Real-Time Camera Pose in a Room

Manmohan Krishna Chandraker, Christoph Stock, and Axel Pinz

Institute of Electrical Measurement and Measurement Signal Processing
Graz University of Technology, Austria
{manu, stock}@emt.tugraz.at, axel.pinz@tugraz.at
http://www.emt.tugraz.at/~tracking

Abstract. Many applications of computer vision require camera pose in real-time. We present a new, fully mobile, purely vision-based tracking system that works indoors in a prepared room, using artificial landmarks. The main contributions of the paper are: improved pose accuracy by subpixel corner localization, high frame rates by CMOS image aquisition of small subwindows, and a novel sparse 3D model of the room for a spatial target representation and selection scheme which gains robustness.

Keywords: Tracking, camera pose, CMOS camera, sparse 3D modeling

1 Introduction

Many applications of computer vision require camera pose in real-time. This is a hard problem which is known as 'real-time pose computation' in computer vision, or as 'vision-based inside-out tracking' in virtual and augmented reality. Existing vision-based systems suffer from several major deficiencies:

- motion blur, rapid change of field of view,
- occlusion, blocked line-of-sight,
- complexity of matching in a cluttered scene,
- multiple motion (ego-, target-, other objects),
- processing speed of the vision-based system in general.

Potential remedies are:

- hybrid systems (vision+inertial, GPS, magnetic, compass,...),
- active vision (laser pointing, control of mobile platform,...),
- *improved concepts for a computer vision system.*

We present a new, fully mobile, purely vision-based tracking system for indoor tracking applications in a prepared room with artificial landmarks. Only points are tracked (corner features). The main contributions of this paper are:

- accuracy (subpixel corner localisation),
- speed (CMOS camera, direct access of small subwindows), and
- representation of the scene in terms of landmarks (target selector).

1.1 Related Work

The problem of recovering camera pose from perspective n points (the PnP-problem) has been discussed as early as 1981 by Fischler and Bolles [6], but linear and real-time PnP algorithms are still an active research topic (e.g. [11, 1]). Vision-based ([9,2,5]), and hybrid sensor pose ([18,12,14,16]) are required in many application areas of virtual and augmented reality [13,4].

2 Tracking Methods

Figure 1 presents a schematic overview of our complete tracking system, which has been designed to work under varying conditions, indoors and outdoors, and with several kinds of sensors like CCD- and CMOS-cameras, and inertial trackers, which motivates the term 'hybrid tracker'. The modular system splits into five distinct blocks.
The tracker continuously delivers pose information (i.e. the sensor's pose w.r.t. the world coordinate system) to the pose client. To cut down the complexity of the correspondence problem a target selector module pre-selects features from the model database in a context-dependent way. Thus, only subsets of the model database are used during the tracking process once the initialization has been finished. As the pose client, the target selector requires the actual pose information to maintain the *active* set of model features. However, compared to the tracking loop, target selector runs take place at a considerably lower rate (about 1 Hz).

The hybrid tracker module has direct access to the sensor pool module which holds the array of sensors and their corresponding calibration data. Additionally, the operator can perform a number of maintenance and control tasks using the operator interface. Though the interfaces between adjacent modules in this design are not restricted to be of any distinct type, we prefer to use TCP/IP connections for data communication, which allows to easily split up the tracker to run on different CPUs. We subsequently describe all components of this system which are involved in the scenario (real-time camera pose in a room) and the experiments of this paper. Several other features of the hybrid tracker, like sensor fusion, are not discussed here.

2.1 Target Design

We want to use corners as our trackable features, so that the target design has to satisfy the following three requirements: High accuracy of the feature position; Uniqueness of targets; Fast correspondence verification. This leads us to the following considerations in target design:

Target Size: We found that for the dimensions of our laboratory (approx. $5 \times 6m$), a target should fit in roughly a $20cm \times 20cm$ area. Too large a target would mean that there are increased chances that fewer than the minimum number of targets (i.e. 2) are within the field of view for a certain camera pose. Equivalently,

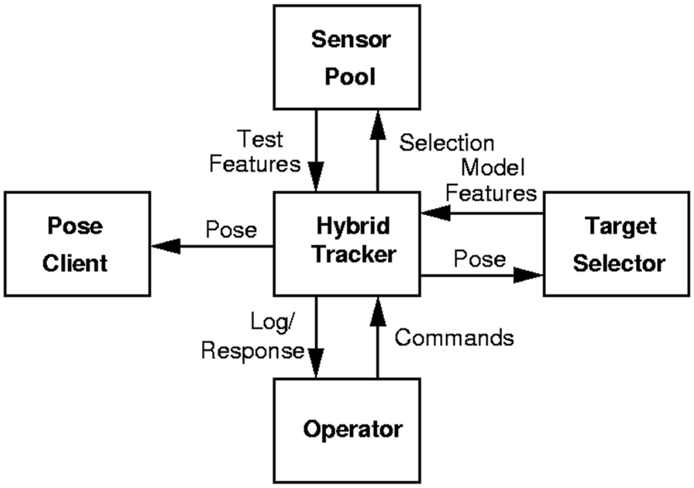


Fig. 1. Schematic outline of our hybrid tracking system.

it means that the minimum distance that we must set between wall and camera would be too large. On the other hand, too small a target would lead to corners on the same edge being too close, so that the corner detector would not be able to distinguish between them.

Target Shape: The basic shape of the target is as drawn in fig. 2. The idea is to have exactly 4 corners along each straight edge of the target which define a cross-ratio which is invariant under perspective projection. For the above mentioned target size constraints, not many unique cross-ratios (about 10) can be generated for accurate identification within limits of experimental error. In our target design, two cross-ratios are constrained to include a common corner, so even 10 cross-ratios can generate (10 choose 2) = 45 unique targets, which is sufficient for a representation of the room. In a recent paper, Naimark and Foxlin present an alternative approach based on circular landmarks with interior patterns for unique landmark identification [12].

2.2 Corner Detection and Subpixel Localisation

The basic procedure for extraction of corners is derived from the morphological scheme detailed in [10]. This approach satisfies our demands on fast corner extraction and high detection performance. A series of dilations and erosions by kernels of four different shapes - square, cross, plus, lozenge - results in a corner strength image.

The problem with such an approach is that a number of false corners are detected around, very close to, the true corner position. A heuristic procedure chooses only one corner out of this cluster, which is either used for pose calcu-

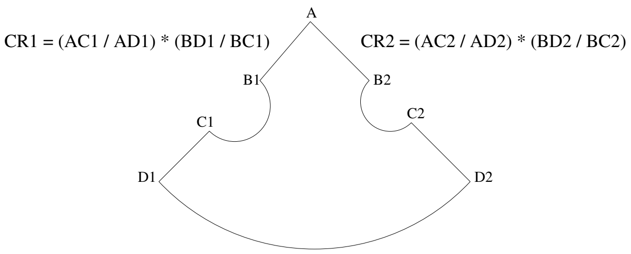

A
CR1 = (AC1 / AD1) * (BD1 / BC1)
CR2 = (AC2 / AD2) * (BD2 / BC2)
B1
B2
C1
C2
D1
D2

Fig. 2. The target design combines two different cross-ratios to identify three corners (A, D1 and D2).

lation or as the initial point for the sub-pixel accurate corner detector described in [17].

Random variations in light intensity, uneven cuts along the target edges and even faulty regions within the sensor array lead to detection of several corners besides the ones which are part of the targets. Complexity and error of the correspondence search algorithm increase drastically with increase in the number of detected corners. The way out is to simply count the ratio of black and white pixels on a pre-defined contour around each corner and to eliminate those corners which do not correspond to a ratio approximately between 60 and 120 degrees (the assumption is that only moderate projective distortions occur).

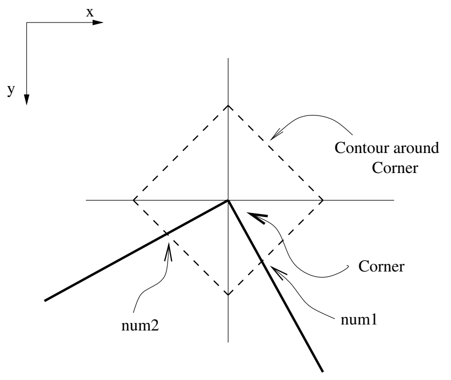
Sets of four corners on a straight line (referred to as quadruples) are formed and the cross-ratio is defined for each such quadruple. There might be quadruples which are formed due to corners from different targets and still have the same cross-ratio as a desired quadruple on the edge of a single target. Such quadruples are eliminated by utilizing certain orientational relationships between corners of the same edge as shown in fig. 3.

2.3 Corner Tracking with a CMOS Camera

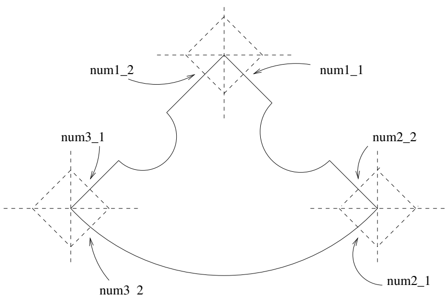
Due to the fact that corners are local features of an image, only small subsets of the whole image are necessary to compute the corner position. After an initialization process over the whole image, only small sub-images suffice to track the corner position from one frame to another.

True random access of pixels, in any image, at any time, requires full asynchronous reading of the sensor. In this case, fixed integration time concepts cannot be maintained. The pixels should therefore not be of the classic charge integrating type. Unlike CCD (Charge-Coupled Device) sensors, CMOS sensors use an active pixel structure type that yields an instantaneous signal that is proportional to the instantaneous light intensity and can be used to access small sub-regions of an image.

Active pixel structures allow random access, which provides high-speed imaging of a new dimension in combination with fast self-setting bandwidth control.



(a) A square contour of 10×10 pixels is drawn around the corner. It intersects the target at num1 and num2. The corner is classified according to the ratio $int((num2 - num1)/9)$ having the value 0, 1, 2 or 3.



(b) Depending upon the values of num1 and num2 for the first and fourth corner of a quadruple, we can straightaway eliminate those quadruples that are formed with corners from more than one target.

Fig. 3. Rejection scheme for (a) false corners, (b) false quadruples.

This means that the region of interest can be individually set by the user. Reducing the image field to marked subsets speeds up the image acquisition drastically. Frame rates of several hundred frames per second (fps) can be easily obtained, which is a big advantage over the currently established video frame rates of 30 to 60 fps.

2.4 Spatial Representation and Selection of Landmarks

The complexity of the tracking process can be reduced greatly by just concentrating on the targets that are within the field of view. The target selector (see fig.1) fulfills this function precisely. Moreover, it enables the isolation of the tracking process from the learning-based methods that could be employed in future context-dependent scenarios.

The target selector receives the current pose estimate from the tracker. Using this pose estimate, it tries to determine which of the targets in the target database is likely to be in the field of view (FoV), by determining whether the pre-specified center of gravity (CoG) of the target is within the FoV. If a target is visible, it proceeds to update the feature list for that target, that is, it sets the visibility flags for each of the features of that target according to whether that feature is within the FoV or not. This updated list of targets is then sent to the tracker.

The target selector computes the 2D-position (in pixels) of the *visible* features using the built-in camera model. Then small images are grabbed around these pixel values, the corner is extracted and the updated value is stored for grabbing in the next frame. If a prediction module is present, the update should also incorporate the motion estimate from the predictor. As of now, there is no prediction. All the extracted corners are used for a robust estimation of the camera pose, as described in the following section.

2.5 Pose Computation

Consistent with our objective of achieving real-time or greater tracking rates, we place greater emphasis on the speed of the pose estimation algorithm. An algorithm that combines concepts from projective geometry and an SVD solution to the absolute orientation problem in a RANSAC paradigm is employed to optimize between speed and robustness, as described below.

If ϵ be the proportion of outliers among a large data set of size S and N samples of size s chosen randomly, the probability p that at least one of the samples is free from outliers is given by

$$p = 1 - \left(1 - (1 - \epsilon)^s\right)^N \tag{1}$$

The algorithm begins with a worst case assumption for the proportion of outliers and this assumption is updated when a larger consensus set is found. A sample is deemed belonging to the consensus set if its distance from the proposed model, K, is less than a threshold, t. Instead of following the more theoretically correct approach of using the probability distribution of the distance of an inlier point (x, y) from the proposed model we use an empirical measure :

$$t = \sqrt{\frac{(x - K(x))^2 + (y - K(y))^2}{(x + K(x))^2 + (y + K(y))^2}}$$

Note that N is also updated to correspond to the best consensus set found so far. If the number of trials exceeds N or a pre-defined limit, the algorithm terminates. The pose corrsponding to the maximal consensus set is declared the correct pose. A sample size of 4 is sufficient to determine the camera pose from a data set of coplanar points. Next, we describe the actual pose computation algorithm.

Let $\{P\}$ be the set of image points and $\{Q\}$ be the set of corresponding object points. A purely geometric method described in [6] computes the translation vector from the origin, $\mathbf{t}$. As formulated in [11], the absolute orientation problem deals with estimation of the rotation matrix, $\mathbf{R}$ and the translation vector, $\mathbf{t}$ from a set of 3D-camera space co-ordinates, $\{q\}$ and the corresponding scene coordinates, $\{p\}$, where each such observation satisfies

$$\mathbf{q}_i = \mathbf{R}\mathbf{p}_i + \mathbf{t} \tag{2}$$

But we have already estimated $\mathbf{t}$ by an independent method. We use this known $\mathbf{t}$ to determine geometrically the camera space coordinates $\{q\}$. The scene geometry is depicted in figure 4. Let $\mathbf{O}$ be the origin of the camera space coordinates.

Let the point **a** be the image of point **A**. Since magnitude of **t** is known and position of **A** is also known (both in scene coordinates), the magnitude of the vector $\overline{OA}$ can be computed. Its direction is determined by the known x and y coordinates ($z = 1$ in normalized camera space) of pixel **a** once the corner has been extracted and the pixel size is known. Then the vector $\overline{OA}$ is precisely the vector **q** corresponding to point **A**.

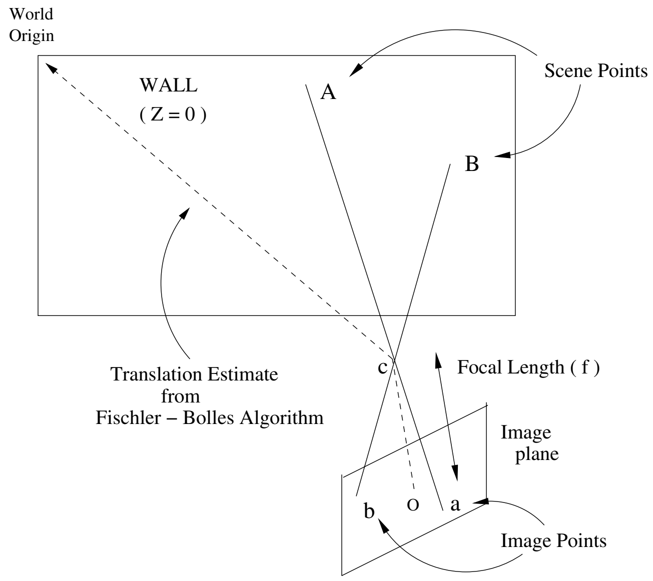


Fig. 4. Computation of **q** from known pixel values and camera translation.

Next, the centroids are computed :

$$\overline{\mathbf{p}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{p}_i \quad , \quad \overline{\mathbf{q}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{q}_i$$

Define

$$\mathbf{p'}_i = \mathbf{p}_i - \overline{\mathbf{p}} \quad , \quad \mathbf{q'}_i = \mathbf{q}_i - \overline{\mathbf{q}}$$

and

$$\mathbf{M} = \sum_{i=1}^{n}\mathbf{q'}_i\mathbf{p'}_i^{\,T} \tag{3}$$

Now, all the four points that we use are on the same plane Z = 0 in scene coordinates. Consequently, the last column of **M** is $[0, 0, 0]^T$ and **M** has a rank 2.

So we cannot use the simple SVD scheme and must use the approach outlined in [8]. This method would fail for $\mathbf{M}$ having rank 1, but such a situation arises only when the measurements are all collinear, in which case the absolute orientation problem can anyway be not solved. Any square matrix $\mathbf{M}$ can be decomposed into the product of an orthonormal matrix $\mathbf{U}$ and a positive semi-definite matrix $\mathbf{S}$. It is well-known that for a non-singular $\mathbf{M}$ having positive eigenvalues

$$\mathbf{S} = \left(\mathbf{M}^T\mathbf{M}\right)^{1/2} = \left(\frac{1}{\sqrt{\lambda_1}}\hat{\mathbf{u}}_1\hat{\mathbf{u}}_1^T + \frac{1}{\sqrt{\lambda_2}}\hat{\mathbf{u}}_2\hat{\mathbf{u}}_2^T + \frac{1}{\sqrt{\lambda_3}}\hat{\mathbf{u}}_3\hat{\mathbf{u}}_3^T\right)^{-1} \tag{4}$$

Here λ_1, λ_2, λ_3 are the eigenvalues of $\mathbf{M}$ and $\hat{\mathbf{u}}_1$, $\hat{\mathbf{u}}_2$, $\hat{\mathbf{u}}_3$ are the corresponding eigenvectors. $\mathbf{U}$ is easily determined as

$$\mathbf{U} = \mathbf{M}\mathbf{S}^{-1}$$

When $\mathbf{M}$ has a rank of only 2, we must use the pseudo-inverse, rather than the inverse, of $\mathbf{S}$. Thus,

$$\mathbf{S}^+ = \left(\frac{1}{\sqrt{\lambda_1}}\hat{\mathbf{u}}_1\hat{\mathbf{u}}_1^T + \frac{1}{\sqrt{\lambda_2}}\hat{\mathbf{u}}_2\hat{\mathbf{u}}_2^T\right)$$

If the Singular Value Decomposition of $\mathbf{MS}^+$ be given by $\mathbf{MS}^+ = U_0\Sigma_0V_0$ and $\hat{\mathbf{u}}_{03}$, $\hat{\mathbf{v}}_{03}$ be the third columns of U_0 and V_0, respectively, then we have

$$\mathbf{U} = \mathbf{MS}^+ \pm \hat{\mathbf{u}}_{03}\hat{\mathbf{v}}_{03}^T \tag{5}$$

The rotation matrix $\mathbf{R}$ is given by choosing the sign in the above expression which makes the determinant of $\mathbf{U}$ positive.

2.6 Reinitialization

The tracker reinitializes itself under two distinct conditions:

1. When there are not enough corners to compute the pose. This can happen if the camera is too close to the wall, so that the minimum number of targets (2, in our case) are not visible. It can also be the case that 2 or more targets are visible, but due to some lighting variations, all corners are not detected. Severe projective distortions can lead to quadruples being rejected as incorrect when the corner is extracted a little away from the true position and the resulting cross-ratio is beyond the tolerance limits.
2. When the difference between successive pose estimates is too large.

The reinitialization conditions must be judiciously set since too many reinitializations slow down the tracking process substantially.

3 Experiments and Results

3.1 Experimental Setup

Figure 5.a shows a simple experimental setup, which we have used for the experiments described below. One can see three of our targets. We use a Fuga 15d 512×512 monochrome CMOS Camera with a common PCI interface, mounted on a tripod. The camera is mounted on top of a pan-tilt unit that can be used for rotation experiments with 'ground truth'. Figure 5.b sketches a fully mobile application scenario - real-time camera pose is used for head-pose estimation in an augmented reality application. All necessary hardware is carried on a backpack (see [15] for more details).

(a) (b)

Fig. 5. Experimental setup: Stationary experiments with a CMOS camera, mounted on a pan-tilt head and our special target design (a). Real-time head-pose is required in mobile augmented reality applications (b).

3.2 Spatial Accuracy

The reliability of the pose computation is demonstrated by evaluating the pose error for a stationary camera. Table 1 shows three different poses w.r.t. planar targets, which were mounted on a wall (see Figure 5.a). Only four corners were needed to compute the pose of the camera. On the one hand the corners were extracted with the Standard-Plessey operator [7] and on the other hand with a new model-based subpixel corner-detector [17]. Due to the fact that the maximal pose error occurs in depth, Table 1 shows only accuracy-results in $z-$direction (depth). View-Point 1 (VP1) indicates a viewing direction perpendicular to the

Table 1. Pose Accuracy for a stationary camera, for three different view points.

		Absolute Pose-Error in m	Standard Deviation	Max. Jitter
VP1	Plessey	0.07	0.0029	0.0715
	Sub-Pixel	0.01	0.008	0.0427
VP2	Plessey	0.44	0.0274	0.2428
	Sub-Pixel	0.034	0.019	0.0919
VP3	Plessey	0.0935	0.0258	0.1258
	Sub-Pixel	0.0177	0.0119	0.0655

Table 2. Run-time Behavior depending on Window-size,
Total Time = Time for Initialization (0.88 sec) + 100 Tracker Runs.

Window Size	Total time (sec)	Time for 1 tracker run (msec)	Tracking Rate
31×31	2.96 sec	$(2.96 - 0.88)/100 = 20.8$ msec	48.0 Hz
11×11	1.43 sec	$(1.43 - 0.88)/100 = 5.5$ msec	181.8 Hz

Table 3. Tracking Rates depending on Window-size and Number of tracked corners.

Number of corners	Tracking Rate(in fps) for Window Size		
	11×11	21×21	31×31
1	1111	526	294
2	417	244	145
3	286	164	97
4	200	120	71
10	79	48	28

wall, the other two entries (VP2 and VP3) depict positions with arbitrarily chosen angles.

Although in the absence of motion prediction it is not possible to quantify the speed of camera motion without making a gross underestimation, we moved the camera along different trajectories and each time noted that the pose error was within acceptable limits. These experiments also give a glimpse of the reinitialization process within the target selector.

3.3 Tracking Speed

The obtained experimental data justifies that grabbing small windows with a CMOS sensor only around the points of interest (which is not possible with a traditional CCD sensor) can dramatically speed up the tracking process. The improvement in frame rates over contemporary trackers' 30-50 Hz is substantial.

Table 2 shows the time-behavior of our algorithm for the initialization process, followed by 100 tracker runs. The time needed for the initialization process is 0.88 sec.

The utility of the Target Selector lies, among others, in the fact that it allows us to get reliable pose estimates without needing to keep track of *all* the corners. It was found that grabbing and corner extraction were the slowest components of the entire algorithm. So, we demonstrate the time behavior of this component of the algorithm across different number of corners and different window-sizes (cf. table 3).

The improvement in frame rate with decrease in window-size is obvious and remarkably high. But a smaller window-size means greater chances of the corner getting lost due to abrupt motion. A good prediction module can overcome this problem.

3.4 Target Selector Functionality

Experiments were conducted that showed the independence of the target selector and the tracker processes. Projective distortions or poor lighting can lead to inaccurate cross-ratio computation at the initialization step. Our target selector is robust to such inaccuracies, as long as the minimum number of 4 corners are identifiable for pose computation. Based on the current pose, the targets within the field of view are reported to be visible after the target selector is invoked and these targets are then tracked.

4 Discussion and Outlook

We have presented a new system for vision-based pose determination. We achieve real-time performance and reasonable pose accuracy by subpixel corner detection and CMOS camera handling combined with a specific target design and target selector.

Our experiences with real-time tracking systems indicate clearly that the overall performance of a complex tracking system depends on many individual system components and their smooth interaction, so that it makes sense to design specific experimental setups and to study *isolated* problems in detail (e.g. the subpixel corner detector [17], or the complexity of correspondence search [3]).

Several future improvements of our system are on the way. We are developing a motion prediction module to allow fast camera motions, especially rotations. Although we have demonstrated the feasibility of a purely vision-based approach, robustness will be improved by using complementary sensors, as was shown for our hybrid outdoor system [15]. Whenever the system loses track, a rather time-consuming bootstrap (re-initialization) of the tracker is required. Dynamic interpretation tree as presented in [3] will be used to cut down complexity. Several other tracking systems use prepared environments and track artificial blob-like features (e.g. [12]). We hope that for our approach (tracking of corners), it will be easier to adapt to unprepared 'natural' scenes.

Acknowledgements. This research was supported by the following projects: VAMPIRE - Visual Active Memory Processes and Interactive REtrieval (EU-IST Programme - IST-2001-34401), and MCAR — Mobile Collaborative Augmented Reality (Austrian 'Fonds zur Förderung der wissenschaftlichen Forschung' project number P14470-INF).

References

1. Adnan Ansar and Kostas Daniilidis. Linear pose estimation from points or lines. In A. Heyden et al., editor, *Proc. ECCV*, volume 4, pages 282–296, Copenhagen, Denmark, May 2002. Springer LNCS 2353.
2. Mark Billinghurst, Hirokazu Kato, and Ivan Poupyrev. The MagicBook: a transitional AR interface. *Computers and Graphics*, 25:745–753, 2001.
3. M. Brandner and A. Pinz. Real-time tracking of complex objects using dynamic interpretation tree. In Luc Van Gool, editor, *Pattern Recognition, Proc. 24th DAGM Symposium, Zurich*, volume LNCS 2449 of *Schriftenreihe*, pages 9–16. Springer, 2002.
4. S.K. Feiner. Augmented reality: A new way of seeing. *Scientific American*, 4, 2002.
5. Vittorio Ferrari, Tinne Tuytelaars, and Luc Van Gool. Real-time affine region tracking and coplanar grouping. In *Proc. CVPR*, volume 2, pages 226–233, Kauai, Hawaii, USA, December 2001.
6. M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981.
7. C. Harris and M. Stephens. A combined corner and edge detector. In *Proc. of the 4th Alvey Vision Conference*, pages 189–192, Manchester, 1988.
8. B.K.P. Horn, H.M. Hilden, and S. Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *J.Opt.Soc.Am.A*, 5:1127–1135, 1988.
9. Dieter Koller, Gudrun Klinker, Eric Rose, David Breen, Ross Whitaker, and Mihran Tuceryan. Real-time vision-based camera tracking for augmented reality applications. In *Proc. CVPR*, 1997.
10. R. Laganière. Morphological corner detection. In *Proc. ICCV*, pages 280–285, Bombay, 1998.
11. C.P. Lu, G.D. Hager, and E. Mjolsness. Fast and globally convergent pose estimation from video images. *IEEE Trans. PAMI*, 22(6):610–622, June 2000.
12. L. Naimark and E. Foxlin. Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In *Proceedings ISMAR 2002, Darmstadt*, pages 27–36. IEEE Comp Soc, 2002.
13. U. Neumann. STAR: Tracking for object-centric augmented reality. In W. Barfield and T. Caudell, editors, *Fundamentals of Wearable Computers and Augmented Reality*, chapter 10. Lawrence Erlbaum Associates, 2001.
14. U. Neumann, S. You, Y. Cho, J. Lee, and J. Park. Augmented reality tracking in natural environment. In *International Symposium on Mixed Realities*, Tokyo, Japan, 1999.
15. M. Ribo, H. Ganster, M. Brandner, P. Lang, Ch. Stock, and A. Pinz. Hybrid tracking for outdoor AR applications. *IEEE Computer Graphics and Applications Magazine*, 22(6):54–63, 2002.
16. Kiyohide Satoh, Mahoro Anabuki, Hiroyuki Yamamoto, and Hideyuki Tamura. A hybrid registration method for outdoor augmented reality. In *Proc. ISAR*, pages 67–76, 2001.

17. C. Stock, U. Mühlmann, M. Chandraker, and A. Pinz. Subpixel corner detection for tracking applications using cmos camera technology. In *26th Workshop of the AAPR/ÖAGM*, volume 160, pages 191–199, Graz, 2002. OCG Schriftenreihe.
18. S. You, U. Neumann, and R.T. Azuma. Hybrid inertial and vision traking for augmented reality registration. In *IEEE Conference on Virtual Reality*, pages 260–267, Houston, USA, March 1999.