi-frame Point Correspondence. In *Proc. ICCV*, October 2003.

[23] P. Sturm and B. Triggs. A factorization based algorithm for multi-image projective structure and motion. In *Proc. ECCV*, pp. 709-720, April 1996.

[24] C. Tomasi and T. Kanade. Shape from motion from image streams under orthography: a factorization method. *IJCV*,9(2):137-154, November 1992.