Vision Based 3D Tracking and Pose Estimation for Mixed Reality

P. Fua and V. Lepetit
Ecole Polytechnique Fédérale de Lausanne (EPFL)
Computer Vision Laboratory
CH-1015 Lausanne, Switzerland

Abstract

Mixed Reality applications require accurate knowledge of the relative positions of the camera and the scene. When either of them moves, this means keeping track in real-time of all six degrees of freedom that define the camera position and orientation relative to the scene, or, equivalently, the 3D displacement of an object relative to the camera.

Many technologies have been tried to achieve this goal. However, Computer Vision is the only one that has the potential to yield non-invasive, accurate and low-cost solutions to this problem, provided that one is willing to invest the effort required to develop sufficiently robust algorithms. In this chapter, we therefore discuss some of the most promising approaches, their strengths, and their weaknesses.

INTRODUCTION

Tracking an object in a video sequence means continuously identifying its location when either the object or the camera are moving. More specifically, 3D tracking aims at continuously recovering all six degrees of freedom that define the camera position and orientation relative to the scene, or, equivalently, the 3D displacement of an object relative to the camera.

Many other technologies besides vision have been tried to achieve this goal, but they all have their weaknesses: Mechanical trackers are accurate enough, although they tether the user to a limited working volume. Magnetic trackers are vulnerable to distortions by metal in the environment, which are a common occurrence, and also limit the range of displacements. Ultrasonic trackers suffer from noise and tend to be inaccurate at long ranges because of variations in the ambient temperature. Inertial trackers drift with time.

By contrast, vision has the potential to yield non-invasive, accurate and low-cost solutions to this problem, provided that one is willing to invest the effort required to develop sufficiently robust algorithms. In some cases, it is acceptable to add fiducials, such as LEDs or special markers, to the scene or target object to ease the registration task, as will be discussed in Section 1. Of course, this assumes that one or more fiducials are visible at all times. Otherwise, the registration falls apart. Moreover, it is not always possible to place fiducials. For example, Augmented Reality end-users do not like markers because they are visible in the scene and it is not always possible to modify the environment before the application has to run.

It is therefore much more desirable to rely on naturally present features, such as edges, corners, or texture. Of course, this makes tracking far more difficult: Finding and following feature points or edges on many every day's objects is sometimes difficult because there may be few of them. Total, or even partial occlusion of the tracked objects typically results in tracking failure. The camera can easily move too fast so that the images are motion blurred; the lighting during a shot can change significantly; reflections and specularities may confuse the tracker. Even more importantly, an object may drastically change its aspect very quickly due to displacement. For example this happens when a camera films a building and goes around the corner, causing one wall to disappear and a new one to appear. In such cases, the features to be followed always change and the tracker must deal with features coming in and out of the picture. In sections 2 and 3, we focus on solutions to these difficult problems and show how planar, non-planar, and even deformable objects can be handled.

For the sake of completeness, we provide in appendix a brief description of the camera models that all these techniques rely on, as well as pointers to useful implementations and more extensive descriptions.

1 Fiducial-Based Tracking

Vision-based 3D tracking can be decomposed into two main steps: First image processing to extract some information from the images, and second pose estimation itself. The addition in the scene of *fiducials*, also called *landmarks* or *markers*, greatly helps both steps: They constitute image features easy to extract, and they provide reliable, easy to exploit measurements for pose estimation.

1.1 Point-Like Fiducials

Fiducials have been used for many years by close-range photogrammetrists. They can be designed in such a way that they can be easily detected and identified with an *ad hoc* method. Their image locations can also be measured to a much higher accuracy than natural features. In particular, circular fiducials work best, because the appearance of circular patterns is relatively invariant to perspective distortion, and because their centroid provides a stable 2D position, which can easily be determined with sub-pixel accuracy. The 3D positions of the fiducials in the world coordinate system are assumed to be precisely known: This can be achieved by hand, with a laser, or with a structure-from-motion algorithm. To facilitate their identification, the fiducials can be arranged in a distinctive geometric pattern. Once the fiducials are identified in the image, they provide a set of correspondences that can be used to retrieve the camera pose.

For high-end applications, companies such as Geodetic services, Inc., Advanced Real-time Tracking GmbH, Metronor, ViconPeak, AICON 3D Systems GmbH propose commercial products based on this approach. Lower-cost, and lower-accuracy solutions, have also been proposed by the Computer Vision community. For example, the Concentric Contrasting Circle (CCC) fiducial [Hoff *et al.*, 1996] is formed by placing a black ring on a white background, or vice-versa. To detect these fiducials, the image is first thresholded, morphological operations are then applied to eliminate too small regions, and a connected component labeling operation is performed to find white and black regions, as well as their centroids. Along the same lines, [State *et al.*, 1996] uses color-coded fiducials for a more reliable identification. Each fiducial consists of an inner dot and a surrounding outer ring, four different colors are used, and thus 12 unique fiducials can be created and identified based on their two colors. Because the tracking

range is constrained by the detectability of fiducials in input images, [Cho *et al.*, 1998] introduces a system that uses several sizes for the fiducials. They are composed of several colored concentric rings, where large fiducials have more rings than smaller ones, and diameters of the rings are proportional to their distance to the fiducial center, to facilitate their identification. When the camera is close to fiducials, only small size fiducials are detected. When it is far from them, only large size fiducials are detected.

While all the previous methods for fiducial detection use *ad hoc* schemes, [Claus and Fitzgibbon, 2004] uses a machine learning approach which delivers significant improvements in reliability. The fiducials are made of black disks on white background, and sample fiducial images are collected under varying perspective, scale and lighting conditions, as well as negative training images. A cascade of classifiers is then trained on these data: The first step is a fast Bayes decision rule classification, the second one a powerful but slower nearest neighbor classifier on the subset passed by the first stage. At run-time, all the possible sub-windows in the image are classified using this cascade. This results in a remarkably reliable fiducial detection method.

1.2 Extended Fiducials

The fiducials presented above were all circular and only their center was used. By contrast, [Koller *et al.*, 1997] introduces squared, black on white, fiducials, which contain small red squares for their identification. The corners are found by fitting straight line segments to the maximum gradient points on the border of the fiducial. Each of the four corners of such fiducials provides one correspondence and the pose is estimated using an Extended Kalman filter.

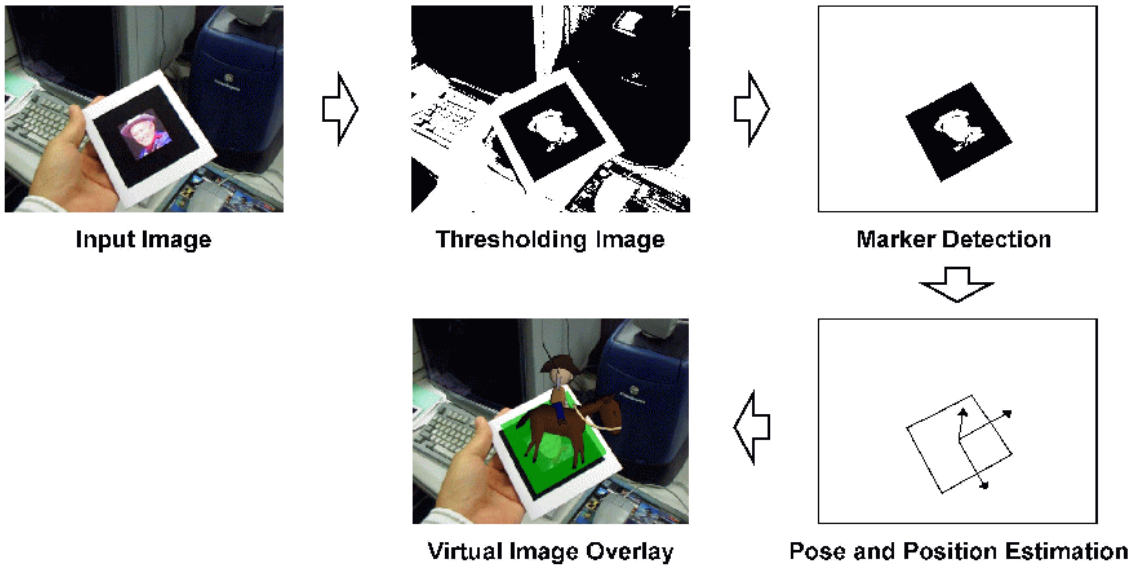


Figure 1: Processing flow of ARToolKit: The marker is detected in the thresholded image, and then used to estimate the camera pose. Courtesy of H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto and K. Tachibana.

Planar rectangular fiducials are also used in [Rekimoto, 1998, Kato and Billinghurst, 1999, Kato *et al.*, 2000] and it is shown that a single fiducial is enough to estimate the pose. Fig. 1 depicts their

approach. It has become popular, because it yields a robust, low-cost solution for real-time 3D tracking, and a software library called ARToolKit is publicly available [ART,].

The whole process, the detection of the fiducials and the pose estimation, runs in real-time, and therefore can be applied in every frame: The 3D tracking system does not require any initialization by hand, and is robust to fiducial occlusion. In practice, under good lighting conditions, the recovered pose is also accurate enough for Augmented Reality applications. These characteristics make ARToolKit a good solution to 3D tracking, whenever the engineering of the scene is possible.

2 Using Natural Features

Using markers to simplify the 3D tracking task requires engineering of the environment, which end-users of tracking technology do not like or is sometimes even impossible, for example in outdoor environments. Whenever possible, it is therefore much better to be able to rely on features naturally present in the images. Of course, this approach makes tracking much more challenging and some 3D knowledge is often required to make things easier. For MR applications, this is not an issue since 3D scene models are typically available and we therefore focus here on model-based approaches.

We distinguish here two families of approaches depending on the nature of the image features being used. The first one is formed by edge-based methods that match the projections of the target object 3D edges to area of high image gradient. The second family includes all the techniques that rely on information provided by pixels inside the object's projection.

2.1 Edge-Based Methods

Historically, the early approaches to tracking were all edge-based mostly because these methods are both computationally efficient, and relatively easy to implement. They are also naturally stable to lighting changes, even for specular materials, which is not necessarily true of methods that consider the internal pixels, as will be discussed later. The most popular approach is to look for strong gradients in the image around a first estimation of the object pose, without explicitly extracting the contours [Harris, 1992, Armstrong and Zisserman, 1995, Marchand *et al.*, 2001, Drummond and Cipolla, 2002, Comport *et al.*, 2003, Vacchetti *et al.*, 2004a], which is fast and general.

2.1.1 RAPiD

Even though RAPiD [Harris, 1992] was one of the first 3D tracker to successfully run in real-time and many improvements have been proposed since, many of its basic components have been retained in more recent systems. The key idea is to consider a set of 3D points on the object, called control points, which lie on high contrast edges in the images. As shown in Figure 2, the control points can be sampled along the 3D model edges and in the areas of rapid albedo change. They can also be generated on the fly as points on the occluding contours of the object. The 3D motion of the object between two consecutive frames can be recovered from the 2D displacement of the control points.

Once initialized, the system performs a simple loop: For each frame, the predicted pose, which can simply be the pose estimated for the previous frame, is used to predict which control points will be

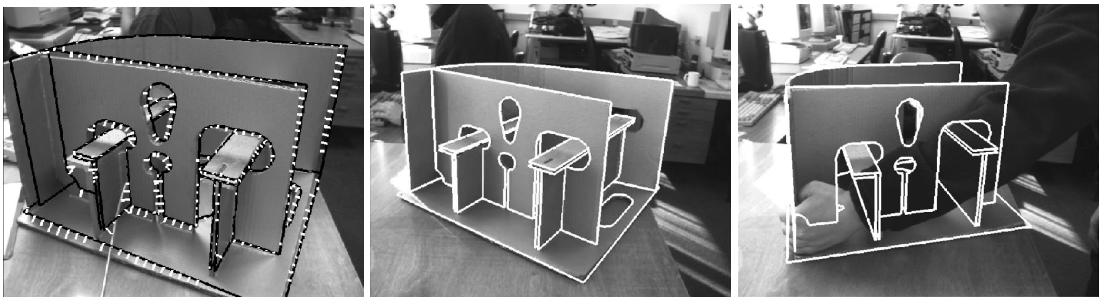


Figure 2: In RAPiD-like approaches, control points are sampled along the model edges. The small white segments in the left image join the control points in the previous image to their found position in the new image. The pose can be inferred from these matches, even in presence of occlusions by introducing robust estimators. Figure courtesy of T. Drummond and R. Cipolla.

visible and what their new locations should be. The control points are matched to the image contours, and the new pose estimated from these correspondences via least-squares minimization.

In [Harris, 1992], some enhancements to this basic approach are proposed. When the edge response at a control point becomes too weak, it is not taken into account into the motion computation, as it may subsequently incorrectly latch on to a stronger nearby edge. As we will see below, this can also be handled using a robust estimator. An additional clue that can be used to reject incorrect edges is their polarity, that is whether they correspond to a transition from dark to light or from light to dark. A way to use occluding contours of the object is also given.

2.1.2 Making RAPiD Robust

The main drawback of of the original RAPiD formulation is its lack of robustness. The weak contours heuristics is not enough to prevent incorrectly detected edges from disturbing the pose computation. In practice, such errors are frequent. They arise from occlusions, shadows, texture on the object itself, or background clutter.

Several methods have been proposed to make the RAPiD computation more robust. [Drummond and Cipolla, 2002] uses a robust estimator and replaces the least-squares estimation by an iterative re-weighted least-squares to solve the new problem. Similarly, [Marchand *et al.*, 2001] uses a framework similar to RAPiD to estimate a 2D affine transformation between consecutive frames, but also replaces standard least-squares by robust estimation.

In the approaches described above, the control points were treated individually, without taking into account that several control points are often placed on the same edge, and hence their measurements are correlated. By contrast, in [Armstrong and Zisserman, 1995, Simon and Berger, 1998] control points lying on the same object edge are grouped into primitives, and a whole primitive can be rejected from the pose estimation. In [Armstrong and Zisserman, 1995], a RANSAC methodology [Fischler and Bolles, 1981] is used to detect outliers among the control points forming a primitive. If the number of remaining control points falls below a threshold after elimination of the outliers, the primitive is ignored in the pose update. Using RANSAC implies that the primitives have an analytic expression, and precludes tracking free-form curves. By contrast, [Simon and Berger, 1998] uses a robust estimator to compute a local

residual for each primitive. The pose estimator then takes into account all the primitives using a robust estimation on the above residuals.

When the tracker finds multiple edges within its search range, it may end-up choosing the wrong one. To overcome this problem, in [Drummond and Cipolla, 2002], the influence of a control point is inversely proportional to the number of edge strength maxima visible within the search path. [Vacchetti *et al.*, 2004a] introduces another robust estimator to handle multiple hypotheses and retain all the maxima as possible correspondents in the pose estimation.

2.2 Texture-Based Methods

If the object is sufficiently textured, information can be can be derived from optical flow [Li *et al.*, 1993, Basu *et al.*, 1996, Decarlo and Metaxas, 2000], template matching [Hager and Belhumeur, 1998, Cascia *et al.*, 2000, Jurie and Dhome, 2001, Jurie and Dhome, 2002] or interest-point correspondences. However the latter is probably the most effective for MR applications because they rely on matching local features. Given such correspondences, the pose can be estimated by least-square minimization, or even better, by robust estimation. They are therefore relatively insensitive to partial occlusions or matching errors. Illumination invariance is also simple to achieve. And, unlike edge-based methods, they do not get confused by background clutter and exploit more of the image information, which tends to make them more dependable.

2.2.1 Interest Point Detection and 2D Matching

In interest point methods, instead of matching all pixels in an image, only some pixels are first selected with an "interest operator" before matching. This reduces the computation time while increasing the reliability if the pixels are correctly chosen. [Förstner, 1986] presents the desired properties for such an interest operator: Selected points should be different from their neighbors, which eliminates edge-points; the selection should be repeatable, that is the same points should be selected in several images of the same scene, despite perspective distortion or image noise. In particular, the precision and the reliability of the matching directly depends on the invariance of the selected position. Pixels on repetitive patterns should also be rejected or at least given less importance to avoid confusion during matching.

Such an operator was already used in the 1970's for tracking purposes [Moravec, 1977, Moravec, 1981]. Numerous other methods have been proposed since and [Deriche and Giraudon, 1993, Smith and Brady, 1995] give good surveys of them. Most of them involve second order derivatives, and results can be strongly affected by noise. Several successful interest point detectors [Förstner, 1986, Harris and Stephens, 1988, Shi and Tomasi, 1994] rely on the auto-correlation matrix computed at each pixel location. It is a 2×2 matrix, whose coefficients are sums over a window of the first derivatives of the image intensity with respect to the pixel coordinates, and its measures the local variations of the image. As discussed in [Förstner, 1986], the pixels can be classified from the behavior of the eigenvalues of the auto-correlation matrix: Pixels with two large, approximately equal eigenvalues are good candidates for selection. [Shi and Tomasi, 1994] shows that locations with two large eigenvalues can be reliably tracked, especially under affine deformations, and considers locations where the smallest eigen value is higher than a threshold. Interest points can then taken to be the locations that are local maxima of the chosen measure above a predefined threshold. The derivatives involved in the auto-correlation matrix can be weighted using a Gaussian kernel to increase robustness to noise [Schmid and

Figure 3: Face tracking using interest points and one reference image shown on the top left.

Mohr, 1997]. The derivatives should also be computed using a first order Gaussian kernel. This comes at a price since it tends to degrade both the localization accuracy and the performance of the image patch correlation procedure used for matching purposes.

For tracking purpose, it is then useful to match two sets of interest points and extracted from two images taken from similar viewpoints. A classical procedure [Zhang *et al.*, 1995] runs as follows: For each point in the first image, search in a region of the second image around its location for a corresponding point. The search is based on the similarity of the local image windows centered on the points, which strongly characterize the points when the images are sufficiently close. The similarity can be measured using the zero-normalized cross-correlation that is invariant to affine changes of the local image intensities, and make the procedure robust to illumination changes. To obtain a more reliable set of matches, one can reverse the role of the two images, and repeat the previous procedure. Only the correspondences between points that chose each other are kept.

2.2.2 Eliminating Drift

In the absence of points whose coordinates are known *a priori*, all methods are subject to error accumulation, which eventually results in tracking failure and precludes of truly long sequences.

A solution to this problem is to introduce one or more *keyframes* such as the one in the upper left corner of Figure 3, that is images of the target object or scene for which the camera has been registered *beforehand*. At runtime, incoming images can be matched against the keyframes to provide a position estimate that is drift-free [Ravela *et al.*, 1995, Genc *et al.*, 2002, Tordoff *et al.*, 2002]. This, however, is more difficult than matching against immediately preceding frames as the difference in viewpoint is likely to be much larger. The algorithm used to establish point correspondences must therefore both be fast and relatively insensitive to large perspective distortions, which is not usually the case for those used by the algorithms of Section 2.2.1 that need only handle small distortions between consecutive frames.

In [Vacchetti *et al.*, 2004b], this is handled as follows. During a training stage, the system extracts interest points from each keyframe, back-projects them to the object surface to compute their 3D posi-

tion, and stores image patches centered around their location. During tracking, for each new incoming image, the system picks the keyframe whose viewpoint is closest to that of the last known viewpoint. It synthesizes an *intermediate image* from that keyframe by warping the stored image patches to the last known viewpoint, which is typically the one corresponding to the previous image. The intermediate and the incoming images are now close enough that matching can be performed using simple, conventional, and fast correlation methods. Since the 3D position in the keyframe has been precomputed, the pose can then be estimated by robustly minimizing the reprojection error. This approach handles perspective distortion, complex aspect changes, and self-occlusion. Furthermore, it is very efficient because it takes advantage of the large graphics capabilities of modern CPUs and GPUs.

However, as noticed by several authors [Ravela *et al.*, 1995, Chia *et al.*, 2002, Tordoff *et al.*, 2002, Vacchetti *et al.*, 2004b], matching only against keyframes does not, by itself, yield directly exploitable results. This has two main causes. First, wide-baseline matching as described in the previous paragraph, is inherently less accurate than the short-baseline matching involved in frame-to-frame tracking, which is compounded by the fact that the number of correspondences that can be established is usually less. Second, if the pose is computed for each frame independently, no temporal consistency is enforced and the recovered motion can appear to be jerky. If it were used as is by an MR application, the virtual objects inserted in the scene would appear to *jitter*, or to tremble, as opposed to remaining solidly attached to the scene.

Temporal consistency can be enforced by some dynamical smoothing using a motion model. Another way proposed in [Vacchetti *et al.*, 2004b] is to combine the information provided by the keyframes, which provides robustness, with that coming from preceding frames, which enforces temporal consistency. This does not make assumptions on the camera motion and improves the accuracy of the recovered pose. It is still compatible with the use of dynamical smoothing that can be useful to in case where the pose estimation remains unstable, for example when the object is essentially fronto-parallel.

3 Tracking by Detection

The recursive nature of traditional 3D tracking approaches provides a strong prior on the pose for each new frame and makes image feature identifications relatively easy. However, it comes at a price: First, the system must either be initialized by hand or require the camera to be very close to a specified position. Second, it makes the system very fragile. If something goes wrong between two consecutive frames, for example due to a complete occlusion of the target object or a very fast motion, the system can be lost and must be re initialized in the same fashion. In practice, such weaknesses make purely recursive systems nearly unusable, and the popularity of ARToolKit [Kato *et al.*, 2000] in the Augmented Reality community should come as no surprise: It is the first vision-based system to really overcome these limitations by being able to detect the markers in every frame without constraints on the camera pose.

However, achieving the same level of performance *without* having to engineer the environment remains a desirable goal. Since object pose and appearance are highly correlated, estimating both simultaneously increases the performances of object detection algorithms. Therefore, 3D pose estimation from natural features without *a priori* knowledge of the position and object detection are closely related problems. Detection has a long history in Computer Vision. It has often relied on 2D detection even for 3D objects [Nayar *et al.*, 1996, Viola and Jones, 2001]. However, there has been sustained interest in simultaneous object detection and 3D pose estimation. Early approaches were edge-based [Lowe, 1991,

Jurie, 1998], but methods based on feature points matching have become popular since local invariants were shown to work better for that purpose [Schmid and Mohr, 1997].

Feature point-based approaches to be the most robust to scale, viewpoint, and illumination changes, as well as partial occlusions. They typically operate on the following principle. During an offline training stage, one builds a database of interest points lying on the object and whose position on the object surface can be computed. A few images in which the object has been manually registered are often used for this purpose. At runtime, feature points are first extracted from individual images and matched against the database. The object pose can then be estimated from such correspondences. RANSAC-like algorithms [Fischler and Bolles, 1981] or the Hough transform are very convenient for this task since they eliminate spurious correspondences while avoiding combinatorial issues.

The difficulty in implementing such approaches comes from the fact that the database images and the input ones may have been acquired from very different viewpoints. As discussed in Section 2.2.1, unless the motion is very quick, this problem does not arise in conventional recursive tracking approaches because the images are close to each other. However, for tracking-by-detection purposes, the so-called *wide baseline* matching problem becomes a critical issue that must be addressed.

In the remainder of this section, we discuss in more detail the extraction and matching of feature points in this context. We conclude by discussing the relative merits of tracking-by-detection and recursive tracking.

3.1 Feature Point Extraction

To handle as wide as possible a range of viewing conditions, feature point extraction should be insensitive to scale, viewpoint, and illumination changes. Note that the stability of the extracted features is much more crucial here than for the techniques described Section 2.2.1 where only close frames were matched. Different techniques are therefore required and we discussed them below.

As proposed in [Lindeberg, 1994], scale-invariant extraction can be achieved by taking feature points to be local extrema of a Laplacian-of-Gaussian pyramid in scale-space. To increase computational efficiency, the Laplacian can be approximated by a Difference-of-Gaussians [Lowe, 1999]. Research has then focused on affine invariant region detection to handle more perspective changes. [Baumberg, 2000, Schaffalitzky and Zisserman, 2002, Mikolajczyk and Schmid, 2002] used an affine invariant point detector based on the Harris detector, where the affine transformation that makes equal the two eigen values of the auto correlation matrix is evaluated to rectify the patch appearance. [Tuytelaars and Gool, 2000] achieves such invariance by fitting an ellipse to the local texture. [Matas *et al.*, 2002] proposes a fast algorithm to extract Maximally Stable Extremal Regions demonstrated in a live demo. [Mikolajczyk *et al.*, 2005] gives a good summary and comparisons of the existing affine invariant regions detectors.

3.2 Wide Baseline Matching

Once a feature point has been extracted, the most popular approach to matching it is first to characterize it in terms of its image neighborhood and then to compare this characterization to those present in the database. Such characterization, or *local descriptor*, should be not only invariant to viewpoint and illumination changes but also highly distinctive. We briefly review some of the most representative below.

(a) (b) (c)

Figure 4: Using SIFT for tracking-by-detection. (a) Detected SIFT features. (b,c) They have been used to track the pose of the camera and add the virtual teapot. Figure courtesy of D.G. Lowe and I. Gordon.

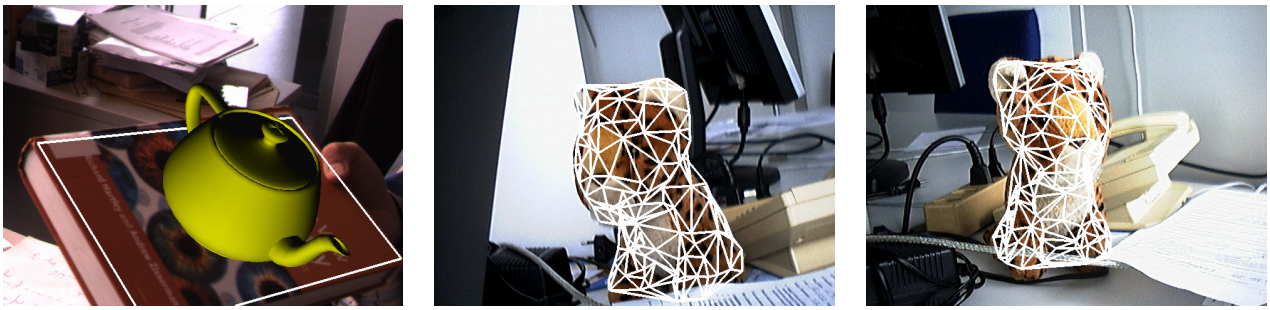
3.2.1 Local Descriptors

Many such descriptors have been proposed over the years. For example, [Schmid and Mohr, 1997] computes rotation invariant descriptors as functions of relatively high order image derivatives to achieve orientation invariance; [Tuytelaars and Gool, 2000] fits an ellipse to the texture around local intensity extrema and uses the Generalized Color Moments [Mindru *et al.*, 1999] as a descriptor. [Lowe, 2004] introduces a descriptor called SIFT based on multiple orientation histograms, which tolerates significant local deformations. This last descriptor has been shown in [Mikolajczyk and Schmid, 2003] to be one of the most efficient. As illustrated by Figure 4, it has been successfully applied to 3D tracking in [Se *et al.*, 2002, Skrypnyk and Lowe, 2004] and we now describe it in more detail.

The remarkable invariance of the SIFT descriptor is achieved by a succession of carefully designed techniques. First the location and scale of the keypoints are determined precisely by interpolating the pyramid of Difference-of-Gaussians used for the detection. To achieve image rotation invariance, an orientation is also assigned to the keypoint. It is taken to be the one corresponding to a peak in the histogram of the gradient orientations within a region around the keypoint. This method is quite stable under viewpoint changes, and achieves an accuracy of a few degrees. The image neighborhood of the feature point is then corrected according to the estimated scale and orientation, and a local descriptor is computed on the resulting image region to achieve invariance to the remaining variations, such as illumination or out-of-plane variation. The point neighborhood is divided into several, typically 4×4, subregions and the contents of each subregion is summarized by an height-bin histogram of gradient orientations. The keypoint descriptor becomes a vector with 128 dimensions, built by concatenating the different histograms. Finally, this vector is normalized to unit length to reduce the effects of illumination changes.

3.2.2 Statistical Classification

The SIFT descriptor has been empirically shown to be both very distinctive and computationally cheaper than those based on filter banks. To shift even more of the computational burden from matching to training, which can be performed beforehand, we have proposed in our own work an alternative approach based on machine learning techniques [Lepetit *et al.*, 2005, Lepetit and Fua, 2006]. We treat wide baseline matching of keypoints as a classification problem, in which each class corresponds to the set of all



(a) (b) (c)

Figure 5: Detection and computation in real-time of the 3D pose. (a) A planar object. (b,c) A full 3D object.

possible views of such a point. Given one or more images of a target object, the system synthesizes a large number of views, or image patches, of individual keypoints to automatically build the training set. If the object can be assumed to be locally planar, this is done by simply warping image patches around the points under affine deformations, otherwise, given the 3D model, standard Computer Graphics texture-mapping techniques can be used. This second approach relaxes the planarity assumptions.

The classification itself is performed using randomized trees [Amit and Geman, 1997]. Each non-terminal node of a tree contains a test of the type: "Is this pixel brighter than this one ?" that splits the image space. Each leaf contains an estimate based on training data of the conditional distribution over the classes given that a patch reaches that leaf. A new image is classified by simply dropping it down the tree. Since only pixel intensities comparisons are involved, this procedure is very fast and robust to illumination changes. Thanks to the efficiency of randomized trees, it yields reliable classification results. As depicted by Figure 5, this method has been successfully used to detect and compute the 3D pose of both planar and non-planar objects.

As shown in Figure 6, this approach has been extended to deformable objects by replacing the rigid models by deformable meshes and introducing a well designed robust estimator. This estimator is the key to dealing with the large number of parameters involved in modeling deformable surfaces and rejecting erroneous matches for error rates of up to 95%, which is considerably more than what is required in practice [Pilet *et al.*, 2005b, Pilet *et al.*, 2005a]. It can then combined with a dynamic approach to estimating the amount of light that reaches individual image pixels by comparing their gray levels to those of the reference image. This lets us either erase patterns from the original images and replace them by blank but correctly shaded areas, which we think of as *Diminished Reality*, or to replace them by virtual ones that convincingly blend-in because they are properly lighted. As illustrated by Figure 7, this is important because adequate lighting is key to realism. Not only is this approach very fast and fully automated, but it also handles complex lighting effects, such as cast shadows, specularities, and multiple light sources of different hues and saturation.

3.2.3 From Wide Baseline Matching to 3D Tracking

As mentioned before, wide baseline matching techniques can be used to perform 3D tracking. To illustrate this, we briefly describe here the SIFT-based implementation reported in [Skrypnyk and Lowe, 2004].

(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 6: Real-time detection of a deformable object. Given a model image (a), the algorithm computes a function mapping the model to an input image (b). To illustrate this mapping, the contours of the model (c) are extracted using a simple gradient operator and used as a validation texture which is overlaid on the input image using the recovered transformation (d). Additional results are obtained in different conditions (e to h). Note that in all cases, the white outlines project almost exactly at the right place, thus indicating a correct registration and shape estimation. The registration process, including image acquisition, takes about 80 ms and does not require any initialization or *a priori* pose information.

First, during a learning stage, a database of scene feature points is built by extracting SIFT keypoints in some reference images. Because the keypoints are detected in scale-space, the scene does not necessarily have to be well-textured. Their 3D positions are recovered using a structure-from-motion algorithm. Two-view correspondences are first established based on the SIFT descriptors, and chained to construct multi-view correspondences while avoiding prohibitive complexity. Then the 3D positions are recovered by a global optimization over all camera parameters and these point coordinates, which is initialized as suggested in [Szeliski and Kang, 1994]. At run-time, SIFT features are extracted from the current frame, matched against the database, resulting in a set of 2D / 3D correspondences that can be used to recover the pose.

The best candidate match for a SIFT feature extracted from the current frame is assumed to be its nearest neighbor, in the sense of the Euclidean distance of the descriptor vectors, in the point database. The size of the database and the high dimensionality of these vectors would make the exhaustive search intractable, especially for real-time applications. To allow for fast search, the database is organized as a k-d tree. The search is performed so that bins are explored in the order of their closest distance from the query description vector, and stopped after a given number of data points has been considered, as described in [Beis and Lowe, 1997]. In practice, this approach returns the actual nearest neighbor with high probability.

As discussed Section 2.2.2, recovering the camera positions in each frame independently and from noisy data typically results in jitter. To stabilize the pose, a regularization term that smoothes camera motion across consecutive frames is introduced. Its weight is iteratively estimated to eliminate as much jitter as possible without introducing drift when the motion is fast. The full method runs at four frames

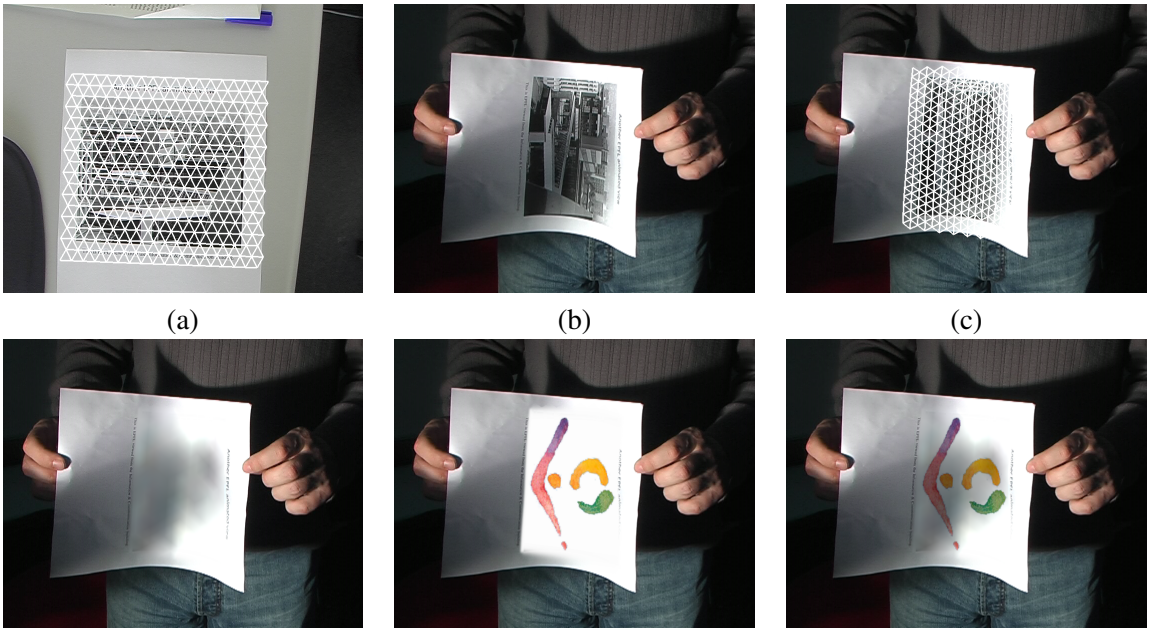


Figure 7: Augmenting a deformable object. (a) Reference image with overlaid mesh. (b) Input image. (c) Deformed mesh registered to the input image. (d) The original pattern has been erased and replaced by a blank but correctly shaded image. (e) A virtual pattern replaces the original one. It is correctly deformed but not yet relighted. (f) The virtual pattern is deformed and relighted.

per second on a 1.8 GHz ThinkPad.

3.3 The End of Recursive Tracking?

Since real-time tracking-by-detection has become a practical possibility, one must wonder if the conventional recursive tracking methods that have been presented in the previous sections of this survey are obsolescent.

We do not believe this to be the case. As illustrated by the case of the SIFT-based tracking system [Skrypnyk and Lowe, 2004] discussed above, treating each frame independently has its problems. Imposing temporal continuity constraints across frames can help increase the robustness and quality of the results. Furthermore, wide baseline matching tends to be both less accurate and more computationally intensive than the short baseline variety.

As shown in Section 2.2.2, combining both kinds of approaches can yield the best of both worlds: Robustness from tracking-by-detection, and accuracy from recursive tracking. In our opinion, this is where the future of tracking lays. The challenge will be to become able, perhaps by taking advantage of recursive techniques that do not require prior training, to learn object descriptions online so that a tracker can operate in a complex environment with minimal *a priori* knowledge.

4 Conclusion

Even after more than twenty years of research, practical vision-based 3–D tracking systems still rely on fiducials because this remains the only approach that is sufficiently fast, robust, and accurate. Therefore, if it is practical to introduce them in the environment the system inhabits, this solution surely must be retained. ARToolkit [ART,] is a freely available alternative that uses planar fiducials that may be printed on pieces of paper. While less accurate, it remains robust and allows for fast development of low-cost applications. As a result, it has become popular in the Augmented Reality Community.

However, this state of affairs may be about to change as computers have just now become fast enough to reliably handle natural features in real-time, thereby making it possible to completely do away with fiducials. This is especially true when dealing with objects that are polygonal, textured, or both [Drummond and Cipolla, 2002, Vacchetti *et al.*, 2004b]. However, the reader must be aware that the recursive nature of most of these algorithms makes them inherently fragile: They must be initialized manually and cannot recover if the process fails for any reason. In practice, even the best methods suffer such failures all too often, for example because the motion is too fast, a complete occlusion occurs, or simply because the target object moves momentarily out of the field of view.

This can be addressed by combining image data with data provided by inertial sensors, gyroscopes, or GPS [Ribo and Lang, 2002, Foxlin and Naimark, 2003, Klein and Drummond, 2003, Jiang *et al.*, 2004]. The sensors allow a prediction of the camera position or relative motion that can then be refined using vision techniques similar to the ones described in this chapter. When instrumenting the camera is an option, this combination is very effective for applications that require positioning the camera with respect to a static scene. However, it would be of no use to track moving objects with a static camera.

A more generic and desirable approach is therefore to develop purely image-based methods that can detect the target object and compute its 3D pose from a single image. If they are fast enough, they can then be used to initialize and re-initialize the system as often as needed, even if they cannot provide the same accuracy as traditional recursive approaches that use temporal continuity constraints to refine their estimates. Techniques able to do just this are just beginning to come online [Skrypnyk and Lowe, 2004, Lepetit *et al.*, 2005, Lepetit and Fua, 2006]. And, since they are the last missing part of the puzzle, we expect that we will not have to wait for another twenty years for purely vision-based commercial systems to become a reality.

Appendix: Camera Models

Most cameras currently used for tracking purposes can be modeled using the standard pinhole camera model that defines the imaging process as a projection from the world to the camera image plane. It is often represented by a projection matrix that operates on projective coordinates and can be written as the product of a camera calibration matrix that depends on the internal camera parameters and an rotation-translation matrix that encodes the rigid camera motion [Faugeras, 1993]. Note, however, that new camera designs, such as the so-called omnidirectional cameras that rely on hyperbolic or parabolic mirrors to achieve very wide field of views, are becoming increasingly popular [Geyer and Daniilidis, 2003, Swaminathan and Nayar, 2003].

4.1 Camera Matrices

The 3D tracking algorithms described here seek to estimate the rotation-translation matrix. It is computed as the composition of a translation and a rotation that must be appropriately parameterized for estimation and numerical optimization purposes. While representing translations poses no problem, parameterizing rotation is more difficult to do well. Several representations have been proposed, such as Euler angles, quaternions, and exponential maps. All of them present singularities, but it is generally accepted that the exponential map representation is the one that behaves best for tracking purposes [Grassia, 1998].

Since distinguishing a change in focal length from a translation along the camera Z-axis is difficult, in most 3D tracking methods, the internal camera parameters are assumed to be fixed. In other words, the camera cannot zoom. These parameters can be estimated during an offline camera calibration stage, for example by imaging once a calibration grid of known dimensions [Tsai, 1987, Faugeras, 1993] or several times a simpler 2D grid seen from several positions [Zhang, 2000, Sturm and Maybank, 1999]. The latter method is more flexible and executable code can be downloaded from [Cal, , Ope,].

4.2 Handling Lens Distortion

The pinhole camera model is very realistic for lenses with fairly long focal lengths but does not represent all the aspects of the image formation. In particular, it does not take into account the possible distortion from the camera lens, which may be non negligible especially for wide angle lenses.

Since they make it easier to keep target objects within the field of view, it is nevertheless desirable to have the option to use them for 3D tracking purposes. Fortunately, this is easily achieved because lens distortion mostly is a simple 2D radial deformation of the image. Given an estimate of the distortion parameters, it can be efficiently undone at run-time using a look-up table, which allows the use of the standard models discussed above.

The software package of [Ope,] allows the estimation of the distortion parameters using a method derived from [Heikkila and Silven, 1997]. This is a convenient method for desktop systems. For larger workspaces, plumb line based methods [Brown, 1971, Fryer and Goodin, 1989] are common in photogrammetry. Without distortion, the image of a straight line will be a straight line, and conversely the distortion parameters can be estimated from images of straight lines by measuring their deviations from straightness. This is a very practical method in man-made environments where straight lines, such as those found at building corners, are common.

References

[Amit and Geman, 1997] Y. Amit and D. Geman. Shape Quantization and Recognition With Randomized Trees. *Neural Computation*, 9(7):1545–1588, 1997.

[Armstrong and Zisserman, 1995] M. Armstrong and A. Zisserman. Robust Object Tracking. In *Asian Conference on Computer Vision*, pages 58–62, 1995.

[ART,] Artoolkit. http://www.hitl.washington.edu/artoolkit/.

[Basu *et al.*, 1996] S. Basu, I. Essa, and A. Pentland. Motion Regularization for Model-Based Head Tracking. In *International Conference on Pattern Recognition*, 1996.

[Baumberg, 2000] A. Baumberg. Reliable Feature Matching Across Widely Separated Views. In *Conference on Computer Vision and Pattern Recognition*, pages 774–781, 2000.

[Beis and Lowe, 1997] J. Beis and D.G. Lowe. Shape Indexing Using Approximate Nearest-Neighbour Search in High-Dimensional Spaces. In *Conference on Computer Vision and Pattern Recognition*, pages 1000–1006, 1997.

[Brown, 1971] D.C. Brown. Close Range Camera Calibration. *Photogrammetric Engineering*, 37(8):855–866, 1971.

[Cal,] A Flexible New Technique for Camera Calibration. http://research.microsoft.com/ zhang/Calib.

[Cascia *et al.*, 2000] M. Cascia, S. Sclaroff, and V. Athitsos. Fast, Reliable Head Tracking Under Varying Illumination: an Approach Based on Registration of Texture-Mapped 3D Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(4), April 2000.

[Chia *et al.*, 2002] K.W. Chia, A.D. Cheok, and S.J.D. Prince. Online 6 Dof Augmented Reality Registration from Natural Features. In *International Symposium on Mixed and Augmented Reality*, 2002.

[Cho *et al.*, 1998] Y. Cho, W.J. Lee, and U. Neumann. A Multi-Ring Color Fiducial System and Intensity-Invariant Detection Method for Scalable Fiducial-Tracking Augmented Reality. In *International Workshop on Augmented Reality*, 1998.

[Claus and Fitzgibbon, 2004] D. Claus and A. Fitzgibbon. Reliable Fiducial Detection in Natural Scenes. In *European Conference on Computer Vision*, pages 469–480, May 2004.

[Comport *et al.*, 2003] A.I. Comport, E. Marchand, and F. Chaumette. A Real-Time Tracker for Markerless Augmented Reality. In *International Symposium on Mixed and Augmented Reality*, September 2003.

[Decarlo and Metaxas, 2000] D. Decarlo and D. Metaxas. Optical Flow Constraints on Deformable Models With Applications to Face Tracking. *International Journal of Computer Vision*, 38:99–127, 2000.

[Deriche and Giraudon, 1993] R. Deriche and G. Giraudon. A Computational Approach for Corner and Vertex Detection. *International Journal of Computer Vision*, 10(2):101–124, 1993.

[Drummond and Cipolla, 2002] T. Drummond and R. Cipolla. Real-Time Visual Tracking of Complex Structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):932–946, July 2002.

[Faugeras, 1993] O.D. Faugeras. *Three-Dimensional Computer Vision: a Geometric Viewpoint*. MIT Press, 1993.

[Fischler and Bolles, 1981] M.A Fischler and R.C. Bolles. Random Sample Consensus: a Paradigm for Model Fitting With Applications to Image Analysis and Automated Cartography. *Communications ACM*, 24(6):381–395, 1981.

[Förstner, 1986] W. Förstner. A Feature-Based Correspondence Algorithm for Image Matching. *Journal of Machine Learning Research*, 26(3):150–166, 1986.

[Foxlin and Naimark, 2003] E. Foxlin and L. Naimark. Miniaturization, Claibration and Accuracy Evaluation of a Hybrid Self-Tracker. In *International Symposium on Mixed and Augmented Reality*, 2003.

[Fryer and Goodin, 1989] J.G. Fryer and D.J. Goodin. In-Flight Aerial Camera Calibration from Photography of Linear Features. *Photogrammetric Engineering and Remote Sensing*, 55(12):1751–1754, 1989.

[Genc *et al.*, 2002] Y. Genc, S. Riedel, F. Souvannavong, and N. Navab. Marker-Less Tracking for Augmented Reality: a Learning-Based Approach. In *International Symposium on Mixed and Augmented Reality*, 2002.

[Geyer and Daniilidis, 2003] C.M. Geyer and K. Daniilidis. Omnidirectional Video. *The Visual Computer*, 19(6):405–416, October 2003.

[Grassia, 1998] F.S. Grassia. Practical Parameterization of Rotations Using the Exponential Map. *Journal of Graphics Tools*, 3(3):29–48, 1998.

[Hager and Belhumeur, 1998] G.D. Hager and P.N. Belhumeur. Efficient Region Tracking With Parametric Models of Geometry and Illumination. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(10):1025–1039, 1998.

[Harris and Stephens, 1988] C.G. Harris and M.J. Stephens. A Combined Corner and Edge Detector. In *Fourth Alvey Vision Conference*, 1988.

[Harris, 1992] C. Harris. *Tracking With Rigid Objects*. MIT Press, 1992.

[Heikkila and Silven, 1997] J. Heikkila and O. Silven. A Four-Step Camera Calibration Procedure With Implicit Image Correction. In *Conference on Computer Vision and Pattern Recognition*, pages 1106–1112, 1997.

[Hoff *et al.*, 1996] W. A. Hoff, K. Nguyen, and T. Lyon. Computer Vision-Based Registration Techniques for Augmented Reality. In *Proceedings of Intelligent Robots and Control Systems XV, Intelligent Control Systems and Advanced Manufacturing*, pages 538–548, November 1996.

[Jiang *et al.*, 2004] B. Jiang, U. Neumann, and S. You. A Robust Tracking System for Outdoor Augmented Reality. In *IEEE Virtual Reality Conference 2004*, 2004.

[Jurie and Dhome, 2001] F. Jurie and M. Dhome. A Simple and Efficient Template Matching Algorithm. In *International Conference on Computer Vision*, July 2001.

[Jurie and Dhome, 2002] F. Jurie and M. Dhome. Hyperplane Approximation for Template Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):996–100, July 2002.

[Jurie, 1998] F. Jurie. Tracking Objects With a Recognition Algorithm. *Journal of Machine Learning Research*, 3-4(19):331–340, 1998.

[Kato and Billinghurst, 1999] H. Kato and M. Billinghurst. Marker Tracking and Hmd Calibration for a Video-Based Augmented Reality Conferencing System. In *IEEE and ACM International Workshop on Augmented Reality*, October 1999.

[Kato *et al.*, 2000] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. Virtual Object Manipulation on a Table-Top Ar Environment. In *International Symposium on Augmented Reality*, pages 111–119, 2000.

[Klein and Drummond, 2003] G. Klein and T. Drummond. Robust Visual Tracking for Non-Instrumented Augmented Reality. In *International Symposium on Mixed and Augmented Reality*, pages 36–45, October 2003.

[Koller *et al.*, 1997] D. Koller, G. Klinker, E. Rose, D.E. Breen, R.T. Whitaker, and M. Tuceryan. Real-Time Vision-Based Camera Tracking for Augmented Reality Applications. In *ACM Symposium on Virtual Reality Software and Technology*, pages 87–94, September 1997.

[Lepetit and Fua, 2006] V. Lepetit and P. Fua. Keypoint Recognition Using Randomized Trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1465–1479, September 2006.

[Lepetit *et al.*, 2005] V. Lepetit, P. Lagger, and P. Fua. Randomized Trees for Real-Time Keypoint Recognition. In *Conference on Computer Vision and Pattern Recognition*, June 2005.

[Li *et al.*, 1993] H. Li, P. Roivainen, and R. Forchheimer. 3D Motion Estimation in Model-Based Facial Image Coding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):545–555, June 1993.

[Lindeberg, 1994] T. Lindeberg. Scale-Space Theory: a Basic Tool for Analysing Structures at Different Scales. *Journal of Applied Statistics*, 21(2):224–270, 1994.

[Lowe, 1991] D. G. Lowe. Fitting Parameterized Three-Dimensional Models to Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(5):441–450, June 1991.

[Lowe, 1999] D.G. Lowe. Object Recognition from Local Scale-Invariant Features. In *International Conference on Computer Vision*, pages 1150–1157, 1999.

[Lowe, 2004] D.G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 20(2):91–110, 2004.

[Marchand *et al.*, 2001] E. Marchand, P. Bouthemy, and F. Chaumette. A 2d-3d Model-Based Approach to Real-Time Visual Tracking. *Journal of Image and Vision Computing*, 19(13):941–955, 2001.

[Matas *et al.*, 2002] J. Matas, O. Chum, U. Martin, and T. Pajdla. Robust Wide Baseline Stereo from Maximally Stable Extremal Regions. In *British Machine Vision Conference*, pages 384–393, September 2002.

[Mikolajczyk and Schmid, 2002] K. Mikolajczyk and C. Schmid. An Affine Invariant Interest Point Detector. In *European Conference on Computer Vision*, pages 128–142, 2002.

[Mikolajczyk and Schmid, 2003] K. Mikolajczyk and C. Schmid. A Performance Evaluation of Local Descriptors. In *Conference on Computer Vision and Pattern Recognition*, pages 257–263, June 2003.

[Mikolajczyk *et al.*, 2005] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A Comparison of Affine Region Detectors. *International Journal of Computer Vision*, 65(1/2):43–72, 2005.

[Mindru *et al.*, 1999] F. Mindru, T. Moons, and L. Van Gool. Recognizing Color Patterns Irrespective of Viewpoint and Illumination. In *Conference on Computer Vision and Pattern Recognition*, pages 368–373, 1999.

[Moravec, 1977] H.P. Moravec. Towards Automatic Visual Obstacle Avoidance. In *International Joint Conference on Artificial Intelligence*, page 584, August 1977.

[Moravec, 1981] H. Moravec. *Robot Rover Visual Navigation*. UMI Research Press, 1981.

[Nayar *et al.*, 1996] S. K. Nayar, S. A. Nene, and H. Murase. Real-Time 100 Object Recognition System. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(12):1186–1198, 1996.

[Ope,] Open Source Computer Vision Library. http://www.intel.com/technology/computing/opencv/.

[Pilet *et al.*, 2005a] J. Pilet, V. Lepetit, and P. Fua. Augmenting Deformable Objects in Real-Time. In *International Symposium on Mixed and Augmented Reality*, October 2005.

[Pilet *et al.*, 2005b] J. Pilet, V. Lepetit, and P. Fua. Real-Time Non-Rigid Surface Detection. In *Conference on Computer Vision and Pattern Recognition*, June 2005.

[Ravela *et al.*, 1995] 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*, pages 174–180, 1995.

[Rekimoto, 1998] J. Rekimoto. Matrix: a Realtime Object Identification and Registration Method for Augmented Reality. In *Asia Pacific Computer Human Interaction*, 1998.

[Ribo and Lang, 2002] P. Ribo and P. Lang. Hybrid Tracking for Outdoor Augmented Reality Applications. In *Computer Graphics and Applications*, pages 54–63, 2002.

[Schaffalitzky and Zisserman, 2002] F. Schaffalitzky and A. Zisserman. Multi-View Matching for Unordered Image Sets, Or"how Do I Organize My Holiday Snaps?". In *European Conference on Computer Vision*, pages 414–431, 2002.

[Schmid and Mohr, 1997] C. Schmid and R. Mohr. Local Grayvalue Invariants for Image Retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–534, May 1997.

[Se *et al.*, 2002] S. Se, D. G. Lowe, and J. Little. Mobile Robot Localization and Mapping With Uncertainty Using Scale-Invariant Visual Landmarks. *International Journal of Robotics Research*, 22(8):735–758, 2002.

[Shi and Tomasi, 1994] J. Shi and C. Tomasi. Good Features to Track. In *Conference on Computer Vision and Pattern Recognition*, June 1994.

[Simon and Berger, 1998] G. Simon and M.-O. Berger. A Two-Stage Robust Statistical Method for Temporal Registration from Features of Various Type. In *International Conference on Computer Vision*, pages 261–266, January 1998.

[Skrypnyk and Lowe, 2004] I. Skrypnyk and D. G. Lowe. Scene Modelling, Recognition and Tracking With Invariant Image Features. In *International Symposium on Mixed and Augmented Reality*, pages 110–119, November 2004.

[Smith and Brady, 1995] S. M. Smith and J. M. Brady. Susan – A New Approach to Low Level Image Processing. Technical report, Oxford University, 1995.

[State *et al.*, 1996] A. State, G. Hirota, T. David, W.F. Garett, and M.A. Livingston. Superior Augmented-Reality Registration by Integrating Landmark Tracking and Magnetic Tracking. In *ACM SIGGRAPH*, pages 429–438, August 1996.

[Sturm and Maybank, 1999] P. Sturm and S. Maybank. On Plane-Based Camera Calibration: a General Algorithm, Singularities, Applications. In *Conference on Computer Vision and Pattern Recognition*, pages 432–437, June 1999.

[Swaminathan and Nayar, 2003] R. Swaminathan and S. K. Nayar. A Perspective on Distortions. In *Conference on Computer Vision and Pattern Recognition*, June 2003.

[Szeliski and Kang, 1994] R. Szeliski and S.B. Kang. Recovering 3D Shape and Motion from Image Streams Using Non Linear Least Squares. *Journal of Machine Learning Research*, 5(1):10–28, 1994.

[Tordoff *et al.*, 2002] B. Tordoff, W.W. Mayol, T.E. de Campos, and D.W. Murray. Head Pose Estimation for Wearable Robot Control. In *British Machine Vision Conference*, pages 807–816, 2002.

[Tsai, 1987] R.Y. Tsai. A Versatile Cameras Calibration Technique for High Accuracy 3D Machine Vision Mtrology Using Off-The-Shelf Tv Cameras and Lenses. *Journal of Robotics and Automation*, 3(4):323–344, 1987.

[Tuytelaars and Gool, 2000] T. Tuytelaars and L. Van Gool. Wide Baseline Stereo Matching Based on Local, Affinely Invariant Regions. In *British Machine Vision Conference*, pages 412–422, 2000.

[Vacchetti *et al.*, 2004a] L. Vacchetti, V. Lepetit, and P. Fua. Combining Edge and Texture Information for Real-Time Accurate 3D Camera Tracking. In *International Symposium on Mixed and Augmented Reality*, November 2004.

[Vacchetti *et al.*, 2004b] L. Vacchetti, V. Lepetit, and P. Fua. Stable Real-Time 3D Tracking Using Online and Offline Information. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(10):1385–1391, October 2004.

[Viola and Jones, 2001] P. Viola and M. Jones. Rapid Object Detection Using a Boosted Cascade of Simple Features. In *Conference on Computer Vision and Pattern Recognition*, pages 511–518, 2001.

[Zhang *et al.*, 1995] Z. Zhang, R. Deriche, O. Faugeras, and Q. Luong. A Robust Technique for Matching Two Uncalibrated Images through the Recovery of the Unknown Epipolar Geometry. *Artificial Intelligence*, 78:87–119, 1995.

[Zhang, 2000] Z. Zhang. A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.