A Method for Predicting Marker Tracking Error

Russell M. Freeman*
University College London

Simon J. Julier†
University College London

Anthony J. Steed‡
University College London

ABSTRACT

Many Augmented Reality (AR) applications use marker-based vision tracking systems to recover camera pose by detecting one or more planar landmarks. However, most of these systems do not interactively quantify the accuracy of the pose they calculate. Instead, the accuracy of these systems is either ignored, assumed to be a fixed value, or determined using error tables (constructed in an off-line ground-truthed process) along with a run-time interpolation scheme. The validity of these approaches are questionable as errors are strongly dependent on the intrinsic and extrinsic camera parameters and scene geometry. In this paper we present an algorithm for predicting the statistics of marker tracker error in real-time. Based on the Scaled Spherical Simplex Unscented Transform (SSSUT), the algorithm is applied to the Augmented Reality Toolkit Plus (ARToolKitPlus). The results are validated using precision off-line photogrammetric techniques.

CR Categories and Subject Descriptors: I.4.8 [Image Processing and Computer Vision]: Scene Analysis – Tracking.
Additional Keywords: augmented reality, tracking, ARToolKit, unscented transforms.

1 INTRODUCTION

Performing fully automatic real-time vision-based camera tracking is an extremely challenging task. One way to simplify this problem is to populate the environment with one or more planar fiducial markers of known appearance [5]. The tracking problem is reduced to one of identifying the markers and computing their pose. Several mature tracking systems, including ARToolKit [7], ARToolKitPlus [8] and ARTag [9] have been developed and are being widely used within the augmented and mixed reality communities. In all these systems the pose is always recovered with a degree of error.

Quantifying such tracker errors is important for at least two reasons. First, error statistics can be used by applications to adapt their interfaces depending upon the level of error [6][18]. Second, if the recovered poses are to be fused using another estimation algorithm [18], such as a particle filter [11] or a Kalman filter [12], optimal performance can only be achieved if a statistical characterization of the errors in the estimates is known.

To this end, several studies have been carried out to characterize the errors of such marker tracking systems [1][2][3][4]. In all of these cases the characterizations are carried out empirically: the tracking system is used to measure the pose of a platform, whilst the real pose is calculated by an external and more accurate tracking or measurement system. Through comparing these two results the error characteristics can be derived. Such metrics have either divided up the space around the marker into known error zones or sought to find generalities about the tracker's overall performance. However, there are two major drawbacks with this approach. First, because the errors are highly dependent on the configuration of the markers and camera, [13], it is not clear that the errors calculated in one configuration can be generalized to a different configuration in a different situation. Second, the analysis has currently only been carried out for a few select tracking systems and would therefore need to be repeated for any further systems that might be developed or used.

Rather than rely on empirical methods, in this paper we develop a generic technique for predicting the statistics of tracking error in real-time. Based on the Scaled Spherical Simplex Unscented Transform (SSSUT) [10], we apply our algorithm to the ARToolKitPlus. However, we believe that the same approach can be applied to many other classes of tracking systems in which pose is recovered by passing a set of detected image parameters to a pose recovery process.

Section 2, describes the problem in more detail, including sources of errors and how they propagate through a tracking system. Section 3 then describes a possible solution that uses SSSUT to propagate known errors through nonlinear estimation algorithms. The experimental setup is briefly described in Section 4, before Section 5 uses the ARToolKitPlus marker tracker [8] to provide some results. Conclusions are drawn in Section 6.

2 MARKER TRACKING ISSUES

Using a set of easy-to-recognize fiducial markers, highly efficient algorithms can be employed to find their edges and corners in an image, before finally estimating their pose (rotation and translation) relative to a camera. Two widely used vision tracking algorithms, the ARToolKit [7] and ARToolKitPlus [8], use a visual intensity threshold algorithm to highlight a marker's edges. The four corners of the markers can then be assumed and the relative pose calculated. Figure 1 illustrates the processes and work flow involved; images are captured by a camera before a corner and edge detector extracts features. These features are finally passed through a nonlinear optimizer to calculate the relative pose.

Errors in computing the pose can arise from several sources. These include errors in detecting the visual parameters (sensor thermal noise, pixelization and quantization), calibration (incorrect intrinsic parameters), and errors in the pose recovery algorithm itself (many algorithms use optimization schemes and can become trapped in local minima [14]).

Several studies [1][2] model how errors exhibit themselves in ARToolKit [7]. These works seek to form off-line error tables or generalities that can be 'hard-coded' into subsequent AR applications in the form of reference Look-Up-Tables (LUTs). Such LUTs define ranges and zones inside which an estimated error metric can be assumed. However, two issues exist with the application of these tables. First, the tables do not account for calibration issues. Errors or differences in the cameras calibration can lead to large variations in how errors may exhibit themselves in the final pose solutions. Second, LUTs are computed for specific poses of the camera. It is not clear how to interpolate or extrapolate the errors between the measured error zones.

For these and many other reasons, a more generalized approach for predicting the potential estimation errors in real-time is highly desirable.

*e-mail: r.freeman@cs.ucl.ac.uk
†e-mail: s.julier@cs.ucl.ac.uk
‡e-mail: a.steed@cs.ucl.ac.uk

978-1-4244-1750-6/07/$25.00 ©2007 IEEE

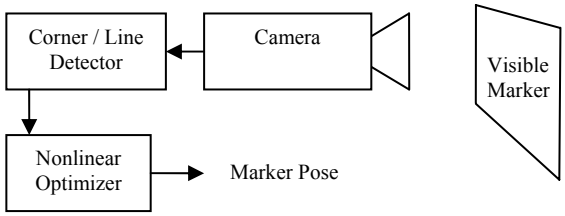

Figure 1. Basic steps in the normal pose recovery algorithm.

3 ADAPTIVE ERROR CALCULATION SCHEME

In this paper we consider the effects of inaccuracies in the identification of the marker's corners and edges. Although the magnitudes of these errors are often relatively small, less than a single pixel in the 2D image space (*see Section 3.1*), they can lead to much larger errors in the resulting pose estimations.

Because the mapping between detected corners and edges to pose is extremely complex, and involves multiple nonlinear transformations with an iterative optimizer performing in a closed form, analytical methods are extremely difficult to derive. Therefore we have adopted an alternative numerical method, illustrated in Figure 2.

3.1 The Numerical Sampling Technique

Our approach is based on the Spherical Symmetric Unscented Transformation (SSSUT) [10]. Like Unscented Transforms (UT), it calculates resultant estimates correctly to a second order. The advantage of using this sampling technique is that for an *n*-dimensional space only *n*+2 points are needed, offering significant computational savings. In our application there are ample computational resources and so such savings are not significant, although it has the potential to be used in more complicated environments and situations.

Producing *n*+2 sample points (known as sigma points), which lie on an *n*-dimensional hypersphere with fixed radius *r*, the standard UT algorithm is as follows:

1) Choose $0 \le W_0 \le 1$.
2) Choose weight sequence:
$$W_i = \frac{1-W_0}{n+1}$$
3) Initialize vector sequence as:
$$x_0^1 = [0], x_1^1 = \begin{bmatrix} -\frac{1}{\sqrt{2W_1}} \end{bmatrix} \text{ and } x_2^1 = \begin{bmatrix} \frac{1}{\sqrt{2W_1}} \end{bmatrix}$$
4) Expand vector sequence for $j = 2,\ldots,n$ according to
$$x_i^j = \begin{cases} \begin{bmatrix} x_0^{j-1} \\ 0 \end{bmatrix} & \text{for } i = 0 \\ \begin{bmatrix} x_i^{j-1} \\ -\frac{1}{\sqrt{j(j+1)W_1}} \end{bmatrix} & \text{for } i = 1,\ldots,j \\ \begin{bmatrix} \mathbf{O}_j \\ \sqrt{j(j+1)W_1} \end{bmatrix} & \text{for } i = j+1 \end{cases}$$

A notable feature of this algorithm is that the weight on each sigma point (apart from the 0th point) is the same and proportional to $(1-W_0)/(n+1)$. Also, all of the sigma points (apart from the 0th point) lie on the hypersphere of radius $\sqrt{n}/(1-W_0)$. For the case of computing camera pose, $n = 8$, 10 points are required, and the points are all chosen to perturb pixels with radius $2\sqrt{2}/(1-W_0)$.

The effects of this sampling strategy are shown in figure 3, where a family of proposed target shapes is produced. Although

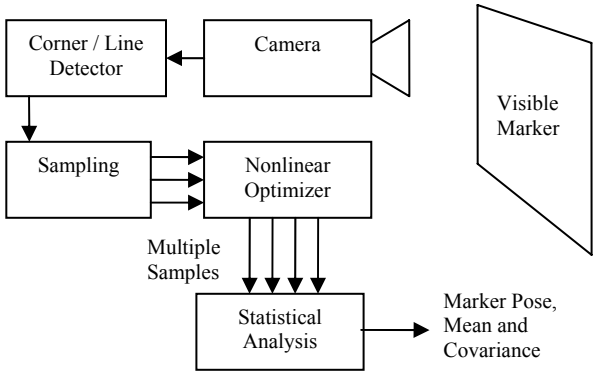

Figure 2. The error numerical calculation scheme intercepts the optimization step and calls it repeatedly with to compute the statistics of the optimizer.

the displacements used to generate this figure are an order of magnitude larger than the actual points generated, it does illustrate another problem; the pixel perturbations can be large compared to the actual detection errors.

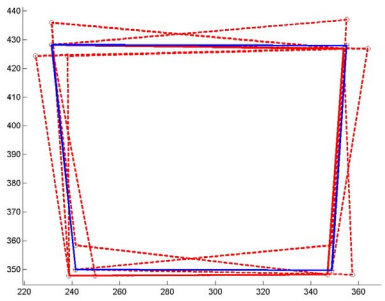

Figure 3. The solid outline is the polygon formed by the pixel coordinates of the detected corner positions. The dashed lines are polygons drawn with various perturbed configurations. Note that for the purposes of this figure the points are sampled with W_0 = 0.9 to magnify the displacement by a factor of 10.

If $W_0 = 0$ (the value we actually use in the main algorithm), the perturbed points all lie on a sphere of radius $2\sqrt{2} \approx 2.6$ pixels. However, detection errors are often sub-pixel (we found an average radial error of 0.819 pixels, with a standard deviation of 0.161, in 120 images of a target marker using the ARToolKitPlus, see Section 4), consequently using such a large radial displacement will often lead to spurious results due to nonlinear 'flipping' of the solutions [14]. To reduce this sample radius to a more realistic scale we therefore used the scaled unscented transformation instead of the standard unscented transformation to compute the error statistics.

3.2 Scaled Unscented Transformation

Suppose a set of sigma points δ have been constructed with mean $\bar{x}$ and covariance Σ_x. The same mean can be calculated by applying a different sigma point set $\tilde{\delta} = \left\{ 0,\ldots,p : \tilde{x}^{(i)}, \tilde{w}^{(i)} \right\}$ to $h[\cdot]$ [10]. $\tilde{\delta}$ and δ are related according to the expression:

$$\tilde{x}^i = \alpha x^i + (1-\alpha) x^0$$
$$\tilde{W}^i = \begin{cases} \left(W^0 + \mu - 1 \right) / \mu & i = 0 \\ W^i / \mu & i \ne 0 \end{cases}$$

Given this set of points, the scaled Unscented Transform calculates its statistics as follows:

$$\bar{z}^i = h\left[\bar{x}^i\right]$$

$$\bar{z} = \sum_{i=0}^{p} \bar{W}^i \bar{z}^i$$

$$\Sigma_z = \sum_{i=0}^{p} \bar{W}^i \left\{\bar{z}^i - \bar{z}\right\}\left\{\bar{z}^i - \bar{z}\right\}^T + \left(1-\alpha^2\right)\left\{\bar{z}^0 - \bar{z}\right\}\left\{\bar{z}^0 - \bar{z}\right\}^T$$

Using this scaled unscented transform we are able to produce sample sigma points with a significantly smaller radius, more in line with what might be expected for the corner detection errors in a marker tracking systems image space.

4 Experimental Setup

To demonstrate the effectiveness of our proposed technique we captured 120 images of a target marker placed on a calibration rig at varying distances and incident angles and orientations.

During the image capture process the camera was held and moved around the marker in approximately 15º increments. The horizontal and vertical distances were also varied from approximately 1*m* to 2*m* and 20*cm* to 50*cm* respectively. The square marker's edge length was 23*cm,* and was stuck onto a rigid flat board. The camera used for the experiment was a simple USB Logitech QuickCam Pro 5000 at a resolution of 640x480 pixels.

4.1 Camera Calibration and Ground Truth

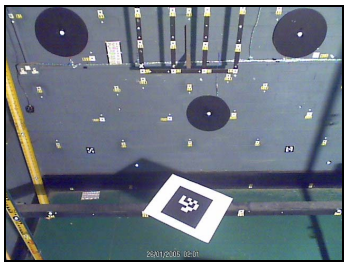

Figure 4. Target marker visible on calibration rig.

To provide an accurate result we captured the 120 images within a highly calibrated environment (*see Figure 4*). The intrinsic parameters of the camera and accurate ground truth of the relative camera to marker pose was created using precision photogrammetry software, the Vision Measurement System (VMS) [16][17].

4.2 ARToolKitPlus Application

Using the ARToolKitPlus [8] *C++* marker tracking library we created a lightweight application that could take a set of images to find both the normally estimated ARToolKitPlus pose for any visible markers, and also calculate our SSSUT error prediction method in real-time. To allow us to add perturbation values to the detected marker corners we made some minor alterations to the ARToolKitPlus library; once the edges and corners are found, but before the optimizer has been engaged, we adjusted the toolkit to enable us to add our perturbation values and run the optimizer several times.

The set of computed perturbed pose estimates were converted into separate 3D positions and quaternion rotations, before a linear average was computed. Although this is not strictly the correct way to average quaternion rotations [15], we felt this approximation was valid for the initial development of our algorithm.

Two configurations of the ARToolkitPlus application were tested. The first explored the results of the default ARToolKitPlus pose recovery algorithm, whilst the second exploited the Robust Planar Pose algorithm (RPP) [14]. The implementation was altered to exploit the pose algorithm's ability to take an initial estimate. Specifically, the 0^{th} (unperturbed nominal) solution computed was used as the initial estimate for all subsequent pose calculations. Due to the limited space of this paper, we will only present the results of the RPP algorithm.

5 Results

The average corner position perturbation radius was chosen to be 1.638 pixels, exactly twice the average actual corner detection error found in the 120 images by ARToolKitPlus when compared to the ground truth. Figure 5 displays the calculated camera to marker distance for the ground truth, the ARToolKitPlus estimation and our SSSUT predicted error results for the complete set of 120 images.

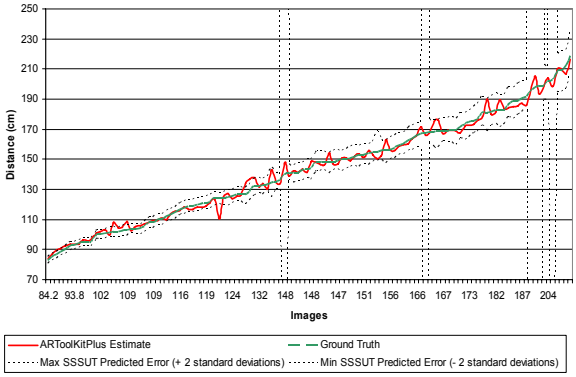

Figure 5. Results, calculated with the RPP optimizer, for the direct distance to marker. Images sorted into order of distance to marker.

To better illustrate the predicted errors in the above estimates, figure 6 shows the differences of the results from the ground truth. As might be expected, we can now clearly see that the error variation of the results predicted by our SSSUT method covers the variation in the ARToolKitPlus estimation accuracy by keeping the zero line within its bounds.

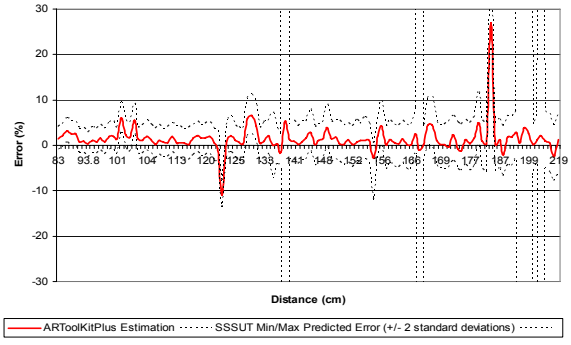

Figure 6. Errors in the ARToolKitPlus direct distance estimation from ground truth, calculated using the RPP optimizer. Sorted into order of distance to marker.

Due to sensitivities in the RPP algorithm the results of the SSSUT predictions showed points of inflection. At certain viewing angles and distances the perturbation values added to the corner detections were large enough to cause the computed solution to flip and distort leading to "spike-like" errors in the graphs. It can also be noted from our results, displayed in figure 5 and 6, that this nonlinear 'flipping' occurs more often (for this particular sigma point radial value of 1.638 pixels) when the marker is at a distance of more than around 190*cm*. This is because the added perturbation values become large enough, in

comparison the visible marker size, at this distance to make the pose estimation potentially ambiguous for some of the sigma point samples.

5.1 Predicted Performance Characteristics

By increasing the sigma point sample radius further we can use our algorithm to also examine where other potential points of instability in the ARToolKitPlus's pose estimation algorithm might be. In figure 7, which is sorted by viewing angle, we can clearly see strong periodic instabilities in the translational estimation occurring around +/- 45˚.

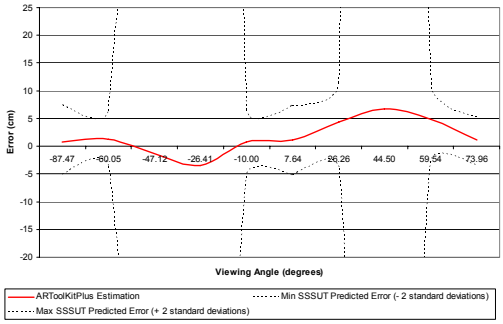

Figure 7. Using a much larger sigma point radius of 4.095 pixels, the predicted error variances can be seen to inflect periodically with the viewing angle. Images ordered by viewing angle +/- 90˚.

This predicted result is confirmed in the same figure by the errors in the ARToolKitPlus's own estimates at those viewing locations, which are also greater than at other angles. This regular translational error effect was described in [1] for the older ARToolKit marker tracker, where the errors in the translation estimations were directly related to the viewing angles. It is around these orientation angles that the SSSUT predictions also show signs of instability in the ARToolKitPlus's RPP optimization algorithm.

As might also be expected, and again observed by [1], our SSSUT method predicts a greater translational error as the distance to the marker increases. Figures 5 and 6 clearly show this linear correlation of the variance compared to the distance.

5.2 Computational Workload

An important aspect of our method is that it operates in real-time. With an image size set at 640x480 pixels we were able to achieve an average of around 300 frames per second without our SSSUT error prediction method, and around 100 frames a second with it. This implies a 30% reduction in speed for our method when compared to normal operation, which is not unexpected since the method is solving for pose multiple times.

6 Conclusions

In this paper we have highlighted the continuing need for real-time marker tracking error quantification, and the shortcomings of the current off-line LUT approaches. Furthermore, we have outlined our new proposed method, utilizing SSSUT to propagate errors through estimation algorithms, to both quantify the errors and improve the estimation results. Using the ARToolKitPlus we have illustrated the potential of our proposed solution to perform in real-time with marginal performance costs.

Acknowledgements

Many thanks to Dr Stuart Robson and the UCL Department of Geomatic Engineering for their help and use of the VMS photogrammetry software and calibrated environment.

This work was funded by the UK EPRSC Interdisciplinary Research Collaboration Equator (Grant GR/N15986/01).

References

[1] Pierre Malbezin, Wayne Piekarski and Bruce H. Thomas. Measuring ARToolKit Accuracy in Long Distance Tracking Experiments. In the First IEEE International Augmented Reality Toolkit Workshop, 2002.

[2] Daniel F. Abawi, Joachim Bienwald, Ralf Dörner. Accuracy in Optical Tracking with Fiducial Markers: An Accuracy Function for ARToolKit. In Proceedings of the Third IEEE and ACM ISMAR, 2004.

[3] Martin Bauer, Michael Schlegel, Daniel Pustka, Nassir Navab and Gudrun Klinker. Predicting and Estimating the Accuracy of Optical Tracking Systems. In Proceedings of the IEEE International Symposium on Mixed and Augmented Reality, 2006.

[4] L. Davis, E. Clarkson, and J. P. Rolland. Predicting accuracy in pose estimation for marker-based tracking. In Proceedings of the IEEE International Symposium on Mixed and Augmented Reality, 2003.

[5] J. Newman, M. Wagner, M. Bauer, A. MacWilliams, T. Pintaric, D. Beyer, D. Pustka, F. Strasser, D. Schmalstieg, and G. Klinker. Ubiquitous Tracking for Augmented Reality. In Proceedings of the Third IEEE and ACM International Symposium on Mixed and Augmented Reality, 2004. (ISMAR 2004), pages 192-201, Arlington, VA, USA, 2-5 November, 2004.

[6] B. MacIntyre, E. M. Coelho and S. J. Julier. OSGAR: A Scenegraph with Uncertain Transformations. In Proceedings of the IEEE International Symposium on Mixed and Augmented Reality, 2004.

[7] ARToolKit: http://www.hitl.washington.edu/artoolkit, [cited 22nd May, 2007].

[8] ARToolkitPlus: http://studierstube.icg.tu-graz.ac.at/handheld_ar/artoolkitplus.php, [cited 22nd May, 2007].

[9] M. Fiala. ARTag, a fiducial marker system using digital techniques. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005.

[10] S. J. Julier and J. K. Uhlmann. Unscented Filtering and Nonlinear Estimation. IEEE Review, 92(3), 2004.

[11] Mark Pupilli and Andrew Calway. Real-time Camera Tracking Using Known 3D Models and a Particle Filter. International Conference on Pattern Recognition, 2006.

[12] D. Pustka. Handling Error in Ubiquitous Tracking Setups. Master thesis, Technische Universitat Munchen, 15 August 2004. Available from: http://campar.in.tum.de/Students/DaPustka [cited 5th June, 2007].

[13] W. Hoff and T. Vincent. Analysis of head pose accuracy in augmented reality. In Proceedings of the IEEE Transactions on Visualization and Computer Graphics, 2000

[14] G. Schweighofer and A. Pinz. Robust pose estimation from a planar target. Technical Report TR-EMT-2005-01, Graz University of Technology, May 2005. Available from: http://www.emt.tu-graz.ac.at/publications/EMT_TR/TR-EMT-2005-01.pdf [cited 1st June, 2007].

[15] E. Kraft. A quaternion-based unscented Kalman filter for orientation tracking. In Proceedings of the Sixth International Conference of Information Fusion, 2003, pages 47-54, Cairns, Queensland, Australia, 8-11 July 2003.

[16] Vision Measurement System [online]. 2007. Available from: http://www.geomsoft.com/ [cited 1st June, 2007].

[17] James, M., R., Robson, S., Pinkerton, H., Ball, M.. Oblique photogrammetry with visible and thermal images of active lava flows. Bulletin of Volcanology 69, 105-108. (2006)

[18] Freeman, R. M., Steed, A. J. Interactive Modelling and Tracking for Mixed and Augmented Reality, In the Proceedings of ACM Virtual Reality Software and Technology, 2006.