Tracking Based Structure and Motion Recovery for Augmented Video Productions

Kurt Cornelis[*]
K. U. Leuven, ESAT-PSI
Kasteelpark Arenberg 10
B-3001 Leuven-Heverlee, Belgium
kurt.cornelis@esat.kuleuven.ac.be

Marc Pollefeys[†]
K. U. Leuven, ESAT-PSI
Kasteelpark Arenberg 10
B-3001 Leuven-Heverlee, Belgium
marc.pollefeys@esat.kuleuven.ac.be

Luc Van Gool
K. U. Leuven, ESAT-PSI
Kasteelpark Arenberg 10
B-3001 Leuven-Heverlee, Belgium
luc.vangool@esat.kuleuven.ac.be

ABSTRACT

Augmented Reality (AR) can hardly be called uncharted territory. Much research in this area revealed solutions to the three most prominent challenges of AR: accurate camera state retrieval, resolving occlusions between real and virtual objects and extraction of environment illumination distribution. Solving these three challenges improves the illusion of virtual entities belonging to our reality. This paper demonstrates an elaborated framework that recovers accurate camera states from a video sequence based on feature tracking. Without prior calibration knowledge, it is able to create AR Video products with negligible/invisible jitter or drift of virtual entities starting from general input video sequences. Together with the referenced papers, this work describes a readily implementable and robust AR-System.

Keywords

Augmented Reality, accurate registration, jitter reduction

1. INTRODUCTION

1.1 Previous Work

The literature on Augmented Reality has become abundant [2]. Research reports on accurate registration [22, 28, 30], resolving virtual-real occlusions [4, 7, 33] and extraction of environment illumination [11, 20, 21] are readily available. Fast and efficient camera retrieval algorithms using affine representations have been implemented successfully [16], however these fail to model completely a real perspective camera. Camera state retrieval obeying the full perspective model has been proven possible on sparse image sequences [3, 19], but these require some baseline between 2 consecutive images for corner matching which often lacks between videoframes. A simple extension of [19] for video is explained in [10]. It consists of selecting a sparse image sequence from the video and applying [19]. All other frames are interpolated for their camera state, therefore the time-continuity and abundance of videoframes is not exploited. Other algorithms are based on the small interframe camera motion of a video [9, 31]. They solve approximately for the camera motion between 2 consecutive frames based solely on the image motion of features. The fact that features originate from a single 3D point is not or weakly exploited while this could reduce drift. This fact, however, is exploited by some recursive techniques [1, 5, 8] that use Kalman Filter theory. Kalman Filtering requires a system model which is often not adequate for real camera motions, e.g. the assumption of constant velocity or random walk camera motion, and therefore slightly biases the output to follow the system model. This paper describes an AR-System that exploits the time-continuity of the input video to extend [10] by using feature tracking, exploits the fact that a feature originates from a single 3D point to reduce drift and uses no other model than the perspective camera model.

[*]Kurt Cornelis is a research assistant of the Fund for Scientific Research - Flanders (Belgium)(F.W.O. - Vlaanderen)
[†]Marc Pollefeys is a postdoctoral fellow of the Fund for Scientific Research - Flanders (Belgium)(F.W.O. - Vlaanderen)

1.2 Overview

The paper outlines the process from input video towards Augmented Video. First the most fundamental element, feature tracking, is discussed. The resulting tracks are used by an iterative Structure and Motion Recovery (SMR) algorithm, as explained in a following section. The framework needs initialization and final error minimization, both covered in separate sections. The merging of virtual entities and results on real-life footage will be demonstrated in the final sections along with experimental determination of error buildup. We will finish with a short summary and discuss future research topics.

2. TRACKING FEATURES

2.1 Point Feature Tracking

SMR is made possible by correspondences between images. These correspondences constitute the framework's corner stone. They exist in all forms: points, lines [15, 32] and even regions [29]. Points are the most elementary and present in almost any footage of real environments. As we want to process the most diverse videos, we opt to track points. A reliable point feature tracker, the '*KLT-tracker*', is discussed in [24, 25]. As the tracked points are proven to be the ones that are tracked best, it supplies a solid theoretical basis. Free source code at *http://vision.stanford.edu/˜birch/klt/* makes this is a readily implementable building block.

2.2 Improving Feature Tracking

After experimentation with the original tracker code some shortcomings surfaced. First, it attempts to track a constant number of features. It can happen that features pile up in one part of the image. However, having uniformly distributed features across the image increases accuracy and robustness of SMR algorithms. Consequently, we divide the image into sections of uniform size in which a minimum of features are tracked. Six sections along the smallest image side proved satisfactory.

Another problem popped up while processing a turntable sequence. Figure 1 shows how the turntable motion doesn't alter the silhouette of the vase. This static silhouette is referred to as '*an apparent contour*'. Features can get stuck on them, leading to a feature pile-up. Detection of feature collisions is an effective remedy as feature points initially distant that become too close to each other, e.g. 3 pixels, can indicate such a pile-up. The tracking of colliding features can be terminated, making room for new features. This is also advantageous because 2 colliding features become visually indistinguishable, leading to redundancy as two 3D points projecting onto the same 2D point yield as much information for camera recovery as a single 3D point.

2.3 Erroneous Feature Tracks

Structure and Motion Recovery is based on the static part of the scene. Therefore, tracks are associated preferably with static 3D points. However, different scenarios can yield tracks corresponding to moving 3D points. For instance, a feature track with associated static point can suddenly lock onto another static 3D point. This can be interpreted as a 3D point remaining static for a while and then jumping abruptly to another static position. Also T-junctions spanning depth discontinuities, see Figure 1, don't correspond to static 3D points as the feature is formed by an intersection in the image, not by a real intersection in the scene. It is the tracked position of a 3D point moving on either one of the scene parts giving rise to the T-junction. Also features stuck on '*apparent contours*' can be seen as 3D points moving on the object's surface such that their projection is always on the silhouette. Feature collision detection reduced their number. Naturally, tracked features corresponding to moving real objects also violate the static 3D point assumption.

The fore-mentioned scenarios could be detected by correlation - based matching with a local window at the frame of feature instantiation [24]. Instead we shift the problem to the stage of the SMR algorithm where a moving point can be detected through geometric reasoning. When a 3D point's projection starts to deviate from the expected projection observed by the feature track, this probably indicates a moving 3D point and it can therefore be eliminated.

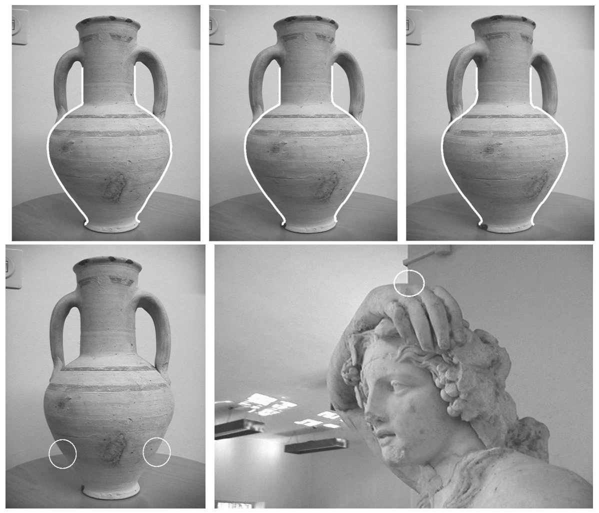

Figure 1: Top: 3 frames from a turntable sequence. On rotating the turntable the silhouette (white) of the vase remains practically the same. However, this silhouette doesn't correspond to any static 3D geometry entity, and is therefore called an *'apparent contour'*
Bottom: The intersection of high-gradient edges at different depths yield a perfect feature to track. However, it doesn't correspond to any physical point as the intersection only takes place in the image. This is called a *'T-junction'*

3. STRUCTURE AND MOTION RECOVERY (SMR)

3.1 Iterative Core Algorithm

Tracked features yield correspondences between frames that can be used to recover structure and motion. Structure and motion go hand in hand as the availability of one allows to solve for the other, leading to an iterative scheme. We're left with a 'chicken-egg' problem which will be solved in section 4. Figure 2 describes the iterative nature: knowing some camera states, corresponding 2D points can be reconstructed. Knowing these 3D points and their 2D positions in other frames, we can compute the camera state of these frames. In turn these allow reconstruction of new 3D points, completing the cycle.

Keeping future real-time implementation in mind, camera states of consecutive frames are recovered from 3D-2D correspondences as time advances. New 3D points are reconstructed when there are insufficient 3D-2D correspondences for reliable camera state estimation. Only earlier frames are used to reconstruct new 3D points to respect causality.

3.2 Accuracy Considerations

SMR accuracy is achieved by minimizing image reprojection errors. These are the errors visible to human observers and therefore form the natural goal for minimization. A single camera state recovery consists of 2 steps. First, a RANSAC algorithm [13] determines the sample, a minimum number of 3D-2D correspondences needed to compute the camera state, whose corresponding camera state yields the

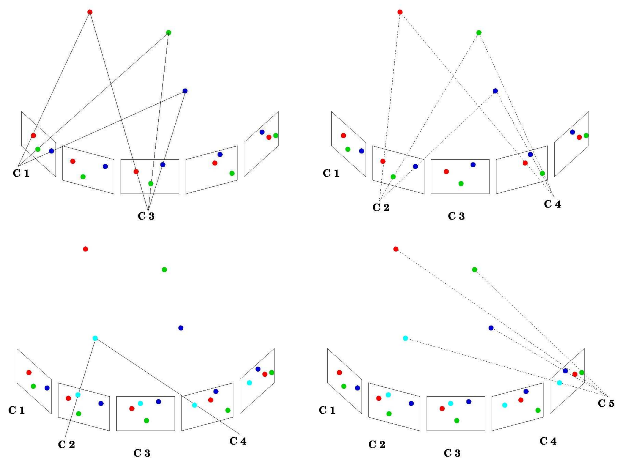

Figure 2: left to right, top to bottom: knowing some cameras allows to reconstruct 3D points. Together with tracked 2D positions in other frames these allow to solve for new cameras. The latter can then be used to reconstruct new points, etc. (This figure is reproduced in color on page 000.)

largest number of 3D points that project close enough to the 2D positions observed by the feature tracks. These 3D-2D correspondences are the inliers of the camera state. Next, an iteratively re-weighted least squares optimization [14] adapts the found camera state to minimize the total reprojection error of all inliers.

In the same way accurate 3D point reconstruction associated with a feature track is achieved. First, a RANSAC algorithm determines the sample, the 3D point reconstruction using back-projection from only 2 views, which yields the largest number of frames in which the reprojection lies close enough to the 2D position observed by the feature track. These frames form the inliers. Next, an iteratively re-weighted least squares algorithm adapts the reconstruction to minimize the total reprojection error in all inlier frames.

RANSAC stands for *RANdom SAmpling Consensus*, but by taking advantage of the video's time-continuity we can get rid off the '*random*' part. After recovery of the previous frame's camera state, the 3D-2D correspondences are ordered according to reprojection error. 3D-2D correspondences which yield the smallest reprojection error in the previous frame will be good inliers for the current frame and therefore can be used as a good sample to estimate the current camera state. This speeds up the search for good samples.

3.3 Removing Erroneous Tracks

SMR is based on the static scene part and therefore the detection of moving scene points is important. This detection is accomplished using multiple reprojection inlier thresholds. 3D points which project within the inlier threshold distance of the 2D tracker position are called inliers. For camera state estimation a very strict inlier threshold is used to exclude wrong 3D-2D correspondences from the iteratively re-weighted least squares minimization. This is crucial as least squares solutions are very sensitive to outliers. Because of the very strict inlier threshold correct 3D-2D correspondences can be outliers due to noise, radial distortions, etc. So we incorporate a second, more flexible inlier threshold which, when violated, signals moving 3D points.

The ability to detect/reject moving points makes this framework resilient to erroneous tracks resulting from T-junctions and apparent contours, which can even appear in static scenes. Also moving real objects can be handled as long as the static scene part outweighs them in each frame. This follows from RANSAC selecting the camera state which yields the largest number of 3D-2D inliers, namely the static scene part.

3.4 Recovering Feature Tracks

Some scene parts can return further along the input video. Recognizing corresponding feature tracks in different video parts as resulting from a single 3D point can reduce drift. Often feature tracks are lost shortly due to noise or blur, but a new feature track is re-instantiated at the same place as the lost feature, therefore actually representing the same 3D point. On one hand, without recovery of corresponding tracks different 3D reconstructions would be made at the same location leading to a storage overhead. On the other hand, and more importantly, the drift during iterative SMR can be diminished by locking back onto already instantiated 3D points.

The recovery is accomplished by reprojecting already created 3D points onto the current image. When a reprojection lies close to a feature track that not yet has an associated 3D point, the reprojected 3D point is assigned as its reconstruction. A single 3D point cannot be recovered by 2 different feature tracks in the same frame as these tracks would have to lie too close and are therefore eliminated beforehand by the feature collision detection. As it is physically impossible for a 3D point to have 2 projections in the same frame the infeasibility of this situation is mandatory. However, it is possible for several 3D points to project onto the same 2D coordinates and therefore assigned to the same feature track. To solve this ambiguity the texture environment of the feature track can be compared with that expected by the different 3D points. However, comparing textures through normalized cross correlation is not always effective. A simple, and more robust, solution relies solely on geometric information and the ability to remove 3D-2D correspondences belonging to moving 3D points. If several 3D points are assigned to a single feature track, only one will be correct. The correct correspondence will project close to the 2D position observed by the feature track in all images. The wrong correspondences most probably will not as demonstrated in Figure 3. The camera motion reveals the correct correspondence.

4. INITIALIZATION OF STRUCTURE AND MOTION RECOVERY

4.1 Initial Motion Recovery

Either known structure or motion can initialize the iterative SMR process. Sometimes scene knowledge is available through measured landmark positions or calibration grids. As we want to handle the most general videos, we need to cope with a lack of scene information. The remaining

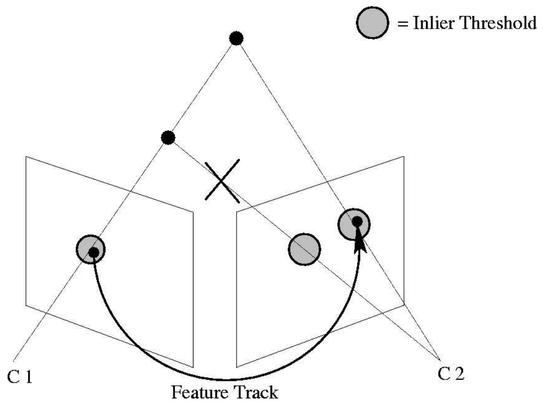

Figure 3: Although in the left image multiple 3D-2D correspondences can be assigned, the camera motion revealed the correct correspondence

option is to find some initial camera states using only the feature tracks that depict correspondences between frames. The correspondences between 2 frames are also governed by epipolar geometry expressed by a fundamental matrix $\mathbf{F}$. Robust and accurate methods to solve for fundamental matrices are explained in [17]. $\mathbf{F}$ depends on the camera states of both frames and therefore these are in some form present within $\mathbf{F}$. We are left with the problem of extracting them. $\mathbf{F}$ can be expressed in terms of internal and external camera calibration parameters:

$$\mathbf{F} = [e_2]_x \mathbf{K}_2 \mathbf{R}_{rel} \mathbf{K}_1^{-1} \text{ with } \mathbf{K}_i = \begin{bmatrix} f_i & s_i & u_i \\ 0 & r_i f_i & v_i \\ 0 & 0 & 1 \end{bmatrix}$$

in which f_i denotes the focal length, r_i the pixel aspect ratio, s_i the skew and (u_i, v_i) the principal point of camera i. $\mathbf{R}_{rel}$ is the relative rotation between both cameras and $[e_2]_x$ is the vector product with the epipole in image 2 expressed as a matrix multiplication. The 9 elements of $\mathbf{F}$ give 7 constraints as $\mathbf{F}$ can only be determined up to a scale factor and is rank deficient. The variables are focal length, aspect ratio, skew and principal point in twofold and the relative orientation and translation between both frames. Due to scale ambiguity the translation's magnitude is undetermined, and so it has only 2 degrees of freedom.

Because the variables outnumber the constraints we fix some variables on realistic values. For realistic cameras skew can be assumed zero, aspect ratio constant and the principal point will lie close to the image center. Fixing these variables allows to solve for the other variables [6]. Depending on the values we attribute to the fixed variables we end up with different camera state variables. We opt to look for the decomposition of $\mathbf{F}$ into 2 camera states with almost equal focal length because in any video there are parts where the focal length doesn't change.

4.2 Automatic self-calibration

Structure and motion from uncalibrated video can only be recovered up to an unknown projective transformation if no further scene or camera geometry assumptions are made. Up to a similarity transformation only one reconstruction corresponds to the real Euclidean environment. In AR we want to incorporate Euclidean virtual entities on which we can perform only similarity transformations so it is important to recover Euclidean structure and motion.

This has different implications for offline and real-time AR. In real-time AR, Euclidean structure has to be recovered from the beginning. This is done at the moment of virtual object insertion. The virtual objects are Euclidean and therefore their 3D model vertices have Euclidean coordinates. By imposing the projection of some vertices in several frames, the Euclidean camera state of these frames can be recovered as one disposes of Euclidean 3D-2D correspondences. Given the original and corresponding Euclidean camera states one can find the projective transformation that updates the framework to Euclidean space. For offline AR the whole SMR can be done in any projective framework and upgraded at the end to Euclidean using self-calibration techniques [6, 12, 18, 26]. But working in Euclidean space from the start can be interesting. Scene points in the far range are less important to the viewer and reconstructed with poor accuracy because the lines of sight by which they are reconstructed intersect at a very small angle. When working in Euclidean space, reconstructions in the far range can be detected and avoided. Also a human observer can easily spot when the SMR is failing as he/she has some expectations of the Euclidean outcome.

As more frames are processed we can determine whether or not the current framework is really Euclidean. If recent recovered camera states have large skews, unrealistic aspect ratios, focal lengths or principal points we know the framework is probably not Euclidean. We can compute a transformation which brings it closer to the Euclidean space by applying a simple self-calibration technique: from all available camera states we can take a representative sample containing some that already are realistic and others that are not. Between each pair the fundamental matrix can be recomposed and decomposed into 2 camera states which are realistic in a Euclidean world. The projective transformation that brings the original camera pair to the new realistic camera pair can be determined. This is done for every sample pair after which a RANSAC and least squares algorithm determines the transformation that leads to the largest number of realistic cameras.

5. REDUCING JITTER AND DRIFT OF VIRTUAL ENTITIES

Several sources contribute to image reprojection errors: errors on recovered structure, camera states, etc. Bundle adjustment procedures [27] put together both structure and motion in one large optimization that minimizes the total reprojection error. This will render the remaining jitter and drift of the virtual entities negligible/invisible. The moment of bundle adjustment application depends on the use of the AR-System. For offline AR the bundle adjustment can be postponed until all structure and motion is recovered by the iterative SMR scheme. For real-time AR 2 stages exist. First, an initial video is used to reconstruct and bundle-adjust the scene offline. Next, the bundled 3D points are tracked during real-time operation. No new feature points are instantiated because they lack the accuracy of the bundled 3D points and are liable to error buildup. Recovering feature tracks corresponding to the same bundled 3D point is still desirable.

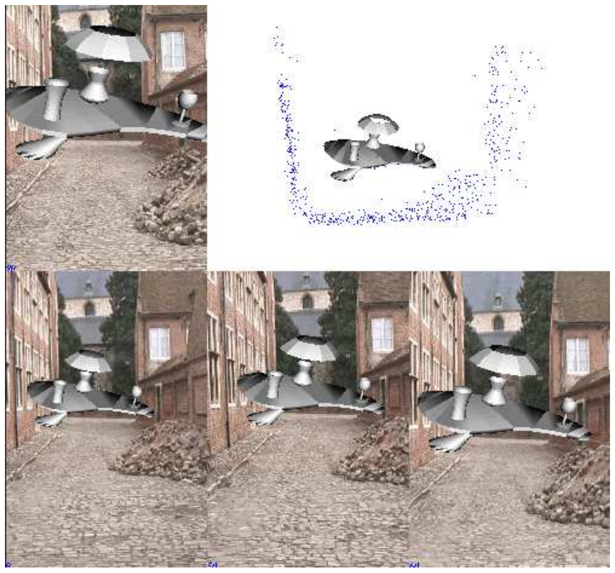

Figure 4: The AR Video Production interface : In the top right view the virtual objects can be roughly placed within the reconstructed 3D environment. The result of this placement can be viewed instantaneously on some selected frames. The pose of the virtual entities can then be fine-tuned to meet expectations in the viewed images. (This figure is reproduced in color on page 000.)

6. CREATION OF THE AUGMENTED VIDEO

Having obtained the bundled structure and motion these allow to incorporate virtual entities into the video using a typical computer graphics package, e.g. the freeware code of VTK [23], *The Visualization Toolkit.* Merging is achieved by assigning the original frame as a background environment map and rendering the virtual objects on top. In this way anti-aliasing between the virtual objects and the real world is obtained automatically. For the moment virtual objects can occlude real ones but not vice versa. The virtual objects can be positioned in 3D by using similarity transformations and using recovered scene structure to guide the placement, see Figure 4. The result for some selected frames can be viewed interactively. One can then fine-tune the state of the virtual entities by using all visual available information in those frames, that may not have been reconstructed in 3D by the SMR algorithm.

7. EXPERIMENTAL RESULTS

This section runs through the whole production cycle of an AR Video. The input video shows an ancient fountain at the archaeological site of Sagalassos in Turkey. The pillars and roof have not yet been restored by the archaeologists. Using the AR software we are able to incorporate them to get an idea of the final restoration beforehand. Figure 5 shows several frames, resolution 720x576, of the input video. Tracked features, around 800 per frame, are displayed in Figure 6. Next, 2 initial frames are selected to obtain their camera states by decomposition of their fundamental matrix. Subsequently, the iterative SMR algorithm is triggered, ending up with scene structure and camera states. These suffer from error buildup as the walls bend backwards, see Figure 7. A bundle adjustment minimizes reprojection errors and straightens the walls as shown in the same figure. After satisfactory placement of the virtual entities the recovered camera states are used to produce the Augmented 'restoration' of the ancient fountain, Figure 8. The resulting video can be downloaded from *http://www.esat.kuleuven.ac.be/˜kcorneli/ARVideos.* Jitter and drift are negligible/invisible. Therefore this framework is able to comply with the primary requirement of AR-Systems: accurate registration of virtual objects within a real environment.

We also performed some experiments to visualize the error buildup during the iterative SMR algorithm. The final bundle adjustment, section 5, should minimize this buildup. However since the buildup of errors jeopardizes the recovery of feature tracks in the SMR algorithm, we determined the error buildup before bundling to get a better idea. We generated an artificial turntable sequence in which one round-trip consists of 100 equally spaced frames of 720x576 pixels. We made 11 consecutive round-trips so the sequence has 1100 frames in total. As the sequence is periodic with period 100 we know that all cameras and images with frame numbers separated by a multiple of 100 should be the same, e.g. frame 25, frame 125, frame 225, etc. .

The reprojection error is the error visible to human observers and therefore we like to express error buildup in terms of reprojection error. Because we know that images separated by a multiple of 100 should have the same camera/reprojections the error buildup can be determined as follows: we take a set of cameras which are all separated by a multiple of 100 frames. Subsequently we take the reconstructed scene points and project them into these images using their corresponding estimated cameras. The mean distance between reprojection positions can then be considered as a measure for the error buildup between round-trips. The mean reprojection error is averaged over all possible camera sets to be statistically relevant. This mean reprojection error is calculated for cameras which are separated by a single round-trip, see Table 1, and also between the first and all other round-trips yielding the total error buildup starting from the beginning of the sequence, see Table 2. The mean reprojection errors are expressed in pixels.

We calculated these error buildups for different scenarios. Each scenario consists of using the whole framework except a single improvement. In one scenario we left out the fact that we covered the image plane with sections/tiles to yield uniformly distributed features. In another we left out the collision detection of features to avoid feature pile-up. Removing the refinement of 3D point reconstructions and camera estimation, recovery of already reconstructed tracks, removal of moving 3D points and auto-calibration are also considered as different scenarios. The results when the complete framework is used are given at the bottom of each table for benchmarking. From these results we immediately notice the importance of recovering already reconstructed 3D points for the reduction of drift due to error buildup. When we didn't remove the moving 3D points the error buildup increased rapidly despite of the recovery of feature tracks. Because no feature tracks were being removed anymore we had problems of getting passed the sixth round-trip as the memory usage became too large.

Table 1: relative error buildup: frame 100x(i-1) - frame 100xi

Scenario \ i	1	2	3	4	5	6	7	8	9	10
tiling	1.121	0.070	0.068	0.066	0.046	0.051	0.040	0.044	0.058	0.045
collision	0.285	0.049	0.045	0.048	0.050	0.038	0.037	0.046	0.047	0.048
refine scene points	0.448	0.100	0.079	0.073	0.089	0.068	0.061	0.065	0.063	0.063
refine cameras	3.051	0.585	0.865	0.711	0.626	0.691	0.532	0.467	0.423	0.383
recover tracks	1.108	1.173	1.181	1.139	1.141	1.189	1.361	1.222	1.212	1.427
moving 3D points	1.604	1.394	1.295	1.511	1.490	-	-	-	-	-
auto-calibration	0.532	0.0520	0.039	0.035	0.040	0.039	0.040	0.036	0.036	0.042
complete	0.193	0.040	0.027	0.031	0.030	0.036	0.039	0.036	0.032	0.034

Table 2: total error buildup: frame 0 - frame 100xi

Scenario \ i	1	2	3	4	5	6	7	8	9	10
tiling	1.121	1.117	1.116	1.113	1.115	1.113	1.112	1.112	1.115	1.112
collision	0.285	0.267	0.266	0.267	0.263	0.262	0.268	0.268	0.260	0.273
refine scene points	0.448	0.481	0.469	0.463	0.465	0.475	0.470	0.475	0.464	0.478
refine cameras	3.051	3.215	3.234	3.236	3.405	3.192	3.205	3.261	3.212	3.205
recover tracks	1.108	2.177	3.281	4.296	5.362	6.432	7.720	8.883	9.963	11.264
moving 3D points	1.604	2.871	4.082	5.516	6.622	-	-	-	-	-
auto-calibration	0.532	0.534	0.526	0.519	0.526	0.524	0.526	0.522	0.525	0.520
complete	0.193	0.180	0.177	0.178	0.177	0.179	0.175	0.179	0.177	0.177

8. CONCLUSION

This paper described the development of a complete Structure and Motion Recovery (SMR) algorithm based on point feature tracking for Augmented Video production.

In a first section we discussed the basic building blocks of the framework, tracked point features. It was shown how free available tracker code could be tuned to meet expectations. A following section explained how these tracked features are used by an iterative SMR algorithm. Accuracy considerations were mentioned, the recovery of feature tracks arising from the same 3D point was discussed and its importance to reduction of error build up emphasized. The initialization of the iterative SMR algorithm was the topic of a next section. We demonstrated how using the tracked features and epipolar geometry between 2 frames, the corresponding fundamental matrix could be calculated and decomposed into 2 realistic camera states. Subsequently automatic self-calibration was described. From the moment camera states showed lack of Euclidean properties, the framework was brought closer to the real Euclidean framework. Working in Euclidean frameworks was exploited by discarding the inaccurate reconstruction of 3D points in the far range. The following section showed how a bundle adjustment procedure could minimize the image reprojection error after all structure and motion had been solved for. This rendered the jitter of virtual entities invisible in the final Augmented Video. An experimental run through the complete framework on real-life footage showed that the performance meets the primary requirement of AR-Systems: accurate registration of virtual entities within the real environment. If this requirement was not met, the illusion of virtual objects belonging to our real world would be lost.

9. FUTURE WORK

The AR framework described in this paper is mainly focussed on offline Augmented Video production. However, at each step the future real-time extension was taken into account. This extension will form the subject of future research. For offline production a huge number of features can be tracked to improve accuracy and robustness. However, in real-time implementations the number of features has to be reduced to meet computational requirements. It becomes important to figure out which are the best features to track. A trade-off has to be made between time-efficiency and accuracy.

The use of texture correlation to remove erroneous tracks and recover already reconstructed 3D points was replaced by a pure geometric reasoning about removing moving 3D points. However, these correlation-based techniques can be added to make the system more robust in future versions.

10. ACKNOWLEDGMENTS

We wish to gratefully acknowledge the financial support of the I.W.T.(Institute for the Promotion of Innovation by Science and Technology in Flanders) ITEA99002 BEYOND project, 3D Murale project and the F.W.O.(Fund for Scientific Research - Flanders) G.0223.01 project.

11. ADDITIONAL AUTHORS

Additional authors: Maarten Vergauwen and Frank Verbiest (email:`vergauwe@esat.kuleuven.ac.be and verbiest@esat.kuleuven.ac.be`).

12. REFERENCES

[1] A. Azarbayejani and A. Pentland. Recursive estimation of motion, structure, and focal length. *PAMI*, 17(6):562–575, June 1995.

[2] R. Azuma. A survey of augmented reality. *ACM SIGGRAPH '95 Course Notes No. 9 - Developing Advanced Virtual Reality Applications*, August 1995.

[3] P. Beardsley, P. Torr, and A. Zisserman. 3d model acquisition from extended image sequences. *Computer*

Vision - ECCV'96, Lecture Notes in Computer Science, 1065:683–695, 1996.

[4] M.-O. Berger. Resolving occlusion in augmented reality: a contour based approach without 3d reconstruction. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '97), IEEE, IEEE Computer Society Press*, pages 91–96, 1997.

[5] J.-Y. Bouget and P. Perona. Visual navigation using a single camera. *ICCV5, Los Alamitos, CA, IEEE Computer Society Press*, pages 645–652, 1995.

[6] S. Bougnoux. From projective to euclidean space under any practical situation, a criticism of self-calibration. *Sixth International Conference on Computer Vision*, pages 790–796, January 1998.

[7] D. E. Breen, R. T. Whitaker, and E. Rose. Interactive occlusion and collision of real and virtual objects in augmented reality. *Technical Report ECRC-95-02, ECRC, Munich, Germany*, 1995.

[8] T. Broida, S. Chandrashekhar, and R. Chellappa. Recursive 3-d motion estimation from a monocular image sequence. *IEEE Trans. Aerospace and Electronic Systems*, 26:639–656, 1990.

[9] M. J. Brooks, L. Baumela, and W. Chojnacki. An analytical approach to determining the egomotion of a camera having free intrinsic parameters. *Tech. Rep. 96-04, Dept. Computer Science, University of Adelaide*, January 1996.

[10] K. Cornelis, M. Pollefeys, M. Vergauwen, and L. V. Gool. Augmented reality from uncalibrated video sequences. *3D Structure from Images - SMILE 2000, Lecture Notes in Computer Science*, 2018:144–160, 2001.

[11] P. Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. *Proceedings SIGGRAPH 98*, pages 189–198, July 1998.

[12] O. Faugeras, Q.-T. Luong, and S. Maybank. Camera self-calibration: Theory and experiments. *Computer Vision - ECCV'92, Lecture Notes in Computer Science*, 588:321–334, 1992.

[13] M. Fischler and R. Bolles. Random sampling consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun. Assoc. Comp. Mach.*, 24:381–395, 1981.

[14] R. Haralick, H. Joo, C. Lee, X. Zhuang, V. Vaidya, and M. Kim. Pose estimation from corresponding point data. *IEEE Trans. SMC*, 19(6):1426–1446, November/December 1989.

[15] C. Harris and M. Stephens. A combined corner and edge detector. *Fourth Alvey Vision Conference*, pages 147–151, 1988.

[16] K. N. Kutulakos and J. Vallino. Affine object representations for calibration-free augmented reality. *IEEE Virtual Reality Annual International Symposium (VRAIS)*, pages 25–36, 1996.

[17] Q.-T. Luong and O. Faugeras. The fundamental matrix: theory, algorithms, and stability analysis. *Intl. Journal of Computer Vision*, 17(1):43–76, 1996.

[18] M. Pollefeys, R. Koch, and L. V. Gool. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. *International Journal of Computer Vision*, 32(1):7–25, 1999.

[19] M. Pollefeys, R. Koch, M. Vergauwen, and L. V. Gool. Hand-held acquisition of 3d models with a video camera. *Proceedings Second International Conference on 3-D Imaging and Modeling (3DIM'99), IEEE Computer Society Press, Los Alamitos*, pages 14–23, 1999.

[20] I. Sato, Y. Sato, and K. Ikeuchi. Acquiring a radiance distribution to superimpose virtual objects onto a real scene. *IEEE Transactions on Visualization and Computer Graphics*, 5(1), January-March 1999.

[21] I. Sato, Y. Sato, and K. Ikeuchi. Illumination distribution from brightness in shadows: Adaptive estimation of illumination distribution with unknown reflectance properties in shadow regions. *Proceedings of IEEE International Conference on Computer Vision (ICCV'99)*, pages 875–882, September 1999.

[22] C. Schütz and H. Hügli. Augmented reality using range images. *SPIE Photonics West, The Engineering Reality of Virtual Reality 1997, San Jose*, 1997.

[23] W. Schroeder, K. Martin, and B. Lorensen. The visualization toolkit 2nd edition. *Prentice Hall, New Jersey*, 1998.

[24] J. Shi and C. Tomasi. Good features to track. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 593–600, 1994.

[25] C. Tomasi and T. Kanade. Detection and tracking of point features. *Carnegie Mellon University Technical Report CMU-CS-91-132*, April 1991.

[26] B. Triggs. The absolute quadric. *Proc. 1997 Conference on Computer Vision and Pattern Recognition, IEEE Computer Soc. Press*, pages 609–614, 1997.

[27] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment - a modern synthesis. *Vision Algorithms: Theory and Practice*, pages 298–372, 1999.

[28] M. Tuceryan, D. S. Greer, R. T. Whitaker, D. Breen, C. Crampton, E. Rose, and K. H. Ahlers. Calibration requirements and procedures for augmented reality. *IEEE Transactions on Visualization and Computer Graphics*, pages 255–273, September 1995.

[29] T. Tuytelaars and L. V. Gool. Content-based image retrieval based on local affinely invariant regions. *Third International Conference on Visual Information Systems, Visual99*, pages 493–500, June 1999.

[30] M. Uenohara and T. Kanade. Vision-based object registration for real-time image overlay. *Proceedings CVRMED'95*, pages 14–22, 1995.

[31] T. Viéville and O. Faugeras. The first order expansion of motion equations in the uncalibrated case. *Computer Vision and Image Understanding*, 64(1):128–146, July 1996.

[32] T. Viéville, Q. Luong, and O. Faugeras. Motion of points and lines in the uncalibrated case. *International Journal of Computer Vision*, 17(1):7–42, 1996.

[33] M. Wloka and B. Anderson. Resolving occlusion in augmented reality. *ACM Symposium on Interactive 3D Graphics Proceedings*, pages 5–12, April 1995.

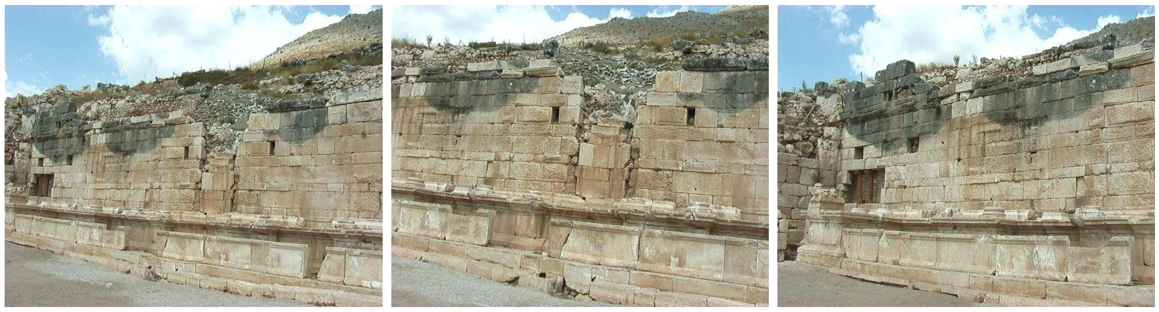

Figure 5: Some frames from the original input video (This figure is reproduced in color on page 000.)

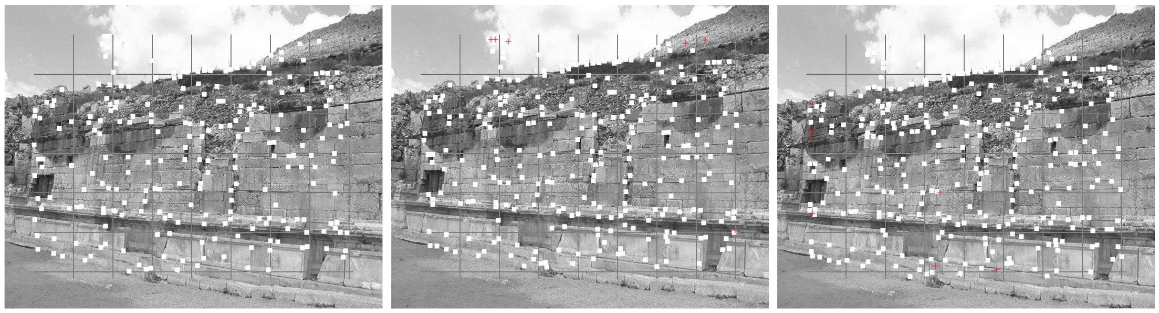

Figure 6: Some features tracked throughout consecutive frames

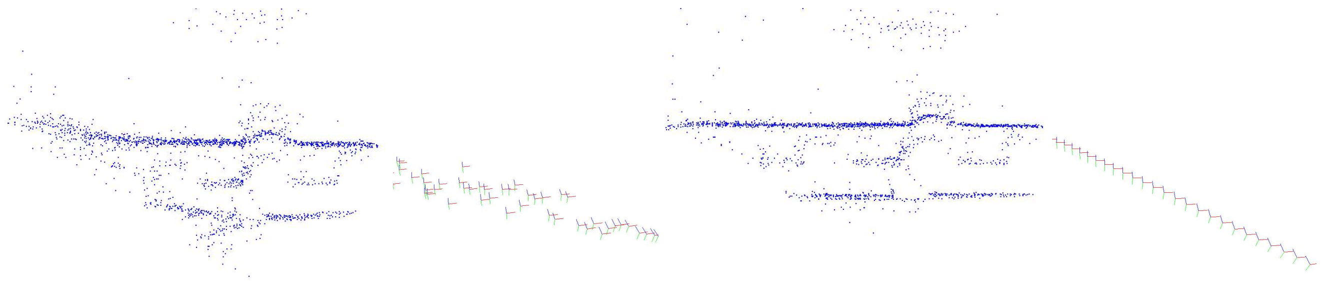

Figure 7: left: the recovered scene structure (top view of fountain) and camera path before applying a bundle adjustment. right: scene structure and camera path after applying a bundle adjustment

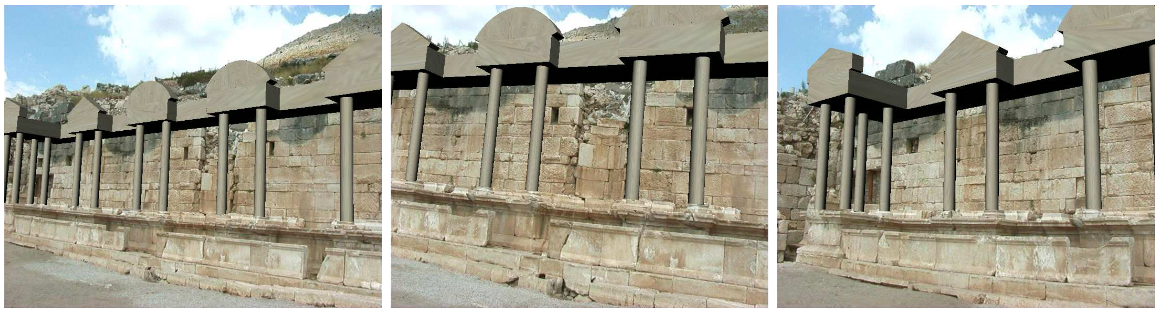

Figure 8: Some frames from the resulting Augmented Video (This figure is reproduced in color on page 000.)