A Combined Studio Production System for 3-D Capturing of Live Action and Immersive Actor Feedback

Oliver Grau, Tim Pullen, and Graham A. Thomas

***Abstract*—A new studio system for the production of three-dimensional (3-D) content is introduced. The system combines the ability to capture dynamic scenes, based on a multicamera system in a chroma-key environment, with a view-dependent projection for actor feedback. The system allows the generation and rendering of 3-D models in preview quality for on-set visualization in real time and in high quality for postproduction applications in an offline phase. The shape reconstruction is based on the computation of the visual hull. The view-dependent projection component allows actors to be immersed into a virtual scene and to interact with virtual components. The functional modules of the system were implemented in a highly distributed system using standard, inexpensive IT components. Therefore, the system is quite scalable and can be fitted into most studio environments. For the integration of the system components, a middleware architecture was developed.**

***Index Terms*—Distributed system, studio system, three-dimensional (3-D) reconstruction, view-dependent projection, virtual reality.**

I. Introduction

DUE to recent advances in three-dimensional (3-D) reconstruction, there are a number of applications that make increasing use of captured 3-D models, particularly in productions that merge real and virtual scene content. This contribution introduces a new studio system that is able to capture the 3-D shape and texture of an actor or any dynamic object and at the same time gives the actor an immersive feedback of the virtual scene.

The production of 3-D content has many applications in communications, broadcast, and film applications [1]. Content generation is generally divided into two stages: the reconstruction of the 3-D shape and the creation of a texture map. In recent image-based modeling techniques, such as that in [2], these processes are not necessarily distinct, i.e., there is no explicit texture map generated, instead image interpolation techniques are applied during the rendering stage. Production or modeling systems for nonexisting, virtual content still mainly use 3-D surface models and material descriptions, like texture maps. For integration of virtual and real content in current 3-D production systems, the preferred method to represent the virtual elements is also with separate 3-D surface description and texture.

Manuscript received January 8, 2003; revised September 18, 2003. This work was supported by the European Project IST-2000-28436 ORIGAMI.

The authors are with the Research and Development Department, British Broadcasting Corporation, Surrey KT20 6NP, U.K. (e-mail: Oliver.Grau@rd.bbc.co.uk; Tim.Pullen@rd.bbc.co.uk; Graham.Thomas@rd.bbc.co.uk).

Digital Object Identifier 10.1109/TCSVT.2004.823397

In recent years, a great deal of research has been carried out in the field of 3-D scene reconstruction. The approaches can be divided into active and passive methods.

Active 3-D reconstruction systems use an active illumination technique and are usually more robust compared to most passive techniques. There are several products available using active techniques from companies such as lasers or structured light, like Cyberware (U.S.A.), Wicks & Wilson (U.K.), Vitronic (Germany), and others. Unfortunately, none of these systems are able to capture a moving 3-D scene including texture at a normal video frame rate. Promising systems that do allow capture at a full frame rate are, e.g., the Z-Cam from 3DV Systems (Israel) [3], which uses a depth-sensing method based on time-of-flight of light, and the face scanner from Eyetronics (Belgium) based on structured light. Both systems create a 3-D profile of the scene (or face).

Passive 3-D reconstruction techniques are use only camera images without any active illumination. An example of this class of techniques is stereo vision [4]–[8]. These systems are not currently in common use in a studio environment, mainly due to their restrictions in accuracy and robustness and the difficulty of achieving real-time operation.

For the 3-D-capture of dynamic scenes, in particular of actors in a studio environment, the computation of the visual hull from two-dimensional (2-D) silhouettes has shown to be quite robust and suited for many practical object classes [2], [1], [9]. The approach is based on the shape from silhouette method [10]–[12], which is relatively simple and can be computed in real time in a specially equipped studio using chroma-keying or difference-keying techniques.

In 3-D content production where an actor is integrated into a virtual scenario, like a virtual studio, the on-set visualization of the virtual scene for the actor becomes an important issue. In particular, if the actor or presenter has to interact with a virtual object, e.g., a virtual character, then there is often a difference between the direction in which the actor looks and the position of the virtual character that he should be looking at (the so-called eye-line problem). This can be quite disturbing in the final program.

An approach to provide the actor or presenter with a visual cue of objects in a virtual set is described in [13]. The system projects an outline of virtual objects onto the floor and walls. However, this method is restricted to show only the point of intersection of the virtual objects with the particular floor or wall. This means that a virtual actor in the scene would only be visualized as footprints and the eye-line problem persists.

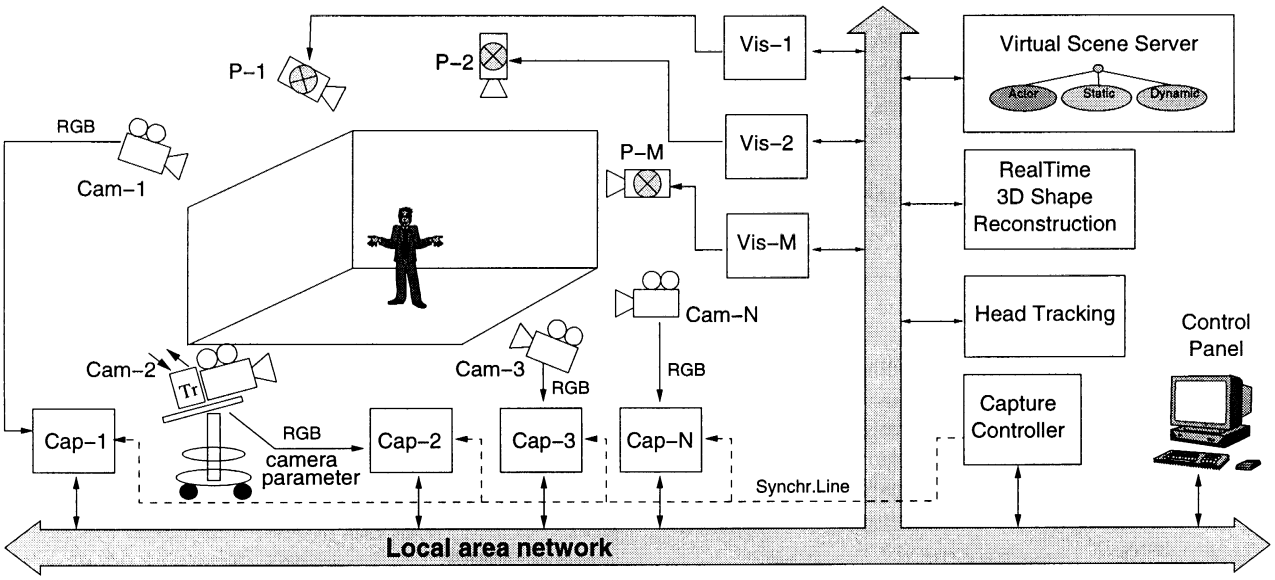

Fig. 1. System overview. Each camera (Cam-1 .. Cam-N) is connected to a capturing server (Cap-1 .. Cap-N). Equally the data projectors (P-1 .. P-N) are driven by a projection server (Vis-1 .. Vis-N).

Systems that do provide the required functionality are projection-based virtual reality (VR) systems, like the CAVE [14] or the office of the future [15]. The main application of these systems is to provide an immersive, collaborative environment. Therefore, a CAVE system tracks the position of the viewer's head and computes an image for that particular viewpoint, which is then projected onto a large screen forming one wall of the environment. Several screens are usually used to provide a wide field-of-view, and some installations provide an all-around view using six projection screens that completely surround the viewer. Therefore, it is possible to present objects that appear virtual in space, giving the viewer an immersive experience. The head position of the viewer is usually tracked using a special device, e.g., a helmet with an electromagnetic transmitter, and, if stereo projection is desired, a pair of shutter glasses must be worn. Such devices cannot be used in a production environment, because they would be visible in the final program.

Although such collaborative projection-based VR systems would probably provide a good immersive feedback for an actor, they are not designed to create 3-D models of the person. The main problem with the integration of a 3-D capturing component into a CAVE-like system is that there should not be any interference between both subsystems. A concept to integrate 3-D capturing into a CAVE-like system was proposed in the blue-c system [16]. This system uses back-projection onto a special screen that can be switched in three cycles between transparent and opaque modes. In the transparent mode, images are taken in parallel from several cameras. In the opaque mode, a projector is projecting an image for the left and right eye of the virtual scene in two separate cycles in order to give a stereo cue. The user has to wear shutter glasses that separate the images for both eyes. For the 3-D reconstruction, an approach similar to that in [2] based on difference keying is proposed.

The system discussed in this contribution was mainly designed for the production of 3-D content. Therefore, the quality of the final 3-D content has the highest priority. Furthermore, the system is intended to be used in a standard studio environment. Taking these requirements into account, a view-dependent front-projection system was developed that requires less space than a rear projection system and can be fitted into most existing studios. Images are projected onto a special retroreflective cloth that allows a robust chroma-key in conjunction with cameras equipped with a ring of monochrome (blue) LEDs.

A problem in a front-projection system is that there could be projected light falling onto an actor. One way to make this unwanted light invisible to the cameras is by operating the cameras in a shuttered mode and projecting only during the time that the shutters are closed, as was proposed in [13]. One disadvantage of this approach is that the light intensity of the projectors must be very high in order for the actors to be able to see the projected image under normal studio lighting conditions. Furthermore, the need to shutter the cameras may force the use of higher levels of studio illumination. Also, there remains the potential problem that the actors may be dazzled when looking toward a projector. An alternative approach, proposed in this contribution, is to generate a mask from the captured 3-D shape and overlay it onto the projected image. This removes the need for high-intensity projectors with a rapid temporal response, so that conventional data projectors may be used, and this also solves the problem of dazzle.

The paper is organized as follows. Section II gives an overview of the system. In Section III, the arrangement in the studio is introduced. Section IV presents the 3-D reconstruction method. The system architecture and real-time rendering method are described and discussed in Section V. Finally, we conclude this paper in Section VI.

II. System Overview

The system is composed of a number of modular components, as depicted in Fig. 1. The communication and data exchange between these components is predominantly based upon

a local-area network (LAN) and standard IT components. This concept provides a cost-effective way of implementing a system which can be easily configured to suit particular requirements. For example, the number of cameras can be varied depending on the available space in the studio and the specific production needs.

Each camera (Cam-1 .. Cam-N) is connected to a capturing server (Cap-1 .. Cap-N). The capturing servers are standard PCs, equipped with a frame grabber card. Each capturing server also has a RAID disc array. This allows the incoming video from the cameras to be saved to disc as uncompressed video (704×576 pixel, 24-bit color resolution at 25 fps, progressive scan). This data is later processed in an offline phase to produce high-quality 3-D models for the final program.

Usually the cameras are fixed and their parameters are determined with a calibration procedure. In addition, cameras equipped with a real-time tracking system can be used, as shown for Cam-2 in Fig. 1. The real-time tracker delivers exact position and orientation and the internal camera parameters. We are using our previously developed "Free-D" camera tracking system [17] for our experiments.

In addition to their function as an image sequence recorder, the capturing servers provide several online services over the network. On request they can send the latest grabbed image and provide a chroma-keying service, allowing the other software components to request alpha masks. One of the components making use of this is the real-time 3-D shape reconstruction that synchronously requests alpha masks from all of the capture servers and uses these to compute a 3-D model of the actor. The methods used by this module are discussed in Section IV.

The 3-D data are used by the head tracker and passed to the virtual scene server. The head position of the actor is used by the projection system to render a view-dependent image of the scene.

The virtual scene server provides a description of the virtual scene. This includes static scene elements, usually the virtual set, dynamic parts, for example, any virtual characters involved in the scene, and the actor, provided by the 3-D shape reconstruction module. The virtual scene server synchronizes all scene updates and distributes the scene updates to it.

The visualization servers (Vis-1 .. Vis-M) are standard PCs equipped with OpenGL accelerated graphics cards that render the scene and drive the data projectors (P-1, .. , P-M). They access the latest scene data from the scene server and the actor's head position from the head tracking module. Each server uses this data, together with the calibrated position of the corresponding projector, to calculate and render the scene for the viewpoint of the actor. The generated image is then projected onto the retroreflective cloth.

The control panel is the main interface to the system and consists of several components.

1) The *camera remote control* allows the remote setting of camera parameters, like focus, zoom, and aperture. This is particularly important because one or more cameras may be hanging from the ceiling and hence will not be accessible.
2) The *capture control* is used to start and stop the capturing service on the servers.
3) The *animation control* is able to start predefined 3-D animations or control them live.
4) The *3-D preview* provides a preview of the production for the director and the camera operators. This allows the director to see a view of the scene from any position he chooses, allowing planning for camera shots. A textured 3-D mesh of the actor is inserted in the virtual scene to give a preview of the final composited program.

The control panel can be physically distributed over several workstations. This allows tasks to be delegated to different people or to be controlled from different locations. Moreover, the preview module can be instantiated several times, e.g., a view for the director, for a camera operator, for an animator, and so on. The particular viewpoint of each preview can be set individually and is dynamic.

The capture controller synchronizes control events and the actual capturing. Therefore, all capture servers are connected with a hardware line (indicated as "Synchr.Line" in Fig. 1) that allows frame-accurate synchronization of the capture.

III. Studio Setup

The main elements of the studio system are the cameras with their associated capture servers, the data projectors with their associated visualization servers, and the cyclorama surrounding the acting area.

A. Cameras and Capture Servers

Each fixed camera is mounted rigidly in positions chosen to give a view of the acting area from a range of different positions, including from above. We are using Sony DXC-9100P cameras, which have the useful capability of being able to operate at 25- or 50-Hz progressive scan, as well as in the more usual interlaced mode. For the capture purpose, the cameras are used in progressive mode. If interlaced mode is required as output format of the final program, then the image synthesis module (3-D renderer) has to provide this.

The RGB video signal from each camera is fed to a frame grabber in its respective capture server. The capture server also has an RS-232 connection to its camera, allowing remote control of the camera's parameters, including zoom, focus, and shutter speed. The cameras are synchronized using a video genlock signal. In order to ensure that each PC starts capturing at the same frame, a trigger signal is generated by the capture controller PC and fed to one of the handshaking line inputs on the RS-232 port of each capture server. This eliminates potential problems with network latency.

In addition to the fixed cameras, one or more movable cameras, equipped with a suitable tracking system, can also be used. This is particularly useful to provide close-ups of important parts of the scene, such as an actor's face. We currently have one camera equipped with the "Free-D" tracking system [17], which measures the position, orientation, and field-of-view of the camera at field rate. Typical measurement accuracies are of the order of 1 mm and 0.01 degrees. The data is received via an RS232 port on a PC and recorded to disk.

B. Retroreflective Cyclorama

A virtual studio usually relies on having a colored background and floor, so that the actors and props can be keyed onto the virtual background.

In order to generate a good key signal with low noise, it is necessary to have the colored background illuminated brightly and evenly. This gives rise to several problems, particularly when used in large studios:

- It is often time-consuming and difficult to provide bright and even illumination over a large area.
- It is sometimes difficult to light the actors to give the desired artistic effect, since the lighting is primarily determined by the technical requirements of the key generation. Scenes with low lighting levels pose particular problems.
- Areas where dark shadows cannot be avoided, such as under tables, do not produce clean key signals.
- Colored light scattered from the brightly lit background and floor lands on actors and props, changing their hue and giving an unnatural appearance. This is known as "spill."

In order to address these problems, we previously developed a retroreflective background cloth, in collaboration with a manufacturer.[1] The cloth is coated with a large number of randomly orientated microscopic half-silvered glass beads. Light entering the front of a bead is returned along the path it came from. By mounting a ring of lights around the camera lens having the chosen key color, the camera will see the background as having the desired uniform color, regardless of the setting of the studio lights. The reflectivity of the cloth is sufficiently high that a low level of illumination from the camera is sufficient, and the amount of light scattered from actors is generally negligible.

It is important that the cloth exhibits relatively uniform reflectivity regardless of the angle of incidence, since it needs to work on the floor as well as on the walls. Conventional retro-reflective material, such as that used on road signs, has all the glass beads oriented the same way, with the silvered side of the bead at the back. It thus only reflects strongly when the incident light is within about 60° of the surface normal. An example of such a material is 3M Scotchlite Reflective Material, no. 8850. Conversely, the cloth maintains a strong reflection beyond 80°, due to the random orientation of the half-silvered glass beads, and the rough nature of the cloth surface. Fig. 2 shows the measured reflectivity of a sample of cloth compared to that of some 3M material. The reflectivity for each sample is expressed as a ratio of the reflectivity for light reflecting from the sample parallel to the surface normal. Note that the absolute level of reflectivity of the cloth at normal incidence is about six times lower than that of the conventional material, but it is still highly reflective (appearing to be 50–100 times brighter than a sheet of white paper at a distance of several meters, illuminated by the same light).

Fig. 3 shows the cloth in use in a virtual studio.

C. Projecting Onto the Reflective Cloth

Occasionally, when chroma-keying is used in TV production, an image is projected at a low level onto a conventional blue or green chroma-key background, to give a visual cue to the presenter. One example is in some weather bulletins, where the weather map is rear-projected onto the blue screen behind the presenter, so that they can see where to point. However, in order to maintain an acceptable key, the projected image must be restricted both in its brightness and its color content, so a low-level monochrome image is usually used.

We have found that the reflective cloth works particularly well as a combined projection screen and chroma-key background. This is because the camera always sees the key-colored light much more strongly than other light scattered from the cloth, since the key light is mounted close to the camera lens. Conversely, the performer sees a roughly equal proportion of all light landing on the cloth, as the cloth scatters a significant proportion of incident light. Thus it is possible to give the performer a relatively high-contrast image whilst the camera sees an image composed mostly of the key color.

[1]The cloth is commercially available under the name Chromatte from Reflecmedia.

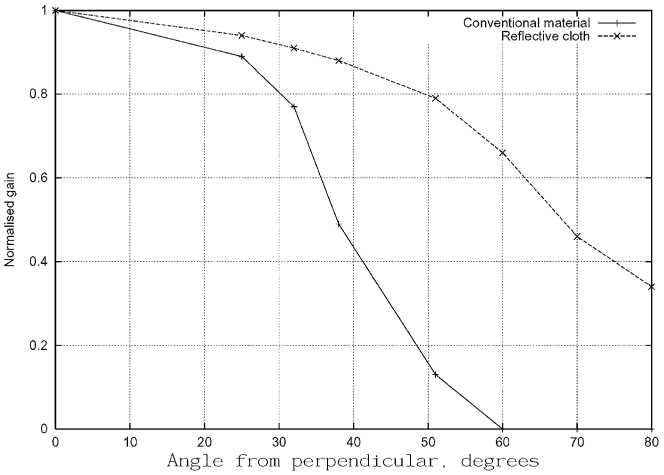

Fig. 2. Reflectivity of the cloth compared to conventional retroreflective material as a function of angle of incidence.

Fig. 3. Retroreflective cloth in use in a virtual studio.

D. Results

Fig. 4 shows a view from one camera fitted with a ring of 36 blue LEDs, each with a luminous intensity of 0.35 cd and an illumination angle of 45°, consuming a total of about 4 W. The image is sufficiently blue over the whole of the background to produce a good key signal, as shown in Fig. 5.

Fig. 4. Image from one camera.

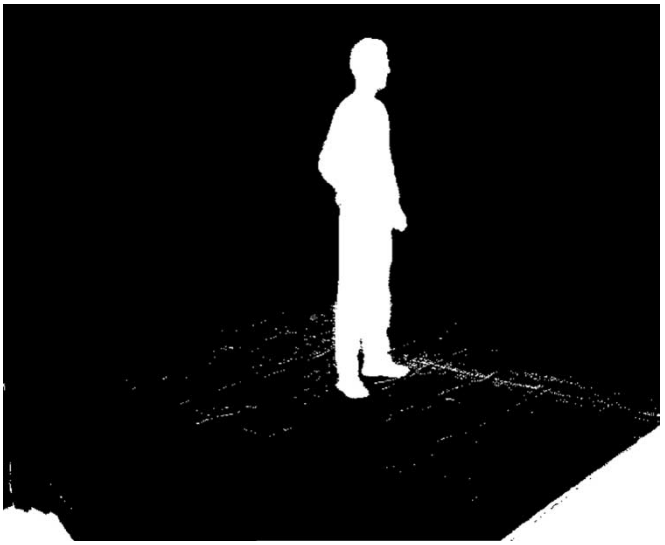

Fig. 5. Silhouette image computed with chroma-keying.

An example of projection onto the cloth is shown in Section V.

IV. 3-D Processing

The 3-D processing described in this section includes the calibration of the cameras, the 3-D reconstruction and the head tracking in the studio system. A texture mapping method we developed as part of the real-time rendering system is described in Section V.

A. Projection Model and Camera Calibration

For the 3-D reconstruction a pin-hole camera model is used, based on the CAHV notation [18]. In addition, radial distortions of the lenses are considered. As depicted in Fig. 6, the external parameters of a camera Cam-j are: the center point of the camera $\vec{C}$, the optical axis $\vec{A}$, the horizontal direction $\vec{H}_0$, and the vertical direction of the camera $\vec{V}_0$.

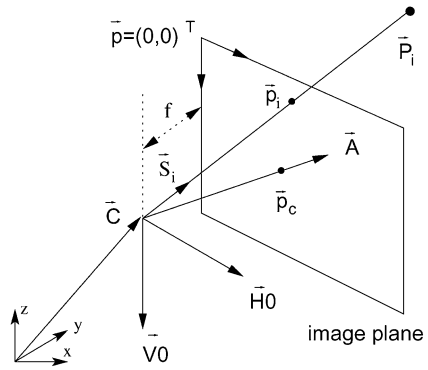

Fig. 6. Central projection camera model.

The vectors $(\vec{A}, \vec{H}_0, \vec{V}_0)$ define an orthonormal right-hand (local) coordinate system of the camera.

The internal camera parameters are specified by: the number of pixels N_x, N_y, focal length f, pixel spacing s_x, s_y, center point shift h_x, h_y, and radial distortion terms $(rd_3, rd_5, ..)$.

The center point or principal point $\vec{p_c}$ is the projection of $\vec{C}$ onto the image target. It is located in the middle of the camera target corrected by the center point shift (h_x, h_y) as follows:

$$\vec{p_c} = \left(\frac{N_x - 1}{2} + h_x, \frac{N_y - 1}{2} + h_y \right)^T. \tag{1}$$

The projection of a point $\vec{P_i}$ into the image plane can be expressed as

$$\vec{p_i} = (x_i, y_i)^T = \left(\frac{\left(\vec{P_i} - \vec{C}\right) \cdot \vec{H}}{\left(\vec{P_i} - \vec{C}\right) \cdot \vec{A}}, \frac{\left(\vec{P_i} - \vec{C}\right) \cdot \vec{V}}{\left(\vec{P_i} - \vec{C}\right) \cdot \vec{A}} \right)^T \tag{2}$$

with

$$\vec{H} = \frac{f}{s_x}\vec{H}_0 + h_x\vec{A}\,,\ \vec{V} = \frac{f}{s_y}\vec{V}_0 + h_y\vec{A}. \tag{3}$$

Radial distortions of a camera can be expressed as a correction relative to the center or principle point $\vec{p_c}$ in the image plane and is approximated as

$$r = r' + rd_3 r'^3 + rd_5 r'^5 + rd_7 r'^7 + \cdots \tag{4}$$

The distance r represents the undistorted and r' the distorted distance to the center point. Depending on the characteristics of the lenses used, only the third-order term and occasionally the fifth-order term of (4) are considered.

For the calibration of the cameras, a calibration technique was developed, similar to the Tsai method [19]. A single-stage process determines the parameters for all cameras simultaneously and places them in a defined reference frame in the studio, with the floor in the xz plane.

To carry out a calibration, around 8–12 images of a planar 1 m × 1 m calibration object are captured synchronously from all cameras. The object is moved around to explore most of the volume of the acting space. One image is taken with the object in a predefined position on the floor to set the absolute coordinate system. Other images are captured with the object lying elsewhere on the floor to help further constrain the xz plane.

Several additional images are captured with the object held in the air. The image coordinates of 100 feature points on the calibration object are extracted from each image. An iterative minimization process is used to adjust the unknown camera parameters, plus the position and orientation of the object in each frame, until convergence is reached. This typically takes about 10 iterations (1 min of processing time on a modern PC). Constraints can be placed on the object position, for example, to force the y coordinate and x and z angles to zero for the frames in which the object was on the floor. Typical errors after reprojecting the feature points into the images are of the order of 0.2 pixels rms. This level of error is also achieved using images of the calibration object that are omitted from the calibration process, confirming the validity of the calibration.

B. Real-Time 3-D Shape Reconstruction

The real-time 3-D shape reconstruction is required in order to provide an immediate feedback to the director and optionally a camera operator. Furthermore, we use the derived 3-D information to create a mask signal in the projection system. As mentioned previously, the camera images can also be stored on a disk array and used later in an offline phase. Both the real-time and offline 3-D shape reconstruction are based on the computation of the visual hull from object silhouettes. As the offline processing does not need to operate in real time, higher spatial resolutions and computationally more expensive algorithms like voxel coloring or shape carving [20] can be used.

This contribution focuses on the real-time visual hull reconstruction that has been implemented for the feedback system. The approach requires a set of silhouette images from calibrated cameras. A silhouette image is a binary image where each pixel indicates whether this pixel belongs to the object or not. The silhouette information can be determined by any suitable segmentation process. In most cases this is done using chroma-keying or difference keying. For the study in this paper, chroma-keying was used as depicted in Figs. 4 and 5.

The 3-D shape reconstruction can be formulated as the intersection of generalized cones of the silhouette images. A generalized cone is the union of visual rays from all silhouette points of a particular image. This intersection gives only an approximation of the real object shape and is called the visual hull. In particular, concavities cannot be modeled with this method, but for many practical tasks this restriction can be tolerated.

Many algorithms have been published for the computation of the visual hull, e.g., [21], [10]–[12]. These approaches solve the problem in a volumetric space representation. The most common of these representations is to subdivide a 3-D box in Euclidian three-space into a set of voxels of discrete size. In order to save memory, these are often represented as Octrees [10], [11] or run-length encoded [12]. An overview of volumetric reconstruction can be found in [20].

The method described here uses a line segment representation and a silhouette cut. The cut algorithm is similar to that proposed in [12], but our method shows less discretization noise in the reconstructed surfaces. In this representation, an object is sampled by a set of line segments. The set of line segments $\mathcal{S}$ is organized in a 2-D array as depicted in Fig. 7. The array axes $x_1 = m$ and $x_2 = n$ are the expansion base of the line segment

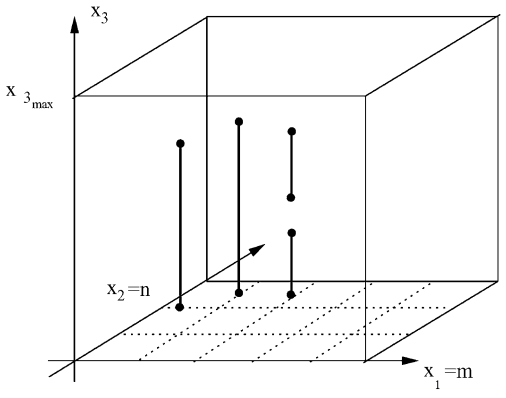

Fig. 7. Objects are sampled by a set of line segments.

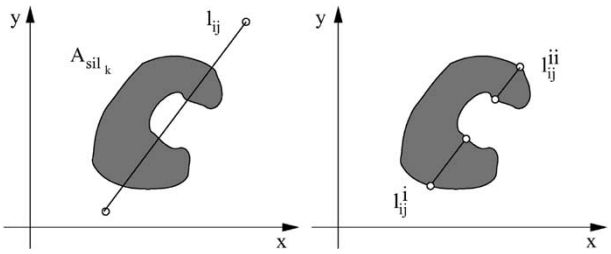

Fig. 8. Intersection of a line segment projection with a silhouette.

set. Each position (m, n) in this expansion base keeps a list of line segments, that can be empty or may have several entries. The line segments in a particular column (m, n) lie on the same base line as follows:

$$g(m,n) : \vec{x} = \vec{x_0} + \tau \vec{x_d}. \tag{5}$$

Usually the base lines are chosen to be parallel to the third, continuous axis x_3 over the expansion basis. For an expansion base in the xy plane, and continuous axis along the z axis, the line equations in local coordinates are

$$g(m,n) : (m\, s_{x_1}, n\, s_{x_2}, 0)^T + \tau(0,0,1)^T \tag{6}$$

with a line spacing of s_{x_1} and s_{x_2}. Each line segment is characterized by its start and end point $L_i = (\vec{P_{si}}, \vec{P_{ei}})$.

To initialize the line segment set for the silhouette cut the defined volume will be entirely populated with one initial line segment for each column (m, n) covering the definition range $0 \leq x_3 \leq x_{3_{\max}}$ of the volume. The silhouette cut algorithm for the line segment representation can be expressed as in Fig. 9.

In order to use the reconstructed 3-D model in a conventional rendering system, a surface description, usually a polygonal mesh, has to be generated. An algorithm that is often used is the marching cubes algorithm [22] which creates an isosurface of a volumetric data set. The marching cube algorithm is quite robust and fast. In this work, a modified marching cubes algorithm was developed that works directly on the line segment set.

C. Silhouette Cut for a Particular View Point

In (6), the base lines of the line segments were defined to be parallel to the expanded euclidian axis. An alternative parame-

For all silhouette images I_{sil_j}
For all line segments L_i in the line segment set $\mathcal{S}$
Project line segment L_i into silhouette image I_{sil_j} using camera projection parameter Cam_j. The result is a line segment l_{ij} (Fig. 8 left).
For each line segment l_{ij} create a list $P_{ij} = \{l_{ij}^i, .., l_{ij}^N\} = l_{ij} \cap A_{sil_k}$ with those line segments that are intersecting with the silhouette A_{sil_k}. This list might be empty if there is no intersection with the silhouette.
Replace L_i with the re-projected line segments in P_{ij}. The re-projection of a line segment can be found by intersecting the line of sight of each end points with the base line $g(m,n)$ of a line segment.

Fig. 9. Line-segment-based silhouette cut algorithm.

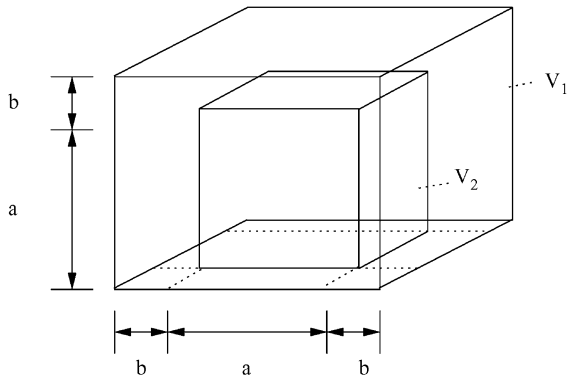

Fig. 10. Template for head tracking.

terization can be used, so that the base lines are lying along the lines of sight of a camera as follows:

$$g_{vp}(m,n) : \vec{x} = \vec{C} + \lambda \vec{S_i} \quad (7)$$

$$\vec{S_i} = ((ms_x - p_{c_x})\vec{A} - \vec{H}) \times ((ns_y - p_{c_y})\vec{A} - \vec{V}). \quad (8)$$

This parameterization is equivalent to a viewing frustum as used for ray tracing [23] or similar rendering algorithms. The silhouette cut algorithm as described above can be applied directly on this parameterization without any change.

Because of the chosen parameterization the line segments in a set $\mathcal{S}_{vp}$ represent one pixel of a particular viewpoint of any updated set of camera parameters. This idea is used in [2] to synthesise new images of the real object without creating an explicit surface model.

For the case that only a binary indication is needed of whether a pixel (m, n) belongs to the object or to the background, this can be implemented by testing if there is at least one line segment in the particular column. This feature can be used to generate a mask signal for the projector viewpoint and is discussed in Section V.

D. Head Tracking

The view-dependent actor feedback system requires information on where the head of the actor is. For this purpose a fast 3-D template filter is applied. The filter consists of two boxes, V_1 and V_2, as depicted in Fig. 10. The filter computes a match

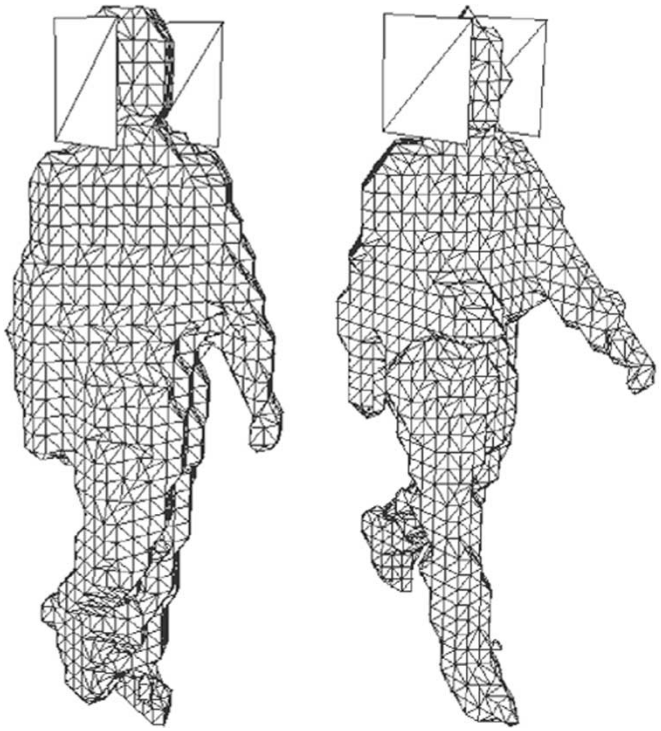

Fig. 11. Polygonal model generated from six images. The rectangles mark the found head position.

$M(P)$ for all positions $P = (m,n,k)$ in a (discrete) volume as follows:

$$M(P) = 2a_2(P) - a_1(P) \quad (9)$$

$$a1(P) = Match(\mathcal{S}, V_1(P)) \quad (10)$$

$$a2(P) = Match(\mathcal{S}, V_2(P)) \quad (11)$$

$$P_{Head} = P : M(P) \rightarrow \max. \quad (12)$$

The match function in (10) and (11) counts the number of voxels of S inside the test box $V_1(p)$ and $V_2(p)$.

E. Results

Fig. 11 shows two polygonal surface models generated from six cameras. The resolution of the line segment set was 64×64, covering 3 m $\times$ 3 m $\times$ 3 m. For this resolution, an update rate of approximately 15–20 fps can be computed by the surface reconstruction. Due to time delays in the overall system, the mesh update in the rendering subsystem is currently below this rate (ca. 5–10 fps). The main reason for these delays is the CORBA implementation currently used which is not well suited for real-time applications. A migration to a real-time CORBA implementation, like ACE Tao, and further code optimizations should improve the frame rate of the system significantly.

The 3-D shape reconstruction module requests the alpha mask from the current frame from each capturing server. This is calculated and run-length encoded to reduce network bandwidth. The resultant alpha mask is at 704 by 576 pixel resolution and is compressed to around 4000 bytes on average, excluding any protocol overheads. This allows the alpha masks from a number of capture servers to be transmitted over the network with minimal latency.

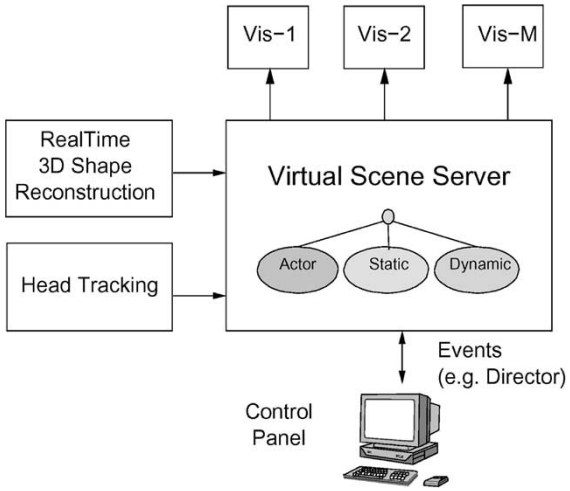

Fig. 12. Data flow in the real-time rendering system.

The head position found by the match filter, as described above, is overlaid in Fig. 11. The method is relatively reliable and can be applied on the 64×64 resolution with an update rate of approximately 5 fps.

V. Real-Time Rendering System

The real-time rendering system provides the view-dependent actor feedback and the preview for director and operators. It is distributed into several submodules and integrated into the studio system, as depicted in Fig. 1. The main components are: the virtual scene server, the visualization servers (Vis-M in Fig. 12) that drive the data projectors (P-M) and instances of the 3-D preview as part of the control panel.

This distributed system implements several methods and services: The mask generation to prevent actors being lit by the projectors, a real-time texturing module, the previsualization modules for the director and the view-dependent rendering. These modules make use of data provided by the 3-D shape reconstruction, the head tracker, and the live user input from the control panel. The next subsections explain the underlying data transport and the implemented services.

A. Data Transfer

The various components of the system communicate with each other using CORBA, a middleware software layer that allows programs to perform actions transparently on one or several software components running on the same or several computers. This allows the location of the various software components to be decided at runtime. The communication uses a polling mechanism, whereby one component can request the time code of the latest piece of data, such as an alpha-mask, and then requests the data itself if it has been updated since the last check. This allows only the pieces of data that are actually used to be moved around the computer network, cutting down the use of bandwidth. However, this adds latency to the system. For this reason, mechanisms that allow data to be "pushed" rather than "pulled" are being investigated. These include the use of multicast protocols instead of pure point-to-point connections.

The 3-D surface data of the actor, however, is "pushed" to the virtual scene server as soon as it has been created. This is done using a TCP/IP socket mechanism. The virtual scene server always needs the latest data, therefore this connection has a high priority.

B. Virtual Scene Server and Rendering Clients

The virtual scene server keeps the master copy of the current scene graph. It is updated by the real-time 3-D shape reconstruction, the head tracking, and events from the control panel. The latter is mainly influencing the dynamic scene, i.e., animations of the virtual scene components.

The master scene graph is distributed to all rendering clients that request it. These are the visualization servers (Vis-N in Fig. 12) that are driving the data projectors for the actor feedback and any instance of the director's view on the control panel. Each visualization server makes use of the actor's head position in order to create a view-dependent visualization. It also uses the 3-D shape model for the computation of a mask to avoid light falling on the actor. The viewers for director and operators allow the viewpoint and parameters of the virtual camera to be set independently and provide a texturing of the live actor 3-D model in order to generate a photo-realistic visualization.

C. Mask Generation

A mask is needed to prevent light from the projector falling on the actor. This is generated from the 3-D surface model. The most recently computed surface model is therefore placed into the virtual scene and rendered completely in black with the z buffer of the rendering system disabled. This guarantees that from the viewpoint of the projector all light rays that could fall onto the actor surface are masked. Due to the latency of the system, light may still fall on the edge of a moving actor, particularly during fast movement. Therefore, the mask is enlarged by a security factor that can be adapted to the latency and the fastest motion of the actor that is expected.

D. View-Dependent Rendering

The rendering engines in the visualization server for the projectors request the latest head position from the head tracker and use this to calculate and render the projected image from the point of view of the actor. This is done by setting a viewing frustum with the origin at the actor's head position. In addition to that a homographic distortion (eight parametric perspective image transform) is added to the projection matrix of the virtual camera, taking into account the relative positions of the projection screen (wall) and the projector position.

The view-dependent rendering allows the actor, if required, to keep looking at the face of the virtual character as he walks around him. That means the virtual scene components appear in space and the actor is "immersed."

The renderer also receives any updates in the scene from the virtual scene server. If the virtual character was to move, then the actor would see these movements. The combination of the scene updates and the viewpoint-dependent rendering thus allows complex interaction between the virtual and the real scene elements.

E. Texturing

For the director or camera operator, a renderer is provided that gives a previsualization of the final composited scene, i.e.,

Fig. 13. Viewpoint-dependent projection.

the virtual and real scene elements. This renderer receives updates from the virtual scene server and grabs the latest 3-D shape model of the actors. It can then generate a 3-D representation of the scene and allows the director to view the scene from any position. This position can be dynamically updated to allow simulation of shots where the camera is moving.

In order to give a more realistic image, the 3-D shape model of the actor is textured with a view from one of the cameras. Therefore, the renderer determines the studio camera that has the smallest angle to the virtual camera. The 3-D shape model is stamped with the time-code of the alpha masks used to generate it, so the renderer requests the image from that time-code from the relevant capturing server and uses it to texture the 3-D shape model.

F. Results

The studio system was used in a demonstration production. The interior of a museum was digitized and used as a virtual background. In addition to that, a "flying skull" was inserted.

Fig. 13 shows a child acting in the studio. The scene was projected onto the retroreflective cloth for his viewpoint. The boy was able to turn his head to look exactly toward the bright virtual object in the top-right (skull). In order to make the important objects recognizable for the actor, the contrast of these is set as high as possible, so the appearance is not necessarily realistic.

Fig. 14 shows the actual image rendered for one projector. The black region near the center is the mask generated from the 3-D actor model.

Fig. 15 shows the composited scene with texture for the "director's view." The image gives a previsualization of the composed scene. On this basis, the director can decide on changes to the camera perspectives or positions of virtual components or the actors.

VI. Conclusion

A studio system for the production of 3-D content was introduced. The system combines the ability to capture dynamic scenes in a multicamera system with an actor feedback component. A key element of the system is the special retroreflective cloth that allows robust chroma-keying and front-projection of synthetic images of the virtual scene at the same time.

Fig. 14. Projected image. The black region in the bottom center is the mask generated from the 3-D surface model.

Fig. 15. Director's view of two actors in a digitized model of a museum.

Program production with the system is divided into an online or capture phase and an offline phase. During the capture phase, the images from the multicamera system are stored for the later offline phase. In addition, the live captured camera images are used by a real-time 3-D reconstruction and head-tracking module. These modules pass the derived data to a rendering subsystem for on-set visualization.

The real-time shape reconstruction module uses the silhouettes of objects for the computation of their visual hull. The implementation uses a line segment representation of the 3-D space and is sufficiently fast to give update rates of 10–20 fps. The computed 3-D models are used in conjunction with real-time texturing for previsualization during the production phase. First test productions with the system have shown that the director found this previsualization to be very useful and of acceptable quality.

The head-tracking module is currently using the computed 3-D data as an input and determining the actor's head position from that. Although the implemented match filter is quite fast and robust, the overall computation shows some latency because it is based on the 3-D shape reconstruction. Therefore,

an image-based technique is envisaged, using a number of 2-D head-tracking programs running on the capturing servers.

For the previsualization, a real-time renderer was developed that shows the content of a (master) scene graph provided by the virtual scene server. The renderer can either be used to show the complete virtual scene including the digitized actors as a "director's view" or as a view-dependent projection for the actor. For the latter mode, we are using the reconstructed 3-D surface model of the actor to generate a mask. The mask prevents light from the data projectors from falling onto the actor and becoming visible in the final images. The advantage of this new method compared to time multiplexing, i.e., operating the projectors only during the period in which the cameras are shuttered, is that a normal data projector is able to project enough light so that the actor is able to see the scene under normal studio lighting conditions.

The functional modules of the studio system described were implemented in a highly distributed system using standard inexpensive IT components. Therefore, the system is quite scalable and can be adapted for most studio environments. For the integration of the individual system components, a CORBA middleware architecture was developed. The use of the platform-independent CORBA standard allows the exchange and update of components.

First test productions with the new system were quite successful and showed the potential of the system for cost-effective program making. Currently the use of the system is envisaged for TV and film productions, in particular for those that incorporate virtual scene components. The system has shown a high potential for cost saving in feature film production as a previsualization tool, in particular for special effects. A major benefit is the better feedback for the actor that allows correct eye-lines to virtual components and the synchronization of events between virtual and real. Furthermore, the preview for the director is a very helpful tool to support decisions that could otherwise only be made after postproduction. The postproduction is usually completely separated from the studio work and introduces additional workflow cycles.

The application range of the complete proposed 3-D modeling system covers several types of programs. Besides special effects, these include 3-D TV and interactive 3-D applications.

ACKNOWLEDGMENT

Virtual backgrounds in the result section were provided by Politecnico di Milano and Christian-Albrechts-University Kiel.

REFERENCES

[1] O. Grau and G. A. Thomas, "Use of image-based 3d modeling techniques in broadcast applications," in *Proc. Tyrrhenian Int. Workshop Digital Communications*, Capri, Italy, Sept. 2002, pp. 177–183.

[2] W. Matusik, C. Buehler, R. Raskar, S. J. Gortler, and L. McMillan, "Image-based visual hulls," in *Proc. ACM SIGGRAPH*, K. Akeley, Ed., 2000, pp. 369–374.

[3] G. Iddan and G. Yahav, "3d imaging in the studio (and elsewhere….)," in *Proc. SPIE Conf. Videometrics and Optical Methods for 3D Shape Measurement*, vol. 4298, Jan. 2001, pp. 48–55.

[4] R. Szeliski, "Stereo algorithms and representations for image-based rendering," in *Proc. British Machine Vision Conf.*, vol. 2, 1999, pp. 314–328.

[5] L. Falkenhagen, "Depth estimation from stereoscopic image pairs assuming piecewise continuous surfaces," *Image Processing for Broadcast and Video Production*, pp. 115–127, Nov. 1994.

[6] M. Ziegler *et al.*, "Evolution of stereoscopic and three-dimensional video," *Signal Processing: Image Commun.*, vol. 14, pp. 173–194, 1998.

[7] J.-R. Ohm *et al.*, "A realtime hardware system for stereoscopic videoconferencing with viewpoint adaptation," *Signal Processing: Image Commun.*, vol. 14, pp. 173–363, 1998.

[8] P. Rander, P. Narayanan, and T. Kanade, "Virtualized reality: Constructing time-varying virtual worlds from real events," in *Proc. IEEE Visualization*, Oct. 1997, pp. 277–283.

[9] K. Cheung, S. Baker, and T. Kanade, "Shape-from-silhouette of articulated objects and its use for human body kinematics estimation and motion capture," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, June 2003, pp. 77–84.

[10] M. Potmesil, "Generating octree models of 3D objects from their silhouettes in a sequence of images," *Comput. Vis., Graph. Image Processing*, vol. 40, pp. 1–29, 1987.

[11] R. Szeliski, "Rapid octree construction from image sequences," *CVGIP: Image Understanding*, vol. 58, no. 1, pp. 23–32, July 1993.

[12] W. Niem, "Robust and fast modeling of 3d natural objects from multiple views," in *Proc. SPIE Image and Video Processing II*, vol. 2182, San Jose, CA, Feb. 1994, pp. 388–397.

[13] Tzidon *et al.*, "Prompting Guide for Chroma Keying," U.S. Patent 5 886 747, Mar. 23, 1996.

[14] C. Cruz-Neira, D. Sandin, T. DeFanti, R. Kenyon, and J. Hart, "The cave: Audio visual experience automatic virtual environment," *Commun. ACM*, vol. 35, no. 6, pp. 65–72, June 1992.

[15] R. Raskar, G. Welch, M. Cutts, A. Lake, L. Stesin, and H. Fuchs, "The office of the future: A unified approach to image-based modeling and spatially immersive displays," in *Comput. Graph.*, 1998, vol. 32, Annual Conference Series, pp. 179–188.

[16] M. Gross, S. Wuermlin, M. Naef, E. Lamboray, C. Spagno, A. Kunz, E. Koller-Meier, T. Svoboda, L. V. Gool, S. Lang, K. Strehlke, A. V. Moere, and O. Staadt, "Blue-c: A spatially immersive display and 3d video portal for telepresence," in *Proc. ACM SIGGRAPH*, San Diego, CA, July 2003, pp. 819–827.

[17] G. A. Thomas, J. Jin, T. Niblett, and Urquhart, "A versatile camera position measurement system for virtual reality tv production," in *Proc. Conf. Int. Broadcasting Convention*, Sept. 1997, pp. 284–289.

[18] Y. Yakimovsky and R. Cunningham, "A system for extracting three-dimensional measurements from a stereo pair of tv cameras," *Comput. Graph. Image Processing*, vol. 7, pp. 195–210, 1978.

[19] R. Tsai, "A versatile camera calibration technique for high-accuracy 3d machine vision metrology using off-the-shelf tv cameras and lenses," *IEEE J. Robot. Automat.*, vol. 3, pp. 323–344, Aug. 1987.

[20] C. R. Dyer, "Volumetric scene reconstruction from multiple views," in *Foundations of Image Understanding*, L. S. Davis, Ed. Boston, MA: Kluwer, 2001, pp. 469–489.

[21] W. Martin and J. K. Aggarwal, "Volumetric descriptions of objects from multiple views," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 5, pp. 150–158, Mar. 1983.

[22] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3D surface construction algorithm," in *Proc. 14th Annu. Conf. Computer Graphics and Interactive Techniques*, 1987, pp. 163–169.

[23] A. Watt, *3D Computer Graphics*. Reading, MA: Addison-Wesley, 1999.

Oliver Grau received the Diploma degree in electrical engineering and the Ph.D. degree from the University of Hanover, Hanover, Germany, in 1991 and 1999, respectively.

From 1991 to 2000, he worked as a Research Scientist with the University of Hanover. He has been involved in several industrial, national, and international projects in the field of industrial image processing and three-dimensional scene reconstruction for computer graphics applications. In April 2000, he joined the BBC Research & Development Department, Surrey, U.K., as a Senior Research Engineer in the virtual production team. His main interest is in visual engineering, i.e., the application of image processing, computer vision and computer graphics for the production of visual media.

Tim Pullen received the M.Sc. degree in mathematics and physics from the University of Bristol, Bristol, U.K., in 2000.

In September 2000, he joined the BBC Research & Development Department, Surrey, U.K., and has worked on the reception of digital terrestrial television and on virtual production. He is currently involved in the development of an electronic program guide data management system.

Graham A. Thomas received the honors degree in physics from the University of Oxford, Oxford, U.K., in 1983 and the Ph.D. degree with the University of Essex, Essex, U.K., in 1990 for work on motion estimation.

He joined the BBC Research & Development (R&D) Department, Surrey, U.K., in September 1983. He has worked mainly on projects concerned with image processing, including standards conversion, novel methods for PAL coding, and video compression. Most of his recent work has been concerned with techniques for virtual studio production, including the development of the award-winning Free-d camera tracking system. He recently led the PROMETHEUS project, a U.K. DTI-funded project with eight partners, developing an end-to-end object-based 3DTV system, and is currently leading a team of engineers at BBC R&D developing technology related to virtual production. He is the coinventor of about 20 patents and the author of many published papers.