Space-Time-Scale Registration of Dynamic Scene Reconstructions

Kemal E. Ozden[1], Kurt Cornelis[1], and Luc Van Gool[1,2]

[1] ESAT/PSI, Computer Vision Lab., K.U. Leuven, Belgium
{Egemen.Ozden, Kurt.Cornelis, Luc.Vangool}@esat.kuleuven.be
http://www.esat.kuleuven.be/psi/visics/
[2] BIWI, ETH, Zurich, Switzerland
vangool@vision.ee.ethz.ch
http://www.vision.ee.ethz.ch/

Abstract. The paper presents a method for multi-dimensional registration of two video streams. The sequences are captured by two hand-held cameras moving independently with respect to each other, both observing one object rigidly moving apart from the background. The method is based on uncalibrated Structure-from-Motion (SfM) to extract 3D models for the foreground object and the background, as well as for their relative motion. It fixes the relative scales between the scene parts within and between the videos. It also provides the registration between all partial 3D models, and the temporal synchronization between the videos. The crux is that not a single point on the foreground or background needs to be in common between both video streams. Extensions to more than two cameras and multiple foreground objects are possible.

1 Introduction

Structure-from-Motion (SfM) techniques have made impressive progress in the last two decades [11]. They can compute 3D models and camera motion out of image sequences coming from a single moving camera. However, most of these methods can only cope with static scenes and this limitation is one of the biggest challenges for applying SfM in real life, with its many dynamic elements. Nevertheless, some SfM techniques for dynamic scenes are around [2, 3, 6, 9] and the subject is getting more and more attention.

A subtle issue with uncalibrated SfM is that scenes can only be reconstructed up to an unknown scale between the independently moving objects. This does not only have an impact on the relative sizes of the objects, but actually also on the shape of their trajectories with respect to the background. Typical approaches to solve this problem, explicitly or implicitly, make assumptions about the object motion, e.g. [13, 15, 1]. Here we propose an alternative solution. It requires multiple cameras but it works with generic object motions and without any corresponding features between the video streams. Not only does it allow to determine the relative scales, the videos are also synchronized and the resulting 3D extracted from each stream is spatially registered.

A. Leonardis, H. Bischof, and A. Pinz (Eds.): ECCV 2006, Part IV, LNCS 3954, pp. 173–185, 2006.

Obviously, this is far from the first work using multiple cameras in a SfM context. Here we only mention the most related work. In [7], the relative displacement between the cameras of a stereo rig is computed using several motions of the rig. In [24], a self-calibration method for a moving rig is presented where the rig itself does not need to be rigid, however some constraints on the camera orientations are still required. In [25], a non-rigid scene is reconstructed with static orthographic stereo cameras. All of the above work has the common advantage of being correspondence-free, i.e. there are no stereo correspondences between the cameras.

Here, we use two (hand-held) cameras moving completely independently with respect to each other, still not assuming knowledge of any correspondences. The price to pay for this freedom is that at least one moving and rigid object ought to be observed by both cameras as the information from the background itself is not enough to solve the problem. The fact that the object should move the same way with respect to the background in both sequences is the trivial but key observation exploited by the algorithm. It can thereby fix the scales of the object and the background, bring their partial 3D reconstructions into registration, and even synchronize - i.e. temporally align - the two videos.

Video synchronization in combination with (partial) camera calibration has also been studied by several researchers, and the exploitation of moving objects in particular as well. For example, Caspi et al. [4] use point trajectories to find a suitable transformation to spatio-temporally align image sequences. Sinha and Pollefeys [18] also combine camera calibration with synchronization, but they use fixed cameras. These papers still require the visibility of the same points at the same time. Caspi et al. [5] could lift this restriction by using moving cameras with the same optical center. The views were aligned in space (through a homography) and in time.

2 Problem

In this paper we consider two hand-held cameras which move independently with respect to each other. Furthermore, we consider a single object moving independently against a static background. The cameras may view the moving object from totally different directions, so it is well possible that there are no common feature points between the video sequences both for the background and the foreground. However it is required that the cameras see the same rigidly moving object, though possibly different parts thereof.

In order to reconstruct such a scene the first step is to segment the foreground object from the background for which several solutions are available (e.g. [6, 12, 16, 20, 21]), though currently we are doing it manually. This then allows a typical uncalibrated SfM algorithm [11] to be applied to the object and background segments in each of the videos. This results in four 3D point clouds and four sets of camera matrices (trajectories relative to the capturing camera). These cannot be readily integrated however, not even for the object and background data derived from the same camera. Several parameters need to be determined first.

Three of those parameters come from the fact that uncalibrated SfM is defined only up to a scale factor, i.e. the scene reconstructed from a single moving camera has a scale ambiguity for every independently moving object. This scale ambiguity is not a serious problem in the case of an entirely rigid scene, since then everything has correct relative scales and only the absolute scale is missing. However, for independently moving objects the correct relative scale with respect to the background is also unknown and may result in unrealistic reconstructions. This is a problem that cannot be resolved through hard geometric constraints among the frames of a single video. As a matter of fact, there is an entire one-dimensional family of relative scales + corresponding trajectories for the object with respect to the background, which are all *fully* compatible with all image data within the sequence [13, 15]. Many special effects in movies are based on this fact. When seeing a car driving on a road it could actually also be a miniature car close to the camera with a motion quite different from that of an actual car.

In the case of monocular input this ambiguity can be lifted by assuming that the object is following a plausible motion constraint [13, 15]. Here it is shown that the use of two cameras allows the objects to move arbitrarily. We will have to determine the 3D similarity transformation between the reconstructed backgrounds in the two videos, as well as the relative scales of the foreground in each video with respect to its background. This will require the synchronization of the two videos.

Suppose we fix the scale for the background in one video. We will have to determine the correct relative scale of the foreground in the same video, as well as the scales which then have to be applied to the foreground and the background in the other video. Therefore, two unknown relative scales and one inter-camera scale ambiguity count for the three scale parameters we have to solve for. Obviously, there is also an Euclidean transformation (defined by six parameters) between the reconstructions coming from the different cameras. So in total we have to solve for nine parameters. The wrong choice for these parameters will result in a different object motion for each different video stream which actually must be identical. Our goal is to search for those parameters which will make the object motions for both sequences identical or if stated in a different way, which will make the overall object motion the most rigid since if the objects motions as seen from both cameras are identical, the related foreground point clouds must move rigidly.

3 Notation and Basic Formulation

The two cameras are arbitrarily labeled as the first and the second camera. Applying SfM to the first sequence yields the following object transformation matrices:

$$\mathbf{M}^i = \mathbf{T}^i\mathbf{M} = \begin{bmatrix} \mathbf{R}_o^i & \mathbf{t}_o^i \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{M} \tag{1}$$

which describe the motion of a 3D homogeneous point $\mathbf{M}$, which is a fixed point in the object coordinate system. $\mathbf{M}^i$ is the position of point $\mathbf{M}$ at frame index

i in the world coordinate system. Typically the camera pose of the first frame is chosen as the world coordinate system, which is also the case here. $\mathbf{R}_o^i$ is a 3×3 rotation matrix and $\mathbf{t}_o^i$ is a 3×1 translation vector. It is very important to note that $\mathbf{T}^i$ is computed by multiplying the inverse of the related background motion matrix with the relative motion matrix of the foreground, both of which are direct outputs of the SfM algorithm (see [13, 15]).

However, due to unknown relative scales, there exists a one-parameter family of solutions [13, 15] for these object transformation matrices because:

$$\mathbf{t}_o^i = m\left(\mathbf{t}_{of}^i - \mathbf{t}_c^i\right) + \mathbf{t}_c^i \tag{2}$$

where $\mathbf{t}_{of}^i$ is a particular solution for the object translation and $\mathbf{t}_c^i$ is the position of the optical center in the world coordinate system, which are both returned by SfM. The one-parameter family is described by scale factor m. To give an intuitive explanation we can interpret Eq (2) as a set of 3D lines which pass through the optical center at each frame index i. Consequently, every point on these lines project to the same location in the same image given any value m.

The world coordinate system will be different for both image sequences since the camera poses for the first frame will differ. However, a similarity transformation exists which aligns the world coordinate systems of both sequences:

$$\mathbf{M}^i = \mathbf{X}\mathbf{M}'^i \text{ with } \mathbf{X} = \begin{bmatrix} k\mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \tag{3}$$

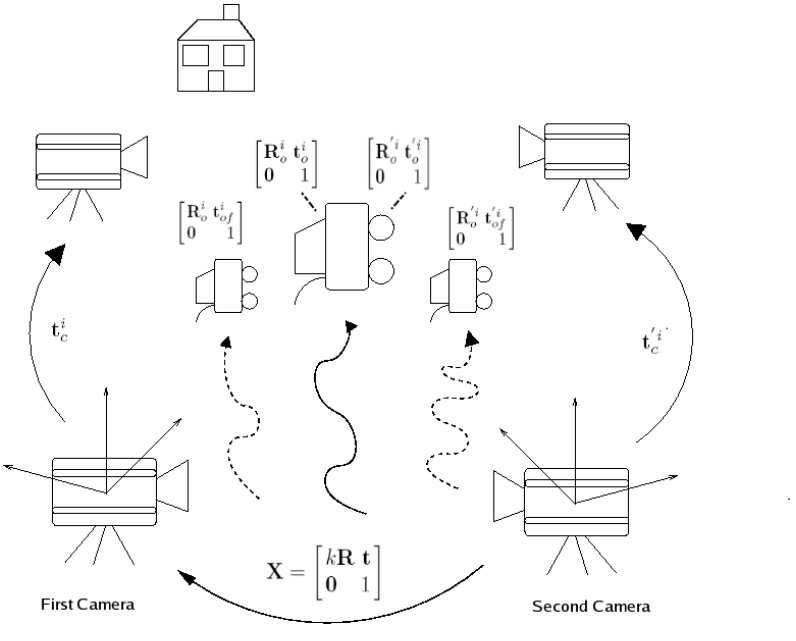

Fig. 1. A depiction of the transformations. Due to the relative scale ambiguity, different cameras see arbitrarily scaled objects and ambiguous object translations.

where $\mathbf{M}^{'i}$ is a point in the second image sequence which corresponds to point $\mathbf{M}^i$ in the first sequence. Here, it is important to stress that $\mathbf{X}$ is a transformation between the 3D reconstruction reference frames, and not between the moving cameras. Therefore, $\mathbf{X}$ is constant throughout the sequence but the transformation between the camera local coordinate systems is allowed to change freely.

The aforementioned transformations are all illustrated in Fig. 1 in which the superscript $'$ accompanies the symbols related to the second sequence.

4 Solution

Combining Eq. (1) and Eq. (3) for both sequences, we arrive at:

$$\begin{bmatrix} \mathbf{R}_o^i & \mathbf{t}_o^i \\ \mathbf{0} & 1 \end{bmatrix} = \mathbf{X} \begin{bmatrix} \mathbf{R}_o^{'i} & \mathbf{t}_o^{'i} \\ \mathbf{0} & 1 \end{bmatrix} \mathbf{X}^{-1} \tag{4}$$

which is a different form of the famous hand-eye calibration problem. A common technique in hand-eye calibration is to solve for the rotation part first:

$$\mathbf{R}_o = \mathbf{R}\mathbf{R}_o^{'}\mathbf{R}^T \tag{5}$$

and subsequently solve for the translation part:

$$\mathbf{t}_o = -\mathbf{R}_o\mathbf{t} + k\mathbf{R}\mathbf{t}_o^{'} + \mathbf{t} \tag{6}$$

where the frame indices have been dropped for ease of notation. We will follow the same approach. As to the solution of Eq. (5) it is known that every rotation matrix has an axis which remains unaffected under that particular rotation. Let $\mathbf{n}$ be that axis for $\mathbf{R}_o$ and $\mathbf{n}^{'}$ be that axis for $\mathbf{R}_o^{'}$. It can be derived and intuitively understood that after alignment of the world coordinate systems of both sequences the axes of the related rotations must be identical:

$$\mathbf{n} = \mathbf{R}\mathbf{n}^{'} \tag{7}$$

Many solutions have been proposed to solve the above equation, e.g. [8, 17, 19]. We chose the unit quaternion approach by Faugeras and Hebert [8] which is also detailed in Horaud and Dornaika [10]. Unit quaternions are 4-parameter imaginary representations of rotations in 3D. Since their length is one, their degree of freedom is three as expected from a rotation representation. The operation of a rotation matrix on other rotation matrices and 3D vectors can be easily represented as quaternion multiplications. The rotation of a 3D vector $\mathbf{n}$ with the rotation matrix $\mathbf{R}$ can be written as:

$$\mathbf{q} * \mathbf{n}_q^{'} * \overline{\mathbf{q}} = \mathbf{R}\mathbf{n}^{'} \tag{8}$$

where $\mathbf{q}$ is the quaternion representation of $\mathbf{R}$, $\overline{\mathbf{q}}$ is the conjugate of $\mathbf{q}$, $\mathbf{n}_q^{'}$ is the quaternion representation of the vector $\mathbf{n}^{'}$ and $*$ is quaternion multiplication.

As we have many frames and noisy data, it is practically impossible to find a perfect solution to Eq. (7) so we should minimize an error criterion. The one used here is the total 3D squared distance between the corresponding rotation axes after the application of rotation $\mathbf{R}$ which can be written as:

$$E_1 = \sum_{i=1}^{\#frames} \left| \mathbf{n}_q^i - \mathbf{q} * \mathbf{n}_q^{'i} * \overline{\mathbf{q}} \right|^2 \tag{9}$$

Since quaternions are of unit length, the statement inside the summation can be written as :

$$\left| \mathbf{n}_q^i - \mathbf{q} * \mathbf{n}_q^{'i} * \overline{\mathbf{q}} \right|^2 = \left| \mathbf{n}_q^i - \mathbf{q} * \mathbf{n}_q^{'i} * \overline{\mathbf{q}} \right|^2 |\mathbf{q}|^2 \tag{10}$$

$$= \left| \mathbf{n}_q^i * \mathbf{q} - \mathbf{q} * \mathbf{n}_q^{'i} \right|^2 \tag{11}$$

$$= \mathbf{q}^T \mathbf{A}^i \mathbf{q} \tag{12}$$

where $\mathbf{A}^i$ is a 4×4 matrix whose elements are computed from $\mathbf{n}^i$ and $\mathbf{n}^{'i}$ [8, 10]. In the end we have a minimization of the form :

$$E_1 = \mathbf{q}^T \mathbf{A} \mathbf{q} \tag{13}$$

where $\mathbf{A} = \sum_{i=1}^{\#frames} \mathbf{A}^i$. When we try to minimize Eq. (13) with the constraint that quaternions are of unit length, the quaternion turns out to be the eigenvector of A corresponding to the minimum eigen-value.

Now that the rotation parameters are computed, we can proceed to solve Eq. (6) for the translation and the scale parameters. Inserting Eq.(2) for both sequences into Eq.(6) results in:

$$\mathbf{t}_c = (\mathbf{t}_c - \mathbf{t}_{of})\, m + (\mathbf{I} - \mathbf{R}_o)\, \mathbf{t} + \left(\mathbf{R}\mathbf{t}_c^{'} \right) k + \left(\mathbf{R}\mathbf{t}_{of}^{'} - \mathbf{R}\mathbf{t}_c^{'} \right) k m^{'} \tag{14}$$

which is a linear equation in terms of $\mathbf{t}$, k,m and $km^{'}$. In a typical scenario, we would have redundant equations so a simple linear least squares scheme is applicable here.

Since the rotation is estimated separately from other parameters, it is desirable to minimize an error criterion which handles all parameters simultaneously. We must also note that our final aim is to come up with a solution where the foreground objects as reconstructed from both sequences move as rigidly as possible with respect to each other. However, a minimization in transformation space does not necessarily result in the best rigid motion for the foreground objects since it minimizes an algebraic error rather than a geometric one. A good way to express rigidity is by stating that distances between points remain the same. Therefore, the current solution is used as an initialization of a non-linear iterative refinement technique like Levenberg-Marquardt with the following error criterion:

$$E_2 = \sum_{k=1}^{6} F\left(\mathbf{p}_k\right) \tag{15}$$

$$F(\mathbf{p}) = \sum_{i=1}^{\#frames} \left| \mathbf{T}^i \mathbf{p} - \mathbf{X}\mathbf{T}'^i \mathbf{X}^{-1} \mathbf{p} \right|^2 \quad (16)$$

where $\mathbf{T}^i$ and $\mathbf{T}'^i$ are euclidean transformation matrices describing the object motion in the i^{th} frame for the first and the second camera. F is an error measure between the paths of a 3D point when the motion matrices computed for the first and the second image sequence are applied separately and $\mathbf{p}_k$ is a specific point in the object coordinate system of the first camera. As to the choice for $\mathbf{p}_k$, we followed some guidelines. First of all, a 3D Euclidean transformation is defined by the motion of at least 3 non-linear points, so the number of points must be more or equal to three and they must be non-linear. Secondly as the SfM measurements are valid only around the reconstructed object, the points may not be far away from the 3D point cloud of the object but also may not be very close to each other in order not to degenerate to a single point. So in order to satisfy all these criteria, we decided to take the PCA transform of the point cloud and choose the end points of the computed axes which result in six points in total.

So far we implicitly assumed that both video streams are synchronized in time. However, with hand-held cameras this is usually not the case. To overcome this difficulty, researchers proposed different techniques, e.g. [4, 18, 25], and the problem of time synchronization becomes more and more popular.

In our case, Eq.(5) and Eq.(6) give a geometric relationship between two frames and we would expect that these equations do not hold when two frames do not correspond to each other in time, just like any other geometric relationship like the fundamental matrix etc. So the technique we propose for time synchronization is to shift the video sequences with respect to each other within a reasonable range and compute the residual of the solution to Eq. (15). We expect that the correct time shift corresponds to the lowest residual. After a rough discrete shift value is found, the residual graph can be interpolated to search for the solution at sub-frame accuracy. To achieve this, a sub-frame time shift parameter λ is incorporated into Eq. (16) which results in:

$$F_{sub}(\mathbf{p}) = \sum_{i=1}^{\#frames} \left| \lambda \mathbf{T}^{i+shift} \mathbf{p} + (1-\lambda)\, \mathbf{T}^{i+1+shift} \mathbf{p} - \mathbf{X}\mathbf{T}'^i \mathbf{X}^{-1} \mathbf{p} \right|^2 \quad (17)$$

where λ is restricted to be between 0 and 1 and $shift$ is the rough discrete time-shift value. This equation basically introduces linear interpolation to the paths defined by the principal points.

5 Experiments

We conducted two different experiments to demonstrate the effectiveness of the proposed technique. In the first experiment, a person is pushing a dolly on which a pile of boxes are placed. The person and the background are recorded by two freely moving hand-held cameras whose viewing angles are quite different so it

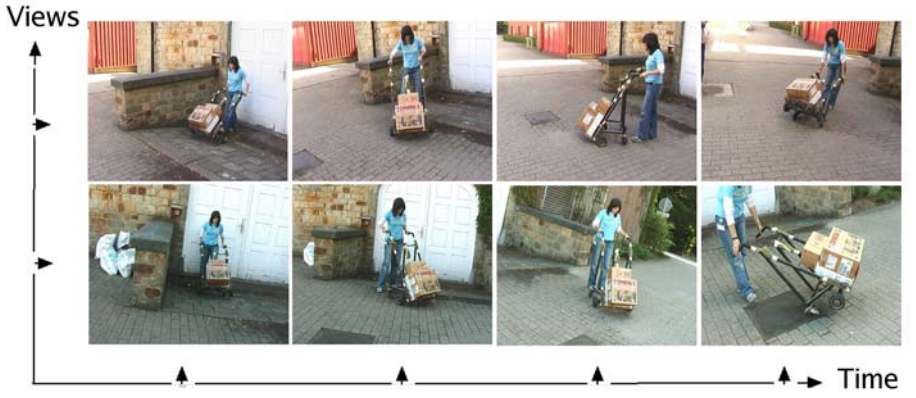

Fig. 2. Samples from the original image sequence. Each row belongs to a separate camera, each column is related to a different time instant.

is hard to find common photometric features between the two image sequences. Some example frames from the first and the second camera can be seen in Fig. 2. The careful reader might notice that although the set-up is very wide-baseline, there are still some common feature points. However those points will only be used for *verification* of the computed registration parameters. Although our algorithm does not require their existence, it helps us to demonstrate that the algorithm works well.

The sequence is 180 frames long (image size is 720×576) and the dolly passes through different poses. Both sequences are segmented beforehand as foreground and background sequences, are reconstructed separately using SfM and subsequently fed to our algorithm. The time-shift between the sequences is approximately known to be 5 frames which is quite close to 5.13, the value computed by the algorithm.

Fig. 3 shows the background reconstructions from two different cameras which are registered together. It can be clearly seen that the corresponding ground planes and walls are aligned quite well. To give a different view of the result we manually chose three common features from the first sequence, computed their 3D positions and projected them in the second sequence using the registration parameters we computed. In Fig. 4, the black circles denote the actual position of the feature points and the white squares nearby depict the the reprojection of the corresponding 3D points of the second sequence after transfer to the second sequence. The average pixel error is 6 pixels. but we must note that this error value highly depends on the image we use to reproject into. If we have a good registration, we also expect the foreground motions to be the same. So in order to test the latter, we chose a 3D point from the foreground object of the first sequence and computed its 3D path according to the motion parameters from the first sequence and also according to the *registered* motion parameters computed from the second sequence. Fig. 5 demonstrates such a registration for an arbitrary 3D point. The circles and the triangles correspond to point paths computed with

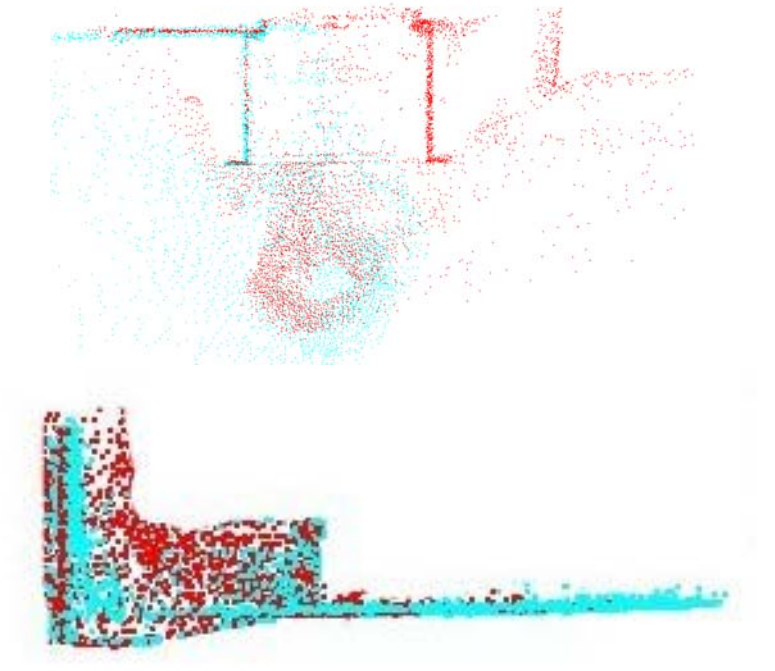

Fig. 3. A top and a side view of the reconstruction. Notice how well the walls and the ground planes are registered.

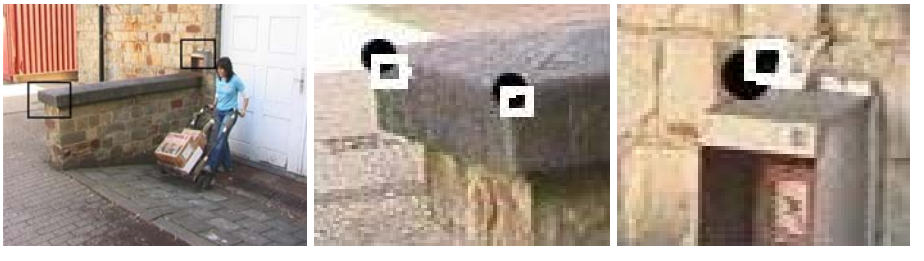

Fig. 4. Manually tracked features from one image sequence are projected into the other image sequence. In the region of interest, the original features are depicted by black circles, whereas their reprojections are depicted by white squares.

object motions from the two different video streams. The error measure, which is the average distance between the corresponding point positions divided by the path length, is 0.8% which is quite low as expected.

As for the second experiment, we recorded a 330 frames-long (image size is 720×576) sequence where a person himself is carrying boxes on a staircase and is moving arbitrarily but rigidly. Fig. 6 show some example frames. The cameras are also moving freely and view the scene from quite different angles. We computed the reconstructions and registration parameters in the same way as

Fig. 5. Different views on the resulting path of an arbitrary 3D point on the foreground in the first image sequence when displaced by object transformations coming from the first sequence (circles) and the second sequence (triangles) after registration

Fig. 6. Samples from another image sequence. Each row belongs to a separate camera, each column is related to a different time instant.

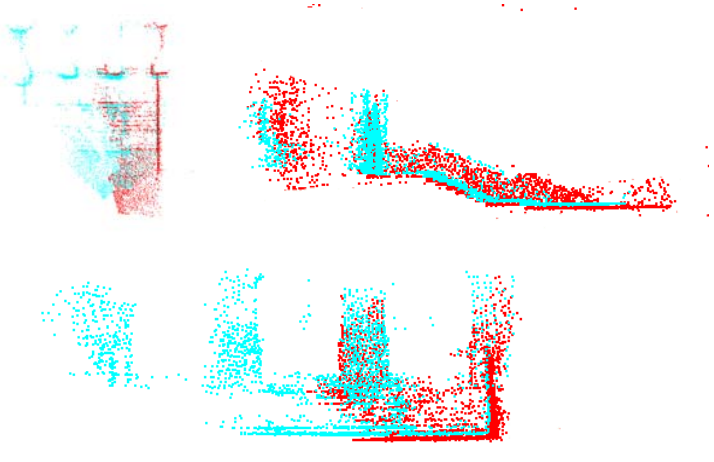

Fig. 7. A top, side and front view of the registered reconstructions. Notice the good registration of the stairs, the ground plane, the right wall and the pillars.

the previous experiment. Fig. 7 shows the registered background reconstructions. As can be seen, the ground plane, the stairs, the walls and the pillars are very well registered. Fig. 8 demonstrates the reprojection of some common feature points having an average pixel error of 15, but as mentioned earlier this error

Fig. 8. Just like the previous experiment, manually selected features from one image sequence are projected into the other image sequence

Fig. 9. Different views on the resulting path of an arbitrary 3D point on the foreground in the first image sequence when displaced by object transformations coming from the first sequence (circles) and the second sequence (triangles) after registration

value highly depends on the image chosen for reprojection. Fig. 9 demonstrates the 3D point paths computed from the object motions from two different videosets. The error measure, which is the average distance between the corresponding point positions divided by the path length, is 0.4% which is quite low as expected.

6 Conclusion and Discussion

In this paper, we presented a novel technique which finds the space-time-scale parameters between two reconstructions of a scene coming from two independently moving hand-held cameras. Rather than matching photometric features like points, lines etc., it tries to find a consistent transformation which results in the most similar motion for the independently moving foreground object. As a consequence, the cameras are free to observe the scene from totally different angles with the restriction that at least one rigidly moving foreground object is required. Although we presented our initial results here, there are still open questions and possible improvements. As an initial improvement, the basic approach can easily be extended to scenarios which contain more than two cameras and multiple rigidly moving foreground objects. Although we have not used common feature points we can find such features much more easily after an initial registration and use them as well in a global optimization. As an interesting fact, such features need not be simultaneously visible in both cameras which is a necessity in many multicamera systems. Another interesting remark would be how to determine which part of the segmented scene correspond to the background and which to the foreground. Upto now, we assumed this to be known apriori. This, however, can be achieved automatically in several ways, e.g. with a typical

assumption that the biggest object is the background, or with a more elaborate technique [14] if the foreground motion complies to a certain constraint. However, our framework itself is also capable of identifying the corresponding segmentation parts between the two sequences, since a wrong choice would result in a higher error value after the final minimization.

Unfortunately, the proposed technique can not handle certain foreground object motions which are degenerate cases. As to the solution of the Eq. (5), it is known that the existence of at least two rotation axes is necessary and as the number of axes increase the solution becomes more stable. We also noticed a certain degeneracy when the foreground object motion is a pure rotation around a single point. However, we expect that the existence of multiple moving objects would significantly decrease such problems.

Acknowledgments

The authors gratefully acknowledge support by the KULeuven Research Council GOA project 'MARVEL' and the Flemish Fund for Scientific Research FWO.

References

1. Avidan, S., Shashua, A. Trajectory triangulation: 3D reconstruction of moving points from a monocular image sequence. PAMI Vol. 22(4) pages 348-357 (2000)
2. Brand, M. Morphable 3d models from video. CVPR, Vol.2, pages 456-463, (2001)
3. Bregler, C., Hertzmann, A., Biermann, H. Recovering non-rigid 3d shape from image streams. CVPR, pages 2690-2696, (2000)
4. Caspi, Y., Irani, M. Spatio-temporal Alignment of Sequences. PAMI, Vol 24(11), pages 1409- 1424, (2002)
5. Caspi, Y., Irani, M. Alignment of Non-Overlapping Sequences. ICCV, pages 76-83, (2001)
6. Costeira, J., Kanade, T. A Multi-Body Factorization Method for Motion Analysis. ICCV, pages 1071-1076, (1995)
7. Dornaika, F., Chung, R. Self-calibration of a stereo rig without stereo correspondence. Vision Interface, pages 264-271, (1999)
8. Faugeras, O.D.,Hebert,M. The representation, recognition and locating of 3D objects. International Journal of Robotics Research, Vol 5(3), pages 27-52 (1986)
9. Fitzgibbon, A.W., Zisserman, A. Multibody structure and motion: 3-D reconstruction of independently moving objects. ECCV, pages 891-906, (2000)
10. Horaud, R., Dornaika, F. Hand-Eye Calibration. International Journal of Robotics Research, Vol 14(3), pages 195-210, (1995)
11. Hartley, R., Zisserman A. *Multiple View Geometry.* Cambridge University Press (2000)
12. Machline, M., Zelnik-Manor, L., Irani, M. Multi-body segmentation: Revisiting motion consistency. ECCV Workshop on Vision and Modeling of Dynamic Scenes, Copenhagen (2002)
13. Ozden, K. E., Cornelis, K., Van Eycken, L., Van Gool ,L. Reconstructing 3D independent motions using non-accidentalness. CVPR, Vol. 1, pages 819-825, (2004)
14. Ozden, K. E., Van Gool ,L. Background recognition in dynamic scenes with motion constraints. CVPR, Vol. 1, pages 250-255 (2005)

15. Ozden, K. E., Cornelis, K., Van Gool ,L. Reconstructing 3D trajectories of independently moving objects using generic constraints. CVIU, pages 453-471, December 2004.
16. J. Shi, J. Malik. Motion segmentation and tracking using normalized cuts. ICCV, 1154-1160, (1998)
17. Shiu,Y.C.,Ahmad,S. Calibration of wrist mounted robotic sensors by solving homogenous transform equations of the form AX=XB. IEEE J. Robot. Automation 5(1), pages 16-19 (1989)
18. Sinha, S.N., Pollefeys, M. Synchronization and Calibration of Camera Networks from Silhouettes. ICPR, Vol. 1, pages 116-119 (2004)
19. Tsai, R.Y. A versatile camera calibration technique for high accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. IEEE J. Robot. Automotion RA-3(4), pages 323-344 (1987)
20. Torr, P. H. S. Geometric motion segmentation and model selection. Phil. Trans. Royal Society of London A, 356(1740):1321-1340 (1998)
21. Vidal, R., Soatto, S., Ma, Y., Sastry, S. Segmentation of dynamic scenes from the multibody fundamental matrix. ECCV Workshop on Vision and Modeling of Dynamic Scenes, Copenhagen (2002)
22. Vidal, R., Soatto, S., Ma, Y., Sastry, S. A Factorization Method for 3D Multi-body Motion Estimation and Segmentation. Technical Report, (2002).
23. Wolf, L.,Shashua, A.. On projection matrices P^k → P^2, k=3,...,6, and their applications in computer vision. ICCV, pages 412-419, (2001)
24. Wolf, L., Zomet, A. Sequence-to-Sequence Self Calibration. ECCV Vol 2., pages 370-382, (2002)
25. Wolf, L., Zomet, A. Correspondence-free synchronization and reconstruction in a non-rigid scene. ECCV Workshop on Vision and Modeling of Dynamic Scenes, Copenhagen (2002)