

# Interactive Reconstruction of Virtual Environments from Photographs, with Application to Scene-of-Crime Analysis

Simon Gibson and Toby Howard  
Advanced Interfaces Group, University of Manchester  
Oxford Road, Manchester  
M13 9PL, UK.  
+44 (0) 161 275 6141  
sg,toby@cs.man.ac.uk

## ABSTRACT

There are many real-world applications of Virtual Reality that require the construction of complex and accurate three-dimensional models, suitably structured for interactive manipulation. In this paper, we present semi-automatic methods that allow such environments to be quickly and easily built from photographs taken with uncalibrated cameras, and illustrate the techniques by application to the real-world problem of scene-of-crime reconstruction.

## Categories and Subject Descriptors

I.2.10 [Artificial Intelligence]: Vision and Scene Understanding—*Modelling and Recovery of Physical Attributes*; I.3.5 [Computer Graphics]: Computational Geometry and Object Modelling—*Modelling packages*; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—*Color, Shading, Shadowing and Texture*; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—*Virtual Reality*; I.3.8 [Computer Graphics]: Applications; I.4.1 [Image Processing]: Scene Analysis—*Stereo*

## General Terms

Algorithms, Measurement.

## Keywords

Virtual Reality, Computer Vision, Photogrammetry, Model Building, Texture, Scene of Crime Reconstruction.

## 1. INTRODUCTION

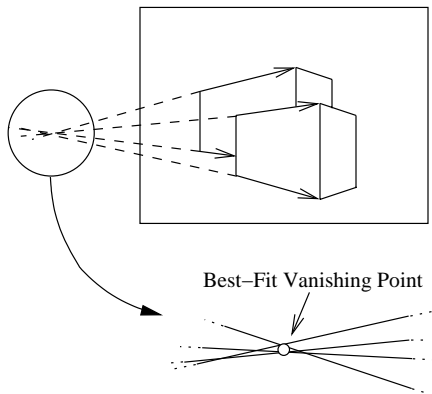
Building models suitable for use in interactive Virtual Environments (VEs) has always been a difficult problem. When the environment must resemble an existing scene, this requires obtaining accurate three-dimensional geometry, as well as surface materials or textures. In addition to the appearance of the environment, modelling the behaviour of objects is also very important if the environment is to allow any kind of non-passive user interaction. Generally, a scene hierarchy is constructed by specifying the relationships be-

tween objects in the scene. These relationships can then be used to assist the user in interacting with the environment.

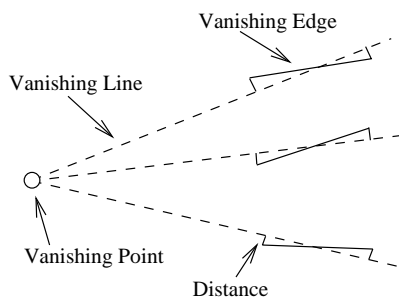
Traditional methods of constructing scene models involve a skilled user and a three-dimensional CAD (Computer Aided Design) program. Accurately modelling a real environment in such a way can only be done if the user has obtained blueprints or similar documents, or has access to the actual scene in order to take precise physical measurements. Either way, the process is slow and laborious for anything but the simplest of scenes. Obtaining surface materials and textures has also traditionally been very difficult. These attributes are essential to enhance the appearance of the environment and can greatly increase the feeling of immersion a user can experience in a VE.

In the fields of photogrammetry and computer vision, many techniques have been developed that allow three-dimensional structures to be constructed from photographs and video sequences [10, 13, 19, 2]. Typically, these algorithms involve taking multiple images of an environment using calibrated cameras or laser-range finders, and then analysing those images in order to infer the three-dimensional structure of the scene. Recently, researchers have developed algorithms that are capable of automatically generating three-dimensional models of simple environments [2, 3, 11]. The output of such algorithms are generally dense sets of points, corresponding to important features in the scene, and these must be converted into polygon meshes and subsequently segmented into objects in order to generate suitable representations for a VE. Similar segmentation algorithms are required when expensive laser-range scanners are used to sample the scene. Such automatic techniques are also 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 and object relationships so that a user can interact with important objects.

In this paper, we discuss how VEs may be quickly and efficiently built using semi-automatic computer vision reconstruction methods similar to those presented in [4, 9, 6]. The benefit of using semi-automatic, rather than fully automatic algorithms, is that we can accurately model a real environment to whatever level of detail we require, and an object hierarchy may be easily maintained during the construction of the scene. Indeed, we will show how specifying the relationships between objects that are useful for user interaction can also simplify scene reconstruction, and increase the robustness of the reconstruction process.



**Figure 1: Camera calibration using vanishing points.** The direction of three sets of orthogonal edges are marked on the image by the user. The infinite vanishing-point lines are intersected to find the vanishing point. Due to image inaccuracies, a “best-fit” intersection is performed.



**Figure 2: The vanishing point is selected so as to minimise the distance between the vanishing lines and their associated image edges.**

Environments can also be built 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 environment. Another benefit of our approach to building VEs is that we do not require that the photographs are taken with calibrated cameras, and we can even produce useful results from a single image. This allows environments to be built from archived photographs, where access to the original scene is no longer possible.

The remainder of this paper is structured as follows: in the next two sections, we describe our approach to building VEs from image sequences, presenting our approach to camera calibration in Section 2, and model construction in Section 3. Following that, in Section 4 we describe how surface materials and textures may be extracted from the images and used to increase the realism of the environment. Section 5 presents results from our application case-study: reconstruction of virtual crime-scenes. Finally, we draw conclusions in Section 6.

## 2. CAMERA CALIBRATION

Before we can start to reconstruct a three-dimensional model of the scene from a set of photographs, the cameras with which those photographs were taken must be calibrated. This involves estimating values for the camera’s intrinsic parameters such as focal length and principal point (or centre of projection), as well as its extrinsic parameters such as position and orientation in the world coordinate frame.

We perform this calibration using the technique of *vanishing-point estimation* [5, 7]. Using this simple approach, the user marks three sets of edges in an image, with each set corresponding one of the orthogonal axes of the world coordinate frame (Figure 1). The common intersection points of edges for each axis are then found.

Because of inaccuracies in the positioning of the edges by the user, they will not generally intersect at a single point. We therefore perform a non-linear optimisation that attempts to find the “best-fit” for the intersection point [16]. We constrain the vanishing lines to pass through a single vanishing point, and select the position of this point whilst attempting to minimise the orthogonal distances between each vanishing line and its associated vanishing edge (Figure 2)<sup>1</sup>. Once the vanishing point for each coordinate axis has been found, we can use it to determine the intrinsic camera parameters using the simple geometric relationship presented in [5]. This allows the construction of the *camera calibration matrix*,  $\mathbf{K}$ :

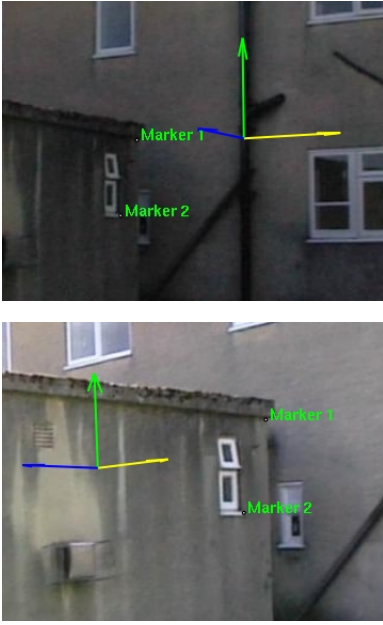
$$\mathbf{K} = \begin{bmatrix} \alpha & 0 & u_0 \\ 0 & \alpha & v_0 \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

where  $\alpha$  is the equivalent focal length, and  $u_0, v_0$  are the image-space coordinates of the principal point. Note that we are assuming that the image has zero skew, and pixels have unit aspect ratio.

### 2.1 Extrinsic Calibration

The position of the vanishing points can also be used to determine the orientation of each camera with respect to the world coordinate frame [8]. This is achieved by expressing the product of the camera calibration matrix  $\mathbf{K}$  and its rotation matrix  $\mathbf{R}$  in terms of the

<sup>1</sup>We use the Levenberg-Marquardt minimisation algorithm found in MINPACK, available from <http://www.netlib.org>



**Figure 3: After the orientations of each camera has been found, markers corresponding to the same point in the world coordinate frame are positioned in each image by the user.**

coordinates of the vanishing points:

$$\mathbf{KR} = \begin{bmatrix} u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 & 0 \\ 0 & \lambda_2 & 0 \\ 0 & 0 & \lambda_3 \end{bmatrix} \quad (2)$$

where  $u_i, v_i$  are the image-space coordinates of the  $i$ th vanishing point, and  $\lambda_i$  are their scaling factors [7]. Because of the simple form of the calibration matrix, the matrix product  $\mathbf{KR}$  can be easily decomposed to find all nine elements of the rotation matrix.

Once the intrinsic camera parameters and rotation matrices have been determined, the absolute positions of the cameras must be found. This is achieved by placing a number of *markers* in the images, where a marker corresponding to one (unknown) world coordinate point is placed in all images in which that point is visible (Figure 3). In order to fix the position of the world coordinate frame, we then choose one of these markers (typically, the one shared by the largest number of images) to be the origin point.

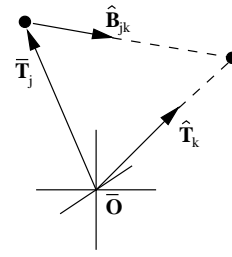
For each camera in which the origin marker is visible, the direction of translation from the origin can be found. Using the camera translation matrix,  $\mathbf{T}$ :

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & \bar{T}_x \\ 0 & 1 & 0 & \bar{T}_y \\ 0 & 0 & 1 & \bar{T}_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

the transformation of a known world coordinate,  $\mathbf{W}$ , into image-space can be expressed as follows:

$$\bar{\mathbf{W}}' = \mathbf{KRT} \cdot \bar{\mathbf{W}} \quad (4)$$

For the origin marker,  $\bar{\mathbf{W}} = [0, 0, 0]^T$ , and we can solve Equation 4 for the components of  $\bar{\mathbf{T}}$  up to an unknown scale factor. Because  $\bar{\mathbf{T}}$  is only known up to an unknown scale for each image in which the origin marker is visible, we can select an arbitrary scale factor for



**Figure 4: Geometry for estimation of absolute camera position using unit baseline vectors**

one image, thereby fixing that image's absolute camera position in the world coordinate frame.

If we are working with more than one image, the remaining absolute camera positions must also be fixed. Provided that at least two markers are shared between a pair of images, a unit vector  $\hat{\mathbf{B}}$  may be found that represents the relative direction of translation between the two cameras [14, 4, 6]. For two images  $l$  and  $r$ , let  $\hat{\mathbf{M}}_l$  be a unit vector corresponding to the image position  $\bar{\mathbf{I}}_l$  of a shared marker in the first image:

$$\hat{\mathbf{M}}_l = \mathbf{R}_l^{-1} \cdot \frac{\mathbf{K}_l^{-1} \cdot \bar{\mathbf{I}}_l}{\|\mathbf{K}_l^{-1} \cdot \bar{\mathbf{I}}_l\|} \quad (5)$$

and similarly for the second image. The best fit for the baseline vector between images  $l$  and  $r$  is the eigenvector associated with the smallest eigenvalue<sup>2</sup> of the  $3 \times 3$  matrix  $\Gamma$ :

$$\Gamma = \sum_{i=1}^N \hat{\mathbf{C}}_i \hat{\mathbf{C}}_i^T \quad (6)$$

where, for the  $i$ th shared marker,  $i = 1 \dots N$

$$\hat{\mathbf{C}}_i = \hat{\mathbf{M}}_{l_i} \times \hat{\mathbf{M}}_{r_i} \quad (7)$$

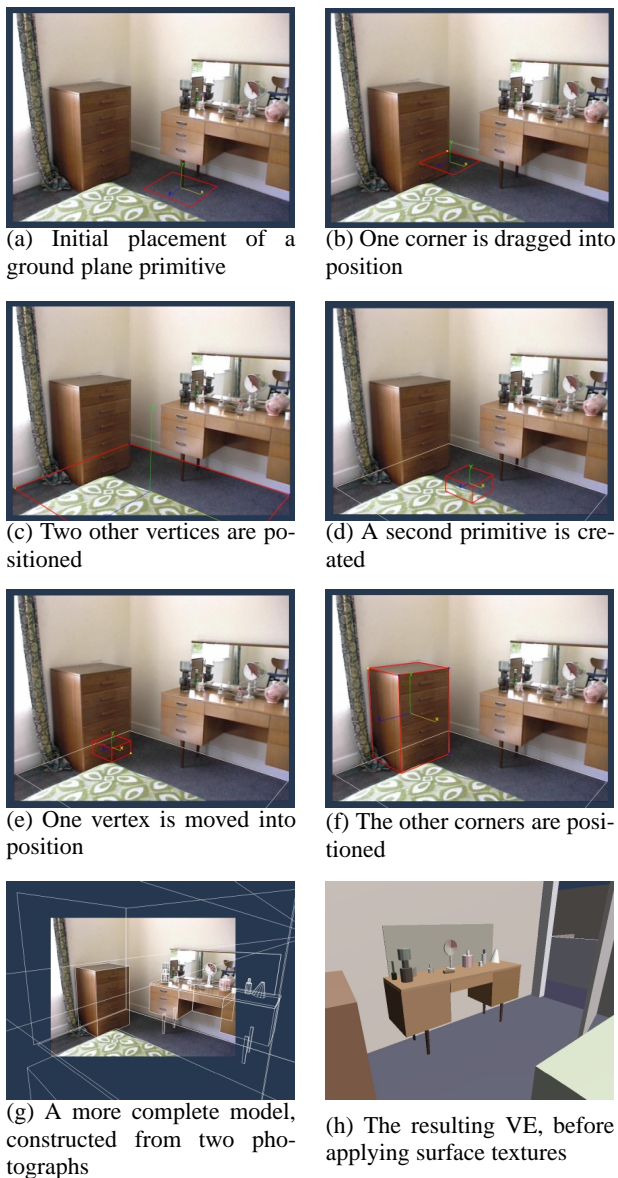
Once the unit baseline vector  $\hat{\mathbf{B}}_{jk}$  has been computed between image  $j$ , for which the absolute camera position is known, and another image  $k$  which shares the origin marker, we can estimate the absolute camera position for image  $k$  by calculating the intersection of the line from the origin  $\bar{\mathbf{O}}$  in direction  $\hat{\mathbf{T}}_k$  and the line from the known absolute camera position  $\bar{\mathbf{T}}_j$  in direction  $\hat{\mathbf{B}}_{jk}$  (Figure 4). Images in which the origin marker is not visible must be registered to the model later, after the reconstruction process has started. This will be discussed further in Section 3.2.

### 3. MODEL RECONSTRUCTION

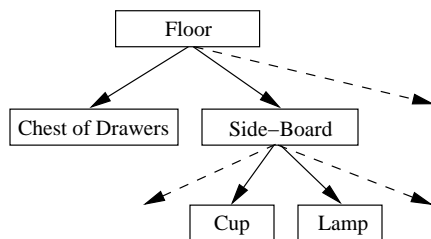
After a small number of images have been calibrated, we are in a position to start reconstructing the model of the scene. We build the three-dimensional model by interactively specifying the placement of three-dimensional parametric primitives, such as cubes, cylinders, cones etc.. in an image. The user manipulates the position and size of these primitives in image space, attempting to match the features in the image. As these manipulations are performed, corresponding updates to the three-dimensional structure of the scene are made using a non-linear optimization algorithm (Section 3.1).

An example of the placement procedure is given in Figure 5. In Figure 5(a), a horizontal ground-plane primitive has been created

<sup>2</sup>We compute the eigenvectors and eigenvalues using LAPACK, available from <http://www.netlib.org>



**Figure 5: An example of the interactive placement of primitives to match image features.**



**Figure 6: A section of a typical scene-graph constructed to represent the relationships between objects in the VE.**

by the user. In this case, the primitive is parameterized by its length and width. Two sets of edges are overlaid on top of the image. White edges show the projections of the edges of the primitive onto the image-plane. Red edges are those which the user manipulates in order to set the correct position and size of the primitive. In Figure 5(a), the white edges are directly underneath the red edges, and are therefore not visible.

The user then selects one vertex of this ground-plane, and positions it in the correct position in the image (Figure 5(b)). As this vertex is moved, it alters the position of the red edges in the image. When this occurs, we automatically update the estimates of the primitive’s parameters and position in order to minimize the difference between the red edges and the projected white edges. The parameters are estimated using a non-linear minimisation process<sup>3</sup>, the details of which are given in Section 3.1.

In Figure 5(c), the user has moved two more vertices of this primitive, thereby fixing the position of the ground-plane. Once this is done, additional primitives may be added to the environment. In Figure 5(d), a box primitive is created which will be used to model the chest of drawers on the left of the image. As successive primitives are added, we can specify their relationships to the other primitives in the scene. These relationships are given in terms of constraints on the position and orientation of each object, and allow us to interactively construct a scene hierarchy as we build the environment. In this case, the chest of drawers is resting on the floor, so we constrain its lower face to be at the same height as the ground-plane, and only allow it to rotate around its vertical axis. As this new object’s vertices are positioned in the image (Figures 5(e) and (f)), the optimization procedure obeys the constraints imposed on both the primitive’s parameters and the translation and rotation parameters that specify its relationship with the ground-plane. A more complete set of primitives is shown in Figure 5(g), and the resulting VE in Figure 5(h).

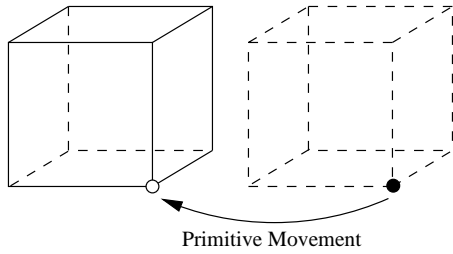
As further primitives are created, a scene-graph is incrementally constructed that specifies the relationship between the objects in the scene. A section of a typical scene-graph is given in Figure 6. Primitives may be collected together to form separate virtual objects such as the cup or lamp, maintaining the relationship constraints specified during the construction process. This scene-graph is then used by our system to aid in constraining user manipulation of the objects.

A wide range of primitives is available for constructing objects. For those more complex objects which cannot be represented as combinations of simple primitives we are currently investigating the use of dense-correspondence algorithms [21] to extract their surfaces from the images.

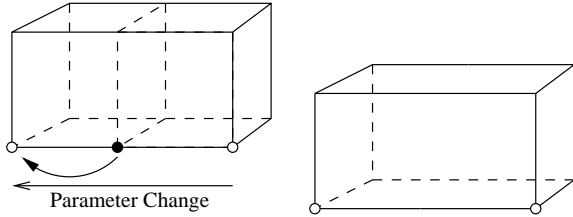
### 3.1 Parameter Estimation

During the process of primitive placement, the non-linear optimization algorithm attempts to minimize the difference between the red edges the user is manipulating, and the white edges, which are projections of the three-dimensional primitive onto the image plane. In order to keep this process robust and efficient, we must carefully select which parameters to optimize. This is achieved using a process of vertex “fixing”, whereby the user interactively fixes vertex

<sup>3</sup>We use a bounded version of the Levenberg-Marquardt minimization algorithm available from the PORT library at <http://www.netlib.org>



(a) When no vertices are fixed, the position of the primitive in the world coordinate frame is determined by any vertex movements.



(b) Movement of additional vertices causes changes in one or more of the primitive’s parameters. After fixing the second vertex, the primitive has been positioned and sized to match a feature in the image (right).

**Figure 7: The selection of which parameters to change depends on the configuration of fixed vertices in the primitive. White vertices are fixed, and black vertices are being moved by the user.**

projections to appropriate positions in image-space, thereby constraining the position of the vertex in the world-coordinate frame.

Figure 7 illustrates this process, and shows how it is used to manipulate primitive parameters. On the right of Figure 7(a), we show the projection of a box primitive onto the image-plane. Initially, no vertices are fixed. As the user selects a vertex and moves the primitive to the left, the position of the primitive in the world coordinate frame is updated. After the user has positioned one vertex, it becomes fixed to the image. Movement of other vertices (Figure 7(b)) now causes one or more parameters of the primitive to be updated, so that the previously fixed vertex remains in position.

As long as one primitive is being positioned in relationship to another, selecting its position in the world-coordinate frame (Figure 7 (a) above) will be possible. Without any translational constraints, however, the solution for its position will be undefined. This happens when the first primitive is being placed in the scene, and in this case, we specify a translational relationship with respect to one of the markers placed on the image during the calibration process.

During the process of primitive placement, we also update the translational and rotational relationships between the current primitive and its immediate parent and children in the hierarchy. In order to keep the optimization of the rotational components robust, we perform local re-parameterizations at each iteration of the minimization algorithm [13]. Typically, the total number of parameters which we must optimize over is less than 30, and the optimization algorithm is capable of running in real-time, as the user drags and fixes vertices over an image.

### 3.2 Re-Calibration



**Figure 8: An off-screen frame-buffer (right) is used to store the identification number for each primitive in the model (left).**

Due to the fact that calibration of extrinsic parameters, particularly camera position, from the shared markers is not as accurate as is often required, it sometimes becomes necessary to re-calibrate the camera positions and object parameters during the reconstruction process. This also gives us the opportunity to calibrate to position of images which did not share the origin marker.

Re-calibration is performed by repeating the non-linear optimization over all the object and relationship parameters, as well as camera position and rotation, using the current values as an initial guess. Typically, this process takes several seconds, and noticeably improves the quality of the reconstruction.

The intrinsic parameters and camera orientations of additional images are calibrated using the same vanishing-point calibration discussed in Section 2. Provided that two or more primitive vertices of known position are visible in the image, their absolute camera translations can then be found using similar techniques to Section 2.1.

## 4. TEXTURE EXTRACTION

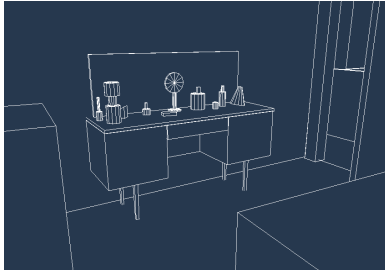
While positioning primitives, we are automatically specifying the spatial regions that the primitives occupy within each image. These regions can be used to extract surface colour estimates or textures for each primitive at any stage of the reconstruction process.

The image regions occupied by each primitive are identified using a standard hidden-surface removal algorithm. A BSP-tree [12] is first constructed, using the reconstructed three-dimensional model of the scene. The BSP-tree encodes the spatial positioning of all faces of the scene primitives, and it is traversed to determine the back-to-front ordering of each primitive face. For each face, the camera’s position, orientation and calibration matrix are used to determine the image-space coordinates,  $\bar{\mathbf{I}}$  of each vertex  $\bar{\mathbf{V}}$ :

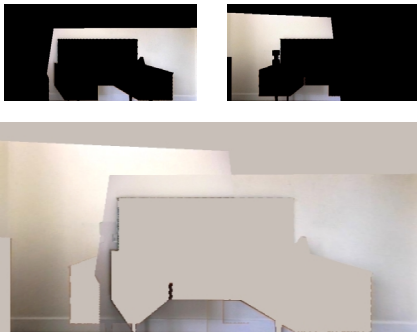
$$\bar{\mathbf{I}} = \mathbf{KRT} \cdot \bar{\mathbf{V}} \quad (8)$$

These image-space coordinates are then used to draw each primitive into an off-screen frame-buffer. The frame-buffer stores a colour-coded identification number of the closest primitive at each pixel in an image (Figure 8).

Once such a buffer has been constructed for each image, we can use it to extract surface colours or textures for each primitive. Deciding whether a particular pixel in an image should contribute to each primitive is done by simply comparing the identification number stored in the frame-buffer with the primitive’s own identification number. Simple estimates of surface colour can be found by averaging the colours of all pixels associated with one face of a primitive. An example VE which has been coloured in this way is shown in Figure 9.



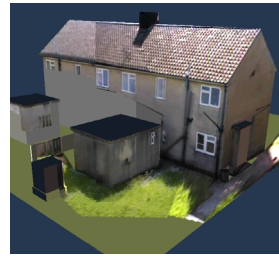
**Figure 9: Model reconstruction in wire-frame (top), surface colour (middle), and texture (bottom).**



**Figure 10: Textures may be extracted from multiple images (top) and merged together (bottom).**



**Figure 11: Manual reconstruction of a real crime-scene, built using traditional techniques.**



**Figure 12: Reconstructed exterior and interior scenes.**

Alternatively, surface textures may be extracted from each image. This is achieved by placing a grid of texels over each surface, and calculating the projected position of each texel in an image. Those texels which fall onto pixels whose identification number matches that of the primitive are placed into the texture map. Figure 10 shows an example of this, with a pair of texture maps extracted from two images. Areas coloured black indicate regions of the texture map which are not visible in the image.

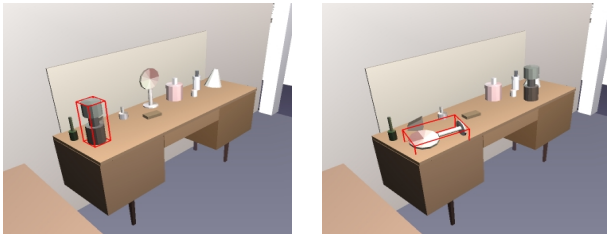
After textures have been extracted from each image, they must be merged together. For texels visible in more than one image, we perform a simple averaging procedure. We calculate a weighted average of texel colours, where each weight is the product of the projected texel size and the cosine of the angle between the surface normal and direction towards the camera position. Finally, we fill in regions of the texture map which are not visible in any image with the average texel colour (Figure 10, bottom). We are currently investigating more complex blending methods [18] and histogram matching techniques to improve texture extraction, as well as texture synthesis algorithms to fill in missing regions. We are also developing inverse illumination methods [20] to improve the quality of the estimated textures and materials.

## 5. APPLICATION CASE-STUDY

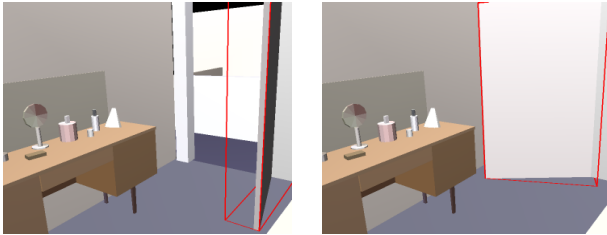
We are interested in applying these techniques and algorithms to real-world problems, where the ability to quickly construct an accurate VE corresponding to a real scene can be an extremely powerful tool. In particular, we are studying the problem of constructing VEs of crime scenes, using forensic photographs 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) [1].

An earlier pilot project [17, 15] 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 (Figure 11). The construction was extremely labour-intensive, and the resulting VE was essentially passive – it was not possible for a user to interact in any meaningful way with objects in the scene. However, discussions with police officers and forensic scientists and trainers who had evaluated the pilot project demonstrated that VE reconstructions of crime scenes can be of great benefit to scene-of-crime officers and forensic scientists, 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.

We have since applied the techniques presented in this paper to the



(a) Interactive object movement. Objects were constructed from several primitives, but may be manipulated as single entities.



(b) The door rotates around its vertical axis and closes by obeying rotational constraints specified during the reconstruction process.

**Figure 13: Examples of interactive object manipulation in the VE.**

reconstruction of a real scene – a house at the NTCSSCI, in which crime scenes are simulated for the purpose of training police officers and forensic personnel. A simple environment has been reconstructed from a small set of images taken with a hand-held digital video camera. The exterior of the building and a section of the interior have been reconstructed, and images of the resulting VE are shown in Figure 12.

Typical examples of user-interaction within the interior VE are also given in Figure 13. During user interaction, the translational and rotational constraints specified during the reconstruction of the environment were used to constrain object movement. For example, in Figure 13(b), the rotational constraint that the door can only rotate around a vertical axis was used to constrain its movement whilst the user “closed” the door.

## 5.1 Forensic Applications

There are several specific uses of reconstructed VEs representing crime scenes which have already been proposed, which future work will investigate these in detail. This section outlines a number of the proposals.

- **Data analysis.** The fact that the VE is a true three-dimensional reconstruction of the real scene affords the opportunity to superimpose onto the scene visualizations of data captured during forensic analysis. One important application would be the visualisation of bullet trajectories and forensic evidence.
- **Witness statement evaluation.** Most scenes of crime are short-lived. In many cases it is impractical to leave the scene undisturbed indefinitely, and after investigators and forensic personnel have visited the scene to collect evidence, the scene must soon be cleaned, tidied, and returned to its normal function. In effect, the actual scene of crime has disappeared. Capturing the original scene as a VE, however, “freezes” it, and allows a degree of subsequent investigation to take place.

One application would be to evaluate the accuracy of witness statements. For example, it is simple to determine what parts of the scene are visible from different vantage points. If a witness claimed to be able to see an object or some action from their viewpoint, this could be checked interactively using the VE – an evaluation which can prove awkward using only paper data and photographs.

- **Office briefings.** A reconstructed scene of crime VE has significant potential as a briefing tool. Using a large-screen stereoscopic display, multiple participants can share in interactive analysis of the layout of the scene, and the superimposition of forensic and other data as described above. It is our experience that the combination of a large-screen stereoscopic display, a detailed geometrical model of the scene, realistic lighting, and interactive walk-through, results in most observers having a useful sense of “presence” in the scene. This has obvious psychological benefits when the observers are exploring the relationships between objects and conditions within the scene.
- **Hypothesis evaluation.** Investigating officers often wish to suggest and test alternative hypotheses for how events may have occurred at a scene of crime. Traditionally this has involved the use of drawings and partial reconstructions using scale or life-sized models. A VE provides an alternative approach which is in many ways more flexible. For example, objects connected with the crime might be interactively moved around the scene in order to determine how they might have been used or moved; bullet trajectories and impact data might be used to estimate the positions of individuals using weapons, for example.
- **Training.** Within the police force training now accounts for a significant share of the budget. Thus, the development of VEs for training is a fruitful area to explore. Current methods for training officers include the use of buildings which contain full-size purpose-built scenes of crime, in which crimes are simulated, and which contain “evidence” for trainee officers to identify and evaluate. An interactive VE, however, would provide the ability to move furniture, open and close doors, adjust lighting conditions, as well as moving around, checking sight lines, and so on. Creating traditional “real-life” scenes is a costly and labour-intensive activity, and while they are likely to remain a vital “hands-on” aspect of training, VEs have the potential for assisting with this kind of evidence-gathering training at a greatly reduced cost.
- **Lighting evaluation.** Currently, scenes of crime are recorded using still photographs and video. Because artificial lighting will sometimes be used in order to capture images which are bright enough for subsequent study, the lighting in the images will be incorrect. Further, the crime may actually have been committed in the dark, or under greatly different lighting conditions than captured in the forensic images. With a VE, we have the facility to experiment with different lighting conditions, by changing the parameters associated with the light sources (including light coming through windows), and recomputing the overall illumination of the scene. This provides a powerful investigative tool which normal photography and video recording cannot offer.
- **Security planning and crime prevention.** A VE has potential for planning security operations involving the protection

of individuals. Such models could also be used for briefing colleagues and other agencies. For crime prevention, the emphasis would be on designing new buildings, or housing schemes, or re-developing existing ones. Different views could be obtained and different lighting conditions simulated.

## 6. CONCLUSION

In this paper, we have presented methods for semi-automatically reconstructing Virtual Environments from sets of photographs. We have discussed the methods we use to calibrate cameras, as well as our incremental method for model building. We have shown examples of an application case study, namely the reconstruction of virtual crime scenes, and outlined several uses to which these reconstructions could be put.

Future work includes evaluating the accuracy of the geometric reconstructions, and investigating improved methods of texture extraction. We are also currently developing inverse-illumination algorithms that attempt to estimate surface material properties. Application of a global-illumination algorithm will then allow us to accurately re-light the VE under different lighting conditions.

## 7. 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 UK National Training Centre for Scientific Support to Crime Investigation for their continued support.

## 8. REFERENCES

- [1] <http://www.forensic-training.police.uk/>.
- [2] P. Beardsley, P. Torr, A., and Zisserman. 3D model acquisition from extended image sequences. In *Proc. 4th European Conference on Computer Vision, LNCS 1065, Cambridge*, pages 683–695, 1996.
- [3] P. A. Beardsley, A. Zisserman, and D. W. Murray. Sequential updating of projective and affine structure from motion. *International Journal of Computer Vision*, 23(3):235–260, 1997.
- [4] S. Becker and V. M. B. Jr. Semi-automatic 3-d model extraction from uncalibrated 2-d camera views. *SPIE Symposium on Electronic Imaging: Science & Technology, San Jose*, February 1995.
- [5] B. Caprile and V. Torre. Using vanishing points for camera calibration. *International Journal of Computer Vision*, 4:127–139, 1990.
- [6] R. Cipolla and E. Boyer. 3d model acquisition from uncalibrated images. *Proc. IAPR Workshop on Machine Vision Applications, Chiba, Japan*, pages 559–568, November 1998.
- [7] R. Cipolla, T. Drummond, and D. Robertson. Camera calibration from vanishing points in images of architectural scenes. *Proc. British Machine Vision Conference, Nottingham*, 2:382–391, September 1999.
- [8] R. Cipolla, D. Robertson, and E. Boyer. Photobuilder - 3d models of architectural scenes from uncalibrated images. *Proc. IEEE International Conference on Multimedia Computing and Systems*, 1:25–31, June 1999.
- [9] P. Debevec, C. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *Proc. SIGGRAPH 96, Computer Graphics Proceedings, Annual Conference Series*, pages 11–20, August 1996.
- [10] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, 1993.
- [11] A. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. In *Proc. European Conference on Computer Vision*, pages 311–326. Springer-Verlag, June 1998.
- [12] H. Fuchs, Z. M. Kedem, and B. Naylor. Predetermining visibility priority in 3-d scenes. *Computer Graphics (Proceedings of SIGGRAPH 79)*, 13(3):175–181, August 1979.
- [13] R. Hartley. Euclidean reconstruction from uncalibrated views. *Applications of Invariance in Computer Vision, LNCS-Series*, 825:237–256, 1994.
- [14] B. Horn. Relative orientation. *International Journal of Computer Vision*, 4:59–78, 1990.
- [15] T. Howard, S. Gibson, and A. Murta. Virtual environments for scene of crime reconstruction and analysis. In *Proc. SPIE/IS&T, San Jose, CA*, volume 3960, January 2000.
- [16] D. Liebowitz, A. Criminisi, and A. Zisserman. Creating architectural models from images. *Computer Graphics Forum (Proc. Eurographics '99)*, 18(3), September 1999.
- [17] A. Murta, S. Gibson, T. Howard, R. Hubbard, and A. West. Modelling and rendering for scene of crime reconstruction: A case study. In *Proc. Eurographics UK, Leeds*, pages 169–173, March 1998.
- [18] E. Ofek, E. Shilat, A. Rappoport, and M. Werman. Multiresolution textures from image sequences. *IEEE Computer Graphics and Applications*, 17(2), March–April 1997.
- [19] C. Taylor and D. Kriegman. Structure and motion from line segments in multiple images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, November 1995.
- [20] Y. Yu, P. Debevec, J. Malik, and T. Hawkins. Inverse global illumination: Recovering reflectance models of real scenes from photographs. *Proc. SIGGRAPH 99, Computer Graphics Proceedings, Annual Conference Series*, pages 215–224, August 1999.
- [21] C. L. Zitnick and T. Kanade. A volumetric iterative approach to stereo matching and occlusion detection. Technical Report CMU-RI-TR-98-30, The Robotics Institute, Carnegie Mellon University, December 1998.