



PERGAMON

Computers & Graphics 27 (2003) 293–301

COMPUTERS
& GRAPHICS

www.elsevier.com/locate/cag

Technical section

Interactive reconstruction of virtual environments from video sequences

S. Gibson*, R.J. Hubbard, J. Cook, T.L.J. Howard

Advanced Interfaces Group, Department of Computer Science, University of Manchester, Oxford Road, Manchester, M13 9PL, UK

Accepted 27 November 2002

Abstract

There are many real-world applications of Virtual Reality requiring the construction of complex and accurate three-dimensional models that represent real environments. In this paper, we describe a rapid and robust semi-automatic system that allows such environments to be quickly and easily built from video sequences captured with standard consumer-level digital cameras. The system combines an automatic camera calibration algorithm with an interactive model-building phase, followed by automatic extraction and synthesis of surface textures from frames of the video sequence. The capabilities of the system are illustrated using a variety of example reconstructions.

© 2003 Elsevier Science Ltd. All rights reserved.

Keywords: Virtual reality; Computer vision; Model building; Camera calibration; Structure and motion; Texture

1. Introduction

Building three-dimensional models that resemble real environments has always been a difficult problem. Generally, this requires modelling accurate three-dimensional geometry, as well as surface materials or textures covering each surface. In addition to the appearance of the environment, modelling the behaviour of objects is also very important if a Virtual Environment (VE) is to allow any kind of user interaction. Generally, a scene hierarchy is constructed by specifying the relationships between objects in the scene. These relationships can then be used to assist the user when interacting with the environment.

Traditional methods of constructing models have involved a skilled user and a three-dimensional computer-aided design (CAD) program. Accurately modelling

a real environment in such a way can only be done if the user has obtained maps or blueprints of the scene, or has access to the scene in order to take precise physical measurements. Either way, the process is slow and laborious for anything but the simplest of scenes. Obtaining surface texture information has also traditionally been very difficult, requiring manual warping and mapping of photographic images onto each surface.

In the field of computer vision, automatic techniques have recently been developed that allow three-dimensional information to be constructed directly from photographs of the scene [1–4]. Typically, these algorithms analyse multiple images of an environment in order to infer the position and attributes of each camera, as well as the three-dimensional location of a dense set of points corresponding to important features in the images. In order to build more useful polygonal models, these points must be triangulated and subsequently segmented into separate objects. Similar triangulation and segmentation algorithms are required when expensive laser-range scanners are used to sample the scene geometry [5,6].

*Corresponding author. Tel.: +44-161-275-6141; fax: +44-161-275-6204.

E-mail addresses: sg@cs.man.ac.uk (S. Gibson), roger@cs.man.ac.uk (R.J. Hubbard), cookj@cs.man.ac.uk (J. Cook), toby@cs.man.ac.uk (T.L.J. Howard).

Automatic reconstruction techniques are, however, not yet robust enough to build useful VEs, which must have a simple enough form to allow them to be rendered in real-time and must contain enough structure so that a user can interact with important objects. Additionally, automatic algorithms are able only to reconstruct geometry that is seen explicitly in the images. This causes problems such as the appearance of holes and gaps in the model in regions that are not visible in any image, but which might well become visible when the model is being examined from different viewpoints.

In order to overcome problems such as these, semi-automatic approaches have been proposed that effectively combine user-guided segmentation of objects with automatic calculation of the position of those objects and the camera parameters [7–12]. The benefit of using semi-automatic, rather than fully automatic algorithms, is that we can employ user knowledge when modelling the environment: the walls of a room may be identified by the user and modelled as single large polygons, thereby overcoming problems caused by object occlusion. An object hierarchy may also be easily maintained during the construction of the scene, and environments may be constructed in an incremental fashion, with large

features specified at the start of the construction process and extra details added as necessary depending upon the envisaged use of the model.

1.1. Overview

The approach described in this paper falls between these automatic and semi-automatic categories. One significant disadvantage of current semi-automatic techniques is that they are limited to small numbers of input images due to the large amount of user interaction required to identify enough common features for camera calibration. In contrast, the main source of input data for our system is video sequences. Because of this, we can combine automated feature tracking [13], robust structure-from-motion [14] and self-calibration algorithms to automatically determine the camera parameters for each frame of the sequence. An overview of the calibration process is given in Fig. 1.

Once these calibration data have been obtained, we use semi-automatic techniques to build geometric descriptions of the objects in the scene. This is achieved by interactively manipulating the position, orientation and size of simple objects such as polygons, boxes and

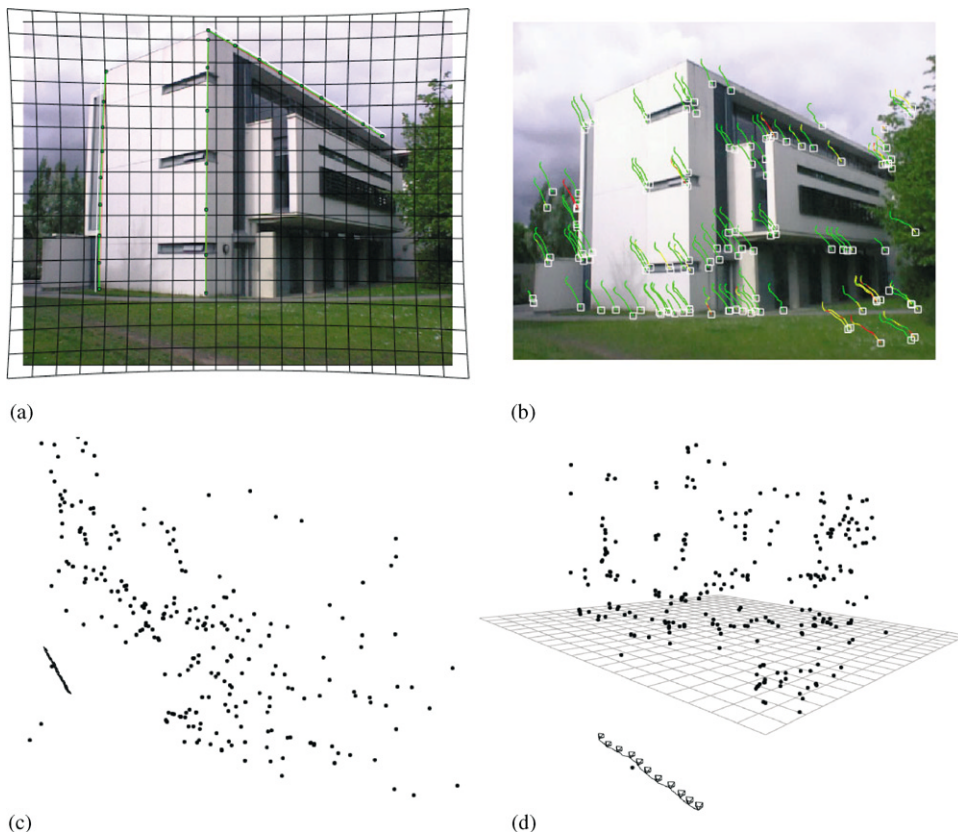


Fig. 1. An overview of the calibration process: (a) lens distortion removal, (b) feature tracking, (c) projective reconstruction in an unknown projective basis and (d) self-calibration and coordinate-frame orientation.

cylinders so that their projections into the frames of the video sequence match the projections of real objects. This manipulation is achieved using hierarchical parent–child constraints and image-based constraints that indicate the preferred projections of object vertices into frames of the sequence. We will describe a non-linear optimization algorithm that is capable of manipulating the position of these objects in real-time, so that the constraints specified by the user are best satisfied. Combining automatic camera calibration algorithms with interactive geometry reconstruction allows the user to spend more time modelling important features of the scene, rather than preparing the system for camera calibration.

We are interested in applying these techniques and algorithms to real-world problems, where the ability to quickly construct an accurate VE corresponding to a real scene can be an extremely powerful tool. In particular, we are studying the problem of constructing VEs of crime scenes, using forensic photographs [15] and video images and are cooperating with the Greater Manchester Police force (UK) and the UK's National Training Centre for Scientific Support to Crime Investigation (NTCSCI) [16].

An earlier pilot project [17,18] examined the feasibility of using VE reconstructions for police work. In the pilot project we undertook an entirely manual creation of a VE corresponding to a real crime scene, using architectural drawings and forensic photographs. The construction was extremely labour intensive, however, discussions with police officers, forensic scientists and trainers who had evaluated the pilot project demonstrated that such reconstructions can be of great benefit, offering new possibilities for analysis, training and briefing presentations. For such applications, the accuracy and fidelity of the VE are clearly very important, although different applications will be satisfied with various degrees of fidelity. The techniques described in this paper are currently being evaluated with reference to this area, although the algorithms described here are general enough to build reconstructions for a variety of different applications. Due to confidentiality requirements, the example reconstructions we show at the end of the paper are not from real crime scenes. Instead, we show examples of the system in use for typical indoor and outdoor reconstructions.

The remainder of this paper is structured as follows: in the next section we describe our approach to automatic calibration of video sequences. Following that, Section 3 describes how this calibration data may be used in an interactive setting to reconstruct geometric descriptions of objects in the scene. Once geometry has been reconstructed, we describe our automatic algorithm for extracing surface texture data in Section 4. Finally, different examples of the system in use are given in Section 5, and the paper closes with a discussion of current and future work in Section 6.

2. Video sequence calibration

Before we can start to reconstruct a three-dimensional model of the scene, we need to calibrate the camera used to record the video sequence. This involves estimating values for lens distortion, the camera's intrinsic parameters such as focal length and principal point, as well as its extrinsic parameters such as position and orientation, for each frame of the sequence. This section provides an overview of the techniques used. Further details can be found in [14].

Geometric lens distortion must be removed from the video sequence before calibration and reconstruction can begin. This is achieved in a semi-automatic manner by requesting that the user mark the projections of one or more straight lines in the image. These lines will be curved due to lens distortion, and we use a non-linear optimization algorithm [19] to estimate the distortion parameters so that these lines are straightened. Once this is done, the effects of distortion are automatically removed from the video sequence. Further details of the techniques used to calculate these distortion parameters can be found in [14].

Common approaches to estimating the relative motion between frames of a video sequence or a set of images require the identification of a number of features that are shared between each frame [20]. Typically, in interactive reconstruction systems these features (such as lines or points) are identified in each frame by the user [8–12]. When a large number of images are to be calibrated, or a large number of features are required for increased accuracy, this process can often dominate the time required to produce a reconstruction. By way of contrast, we take an automated approach to feature selection and tracking throughout video sequences. Video sequences have the benefit of a high frame rate, meaning frame-to-frame disparities are small, and features may be tracked easily from one frame into the next. When combined with robust structure-from-motion and auto-calibration algorithms, as described below, this results in a stable and easy-to-use calibration algorithm which forms the basis of our entire reconstruction system.

2.1. Feature tracking

Our automatic feature tracking algorithm is based upon the iterative approach described by Shi et al. [13], modified to account for colour images and changes in pixel brightness and contrast [21,22]. Features of interest are selected using the SUSAN corner detector [23]. We perform an additional check for each potential feature, immediately rejecting those that fail to track successfully into the next frame.

As features are tracked, they may be lost due to poor localization by the tracking algorithm, or because they move beyond the frame boundary. When this occurs, we

select new features to replace those lost and continue tracking, always trying to maintain a constant number of features in each frame. One novel aspect of our tracking algorithm is that after the first tracking phase has completed, a second pass moves from the end frame back towards the first, tracking those new features that were introduced as replacements. This second tracking pass increases the tracking duration for each replacement feature and improves the overall robustness of the calibration.

2.2. Structure from motion

Once a suitable set of features has been identified, we employ a merging-based projective reconstruction algorithm to estimate the feature locations and camera projection matrices for each frame of the sequence [14]. This is achieved in a novel fashion, by first identifying a set of key frames with which to start the reconstruction.

Key frames are selected so that the epipolar geometry¹ between each can be estimated reliably. This is achieved by examining various attributes of the features and camera motion, such as how well a planar homography² fits the feature motion, the overall epipolar error, and the number of features retained between adjacent key frames [14]. Once a set of key frames has been selected, pairwise projective reconstructions are built for each pair of key frames, using robust sampling algorithms [24] to estimate the epipolar geometry and projective structure. The camera projection matrices for non-key frames between each pair are estimated using a standard resectioning algorithm [20].

Pairwise sequences are then merged hierarchically until a single sequence is obtained. This merging is achieved by robustly estimating the collineation between two adjacent and partially overlapping reconstructions [14]. At the end of the merging process, bundle adjustment [3,25] is applied to distribute any residual error evenly across the sequence.

2.3. Self-calibration

The next stage of video sequence calibration involves upgrading the reconstruction from a projective to a

¹The epipolar geometry describes the relationship between two pinhole cameras. The projection of a single point into each image is related by the so-called *fundamental matrix*. The fundamental matrix transforms a point projection in one view to an *epipolar line* in the other, where the epipolar line is constrained to pass through the second point projection [20].

²In this case, a planar homography is a projective transformation that maps points in one frame to their corresponding positions in a second frame, where the camera motion between the two frames contains no translation component.

metric framework. This is necessary because the projective framework in which the sequence is reconstructed does not preserve important quantities such as distances and angles [20]. The projective transformation that maps back into metric space is also unknown and so this upgrade must be achieved using *self-calibration* algorithms. Typically, these involve estimating the position of the absolute conic [26,27]. For long sequences, an accumulation of error in the projective reconstruction means that the position of the conic often varies slightly along the sequence. Using previous methods for self-calibration, this variation often causes the conic estimation algorithm to fail, due to the non-positive semi-definite nature of its matrix representation.

We take a novel, robust sampling approach to self-calibration (see [14] for further details). Using this method, we select an appropriate set of frames with which to estimate the position of the absolute conic and are able to automatically reject frame sets that cause conic estimation to fail. In addition, the use of random sampling allows soft, non-linear constraints on camera parameters to be applied. Our algorithm is capable of enforcing known or constant focal length constraints, as well as selecting a conic that minimizes camera skew and deviations from a known aspect ratio or principal point. This significantly improves the robustness of the calibration procedure for longer video sequences. Estimates of focal lengths for frames of the sequence can be obtained, if necessary, using vanishing point information provided by the user [28].

Once a metric calibration has been obtained, a Levenberg–Marquardt metric bundle adjustment algorithm is applied. In addition to distributing error across the calibration, we have extended the metric bundle adjustment algorithm to minimize deviations from known camera parameters such as zero skew and known aspect ratio and also to enforce smoothness constraints by minimizing the second temporal derivatives of camera motion. This novel approach to bundle adjustment significantly reduces jittering effects seen in many previous automatic calibration algorithms, resulting in a visibly more stable calibration. Examples of the calibration algorithm in use are given in [14].

The final stage of the calibration process is the orientation of the world coordinate system. This is important for later reconstruction (see Section 3) and is achieved by asking the user to identify two or more orthogonal vanishing points in one frame of the video sequence. These vanishing points correspond to directions parallel to the X -, Y - or Z -axis and are used to determine the required orientation of the camera. The origin of the world coordinate system is also set by asking the user to identify a suitable point in the scene and to mark its projection into two or more frames of the video sequence. We then solve for the location of the origin point using multiple viewpoint triangulation [29].

3. Model reconstruction

Once the camera calibration data have been obtained for each frame of a video sequence using the algorithms described above, the process of interactive model reconstruction can begin. The user builds the model by interactively specifying the position, orientation and size of objects from a user-extensible library of shapes. As these primitives are created, a scene graph is maintained that describes the layout of the scene. The user manipulates these primitives in image space, attempting to match the features in the image. Manipulations of each object in image space are mapped into object space using a set of user-specified constraints and a non-linear optimization algorithm, described in Section 3.1.

Two types of constraints are used to assist the user in primitive manipulation: hierarchical constraints are strictly enforced and affect the position of one primitive with respect to its parent in the scene graph. Image-based constraints, on the other hand, are less strictly enforced and indicate image locations onto which primitive vertices should project. As the user manipulates these constraints, the non-linear optimization algorithm updates the parameters of the primitives so that all hierarchical constraints are satisfied exactly, and all image-based constraints are satisfied as accurately as possible.

An example of the manipulation procedure is shown in Fig. 2. In the first row, a single box primitive has been created by the user. This box will be manipulated to model the main part of the building seen in the image on the left. The right-hand column in Fig. 2 shows a slightly elevated view of the reconstructed model.

The user selects one vertex of the box primitive and drags it over the image plane into position at the base of the building (second row). The box primitive is constrained to sit on top of the ground plane by a hierarchical constraint. As this vertex is moved, the optimization algorithm changes the position of the box in object space, so that its corner vertex projects onto the image plane at the position indicated by the mouse pointer. After the user releases the vertex in its final position, an image constraint is created at that location. When the user selects a second vertex and changes its position (third row), the optimization algorithm attempts to satisfy both the projection to the mouse location and the previous image constraint. In this case, this involves altering the primitive's position and size. At any time, the user may move backwards and forwards through the frames of the video sequence in order to place additional image constraints and check the validity of the model that is being built (fourth row). Further details of how these constraints are satisfied are given in the next Section.

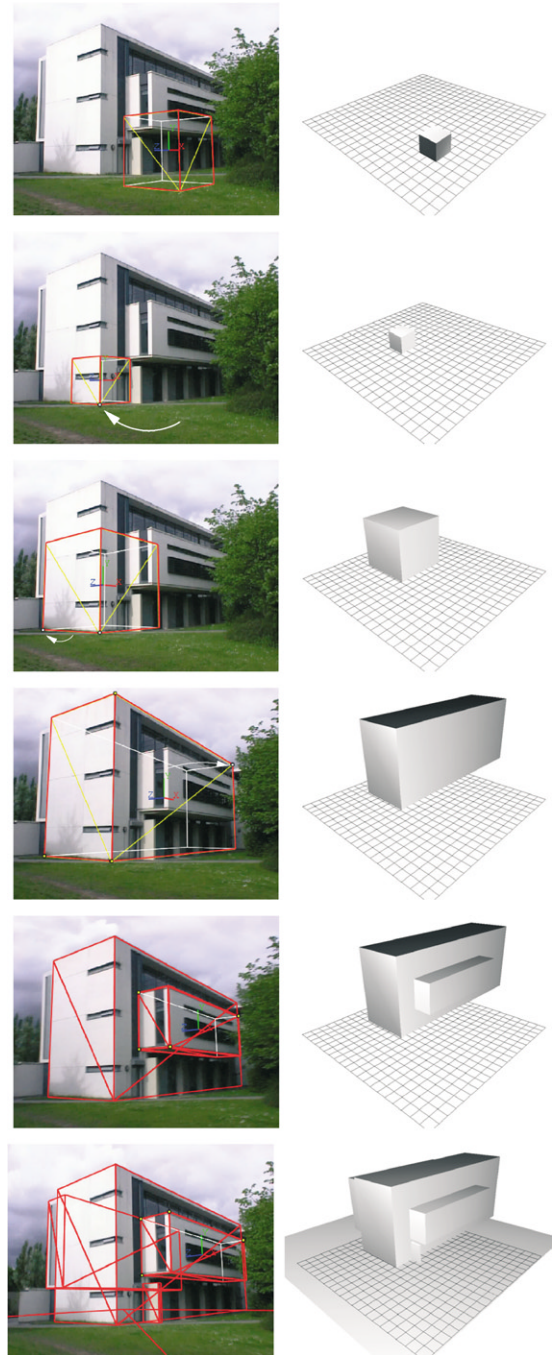


Fig. 2. Primitive manipulation (left) and corresponding model reconstruction (right). See text for details.

As further primitives are created, a scene graph is incrementally constructed that specifies the relationship between the objects in the scene. The position of one primitive with respect to its parent may be restricted using hierarchical constraints. By default, a

primitive is constrained so that the bottom face of its bounding box sits on top of the top face of its parent's bounding box. These constraints are easily changed so that, for instance, an object may be placed on the right-hand side of its parent (Fig. 2, fifth row). As primitives are manipulated, the non-linear optimization algorithm is applied recursively up and down the scene graph. Because the hierarchical and image constraints attempt to fix a primitive in space, recursion can almost always be limited to at most one level above or below a primitive.

A more complete example of the reconstruction, built from five primitives, is shown in the bottom row of Fig. 2. Reconstruction time for this simple model was less than 3 min.³ More complex examples will be given in Section 5.

3.1. Parameter estimation

The process of primitive manipulation involves estimating parameters such as position, size and orientation, such that all the hierarchical and image constraints are satisfied. This is achieved using a non-linear minimization algorithm, with an objective function that measures the sum of differences between each image constraint position and the projection of the corresponding object vertex into the frame of the video sequence. More formally, for N constraints on vertices \mathbf{X}_i we want to find

$$\min_S \sum_{i=1}^N (\mathbf{P}_i S(\mathbf{X}_i) - \mathbf{I}_i)^2, \quad (1)$$

where \mathbf{P}_i is the projection matrix associated with the frame in which image constraint i has been placed, \mathbf{I}_i is the location of the image constraint in that frame, and $S(\mathbf{X}_i)$ represents the location of vertex \mathbf{X}_i in terms of the parameter set, S , of the primitive.

The parameter set for one primitive consists of at most nine parameters: a translation vector $\mathbf{T} = (t_x, t_y, t_z, 1)$, a rotation matrix $\mathbf{R}(\theta, \phi, \psi)$ and a scale matrix $\mathbf{S} = \text{diag}(s_x, s_y, s_z, 1)$, where (s_x, s_y, s_z) are the scale parameters for each coordinate axis, and (θ, ϕ, ψ) are the corresponding rotations.

The vertices of each primitive are stored in the primitive's local coordinate system, and the transformation into world coordinates, $S(\mathbf{X}_i)$, is simply

$$S(\mathbf{X}_i) = \mathbf{R}\mathbf{S}\mathbf{X}_i + \mathbf{T}. \quad (2)$$

A subplex-based optimization algorithm [19] is used to minimize Eq. (1). At each stage of the optimization process, Eq. (2) is evaluated for each image constraint to determine the location of the constrained vertex.

Hierarchical scene-graph constraints are used to reduce the number of parameters that need to be calculated for each primitive. By default, a primitive is constrained to sit on top of its parent but is allowed to move freely in the X - Z plane. This constraint removes the need to calculate the Y coordinate of the primitive's position vector. Different hierarchical constraints may be associated with the X -, Y - or Z -axis of a primitive. The primitive's translation in each axis direction may be fixed relative to its parent primitive.

In addition to hierarchical constraints, the number of image constraints associated with a single primitive are also used to remove additional degrees of freedom, in order to ensure that the optimization process is well defined and will converge to a sensible solution. When the user first moves a primitive, a single image constraint is associated with the vertex that is being moved. Because only a single image constraint is available, only two parameters may be solved for in a sensible way.⁴ Because the default hierarchical constraint restricts movement to the X - Z plane, as described above, it is these two parameters that are solved for during the optimization process and the primitive scale and rotation are held fixed. As further image constraints are added, more parameters can come into play, provided their values are not restricted by the hierarchical constraints specified by the user.

Because of the small number of parameters that must be estimated, the minimization algorithm is typically able to run in real-time as the user manipulates a primitive. The parameter set for one frame is then used as the starting guess for the next optimization. The parameter changes required from frame-to-frame are typically small and this helps the optimization algorithm converge quickly to the desired result.

4. Texture extraction

Once a geometric description of the environment is available, texture maps may be extracted from the video footage and mapped onto surfaces. In order to achieve this, a buffer is constructed for each frame and the primitives are scan-converted into each buffer, colour-coded with a unique identification number for each polygon. This allows us to quickly determine the region in each frame that contributes to the texture map for each surface.

For every polygon in the scene, a texture fragment is extracted from each frame in which it is visible. These

³This is in addition to the time required for automatic video sequence calibration, which was less than 10 min for the 200 frame sequence on a 1.4 GHz Athlon CPU.

⁴Each image constraint in Eq. (1) effectively provides two constraints—one for the x pixel location and one for the y location.

fragments are then weighted according to their distance from the camera viewpoint and the angle between the surface normal and the direction towards the camera centre. Fragments are then blended together according to their weights. This means that texture fragments that are nearer a camera make a larger contribution towards the final texture map than those fragments that are more distant.

Finally, texture synthesis algorithms are used to try to fill in texture information for parts of surfaces that are not visible in any frame of the video sequence. Missing texels are calculated using a simple distance weighted sum of the nearest texels that contain valid texture information.

5. Results

Figs. 3–5 show some more complex examples of reconstructions performed using the techniques described in this paper. In each figure, two frames of the video sequence are shown on the left and in the middle, with a wireframe representation of the model overlaid. On the right-hand side is a texture-mapped rendering of the final reconstruction, shown from a novel viewpoint. Calibration and reconstruction times are given with each figure.

Fig. 5 was reconstructed from a video sequence captured using a camera mounted on a tripod. The camera was allowed to pan, tilt and zoom, and was



Fig. 3. An example reconstruction from a 200 frame video sequence. Two frames from the sequence are shown, with the reconstructed model overlaid in wireframe. The texture-mapped reconstruction is shown on the right from a different viewpoint. Total calibration time for this sequence was less than 10 min and model reconstruction took an additional 5 min.

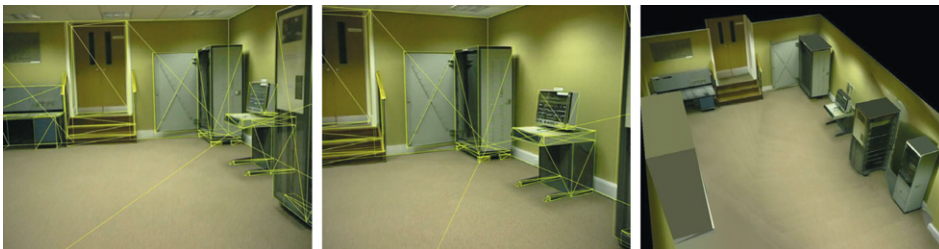


Fig. 4. A second reconstruction, taken from a 400 frame video sequence and built with 40 primitives. Calibration time for this sequence was around 20 min and the time required for model reconstruction was around 15 min.



Fig. 5. A reconstruction from a pan/tilt/zoom video sequence. Total calibration time for this sequence under 10 min and model reconstruction took an additional 10 min.

calibrated using algorithms similar to those described above. More complete descriptions of the calibration process can be found in [14].

6. Conclusion and future work

In this paper, we have presented methods for reconstructing Virtual Environments from video sequences. We have briefly discussed the methods we use to automatically calibrate the camera for each frame of a sequence, as well as our interactive method for model reconstruction. We have also shown examples of reconstructions that have been built using these methods.

There are still several weaknesses of the approach that will be addressed in the future. Most importantly, reconstructing complex, curved surfaces is currently very difficult. The library of primitives used for reconstruction may be easily extended by the user, simply by loading triangular-mesh objects into the system. Complex shapes, however, must still be approximated and the quality of parts of the reconstruction in Fig. 5 reflects this.

The benefit of using texture synthesis algorithms to fill in missing texture detail is clear. The algorithm that is currently used, however, will often introduce artifacts into the reconstruction, especially when important structures are partially occluded in the texture map (again, see the example on the right of Fig. 5). More involved techniques to synthesize these missing features would improve the overall visual quality of the reconstructions.

Future work also includes performing evaluations of the accuracy of the entire reconstruction process. As mentioned previously, one benefit of semi-automatic approaches to reconstruction is that a model may be reconstructed incrementally to an accuracy that depends on its intended use. It will be important to assess the amount of time required to build both a qualitatively and quantitatively accurate reconstruction for a given application.

We are also currently developing illumination reconstruction algorithms that allow us to reconstruct surface reflectance maps, rather than simple texture maps [30]. Application of a global-illumination algorithm will then allow us to render the Virtual Environment under novel lighting conditions. This is of particular importance for the application of crime-scene reconstruction, since forensic photographs and video footage are almost always captured under different lighting conditions than when the crime occurred. Combination of scene reconstruction with accurate simulation of lighting conditions will therefore provide a powerful investigative tool which normal photography and video recordings cannot offer.

The ICARUS system described in this paper is available for download from <http://iaig.cs.man.ac.uk/icarus>, and is free for non-commercial use.

Acknowledgements

The authors would like to thank their colleagues in the Advanced Interfaces Group for helpful discussions. This research has been funded by the UK's Engineering and Physical Sciences Research Council (EPSRC), under Grant Number GR/M14531, "REVEAL: Reconstruction of Virtual Environments with Accurate Lighting". We would also like to thank Detective Inspector David Heap and Pat Davis from Greater Manchester Police, and Nick Sawyer from the NTCSSCI for their continued support.

References

- [1] Beardsley P, Torr P, Zisserman A. 3D model acquisition from extended image sequences. In: Buxton B, Cipolla R (Eds.), Proceedings of the Fourth European Conference on Computer Vision, Lecture Notes in Computer Science, vol. 1065, Cambridge, Springer, Berlin, 1996. p. 683–95.
- [2] Fitzgibbon A, Zisserman A. Automatic camera recovery for closed or open image sequences. In: Burkhard H, Neumann B (Eds.), Proceedings of the European Conference on Computer Vision, Springer, Berlin, 1998. p. 311–26.
- [3] Hartley R. Euclidean reconstruction from uncalibrated views. Applications of Invariance in Computer Vision, Lecture Notes in Computer Science, Series 825, Springer, Berlin, 1994. p. 237–56.
- [4] Pollefeys M, Koch R, van Gool L. Structure and motion from image sequences. In: Kahmen G, editor. Proceedings of the Conference on Optical 3-D Measurement Techniques V, Vienna, Austria, 2001. p. 251–8.
- [5] Yu Y, Ferencz A, Malik J. Extracting objects from range and radiance images. IEEE Transactions on Visualization and Computer Graphics 2001;7(4):351–64.
- [6] 3rdTech, Deltasphere 3d scene digitizer, <http://www.3rdtech.com>.
- [7] Becker S, Michael Bove V Jr. Semi-automatic 3-d model extraction from uncalibrated 2-d camera views. SPIE Symposium on Electronic Imaging: Science & Technology, San Jose, 1995.
- [8] Debevec P, Taylor C, Malik J. Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. In: Proceedings of the SIGGRAPH 96, New Orleans, USA, Computer Graphics Proceedings, Annual Conference Series, 1996. p. 11–20.
- [9] Bougnoux S, Robert L. Totalcalib: a fast and reliable system for off-line calibration of image sequences. In: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico, 1997.

- [10] Cipolla R, Boyer E. 3d model acquisition from uncalibrated images. Proceedings of the IAPR Workshop on Machine Vision Applications, Chiba, Japan, 1998. p. 559–68.
- [11] Poulin P, Ouimet M, Frasson M. Interactive modeling with photogrammetry. In: Proceedings of the Eurographics Workshop on Rendering, Vienna, Austria, 1998.
- [12] Hakim SE. A practical approach to creating precise and detailed 3d models from single and multiple views. In: International Archives of Photogrammetry and Remote Sensing, B5A, Commission V, vol. 33, Amsterdam, 2000. p. 122–9.
- [13] Shi J, Tomasi C. Good features to track. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Seattle, USA, 1994. p. 593–600.
- [14] Gibson S, Cook J, Hubbard R, Howard T. Accurate camera calibration for off-line, video-based augmented reality. In: ACM and IEEE International Symposium on Mixed and Augmented Reality, Darmstadt, Germany, 2002.
- [15] Gibson S, Howard TLJ. Interactive reconstruction of virtual environments from photographs, with application to scene-of-crime analysis. In: Proceedings of the ACM Symposium on Virtual Reality and Software Technology (VRST), Seoul, South Korea, 2000.
- [16] National Training Centre for Scientific Support to Crime Investigation, <http://www.forensic-training.police.uk/>.
- [17] Murta A, Gibson S, Howard TLJ, Hubbard RJ, West AJ. Modelling and rendering for scene of crime reconstruction: a case study. In: Proceedings of the Eurographics Leeds, UK, 1998. p. 169–73.
- [18] Howard TLJ, Gibson S, Murta A. Virtual environments for scene of crime reconstruction and analysis. In: Proceedings of the SPIE/IS&T, vol. 3960, San Jose, CA, 2000.
- [19] Rowan T. Functional stability analysis of numerical algorithms. Ph.D. Thesis, University of Texas at Austin, 1990.
- [20] Hartley RI, Zisserman A. Multiple view geometry in computer vision, Cambridge, UK, 2000.
- [21] Heigl B, Paulus D, Niemann H. Tracking points in sequences of color images. In: Radia B, Niemann H, Zhuravlev Y, Gowrevitch I, Lipton I (Eds.), Proceedings of the Fifth Open German–Russian Workshop on Pattern Recognition and Image Understanding, Lecture Notes in Computer Science, Springer, Berlin, 1999.
- [22] Jin H, Favaro P, Soatto S. Real-time feature tracking and outlier rejection with changes in illumination. In: Proceedings of the International Conference on Computer Vision, Vancouver, Canada, 2001.
- [23] Smith SM. A new class of corner finder. In: Proceedings of the Third British Machine Vision Conference, University of Leeds, UK, 1992. p. 139–48.
- [24] Torr PH, Zisserman A. MLESAC: a new robust estimator with application to existing image geometry. Computer Vision and Image Understanding 2000;78(1):138–56.
- [25] Triggs W, McLauchlan P, Hartley RI, Fitzgibbon AW. Bundle adjustment: a modern synthesis. In: Triggs B, Zisserman A, Szeliski R (Eds.), Vision Algorithms: theory and practice, Lecture Notes in Computer Science, 1883, Springer, Berlin, 2000.
- [26] Triggs W. Auto-calibration and the absolute conic. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico, 1997. p. 609–14.
- [27] Pollefeys M, Koch R, van Gool L. Self-calibration and metric reconstruction in spite of varying and unknown internal camera parameters. International Journal of Computer Vision 1999;32(1):7–25.
- [28] Cipolla R, Drummond T, Robertson D. Camera calibration from vanishing points in images of architectural scenes. Proceedings of the British Machine Vision Conference, vol. 2, Nottingham, 1999. p. 382–91.
- [29] Hartley R, Sturm P. Triangulation. In: Proceedings of the Conference Computer Analysis of Images and Patterns, Prague, Czech Republic, 1995.
- [30] Gibson S, Howard TLJ, Hubbard RJ. Flexible image-based photometric reconstruction using virtual light sources. Computer graphics forum, Proceedings of the Eurographics 2001, vol. 19, No. 3, Manchester, UK, 2001.