FORTH-ICS / TR-324 September 2003

Efficient 3D Camera Matchmoving Using Markerless, Segmentation-Free Plane Tracking†

Manolis I.A. Lourakis and Antonis A. Argyros

We address the problem of tracking the position and orientation of a camera in 3D space, using the images it acquires while moving freely in unmodeled, arbitrary environments. This task has a broad spectrum of useful applications in domains such as augmented reality and video post production. Most of the existing methods for camera tracking are designed to operate in a batch, off-line mode, assuming that the whole video sequence to be tracked is available before tracking commences. Typically, such methods operate non-causally, processing video frames backwards and forwards in time as they see fit. Furthermore, they resort to optimization in very high dimensional spaces, a process that is computationally intensive. For these reasons, batch methods are inapplicable to tracking in on-line, time-critical applications such as video see-through augmented reality. This paper puts forward a novel feature-based approach for camera tracking. The proposed approach operates continuously as images are acquired, has realistic computational requirements and does not require modifications of the environment. At its core lies a novel, feature-based 3D plane tracking technique, which permits the estimation of the homographies induced by a virtual 3D plane between successive image pairs. Knowledge of these homographies allows the corresponding projection matrices encoding camera motion to be expressed in a common projective frame and, therefore, to be recovered directly. Projective camera matrices are then upgraded to Euclidean and used for recoverying 3D structure, which is in turn employed for refining the projection matrices through local bundle adjustment. Sample experimental results demonstrating the feasibility of the approach on several image sequences are also provided.

† This work was partially supported by the EU IST-2001-34545 project LifePlus.

Efficient 3D Camera Matchmoving Using Markerless, Segmentation-Free Plane Tracking

Manolis I.A. Lourakis and Antonis A. Argyros

Computational Vision and Robotics Laboratory
Institute of Computer Science (ICS)
Foundation for Research and Technology — Hellas (FORTH)
Science and Technology Park, Heraklio, Crete
P.O. Box 1385, GR-711-10 Greece

Web: http://www.ics.forth.gr/cvrl
E-mail: {lourakis | argyros}@ics.forth.gr
Tel: +30 2810 391716, Fax: +30 2810 391601

Technical Report FORTH-ICS / TR-324 — September 2003

We address the problem of tracking the position and orientation of a camera in 3D space, using the images it acquires while moving freely in unmodeled, arbitrary environments. This task has a broad spectrum of useful applications in domains such as augmented reality and video post production. Most of the existing methods for camera tracking are designed to operate in a batch, off-line mode, assuming that the whole video sequence to be tracked is available before tracking commences. Typically, such methods operate non-causally, processing video frames backwards and forwards in time as they see fit. Furthermore, they resort to optimization in very high dimensional spaces, a process that is computationally intensive. For these reasons, batch methods are inapplicable to tracking in on-line, time-critical applications such as video see-through augmented reality. This paper puts forward a novel feature-based approach for camera tracking. The proposed approach operates continuously as images are acquired, has realistic computational requirements and does not require modifications of the environment. At its core lies a novel, feature-based 3D plane tracking technique, which permits the estimation of the homographies induced by a virtual 3D plane between successive image pairs. Knowledge of these homographies allows the corresponding projection matrices encoding camera motion to be expressed in a common projective frame and, therefore, to be recovered directly. Projective camera matrices are then upgraded to Euclidean and used for recoverying 3D structure, which is in turn employed for refining the projection matrices through local bundle adjustment. Sample experimental results demonstrating the feasibility of the approach on several image sequences are also provided.

1 Introduction

Camera matchmoving is an application involving synthesis of real scenes and artificial objects, in which the goal is to insert computer-generated graphical 3D objects into live-action footage depicting unmodeled, arbitrary scenes. Graphical objects should be inserted in a way so that they appear to move as if they were a part of the real scene. Seamless, convincing insertion of graphical objects calls for accurate 3D camera motion tracking (i.e. pose estimation), stable enough over extended sequences so as to avoid the problems of jitter and drift in the location and appearance of objects with respect to the real scene. Additionally, the placement of the objects with respect to the real scene often requires the availability of some 3D geometry information; for instance, accurate 3D reconstruction of a few guiding control points is in most cases sufficient. Matchmoving finds several important applications in augmented reality as well as virtual studio shooting and the creation of special effects in the post-production/film-making industry [40]. To provide the versatility required by such applications, very demanding camera tracking requirements, both in terms of accuracy and speed, are imposed [4].

Optical and electromechanical camera tracking are technologies that have successfully proven themselves in applications such as live TV broadcasting [41]. Nevertheless, apart from suffering from range limitations, such technologies call for special modifications of the environment that render them inapplicable for tracking in unprepared, unstructured scenes, large scale environments or archive footage. Being non-intrusive, passive and capable of covering large fields of view, computer vision techniques provide an attractive alternative to optical and electromechanical tracking methods for recovering camera motion. Tracking the position and orientation of a camera using the images it acquires while moving freely in unmodeled, arbitrary environments, is a very challenging problem in visual motion analysis. During the last fifteen years, numerous research efforts have focused on vision-based camera tracking within the framework of the more general structure from motion problem [14]. Before briefly reviewing a few representative ones, it is pointed out that our requirement for operation in unprepared environments exclude methods such as [23, 22] that rely upon the presence of fiducial markers or special calibration objects in the environment. Additionally, due to the problems pertaining to the accurate estimation of optical flow when the inter-frame image motion is not infinitesimal, we have chosen to focus our attention to feature-based approaches only. For more details regarding direct, i.e. flow-based approaches, see [24] and references therein.

Methods that avoid making any assumptions regarding the environment exploit geometric constraints that arise from the automatic extraction and matching of appropriate 2D image features such as corner points. Depending on their mode of operation, proposed approaches can be classified into two categories. The first category consists of methods designed for off-line use on pre-recorded image sequences. Such methods process image data in a batch mode and usually are non-causal, employing both past and future frames for deducing the camera motion corresponding to the current frame.

Viéville et al [39], for instance, rely upon a set of point and line features extracted from an uncalibrated monocular image sequence and recover structure and motion by formulating a huge non-linear optimization problem that is solved by alternating between structure and motion estimation. Cornelis et al [8, 7] describe a system that relies on pairwise fundamental matrices for simultaneously recovering structure and motion for augmented reality applications. Camera motions for all frames are determined in a single final step using bundle adjustment. Fitzgibbon and Zisserman [13] take a different direction and recover structure and motion using a hierarchical bundle adjustment approach based on image triplets and associated trifocal tensors. Commercially available camera tracking software products such as boujou™, MatchMover™, 3D-equalizer™and PFTrack™also fall into this category[1]. Albeit accurate, batch techniques share the limitation of being extremely time-consuming due to the use of global bundle adjustment, which involves the solution of large, non-linear optimization problems [38]. This, plus the requirement of operating on the whole sequence at once, makes batch methods inappropriate for use in time-critical applications.

Methods operating in a continuous mode, in which images are processed incrementally as acquired, constitute the second class of camera tracking techniques. Typically, such methods exploit the natural ordering of images and are causal, i.e. they rely only on past frames for estimating the camera motion for the current image. The work by Beardsley et al [5], who estimate camera matrices from 3D structure that is recovered incrementally, was one of the first to propose such an approach. In some cases, structure recovery is completely avoided. For example, Simon et al [37] describe a camera tracking system that relies on continuous tracking of a 3D plane that is assumed to be present in the scene. The plane is tracked by estimating pairwise planar homographies with the aid of tracked interest points. The disadvantage of this technique is that it requires the tracked plane to be continuously visible and its image segmented from the rest of the scene. Moreover, the tracking scheme employed requires manual intervention to bootstrap and cannot incorporate information from points off the plane or from more than two images. Avidan and Shashua [3] follow a direct approach for recovering a set of consistent projective camera matrices without reconstructing the 3D scene. The main contribution of this work is a "threading" operation on two consecutive fundamental matrices that uses the trifocal tensor as the connecting thread. Their method is based on tracking a scene plane along an image sequence and provides, as a byproduct, the homography matrices it induces between adjacent views. However, owing to the use of constraints involving algebraic distances, the estimated homographies are not statistically optimal. Besides, the experimental results provided mainly focus on the performance of plane tracking rather than on the recovery of the underlying Euclidean camera motion and its accuracy. Zhang and Shan [44] propose yet another incremental motion estimation algorithm, which simultaneously recovers motion and 3D structure by applying a series of local bundle adjustments to

[1] See http://www.2d3.com, http://www.realvis.com, http://www.3dequalizer.com and http://www.thepixelfarm.co.uk respectively.

a sliding window of image triplets. By restricting the permissible camera motions to pure rotations, Prince et al [32] propose a camera tracking algorithm for augmented reality applications. Despite its computational efficiency, however, its applicability is severely limited by the restricted camera motion model it employs. By introducing probabilistic simultaneous localization and mapping (SLAM) techniques in computer vision, Davison [9] has recently proposed an interesting approach to camera tracking. By employing a camera motion model and explicitly modeling uncertainty, his method is capable of determining in real-time the camera position/orientation and recovering sparse 3D information regarding the environment. Still, the method relies upon certain prior knowledge regarding the imaged environment, requires manual intervention for bootstrapping and employs image features that are of limited viewpoint-invariance, thus restricting its application in small scale environments.

In this paper, a novel feature-based approach to camera tracking is presented. The method is based on tracking a 3D plane through a homography "chaining" operation that is applied to triplets of consecutive images through a sliding time window and exploits the fact that all images of a planar surface acquired by a rigidly moving observer depend upon the same 3D geometry. The tracked plane is not required to be physically present in the scene; it can be a virtual one. Plane tracking is achieved by tracking the 2D projections of points from all over the scene. By doing so, all information conveyed by matching points is taken into account, without the need for continuously maintaining a segmentation of the tracked plane from the scene. The motion model estimated for the tracked plane is exact and fully projective (i.e. a homography) and no camera calibration information or 3D structure recovery is necessary. Knowledge of the homographies induced by the virtual 3D plane between each pair of successive images, allows the corresponding projection matrices encoding camera motion to be expressed in a common projective frame and therefore to be recovered directly, without the need for retrieving structure. Then, 3D structure is recovered from the projection matrices and used for refining the latter through local bundle adjustment. Intended for use in close to on-line applications, the proposed method is designed to operate in a continuous mode. The proposed method follows a strategy similar to [3]. However, it is based on much simpler constraints whose derivation is shorter and does not involve neither the trifocal tensor nor tensorial notation. Moreover, our method tracks 3D planes by minimizing a geometrically meaningful criterion with respect to a set of four free parameters, which, according to the subspace constraint of [34], is a theoretically minimal one. Compared to [3] and [37] which estimate twelve and eight parameters respectively, the estimation of just four parameters is both faster and more accurate.

The rest of the paper is organized as follows. Section 2 explains the notation that will be used throughout all equations and provides some background knowledge. Section 3 briefly describes plane tracking and Section 4 builds upon it for solving the problem of camera tracking. Since the tracked plane is not required to be physically present in the scene, any virtual 3D plane suffices for the purposes of camera track-

ing. Section 5 explains how can such a virtual plane be selected. Section 6 deals with the problem of aligining the coordinate system used for camera tracking with that employed internally by a graphics subsystem. Implementation issues and sample experimental results are reported in Section 7. The paper is concluded with a brief discussion in Section 8.

2 Notation and Background

In the following, vectors and arrays appear in boldface and are represented using projective (homogeneous) coordinates. The symbol $\simeq$ denotes equality of vectors up to an arbitrary scale factor. 3D points are written in uppercase, while their image projections in lowercase (e.g. $\mathbf{X}$ and $\mathbf{x}$).

A well-known constraint for a pair of perspective views of a rigid scene is the *epipolar constraint*. This constraint states that for each point in one of the images, the corresponding point in the other image must lie on a straight line. Assuming that no calibration information is available, the epipolar constraint is expressed mathematically by a 3×3 singular matrix, known as the *fundamental matrix* and denoted by $\mathbf{F}$. Denoting by O and $O^{'}$ the centers of projection corresponding to the two perspective views, the points of intersection of the line $\overline{OO'}$ with the first and second image planes are the *epipoles*, depending on relative translational motion only and being denoted by $\mathbf{e}$ and $\mathbf{e}^{'}$, respectively. For example, epipole $\mathbf{e}^{'}$ corresponds to the uncalibrated translational component of the camera motion from the first to the second image. Given $\mathbf{F}$, the epipoles can be recovered by finding the kernels of $\mathbf{F}$ and $\mathbf{F}^T$. Another important concept in projective geometry is the *plane homography* $\mathbf{H}$, a nonsingular 3×3 matrix which relates two uncalibrated retinal images of a 3D plane. More specifically, if $\mathbf{x}$ is the projection in one view of a point on the plane and $\mathbf{x}^{'}$ is the corresponding projection in a second view, then the two projections are related by the linear projective transformation

$$\mathbf{x}^{'} \simeq \mathbf{H}\mathbf{x}. \tag{1}$$

For more detailed treatments of the application of projective geometry to computer vision, the interested reader is referred to [18, 11].

As shown in [34], the fundamental matrix and plane homographies are tightly coupled. More specifically, the entire group of all possible homography matrices between two images lies in a subspace of dimension 4, i.e. it is spanned by 4 homography matrices. These 4 homography matrices are such that their respective planes do not all coincide with a single point. Shashua and Avidan show in [2] that given the fundamental matrix $\mathbf{F}$ and the epipoles $\mathbf{e}$ and $\mathbf{e}^{'}$ in an image pair, a suitable basis of 4 homography matrices $\mathbf{H}_1, \ldots, \mathbf{H}_4$, referred to as "primitive homographies", is defined as follows

$$\mathbf{H}_i = [\epsilon_i]_\times \mathbf{F}, \quad i = 1, 2, 3 \qquad \text{and} \qquad \mathbf{H}_4 = \mathbf{e}^{'} \delta^T, \tag{2}$$

where ϵ_i are the identity vectors $\epsilon_1 = (1, 0, 0)$, $\epsilon_2 = (0, 1, 0)$ and $\epsilon_3 = (0, 0, 1)$, $[.]_\times$

designates the skew symmetric matrix representing the vector cross product (i.e. for a vector $\mathbf{a}$, $[\mathbf{a}]_\times$ is such that $[\mathbf{a}]_\times \mathbf{b} = \mathbf{a} \times \mathbf{b}, \ \forall\, \mathbf{b}$) and δ is a vector such that $\delta^T \mathbf{e} \neq 0$. This last requirement can, for example, be satisfied by defining vector δ so that each of its elements has an absolute value of 1 and a sign identical to that of the corresponding element of $\mathbf{e}$. The first three homography matrices ($\mathbf{H}_1$, $\mathbf{H}_2$ and $\mathbf{H}_3$) are of rank 2 and span the subgroup of homography matrices whose underlying 3D planes contain the center of projection $O^{'}$ of the second camera. On the other hand, $\mathbf{H}_4$ by definition corresponds to a 3D plane not coincident with $O^{'}$ but going through the center of projection O of the first camera, thus having rank 1. Knowledge of the 4 primitive homographies allows any other homography $\mathbf{H}$ to be expressed as a linear combination

$$\mathbf{H} = \sum_{i=1}^{4} \lambda_i \mathbf{H}_i, \tag{3}$$

for some scalars λ_i.

Next, a result due to Shashua and Navab [35] that plays a central role in the development of the proposed method is presented. Let Π be an arbitrary 3D plane inducing a homography $\mathbf{H}$ between two images. Let also $\mathbf{X}_0$ be a 3D point not on Π projecting to image points $\mathbf{x}_0$ and $\mathbf{x}_0^{'}$ and assume that $\mathbf{H}$ has been scaled to satisfy the equation $\mathbf{x}_0^{'} \simeq \mathbf{H}\mathbf{x}_0 + \mathbf{e}^{'}$. Then, for any 3D point $\mathbf{X}$ projecting onto $\mathbf{x}$ and $\mathbf{x}^{'}$, there exists a scalar κ such that

$$\mathbf{x}^{'} \simeq \mathbf{H}\mathbf{x} + \kappa \mathbf{e}^{'} \tag{4}$$

Equation (4) dictates that the position of projected points in the second image can be decomposed into the sum of two terms, the first depending on the homography induced by Π and the second involving *parallax* due to the deviation of the actual 3D structure from Π. The term κ in Eq. (4) depends on $\mathbf{X}$ but is invariant to the choice of the second image and is termed as *relative affine structure* in [35]. Given $\mathbf{x}$, $\mathbf{x}^{'}$, $\mathbf{H}$ and $\mathbf{e}^{'}$, the term κ corresponding to $\mathbf{X}$ can be computed by cross-multiplying both sides of Eq. (4) with $\mathbf{x}^{'}$, which after some algebraic manipulation yields

$$\kappa = \frac{(\mathbf{H}\mathbf{x} \times \mathbf{x}^{'})^T (\mathbf{x}^{'} \times \mathbf{e}^{'})}{||\mathbf{x}^{'} \times \mathbf{e}^{'}||^2} \tag{5}$$

3 3D Plane Tracking

Let us begin by explaining the role of point $\mathbf{X}_0$ in the derivation of Eq. (4). Recall that $\mathbf{H}$ and $\mathbf{e}^{'}$ are homogeneous entities, defined up to an arbitrary scale factor. Therefore, by fixing $\mathbf{H}$'s scale, $\mathbf{X}_0$ serves to establish a common relative scale between $\mathbf{H}$ and $\mathbf{e}^{'}$. Notice, however, that in the case that $\mathbf{H}$ has not been scaled with the aid of $\mathbf{X}_0$, Eq. (4) continues to hold for some $\kappa^{'}$ that is a scaled version of κ given by Eq. (5). In addition, in this case κ is not invariant to the choice of the second view. What remains invariant though, is the ratios of κ's computed from different image pairs.

Suppose now that three consecutive images I_1, I_2 and I_3 are available and that a planar homography between I_1 and I_2 has been estimated. Considering the two pairs (I_1, I_2) and (I_2, I_3) formed by the three images, a key observation is the fact that image I_2 is shared by both of these pairs. Hence, the relative affine structure defined when I_2 assumes the role of the first image in Eq. (4) is insensitive to the choice of the second image (i.e. I_1 or I_3) completing the pair. This allows one to estimate the relative affine structure from the pair (I_1, I_2) and the corresponding homography and then use this estimate for computing the plane homography for the pair (I_2, I_3). This, in effect, constitutes a chaining operation involving plane homographies. The process just outlined is explained in more detail in the next section.

3.1 Chaining Homographies Among Consecutive Frames

Assume that N triplets of matching points ($\mathbf{x}_i$, $\mathbf{x}_i^{'}$, $\mathbf{x}_i^{''}$), $i = 1, \ldots, N$, are available across the three images I_1, I_2 and I_3 respectively and that the homography $\mathbf{U}$ from image I_1 to I_2 due to some 3D plane has been estimated. In the remainder of this section, a procedure for estimating the plane homography $\mathbf{V}$ induced by this 3D plane between images I_2 and I_3 will be described.

From the set of matching pairs ($\mathbf{x}_i$, $\mathbf{x}_i^{'}$) the epipolar geometry for images I_1 and I_2 and thus the epipole $\mathbf{e}$ in image I_1 can be estimated. In a similar manner, the epipole $\mathbf{e}^{''}$ in I_3 for the camera motion corresponding to frames I_2 and I_3 can be estimated from the set of matching pairs ($\mathbf{x}_i^{'}$, $\mathbf{x}_i^{''}$). Recalling that the homography from image I_2 to I_1 is simply $\mathbf{U}^{-1}$, Eq. (4) takes the following form for all point matches in those two images

$$\mathbf{x}_i \simeq \mathbf{U}^{-1}\mathbf{x}_i^{'} + \kappa_i \mathbf{e}. \tag{6}$$

By employing Eq. (5), κ_i can then be estimated as

$$\kappa_i = \frac{(\mathbf{U}^{-1}\mathbf{x}_i^{'} \times \mathbf{x}_i)^T (\mathbf{x}_i \times \mathbf{e})}{||\mathbf{x}_i \times \mathbf{e}||^2}. \tag{7}$$

Taking into account point matches in frames I_2 and I_3, Eq. (4) gives

$$\mathbf{x}_i^{''} \simeq \mathbf{V}\mathbf{x}_i^{'} + \kappa_i \mathbf{e}^{''}, \tag{8}$$

where the κ_i are given by Eq. (7). In order for Eq. (8) to hold for the κ_i given by Eq. (7), the scale of $\mathbf{V}$ in it has to be compatible with that of the estimated $\mathbf{e}^{''}$. For this reason, $\mathbf{V}$ in Eq. (8) is no longer a homogeneous 3×3 matrix but rather an ordinary, inhomogeneous one. Equation (8) is thus a vector equation linear in $\mathbf{V}$, providing three linear constraints on the nine unknown elements of $\mathbf{V}$. Due to the presence of an arbitrary, unknown scale factor, only two of those three constraints are linearly independent. Denoting the i-th row of matrix $\mathbf{V}$ by $\mathbf{v}_i^T$, writing $\mathbf{x}_i^{''} = (x_i^{''}, y_i^{''}, 1)^T$ and $\mathbf{e}^{''} = (e_x^{''}, e_y^{''}, e_z^{''})^T$, those two constraints can be explicitly expressed as [2]

$$\mathbf{v}_3^T \mathbf{x}_i^{'} x_i^{''} - \mathbf{v}_1^T \mathbf{x}_i^{'} = \kappa_i e_x^{''} - \kappa_i e_z^{''} x_i^{''}$$

[2]Notice that all available point matches are assumed to originate from actual image points (i.e. corners); no ideal points whose third coordinate is zero exist among them.

$$\mathbf{v}_3^T \mathbf{x}_i^{'} y_i^{''} - \mathbf{v}_2^T \mathbf{x}_i^{'} = \kappa_i e_y^{''} - \kappa_i e_z^{''} y_i^{''}. \quad (9)$$

Notice that Eqs. (9) do not require that the employed point matches have been identified as lying on the plane. Therefore, they do not require that the tracked plane has been segmented from the rest of the scene and are applicable even in the case of tracking a virtual (i.e. not physically present in the scene) plane. Since $\mathbf{v}_j^T \mathbf{x}_i^{'} = \mathbf{x}_i^{'T} \mathbf{v}_j$, Eqs. (9) can be written in matrix form as

$$\begin{bmatrix} -\mathbf{x}_i^{'T} & \mathbf{0}^T & \mathbf{x}_i^{'T} x_i^{''} \\ \mathbf{0}^T & -\mathbf{x}_i^{'T} & \mathbf{x}_i^{'T} y_i^{''} \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 \\ \mathbf{v}_2 \\ \mathbf{v}_3 \end{bmatrix} = \begin{bmatrix} \kappa_i e_x^{''} - \kappa_i e_z^{''} x_i^{''} \\ \kappa_i e_y^{''} - \kappa_i e_z^{''} y_i^{''} \end{bmatrix}. \quad (10)$$

Thus, each triplet of corresponding points provides two equations in the elements of $\mathbf{V}$. Concatenating the equations arising from five triplet correspondences, a matrix equation of the form $\mathbf{M}\mathbf{v} = \mathbf{b}$ is generated, where $\mathbf{M}$ is a 10×9 matrix, $\mathbf{v}$ is a 9×1 vector equal to $(\mathbf{v}_1^T,\ \mathbf{v}_2^T,\ \mathbf{v}_3^T)^T$ and $\mathbf{b}$ is a 10×1 vector. Omitting any row of matrix $\mathbf{M}$, yields a 9×9 linear system with 9 unknowns that may be solved using Gaussian elimination [16]. In the case that more than five triplet matches are available, Eq. (10) gives rise to an over-constrained system from which $\mathbf{V}$ can be estimated in a least squares manner with the aid of SVD.

According to the terminology of [18], ch. 3, the estimation of $\mathbf{V}$ as described up to this point, is achieved with a Direct Linear Transformation (DLT) algorithm. It is well-known that DLT algorithms are not invariant to similarity transformations of the image but depend on the coordinate system in which image points are expressed. To alleviate this and, at the same time, improve the condition number of the DLT constraints, therefore ameliorating the accuracy of results, the normalization technique of [19] is applied to matching points prior to feeding them to the DLT algorithm. Independently for each image, this normalization consists in translating image coordinates so that the centroid of points is brought to the origin of the coordinate system, followed by an isotropic scaling that maps the average point to $(1, 1, 1)^T$. The normalizing transformation for image i is expressed by a 3×3 linear transformation $\mathbf{L}_i$. Notice that in this case, the normalized version $\bar{\mathbf{U}} = \mathbf{L}_2 \mathbf{U} \mathbf{L}_1^{-1}$ of $\mathbf{U}$ must be employed in Eq. (7) along with the normalized points and epipole. The normalized epipole can be recovered from the normalized fundamental matrix $\bar{\mathbf{F}} = \mathbf{L}_2^{-T} \mathbf{F} \mathbf{L}_1^{-1}$. After the application of DLT, the computed homography estimate $\bar{\mathbf{V}}$ needs to be denormalized using ${\mathbf{L}_3}^{-1} \bar{\mathbf{V}} \mathbf{L}_2$.

In practice, the set of available matching point triplets is almost certain to contain errors due to false matches and errors in the localization of image corners. Consequently, in order to prevent such errors from corrupting the computed homography estimate, the group of DLT constraints should be employed within a robust regression framework. In our case, the Least Median of Squares (LMedS) [33] robust estimator is employed to iteratively sample random sets of nine constraints, recover an estimate of matrix $\mathbf{V}$ from each of them and find the estimate that is consistent with the majority of the available constraints. To ensure that those random sets arise from points

having a good spatial distribution over the image, random sampling is based on the bucketing technique of [43]. Finally, $\mathbf{V}$ is recomputed using least squares on the set of constraints having the largest support, i.e. the LMedS inliers.

Since the DLT constraints minimize an algebraic error term with no physical meaning, the estimate computed by LMedS is refined by a non-linear minimization process that involves a geometric criterion. Letting $d(\mathbf{x},\ \mathbf{y})$ represent the Euclidean distance between the inhomogeneous points represented by $\mathbf{x}$ and $\mathbf{y}$, the non-linear refinement minimizes the following sum of squared distances

$$\sum_i \left(d(\mathbf{x}_i^{''},\ \mathbf{V}\mathbf{x}_i^{'} + \kappa_i \mathbf{e}^{''})^2 \quad + \quad d(\mathbf{x}_i^{'},\ \mathbf{V}^{-1}\mathbf{x}_i^{''} - \frac{||\mathbf{x}_i^{''}||}{||\mathbf{V}\mathbf{x}_i^{'} + \kappa_i \mathbf{e}^{''}||}\kappa_i \mathbf{V}^{-1}\mathbf{e}^{''})^2 \right) \quad (11)$$

with respect to $\mathbf{V}$. This criterion involves the mean symmetric transfer error between actual and transferred points in the two images and is minimized by applying the Levenberg-Marquardt iterative algorithm as implemented by MINPACK's LMDER routine [28], initialized with the least squares estimate from the LMedS inliers. To safeguard against point mismatches, the non-linear refinement is performed using only the point features that correspond to inliers of the LMedS homography estimate.

Having presented the basic 3-frame chaining operation, it is straightforward to extend it to handle a sequence of more than three views. For example, in order to track the plane in a new image I_4, the homography $\mathbf{V}$ computed in the previous step between frames I_2 and I_3 becomes the new $\mathbf{U}$ for the triplet I_2, I_3 and I_4. Note also that the epipolar geometry of frames I_2 and I_3 has been computed from the previous iteration, therefore only the epipolar geometry between frames I_3 and I_4 needs to be estimated during this step. A final remark concerning the extension of the chaining operation to more than three frames is that the estimation of $\mathbf{V}$ can benefit from point trajectories that are longer than three frames: If, for example, a four-frame point trajectory is available for images I_1, I_2, I_3 and I_4, the constraints generated by the triplet I_1, I_3 and I_4 can be combined with those arising from I_2, I_3 and I_4. This variant of chaining from multiple triplets can be carried out by maintaining a small moving window of past frames.

3.2 Reducing the DOFs Involved in Plane Tracking

In the following, the basic method of the previous section will be refined, aiming to derive a model involving fewer, therefore easier to estimate, degrees of freedom (i.e. free variables). As already mentioned, the entire group of all possible homography matrices between two images lies in a subspace of dimension 4, spanned by the 4 primitive homographies of Eq. (2). According to Eq. (3), knowledge of those homographies allows any other homography $\mathbf{H}$ to be expressed as a linear combination encompassing 4 scalars λ_i. This implies that when the primitive homographies for frames I_2 and I_3 have been computed, the rows $\mathbf{v}_i^T$ of matrix $\mathbf{V}$ in Eqs. (9) depend on four rather than nine parameters. Therefore, the process described in section 3.1 can be slightly modified to estimate the coefficients λ_i making up $\mathbf{V}$ instead of directly estimating

1. *Extract and match point features across three frames I_1, I_2 and I_3*
2. *Normalize the coordinates of matched points in each frame*
3. *Estimate the fundamental matrices between frames I_1, I_2 and I_2, I_3 along with the epipoles*
4. *Use Eq. (7) with frames I_1, I_2 to compute the κ_i and then use I_2 and I_3 to form the DLT constraints*
5. *Compute the primitive homographies of frames I_2 and I_3 by employing the corresponding epipolar geometry*
6. *Use the LMedS robust estimator to identify the estimates of λ_i that give rise to an estimate of $\mathbf{V}$ having the largest support from the DLT constraints of Eq. (10)*
7. *Reestimate the λ_i using all DLT constraints corresponding to point inliers identified in step 6*
8. *Starting with the estimate of step 6, refine the λ_i and thus $\mathbf{V}$, using Levenberg-Marquardt non-linear minimization of Eq. (11) over the LMedS inliers in step 6*

Figure 1: An overview of the proposed homography chaining operation; see text for details.

the latter. In other words, both the linear and the non-linear estimation processes that have been described above are performed with four rather than nine unknowns. This reduction in the dimensionality of the problem is of utmost importance since fewer degrees of freedom entail less computation time for the homography (particularly for the non-linear refinement) and typically more accurate estimates. Fewer DOFs involve the solution of smaller systems and require less iterations (i.e. samples taken) to find a solution with a given confidence level when embedded within random sampling schemes such as LMedS or RANSAC [12]. It was found experimentally that the execution time for plane tracking using the formulation involving λ_i is by an order of magnitude shorter than that required when estimating $\mathbf{V}$ directly. Space considerations prevent us from deriving here the exact form of Eqs. (9) and Eq. (10) after introducing the coefficients λ_i. Doing so, however, is straightforward and should not present any particular difficulty to the reader. The complete sequence of steps comprising the homography chaining operation is listed in pseudocode in Fig. 1.

4 Camera Tracking

In this section, $\mathbf{H}_{i,j}$ and $\mathbf{e}_{i,j}$ will be used to denote respectively the tracked plane homography and the epipole in I_j for the image pair I_i and I_j. Assume also that $\mathbf{H}_{1,2}$ has been supplied and, using the method outlined in Section 3, the plane homography $\mathbf{H}_{2,3}$ has been estimated from the matching triplets among images I_1, I_2 and I_3.

Recalling that these homographies are, by computation, scale compatible with the corresponding epipoles, Eq. (4) yields the image projections of a 3D point $\mathbf{X}$ as $\mathbf{x} \simeq \mathbf{H}_{2,1}\mathbf{x}^{'} + \kappa\mathbf{e}_{2,1}$ and $\mathbf{x}^{''} \simeq \mathbf{H}_{2,3}\mathbf{x}^{'} + \kappa\mathbf{e}_{2,3}$, implying that $\mathbf{X} \simeq \left[\mathbf{x}^{'T},\ \kappa\right]^{T}$. Therefore, a consistent set of projective camera matrices in canonical form for the three views is given by [18, 3]

$$\mathbf{P}_1 = [\mathbf{H}_{2,1} \mid \mathbf{e}_{2,1}]\,, \quad \mathbf{P}_2 = [\mathbf{I} \mid \mathbf{0}]\,, \quad \mathbf{P}_3 = [\mathbf{H}_{2,3} \mid \mathbf{e}_{2,3}]\,, \tag{12}$$

where $\mathbf{I}$ denotes the 3×3 identity matrix. Since it is customary to express the camera matrices relative to the first image I_1, application of an appropriate 3D projective mapping can transform Eqs.(12) so that $\mathbf{P}_1$ becomes equal to $[\mathbf{I} \mid \mathbf{0}]$. Indeed, right multiplication of the camera matrix $[\mathbf{A} \mid \mathbf{b}]$ by the 4×4 matrix $\mathbf{M}$ given by

$$\mathbf{M} = \left[\begin{array}{c|c} \mathbf{A}^{-1} & -\mathbf{A}^{-1}\,\mathbf{b} \\ \\ \mathbf{0}^T & 1 \end{array}\right] \tag{13}$$

makes the former equal to $[\mathbf{I} \mid \mathbf{0}]$. Therefore, to make $\mathbf{P}_1$ equal to $[\mathbf{I} \mid \mathbf{0}]$, the projection matrices in Eq. (12) should be right multiplied by the matrix given by Eq.(13) for $\mathbf{A} = \mathbf{H}_{2,1}$ and $\mathbf{b} = \mathbf{e}_{2,1}$, which, taking into account that $\mathbf{H}_{j,i}^{-1}\,\mathbf{e}_{j,i} = \mathbf{e}_{i,j}$ and $\mathbf{H}_{2,3}\,\mathbf{H}_{1,2} = \mathbf{H}_{1,3}$, yields after some algebraic manipulation

$$\mathbf{P}_1 = [\mathbf{I} \mid \mathbf{0}]\,, \quad \mathbf{P}_2 = [\mathbf{H}_{1,2} \mid \mathbf{e}_{1,2}]\,, \quad \mathbf{P}_3 = [\mathbf{H}_{1,3} \mid \mathbf{e}_{1,3}]\,. \tag{14}$$

Suppose now that by employing the plane tracker for the image triplet I_2, I_3 and I_4, the homography $\mathbf{H}_{34}$ induced by the tracked plane has been estimated. If $\mathbf{P}_3$ were equal to $[\mathbf{I} \mid \mathbf{0}]$, a projection matrix for I_4 consistent with the projection matrices of the previous three images would simply be $[\mathbf{H}_{3,4} \mid \mathbf{e}_{3,4}]$. Here, the former should be right multiplied by the matrix given by Eq.(13) for $\mathbf{A} = \mathbf{H}_{1,3}$ and $\mathbf{b} = \mathbf{e}_{1,3}$, to account for the fact that the employed coordinate system coincides with that of I_1. Thus, $\mathbf{P}_4$ is equal to $[\mathbf{H}_{3,4}\,\mathbf{H}_{1,3} \mid \mathbf{H}_{3,4}\,\mathbf{e}_{1,3} + \mathbf{e}_{3,4}]$, which in turn is simplified to $[\mathbf{H}_{1,4} \mid \mathbf{e}_{1,4}]$. Clearly, the procedure for obtaining $\mathbf{P}_4$ just described, can be generalized to incorporate the projection matrix $\mathbf{P}_i$ corresponding to any image I_i with $i > 4$. Hitherto, knowledge of the plane homographies has permitted the direct recovery of a set of consistent projective camera matrices, without the need for 3D structure estimation and resectioning.

In order to increase stability and at the same time relieve the camera tracker from the associated computational burden, the camera intrinsic calibration parameters are assumed here to be constant and known, either as a result of a self-calibration algorithm or of an off-line, grid based calibration method [26]. Given the 3×3 camera intrinsic calibration matrix $\mathbf{K}$, a projective camera matrix can be upgraded to Euclidean by right multiplication with the 4×4 matrix defined as

$$\left[\begin{array}{c|c} \mathbf{K} & \mathbf{0} \\ \\ -\mathbf{p}^T\,\mathbf{K} & 1 \end{array}\right], \tag{15}$$

where $\mathbf{p}$ is such that the coordinates of the plane at infinity in the projective reconstruction are given by $\left[\mathbf{p}^T, 1\right]^T$. Following this, the 3D translation and rotation corresponding to the Euclidean camera matrix can be estimated with RQ decomposition [18].

A rough representation of 3D scene structure in the form of a point cloud can be built-up incrementally as new image triplets become available. More specifically, when the camera matrices for a new image triplet have been estimated, the 3D coordinates of points that became visible in the new triplet can be recovered with the aid of a triangulation algorithm [17]. The main problem that needs to be addressed by all triangulation algorithms is the fact that errors in the estimates of camera matrices result in making skew the back-projected 3D lines defined by the camera optical centers and the corresponding image projections. In this work, 3D structure recovery can be achieved in two different ways. The first employs the three view triangulation method of Avidan and Shashua [2]. This method aims to account for the error introduced by the corner detection and camera matrix estimation processes. To accomplish this, the method modifies the coordinates of image points in a way that guarantees that they satisfy the trifocal constraints exactly, while remaining at a minimal distance from their original locations. Then, since the back-projected lines defined by the corrected image points are certain to intersect in space, the corresponding 3D point is determined by a simple linear method [17]. Notice that the 3D point determined in this manner is expressed in a projective coordinate frame; left multiplication by the inverse of the matrix in Eq. (15) upgrades it to Euclidean [18]. The second triangulation method exploits the knowledge of the camera intrinsic parameters to express two back-projected 3D lines in an Euclidean coordinate frame. Then, a 3D point is reconstructed as the midpoint of the minimal length straight line segment whose endpoints lie on the skew back-projected lines [15]. Since an image triplet gives rise to three different image pairs, the reconstructed point is taken here to be the centroid of the three 3D points reconstructed from the triplets' image pairs. To avoid reconstructing points arising from triplets $(\mathbf{x},\ \mathbf{x}^{'},\ \mathbf{x}^{''})$ involving spurious matches, the projection matrices of Eq. (12) are used to compute the corresponding trifocal tensor using a closed form formula [21]. Then, using point matches $\mathbf{x},\ \mathbf{x}^{'}$ from the first two views along with the computed tensor, permits the elimination of point triplets whose transferred third image point lies at a distance further than a certain threshold from the point $\mathbf{x}^{''}$.

By trading some speed for increased accuracy in camera tracking, structure information can be used in a local bundle adjustment framework for evenly distributing the camera tracking error among consecutive images belonging to the same sliding time window. Assume that a narrow window of W past frames in some of which M Euclidean 3D points $\mathbf{X}_j,\ j = 1 \ldots M$ are visible is maintained and let $\mathbf{R}_i$ and $\mathbf{t}_i$ be the estimates of camera orientation and position for frame i. Then, the Euclidean projection matrix $\mathbf{P}_i$ is equal to $\mathbf{K}\ [\mathbf{R}_i \mid \mathbf{t}_i]$. Bundle adjustment amounts to estimating the motion parameters $\mathbf{R}_i$ and $\mathbf{t}_i,\ i = 1 \ldots W$ so that the sum of squared image distances

between reprojected and detected, actual image points is minimized, namely

$$\min_{\mathbf{R}_i,\mathbf{t}_i} \sum_{i=1}^{W} \sum_{j} d(\mathbf{P}_i\,\mathbf{X}_j,\ \mathbf{x}_j^i)^2 \quad \text{with} \quad \mathbf{P}_i = \mathbf{K}\,[\mathbf{R}_i \mid \mathbf{t}_i], \tag{16}$$

where $d(\mathbf{x},\ \mathbf{y})$ denotes the Euclidean distance between the inhomogeneous image points represented by $\mathbf{x}$ and $\mathbf{y}$, and $\mathbf{x}_j^i$ is the detected projection of point j in image i, $i \in \{1,\ldots,W\}$. Rotation matrices $\mathbf{R}_i$ are parametrized using Euler angles; however, a parametrization based on quaternions should work equally well. To keep the computational overhead of the minimization low, observe that Eq. (16) is minimized with respect to the 3D motion only and not the 3D structure. The minimization of Eq. (16) is performed with the aid of a non-linear least squares algorithm [10] that is initialized with the motion parameters computed directly from the tracked plane homography. Finally, since the projective camera matrix for I_i will be needed for determining the camera motion of subsequent frames, it is recomputed as the product of the Euclidean camera matrix amounting to the refined camera motion by the inverse of the matrix in Eq. (15). The complete algorithm for camera tracking using the homographies of a tracked plane is listed in pseudocode in Fig. 2.

1. Initialization: Select a plane in I_1 and I_2 and estimate the homography $\mathbf{H}_{1,2}$
2. For each image I_n, $n > 2$ do
2.a Use the algorithm of Fig. 1 to track the plane between images I_{n-1} and I_n, estimating $\mathbf{H}_{n-1,n}$
2.b if n=3 then
Use Eqs. (14) to compute a canonical set of projection matrices for images I_1, I_2 and I_3
else
Let $\mathbf{P}_{n-1} = [\mathbf{H}_{1,n-1} \mid \mathbf{e}_{1,n-1}]$ be the projection matrix for I_{n-1}
Then, $\mathbf{P}_n := [\mathbf{H}_{n-1,n}\,\mathbf{H}_{1,n-1} \mid \mathbf{H}_{n-1,n}\,\mathbf{e}_{1,n-1} + \mathbf{e}_{n-1,n}] \equiv [\mathbf{H}_{1,n} \mid \mathbf{e}_{1,n}]$
endif
2.c Use $\mathbf{P}_{n-2}$, $\mathbf{P}_{n-1}$ and $\mathbf{P}_n$ to reconstruct 3D points whose projections have appeared for the first time in all images I_{n-2}, I_{n-1} and I_n
2.d Upgrade $\mathbf{P}_n$ to Euclidean using the intrinsic calibration parameters
2.e Recover camera motion using the RQ decomposition of the Euclidean $\mathbf{P}_n$
2.f Refine camera motion through the bundle adjustment of Eq. (16)
2.g Recompute the projective $\mathbf{P}_n$ using the intrinsic parameters and the refined camera motion
endfor

Figure 2: Camera tracking based on tracked plane homographies; see Section 4 for details.

5 Using a Quasi-Metric Virtual Plane

As explained in Section 4, camera tracking requires a 3D plane to be tracked over an image sequence. Although planes abound in man-made environments and fully automatic methods exist for detecting them [25], it would be preferable if the proposed method did not rely on the assumption of a physical 3D plane being present in the scene. To achieve this, recall that any plane is adequate for camera tracking as long as it can be tracked along the whole sequence. In order for plane tracking to commence, the plane homography induced among the first two frames of the sequence must be available. Apart from this requirement, however, no other information regarding the plane must be supplied. The tracked plane can actually be a virtual one, i.e. not corresponding to a physical 3D plane present in the scene. All that is needed is that the plane's homography is compatible with the underlying epipolar geometry. The rest of this Section describes how can such a virtual plane be selected.

Let $\mathbf{x}_i,\ \mathbf{x}_i^{'},\ \ i = 1, \ldots, N$ be a set of matching point pairs in the first two frames. The virtual plane can be chosen so that it approximates the set of available point matches as much as possible. In other words, the virtual plane is situated "in-between" the 3D space points giving rise to the set of available point matches. Assuming that the epipolar geometry corresponding to the two images has been estimated, we therefore seek the planar homography $\mathbf{H}$ for which the contribution of the parallax term in Eq. (4) is as little as possible. It has been explained in Section 2 that any planar homography defined between two images can be expressed as the linear combination of the four primitive homographies of Eq. (2). The sought $\mathbf{H}$ is thus computed from the coefficients $\mu_j,\ \ j = 1, \ldots, 4$ minimizing

$$\sum_{j=1}^{4} (\mu_j\ \mathbf{H}_j)\ \mathbf{x}_i \simeq \mathbf{x}_i^{'}, \quad i = 1, \ldots, N \tag{17}$$

Each of the available point matches provides two independent linear constraints for the μ_j, therefore N matches yield an overdetermined system from which the μ_j can be estimated using robust least squares. The LMedS robust estimator is again employed to find the μ_j corresponding to the homography minimizing Eq. (17) for at least 70% of the available matches; the estimated μ_j are then refined by applying least squares to the constraints corresponding to the LMedS inliers. The plane computed in this manner is referred to as "quasi-metric" in [2] and gives rise to a projective reconstruction of space that is characterized by a small amount of projective distortion.

6 Aligning Coordinate Systems

The camera tracking algorithm described in Section 4 provides the camera trajectory expressed in a coordinate system that coincides with that of the initial location of the camera. Often, augmentation (i.e. computer graphics) modules that need to be combined with the camera tracker use their own coordinate system internally. Thus, it

is clear that some agreement between the two coordinate systems should be achieved. The process of determining the relationship between the coordinate systems employed by the camera tracking and the graphics modules is referred to as coordinate system alignment. When the merging of graphics and video is to be performed off-line, the coordinate system alignment problem can be manually solved by human operators; this is the approach followed by all commercial camera tracking systems. Obviously, in the case of applications such as video see-through augmented reality, this solution is inadequate.

A possible solution to the alignment problem proceeds as follows. Before tracking starts, assume that a reference image of the scene to be tracked is obtained. Assume also that several 2D corners with known 3D coordinates, expressed in the coordinate system used in the graphics module, are selected on the reference image. Such 2D points can for example be supplied by manually identifying the image projections of several "control points" with known 3D coordinates. To bootstrap camera tracking, the camera is required to start its trajectory from a location close to that corresponding to the reference image. The first frame to be tracked is matched with the reference image. Next, the resulting 2D matches along with the 3D - 2D matches that have been predetermined for the reference image permit the extraction of 3D - 2D matches for the first image. A camera pose estimation algorithm [27] can then provide the position and orientation of the camera that acquired the first image in the coordinate system of the graphics module. Using the pose of the first frame, the camera motion estimated for all subsequent frames is then appropriately transformed so as to be expressed in the coordinate system of the graphics module rather than that of the first frame. More specifically, assume that the pose of the first frame in the graphics coordinate system is specified by a rotation matrix $\mathbf{R}_1$ and a translation vector $\mathbf{t}_1$. Assume also that the relative pose of frame $i > 1$ expressed in the coordinate system of the first frame is given by $\mathbf{\Delta R}_i$ and $\mathbf{\Delta t}_i$. Then, the pose of frame i in the graphics coordinate system is given by

$$\begin{aligned} \mathbf{R}_i &= \mathbf{\Delta R}_i \, \mathbf{R}_1 \\ \mathbf{t}_i &= \mathbf{\Delta R}_i \, \mathbf{t}_1 + \mathbf{\Delta t}_i. \end{aligned}$$

It should be noted, however, that in the case that no prior 3D information regarding the scene is available, the alignment problem can only be solved interactively, by manually rotating and translating the 3D structure recovered by camera tracking until it lines up with the 3D graphical objects that are to be inserted in images.

7 Implementation and Experimental Results

A prototype of the proposed camera tracking method has been implemented in C. Linear algebra numerical operations were carried out using LAPACK [1]. The point features that are required as input have been extracted and matched automatically with the aid of the KLT corner tracker [36]. Using the resulting matches, the epipoles

were computed by applying SVD on the fundamental matrices estimated using an implementation of [43]. A technique that directly derives the epipoles from point matches [6] has also been evaluated and was found to produce similar results.

The current implementation of the plane tracker performs chaining based on constraints arising from three frames at a time. Possible camera lens distortions (e.g. radial distortion) are neglected. The intrinsic camera parameters (i.e. focal length, aspect ratio, principal point and skew), were determined by using the auto-calibration method described in [26]. This method exploits constraints arising from a simplified version of the Kruppa equations that is derived with the aid of SVD of pairwise fundamental matrices [20]. Throughout all experiments, the plane homography between the first two images that is necessary for bootstrapping plane tracking was determined as described in Section 5. The second of the triangulation techniques described in Section 4 performed better compared to the first and therefore was chosen for accomplishing 3D reconstruction. W in Eq. (16) is set to 5 and the minimization is carried out using the NL2SOL algorithm [10], as implemented by the DN2G routine in the PORT3 library from Bell Labs. Compared to the Levenberg-Marquardt algorithm as implemented by the LMDER routine [28], NL2SOL was found in this case to converge faster while producing results of similar accuracy. The NLSCON non-linear least squares routine [30] has also been evaluated and yielded results slightly worse than those of DN2G. The jacobians of Eq. (11) and Eq. (16) that are necessary for the non-linear minimizations are computed analytically with the aid of MAPLE™'s symbolic differentiation facilities [3].

Since the performance of plane tracking is crucial for the overall performance of camera tracking, the following subsection presents experimental results demonstrating is effectiveness. Experimental validation results regarding the complete camera tracking system are reported in subsection 7.2.

7.1 Plane Tracking Experiments

In this section, results from three experiments demonstrating the performance of the proposed plane tracker are presented. To aid in the visual interpretation of results, 3D planes that are physically present in the scene have been employed in all experiments. The spatial extend of the employed planes has been defined manually using a polyline in the first frame. Following this, the plane homography between the first two images that is necessary for bootstrapping plane tracking (i.e. $\mathbf{U}$ in Section 3.1), is estimated from the point matches lying within the specified polyline. Alternatively, plane tracking could have been bootstrapped by applying to the first pair of images an automatic plane detection algorithm such as [25].

The first experiment was performed on the well-known "basement" image sequence, two frames of which (namely 0 and 8) are shown in Figs. 3(a) and (b). This sequence consists of 11 frames acquired by a camera mounted on a mobile robot as it approached the scene while smoothly turning left. The plane corresponding to the

[3] Analytical differentiation was prefered over numerical one owing to its better performance and convergence characteristics.

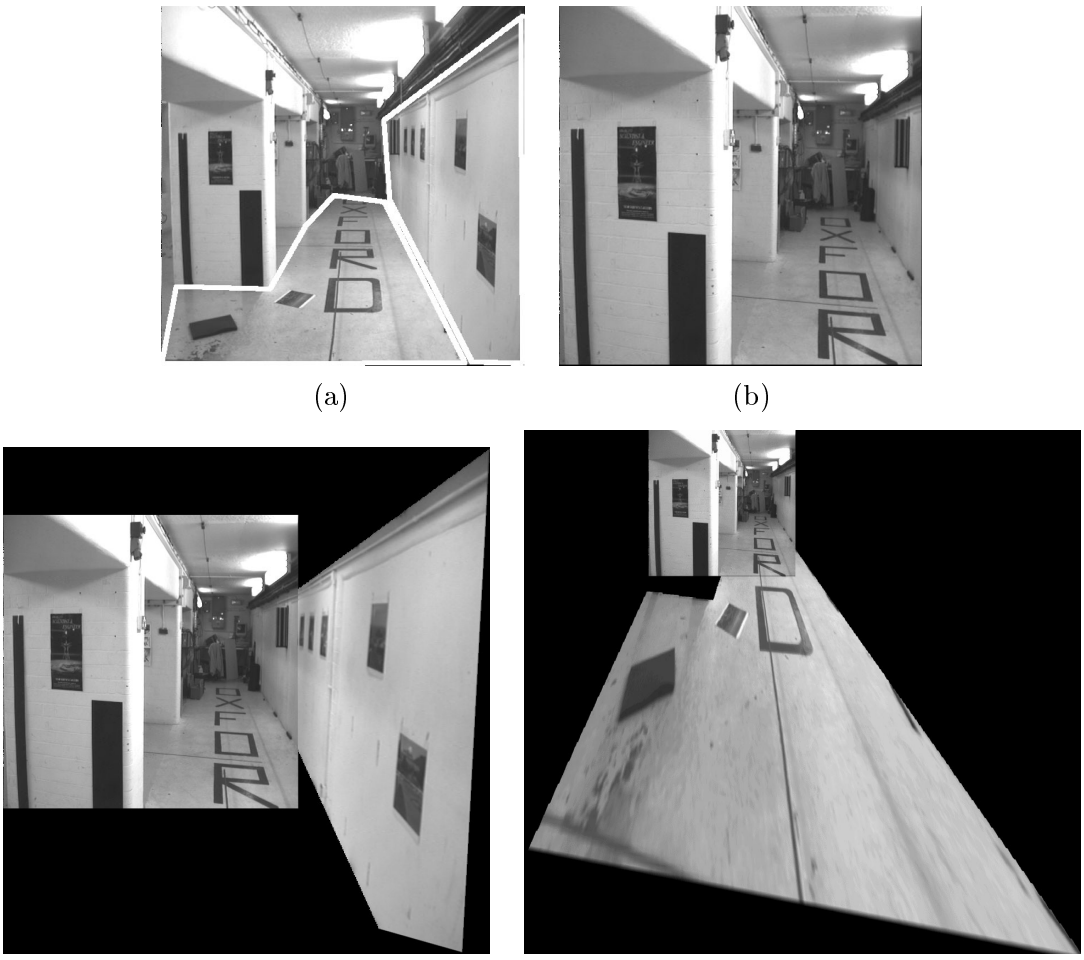
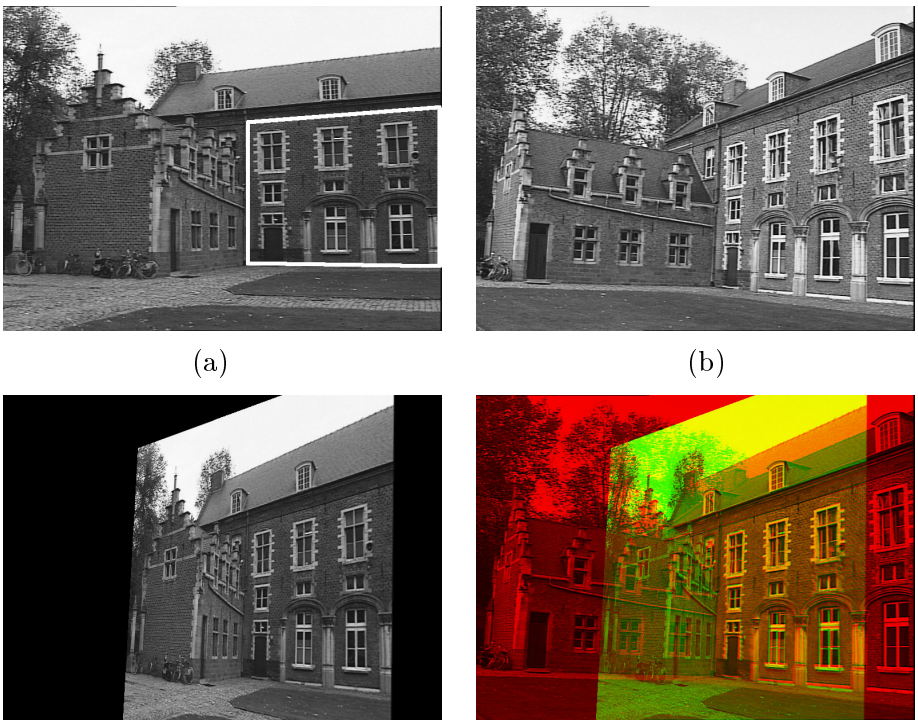


Figure 3: (a), (b) two views of a basement (courtesy of the Oxford Visual Geometry Group). The two polylines in (a) delineate the planar regions tracked over the whole sequence. (c) right wall warped and stitched with (b), (d) floor warped towards and stitched with (b); see text for explanation.

right corridor wall was tracked from frame 0 to frame 8 using the proposed method. Then, by employing the estimated homography, the right wall from frame 0 was warped towards frame 8. Fig. 3(c) shows the warped wall stitched with frame 8. A second plane, namely the one corresponding to the floor, was also tracked between frames 0 and 8. Fig. 3(d) shows the result of warping the floor plane from frame 0 towards frame 8 and stitching them together. As it is clear from the results, the accuracy of the homographies estimated using the proposed method is satisfactory in both cases. Excluding the time required to detect and match corners between successive frames, the average running times for tracking the wall and floor planes in the whole sequence were respectively 63 and 66 ms per frame on an Intel P4@1.8 GHz running Linux.



(a) (b)

(c) (d)

Figure 4: (a), (b) first and last images from the Arenberg castle sequence (courtesy of the University of Leuven VISICS Group), (c) first image warped towards the second using the estimated homography; notice the distortion due to parallax on the left part of the image and (d) warped image in (c) superimposed on (b).

In order to quantitatively evaluate the performance of plane tracking, the floor plane was tracked from frame 0 to frame 10 and then back to frame 0, reversing the order of intermediate frames. This effectively simulates a camera motion that is circular, i.e. ends at the location where it started. Composing the pairwise homographies estimated by the plane tracker, the floor's homography from the first frame through the last and back to itself can be estimated. Ideally, this homography should be equal to the identity matrix. In practice, the deviation in the position of floor points transferred using this homography from their actual locations in the first frame, indicates the accuracy of plane tracking. The root mean square (RMS) error corresponding to the 91 transferred floor points was found to be 19.3 pixels, corresponding to an average RMS error of 0.91 pixels for each of the 21 frames involved in tracking. However, since certain floor points correspond to mismatches or poorly localized corners, a more appropriate error measure is given by the root median square (RMedS) error, which was found to be equal to 8.47 pixels or on average 0.40 pixels per tracked frame.

The second experiment employs another well-known image sequence, the first and last frames of which are shown in Figs. 4(a) and (b). The sequence depicts the Arenberg castle in Belgium and consists of 22 frames acquired with a handheld camera.

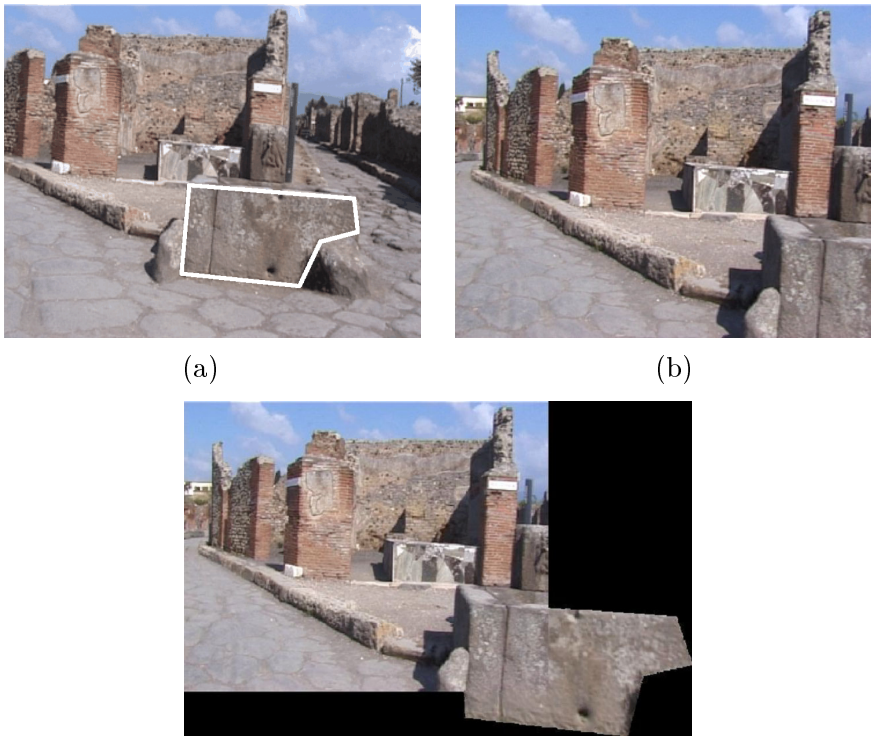
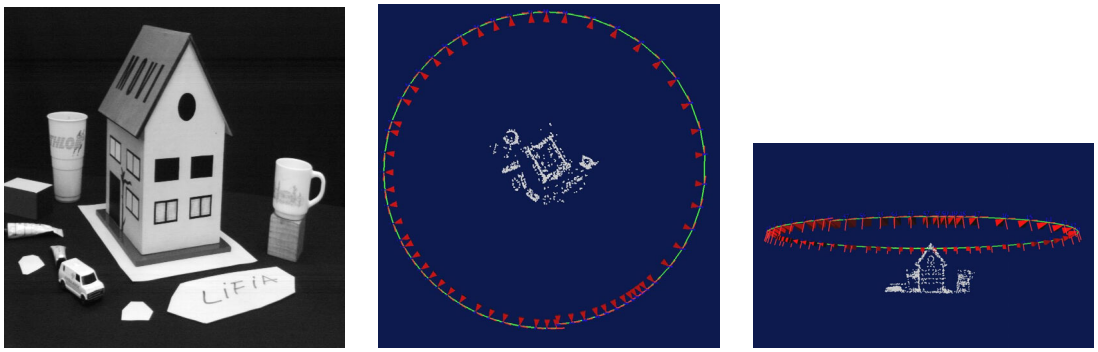


Figure 5: (a), (b) first and last frames from a clip imaging a Pompeian tavern, (c) fountain front face warped and stitched with (b).

Using the proposed method and the image frames between those in Figs. 4(a) and (b), the 3D plane defined by the rightmost wall (see Fig. 4(a)) was tracked throughout the sequence. Fig. 4(c) illustrates the result of warping the first frame towards the last using the estimated homography. To aid in the evaluation of this result, Fig. 4(d) shows it superimposed on Fig. 4(b), using different color channels for each image. As can be clearly seen, image warping according to the estimated homography successfully registers the plane's image in Fig. 4(a) with that in Fig. 4(b). In this case, the average running time for plane tracking was 84 ms per frame. The plane of the right wall was again tracked from the first to the last frame and back (for a total of 43 frames) and the RMS and RMedS errors in this case were 31.7 and 18.3 pixels, amounting to average errors of 0.73 and 0.43 pixels per frame respectively.

The third experiment employs a sequence depicting the remains of a roman tavern (*thermopolium*) in ancient Pompeii. This sequence is quite shaky, due to the fact that it was shot with a camcorder as the operator approached the tavern. It consists of 80 frames, the first and last of which are shown in Fig. 5(a) and (b) respectively. In this experiment, every fourth frame of the original sequence was employed and the plane defined by the front face of the foreground fountain was tracked until the last frame. Fig. 5(c) shows the warped plane from the first plane stitched with the last. Evidently, the homography estimated for the tracked plane is quite accurate. The



(a) (b) (c)

Figure 6: (a) the first frame from the "house" sequence (courtesy of the INRIA MOVI Group), (b) and (c) top and side views of the 3D reconstruction and the camera trajectory. The 3D camera locations are indicated with red pyramids whose apexes are located on the camera optical centers; the green curve connecting the optical centers corresponds to the recovered camera trajectory whereas the white dots illustrate the reconstructed 3D points cloud. Note that the camera trajectory is very close to being a full circle.

average running time for plane tracking was 65 ms per frame. The RMS and RMedS errors computed after tracking the plane of the fountain front face back and forth (39 frames total), were 50.3 and 34.1 pixels respectively, corresponding to average errors of 1.28 and 0.87 pixels per frame respectively.

7.2 Camera Tracking Experiments

Representative results from six of the conducted camera tracking experiments are given in this section. Rigorous performance evaluation of camera tracking for an image sequence is difficult, due to the fact that ground truth for the camera motion is usually unavailable. Therefore, we have chosen to indirectly evaluate camera tracking from the sequences resulting from augmenting the original ones with artificial 3D objects. To achieve this, the estimated camera trajectories were exported to the 3DSMax™graphics package using MaxScript™and then the augmented sequences were generated with the aid of 3DSMax's rendering engine that used the original sequence as a background. The positioning of the artificial graphical objects into the scene was guided by the structure information also provided by the camera tracker. Sample augmented sequences can be found at `http://www.ics.forth.gr/cvrl/demos.html`.

The first experiment was performed on the well known "MOVI house" image sequence, consisting of 119 frames acquired by a fixed camera as a model house on a turntable made a full revolution around its vertical axis. This is equivalent to the camera making a complete circular orbit around the house. The first frame of the sequence is shown in Fig. 6(a), while Figs. 6(b)-(c) illustrate different views of the

VRML 3D model recovered using the proposed method on odd numbered frames. As can be seen from Figs. 6(b), the estimated trajectory is very close to being a circle. The average running time of the proposed tracking method for each image frame was 317 ms on an Intel P4@1.8 GHz running Linux. Most of this time is spent in the bundle adjustment of Eq. (16) and does not include the time required for matching between 200 and 330 points between successive frames. For comparison, note that the time required by existing commercial products such as 2D3's boujou™for batch camera tracking on such sequences is in the order of several minutes for the whole sequence. It should also be mentioned at this point that the performance of bundle adjustment could be considerably improved by employing a sparse variant of the Levenberg-Marquardt minimization algorithm, which will exploit the sparse block structure of the involved normal equations. Additionaly, considering that the task of point matching between frames is characterized by data parallel computations (i.e. correlations), substantial speedups could be gained by adopting a SIMD computation model, employing for example special hardware optimizations such as Pentium's MMX instructions. Figure 7 shows snapshots of the original sequence after augmenting it with the addition of a pine tree. As can be verified from the accompanying videos (`http://www.ics.forth.gr/cvrl/demos.html`) the augmenting object has been convincingly merged with the original sequence.

Figure 7: Snapshots of the augmented "house" sequence corresponding to frames 0, 4, 8, 12, 16 and 20 respectively.

The second experiment employs the "cooks" sequence, the first and last frames of which are shown in Fig. 8 (a) and (b). This sequence is recorded using Digital Air's[4] Timetrack™camera. This camera is made up of an array of lenses that simultaneously photograph a scene from different points of view. If the entire set of images

[4]See http://www.digitalair.com.

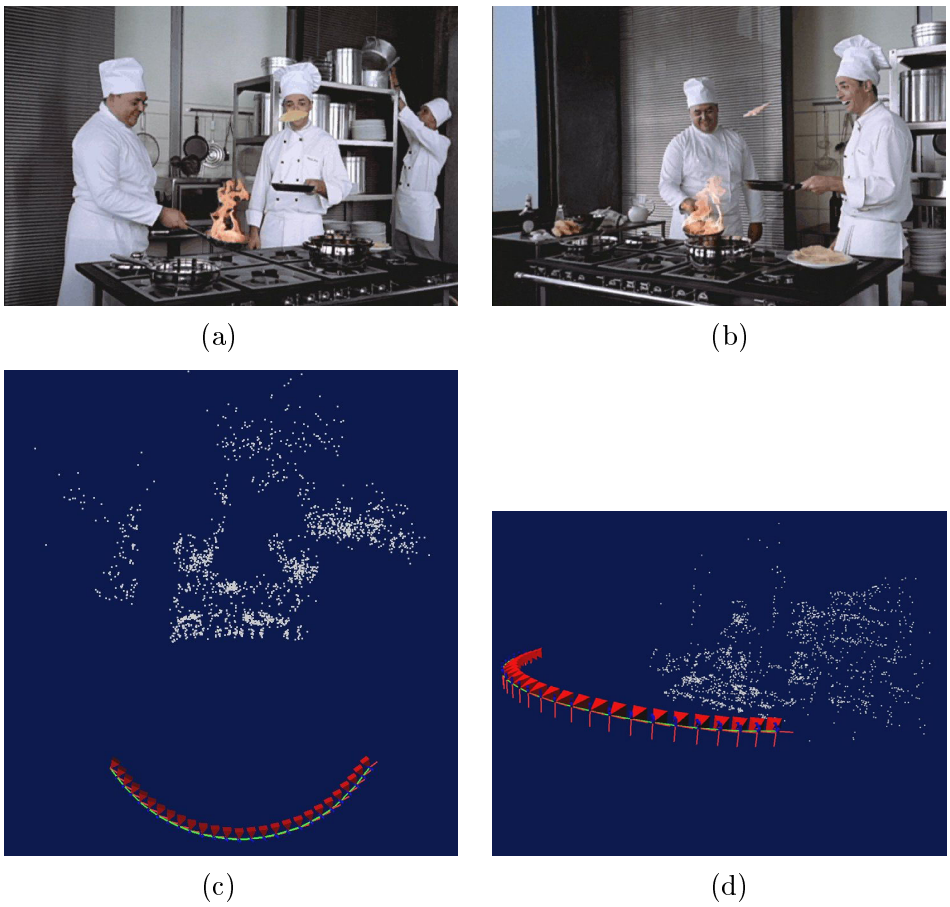


Figure 8: (a), (b) the first and last frames of the "cooks" sequence (courtesy of Dayton Taylor/Digital Air Inc.), (c) and (d) top and side views of the 3D reconstruction and the camera trajectory. Observe that the recovered trajectory is indeed a circular arc.

recorded by all lenses at a specific time instant is played back as a sequence, a viewer has the impression of a camera that appears to move relative to a subject which appears stopped in time. The particular Timetrack camera model used for shooting the "cooks" sequence had 80 lenses configured in a 66 degree arc spanning 3m with a 2.6m radius. In a sense, a Timetrack camera provides sequences that are analogous to those using a fixed camera and a turntable, without, however, the limitations of the latter technique. More details regarding the Timetrack camera can be found at `http://www.virtualcamera.com`. Different views of the VRML 3D model recovered by the proposed method using one every three frames of the sequence are shown in Figs. 8(c) and (d). Clearly, the recovered camera trajectory is a circular arc. The average running time of the proposed tracking method for each image frame was 519 ms, again excluding the time spent for matching about 550 to 550 points between image pairs. Figure 9 illustrates the result of augmenting the original sequence by placing an artificial kettle on top of the stove.

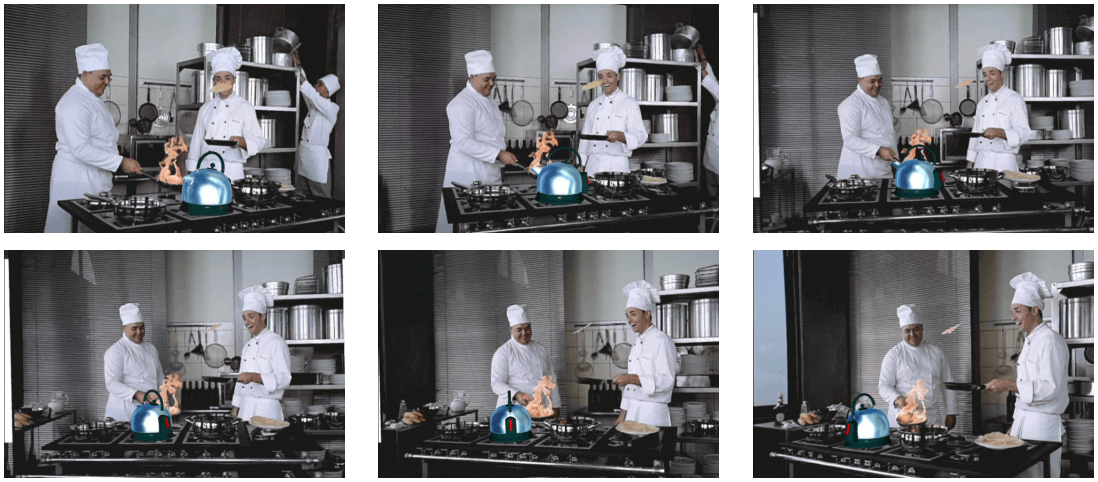


Figure 9: Snapshots of the augmented "cooks" sequence corresponding to frames 0, 5, 10, 15, 20 and 26 respectively.

Figure 10: The "castle" sequence (courtesy of the University of Leuven VISICS Group). Snapshots of the augmented sequence corresponding to frames 0, 4, 8, 12, 16 and 20.

The third experiment refers to the 22 frame Arenberg castle sequence. Fig. 10 shows several frames of the original sequence augmented with a helicopter. A top view of the VRML 3D model that was recovered, showing also the the camera locations and trajectory, is illustrated in Fig. 14(a). On average, camera tracking required 174 ms per frame, employing 250 to 350 corner matches between successive images.

The fourth experiment employs yet another well-known sequence, namely the one

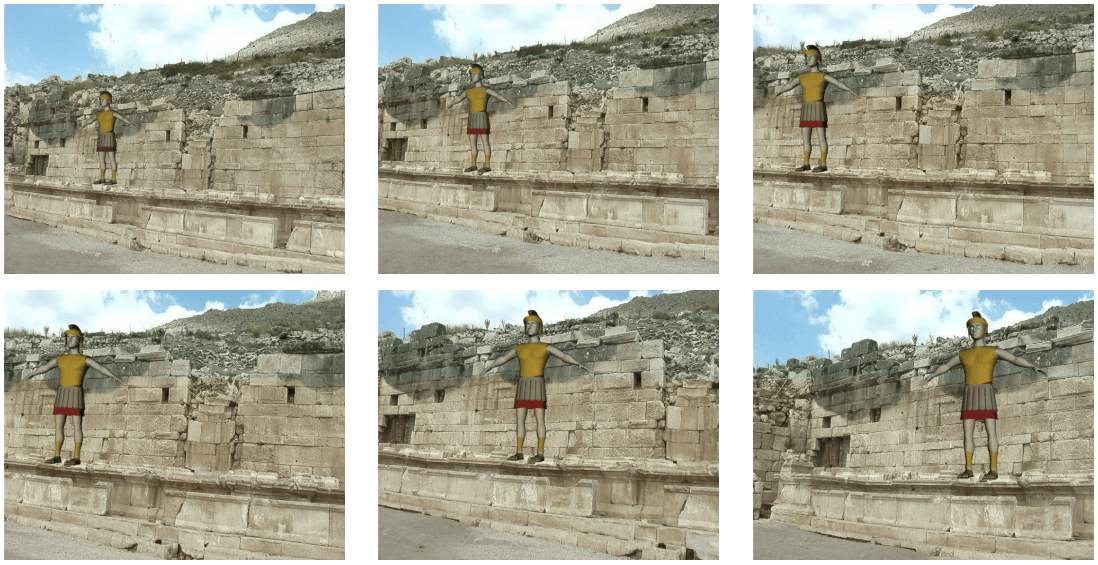


Figure 11: The "sagalassos" sequence (courtesy of the University of Leuven VISICS Group). Snapshots of the augmented sequence corresponding to frames 0, 25, 50, 75, 100 and 125 respectively.

showing the ruins of the Agora in the Sagalassos archaeological site. This sequence has been shot using a camcorder and consists of 126 frames with very small interframe motion. In sequences such as this, a well known issue that can seriously affect the quality of the estimated camera motion is that of selecting appropriate *keyframes*, i.e. frames in the image sequence that are sufficiently apart in time from each other, so that sufficient translational camera motion exists between them. Despite that automatic methods exist for determining keyframes [29], in this experiment the sequence was time subsampled by a factor of five, i.e. every fifth frame was used for computing camera motion. In the case that the camera motion needs to be estimated for frames among the keyframes, this can be achieved using a pose estimation procedure using resectioning based on the 3D structure computed from keyframes [13, 27]. Fig. 11 shows snapshots after augmenting the original sequence with a virtual Roman standing on the wall. A side view of the recovered VRML 3D model is also illustrated in Fig. 14(b). The average running time for each frame in this case was 377 ms, using 500 to 550 matching points between pairs.

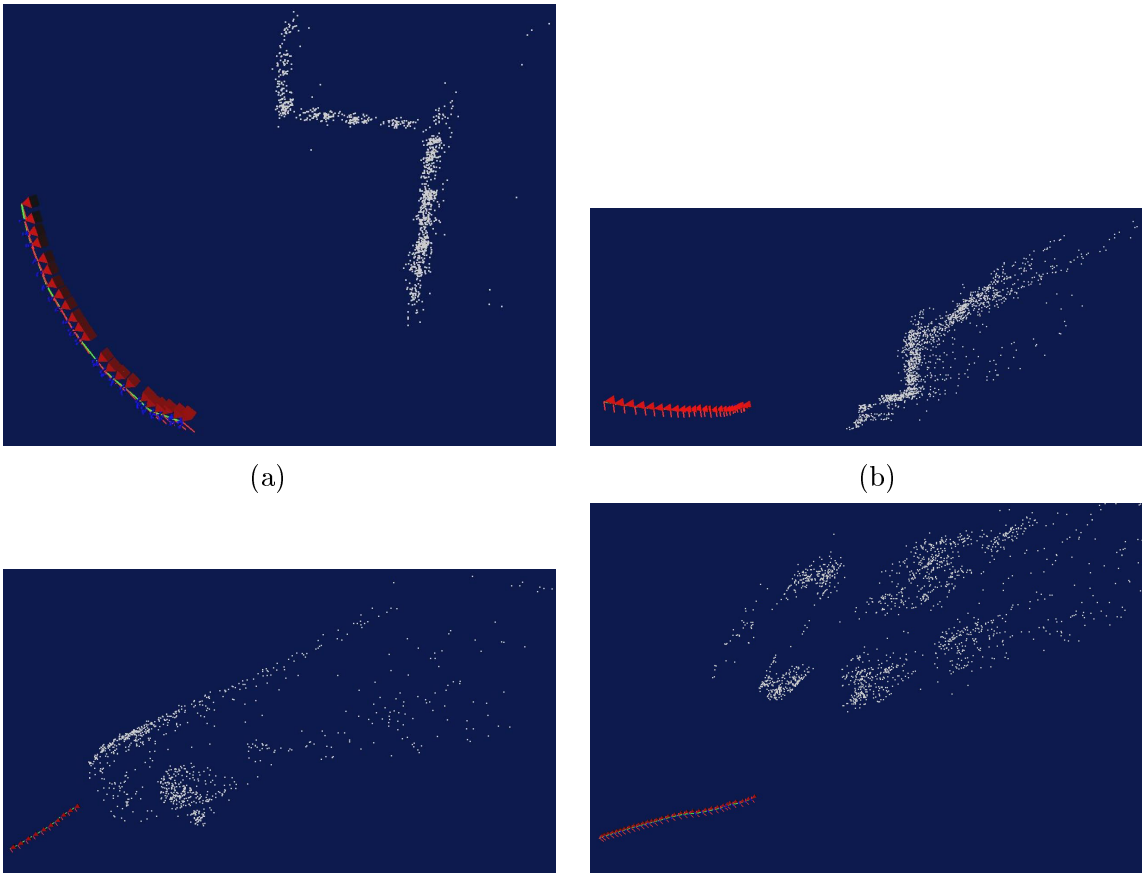
The fifth experiment is based on the "béguinages" sequence, consisting of 90 frames and shot with a camcorder as the operator approached the scene. This forward translational motion results in the baseline for triangulation being small, which in turn makes structure recovery for this sequence difficult. The proposed method was applied to the "béguinages" sequence by time subsampling it by a factor of ten. Fig. 12 shows snapshots after augmenting the original sequence with a statue. A top view of the recovered VRML 3D model is included in Fig. 14(c). On average, 390 ms were required per frame and between 550 to 700 points were matched between frames.

Figure 12: The "béguinages" sequence (courtesy of the University of Leuven VISICS Group). Snapshots of the augmented sequence corresponding to frames 0, 30, 60 and 90 respectively.

Figure 13: The "lab" sequence depicting the robotics lab of ICS/FORTH. Snapshots of the augmented sequence corresponding to frames 5, 10, 15, 20, 25 and 30 respectively.

The sixth and last experiment is based on the "lab" sequence, depicting the robotics lab of ICS/FORTH and consisting of 300 frames shot with a laterally moving camcorder. Time subsampling by a factor of ten was again employed and the tracking

results were used to augment the original sequence with a flower pot placed on the right (darker) robot. Selected snapshots from the augmented sequence are illustrated in Fig. 13. Fig. 14(d) shows a top view of the recovered VRML 3D model. The average running time per frame was 246 ms for 350 to 450 matching points between frames.



(a) (b) (c) (d)

Figure 14: Sample views of different VRML 3D reconstructions corresponding to (a) "castle" top, (b) "sagalassos" side, (c) "béguinages" top and (d) "lab" top.

8 Conclusions

This paper has presented a method for automatic camera tracking across an image sequence acquired without modifying the imaged environment. The method is based on tracking a virtual 3D plane, a task involving the estimation of a quadruple of plane parameters that is achieved using a combination of linear and non-linear optimization techniques operating on sets of corner matches. Knowledge of the homographies induced by the same 3D plane across the whole sequence permits the direct recovery of the camera projection matrices and thus of the Euclidean camera 3D motion, which is later refined through a local bundle adjustment process. The proposed method is

causal and has reasonable computational requirements, permitting an efficient implementation on commodity hardware. Although not statistically optimal in the MLE sense, the results of the proposed method are of very satisfactory accuracy for various types of image sequences.

Future work will investigate alternative ways for making more effective use of the 3D structure information recovered from image triplets. As already mentioned, camera tracking performance would considerably benefit from a sparse version of the bundle adjustment. A limitation of the current camera tracker is that it cannot cope with entirely planar scenes since in this case the fundamental matrix cannot be estimated uniquely [18]. Therefore, it would be desirable to extend the tracker so that it automatically detects such cases and adapts its behavior accordingly. Some work towards this end is reported by Pollefeys et al [31]. Another possible direction concerns the use of subspace constraints on homographies over multiple views. Such constraints have for example been presented in [42] and could lead to considerable accuracy improvements for the estimates of homographies induced by the tracked plane.

References

[1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. SIAM, 3rd edition, 1999.

[2] S. Avidan and A. Shashua. Tensor Embedding of the Fundamental Matrix. In *Proc. of post-ECCV SMILE'98*, volume Springer LNCS 1506, pages 47–62, 1998.

[3] S. Avidan and A. Shashua. Threading Fundamental Matrices. *IEEE Trans. on PAMI*, 23(1):73–77, Jan. 2001.

[4] R. Azuma. Tracking Requirements for Augmented Reality. *CACM*, 36(7):50–51, Jul. 1993.

[5] P.A. Beardsley, A. Zisserman, and D.W. Murray. Sequential Updating of Projective and Affine Structure From Motion. *IJCV*, 23:235–259, 1997.

[6] B. Boufama and R. Mohr. A Stable and Accurate Algorithm for Computing Epipolar Geometry. *IJPRAI*, 12(6):817–840, Sep. 1998.

[7] K. Cornelis, M. Pollefeys, M. Vergauwen, and L. Van Gool. Augmented Reality from Uncalibrated Video Sequences. In *3D Structure from Images - SMILE 2000*, volume LNCS 2018, pages 144–160, 2000.

[8] K. Cornelis, M. Pollefeys, M. Vergauwen, F. Verbiest, and L. Van Gool. Tracking Based Structure and Motion Recovery for Augmented Video Productions. In *Proceedings of VRST'01*, pages 17–24, 2001.

[9] A.J. Davison. Real-Time Simultaneous Localisation and Mapping with a Single Camera. In *Proc. of ICCV'03 (to appear)*, 2003.

[10] J.E. Dennis, D.M. Gay, and R.E. Welsch. An Adaptive Nonlinear Least-Squares Algorithm. *ACM Trans. on Math. Software*, 7(3):348–368, Sep. 1981.

[11] O. Faugeras, Q.-T. Luong, and T. Papadopoulo. *The Geometry of Multiple Images*. MIT Press, 2001.

[12] M.A. Fischler and R.C. Bolles. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *CACM*, 24:381–395, 1981.

[13] A.W. Fitzgibbon and A. Zisserman. Automatic Camera Recovery for Closed or Open Image Sequences. In *Proceedings of ECCV'98*, pages 311–326, 1998.

[14] A. Fusiello. Uncalibrated Euclidean Reconstruction: A Review. *IVC*, 18(6-7):555–563, 2000.

[15] R. Goldman. Intersection of Two Lines in Three-Space. In A.S. Glassner, editor, *Graphics Gems I*, page 304. Academic Press, San Diego, 1990.

[16] G.H. Golub and C.F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 3rd edition, 1996.

[17] R. Hartley and P. Sturm. Triangulation. *CVIU*, 68(2):146–157, 1997.

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

[19] R.I. Hartley. In Defense of the 8-Point Algorithm. *IEEE Trans. on PAMI*, 19(6):580–593, Jun. 1997.

[20] R.I. Hartley. Kruppa's Equations Derived from the Fundamental Matrix. *IEEE Trans. on PAMI*, 19(2):133–135, Feb. 1997.

[21] R.I. Hartley. Lines and Points in Three Views and the Trifocal Tensor. *IJCV*, 22(2):125–140, 1997.

[22] D. Koller, G. Klinker, E. Rose, D. Breen, R. Whitaker, and M. Tuceryan. Real-time Vision-Based Camera Tracking for Augmented Reality Applications. In *Proceedings of VRST'97*, pages 87–94, 1997.

[23] K.N. Kutulakos and J.R. Vallino. Calibration-Free Augmented Reality. *IEEE Trans. on VCG*, 4(1):1–20, Jan. 1998.

[24] M.I.A. Lourakis. Egomotion Estimation Using Quadruples of Collinear Image Points. In *Proc. of ECCV'00*, volume 2, pages 834–848, Jun. 2000.

[25] M.I.A. Lourakis, A.A. Argyros, and S.C. Orphanoudakis. Detecting Planes In An Uncalibrated Image Pair. In *Proc. of BMVC'02*, volume 2, pages 587–596, 2002.

[26] M.I.A. Lourakis and R. Deriche. Camera Self-Calibration Using the Singular Value Decomposition of the Fundamental Matrix. In *Proceedings of ACCV'00*, pages 403–408, 2000. Detailed version in INRIA RR-3748.

[27] C.-P. Lu, G.D. Hager, and E. Mjolsness. Fast and Globally Convergent Pose Estimation from Video Images. *IEEE Trans. on PAMI*, 22(6):610–622, Jun. 2000.

[28] J.J. Moré, B.S. Garbow, and K.E. Hillstrom. User guide for MINPACK-1. Technical Report ANL-80-74, Argonne National Laboratory, Aug. 1980.

[29] D. Nistér. Frame Decimation for Structure and Motion. In *3D Structure from Images - SMILE 2000*, volume LNCS 2018, pages 17–34, 2000.

[30] U. Nowak and L. Weimann. A Family of Newton Codes for Systems of Highly Nonlinear Equations. Technical Report 91-10, Konrad-Zuse-Zentrum fuer Informationstechnik Berlin (ZIB), Dec. 1991. Available at http://www.zib.de/.

[31] M. Pollefeys, F. Verbiest, and L.J. Van Gool. Surviving Dominant Planes in Uncalibrated Structure and Motion Recovery. In *Proc. of ECCV'00*, volume 2, pages 837–851, 2002.

[32] S. Prince, K. Xu, and A. Cheok. Augmented Reality Camera Tracking with Homographies. *IEEE Computer Graphics and Applications*, 22(6):39–45, Nov./Dec. 2002.

[33] P.J. Rousseeuw. Least Median of Squares Regression. *Journal of the American Statistics Association*, 79:871–880, 1984.

[34] A. Shashua and S. Avidan. The Rank-4 Constraint in Multiple View Geometry. In *Proc. of ECCV'96*, volume 2, pages 196–206, 1996.

[35] A. Shashua and N. Navab. Relative Affine Structure: Canonical Model for 3D from 2D Geometry and Applications. *IEEE Trans. on PAMI*, 18(9):873–883, Sep. 1996.

[36] J. Shi and C. Tomasi. Good Features to Track. In *Proceedings of CVPR'94*, pages 593–600, 1994. Free implementation available at `http://robotics.stanford.edu/~birch/klt/`.

[37] G. Simon, A. Fitzgibbon, and A. Zisserman. Markerless Tracking using Planar Structures in the Scene. In *Proc. of Int'l Symposium on Augmented Reality*, 2000.

[38] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle Adjustment – A Modern Synthesis. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, pages 298–372, 1999.

[39] T. Viéville, O. Faugeras, and Q.T. Luong. Motion of Points and Lines in the Uncalibrated Case. *IJCV*, 17(1):7–41, Jan. 1996.

[40] G. Welch and E. Foxlin. Motion Tracking: No Silver Bullet, but a Respectable Arsenal. *IEEE Computer Graphics and Applications*, 22(6):24–38, Nov./Dec. 2002.

[41] Y. Xirouhakis, A. Drosopoulos, and A. Delopoulos. Efficient Optical Camera Tracking in Virtual Sets. *IEEE Trans. on IP*, 10(4):609–622, Apr. 2001.

[42] L. Zelnik-Manor and M. Irani. Multiview Constraints on Homographies. *IEEE Trans. on PAMI*, 24(2):214–223, Feb. 2002.

[43] Z. Zhang, R. Deriche, O. Faugeras, and Q.-T. Luong. A Robust Technique for Matching Two Uncalibrated Images Through the Recovery of the Unknown Epipolar Geometry. *AI Journal*, 78:87–119, 1995. Detailed version in INRIA RR-2273.

[44] Z. Zhang and Y. Shan. Incremental Motion Estimation through Local Bundle Adjustment. Technical Report MSR-TR-01-54, Microsoft Research, May 2001.