Improving Augmented Reality using Image and Scene Constraints

R. A. Smith, A. W. Fitzgibbon and A. Zisserman
Department of Engineering Science, University of Oxford,
Parks Road, Oxford, UK
{richard,awf,az}@robots.ox.ac.uk
http://robots.ox.ac.uk/∼vgg

Abstract

The goal of augmented reality is to insert virtual objects into real video sequences. This paper shows that by incorporating image-based geometric constraints over multiple views, we improve on traditional techniques which use purely 3D information. The constraints imposed are chosen to directly target perceptual cues, important to the human visual system, by which errors in AR are most readily perceived. Imposition of the constraints is achieved by constrained maximum-likelihood estimation, and blends projective, affine and Euclidean geometry as appropriate in different cases. We introduce a number of examples of augmented reality tasks, show how image-based constraints can be incorporated into current 3D-based systems, and demonstrate the improvements conferred.

Keywords: Augmented reality, Structure from motion, Multiple-View Geometry.

1 Introduction

The objective of augmented reality (AR) is to add virtual objects to real video sequences, allowing computer-generated imagery to be overlaid on the video in such a manner as to appear part of the viewed 3D scene. Applications include computer-aided surgery [7], robot teleoperation [11], and special effects for the film and broadcast industries [6, 20, 12]. This paper concentrates on the particular application of video post-production.

Augmentation for special effects (or *compositing*) has traditionally been done by skilled animators, painting 2D images onto each plate of film. This technique ensures that the final composite is visually credible, but is enormously expensive, and is limited to relatively simple effects even for expert animators.

More recently, computer graphics has driven an approach in which the world is modelled in 3D, and then the camera is computed for each frame in the sequence [22]. Graphical models can then be rendered from the same sequence of viewpoints as the original footage, giving the impression that the virtual objects are rigidly attached to the structure in the scene. Furthermore, recent advances in structure and motion recovery have enabled the 3D structure and cameras for each frame to be recovered automatically directly from the image sequence [1, 2, 5, 14, 15, 21].

However, although the camera motion can be recovered to sub-pixel accuracy, the *registration* accuracy of the augmentation must equal or surpass this. The errors resulting from inaccurate registration fall into two main categories: a high-frequency (frame rate)

BMVC 1999 doi:10.5244/C.13.30

jitter, and low-frequency drift. Even if jitter is of the order of tenths of pixels, a task such as line extrapolation (§3) amplifies the error to levels which cause the augmentation to visibly wobble in the composited sequence. Drift is unacceptable because of the human visual system's high sensitivity to perceptual cues such as motion parallax and parallelism, exposing even sub-pixel discrepancies in the final output. At a basic level, drift is a problem of temporal extrapolation—geometry is predicted well in the frames from which it was estimated, but the prediction degrades for extrapolation into other frames. However, although the feature may not be detectable in the "far" images, its position may be inferred by human observers using other cues (for example, a nearby parallel line in §3.1), and errors in extrapolation will be observed.

In the post-production industry, correcting these deficiencies requires manual intervention afterwards to fine-tune the registration. The novelty in this paper is that we explicitly incorporate image and scene constraints into the geometry estimation and achieve perceptually perfect results automatically. The key to the approach is that the perceptual cues by which human observers discern errors in the augmentation are targetted and incorporated into the geometry estimation and rendering processes.

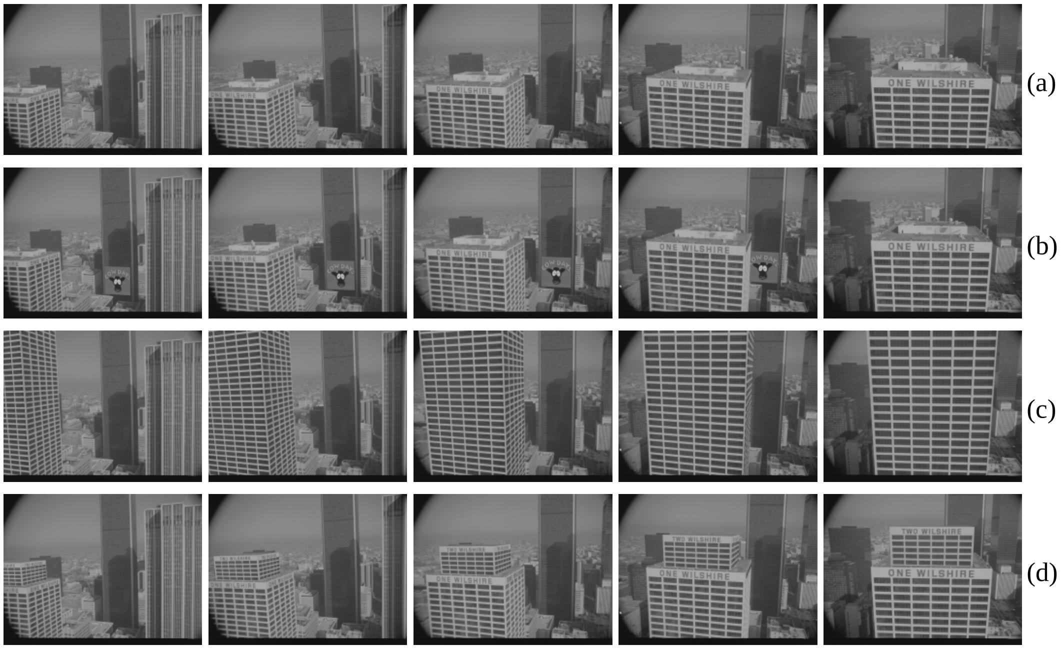


Figure 1: Typical AR tasks: (a) Five frames (between frames 70 and 220) from the Wilshire sequence. The images are 1024×768 pixels. (b) **Plane augmentation with occlusion**—the 'Cowdays' logo is added to the far building and occluded by the sides of the foreground Wilshire building. (c) **Plane extrapolation**—the sides of the foreground "One Wilshire" building are extended. (d) **3D augmentation**—an extra storey "Two Wilshire" is added to the top of the Wilshire building.

1.1 Overview

Three basic types of insertion AR tasks will be considered, imposing progressively more complex requirements on the tracking and estimation algorithms.

1. **Augmentation of a planar surface:** This is the simplest case—only a single plane need be considered, and the computational task is to determine a 2D homography. If the plane is not a foreground plane, occlusion boundaries must be identified and tracked.

2. **Augmentation of connected planes:** To augment a pair of connected planes, it is perceptually important that the line of intersection be accurately located. However, for image processing reasons, this line is often impossible to track through much of the sequence, so must be extrapolated from a small number of closely-spaced views.

3. **3D Augmentation**. In order to introduce a 3D object into the scene, the 3D locations of some reference points must be determined. Again this may only be possible over a short baseline, causing drift in the augmentation if not corrected.

Figure 1 shows typical examples of these techniques applied to a sequence of 340 images of skyscrapers in Los Angeles, viewed with a moving camera mounted on a helicopter. Camera positions for each frame are computed automatically, together with corresponding interest points using the method described in [5]. The sequence provides several thousand 3D points, with RMS image reprojection error of 0.2 pixels. In the following sections we go through each of these examples in detail.

2 Example I: Planar surface augmentation

The objective here is to superimpose an image onto a *target* plane in the scene.

The map between planes in perspective images is a planar homography $\mathtt{H}$ (a plane projective transformation). We will require the map between the plane of the augmenting pattern and the scene plane in each frame of the sequence. Equivalently this map may be computed as the map from the augmentation pattern to the scene plane in one frame, composed with the inter-image homography of the scene plane between this frame and all others in the sequence. We explore two ways of computing this inter-image homography induced by the target plane, one local and one global.

Method 1: A purely 2D approach is to track image features, such as lines or interest points and compute the homographies directly by least squares fitting. The homographies are then optimally estimated by bundle adjustment (see below).

Method 2: A more global approach is to impose the global scene constraint that all the tracked features are coplanar. The homographies are computed by identifying a 3D plane, and projecting it via the precomputed camera matrices. Suppose the 3×4 camera matrices for each view are $\mathtt{P}^j = [\mathtt{A}^j \mid \mathbf{a}^i]$, then if $\mathtt{P}^1 = [\mathtt{I} \mid \mathbf{0}]$ and the plane is defined by

$\boldsymbol{\pi}^\top \mathbf{X} = 0$ with $\boldsymbol{\pi}^\top = [\mathbf{v}^\top \ 1]$ then the homography induced by the plane between views j and 1 is [17]

$$\mathtt{H}^j = \mathtt{A}^j - \mathbf{a}^j \mathbf{v}^\top \tag{1}$$

The parameters of the plane are estimated by least-squares fitting to the same features used to track in method 1, and then by constrained bundle adjustment.

2.1 Implementation

The implementation of both of the above methods depends mainly on three technologies: bundle adjustment, geometrically constrained optimization and line tracking. These will also be central to the more complex estimation procedures later in the paper.

2.1.1 Bundle Adjustment

Bundle adjustment [9, 19] is used to optimally estimate geometric relations over multiple views, for example the homography used in method 1 above. It is a nonlinear least squares technique which gives the maximum likelihood estimate under the assumption that the errors in the measured 2D points are Gaussian.

We wish to estimate homography matrices $\hat{\mathtt{H}}^j$ for each view and ideal 2D points $\hat{\mathbf{x}}_i$ such that the reprojection error to the measured image points $\mathbf{x}_i^j$ is minimized. This corresponds to minimizing the cost function

$$\min_{\hat{\mathtt{H}}^j, \hat{\mathbf{x}}_i} \sum_{ij} d^2(\hat{\mathtt{H}}^j \hat{\mathbf{x}}_i, \mathbf{x}_i^j)$$

where $d(\mathbf{x}, \mathbf{y})$ is the Euclidean image distance between two points $\mathbf{x}$ and $\mathbf{y}$, and distances are included for every view in which there is a correspondence. In this case the number of parameters that must be estimated is $8m + 2n$ for m views (8 for each homography matrix) and n 2D points.

2.1.2 Error metric for line segments

If line segment (as opposed to point) features are tracked, then the error metric is modified. The ideal line is again parametrized by 2 degrees of freedom, and the error is the sum of squares of perpendicular distances between the detected line segment endpoints and the infinite 2D line [16]. This provides a reasonable approximation to the Mahalanobis distance at significantly reduced computational cost.

2.1.3 Geometrically constrained optimization

Because bundle adjustment explicitly parametrizes the geometric relations, it is just as readily applied to the geometrically constrained problem in method 2. In this case, the constraint is that the image features are coplanar in 3D. This is ensured simply by parametrizing the plane by the 3 d.o.f. of $\mathbf{v}$, so that each homography $\hat{\mathtt{H}}^j$ is a function of $\mathtt{P}^j$ and $\mathbf{v}$ as in (1). Again, each 2D point on the 3D plane has 2 parameters, so that the total number of unknowns is $3 + 2n$. This use of constrained optimization features throughout the rest of this paper, where image errors will be minimized over some parametrized geometric model.

2.1.4 Line tracking

Both the techniques discussed above for computing planar homographies require feature correspondences over many frames. A parametrized tracker [3, 8, 10] is used to automatically determine the positions of 2D line segments in the image sequence. An estimate of the line configuration in some image coordinate frame is used both to predict the location of the lines in the next frame, and in the rejection of outliers generated by the detection process. This estimate is initialised by edge detection and line segment fitting in the view closest to a fronto-parallel view of the plane.

In its simplest form, each line is parametrized by its 2 degrees of freedom. For constrained estimation, though, a more general form is used, where an arbitrary collection of 2D lines is parametrized by some user-specified model. For example, a single tracked line might be constrained to pass through a vanishing point, so that it has only one degree of freedom; or a collection of lines may be parametrized by a single homography.

2.2 Comparison

In comparing the two methods for plane augmentation, we can immediately make some qualitative observations.

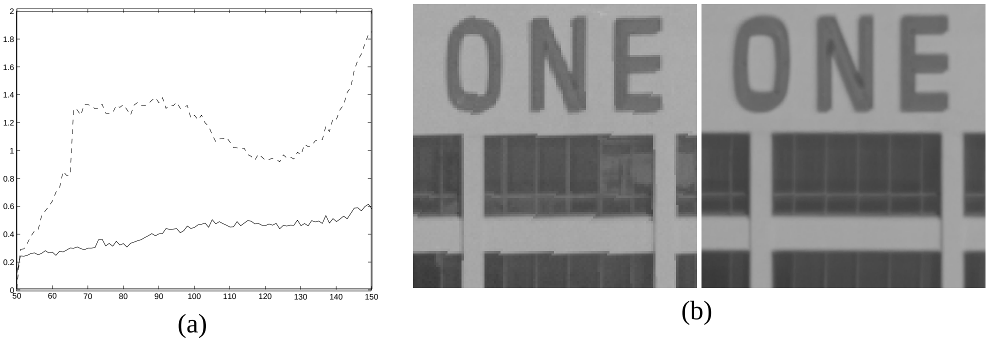
Method 2 *must* be used in images where no features can be detected on the target plane—for example, if the plane is temporarily occluded, or when it is foreshortened prior to going out of shot. Also, plane tracking requires fewer 2D features to define the plane. In figure 3, this proves very important, as the only reliable 2D features on the lower half of the building are a set of parallel lines, which are insufficient to determine a homography using method 1.

However, it is clear that homographies calculated by method 1 will more accurately fit the tracked features, since in method 2 we impose an additional constraint that the homographies must be consistent with the given camera projection matrices. Consequently, the results of method 2 are limited by the accuracy of the supplied camera projection matrices. Figure 2a shows the difference in accuracy of the two techniques.

On the other hand, method 1 may worsen the results if the tracked 2D features have correlated errors. For example, on the front wall of the Wilshire building, the vertical edges correspond in later frames to shadows cast by the wall on the windows. Tracking these features results in a low-frequency drift which produces a large movement of the augmenting surface. Manually deleting the offending features is simple, and results in a greatly improved track, but an automatic method would be preferable.

2.3 Occlusion

If the plane to be augmented is not a foreground plane, it is important that the augmenting pattern be correctly occluded by foreground objects. In the case where the foreground objects have straight edges, the 2D line tracker is used to locate the occluding edge in each image. Simply excluding pixels on the occluded side of the line results in the correct occlusion (Figure 3).



(a)

(b)

Figure 2: (a) **Reprojection error** (RMS in pixels) for planar homography estimation (near-side plane of “One Wilshire”). Method 1 (solid) estimates 8 d.o.f. of H for each frame. Method 2 (dashed) estimates the 3 d.o.f. of the plane $\mathbf{v}$ over the whole sequence and therefore results in a larger error. The method 1 error increases slowly through the 100 frames as the image noise increases, while the method 2 error varies from frame to frame as the relative quality of the $\mathtt{P}$ matrices changes. (b) **“Super-resolution” texture extraction**: The left image shows a single frame warped into canonical position; the right image shows the texture averaged from all images of the surface, after method 1 has registered them into the canonical frame.

3 Example II: Connected plane augmentation

In this example, the prototypical task is to extend the sides of one of the foreground buildings in the sequence vertically. This is achieved by extrapolating spatially the 3D lines which define the sides of the building. This also involves a temporal extrapolation, as the three visible lines are reliably detectable in only a subset of the complete set of frames, but must appear rigidly fixed in all frames of the sequence.

In this case, the visual cue which most tellingly reveals any inaccuracy is parallelism between the re-rendered walls of the building and other verticals in the scene. There-

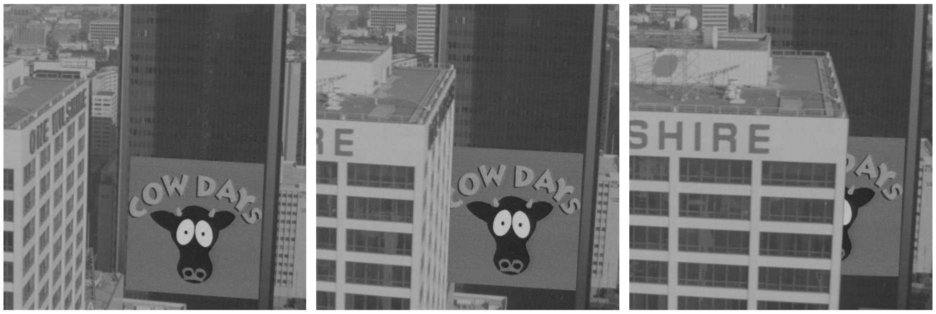


Figure 3: Insertion with occlusion: The “Cowdays” logo added to the far building is correctly occluded by the foreground “One Wilshire” building. The homography for augmentation is computed using method 2, as there are too few features on the building to use method 1.

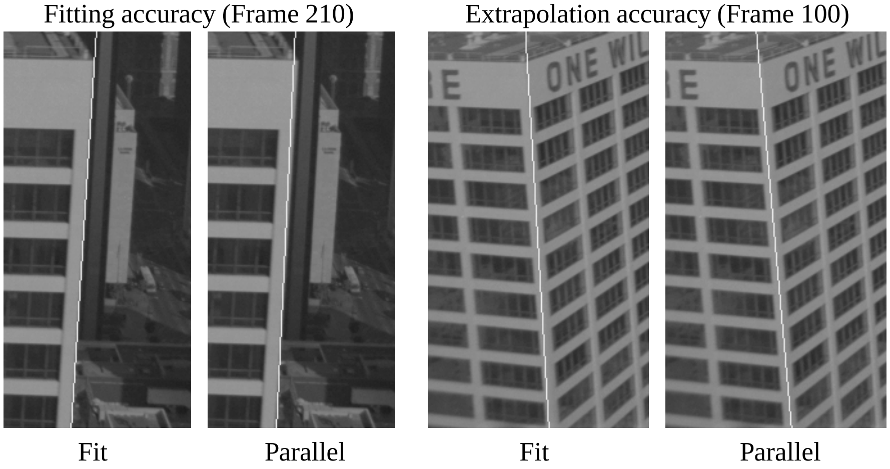

Fitting accuracy (Frame 210)
Extrapolation accuracy (Frame 100)
ONE WIL
Fit
Parallel
Fit
Parallel

Figure 4: Augmentation with parallelism constraints. “Fit”: 3D line fitted to image tracks from frames 190–230. “Parallel”: 3D line from the constraint that all verticals must share an intersection point. In frame 210, “Fit” is more accurate, because this is in the middle of the set of frames from which the line was estimated. Extrapolating back in time to frame 100, however, shows that the line is more accurately estimated over the sequence as a whole by using “Parallel”.

fore, we instantiate the 3D lines so that they are constrained to be parallel. In projective geometry, this is equivalent to saying that the lines are coincident at a point in 3-space. Furthermore, in this case, the lines are parallel to several other vertical lines in the scene, so all can contribute to computation of the vertical intersection point.

3.1 Comparison

Figure 4 shows the reprojected 3D lines with and without the parallelism constraint. In particular, the line on the near corner of the building, which is detected only in extreme frames of the sequence, is more reliably estimated over the whole sequence when constrained to be parallel to more reliably detected lines.

3.2 Implementation

The constraint that the three vertical edges of the building are parallel is implemented over multiple views as follows. We require that the lines in 3-space intersect in a single point. For n vertical lines in 3-space, this reduces the estimation from $4n$ degrees of freedom (four for each 3D line) to $3 + 2n$, three for the intersection point and two for each line. The specific parametrization used represents each 3D line by two points: the common intersection point, and a point on an arbitrary reference plane (2 d.o.f. per line). Then, the intersection point and 3D lines are estimated by minimizing reprojection error (§2.1.2) over all views.

In order to extend the building, a texture image is computed by averaging the images of the tracked plane over many views, resulting in a “super-resolution” texture map.

Figure 2b shows an example from the front wall of the building. Manually editing this texture map to add rows of windows gives the texture that is used for the final result in figure 1c. Again, because all the parallel lines in the scene are used to define the vertical, the difficult *spatial* extrapolation needed to ensure that the top of the building does not wobble is solved.

4 Example III: 3D Augmentation

As a final example, we consider the augmentation of a 3D scene with a 3D object. In this example, we wish to add a 3D box to the roof of the Wilshire building. In order to do so, we require a 3D coordinate system which has its origin in some known position relative to the roof, say a corner, and whose axes are parallel to the building sides and to the vertical.

The traditional way of doing this is to manually select some spatial features on the target object in two widely spaced views, and use these to solve for the 3D coordinate transformation which aligns a Euclidean coordinate system with the target. Given three perpendicular 3D lines which are identified over two widely separated views, and a reference point on the intersection of two of the lines, we solve for the Euclidean transformation that maps the reference point to the origin and aligns the lines with the coordinate axes.

A second method is to define an affine coordinate system [13, 14], and represent the virtual object in that frame. In [14] the affine coordinates are obtained by tracking markers placed in the scene, and a parallel projection approximation is used. In this work, there is no such approximation: we use a perspective camera and track the vanishing points of three sets of parallel lines. The 3D intersection points of the line bundles are estimated as in §3.2. Using the three affine directions* we can rectify the camera matrices to an affine system with respect to the chosen directions.

Figure 5a is a wireframe detail of 3D augmentation resulting from this technique. In this instance, vertical lines inserted into the scene appear to drift relative to image features, this drift being due to error in the computed position of the corresponding horizontal direction. This error arises because the direction is near to the baseline connecting the cameras, so the 2D lines which intersect its vanishing point in any particular image provide only a weak constraint on its 3D position. However, not all image constraints have been enforced—the 3D lines are also known to be coplanar with the vertical lines on the front of the building, so the 3D direction must also be so. Minimizing the position of the vanishing point subject to the constraint that it is coplanar with the vertical lines removes the drift, as shown in figure 5b.

An example of such an augmentation, where a second storey is added to the foreground building is shown in figure 1d. The two methods of estimation show the advantage of using all the available constraints. In the line bundle method the situation is near-degenerate and a poor estimate of the direction is obtained, however if planarity is also imposed the estimation is not degenerate. Of course, many sequences will not provide such rich geometry, but most allow the extraction of one or two directions, which can be incorporated similarly.

*A *direction* is a 3D point on the plane at infinity [18], its homogeneous coordinates are $(\mathrm{X}, \mathrm{Y}, \mathrm{Z}, 0)$. Its projection into an image is the *vanishing point* of that 3D direction.

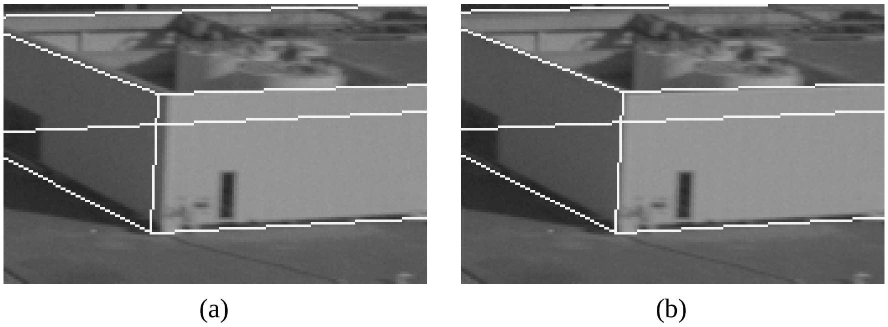


Figure 5: 3D augmentation: detail from Wilshire roof. (a) Unconstrained estimation of the horizontal direction. (b) Horizontal direction constrained to be coplanar with other lines on front face.

4.1 Implementation

Estimation of the scene directions involves essentially the same techniques as the estimation of parallel lines above. In this case, we use the estimated directions rather than the lines themselves. Given three directions and a reference point corresponding to a corner of the building, a projective transformation of 3-space is chosen which maps the directions to the X, Y and Z directions, and the reference point to the origin. Then the 3D model is constructed in this coordinate system and reprojected into the original images.

5 Discussion

This paper has presented a novel paradigm for the execution of augmented reality tasks. The key concept is that augmentation can be improved by incorporating image-based cues, much as current manual animators perform a 2D "touch-up" after 3D augmentation, but automatically.

The technique is not applicable in every situation, as it needs features which can be tracked to lock down the augmentation. However, it is precisely on scenes where many such features exist (such as the example sequence) that erroneous augmentation is most visible. Similarly, we have not made special reference to psychophysics in order to determine which features are most perceptually relevant.

The paper has also shown that it is relatively easy to identify candidate constraints that can be applied, but has made no attempt to taxonomize these constraints or to build a general constraint resolution system [4]. This means that the paradigm currently requires mathematical expertise on the part of the user. However the basic types of constraint are limited: coincidence of 2D points, parallelism, tangency etc; so that only a small number of estimators need to be implemented before the technique is more generally available.

References

[1] M. Bajura and U. Neumann. Dynamic registration correction in video-based augmented reality systems. *IEEE Computer Graphics and Applications*, 15(5), 1995.

[2] P. Beardsley, P. Torr, and A. Zisserman. 3D model acquisition from extended image sequences. In *Proc. ECCV*, LNCS 1064/1065, pages 683–695. Springer-Verlag, 1996.

[3] A. Blake and M. Isard. *Active Contours*. Springer, 1998.

[4] D. Bondyfalat and S. Bougnoux. Imposing euclidean constraints during self-calibration processes. In R. Koch and L. Van Gool, editors, *3D Structure from Multiple Images of Large-Scale Environments, LNCS 1506*. Springer-Verlag, 1998.

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

[6] E. Foxlin, M. Harrington, and G. Pfeifer. Constellation$^{\text{TM}}$: A wide-range wireless motion-tracking system for augmented reality and virtual set applications. In *Proc. ACM SIGGRAPH*, pages 371–378, 1998.

[7] W. E. L. Grimson, T. Lozano-Pérez, W. Wells, G. Ettinger, S. White, and R.Kikinis. An automatic registration method for frameless stereotaxy, image-guided surgery, and enhanced reality visualization. In *Proc. CVPR*, pages 430–436, 1994.

[8] C. J. Harris. Tracking with rigid models. In A. Blake and A. Yuille, editors, *Active Vision*. MIT Press, Cambridge, MA, 1992.

[9] R. I. Hartley. Euclidean reconstruction from uncalibrated views. In J. Mundy, A. Zisserman, and D. Forsyth, editors, *Applications of Invariance in Computer Vision*, LNCS 825, pages 237–256. Springer-Verlag, 1994.

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

[11] W. S. Kim. Virtual reality calibration and preview: Predictive displays for telerobotics. *Presence: Teleoperators and Virtual Environments*, 5(2):173–190, 1996.

[12] G. Klinker, D. Stricker, and D. Reiners. The use of reality models in augmented reality applications. In R. Koch and L. Van Gool, editors, *3D Structure from Multiple Images of Large-Scale Environments, LNCS 1506*, pages 275–289. Springer-Verlag, Jun 1998.

[13] J. J. Koenderink and A. J. van Doorn. Affine structure from motion. *J. Opt. Soc. Am. A*, 8(2):377–385, 1991.

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

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

[16] D. Liebowitz and A. Zisserman. Metric rectification for perspective images of planes. In *Proc. CVPR*, pages 482–488, Jun 1998.

[17] Q. T. Luong and T. Vieville. Canonical representations for the geometries of multiple projective views. *CVIU*, 64(2):193–229, Sep 1996.

[18] J. Semple and G. Kneebone. *Algebraic Projective Geometry*. Oxford University Press, 1979.

[19] C. Slama. *Manual of Photogrammetry*. American Society of Photogrammetry, Falls Church, VA, USA, 4th edition, 1980.

[20] 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, Sep 1997.

[21] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: A factorization approach. *IJCV*, 9(2):137–154, Nov 1992.

[22] M. Tuceryan, D. S. Greer, R. T. Whitaker, D. E. Breen, C. Crampton, E. Rose, and K. H. Ahlers. Calibration requirements and procedures for a monitor-based augmented reality system. *IEEE Transactions on Visualization and Computer Graphics*, 1(3), 1995.