# Reliable automatic calibration of a marker-based position tracking system

David Claus and Andrew W. Fitzgibbon
Department of Engineering Science,
University of Oxford, Oxford OX1 3BN
{dclaus,awf}@robots.ox.ac.uk

## Abstract

*This paper describes an accurate vision-based position tracking system which is significantly more robust and reliable over a wide range of environments than existing approaches. Based on fiducial detection for robustness, we show how a machine-learning approach allows the development of significantly more reliable fiducial detection than has previously been demonstrated. We calibrate fiducial positions using a structure-from-motion solver. We then show how nonlinear optimization of the camera position during tracking gives accuracy comparable with full bundle adjustment but at significantly reduced cost.*

## 1. Introduction

Accurate outdoor and indoor tracking remains a key requirement for augmented reality, and despite significant research effort in markerless approaches [16, 6], fiducial-based tracking is still the technology of choice for construction of AR systems which must provide repeatable and reliable performance. Fiducial detection, for example using the popular ARToolkit software [11], is reliable, and pose estimation from detected targets is robust and accurate. However, two significant challenges remain, and are the topics addressed by this paper:

First, fiducial-based position tracking requires, for optimal accuracy, that at least three fiducials are visible from all points in the environment. Therefore, a large number of fiducials must be placed at known locations throughout the workspace. Surveying fiducial positions is tedious and prone to error. We propose to use an offline camera tracker to automatically generate survey data for fiducials. This means that fiducials can be placed arbitrarily throughout the environment, greatly simplifying the setup of new environments.

Secondly, off-the-shelf fiducial detectors are still restricted in the classes of scenes in which they perform well. Figure 1 shows a range of indoor and outdoor scenes in which we would like to operate. The system in this paper is based on Claus's recently introduced fiducial detector [5] which significantly outperforms ARToolkit in these scenes.

**Figure 1. Example environments in which accurate tracking is required. We can place fiducials in arbitrary positions, automatically survey their positions and calibrate the camera offline. At run time, we can detect them with higher reliability than existing systems, and compute accurate camera positions.**

## 2. Previous work

The use of fiducials for pose estimation is by now an established technique in augmented reality. For most described applications, however, the required pose is that of a single target in the camera's field of view, for example a planar card on which augmentations are superimposed, or a single fiducial in a room [15]. We are interested in determining the camera position from multiple fiducials po-

sitioned in the environment. Kalkusch *et al* [10] describe a system where markers are placed throughout an extensive indoor environment. The position of each marker is physically measured and related to a CAD model of the environment. This is a time-consuming task, in which it is difficult to preclude the possibility of error. Wagner [18] describes another system in which markers are used to separate a large environment into rooms, but the emphasis is on the topological problem of determining which room the camera is in, rather than on accurate position estimation within rooms.

In this paper, we wish to determine camera pose from images of fiducials, and in order to obtain an accurate pose, it is important to use more than a single fiducial. Thus the positions of all fiducials must be known in the same reference frame, and thus an initial calibration stage is required. Aliaga and Carlbrom [2] describe a system where the initial calibration information is refined using bundle adjustment, however the fiducials' world locations must be provided *a priori*. Thomas *et al* [17] describe a system which also uses circular fiducials and an optical calibration procedure to refine the approximate physical measurements. In our system, all initial calibration is automatic.

In the following we describe each component of the system in detail, and discuss the accuracy of this approach.

## 3. System overview

Before describing the system components in detail, we provide an overview of the system. An offline autocalibration stage precedes per-frame fiducial detection and pose estimation. The core of the system is a highly reliable fiducial detector based on machine learning. This allows us to identify to subpixel accuracy the image positions of the four discs on each fiducial. Figure 3 shows some example fiducials. In offline calibration the following steps are performed:

1. Distribute fiducials throughout the environment. These need not be placed in known positions, nor need they be aligned with any specific world features.
2. Capture video of the environment, with the aim of viewing as much of the operating volume as possible.
3. Detect fiducials in all frames of the calibration sequence. Since each fiducial can be uniquely identified, this establishes image point matches between frames.
4. Use an off-the-shelf structure and motion solver to determine camera calibration (focal length and lens distortion) as well as 3D scene geometry for the fiducials. We used *boujou* from 2d3 [1], but the techniques are well established [9] and can be implemented locally if desired. This assigns 3D coordinates to all fiducials, without any user measurement. Calibration is now complete.

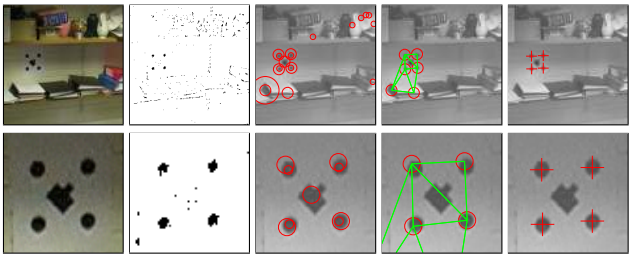


**Figure 2. Overall algorithm to detect fiducials. (a) Input image, (b) output from the fast classifier stage, (c) output from the full classifier superimposed on the original image. Every pixel has now been labelled as fiducial or non-fiducial. The size of the circles indicates the scale at which that fiducial was detected. (d) The target verification step rejects non-target fiducials through photometric and geometric checks. (e) Fiducial coordinates computed to subpixel accuracy.**

5. (Optional) If augmentations are to be placed in a room-referenced coordinate system, align the coordinate systems using one of several methods provided by the structure and motion solver. For example, by identifying three points with known world coordinates, or by identifying three known planes (floors, walls, etc.).

At run-time, position estimation follows the conventional strategy. For each frame of captured video, the following steps are performed.

1. Detect fiducials in each frame, and look up world coordinates for each. In our system, each fiducial comprises four discs on a black background, so four 2D point measurements are obtained for each.
2. Compute the camera pose which relates the 3D and 2D measurements. As four points is the minimum requirement for a unique solution, only one fiducial need be detected to obtain a pose. It is more accurate, however, to detect as many fiducials as possible, because the small angle subtended by the 2D points on a single fiducial yields inaccurate 3D pose.

We now proceed to describe the system components in detail. We first review our fiducial detection scheme, which is designed to offer higher tolerance to lighting and scale change than existing systems. We then discuss pose estimation from multiple targets, and show that a nonlinear refinement of pose offers significant accuracy gains for little computational cost.

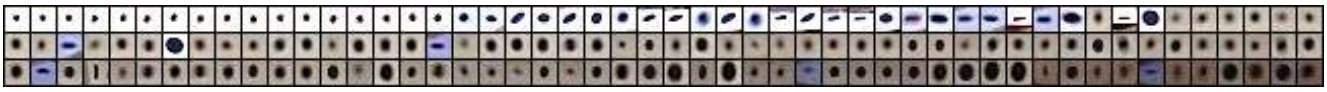

**Figure 3. Representative samples of positive target images. There is a wide variation of marker appearances, which are learned by the system in order to build a robust detector.**

## 4. Fiducial detection

Fiducial detection in modern AR systems is almost invariably achieved using the ARToolkit software [11]. This is a freely available system which allows real-time detection of coded targets, and which provides excellent performance providing lighting conditions are suitable. However, in difficult lighting conditions, such as outdoors or in low light, the toolkit's detection performance drops. In addition, low resolution images, extreme foreshortening, motion blur, and specular reflections all cause performance to drop. Modifications such as the use of adaptive thresholding [14] and homomorphic image processing [13] alleviate some of these problems, but each modification imposes additional computational cost, and introduces additional failure modes to the algorithm. In contrast, we propose that the most effective way to build a fiducial detector is to use machine learning techniques. Essentially we take an algorithm which at first sight is impossibly inefficient—"compare every $12 \times 12$ subwindow of every video frame to every possible fiducial image"—and make it run at video rate.

Our fiducials are composed of the simplest possible primitives: a black dot against a white background. A given marker will have four dots arranged in a square, with area in the centre of the dot for barcodes, icons, or other identification aids. We require that fiducials are detected at all scales, and anywhere in the image, at video rate. The first stage of fiducial detection is to detect candidates in the image. In order to obtain invariance to all of the image degradations described above, we prepared a library of example fiducials by capturing video sequences of the markers in a variety of indoor and outdoor environments, some frames from which are shown in Figure 1. Example fiducial images are shown in Figure 3, including variation due to foreshortening, motion blur, and lighting. The imaged fiducials are all scaled to fit into a $12 \times 12$ window. In total, 8506 example fiducial images were gathered. In addition, 19052 examples of non-fiducial windows were drawn at random from the training sequences.

Given the training images, an ideal but impractical version of the algorithm for fiducial detection is as follows:

Given:
  Input image $I(x, y)$
  Set of $N^+$ positive examples $PositiveExamples$
  Set of $N^-$ negative examples $NegativeExamples$

for $scale = 1 : 4$
  S = reduce($I$, $scale$);
  for $(x, y) \in \{[6..width(S) - 6] \times [6..height(S) - 6]\}$
    $Window$ = 12x12 window of S centred on (x,y)
    **[*]**
    $best\_positive\_score = best\_negative\_score = \infty$
    for $k = 1 : N^+$
      $best\_positive\_score = \min(best\_positive\_score,$
        $\|Window - PositiveExamples(k)\|)$
    for $k = 1 : N^-$
      $best\_negative\_score = \min(best\_negative\_score,$
        $\|Window - NegativeExamples(k)\|)$
    if $best\_positive\_score < best\_negative\_score$
      Report fiducial at (x,y,scale)

A nearest-neighbour classifier is run over every $12 \times 12$ subwindow of the image at every possible scale. Given that in our system $N^+ = 8506$ and $N^- = 19052$, this would take hours per frame, which is clearly of little use. In practice, this can run at video rate with two simple modifications. The first is to insert a fast classifier at the line marked **[*]** which quickly rejects windows which are unlikely to be fiducial candidates. This test is tuned to have few false negatives (of the order of 1%), and reduces the number of windows to be tested by a factor of about 3000. The second modification is to use nearest-neighbour condensing [8] to reduce the total number of nearest-neighbour comparisons. This produces a classifier with the same performance as the full classifier, but using many fewer examples, which is thus much faster. In our application, the number of training examples ($N^+ + N^-$) was reduced from 27558 to 382, a speed increase of two orders of magnitude. It is worth noting that for patterns of this dimensionality, the use of $k$-d trees to accelerate the nearest-neighbour search provided little further speed improvement.

The above algorithm operates at video rate, and emits of the order of 100 fiducial candidates per frame. In order to eliminate the false positives, we use a geometric check based on the assumption that a Delaunay triangulation of the returned fiducial positions will always include the true target in the configuration shown in Figure 2(d). In practice this is a very robust assumption, failing only with extreme perspective effects, or when a false positive is detected very close to the true target. After all stages of the algorithm, typical performance is accurate detection of the target across

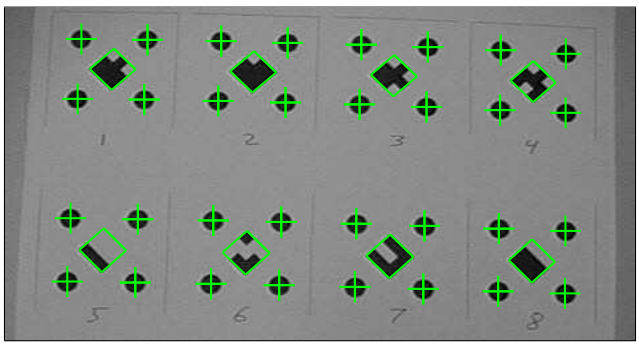

**Figure 4. Multiple fiducials printed out on a single sheet of paper. The white overlay shows the detected marker positions and the squares bounding the marker identification codes. The target detection and assignment of the identification patterns is performed automatically from a single frame of video.**

all environments in about $95\%$ of frames, with a false positive on average every 500–1000 frames.

The task of identifying which target has been found is again handled through a nearest neighbour (sum of squared differences) match with a set of trained patterns (Figure 4). These patterns are easily recorded using the detection algorithm itself. The target dots are automatically located, and the registration pattern is recorded for each target found. In our tests, the patterns consist of non-symmetric binary patterns in a $3 \times 3$ grid, but in general any pattern could be used. A normalization step is used to handle differences in lighting conditions. A higher degree of robustness can be obtained by recording training images under a wide variety of lighting conditions [5], however in this case we want to minimize the amount of storage and computational overhead associated with the target recognition. Therefore a single image is recorded at the expense of some robustness.

## 5. Pose estimation

The key to accurate pose estimation is accurate surveying of the markers in the environment. We automate this difficult task by using an off-the-shelf structure and motion system [1]. Such systems are frequently used for extremely demanding augmented reality tasks in cinema post-production, where virtual objects must be added to real-world footage with jitter of the order of half a pixel in a $4096 \times 3312$ image, and drift over hundreds of frames of the order of a few pixels. However, because they depend on batch computation and bundle adjustment [9], they are unsuitable for online operation, although ideal for offline calibration and surveying.

To survey the environment, a video of the AR workspace is captured, ensuring that each marker of interest is visible in from at least two widely-spaced viewpoints. The structure and motion solver returns a camera trajectory for the calibration sequence, as well as calibration parameters such as lens distortion and focal length. The camera position for a given image is represented as a $3 \times 4$ projection matrix

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

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

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

The structure and motion solver returns a single $\mathtt{K}$ for the entire sequence, and per-frame rotation and translation estimates.

### 5.1. Computing fiducial positions

In surveying the fiducials, we independently compute the 3D position of each of the four discs on each target. This allows for deviations from planarity in paper targets.

In order to determine the position of a particular disc, we gather its 2D positions in every frame in which it was detected as a list of $(x, y, f)$ tuples, where $(x, y)$ is the marker's position in frame $f$. Then the 3D point we require is that which minimizes the reprojection error [9]

$$\epsilon(\boldsymbol{X}) = \sum_{i=1}^{\text{num\_detections}} \left\| \begin{pmatrix} x_i \\ y_i \end{pmatrix} - \pi\left(\mathtt{K}(\mathtt{R}_{f_i}\mathbf{X} + \mathbf{t}_{f_i})\right) \right\|$$

where the 3D-to-2D perspective projection function is defined as $\pi(x, y, z) = (x/z, y/z)$. Although this minimization has no closed-form solution, it is readily solved by initialization using the DLT method [9] followed by nonlinear optimization of $\epsilon(\mathbf{X})$.

By repeating this procedure for all fiducials, we obtain accurate estimates of the fiducial positions.

### 5.2. Computing pose

Given the fiducial positions as computed above, we may now discard the calibration sequence, and use the fiducials in the usual manner for pose computation. In a given frame, 2D marker positions $\mathbf{x}_j$ are obtained, and the corresponding 3D locations $\mathbf{X}_j$ are looked up using the fiducial ID. Then the pose estimation problem is to find $\mathtt{R}$ and $\mathbf{t}$ to minimize the reprojection error

$$\epsilon_2(\mathtt{R}, \mathbf{t}) = \sum_j \left\| \mathbf{x}_j - \pi\left(\mathtt{K}(\mathtt{R}\mathbf{X}_j + \mathbf{t})\right) \right\|$$

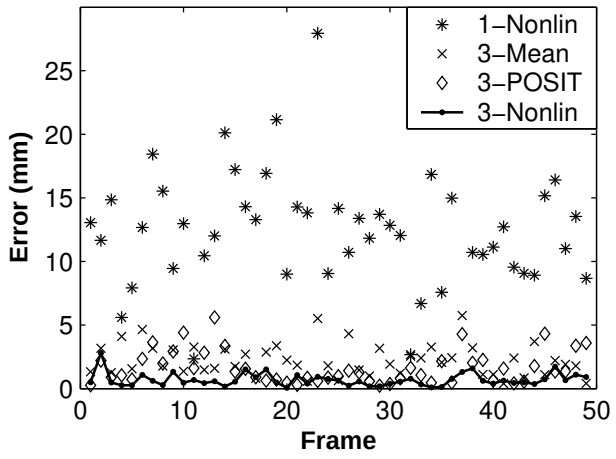


**Figure 5. Errors in camera position computed by various methods. Plot shows absolute error for a sample set of 50 frames.**

| Targets | Method | RMS error (mm) |
|---|---|---|
| 1 | Non-linear reprojection min. | 130 |
| 3 | ARToolkit (see §5.4) | 43† |
| 3 | Mean of three positions | 25 |
| 3 | POSIT | 29 |
| 3 | Non-linear reprojection min. | 9.7 |

† Omitting the 12% of frames where detection failed.

**Table 1. Camera position error relative to ground truth. Non-linear minimization of the reprojection errors for only twelve fiducial markers (three targets) provides centimeter level camera position accuracy.**

Some small observations are pertinent at this stage. First, on a 1GHz PC it is possible to run such a nonlinear optimization very quickly, of the order of one millisecond. For accurate pose estimation, multiple targets are required. A single target (four discs) can provide a pose estimate, but will not in general be accurate, particularly if the target is far from the camera. One often proposed method for obtaining a more accurate camera pose is to compute the average of the solutions obtained from several individual targets. This does provide an improvement over the single target solution (Figure 5 and Table 1), but further reductions are possible through simultaneous estimation. Initialization of the nonlinear iteration can be performed by an algorithm such as POSIT [7], and then followed with non-linear optimization to minimize the reprojection error.

The calibration parameters obtained from the initial sequence were then used to compute the camera pose from fiducial tracks in a second video sequence of the same environment. In order to compare against a measure of ground truth, we also used *boujou* to compute a bundle-adjusted path for the same sequence. As *boujou*'s second run chooses an arbitrary coordinate system, independent of that chosen in the first sequence, we aligned the two paths using a similarity transform, computed using the method of Arun et al [3], before making the comparison.

### 5.3. Zoom lenses

If the AR camera is changing focal length during tracking, this is also readily included in the optimization for pose, although with a reduction in position accuracy. To check the accuracy of this computation, we ran pose estimation including variable focal length on our fixed-focal-length ground-truth sequence. This allowed us to compare the focal length from the MATLAB Camera Calibration Toolbox [4] (846.6 pixels), *boujou*'s fully bundle-adjusted solution (848.7 pixels), the mean estimates from individual frames (849.7 pixels), and sequences of frames (849.1 pixels). If the focal length is known to be fixed it can be obtained by optimizing over the entire sequence simultaneously; for realtime operations or variable focal length it is possible to estimate the focal length from individual frames.

### 5.4. ARToolkit comparison

A sample sequence was recorded with ARToolkit markers placed in the same physical locations as the fiducials described in the above experiments. The mean distance to the camera for this sequence was 3.33 m, which is on the edge of the operating range for the ARToolkit [12]. Training patterns for each marker were recorded under the same lighting and viewpoint conditions as used in the test in order to maximize the marker recognition rate. However, the system was unable to reliably track multiple markers at this scale. The square border detection is successful for all three targets in 88% of the frames, but the pattern identification fails on most frames. In order to make a three target comparison, we exported the points of every square pattern found then manually established the marker identifications and world correspondences. The path computed by non-linear refinement of the POSIT pose estimate is shown in Figure 6. As shown in Table 1, our camera tracking system yields a tenfold improvement in accuracy over the ARToolkit.

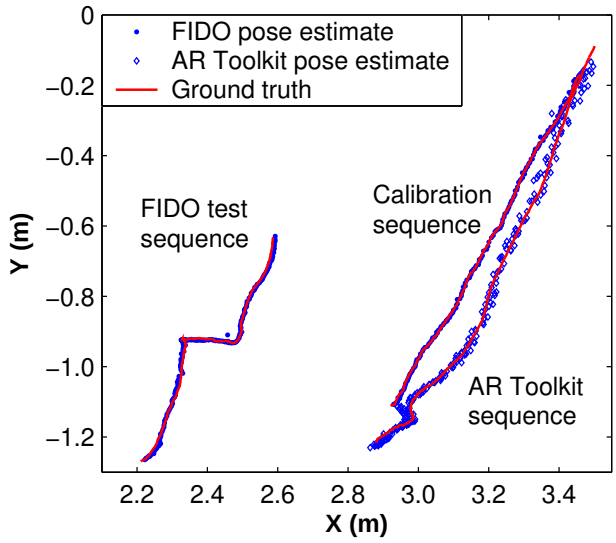


**Figure 6. Camera tracks for three separate sequences in the same calibrated environment. The method presented in this paper (FIDO) demonstrates a much lower position error as compared to the ground truth.**

### 5.5. Conclusions

We have introduced a vision-based tracking system which combines highly reliable fiducial detection with an off-the-shelf structure and motion algorithm to significantly improve the performance and ease of setup of camera tracking. Because the tracker performs a full scan of the image at every frame, recovery after occlusions or extreme motion blur is instantaneous. However, the system is of course not robust to the normal bugbears of vision-based tracking. If the targets are not detected, no pose is reported for that frame, so an inertial or other sensor will still be required in order to provide continuous output.

In our tests, we repeatedly compared against the bundle-adjusted positions instead of measured coordinates, as these are difficult to obtain. For some augmented reality tasks, this is reasonable: after all, cinema-quality augmentation would be more than adequate in many situations. On the other hand, many tracking applications require real-world coordinates, which may prove to be different from the structure-and-motion solution, for example if the camera focal length is misestimated. We hope to use a robot-mounted camera to determine the accuracy of this registration.

## References


[1] 2d3 Ltd. Boujou: Automated camera tracking, 2003. http://www.2d3.com.

[2] D. G. Aliaga and I. Carlbom. Fiducial planning for error-bounded pose estimation of a panoramic camera in large environments. *Special Issue of IEEE Robotics and Automation Magazine: Panoramic Robotics*, 2003.

[3] K. S. Arun, T. Huang, and S. D. Blostein. Least-squares fitting of two 3-D point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):698–700, 1987.

[4] J.-Y. Bouguet. Camera calibration toolbox for MATLAB. http://www.vision.caltech.edu/bouguetj/calib_doc/.

[5] D. Claus and A. W. Fitzgibbon. Reliable fiducial detection in natural scenes. In *European Conference on Computer Vision*, pages 469–480. Springer-Verlag, 2004.

[6] A. I. Comport, E. Marchand, and F. Chaumette. A real-time tracker for markerless augmented reality. In *Int'l Symposium on Mixed and Augmented Reality*, pages 36–45, 2003.

[7] D. DeMenthon and L. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15:123–141, 1995.

[8] P. Hart. The condensed nearest neighbor rule. *IEEE Trans. Information Theory*, 14:515–516, 1968.

[9] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision (Second Edition)*. Cambridge University Press, 2003.

[10] M. Kalkusch, T. Lidy, M. Knapp, G. Reitmayr, H. Kaufmann, and D. Schmalstieg. Structured visual markers for indoor pathfinding. In *Int'l AR Toolkit Workshop*, 2002.

[11] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana. Virtual object manipulation on a table-top AR environment. In *Int'l Symposium on Augmented Reality*, pages 111–119, 2000.

[12] P. Malbezin, W. Piekarski, and B. Thomas. Measuring ARToolkit accuracy in long distance tracking experiments. In *Int'l AR Toolkit Workshop*, 2002.

[13] L. Naimark and E. Foxlin. Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In *Int'l Symposium on Mixed and Augmented Reality*, 2002.

[14] T. Pintaric. An adaptive thresholding algorithm for the Augmented Reality Toolkit. In *Int'l AR Toolkit Workshop*, 2003.

[15] J. Rekimoto. Matrix: A realtime object identification and registration method for augmented reality. In *Asia Pacific Computer Human Interaction Conference*, 1998.

[16] G. Simon and M.-O. Berger. Reconstructing while registering : a novel approach for markerless augmented reality. In *Int'l Symposium on Mixed and Augmented Reality*, 2002.

[17] G. A. Thomas, J. Jin, T. Niblett, and C. Urquhart. A versatile camera position measurement system for virtual reality TV production. In *International Broadcasting Convention*, pages 284–289. IEE Conference Publication, 1997.

[18] M. Wagner. Building wide-area applications with the AR Toolkit. In *Int'l AR Toolkit Workshop*, 2002.