

A Virtual Studio for Live Broadcasting: The Mona Lisa Project

Laurent Blondé, Matthias Buck, Ricardo Galli,
Wolfgang Niem, Yakup Paker, Wolfgang Schmidt,
and Graham Thomas
The Mona Lisa Project

The Virtual Studio concept replaces real background sets with computer-generated synthetic scenes. Image sequences combine real foreground action, shot in the studio, with computer-generated backgrounds. The Mona Lisa project seeks to develop and integrate technologies (algorithms, software, and hardware) needed to construct, handle, and render 3D models in real time.

Recent advances in computer hardware technology, coupled with computer visualization and image generation research, are significantly affecting TV production methods, which have remained unchanged since the introduction of color television. Over the past decade, technologies used in the broadcasting industry have changed greatly, particularly in the recording and postproduction areas. The equipment now used is smaller, lighter, portable, and more flexible. With the introduction of digital technology, the potential for manipulating recorded images becomes virtually limitless.

Traditionally, after a production is planned and designed, the required scenery must be costed and full working drawings prepared for each set. These sets are then constructed, delivered to the studios, and erected; after the broadcast, they must be dismantled, transported, and either stored for further use or broken up. This cycle is quite often the most expensive aspect of a television production, and the time needed dictates both the production timetable and the studio's overall output.

The introduction of virtual studios brings this

archaic process in line with the rest of the program-making process. The virtual studio system will be an important tool for production houses, providing many creative opportunities as well as cost reductions. The flexibility offered by switching easily from one set to another eliminates the heavy cost and time constraints of building, storing, transporting, and re-setting current productions. Several programs can share studio space and equipment, and rapid turnaround times will lead to increased studio output.

Genesis of the project

In 1990, we believed that progress in computer technology and image processing know-how would soon enable real-time processing of video image sequences, thus bringing computers into studios in earnest. Convinced of the timeliness of the virtual studio concept, we proposed the Mona Lisa project. This collaborative project undertook research and development in all the key aspects of virtual studio operation and attempted to build a demonstrator.

At the time the project started, we were aware of some work on certain aspects of virtual production. For example, the Japanese broadcast company NHK was working on a 2D virtual studio system called Synthevision.¹ From a high-definition stored image, this system extracted a window whose position and scale could be changed using signals from mechanical sensors on a camera. This allowed a background image to be generated matching the camera's viewing angle and zoom. At France's Institut National de Audiovisuel (INA), the Synthetic TV project² focused on off-line 3D virtual sets. In recent years, several broadcasters have carried out experiments reported in the press, and some are using virtual studio techniques in daily production. More recently, a number of companies have started to offer virtual studio products; NHK, for example, now has a 3D virtual studio system.³

Mona Lisa (MOdeling NATural Images for Synthesis and Animation) is a collaborative project undertaken by nine European industrial, broadcasting, and academic institutions and supported by the European Union RACE II program. These include the BBC (UK), Daimler-Benz (Germany), Digitale Videosysteme (Germany), University of Hannover (Germany), Queen Mary and Westfield College (UK), Siemens (Germany), Thomson Broadcast Systems (France), Video Art Production (Germany), and the University of the Balearic Islands (Spain).

The key problem of virtual studios: combining real and synthetic images

Whenever electronically generated graphics (anything from a weather map to a complete computer-generated virtual background) are inserted into images from a real TV camera, it is vital that their relative positions be correct. Conventionally, the positions of the camera and the inserted graphics are adjusted until the desired alignment is achieved; then the camera position and orientation are fixed. The requirement that the camera be stationary places significant constraints on the use of these techniques. For example, it is not generally possible to have even a small degree of camera motion to follow the movements of a presenter's head since any picture material inserted electronically would not appear to move concurrently. When the whole set is generated electronically, however, camera motion becomes an important means of maintaining the illusion of reality.

A virtual studio system designed for TV production must be able to place actors, performing in front of a blue background for separation, into any synthetic scene, including those where set components might partially or fully cover actors. This requires accurately measuring real camera parameters (position, orientation, and zoom) and, based on this information, rendering the corresponding view of the synthetic image, which will be mixed into the final image. With these constraints in mind, we have identified two distinct stages for virtual studio operation: virtual set creation (preproduction) and virtual studio production (live shooting), illustrated in Figure 1.

Creating the virtual background scene could involve a number of modules, all off-line and possibly distributed over networks, such as

- archives of 3D models, textures, and trajectories;
- automatic 3D modelers and dynamics modeling;
- interactive 3D CAD modelers and 3D design software;
- texture creation tools and painting and image editing systems; and
- scene generation tools that combine geometry and textures, illumination, and object animation to generate a scene definition compatible with the real-time renderers.

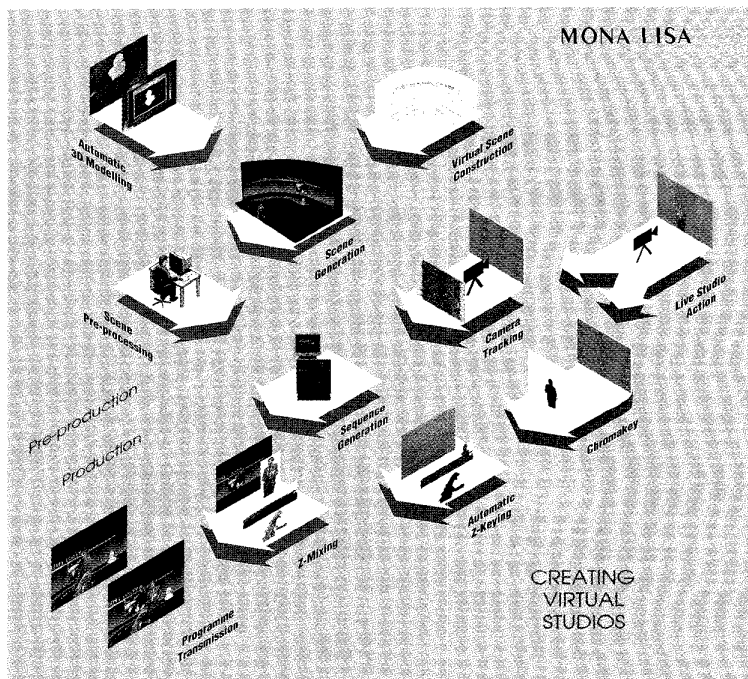


Figure 1. Virtual studio operations.

The virtual studio production itself combines various real-time components for generating video sequences. These include

- cameras shooting the foreground action against a blue background,
- a camera tracking module to provide position information,
- a rendering module to generate images with the correct perspective in the virtual scene, and
- a z-mixing module, combining the foreground and background signals and possibly an insert video, to generate the program to broadcast or store on tape.

Creating virtual sets

Virtual studio sets, like any VR environment, require 3D models that describe the virtual scene and objects in it. A variety of tools (modelers) are available for this purpose, mainly based on CAD methods. These provide accurate descriptions of geometry, yet sometimes are too complex for real-time applications. Building a model from a set design is generally a complicated process, requiring such operations as texture mapping to create realistic effects. Various objects like chairs and

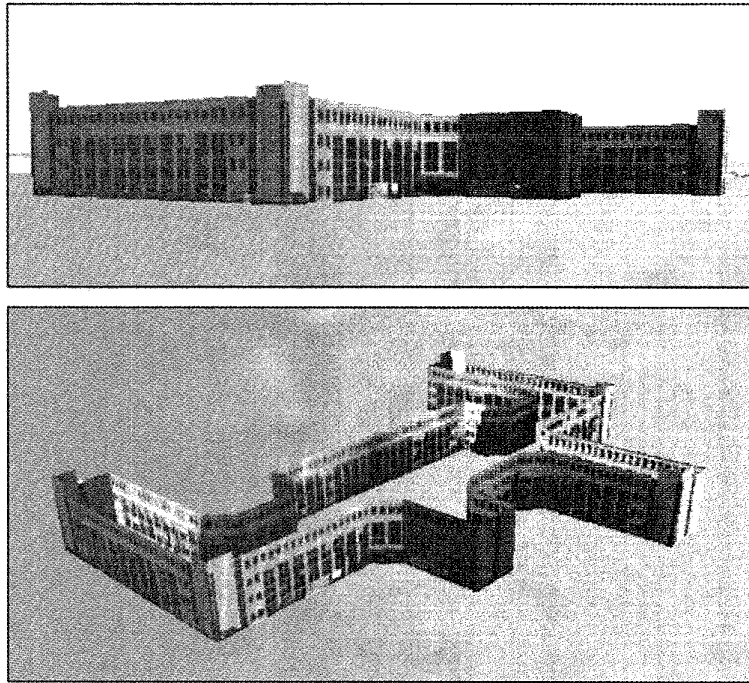


Figure 2. Two views of a model of the Daimler-Benz research center, Ulm, Germany.

tables are often needed in the set, but designing a CAD model for each item could be an elaborate (and expensive) process.

Active methods such as laser scanners and structured light sources, often requiring expertise and expensive hardware, are nonetheless useful for capturing good 3D data for a restricted range of object shapes of limited size (about one meter). To complement existing tools, we concentrated on developing methods that rely on image analysis to automatically generate 3D models from 2D image sequences. We believe that these methods, which use standard cameras, lead to compact model geometries with good texture information.

Modeling buildings

Our approach⁴ requires that the images be perspective projections of real objects and that these objects have straight lines and flat surface elements. In perspective projections, parallel lines in space are projected onto lines that intersect at a common "vanishing point." The location of such a vanishing point is directly correlated with the spatial orientation of the corresponding set of parallel lines. Taking these relations into account, we can proceed as follows:

- *Camera calibration.* To analyze the distortion that is due to the perspective projection, we

derive certain camera parameters (such as focal length and tilt angle) directly from the photographs, provided at least one object has certain known geometric properties.

- *Analysis of the perspective projection.* After we have identified and analyzed the line structures, we determine the vanishing points. From the location of each vanishing point, we derive the spatial orientation of the corresponding set of parallel lines, from which comes the 3D orientation of the object's related surface element. This requires that each planar surface element contain sufficient parallel line structures (such as rows of windows on a building's facade).

- *Extraction of texture information.* Texture maps are crucial for simple yet close-to-reality 3D models that contain flat structures (such as windows or doors of a building). We extract the texture data from the images used for perspective analysis and correct them for perspective distortion to obtain orthogonal texture maps that are attached to the object like wallpaper.

- *Modeling of complex buildings.* From a single photograph, we reconstruct a 3D model of the visible parts of the object. To obtain a complete model, depending on its complexity, we need further views. Finally, all these partial 3D models are integrated into one combined model. For a simple object, about four views are sufficient. The model shown in Figure 2, a more complex object, was reconstructed from 20 images. We used standard cameras and transferred the resulting images in digital form onto a Kodak PhotoCD. The analysis of one such photograph takes just a few minutes. The whole procedure for a complex object like the one shown in Figure 2 takes about two hours, determined mainly by the user's expertise and level of interaction.

Modeling isolated objects

We developed a method to automatically reconstruct the shape and texture of single 3D objects using multiple calibrated camera images. A stationary camera and a turntable with the modeling object placed in the middle constitutes the measurement environment (as shown in Figure 3a). To facilitate object segmentation, the background is of uniform color. Dedicated lighting provides diffuse illumination. The turntable rotates incrementally by a predefined angle, under

computer control, with an image taken each time until the object is completely registered. Three main steps comprise this process.

First, we have to calibrate the system to obtain position, orientation, and focal length of the camera. A test pattern consisting of black circles on a white background is placed on the turntable. Then we use shape estimation to derive a volumetric representation of the object, extracting silhouettes from multiple calibrated views using a color segmentation technique, as is done in TV studios,⁵ and then integrating them. To handle object concavities, we treat each two successive images as a stereo image pair, obtaining the necessary range information without additional equipment such as a laser range finder. Finally, we transform the volume representation to a surface model, using a mesh-growing algorithm that yields smaller triangles at surface regions with higher curvature and larger triangles in surface regions with low curvature. Texture estimation is applied to each surface triangle using the input images, which provide natural-looking 3D models.

This modeling technique was applied to a number of isolated 3D objects on a turntable rotated by 36 equal steps. For each view, we shot a 720×576 image with a stationary calibrated CCD camera. Figure 3b shows a synthesized projection of an automatically generated model of an antelope, represented by a wire frame model and the related texture maps.

Modeling interior scenes

We focused on constructing 3D shape models of interiors by integrating multiple views acquired using a single moving camera.⁶ We placed limited constraints on the acquisition process, assuming that the scene is static and using a calibration pattern at the beginning.

This method exploits the apparent motion in the scene generated by the camera's motion. The deformation resulting from the 3D to 2D projection on the camera focal plane is exploited to derive, first, an optical flow map, and then a depth map for each selected input image. The depth maps are subsequently integrated over time to build a surface 3D model of the scene.

Modeling deformable surface objects

To enhance the realism of the virtual sets, we employed a new class of physically based deformable surface objects. These soft objects, defined as coupled particle systems,⁷ exhibit a dynamic response to external actions such as force

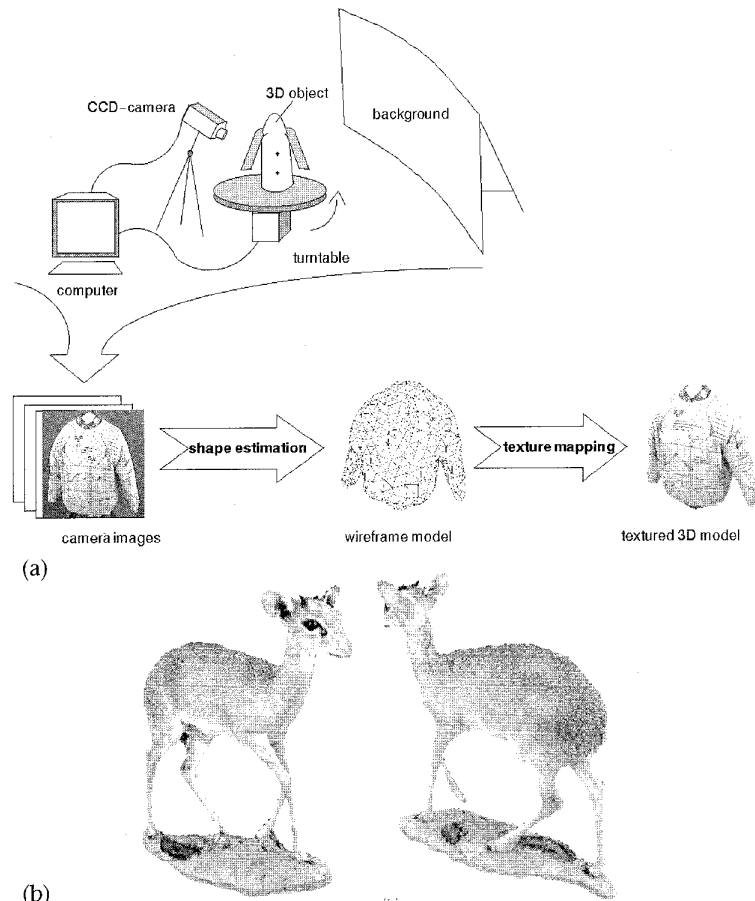


Figure 3. (a) Setup used for modeling isolated objects, plus (b) an example of a model.

fields. This behavior is achieved by simulating the motion dynamics of each particle belonging to the closely coupled set associated with the surface. When modeling such objects, we can also use computer animation to realistically visualize their motion evolution through time. This lets us simulate the behavior of any object affected by wind or gravity, such as curtains, flags, tablecloths, clothes, and water surfaces.

Scene generation modules

We have written special 3D scene generation software that allows objects defined by various modeling techniques to be assembled into a virtual set (see Figure 4, next page). Because of the time pressures in TV productions, we aimed to provide designers with a "what you see is what you get" (WYSIWYG) system that lets them modify complex scenes interactively and render the results immediately on SGI Onyx workstations. Our system consists of two well-defined modules with user interfaces programmed in X11/Motif

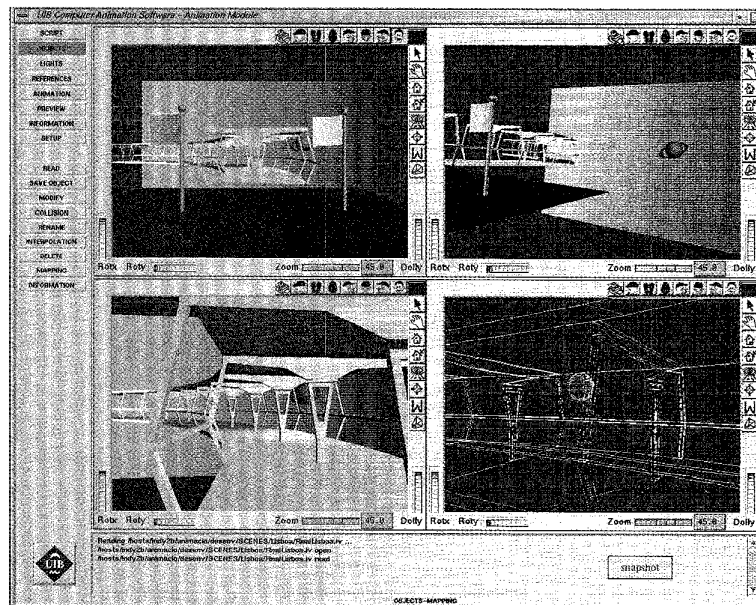


Figure 4. Scene construction.

and OpenInventor. It is also VRML compatible.

The *Scene Construction* module lets users input 3D object models in different formats to construct the set. The system can read objects made using image analysis techniques developed in the project as well as those from standard commercial modelers. Many different types of lights from diverse sources can be used to illuminate a scene. The user can also modify and interactively view linear transformations such as rotation, scaling, and translation, and object parameters such as lighting and shading. Other parameters that can be modified include material information like color (ambient, diffuse, specular, emissive), transparency and specular shininess, and textures.

The set designer may have to align a large number of objects to their desired positions in the 3D environment, often a tedious process. To simplify this, we have provided a collision detection facility to avoid object penetration yet keep objects as close as possible.⁸

The *Animation* module lets users interactively define the animation parameters of moving objects. Our aim was to give users full freedom to animate any visible parameters of the virtual objects in the scene, not just motion but also object and lighting properties such as deformation, material, transparency, textures, color, and so forth.

We have developed a special interface for modeling and editing deformable objects using the techniques described in this section. This interface lets users assign the physical parameters and select

the supporting vertices and faces of the object. To support real-time object rendering, the simulation can be run once and the result (that is, the object geometry for each frame) can be stored in memory or in a file, or both.

For more complex animations, we designed a hierarchical system in the form of a tree whose nodes are references or basic geometric objects. A reference can join different objects and other references to define more complex objects. The system provides a tool to create and modify the hierarchy. This module can also read the real camera parameters. These parameters are computed by the camera tracking module running on the parallel accelerator or from stored values in a file, then transmitted over the network in real time. Because these values affect the perspective transformations of a selected window, the user can visualize the animations in a real shot simulation.

Scene preprocessing

Virtual set images must be rendered in real time with television-image quality. Therefore, we had to address the two conflicting goals of real time and quality. To do so, we defined several steps necessary to prepare the synthetic set model for real-time rendering at production. Preprocessing the illumination and using the simplest and fastest interpolation mode available on the graphics engines increases the real-time performance and the quality of the rendering system. The algorithm design comprises several tasks.

Visibility determination corresponds to the geometrical aspect of the illumination propagation within the 3D scene.⁹ This technique lets us compute or estimate the visible part of a patch as seen from a given point and as possibly occluded by other scene elements. We developed and compared two methods. An analytical method works in object space and shows the exact visible parts of a given surface element. A numerical method uses a panel of pixels to determine the digital visible part of any surface element in the scene.

The *3D sampling* task selects optimal pivot points to store the illumination information. This decreases the number of polygons to be projected and interpolated in real time while preserving image quality. Considering this a problem of minimization, we developed a new method for selecting these points. We defined the minimization criteria in relation to the capacities of the final shading interpolation. Then we selected two sampling criteria: 3D sampling according to the light-source energy repatriation (illumination slope

sampling) and 3D sampling according to occlusions (shadow cutout). The latter uses results of the analytical visibility determination algorithm.

In a scene, energy exchanges between patches depend—as defined by the radiosity method—on the geometrical configuration of the emitting and receiving elements and on occlusions that might occur. We use form factors or configuration factors to evaluate this geometrical visibility. Measuring memory costs, processing speeds, and quality of results, we compared the analytical and numerical methods. When a surface element has a high influence—that is, it illuminates the whole scene—the analytical method is preferable; otherwise, the numerical method works with a generally adequate medium accuracy.

We have developed a high-quality *illumination model* that includes directional effects and shadows, which are applied at given 3D points in the scene. This technique models directional effects for either light emission or reflection on object surfaces based on a generic mathematical representation, Bidirectional Reflection Density Functions using Spherical Harmonics.¹⁰

Virtual studio production (real time)

This stage takes place in a real studio and shoots foreground action against a blue background for chromakeying. The modules used must operate at real-time video rates.

Camera tracking

It is possible to attach mechanical sensors to the camera to measure parameters such as pan, tilt, and zoom and to use these measurements to control the position of the virtual background. However, this solution is not always practical. To develop a more efficient alternative, we created a motion estimation algorithm that can measure global image motion to a high degree of accuracy. This lets us move the camera and track its motion without mechanical sensors. We chose to limit camera motions to pan, tilt, and zoom, significantly simplifying the calculations while satisfying the requirements of program productions such as news, current affairs, and simple drama, where camera positions tend to be largely fixed.

The algorithm is designed to be used with a keying unit and a blue background. The background cannot be a uniform shade of blue but must contain enough information to ensure reliable motion estimation. We have found that using a nonuniform blue background does not cause significant problems for key generation

when using a simple keyer that does not attempt to extract shadows. If shadows are to be extracted, however, choose blue shades with care to ensure that each shade has the same luminance level.

The key signal is used to indicate the areas of the image that correspond to background so that the estimation process can ignore foreground areas (such as actors) that may move independently of the camera. This approach will fail if the visible background area drops below 25 percent of the total image area, in which case it might be necessary to keep the camera stationary.

We chose an algorithm¹¹ that measures translation and scale change by computing the spatial and temporal luminance gradients at many points in successive images. A set of over-determined linear simultaneous equations is formed (one equation per point) relating the measured gradients to the motion parameters. The least-squares solution of these equations yields motion parameter estimates. We adapted this algorithm to reduce the computational requirement and to prevent any significant accumulation of motion estimation error.¹² One advantage of this kind of algorithm is that it works best on unfocused signals, which means that some background blurring due to depth-of-field effects will not upset its operation.

We also considered it important to modify the algorithm to measure image motion relative to a given reference image rather than between consecutive images. This prevents the accumulation of motion estimation errors that occurs when successive measurements are integrated. For the first reference image, we use one viewed by a camera placed in a known position. Only when the degree of overlap between the present image and the stored reference image becomes small do we adopt the present image as the new reference for future measurements. Discarded reference images are held in memory to be reused if the camera returns to an earlier orientation. This further reduces the accumulation of estimation error that could occur each time a new reference image is adopted.

We developed a simple algorithm for choosing which stored reference image to use, given the current camera orientation. Figure 5 shows a block diagram of the camera motion estimation system.

The main algorithm parameter that controls

We measure image motion relative to a given reference image rather than between consecutive images. This prevents motion estimation error accumulation.

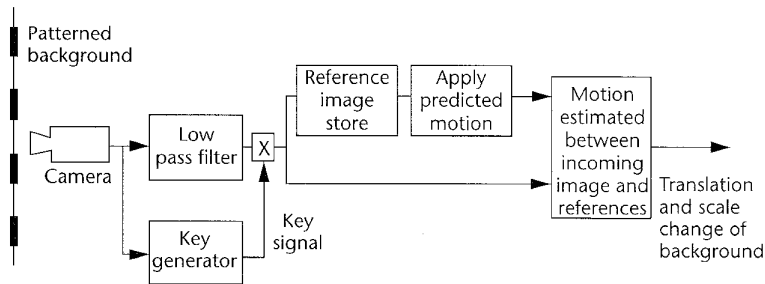


Figure 5. Block diagram of the camera motion estimation system.

the trade-off between accuracy and processing power is the number of image points (and therefore equations) used. Benchmarking showed us that we could make accurate measurements on a range of picture material with as few as 900 measurement points, arranged as a rectangular grid of 30×30 points. Tests showed that a workstation based on an R3000 processor could perform this computation at a rate of approximately 50 images per second (excluding any overheads for image capture). Run with 35×35 points on a single processor of the hardware platform we designed, the algorithm maintained the rate of 25 Hz, including image capture, for real-time operation. Low-pass filtering of the signal before measurement (which improves accuracy) is best carried out on dedicated hardware and was not included in the benchmarking. You could also implement the horizontal component of the filter conveniently using analogue circuitry before digitizing the image signal.

Virtual camera synchronization

To correctly place the actor within the 3D virtual scene, the virtual camera must have the same perspective as the real camera. The initial parameters of the virtual camera are set to correspond to the initial position and zoom of the real camera. The camera tracking system tracks the changes in the pan, tilt, and zoom of the real camera and sends them to the animation control module. The pan and tilt are measured in pixels and then converted to rotation angles, while the zoom is measured by ratios. We have to take into account that the zoom ratio would directly affect the angles of the virtual camera.

It is also necessary to delay the video signal from the camera to compensate for the processing delay in the camera tracking system and the real-time rendering. We achieved this using field delays in the hardware platform. Corresponding delays must also be inserted in the audio signal.

There are ways of avoiding the need for such delays, but each has significant drawbacks. You can use predictive filtering techniques to estimate what the camera parameters will be several fields in advance, but this is bound to introduce inaccuracies and thus loss of registration between real and virtual worlds. Alternatively, you can use robotic camera mounts that provide position data prior to their actual movement, although such mounts are awkward to drive interactively and are disliked by many cameramen.

Sequence generation

This phase renders in real time, following actual camera behavior, the previously computed values of the scene preprocessing module. We use the polygon projection feature of graphic workstation graphics pipelines and take advantage of the hardwired Gouraud (bilinear) shading interpolation between the pivot points previously selected.

Hardware platform

Our development took place on a Silicon Graphics (SGI) Onyx system equipped with a Sirius video board (illustrated in Figure 6). For sequence generation to achieve the necessary real-time speed, we implemented multiprocessing software, splitting the application user interaction, camera parameter management, polygon projection and texturing, depth, and video output among several processing engines. We encountered some problems in trying to meet the real-time requirements. The necessity of sharing platform resources, for instance, imposes constraints on the programming mode and requires managing the inter-process data exchanges with care.

To address the requirements of virtual studio production, other than sequence generation, we designed and built an innovative hardware platform integrating a number of standard and specially built components. This platform's main features include camera tracking, mixing of virtual sets with live video foreground, and insertion of additional video signals (for example, to simulate a "video wall") into a virtual scene. The system is built around a commercially-available intelligent frame buffer (ISP500, built by DVS¹³) that provides all the necessary interfaces to studio video equipment and computer networks.

z-mixing

The chromakey is a well-known television production technique for inserting foreground action, shot in another location, into a selected back-

ground scene. It was used first for filming people in difficult or impossible locations and later for reducing costs. Chromakey is also the starting point for mixing in a virtual studio. It must, however, be able to handle synthetic set components that partially or fully cover actors in the foreground as needed. To achieve this, we developed a new mixing technique that we call z-mixing.

Unlike the usual 2D image layering technique, z-mixing is based on 3D images. In this context, a 3D image is a computer-generated 3D scene description with additional information about the distance between the virtual camera and each pixel of the generated image, called depth or z-value. By comparing the corresponding z-values for the two input images to be mixed, we can decide which pixel to use for the final image.

A hardware z-buffer for real-time 3D rendering is a standard feature on high-end graphics workstations. Therefore, z-values for each pixel of a computer-generated image are normally available. For the camera-shot foreground live action, however, no depth information is available. We thus adopted a simple approach to estimate, or pre-set, the distance between the actor and the camera and to use this value as a constant depth value for the whole foreground image. If the actor's movements clash with the virtual set, this value has to be updated; otherwise, no change is needed.

Separation of the foreground image is achieved by a standard chromakey generator that provides the key signal needed to distinguish between the actor and the blue background. Our z-mixing technique¹⁴ uses what we call a z-key generator to evaluate the chromakey signal as well as the depth values for both inputs, generating a final key signal usable on a normal mixing unit.

We designed, manufactured, and tested a z-mixer called Vmix (Video Mixer), which consists of a baseboard plugged into any 9U VMEbus (such as SGI Onyx), and several I/O and option modules. It incorporates all standard mixer functions including cut, dissolve, fade, key, and downstream key. Additional option slots can be used to expand the device by a chromakeyer, a wipe generator, and a text buffer. The Vmix device is not dedicated to any specific video raster and can process either RGB or YUV data with sampling frequencies of up to 80 MHz per channel and data width of eight or 10 bits. In its present demonstrator form, the Vmix baseboard is equipped with I/O modules for digital video data according to current standards. Figure 6 shows the integration of the Vmix device into the demonstrator of the Mona Lisa project.

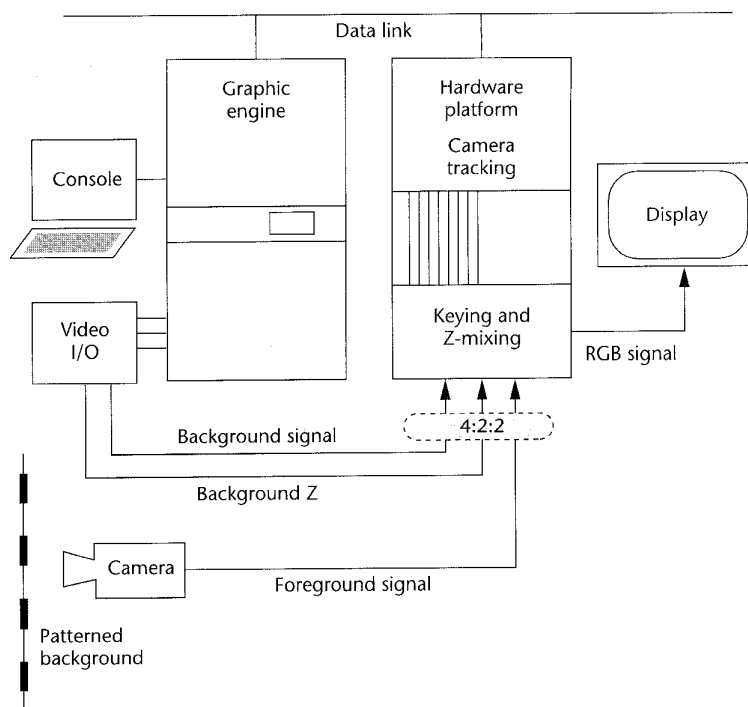


Figure 6. Mona Lisa system demonstrator.

Overlay mixing

Producers often need to incorporate real-time video into a virtual scene to reproduce windows, picture monitors, or "video walls" containing graphics, interviews with reporters, and so on. To achieve this, the inserted video signal must be manipulated to the correct size, position, and perspective within the virtual scene, corresponding to camera movements. This is a typical task for a DVE (digital video effects) unit. For inserting video seamlessly, we adopted an extension of the z-mixing device, a method that renders a panel of a special color into the virtual background image. Using a color-key device to detect the panel, it then drives a mixer to replace the panel with the transformed live video image.

Camera-driven DVE

We designed and built a dedicated DVE unit that uses high-quality interpolation techniques but is limited to providing the range of image transformations required for incorporating live video into a virtual scene. The transformations required to manipulate an image according to camera position include

- horizontal and vertical displacement (pan and tilt),

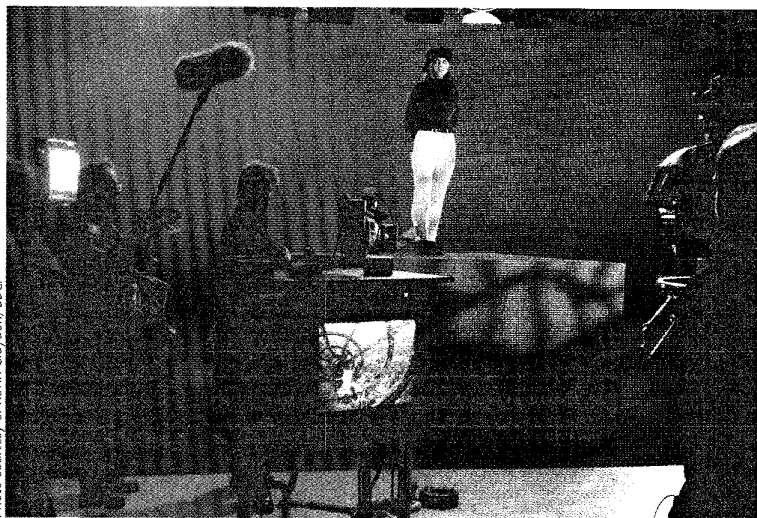


Figure 7. A virtual studio production. The camera-driven DVE is used to produce a BBC program, "The Web." The composite picture can be seen on the monitor in the foreground.

- enlargement and reduction in size (zoom), and
- perspective transformations (such as viewing the image obliquely).

The DVE unit can perform these transformations on real-time video and change the values of each parameter in every video field. It can also achieve a small degree of image rotation should it be required. Note that the perspective transformation is not just a "keystone" distortion but will correctly reproduce the change of scale corresponding to some parts of the image being "further away" from the camera than others. For stationary images, the unit can also grab a single frame, avoiding the need for a separate still store. Our DVE unit measures 300 × 340 mm and occupies a single circuit board. Video input and output interfaces use the serial digital standard at 270 Mbits.

The camera-driven DVE was used in the production of several BBC programs, including "The Web" and a 10-part BBC program called "This Multimedia Business" (see Figure 7). Several DVE units were connected to camera heads fitted with pan/tilt/zoom sensors. This setup allows a pre-rendered background image to be moved to follow the camera motion. A matching key signal can also be manipulated in a companion DVE unit to allow certain elements of the virtual set to obscure actors, providing a simple form of depth keying.

Parallel accelerator

We incorporated a DSP-based multiprocessor accelerator into the hardware platform with direct video channel access to the frame buffer.¹⁵ This

lets us parallel-process video streams fed directly through the frame buffer. The parallel accelerator is hosted by a Unix graphics workstation; this permits network access to the accelerator as a shared resource. The parallel accelerator hardware (illustrated in Figure 8) includes a processor pool of Motorola DSP96002-based boards sitting on a VMEbus backplane. Each processor board has two DSP processors that operate at 40 MHz with a peak performance of 60 MFlops each. For frame buffer access, the processor pool is organized as a set of clusters; each cluster shares an interface board connected to one of the I/O processors. There are currently two such clusters in our system, each with four processors.

The intelligent frame-buffer subsystem consists of an address generator, a set of RAM memory boards, and several FIFO-buffered I/O processor boards that share a high-speed synchronous backplane bus. A Motorola 68040-based single-board processor running the real-time operating system OS9 controls all operations within the frame buffer subsystem as well as Ethernet communications to host computers. A/D and D/A converters connected to the I/O processors enable simultaneous recording and video signal display. The frame buffer subsystem can handle several independent data streams (in or out, or both) in real time up to a total data rate of 400 Mbytes per second.

We have found this system to be capable of running the camera tracking algorithm for the Mona Lisa demonstration in real time for 35 × 35 points, including frame loading time. Benchmarks have indicated that 40 × 40 points is the upper limit for real-time operation (40 ms per frame); a larger number of points requires parallelism.

We have developed sophisticated systems software for this machine based on the SAPS (Self-Adapting Parallel Servers) model. This model specifically addresses the issue of accelerating preselected and computationally heavy procedures in a target application area by building a "server box" (a pool of processors). Such a server box, transparent to its users, can accelerate a library of procedures as requested from networked workstations.

Server support has two benefits. First, each server carries out a well-defined computational task on a set of input data (similar to a library call within an application). Second, since this task is achieved in a parallel fashion, it is accelerated. The application programmer's involvement is limited to distributing the individual application requirements among the available servers.

We developed two graphical user interfaces

with the widely used X-Windows and OSF/Motif systems: the Administration Tool and the Application Tool (or Command Tool). The first interface initializes the accelerator, starting system management servers such as the Pool Manager and the Object Manager under OS9. The second interface supports building parallel applications and managing related data streams.

System integration and demonstration

Integrating the system components described above ran along two main tracks: coordinating the virtual set creation tools (off-line modules) and integrating the virtual studio production system (real time). These converged in a demonstration system exhibited at two events: the International Broadcasting Convention in September 1994 in Amsterdam, and the November 1994 Mona Lisa Workshop in Hamburg.

As Figure 1 showed, the virtual set creation tools consist of software modules for 3D reconstruction and modeling. These provide data that can be combined and manipulated using the scene generation module. The resulting scene is then transferred to the scene preprocessing module. Initially the SGI Inventor, then the OpenInventor 3D, formats were used for data transfer between modules, which proved extremely useful.

Building the virtual studio production system, in particular addressing all its real-time constraints, proved to be a major task. Four types of equipment had to be integrated: classical studio equipment (such as cameras, studio interfaces, and displays), the hardware platform (for accelerating the camera tracking process and the Vmix system), and the Onyx graphics engine with the Sirius video board. The Mona Lisa project has been one of the first to realize the integration of real-time broadcast equipment, custom-developed hardware, and high-end graphics processing to demonstrate a virtual studio. The main difficulties encountered involved interconnecting various components, synchronizing software modules in real time (using multiprocessing), and establishing a system calibration procedure.

The demonstrations included using the camera tracking algorithm for real-time tracking of the pan-tilt-zoom parameters with the background scene following the camera movements. The z-mixing worked particularly well, for the first time producing very convincing images of people moving in a virtual set with virtual objects like a desk and a dinosaur. Thus, as shown in Figure 9, a person behind the desk could walk around it to the

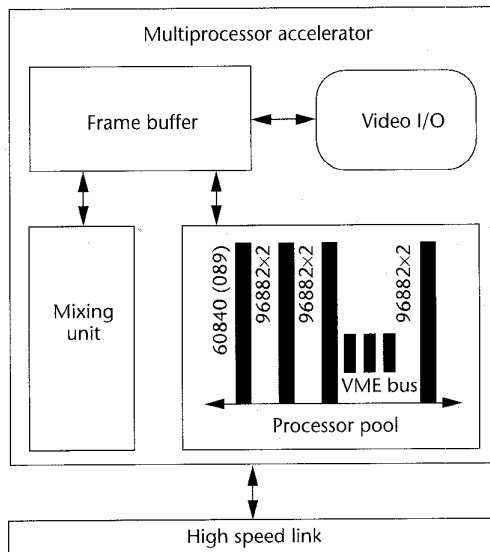


Figure 8. Parallel accelerator hardware for the Mona Lisa system.



Figure 9. Virtual studio mixed image.

front while remaining behind the dinosaur, then walk around the dinosaur to the front and then finally return to the original position behind the table. To make such movement possible, the z-mixing provides images consistent with the view, including occlusions, of the virtual scene. This real-time 3D mixing, a world premiere, was a hit at both of the shows mentioned above.

Conclusions

The Mona Lisa project has clearly demonstrated the potential of the virtual studio as a pioneering production tool. Virtual studios offer greater creative freedom and expression than conven-

tional scenic methods, with the added benefit of reducing the costs and time scales such methods impose. These are, however, still early days in the progress of virtual studios, and there are some areas that need to be further developed to make this a truly workable technique.

Some problems for further research include an elegant solution to the problem of full camera motion—including track and crab—that will operate in normal studio conditions. Researchers are currently working on mechanical methods such as remotely controlled platform-mounted cameras. Sequence generation remains a computing-intensive process, and achieving real-time video rates requires, at the moment, the power of graphics supercomputers. Finally, an alternative technique to chromakeying that does not involve the use of special blue box backgrounds would simplify the studio process even further and enhance the creative possibilities of virtual studios. **MM**

Acknowledgments

We are grateful for the support received for the project entitled "Modeling Natural Images for Synthesis and Animation—Mona Lisa (R2052)" from the European Union under the RACE II program. Many people, too many to name, were involved with this project, and we wish to acknowledge their contribution. However, Richard Kunicki from VAP deserves special mention as he provided the vision and belief in virtual studios from the outset of the project and inspired us all with his enthusiasm and energy to make it a reality. We wish also to acknowledge the contribution of Miguel Dias of Adeti, Lisbon, Portugal, for the deformable object modeling.

References

1. S. Shimoda, M. Hayashi, and Y. Kanatsugu, "New Chromakey Imaging Technique with Hi-Vision Background," *IEEE Trans. on Broadcasting*, Vol. 35, No. 4, Dec. 1989, pp. 357-361.
2. A. Fellous, "Synthetic TV," conference notes, *Imagina*, available from INA-Imagina Forum, 4 Av. de L'Europe, Bry-sur-Marne, 94366 France, 1991.
3. K. Haseba et al., "Real-time Compositing System of a Real Camera Image and a Computer Graphic Image," *IBC '94, IEE Conf. Publ. No. 397*, Institution of Electrical Engineers, London, 1994, pp. 565-660.
4. M. Buck, "Modeling Buildings from Single Images," *Proc. European Workshop on Combined Real and Synthetic Image Processing for Broadcast and Video Production*, Springer-Verlag, Berlin, 1995, pp. 197-206.
5. W. Niem and R. Buschmann, "Automatic Modeling of 3D Natural Objects from Multiple Views," *Proc. European Workshop on Combined Real and Synthetic Image Processing for Broadcast and Video Production*, Springer-Verlag, Berlin, 1995, pp. 181-193.
6. P. Robert and F. Ogor, "Joint Estimation of Depth Maps and Camera Motion in the Construction of 3D Models from a Mobile Camera," *Proc. European Workshop on Combined Real and Synthetic Image Processing for Broadcast and Video Production*, Springer-Verlag, Berlin, 1995, pp. 147-163.
7. D. Breen, D. House, and P. Getto, "A Physically-Based Particle Model for Woven Cloth," *ACM Siggraph Tutorial 16 Notes*, ACM Press, New York, 1992, 5-10, 5-35.
8. J. Jaume et al., "Real-Time Collision Checking for 3D Object Positioning in Sparse Environments," *Proc. European Workshop on Combined Real and Synthetic Image Processing for Broadcast and Video Production*, Springer-Verlag, Berlin, 1995, pp. 216-225.
9. F. Sillion, J. Arvo, and D. Greenberg, "A Global Illumination Solution for General Reflectance Distributions," *ACM Computer Graphics*, Vol. 25, No. 4, July 1991.
10. C. Muller, "Spherical Harmonics," *Lecture Notes in Mathematics*, Springer-Verlag, Hamburg, 1966.
11. M. Hoetter, "Differential Estimation of the Global Motion Parameters Zoom and Pan," *Signal Processing*, Vol. 16, No. 3, March 1989, pp. 249-265.
12. G.A. Thomas, "Global Motion Estimation for Registering Real and Synthetic Images," *Proc. European Workshop on Combined Real and Synthetic Image Processing for Broadcast and Video Production*, Springer-Verlag, Berlin, 1995, pp. 75-83.
13. *Image Storage and Processing System ISP500 User Manual*, DVS GmbH, Hannover, Germany, 1993.
14. W. Schmidt, "Real-time Mixing of Live Action and Synthetic Backgrounds Based on Depth Values," *Proc. European Workshop on Combined Real and Synthetic Image Processing for Broadcast and Video Production*, Springer-Verlag, Berlin, 1995, pp. 26-34.
15. A.V. Sahiner, P. LeFloch, and Y. Paker, "A Parallel Accelerator for Using Synthetic Images in TV and Video Production," *Proc. European Workshop on Combined Real and Synthetic Image Processing for Broadcast and Video Production*, Springer-Verlag, Berlin, 1995, pp. 13-25.



Laurent Blondé is currently a research engineer and project manager at the Thomson Broadcast Systems Research and Development center in Rennes, France. He received a Degree in engineering from École Supérieure d'Optique (IOTA/ESO, Paris). His main interests are image processing, image synthesis, and virtual reality, linked with TV production. He is the coordinator of the Mona Lisa project, responsible for both the technical organization and reporting progress and results to the European Commission.



Matthias Buck is currently a scientist at the Daimler-Benz research center in Ulm, Germany. He received a Diploma in electrical engineering from the University of Stuttgart, Germany. His research interests include model-based image data compression, reconstruction of 3D objects from 2D images, and computer vision for traffic applications and virtual reality.



Ricardo Galli holds a Diploma in computer engineering from the University of Santiago del Estero, Argentina. He also completed the Engineering Thesis at the Bariloche Atomic Center/Balseiro Institute (Rio Negro). He joined the University of Balearic Islands, Spain, as a lecturer at the Mathematics and Computer Science Department. Currently he is working on his PhD, researching computer-supported cooperative working and hypermedia representation and transmission.



Wolfgang Niem is a research assistant at the Institute of Communications Engineering and Information Theory at the University of Hannover, Germany. He received his Diploma in electrical engineering from the University of Hannover in 1992. His research interests are in the field of image analysis and computer graphics, specifically the automatic reconstruction of 3D objects from multiple camera views.



Yakup Paker is a professor of parallel computing at Queen Mary and Westfield College, London University, UK. He received a BS in electrical engineering from the Technical University of Istanbul and a PhD from Columbia University. His research interests include parallel architectures and software environments, and their applications to real-time vision and multimedia.



Wolfgang Schmidt is a research engineer at DVS Digital Video Systems in Hannover, Germany. He received his Diploma in electrical engineering from the University of Hannover in 1988. He is currently working on image storage and retrieval technologies for broadcast and multimedia applications.



Graham Thomas is with BBC Research and Development at Kingswood Warren in Surrey, UK. He received his MA in physics from Oxford University in 1983 and obtained his PhD from the University of Essex in 1990 for work on motion estimation. His research interests include motion-compensated processing and image coding for TV applications.

Readers can contact Paker at the Computer Science Department, Queen Mary and Westfield College, University of London, Mile End Road, London E1 4NS, UK, email paker@dcs.qmw.ac.uk. For more information about the Mona Lisa project, see <http://www.dcs.qmw.ac.uk/research/parallel/monalisa>.