A Fast and Robust Line-based Optical Tracker for Augmented Reality Applications

Didier Stricker, Gudrun Klinker, Dirk Reiners
Fraunhofer Project Group for Augmented Reality at ZGDV
Rundeturmstraße 6, 64283 Darmstadt, Germany
{stricker, klinker, reiners}@igd.fhg.de
http://www.igd.fhg.de/www/igd-a4/

Abstract

The correct alignment of real and virtual objects is one of the key technologies in Augmented Reality (AR). The user's point of view must be determined accurately and tracked over time.

In this paper, we present an optical tracking system, focusing on real time performance and robustness including fast recovery from unpredictable abrupt motions, as well as stable handling of partial occlusions of tracking targets. The algorithm tracks linear features of landmarks and other objects. For increased performance, we do not extract entire lines but rather use only a few sample points. The 3D motion recovery is done iteratively using robust estimation methods.

The system is able to run on standard hardware with high frame rate (20-25 Hz on a O_2) and is robust enough to track a hand-held or head-worn camera.

We finish with a presentation of AR-applications in the domains of exterior construction, 3D user interfaces and games.

1 Introduction

Augmented Reality (AR) integrates computer generated information into views of the real world, being potentially useful in many application domains. It can assist users in maintenance and repair tasks for complex devices [18]. In exterior construction, the CAD model can be overlaid on video of the construction site and thus help during the planning and the construction phase[10]. Furthermore, information from other sensors can also be directly merged into the user's view to provide supplementary information on-line in operations, as presented in the medicine and surgery context [4, 7].

1.1 Related work

The registration of the virtual objects with the real world is a crucial problem in AR. In order to preserve the coherence of the augmented scene the different sensors have to be carefully calibrated [23]. Commercial tracking devices can be used, like magnetic trackers [18] and active LED-systems [20, 25], but the precision or the extent of the working space are not sufficient for a lot of AR applications, so that many researchers are now developing computer vision-based methods. Typically, special landmarks are placed in the scene and used to calibrate and track the camera [9, 5, 28, 15]. Other approaches use directly image features but the initial pose has to be given interactively [17] or the system automatically finds correspondences in already calibrated images, from which the camera pose is derived [24]. For real time applications, the measurement frequency plays a crucial role, thereby limiting the system complexity and the set of motions that can be tracked [3]. Improvements of the robustness can be achieved by fusing the image measurements with informations of other sensors like an inertial tracker [2] or a magnetic tracker [21].

In computer vision, 3D camera motion is a classic topic often related to robotic and navigation tasks. A particularly successful approach is based on the physical modeling of the motion and exploitation of the image measurements with a Kalman filter [6, 30, 1, 11].

In our applications, we assume that motion of head worn cameras can be very abrupt and unpredictable because users can turn their heads very quickly, resulting in major shift in the image. Thus, we do not use a motion model and favor the dynamic properties of the tracker. Our system has been influenced by the approaches related in [12, 27, 13], but we pay more attention to the dynamic and real time aspects by optimizing the feature search process and developing a new adapted motion recovery algorithm.

1.2 Paper outline

The goal of our system is to track a hand-held or head-worn camera with high frame rate using standard hardware. We present an algorithm showing the

feasibility of highly precise and fast 3D tracking with computer vision methods. After an overview of our system we describe the details of the initialization and the 3D tracking algorithms. Finally, we evaluate the system performance and present some applications.

2 System overview

2.1 Requirements

The user should feel free in his movements and actions. This goal imposes the following system features:

- *Automatic and fast initialization.* The system should not require users to interactively support the tracker initialization phase.

- *Real time tracking.* The frame rate should be higher than 10 Hz, ideally more than 20 Hz. The goal is to minimize the dynamic error, i.e., the lag between rendering the augmented view for the user and the view of the real world. This condition is not only relevant for see-through (Figure 1) [2] but also for video feed-through applications or when the image augmentation is done by another process. Fast tracking minimizes also the inter-frame difference and thus the searching distance between tracking features.

- *Error supervision and fast re-initialization.* The system should guarantee error margins for the alignment of the augmentation with the real world. When it fails, it should be able to reinitialize itself with minimal disturbance for the user, i.e., without user interaction and in a very short time.

Figure 1: Camera used as a tracker device

To achieve the above requirements, we use landmarks for fully automatic calibration and we optimize the tracking system developing appropriate robust algorithms.

2.2 System components

Our approach is model-based, using dark rectangles on a bright background. In order to identify the rectangles independently of the current field of view, they contain one or two rows of red marks defining an 4-bit (or 8-bit) code. This kind of bar-code gives the system a significant flexibility, due to the full identification of each landmark and the potentially high number of them (up to 255).

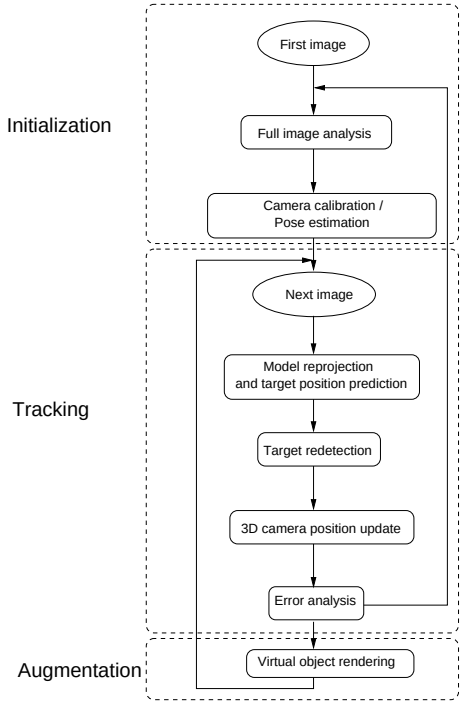


Figure 2: Global software architecture

The initialization is achieved in two steps: First, the corners of the landmarks are extracted and a rough calibration is computed. Next, after back-projecting the landmarks model into the image, the corner positions are refined, enabling an accurate re-calibration. The subsequent image-to-image feature tracking is composed of a position prediction and a target re-detection phase, followed by the update of the 3D position and the corresponding error estimation (see Figure 2).

3 Initialization and calibration

3.1 Robust rectangle detection

The rectangles are first extracted as blob candidates which are defined by three points assumed to lie on three of the four edges. They points are found by scanning the image every n-lines. Two consecutive image gradients white-to-dark and dark-to-white

define the left and right borders of the blob. Starting from their center between them the third point is found by looking vertically for a strong image gradient. We then check the homogeneity of the blob and its average level of blackness along the scanned segments of the blob.

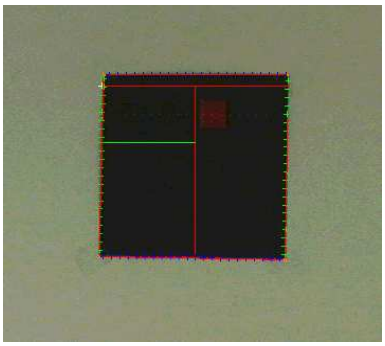


Figure 3: Landmark detection and identification

We then follow the blob contour. Starting from a border pixel, we calculate the gradient norm of all its eight neighbors. To chose the next pixel, we use a criterion C_{ij} defined below. It combines the neighbor gradient, the actual one and the displacement vector.

$$C_{ij} = (\vec{\nabla} I(P_{00}).\vec{v_{ij}}) \, \|\vec{\nabla} I(P_{ij})\|$$

Where $\vec{v_{ij}}$ is the unit vector orthogonal to the displacement from the current pixel at position $(0,0)$ to the neighbor pixel at position (i,j), $I(P_{00})$ and $I(P_{ij})$ represent the the intensity value of the pixel P at position $(0,0)$ and (i,j) $(i=-1,1; j=-1,1)$

To fit rectangles to each blob, the contour samples are classified into four clusters according to their gradient direction, using a standard ISO-data algorithm. After a statistical homogeneity test, i.e. a comparison of cluster sizes, mean values and standard deviations of the gradient norm and direction, we fit a straight line to the edges. The intersection of neighboring lines determines the corner points of the rectangle.

3.2 Identification

To uniquely relate each blob to one of the squares described in a 3D model of the environment, we examine the labeling area within each blob, interpreting the line of red markers as a binary code. The code is read by sampling the line of marks and correlating it with templates representing the encoding of all possible identification numbers. We use the zero mean normalized correlation and consider the rectangle as identified if the highest score can be defined without ambiguity. This method has proven to be very robust and works well also with low quality cameras (like an Indy-Cam) and under bad illumination conditions.

3.3 Calibration

A priori we do not know the kind of virtual object, which will be inserted in the scene and choose for this reason the most general camera model, the pin-hole model. The pose of the camera is defined by the rigid transformation:

$$\begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} = R \begin{pmatrix} x_w \\ y_w \\ z_w \end{pmatrix} + T$$

And the relation to the image coordinate system is:

$$u = f.s_x \frac{x_c}{z_c} + c_{x0}$$

$$v = f.s_y \frac{y_c}{z_c} + c_{y0}$$

We calculate by full calibration the internal (f, s_x, s_y, c_{x0}, c_{x0}) and external parameters (R, T) of the camera using the algorithms described in [26, 22]. The lens distortion is actually not compensated.

4 Tracking algorithm

The main steps of the tracking algorithm are the target position prediction, the re-detection and finally the motion recovery. Each point is detailed in this section.

4.1 Prediction

Due to the randomness character of user head motion, camera motions can be very erratic. We thus limit motion prediction to very basic, linear 2D extrapolations in the image. Our experiments have shown that such fast linear approximations by far outperform in robustness complex, physically more correct 3D motion models. 3D models are much more time-consuming to compute, and thus have to deal with much longer intervals of measurements.

The velocities of the corners of each square are calculated individually from their current and previous position and used to predict their approximate location in the next image. This specifies the search areas within which their exact location will be determined.

4.2 Target re-detection

In the tracking process, linear features of the targets need to be found in the new image. To avoid time consuming feature extraction processing across the entire image, we work locally along search segments. The image, Figure 5, shows those segments, which are defined perpendicular to the re-projected model square sides. In praxis, we limit their directions to the image row, column and diagonal.

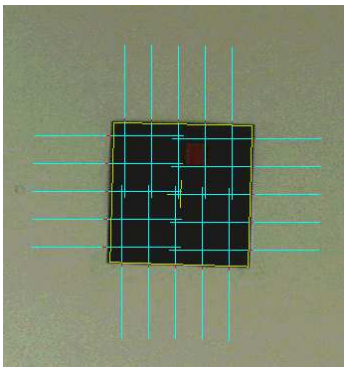


Figure 4: Search segments for target re-detection

We then compute the gradient along each one using a Gaussian kernel and localize the maximum with sub-pixel accuracy by interpolating to the second order. Those new points determine implicitly the new position of the landmarks in the image.

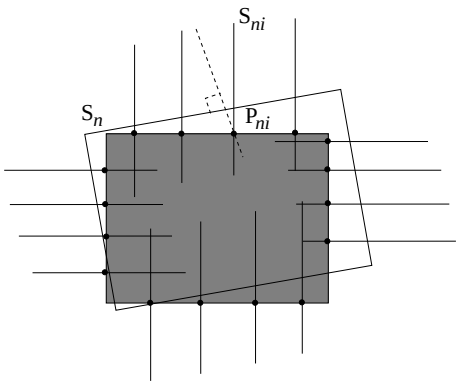


Figure 5: Target re-detection

Let S_n denote the n^{th} segment of the re-projected landmarks model in the image (Figure 5) and S_{ni} the i^{th} search segment of S_n. The point with maximum gradient found along S_{ni} is noted P_{ni}.

4.3 Discussion

Pose and motion estimation is usually done with well defined primitives like points, lines and conics [30, 12].

From the precedent processing, a line could be fitted, through the detected points P_{ni} of a same segment S_n to determine the landmark side. The intersection of them would define the square corner in the image and we could apply points algorithm like in [10].

Yet, we have found formal line fitting and corner computation approaches not to be very robust under the presence of partial occlusions, fast motion and noise: Since the line must be extrapolated from a small number of points and since some points P_{ni} are discarded or poorly localized (because of occlusions or fast motions), the derived location of lines is often imprecise.

On the other hand, most of the originally detected points P_{ni} are well located on the target border. The idea of the following algorithm is to exploit directly those informations and not first derived new (uncertain) primitives from them. We are so statistically more robust because we work with the larger number of independent primitives.

4.4 Recovery of the motion parameters

To recover the motion, we have to define a constraint between the new detected points P_{ni} and the projected model in the image.

The position of the camera should be updated, so that the re-projected segment S_n lies on the landmarks border in the image. That means also that the distance of the points P_{ni} to this segment is null.

We express the distance constraint formally as follows: If U_n is an orthogonal unit vector of the segment S_n and M an arbitrary point of S_n, we have the equality:

$$U_n.(P_{ni} - M) = 0$$

Thus, we recover iteratively the position of the camera taking the previously determined parameters (3 rotation angles and the 3 translation components) as initial value and minimizing the objective function F:

$$F = \sum_{n=1}^{N} \sum_{i=1}^{S} (U_n.(P_{ni} - M))^2$$

with N being the number of segments in the model and S being the number of samples per segment S_n.

This criterion minimizes the orthogonal distance of the points P_{ni} to the corresponding model segment S_n.

4.5 Minimal number of samples

Camera pose is defined by 6 parameters [8]. Thus, pose can in principle be recovered from 3 points with each providing two constraints, one per coordinate axis. Yet, the points P_{ni} are defined along the segments S_{ni}. They are constrained only in the S_{ni} direction. Each point P_{ni} provides therefore only one constraint, so that the algorithm needs at least 6 points, not all collinear.

5 Robustness

We have explore several ways of improving the tracker to make it robust and flexible enough for real

and difficult applications. The critical issues are robustness to occlusion, integration of already existing scene lines and extension to hybrid tracking.

5.1 Partial occlusions and outliers

If marks are partially occluded, some of the points P_{ni} are not determined correctly. To limit their influence on the pose estimation process, we use robust statistic methods, such as the M-estimator [14, 16, 29].

In principle, we can also use outlier detection techniques. Yet, practical experience has shown that outlier detection is not advantageous at this stage: Since the precision of the camera calibration is inherently related to the location of the currently visible squares in the scene, new squares that become visible during head rotations may not fit the current camera model well. Declaring such misfits to be outliers means that they are excluded from influencing the camera model to adapt to scene information beyond the initial field of view.

5.2 Integration of scene lines

The presented algorithm is not only valuable for the designed targets but can be extended to arbitrary polyhedral objects.

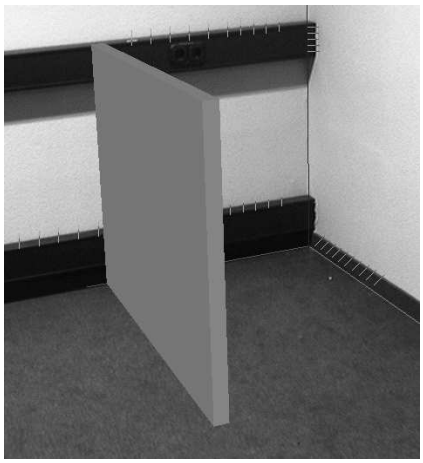


Figure 6: Camera tracking without landmarks

The advantage is twofold: First the user can go away from the targets after the initialization phase (see Figure 6). The camera is still track in 3D. Secondly it stabilizes the tracking when a few number of targets are present in the image.

5.3 Externally available information and hybrid tracking

For monitor-based AR, the camera is often placed on a tripod near the user (see Figure 7). Camera motion then is typically limited to rotations around its fixation point on the tripod. In other cases, other sensors like GPS for wide environment can determine the camera position.

Figure 7: Monitor-based AR: the camera translation is constant

In such scenarios, we estimate only the rotation parameters; the translation is considered constant or provided by the other sensors. The result is an appreciable reduction of computation time combined with increased robustness and stability. Use of such external knowledge is shown for an application on an exterior construction site in section 7.

6 System performance

Figure 8 shows a comparison between the new line based algorithm and a previous corner based approach (dashed line) (refer to section 4.3 and also [10]). Using a synthetic image sequence of 95 pictures, we calculate the relative errors of the camera position for the two algorithms.

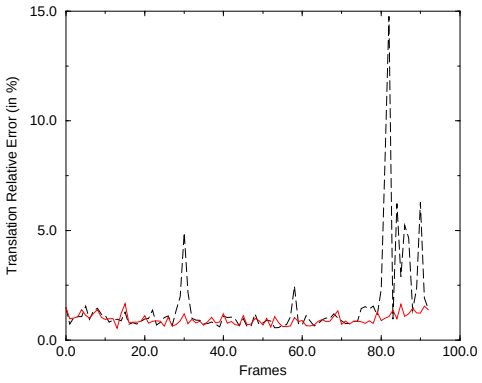


Figure 8: Relative error measurement

The spikes indicate situations when some targets were only partially in the field of view. They are not

taken into account by the precedent algorithm. The position estimation error grows so that a new initialization is necessary. The proposed algorithm maintains the relative error under 2.5 %.

Machine	Initialization	Tracking
O_2	5 Hz	23 Hz
Indy 5000	3 Hz	12 Hz

Table 1: Real time performance

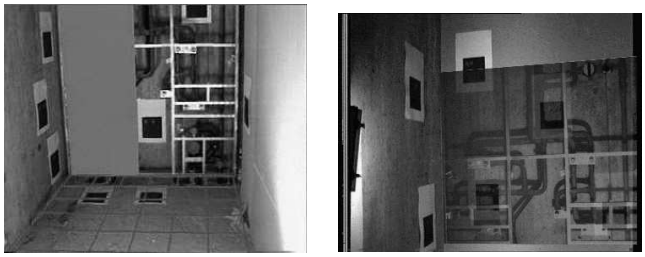
The real time performance of the above table have been measured with four landmarks in image video of size 768x576.

7 Applications

The following applications illustrate the potential of AR and the robustness of the presented system.

7.1 Exterior construction

In Figure 9(a), taken during the construction phase of a building, a virtual grid and a wall are added into the scene to show the next planned step. The original picture of Figure 9(b) was taken after construction of the wall. The user sees the water pipes "through" the wall using an "X-ray augmentation", thus being able to see their real position during a maintenance task.



(a) (b)

Figure 9: Construction planning and maintenance

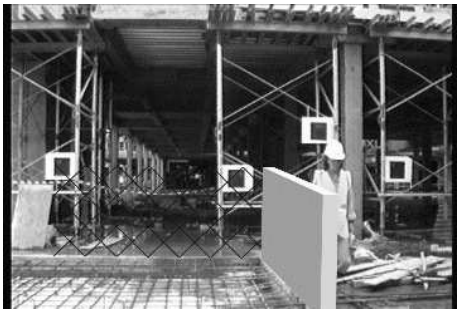


Figure 10: Hybrid tracking with GPS

In the next scenario, the position (translation) of the camera was determined by a differential GPS and the 3D position of the squares was measured with a laser pointer. The system is then able to recover correctly and reliably the camera rotation parameters during a panorama swing, augmenting images with a virtual wall.

7.2 Model presentation and physical interaction

Virtual models can be presented in 3D in the real environment using stereo rendering on a see through head mounted display (Figure 11).

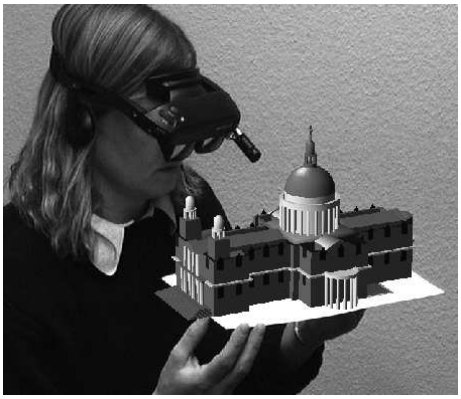


Figure 11: Physical manipulation of a virtual model

The virtual and the real object (the card board) are components of one entity, a *"mixed object"*. The manipulation is then very intuitive because the user interacts with a virtual object having a physical mean.

7.3 Game

The Tic Tac Toe game in Figure 12 explores different interaction schemes in an augmented environment. The user sets his stone and pushes a virtual "GO"-button.

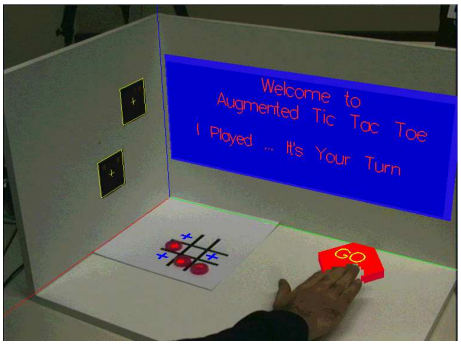


Figure 12: Interaction in an Augmented Environment

The computer analyzes the scene, localizes the stone (by color segmentation), places its own next virtual cross and instructs the user on a virtual panel to continue. All interactions occur directly in the real world - away from the keyboard.

If the camera is static, a background subtraction enables scene change detection, like for example moving hands of the Tic-Tac-Toe player in front of the camera. By initializing the Z-buffer, we can resolve the occlusion problem under this particular assumption; the moving real object are considered to be in foreground of the scene. We have experienced that the dynamic occlusion handling helps users strongly to understand and interact intuitively with the augmented scene. This application runs at about 8 Hz.

8 Summary and conclusions

In this article, we have presented a robust tracking algorithm developed to efficiently recover moderately fast camera motion. The system allows for experiments in AR with a hand-held or head-mounted camera. It does not require interactive support from the user and works on standard hardware.

Nevertheless, the camera motions are currently limited to registered areas with known landmarks, or predefined polyhedral objects. Further developments will integrate an automatic detection of new image features [19], which can be reconstructed in 3D and used for the tracking.

Acknowledgments

The laboratory space and the equipment is provided by the European Computer-industry Research Center (ECRC). The research is partially funded by the European CICC and CUMULI projects.

References

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

[2] R. Azuma and G. Bishop. Improving static and dynamic registration in an optical see-through HMD. In *Proc. Siggraph '94*, pages 197–204, Orlando, FL, July 1994.

[3] R. Azuma and G. Bishop. A frequency-domain analysis of head-motion prediction. In *Proc. Siggraph '95*, pages 401–408, Los Angeles, CA, August 1995.

[4] M. Bajura, H. Fuchs, and R. Ohbuchi. Merging virtual objects with the real world: Seeing ultrasound imagery with the patient. *Computer Graphics*, 26(2):203–210, July 1992.

[5] M. Bajura and U. Neumann. Dynamic registration correction in video-based augmented reality systems. *IEEE Computer Graphics and Applications*, 15(5):52–60, 1995.

[6] Bar-Shalom and Fortmann. *Tracking and Data Association*. Academic Press, 1988.

[7] W.E.L. Grimson, G.J. Ettinger, S.J White, P.L. Gleason, T. Lozano-Perez, W.M Wells III, and R. Kikinis. Evaluating and validating an automated registration system for enhanced reality visualization in surgery. In *Proc. CVRMed '95*, pages 3–12, Nice, France, April 1995.

[8] R.M. Haralick, C.N. Lee, K. Ottenberg, and M. Nolle. Review and analysis of solutions of the 3-point perspective pose estimation problem. *IJCV*, 13(3):331–356, December 1994.

[9] W.A. Hoff, K. Nguyen, and T. Lyon. Computer vision-based registration techniques for augmented reality. In *Proc. of Intelligent Robots and Computer Vision XV, SPIE Vol. 2904*, Boston, MA, Nov. 18–22 1996. SPIE.

[10] G. Klinker, D. Stricker, and D. Reiners. Augmented reality for exterior construction applications. In W. Barfield and T. Caudell, editors, *Augmented Reality and Wearable Computers*. Lawrence Erlbaum Press, 1998.

[11] D. Koller, G. Klinker, E. Rose, D. Breen, R. Whitaker, and M. Tuceryan. Automated camera calibration and 3d egomotion estimation for augmented reality applications. In *Proc. CAIP '97*, Kiel, Germany, September 1997.

[12] R. Kumar and A. R. Hanson. Robust methods for estimating pose and a sensitivity analysis. *CVGIP-IU*, 60(3), November 1994.

[13] D.G. Lowe. Robust model-based motion tracking through the integration of search and estimation. *IJCV*, 8(2):113–122, August 1992.

[14] P. Meer, D. Mintz, D.Y. Kim, and A. Rosenfeld. Robust regression methods for computer vision: A review. *International Journal of Computer Vision*, 6(1):59–70, April 1991.

[15] J.P. Mellor. Realtime camera calibration for enhanced reality visualization. In *Proc. of Computer Vision, Virtual Reality and Robotics in Medicine (CVRMed '95)*, pages 471–475, Nice, France, April 1995. IEEE.

[16] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, UK, 2 edition, 1992.

[17] S. Ravela, B. Draper, J. Lim, and R. Weiss. Adaptive tracking and model registration across distinct aspects. In *International Conference on Intelligent Robots and Systems*, Pittsburgh, PA, August 1995. IEEE.

[18] E. Rose, D. Breen, K.H. Ahlers, C. Crampton, M. Tuceryan, R. Whitaker, and D. Greer. Annotating real-world objects using augmented reality. In *Proc. Computer Graphics: Developments in Virtual Environments*. Academic Press Ltd, 1995.

[19] J. Shi and C. Tomasi. Good features to track. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 593–600, Los Alamitos, CA, USA, June 1994. IEEE Computer Society Press.

[20] T. Starner, S. Mann, B. Rhodes, J. Levine, J. Healey, D. Kirsch, R.W. Picard, and A. Pentland. Augmented reality through wearable computing. *Presence, Special Issue on Augmented Reality*, 6(4):386–398, August 1997.

[21] A. State, G. Hirota, D.T. Chen, B. Garrett, and M. Livingston. Superior augmented reality registration by integrating landmark tracking and magnetic tracking. In *Proc. Siggraph '96*, pages 429–438, New Orleans, August 1996.

[22] R.Y. Tsai. An efficient and accurate camera calibration technique for 3D machine vision. In *Proc. CVPR*, pages 364–374. IEEE, 1986.

[23] M. Tuceryan, D. Greer, R. Whitaker, D. Breen, C. Crampton, E. Rose, and K. Ahlers. Calibration requirements and procedures for a monitor-based augmented reality system. *IEEE Transactions on Visualization and Computer Graphics*, 1, September 1995.

[24] M. Uenohara and T. Kanade. Vision-based object registration for real-time image overlay. In *Proc. CVRMed '95*, pages 13–22, Nice, France, April 1995.

[25] A. Webster, S. Feiner, B. MacIntyre, W. Massie, and T. Krueger. Augmented reality in architectural construction, inspection, and renovation. In *Proc. ASCE Third Congress on Computing in Civil Engineering*, pages 913–919, Anaheim, CA, June 17-19 1996.

[26] J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *PAMI*, 14(10):965–980, 1992.

[27] J. Weng, T.S. Huang, and N. Ahuja. Motion and structure from image sequences, springer. *BERLIN*, 93:1992, 1992.

[28] Jun Park Youngkwan Cho and Ulrich Neumann. Fast color fiducial detection and dynamic workspace extension in video see-through self-tracking augmented reality. *Proceedings of the Fifth Pacific Conference on Computer Graphics and Applications*, 1997.

[29] Z. Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. *Image and Vision Computing Journal*, Vol.15, No.1:59–76, 1997.

[30] Z. Zhang and O.D. Faugeras. *3D Dynamic Scene Analysis*. Springer Verlag, Berlin Heidelberg New York, 1992.