Real-time Panoramic Mosaics and Augmented Reality

Manish Jethwa, Andrew Zisserman and Andrew Fitzgibbon
Robotics Research Group, Department of Engineering Science,
University of Oxford, Oxford, OX1 3PJ, UK.
[u94mj|az|awf]@robots.ox.ac.uk

Abstract

This paper investigates estimating exact imaging transformations accurately, reliably and efficiently. It is shown that in certain common computer vision situations the transformation required can be defined by a small number of parameters. Search is only required over these parameters, and consequently the search algorithms to estimate the transformation can be run at frame rate, without sacrificing robustness or accuracy. Performance is superior to often used approximations to these transformations. Two examples are illustrated: planar panoramic mosaicing, and augmented reality. Both applications run at frame rate on standard desktop machines, such as an SGI Indy or a PC.

1 Introduction

The idea explored in this paper is that by judicious modelling, exact transformations arising in computer vision applications can be specified by a reduced number of parameters. The great advantage of this is the reduction in the search space that must be explored when estimating these transformations, which in turn increases reliability and reduces computational cost. Moreover, because the parameter reduction is not achieved by approximating the exact transformations, there is no concomitant reduction in accuracy.

We illustrate this idea with two example applications, panoramic mosaicing and augmented reality. It is shown that by exploiting the reduced parametrization the exact transformations can be estimated reliably and accurately in real time (frame rate), whereas this would not be possible if the computation involved a search over all the degrees of freedom of the transformation.

In panoramic mosaicing the camera is rotated about its centre and the aim is to seamlessly sew together the acquired images. The transformation in this case is a planar homography. Panoramic mosaics are at the heart of QuicktimeVR [13], where they are used to generate new views. The related problem of acquiring mosaics has had considerable development recently through the work of Irani and Ananadan [6], Szeliski [15, 16, 17] and others [9]. The most impressive real time implementation is currently the video brush algorithm of Peleg [12]. However, in this case the map between images is only an approximating planar Euclidean transformation (with three degrees of freedom), not an exact general homography (eight degrees of freedom). In this paper we show that the exact homography can be computed in real time involving a search on only one parameter for the common case of rotation about a single axis. This is described in section 2.

BMVC 1998 doi:10.5244/C.12.85

The second example is that of Augmented Reality (AR) [7, 8, 10, 14, 18] — where computer generated images are superimposed on "real-life" scenes or videos. The transformation required in this case is the perspective projection map between the 3D world space and the image. In the general uncalibrated case, this map has eleven degrees of freedom. Following the work of Harris and others [4] it is known that the map can be computed at frame-rate by tracking based on Kalman filters. However, in AR any residual localization error (noise tremor) between the superimposed graphical model and real scene is extremely visible. For this reason the registration between artificial and real objects must be extremely accurate at the point of contact. We show here that this registration can be computed using a reduced (eight degrees of freedom) mapping. This is described in section 3. Again all computations (registration and super-position) are at frame rate on standard desktop computers.

2 Mosaicing

Mosaicing involves registering a set of images to form a larger composition representing a portion of the 3D scene. Such mosaicing is possible for any images related to each other by a global mapping. In the case of a camera rotated about its centre, corresponding image points are related by a planar homography (a plane projective transformation).

If $\mathbf{x}$ denotes the homogeneous coordinates $\mathbf{x} = (x, y, 1)^\top$ in one image and $\mathbf{x}'$ denotes the homogeneous coordinates $\mathbf{x}' = (x', y', 1)^\top$ of a corresponding point in another image, then they are related by

$$\mathbf{x}' = \mathtt{H}\mathbf{x} \tag{1}$$

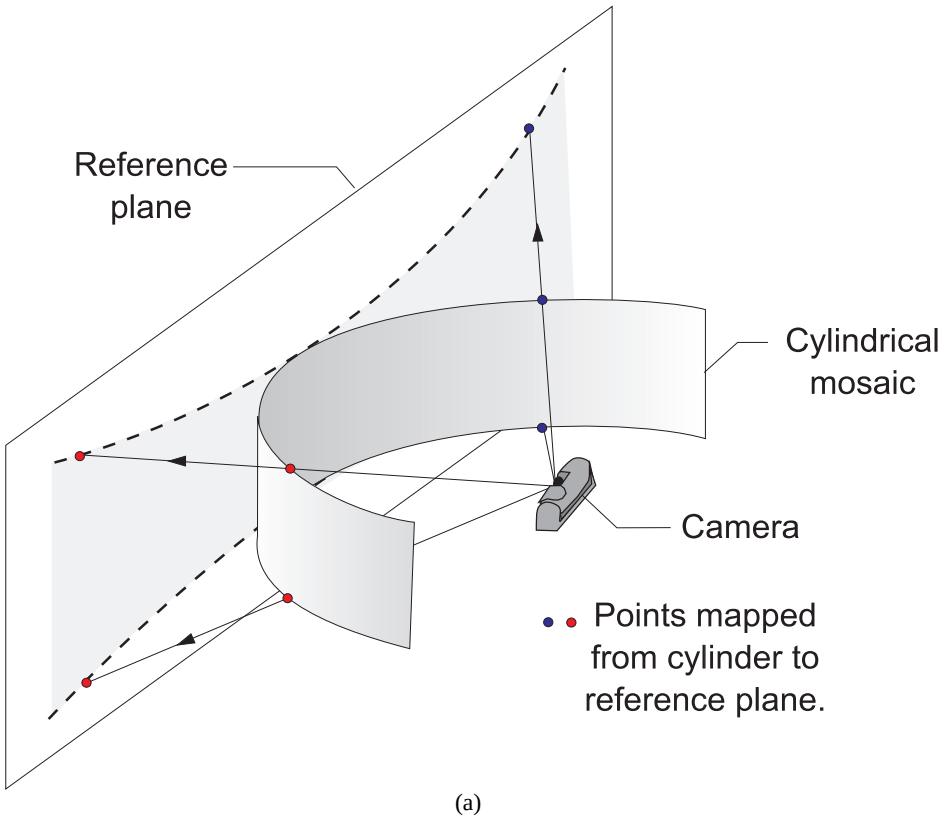
where $\mathtt{H}$ is the 3×3 matrix representing the homography. A homography has 8 degrees of freedom corresponding to the nine parameters of the matrix less one for the overall scaling. If we know the homography then all incoming images can be related to a single image (the image of the first frame for example) and therefore form a single composite image or mosaic.

Thus there are two stages involved in adding a new image to the mosaic:

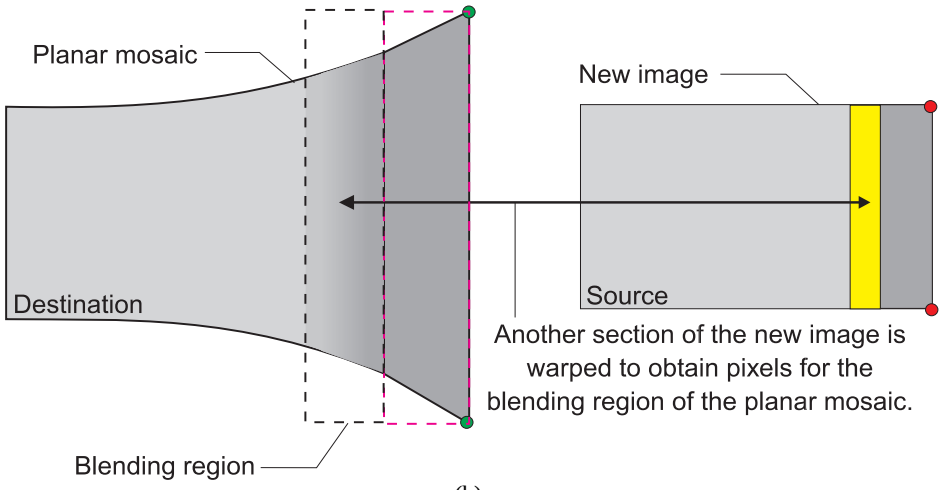
1. **Homography estimation**: Compute the homography between the new frame and a reference frame — in this case the developing mosaic.

2. **Image update and blending**: The mosaic is updated with the non-overlapping part (which must be determined). However, the Automatic Gain Control (AGC) active on most cameras introduces a dynamically changing camera gain and as a result can lead to intensity differences between images. These differences require blending in order to achieve a seamless mosaic.

These stages are described in more detail in the following sections. Both must be performed in real time. The final panoramic mosaic is rendered onto a plane. This has the advantage that straight lines in the scene are imaged as straight lines, but results in a non-rectangular boundary, as illustrated in figure 1a.



(a)



(b)

Figure 1: (a) The warping involved in mapping a panoramic mosaic onto a plane. (b) Blending region between new and old sections of the planar mosaic.

2.1 Parametrizing the homography

It can be shown [5] that under rotation about the camera centre the homography (1) has the form

$$\mathtt{H} = \mathtt{C}\mathtt{R}\mathtt{C}^{-1} \tag{2}$$

where $\mathtt{C}$ is the camera internal parameter matrix and $\mathtt{R}$ the rotation matrix. In general the camera matrix will have the form

$$\mathtt{C} = \begin{bmatrix} f & k & u_0 \\ 0 & \zeta f & v_0 \\ 0 & 0 & 1 \end{bmatrix},$$

where f is the focal length measured in horizontal pixel units, ζ is the dimension-less aspect ratio, k is the skew in the image plane, and (u_0, v_0) is the position of the principal point (the point a which the optical axis pierces the image plane).

Suppose now that the camera matrix $\mathtt{C}$ is known (to at least a good approximation), and that the camera rotates about the image y axis. We can choose the image coordinate system by this pre-calibration such that the matrix $\mathtt{C}$ is reduced to

$$\mathtt{C} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Then (2) reduces to

$$\mathtt{H} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\theta & 0 & -sin\theta \\ 0 & 1 & 0 \\ sin\theta & 0 & cos\theta \end{bmatrix} \begin{bmatrix} 1/f & 0 & 0 \\ 0 & 1/f & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} cos\theta & 0 & -fsin\theta \\ 0 & 1 & 0 \\ sin\theta/f & 0 & cos\theta \end{bmatrix} \tag{3}$$

where θ is the angle of rotation. Thus for a single axis rotation (and assuming f is fixed), estimating the exact (eight degrees of freedom) homography has been reduced to determining one parameter, namely θ.

2.2 Finding the Rotation

In order to determine the rotation angle θ we need a matching score between the original, I_1, and mapped image, I_2. The score used is the Normalised Sum of Squared Differences (NSSD)

$$S_{\text{nssd}} = \sum_i \left(\frac{I_1(x_i)}{< \overline{I_1} >} - \frac{I_2(x_i')}{< \overline{I_2} >} \right)^2. \tag{4}$$

This measure is invariant to the scaling in image intensities $I \rightarrow \alpha I$ which occurs with AGC.

The value of θ is estimated by evaluating the matching score as a function of θ, the lowest score providing the optimal value of θ. There is then the question of the size of the correlation window used. Three examples are shown in figure 2. Using just a single row from each image is sensitive to rotation perturbations of the camera and no obvious global minimum is found. If instead a band of pixels is averaged column wise then the matching process is robust enough to dismiss these perturbations in the camera motion, and a global minimum is obtained. Using a larger region also results in a global minimum but at untenable computational cost.

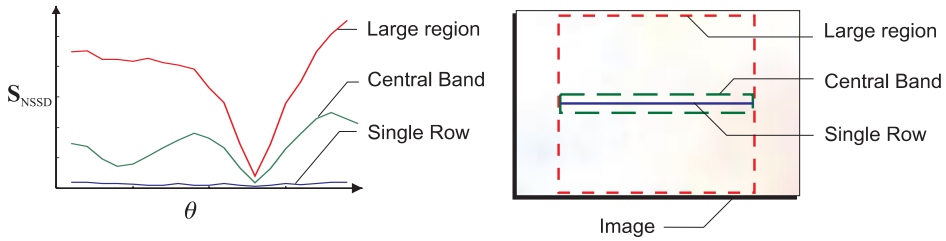


Figure 2: Normalised SSD score versus rotation angle θ for three sizes of correlation window.

2.3 Image update and blending

The image update involves texture mapping the source (newly acquired) image onto the destination mosaic. As usual in texture mapping [3] the inverse mapping is required so that the pixel in the destination mosaic is back-projected to the point in the source image, see figure 1b. The value is then obtained by bi-linear interpolation.

Using video streams to form mosaics inevitably leads to large areas of overlap between adjacent images in the sequence. Simply augmenting the mosaic with a previously unseen part of the image is not viable because in practice the AGC leads to intensity differences between images — placing these images side by side will mean that the join is clearly visible as a step change in intensity. This is of course undesirable when creating mosaics and so these changes need to be blended out to achieve a seamless mosaic.

Here a fixed size blending region of 20 pixel columns is used so that the process takes the same time during each cycle. We also need to define a blending function which will weight the intensity of pixels in each image during combination. A cosine blend is used to give a smooth transition (this produces slightly visibly superior results to a linear blend), as illustrated in figure 1b, a wider region is texture mapped in order to provide the values required for blending.

2.4 Performance and Results

The algorithm executes at a frame rate of 50Hz on an elderly Silicon Graphics Indy workstation. Rotating the camera at $24^{\circ}s^{-1}$ results in approximately 5-12 columns of pixels per frame being added to the mosaic.

The algorithm also contains additional facilities including: (1) the ability to dynamically add new sections to the mosaic at either end; (2) the ability to "go back" over the mosaic and update portions; (3) continuous refresh of a section of the mosaic that corresponds to the current view of the camera (this section is determined automatically). The latter facility is useful when creating mosaics in which people are moving in and out of the field of view.

Figure 3 is an example of a planar mosaic created in real-time. There are no noticeable effects due to inexact calibration.

Figure 3: A planar mosaic made up of over 250 individual frames. The mosaic was created starting from the centre out to the right and then back upon itself to the left.

3 Augmented Reality

In order to augment video streams with virtual objects, the primary requirement is the mapping from world to image for each frame. This section describes how this mapping may be computed. First the case of mapping a 2D image onto a moving plane in the scene is considered, introducing the tracking strategy. Then this case is extended to 3D, showing again that as with planar mosaicing it is possible to compute an exact general map from a simpler one that requires fewer parameters. This transformation is estimated in a real-time AR algorithm.

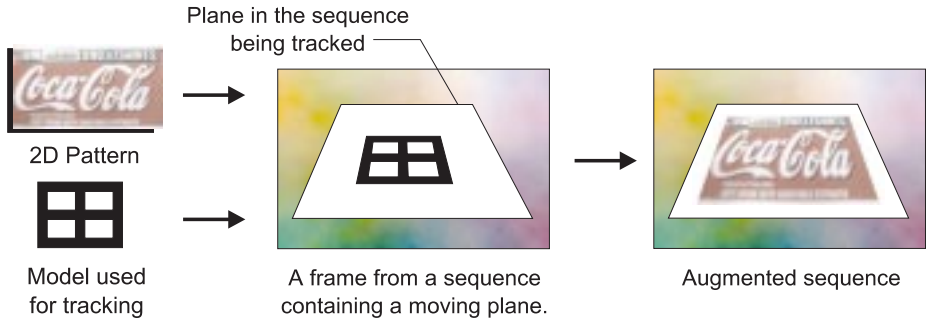


Figure 4: **2D Augmented Reality**: A plane in the sequence is augmented with a planar pattern to give the effect that the pattern is actually *printed* on the plane.

Figure 5: Stills from a sequence showing a plane augmented with a mapped pattern.

3.1 Planar tracking and mapping

In order to map a 2D pattern onto a moving plane in a video sequence, we require a world to image homography $\mathtt{H}$ for each frame. This homography varies as the relative pose of the camera and plane change. If the plane contains a known configuration of features, then the homography may be computed from this known configuration and its image. At least four corresponding features are necessary. Given this computed homography, the 2D pattern is mapped onto the plane and augments the video sequence.

The 2D AR process is illustrated in figure 5. A planar wireframe "target" is tracked in real time using a model based tracker. This target provides the known configuration, and the tracker ensures that the image correspondence is maintained throughout the live video sequence. Tracking is via a general robust tracker RoRAPiD [1], which is a robust version of the Harris tracker RAPiD [4]. The tracker is based on a Kalman filter, with a standard prediction, measurement and update cycle. The tracked object is described as a wireframe model, and the projected wireframe[1] is associated with image features such as significant straight edges. The robustness in RoRAPID is achieved by describing the object as a set of related geometric primitives, and using redundant measurements to facilitate the detection of outliers.

Filter Bandwidth The Kalman filter reliably identifies and tracks the target through long video sequences, resulting in a smooth mapping of the source texture onto the world plane. However, because the homography produced by the tracker is a weighted average of the predicted pose and the new measurement, the mapped texture image shows significant motion relative to the plane onto which it is mapped, which in AR applications leads to unacceptable jitter in the final output. As usual with such filters, increasing the relative weight of measurement over prediction does not resolve the problem because the reliability of the tracker is then reduced.

The solution adopted here is to combine the "low pass" Kalman filter tracker, with a "high pass" estimation of the homography. In the high pass estimate the homography is computed as the least squares solution to $\mathbf{l}'_i = \mathtt{H}^{-\top}\mathbf{l}_i$, where $\mathbf{l}_i$ are the 3-vectors representing the target configuration lines, and $\mathbf{l}'_i$ are the image lines. Only the image lines $\mathbf{l}'_i$ marked as inlying by the robust tracker are used in the estimate.

This combined process, using the Kalman filter for tracking and outlier rejection, and least-squares fitting for $\mathtt{H}$ estimation yields near-perfect registration of the augmented video and the model plane, in sharp contrast to the single filter scheme. Figure 5 shows stills from an example sequence.

3.2 3D Augmented Reality

Considering now the more general problem of augmenting video with virtual 3D objects. We require the transformation from Euclidean (world) 3-space to each image. This transformation is by perspective projection and is represented by a 3×4 projection matrix $\mathtt{P} = \mathtt{C}[\mathtt{R}|\mathbf{t}]$ where $\mathtt{C}$ is the 3×3 camera matrix, $\mathtt{R}$ is the 3×3 rotation matrix and $\mathbf{t}$ the 3D translation vector between the world and camera coordinate frames [2, 11]. Both 3-space and image 2-space are represented by homogeneous coordinates so that the projection is described by $\mathbf{x} = \mathtt{P}\mathbf{X}$, where $\mathbf{X} = (\mathrm{X}, \mathrm{Y}, \mathrm{Z}, 1)^\top$.

[1] Note that as the target is planar, no hidden line removal is necessary.

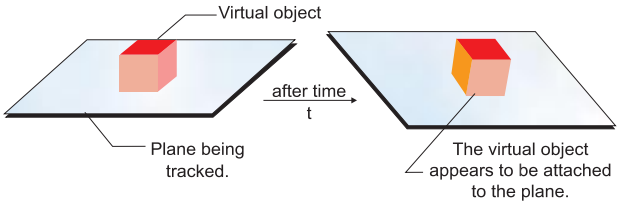


Figure 6: **3D Augmented Reality**: The virtual object must appear fixed to the plane we are tracking even if the plane's pose changes.

We wish to place the virtual object in the sequence with its base firmly on a plane so that the superimposed object does not appear to float in space. We show first that the projection matrix P can be obtained directly from the homography H of a plane. Armed with this result, we can then use the 2D tracker described above to track a planar target, compute H and thus P, and then map 3D objects onto the sequence. The process is illustrated in figure 6.

3.3 Obtaining P from a homography H

We first show the converse, namely that the homography for a plane can be obtained from P. If we choose the world coordinate system such that the plane has zero Z coordinate then the 3×4 matrix P reduces to

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \begin{pmatrix} \mathrm{X} \\ \mathrm{Y} \\ 0 \\ 1 \end{pmatrix} = \begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ p_{31} & p_{32} & p_{34} \end{bmatrix} \begin{pmatrix} \mathrm{X} \\ \mathrm{Y} \\ 1 \end{pmatrix} \quad (5)$$

The final 3×3 matrix represents a general plane to plane homography matrix H. Now, given C, it is straightforward to reverse this derivation to obtain P from H.

Writing the columns of P as

$$\mathtt{C}^{-1}\mathtt{P} = [\mathtt{R}|\mathbf{t}] = [\mathbf{h}_1 \ \mathbf{h}_2 \ \mathbf{p} \mid \mathbf{h}_3]$$

the vector $\mathbf{p}$ is the one we wish to estimate. Since the columns in the rotation matrix R are orthonormal we can write

$$\mathbf{p} = \mathbf{h}_1 \times \mathbf{h}_2 \quad (6)$$

We can now use (6) to compute the $\mathbf{p}$ vector and hence recover the projection matrix P for a known camera matrix C. We have therefore recovered the 3D perspective projection from the 2D homography and in doing so reduced the number of parameters from 11 to 8. If we ensure that the base of our virtual object has zero Z coordinate, then it will be correctly aligned with the plane in the scene and so the virtual object will appear as though it is sitting on the plane.

Effects of error In practice there are various sources of error present. These include line position measurement error, errors due to incorrect camera calibration C, and errors arising

from non-planarity of the tracked target. The result of these errors is that the H matrix, computed purely from line correspondences, will not have the correct structure, namely the columns $\mathbf{h}_1$ and $\mathbf{h}_2$ will not be exactly orthonormal. Ignoring this problem—simply using the $\mathbf{p}$ obtained from the cross product—leads to a secondary jitter phenomenon in which the base of the augmenting model is firmly attached to the tracked plane but the protruding components wobble unnervingly.

The solution adopted here is to use, within the Kalman filter tracker, a parametrization of the homography which explicitly models the rotation and translation of the 3D plane. The matrix P is then composed from the "low-pass" rotation from the filter, together with the "high-pass" translation from the homography computed from world (target) to line correspondences. This compromise reduces the jitter to a visually imperceptible level, since errors in the translational component are subjectively much more significant than those in the rotational component.

3.4 Performance and Results

The algorithm is implemented using OpenGL for the 3D rendering on a Silicon Graphics O_2 workstation, and runs in real-time, with sufficient computing power remaining to texture map live video and display models with an order of 1000 polygons. (The unencumbered O_2 can render about 6K polygons at frame-rate). The tracked target has 8 or more lines. Since only four are required to compute the homography, this allows scope for tracked lines to be lost in some frames and recovered in others, and also is sufficiently overdetermined for a least squares estimation to reduce errors.

Figure 7 shows a few frames from a video sequence before and after augmentation with a virtual object. Figure 8 to 10 show more examples including various virtual objects with and without texture mapping.

Figure 7: Frames in a video sequence are shown before and after augmentation with a virtual cube. The square pattern is the tracked target which is hidden by the virtual cube.

4 Conclusions and further work

We have shown that exact transformations can be estimated at frame rate if suitably parametrized. This work may be extended and improved in several ways. For example,

Figure 8: The virtual cube is texture mapped and appears to be held.

Figure 9: In the palm of your hand ...

1. To extend planar mosaics to unrestricted rotations about the camera centre only involves a three parameter search [16]. This could also be achieved in real time.

2. In the AR application, errors arising from inaccurate camera calibration can in fact provide an error measurement for the calibration matrix `C`. The calibration could then be updated in the Kalman filter along with the homography.

References

[1] M. Armstrong and A. Zisserman. Robust object tracking. In *Proc. Asian Conf. on Computer Vision*, volume I, pages 58–61, 1995.

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

[3] J. D. Foley, A. Van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice*. Addison-Wesley, 1990.

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

[5] R. I. Hartley. Self-calibration from multiple views with a rotating camera. In *Proc. ECCV*, LNCS 800/801, pages 471–478. Springer-Verlag, 1994.

[6] M. Irani, P. Anandan, and S. Hsu. Mosaic based representations of video sequences and their applications. In *Proc. ICCV*, pages 605–611, 1995.

[7] K.N. Kutulakos and J.R. Vallino. Non-euclidean object representation for calibration-free video overlay. In J. Ponce, A. Zisserman, and M. Hebert, editors, *Proc. Object Representation in Computer Vision II*, LNCS 1144, pages 381–401. Springer-Verlag, 1996.

[8] W. Lorensen, H. Cline, C. Nafis, R. Kikinis, D. Altobelli, and L. Gleason. Enhancing reality in the operating room. In *Visualization '93 Conference Proceedings, Los Alamitos, CA*, pages 410–415. IEEE Computer Society Press, 1993.

[9] S. Mann and R. W. Picard. Video orbits of the projective group: A new perspective on image mosaicing. Technical report, MIT, 1996.

[10] P. Milgram, S. Shumin, D. Drascic, and J. Grodski. Applications of augmented reality for human-robot communication. In *International Conference on Intelligent Robots and Systems Proceedings, Yokohama, Japan*, pages 1467–1472, 1993.

[11] J. Mundy and A. Zisserman. *Geometric Invariance in Computer Vision*. MIT Press, 1992.

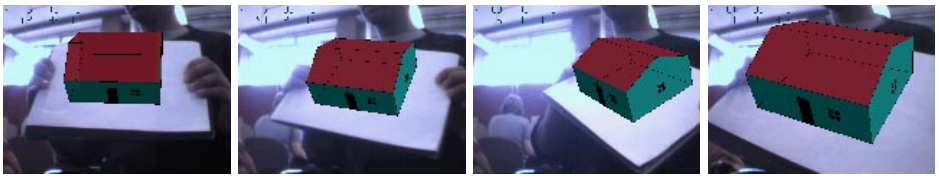


Figure 10: A plane in the scene defined by the paper is augmented with a 3D model of a simple house. As the pose of the plane changes the virtual house also changes its pose such that it always appears attached to the plane.

[12] S. Peleg. Panoramic mosaics by manifold projection. Technical report, Hebrew University of Jerusalem, 1997.

[13] QuicktimeVR. Web site http://www.apple.com/quicktime/qtvr/index.html.

[14] D. Sims. New realities in aircraft design and manufacture. *IEEE Computer Graphics and Applications*, 14(2):91, 1992.

[15] R. Szeliski. Image mosaicing for tele-reality applications. Technical report, Digital Equipment Corporation, Cambridge, USA, 1994.

[16] R. Szeliski and S. Heung-Yeung. Creating full view panoramic image mosaics and environment maps. In *SIGGRAPH*, 1997.

[17] R. Szeliski and S. B. Kang. Direct methods for visual scene reconstruction. In *ICCV Workshop on the Representation of Visual Scenes*, 1995.

[18] M. Tuceyran. Calibration requirements and proceedures for a monitor-based augmented reality system. *IEEE Trans. Visualisation and Computer Graphics*, vol. 1(no. 3):pages 255–273, 1995.