Markerless Tracking using Planar Structures in the Scene

Gilles Simon, Andrew W. Fitzgibbon and Andrew Zisserman
{gs,awf,az}@robots.ox.ac.uk
Robotics Research Group,
Department of Engineering Science,
University of Oxford,
19 Parks Road, Oxford OX1 3PJ, United Kingdom
http://www.robots.ox.ac.uk/~vgg

Abstract

We describe a markerless camera tracking system for augmented reality that operates in environments which contain one or more planes. This is a common special case, which we show significantly simplifies tracking. The result is a practical, reliable, vision-based tracker. Furthermore, the tracked plane imposes a natural reference frame, so that the alignment of the real and virtual coordinate systems is rather simpler than would be the case with a general structure-and-motion system. Multiple planes can be tracked, and additional data such as 2D point tracks are easily incorporated.

1 Introduction

To effectively implement augmented reality in unstructured environments—either in wearable head-mounted displays, or when augmenting archive footage (e.g. for special effects)—the primary requirement is accurate, reliable, fast position tracking. Only optical technologies appear to offer the required accuracy, but unstructured environments do not permit the placement of targets [31] or the preparation of 3D models of the environment *a-priori*. Archive footage also presents special difficulties in augmented reality. Generally, camera calibration is unavailable, or at best inaccurate. Markers are not available, and 3D measurements are not easily obtained for points visible in the provided images. On the other hand, post-hoc augmentation of archive footage is often required, for example in architectural visualisation, in post-production, or even when pre-shoot planning is found to be in error.

Although movematching techniques from photogrammetry and computer vision may be used to compute the camera motion, such techniques are currently difficult to use and extremely time-consuming, while automatic methods [1, 3, 12, 21] are not yet widely available, and are far from being real-time. Even if an automatic, general-purpose movematcher were available, there remains the non-trivial task of aligning the system's arbitrarily chosen coordinate frame with that of the augmenting objects.

In this paper, we describe a vision-based position tracker which simplifies the general camera-tracking problem in the case where there is a planar surface visible somewhere in the scene. Some examples are shown in figure 1. Because this is a special case of the general problem, it is easier to solve, and therefore allows more reliable and faster solutions. However, it is also a very common special case; the ground plane, or a wall is often visible throughout the scene; and indoors, the ceiling is often readily tracked.

We show that the system performs as well as general-motion trackers, but is more reliable and faster. An example implementation is presented, but the basic technology of automatic homography tracking is well established in the broadcast industry so realtime solutions to the plane tracking problem already exist.

2 Background

Before describing our plane-specific tracker, we review the current strategies for markerless AR. These may be divided into two main categories: model-based tracking and move-matching. All strategies depend on making correspondence between 3D world features and 2D image features, and ideally, on making such correspondence automatic.

2.1 Model-based tracking

The most common approach for dealing with unstructured environments is to impose some structure *post-hoc*.

Figure 1. Example scenes with planar elements. Our algorithm automatically tracks the plane and recovers camera position. Accuracy is at or near the level of a full-scene structure and motion solution. In addition, the plane provides a natural coordinate system for augmentation.

By identifying features in the images for which real-world coordinates can be measured, a correspondence between 3D and 2D is set up. Examples include street lamps and 3D curves in [5, 27] and the horizon in [4]. Pose-estimation techniques [8, 10] can then be used to estimate the camera position. Further model-based systems are described in [18, 19, 26, 29, 30]. We now describe point-based pose estimation in order to situate our work and introduce our notation.

For each new video frame, we are given a set of 3D points $\mathbf{X}_i$ whose coordinates are known, and a corresponding set of 2D points $\mathbf{x}_i$. We represent all quantities in homogeneous coordinates, so $\mathbf{x}_i$ is a 3-vector $(x_i, y_i, 1)$, and $\mathbf{X}_i = (\mathrm{X}_i, \mathrm{Y}_i, \mathrm{Z}_i, 1)$. The camera position is represented as a 3×4 projection matrix

$$\mathsf{P} = \mathsf{K}\,[\mathsf{R} \mid \mathbf{t}]$$

where R is a 3×3 rotation matrix, and $\mathbf{t}$ is the translation of the camera. The matrix K represents the internal calibration parameters of the camera:

$$\mathsf{K} = \begin{bmatrix} f & s & u_0 \\ 0 & af & v_0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \begin{cases} f \text{ is focal length;} \\ (u_0, v_0) \text{ is principal point;} \\ a \text{ is aspect ratio; } s \text{ is skew} \end{cases}$$

In most cases, the internal parameters may be assumed known, but sometimes (e.g. when dealing with archive footage) they must be estimated from the data. The pose estimation task is to compute P, given the 2D-3D correspondences

$$\mathbf{x}_i = \lambda_i \mathsf{P} \mathbf{X}_i$$

the homogeneous scale factors λ_i are unknown *a-priori*, so must be estimated simultaneously with R and $\mathbf{t}$. The strategy is to first estimate P using a linear method, and then nonlinearly minimize the *reprojection error*

$$\epsilon(\mathsf{R}, \mathbf{t}) = \sum_i d^2(\mathbf{x}_i, \mathsf{P}\mathbf{X}_i)$$

where

$$d^2\left(\begin{pmatrix} x_1 \\ y_1 \\ w_1 \end{pmatrix}, \begin{pmatrix} x_2 \\ y_2 \\ w_2 \end{pmatrix}\right) = (\frac{x_1}{w_1} - \frac{x_2}{w_2})^2 + (\frac{y_1}{w_1} - \frac{y_2}{w_2})^2$$

For a moderate number of points, this process can be made to run quickly, requiring of the order of 50ms per frame. The primary advantage of model-based tracking is high precision, and absence of drift: as long as (a subset of) the key points are visible, the system is always registering directly to the scene. Furthermore, the coordinate system is well defined beforehand. However, it is commonly true that few points are available for registration, so the tracking suffers from high-frequency jitter, as noise in the image measurements corrupt the estimated pose. More importantly, the technique requires significant manual intervention to construct the model. In many real-world cases, a 3D model is difficult to obtain, or measurable features may not be easily detectable.

2.2 Move-matching

A technology which appears to offer significant possibilities for general, accurate registration is known as structure-and-motion estimation, or move-matching [1, 2, 3, 12, 24]. Such systems simultaneously estimate camera motion, and the 3D structure of the imaged scene. These systems permit extremely accurate registration, with accuracies of around 0.2 pixels, and negligible jitter. However, they have a number of disadvantages that mean that they are not likely to be suitable for real-time implementation in the near future.

First, they are slow, and high accuracy is only achieved by a batch bundle adjustment [12]. This precludes a sequential implementation, as is required for autonomous, or long-running applications. Second, the coordinate system chosen will be arbitrary, generally aligning the world coordinate system with the first camera. Therefore, the computed motion is relative rather than absolute, which means that in order to insert virtual objects the systems must be manually aligned. This is difficult unless some feature correspondences are made between the recovered 3D structure and the augmenting models.

Another algorithm for structure-and-motion based augmented reality tracking was reported by Neumann *et al.* [24]. In their system, optical flow is used to compute differential motion estimates, which are integrated to give camera pose. Our system might be considered a special case of theirs—we compute differential motion estimates for the plane, and then use these estimates to compute the camera motion. However, our system does not assume that the inter-frame motion is small, and can therefore be used in situations where a fast-moving camera would make optic flow difficult to compute.

2.3 Special-case techniques

Some interesting recent work looks at improving AR using special geometric constructions that are found in the scene [20, 28]. In these works, constraints are obtained from structures typically found in man-made environments. For example, sets of parallel or orthogonal lines can be used to determine the 3D reference frame, and compute or refine tracking.

3 Planar-surface tracking

In this paper, we consider an approach which borrows something from each of the above. We describe what is essentially an automatic move-matcher for scenes which contain planar structures. We are thus, like [20], using a limited type of scene. In our case, however, the limitation is slight—we require that a plane or planes be visible in the scene. This is commonly true of indoor environments, where a textured ceiling or ground plane is visible. Outdoors, even rough ground (viz. the grass in Figure 1), provides a good reference for the system. The same plane need not be visible throughout the sequence, as the algorithm can "hand off" tracking from one plane to the next.

Like move-matching, the computation of relative motion from frame to frame is completely automatic, but the proposed approach is significantly more reliable, for reasons discussed in §4.4. Tracking of a plane confers another advantage. A canonical coordinate system is automatically created within the scene, which greatly simplifies the alignment of the system's automatically chosen reference frame and any frame attached to an augmenting model. A few mouse clicks are enough to set the coordinate frame, and it may be changed at any time without restarting the algorithm.

Initialization:

1. Manually indicate the planar region in image 0.
2. Detect interest points in image 0.
3. Initialize camera calibration K.

Steady state, computing H from frame i to $i+1$.

1. Detect N interest points in frame $i+1$, giving the set $\{\mathbf{x}_j^{i+1}\}_{j=1}^N$.
2. Match interest points from frame i to $i+1$. This generates a set of correspondences $\mathbf{x}_j^i \leftrightarrow \mathbf{x}_k^{i+1}$.
3. From the set of correspondences, robustly compute H_i^{i+1} (§4.1).
4. Compute pose from H_i^{i+1} (§3.3).

Figure 2. Algorithm summary.

An overview of the algorithm is shown in Figure 2. The basic primitives used are interest points [14, 23], which are automatically computed for each input image. These interest point operators have the desirable property that the 2D points they generate often correspond to 3D points in the world. By obtaining multiple images of a 3D point, we glean information about the structure of the world, and about the motion of the camera. The transformation which models the 2D movement of coplanar points under perspective projection is given by a 3×3 planar homography. From the planar homography, we can easily compute the camera position and rotation, which provides the motion estimates.

In the following, we first develop the mathematical model used by the system, and then describe our implementation. Finally we comment on reliability in comparison with general motion estimation, and with respect to the well-known singularities of pose estimation from coplanar points.

3.1 Multiple views of a plane

For most of this paper, we may choose coordinates so that we are tracking the $\mathrm{Z} = 0$ plane. Therefore, a 3D point on the plane has the form $\mathbf{X} = (\mathrm{X}, \mathrm{Y}, 0, 1)$, and is defined by just two coordinates X and Y. It is projected into image

i via the 3×4 projection matrix P^i, yielding the measured 2D point $\mathbf{x}^i$

$$\begin{aligned}
\mathbf{x}^i &= \mathsf{P}^i \mathbf{X} \\
&= \mathsf{K}\left[\mathbf{r}_1^i \mathbf{r}_2^i \mathbf{r}_3^i \mathbf{t}^i\right] \begin{pmatrix} \mathrm{X} \\ \mathrm{Y} \\ 0 \\ 1 \end{pmatrix} \\
&= \mathsf{K}\left[\mathbf{r}_1^i \mathbf{r}_2^i \mathbf{t}^i\right] \begin{pmatrix} \mathrm{X} \\ \mathrm{Y} \\ 1 \end{pmatrix} \\
&= \mathsf{H}_w^i \begin{pmatrix} \mathrm{X} \\ \mathrm{Y} \\ 1 \end{pmatrix}
\end{aligned}$$

where the 3×3 matrix H is a planar homography[1] which transforms points on the world plane to the ith image plane. The subscript w refers to the world coordinate system. There is a one-to-one correspondence between the two planes, so a point $(x, y, 1)$ on the image can be back-projected onto the world plane by multiplying it by $(\mathsf{H}_w^i)^{-1} = \mathsf{H}_i^w$. Now, in our case, the world coordinate system is not known, so we cannot measure H_w^i directly. However, we *can* measure H_i^{i+1}, the homography that maps the images of points on the plane from frame i to frame $i+1$. This matrix is given by

$$\mathsf{H}_i^{i+1} = \mathsf{H}_w^{i+1} (\mathsf{H}_w^i)^{-1}$$

Now, H_i^{i+1} relates image coordinates of features that we can measure. Indeed, given four or more corresponding points, we can compute H_i^{i+1} from image data alone.

3.2 Computing H_i^{i+1}

Specifically, if $\mathbf{x}^i = (x, y, 1)$ and $\mathbf{x}^{i+1} = (x', y', 1)$ are images of the same 3D point on the plane, their coordinates are related by

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \lambda \mathsf{H}_i^{i+1} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

where λ is the unknown homogeneous scale factor. Now, cross-multiplying the third components of each side and writing the components of the homography as h_{11} etc, we obtain the linear system of two equations for the elements of H_i^{i+1}

$$\begin{aligned}
x'\,(h_{31}x + h_{32}y + h_{33}) &= h_{11}x + h_{12}y + h_{13} \\
y'\,(h_{31}x + h_{32}y + h_{33}) &= h_{21}x + h_{22}y + h_{23}
\end{aligned}$$

Given four such correspondences, we obtain a matrix equation of the form $\mathsf{A}\mathbf{h} = \mathbf{0}$, where $\mathbf{h}$ is just the components

[1] Also known as collineation, or plane perspective transformation.

$$\begin{pmatrix}
x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1'x_1 & -x_1'y_1 & -x_1' \\
0 & 0 & 0 & x_1 & y_1 & 1 & -y_1'x_1 & -y_1'y_1 & -y_1' \\
x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2'x_2 & -x_2'y_2 & -x_2' \\
0 & 0 & 0 & x_2 & y_2 & 1 & -y_2'x_2 & -y_2'y_2 & -y_2' \\
x_3 & y_3 & 1 & 0 & 0 & 0 & -x_3'x_3 & -x_3'y_3 & -x_3' \\
0 & 0 & 0 & x_3 & y_3 & 1 & -y_3'x_3 & -y_3'y_3 & -y_3' \\
x_4 & y_4 & 1 & 0 & 0 & 0 & -x_4'x_4 & -x_4'y_4 & -x_4' \\
0 & 0 & 0 & x_4 & y_4 & 1 & -y_4'x_4 & -y_4'y_4 & -y_4'
\end{pmatrix}$$

Figure 3. Components of the matrix A.

h_{ij} stacked into a 9-element vector. The matrix A is shown in figure 3. The solution $\mathbf{h}$ is the null space of A, which may be computed using the singular value decomposition [13]. More details, in particular on data normalization, are given in [16].

3.3 Computing P^i

Suppose for a moment that we knew the mapping from world coordinates to one of the frames of the sequence, say frame 0. This mapping is the homography H_w^0. We can measure H_k^{k+1} for any pair of frames $(k, k+1)$ from tracked points, and therefore we can compute

$$\mathsf{H}_0^i = \mathsf{H}_{i-1}^i \mathsf{H}_{i-2}^{i-1} \ldots \mathsf{H}_0^1$$

Finally, using the known value of H_w^0, we obtain

$$\mathsf{H}_w^i = \mathsf{H}_0^i \mathsf{H}_w^0$$

Now recall that $\mathsf{H}_w^i = \mathsf{K}\left[\mathbf{r}_1^i \mathbf{r}_2^i \mathbf{t}^i\right]$ If the calibration (i.e. K) is known (see §3.5), then we can easily extract $\mathbf{r}_1^i$ and $\mathbf{r}_2^i$ from the first two columns of $\mathsf{K}^{-1}\mathsf{H}_w^i$. Then, because R must be a rotation matrix, we know its columns must be orthonormal, so $\mathbf{r}_3$ is given by the cross product $\mathbf{r}_1 \times \mathbf{r}_2$. Therefore we can write

$$\mathsf{P}^i = \mathsf{K}\left[\mathbf{r}_1^i \;\; \mathbf{r}_2^i \;\; (\mathbf{r}_1^i \times \mathbf{r}_2^i) \;\; \mathbf{t}^i\right]$$

Thus, if we know H_w^0, we can compute camera positions for every frame of the sequence, using only the frame-to-frame homographies H_i^{i+1}

3.4 Computing H_w^0: Aligning the real and virtual coordinate systems

In order to correctly align the real and virtual coordinate systems, we need to know the mapping between one image plane and the world plane. This mapping is specified by the planar homography H_w^0, and may be divided into "metric" and "projective" components. The metric component refers

to the arbitrary choice of Euclidean coordinates in the plane, and the overall scale, which can never be determined from the images alone. The projective component concerns the orthogonality of the coordinate axes. We will describe two ways of aligning the reference frames.

The simplest way of setting the frame is to use the mouse to select 4 points on a rectangle in the scene [17]. The four points are then assigned the world coordinates of the rectangle $(0,0)$, $(1,0)$, $(1,s)$, $(0,s)$, where s is the (unknown) aspect ratio of the world rectangle. For the moment, assume $s = 1$. Writing these points as (x_k^w, y_k^w), and with the corresponding mouse-selected coordinates denoted (x'_k, y'_k), we may use the method described above to compute the homography H which relates these two sets of points, i.e. $\mathbf{x}'_k = \lambda_k \mathsf{H}\mathbf{x}_k^w$. Now, because we know K, and observing that the effect of non-unit s is to premultiply the world coordinates $(\mathrm{X}, \mathrm{Y}, 1)$ by the diagonal matrix $\mathsf{D} = \mathrm{diag}(1, s, 1)$, we have

$$\begin{aligned} \mathsf{H} = \mathsf{H}_w^0 \mathsf{D} &= \mathsf{K}\,[\mathbf{r}_1 \;\; \mathbf{r}_2 \;\; \mathbf{t}] \begin{pmatrix} 1 & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{pmatrix} \\ \mathsf{K}^{-1}\mathsf{H} &= [\mathbf{r}_1 \;\; s\mathbf{r}_2 \;\; \mathbf{t}] \end{aligned}$$

Therefore s is given by the ratio of the lengths of the first two columns $\mathbf{h}_{1,2}$ of $\mathsf{K}^{-1}\mathsf{H}$. Thus, the algorithm is:

1. Compute H which maps the unit *square* to (x'_k, y'_k).
2. Compute $s = \frac{\|\mathbf{h}_2\|}{\|\mathbf{h}_1\|}$.
3. Compute $\mathsf{H}_w^0 = \mathsf{H}\mathsf{D}^{-1} = \mathsf{H}\,\mathrm{diag}(1, \frac{1}{s}, 1)$.

However, it is inconvenient to specify four points in order to fix the coordinate frame. The minimum possible is two – one to set the origin, and the second to indicate the direction of, say, the X axis and the overall scene scale. This can be done in a single mouse gesture, where the button down indicates the origin, and the button release is at a point defined to be $(1, 0)$ on the world plane.

3.5 Computing K

If the intrinsic parameters of the camera are unknown, they can be approximated by using two sets of parallel lines [6, 22] (e.g. the rectangle in §3.4) and fixing skew to 0, aspect ratio to 1 and principal point to the center of the image. We now show how to compute the focal length. Let $\mathbf{v} = (x_v, y_v, 1)$ and $\mathbf{w} = (x_w, y_w, 1)$ be the vanishing points of the sets of lines. As the directions $\mathsf{K}^{-1}\mathbf{v}$ and $\mathsf{K}^{-1}\mathbf{w}$ are orthogonal, we have $\mathbf{v}^\mathsf{T}\mathsf{K}^{-\mathsf{T}}\mathsf{K}^{-1}\mathbf{w} = 0$, that is

$$\begin{pmatrix} x_v & y_v & 1 \end{pmatrix} \begin{pmatrix} 1/f^2 & 0 & 0 \\ 0 & 1/f^2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_w \\ y_w \\ 1 \end{pmatrix} = 0$$

which provides

$$x_v x_w + y_v y_w + f^2 = 0$$

and hence f^2.

3.6 "Hand-off"

The previous discussion assumes that the same plane is being tracked throughout the sequence. While this may be feasible in some situations, in many environments it is a serious limitation. However, it is easy to switch tracking from one plane to another at any point, providing both planes are seen simultaneously in a minimum of two frames. This "hand-off" procedure is relatively simple. Imagine, for concreteness, that we are tracking the ground plane in the plaza sequence. After some time, the front wall of the building proves a more reliable choice (see §4.3). To automatically commence tracking using the new plane, we only need to transfer the coordinate-system alignment.

This is easily done by randomly choosing three points on the new plane, which are tracked over two views, i and j, say. Call the points and their correspondents $\mathbf{x}_k^i$ and $\mathbf{x}_k^j$, for $k = 1 \ldots 3$. Because the running (ground-plane) tracker is providing P matrices, we have P^i and P^j. Then we can use standard stereo triangulation [16] to compute the 3D world coordinates $\mathbf{X}_k$. Because we have world coordinates of $\mathbf{X}_k$, we can readily compute $\mathsf{H}_{\text{new}}^0$, and continue tracking using the new plane. Note that i and j are not necessary adjacent. In fact, i and j should be far apart, say the first and last frames in which both planes were seen.

3.7 Incorporating off-plane tracks

Another improvement to the system is to allow individual points off the plane to be tracked, and included in the motion estimate. Currently we implement this by using the plane-computed P matrices to initialize a bundle adjustment [7, 11]. In Figure 6 the cameras were improved by incorporating some two-view correspondences from the object silhouette.

4 Implementation

The previous section has discussed the theory of plane-based tracking. We now provide some details of our implementation. Unsurprisingly, a reliable image-based plane tracker is the most important component of the system. Luckily, plane tracking is a relatively easy problem, and indeed commercial systems are available which provide real-time homographies for live broadcast [25].

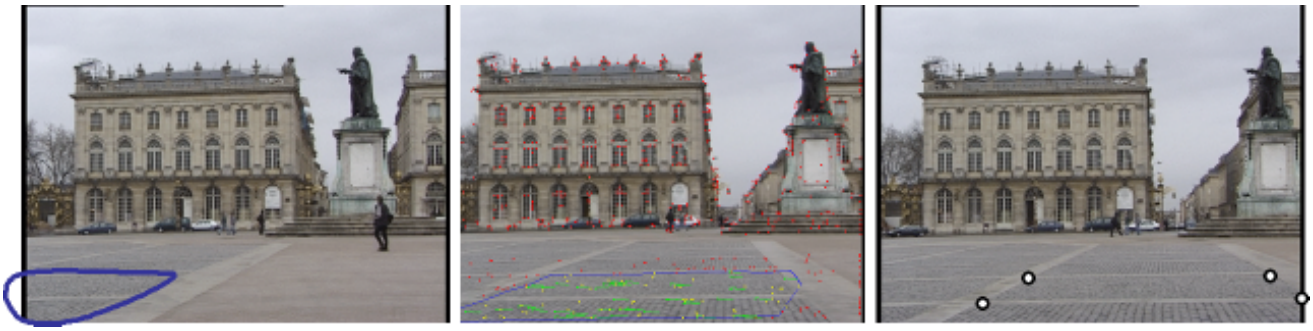

Figure 4. Steps in the algorithm. (1) The plane to be tracked is roughly indicated with the mouse, with the camera stationary. (2) Automatic feature detection and tracking. (3) Setting the coordinate system by indicating 4 points on a rectangle. At no point after (1) need the camera be stationary, nor tracking interrupted.

4.1 Robust computation of H_i^{i+1}

Our system is based on that described in [16], using the successful RANSAC paradigm. Consider two images, with points in the first labelled $\mathbf{x}$, and those in the second $\mathbf{x}'$. The procedure is:

1. Detect interest points in both images. Our implementation uses public-domain code for the Harris corner detector [15], from `www.targetjr.org`. The two sets are $\{\mathbf{x}_j\}_{j=1}^{N}$ and $\{\mathbf{x}'_k\}_{k=1}^{N'}$. Typically, parameters are set so that about 500 corners are computed in the full image, with proportionally fewer in the tracking region.

2. Match interest points:
   For each point $\mathbf{x}_j$, choose the point $\mathbf{x}'_k$ in the next frame which maximizes the cross-correlation in a 7×7 window. This generates a set of correspondences $\mathbf{x}_j \leftrightarrow \mathbf{x}'_k$. Typically, each pixel is compared with 10 others for a maximum image movement of 50 pixels.

3. Robust estimation of H:
   From the set of correspondences, randomly sample subsets of four $\mathbf{x} \leftrightarrow \mathbf{x}'$ pairs. For each sample, compute H as in §3.2. Each candidate H is tested against all the correspondences by computing the distance between $\mathbf{x}'$ and $\mathsf{H}\mathbf{x}$. We choose the H for which the most pairs are within a threshold of say 2.5 pixels.

This process gives a value for H and a set of inlier correspondences. A nonlinear minimization of reprojection error over the inliers is found to significantly reduce jitter.

4.2 Initialization

The system as currently implemented requires that the plane be approximately specified before tracking commences. Corner detection and matching are then restricted to this region, and when the frame-to-frame homography is computed, the region is transformed to the new image. These steps can be improved in a few ways: (1) because of the robust computation of H_i^{i+1}, the plane need only be very approximately indicated; (2) rather than transforming the polyline boundary of the region, we can gather all inliers to the homography, and set the boundary to the convex hull of the 2D points. This will allow the region to expand and contract as necessary through the sequence.

4.3 Automatic plane detection

The RANSAC procedure described in §4.1 can be applied on the full image, giving the homography that corresponds to the largest set of coplanar points. Therefore, the initial indication of the plane can be omitted if there is only one plane in the scene, or if the largest plane is the one we wish to track. Moreover, this automatic plane detection can be used to choose the most reliable plane for the "hand-off" procedure.

4.4 Reliability issues

Plane tracking of interest points has an important reliability advantage over tracking points undergoing general motion. In general motion tracking, the relationship between points in successive frames is described by the *fundamental matrix* [9]. For a given point $\mathbf{x}$ in the first view, the corresponding point in the second view must lie on

Figure 5. Augmentation results, stanislas square. Jitter and drift are sub-pixel over this 40-frame sequence.

the epipolar line, generated by $\mathbf{l} = \mathtt{F}\mathbf{x}$. Assume the image height is 500 pixels. If we consider, as an example, a case where the epipolar line is horizontal and an acceptance threshold of ± 2.5 pixels, then 1% of randomly detected points will be viable matches for $\mathbf{x}$. For homography matches, on the other hand, the transfer is exact, so the corresponding point must lie in a circle of radius 2.5 centered on $\mathtt{H}\mathbf{x}$, a region which is only 0.01% of the image area. Therefore, rogue matches are significantly less likely for homography trackers.

The major source of error in the system as described is drift. There is effectively zero jitter, even for points far from the plane, due to the accuracy of homographies computed from hundreds of corner features. On the other hand, the pose for each new frame is computed by multiplying the homographies for each of the previous frames, so errors will accumulate over time. These errors can be ameliorated in a few ways, but the most important is the matching of points from the first frame to frame $i + 1$ whenever possible.

This is simplified because the computed homography $\mathtt{H}_0^i$ is available, reducing the search region. However, because there will be significant perspective distortion between the two frames, it is necessary to wrap the images using the homography in order to compute the cross-correlation scores.

5 Results

The results of applying the proposed system to the plaza sequence are shown in figure 5. In this case, the ground plane was tracked, and a 3D model superimposed on the sequence to evaluate the registration accuracy. The movie sequence is available at
`http://www.robots.ox.ac.uk/~vgg/isar/a.mpg`.
The reader can notice that there is no perceptible drift or jitter.

The second example, in figure 6, is an outdoor scene, with a hand-held camera circumnavigating a Henry Moore sculpture. The grass provided sufficient texture to track the plane, thanks in part to the inherent robustness of homography tracking. Registration on this sequence is also good, with low jitter, but drift of a few pixels.

6 Discussion

We have presented a markerless optical tracker for augmented reality, which provides accurate and reliable results for scenes which include planar structures. The new mathematical contributions are the framework for uncalibrated plane tracking, and camera recovery there from. A preliminary implementation yields results comparable in accuracy with full structure-and-motion methods but with better reliability. In addition, aligning the real and virtual coordinate systems is greatly simplified by the identification of the plane.

Like any vision-based tracker, our system will fail if the lighting conditions are unfavourable, or the tracked plane goes out of view. However, the entire plane does not have to be visible at any time, and the system can move from one plane to another to maintain tracking. This makes it applicable in a wide range of environments. Of course, a hybrid system would increase robustness particularly in hand-off between planes which cannot be seen simultaneously.

References

[1] A. J. Azarbayejani, B. Horowitz, and A. Pentland. Recursive estimation of structure and motion using relative orientation constraints. In *Proc. CVPR*, pages 294–299, 1993.

[2] R. Azuma, J. W. Lee, B. Jiang, J. Park, S. You, and U. Neumann. Tracking in unprepared environments

Figure 6. Augmentation results, Henry Moore. Jitter is sub-pixel, drift of the order of a few pixels. These works are reproduced by permission of the Henry Moore Foundation.

for augmented reality systems. *IEEE Transactions on Computer Graphics*, 1999.

[3] P. A. Beardsley, P. H. S. Torr, and A. Zisserman. 3D model aquisition from extended image sequences. In *Proc. ECCV*, pages 683–695, 1996.

[4] R. Behringer. Improving the precision of registration for augmented reality in an outdoor scenario by visual horizon silhouette matching. In *First IEEE Workshop on Augmented Reality (IWAR'98)*, 1998.

[5] M.-O. Berger, B. Wrobel-Dautcourt, S. Petitjean, and G. Simon. Mixing Synthetic and Video Images of an Outdoor Urban Environment. *Machine Vision and Applications*, 11(3), 1999.

[6] B. Caprile and V. Torre. Using vanishing points for camera calibration. *IJCV*, pages 127–140, 1990.

[7] G. Cross, A. W. Fitzgibbon, and A. Zisserman. Parallax geometry of smooth surfaces in multiple views. In *Proc. ICCV*, pages 323–329, Sep 1999.

[8] D. Dementhon and L. Davis. Model Based Object Pose in 25 Lines of Code. *IJCV*, 15:123–141, 1995.

[9] O. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? In *Proc. ECCV*, LNCS 588, pages 563–578. Springer-Verlag, 1992.

[10] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 24(6):381–395, 1981.

[11] A. W. Fitzgibbon, G. Cross, and A. Zisserman. Automatic 3D model construction for turn-table sequences. In R. Koch and L. Van Gool, editors, *3D Structure from Multiple Images of Large-Scale Environments, LNCS 1506*, pages 155–170. Springer-Verlag, Jun 1998.

[12] A. W. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. In *Proc. ECCV*, pages 311–326. Springer-Verlag, Jun 1998.

[13] G. H. Golub and C. Van Loan. *Matrix Computations*. The John Hopkins University Press, Baltimore, MD, second edition, 1989.

[14] C. J. Harris and J. M. Pike. 3D positional integration from image sequences. In *Alvey Vision Conf.*, pages 233–236, 1987.

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

[16] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000.

[17] M. Jethwa, A. Zisserman, and A. W. Fitzgibbon. Real-time panoramic mosaics and augmented reality. In *Proc. BMVC*, pages 852–862, 1998.

[18] D. Koller, K. Daniilidis, and H. H. Nagel. Model-Based Object Tracking in Traffic Scenes. In *Proc. ECCV*, volume 588, pages 437–452, 1992.

[19] D. Koller, G. Klinker, E. Rose, D. Breen, R. Whitaker, and M. Tuceryan. Real-time vision-based camera tracking for augmented reality applications. In *ACM Symposium on Virtual Reality, Software and Technology (VRST-97)*, 1997.

[20] K. N. Kutulakos and J. R. Vallino. Calibration-free augmented reality. *IEEE Trans. on Visualization and Computer Graphics*, 4(1):1–20, 1998.

[21] S. Laveau. *Géométrie d'un système de N caméras. Théorie, estimation et applications*. PhD thesis, INRIA, 1996.

[22] D. Liebowitz and A. Zisserman. Combining scene and auto-calibration constraints. In *Proc. ICCV*, Sep 1999.

[23] H. P. Moravec. Towards Automatic Visual Obstacle Avoidance. In *IJCAI*, 1977.

[24] U. Neumann, S. You, Y. Cho, J. Lee, and J. Park. Natural feature tracking for augmented reality. *IEEE Transactions on Multimedia*, 1(1):53–64, Mar 1999.

[25] Princeton Video Image. Product showcase, 2000. `http://www.pvi-inc.com/showcase`.

[26] S. Ravela, B. Draper, J. Lim, and R. Weiss. Tracking Object Motion Across Aspect Changes for Augmented Reality. In *ARPA Image Undertanding Worshop, Palm Spring (USA)*, Aug 1996.

[27] G. Simon and M.-O. Berger. A two-stage robust statistical method for temporal registration from features of various type. In *Proc. ICCV*, pages 261–266, 1998.

[28] R. A. Smith, A. W. Fitzgibbon, and A. Zisserman. Improving augmented reality using image and scene constraints. In *Proc. BMVC*, pages 295–304. BMVA Press, 1999.

[29] D. Stricker, G. Klinker, and D. Reiners. A fast and robust line-based optical tracker for augmented reality applications. In *First IEEE Workshop on Augmented Reality (IWAR'98)*, 1998.

[30] M. Uenohara and T. Kanade. Vision based object registration for real time image overlay. *Journal of Computers in Biology and Medecine*, 1996.

[31] G. Welch and G. Bishop. SCAAT: Incremental tracking with incomplete information. In *Proc. ACM SIGGRAPH*, 1997.