# Virtual*ized* Reality: Concept and Early Results

Takeo Kanade, P. J. Narayanan, Peter Rander

August 17, 1995
CMU-CS-95-153

School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890

A version of this report appeared in the IEEE Workshop on the *Representation of Visual Scenes*, Boston, June 24, 1995.

## Abstract

The visual medium evolved from early paintings to the realistic paintings of the classical era to photographs. The medium of moving imagery started with motion pictures. Television and video recording advanced it to show action "live" or capture and playback later. In all of the above media, the view of the scene is determined at the transcription time, independent of the viewer.

We have been developing a new visual medium called virtualized reality. It delays the selection of the viewing angle till view time, using techniques from computer vision and computer graphics. The visual event is captured using many cameras that cover the action from all sides. The 3D structure of the event, aligned with the pixels of the image, is computed for a few selected directions using a stereo technique. Triangulation and texture mapping enable the placement of a "soft-camera" to reconstruct the event from any new viewpoint. With a stereo-viewing system, virtualized reality allows a viewer to move freely in the scene, independent of the transcription angles used to record the scene.

Virtualized reality has significant advantages over virtual reality. The virtual reality world is typically constructed using simplistic, artificially-created CAD models. Virtualized reality starts with the real world scene and virtualizes it. It is a fully 3D medium as it knows the 3D structure of every point in the image.

The applications of virtualized reality are many. Training can become safer and more effective by enabling the trainee to move about freely in a virtualized environment. A whole new entertainment programming can open by allowing the viewer to watch a basketball game while standing on the court or while running with a particular player. In this report, we describe the hardware and software setup in our "studio" to make virtualized reality movies. Examples are provided to demonstrate the effectiveness of the system.

This work was supported by ARPA grant DAAH04-93-G-0428 and is monitored by the Army Research Office of the Department of the Army. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of ARPA, DoD or the U.S. government.

**Keywords:** Computer Vision, Stereo, Virtual Reality, Computer Graphics, Image Processing

# 1 Introduction

We have a few visual media available today: paintings, photographs, moving pictures, television and video recordings. They share one aspect: the view of the scene is decided by a "director" while recording or transcribing the event, independent of the viewer.

We describe a new visual medium called *virtualized reality*. It delays the selection of the viewing angle till *view time*. To generate data for such a medium, we record the events using many cameras, positioned so as to cover the event from all sides. The time-varying 3D structure of the event, described in terms of the depth of each point and aligned with the pixels of the image, is computed for a few of the camera angles — called the *transcription angles* — using a stereo method. We call this combination of depth and aligned intensity images the *scene description*. The collection of a number of scene descriptions, each from a different transcription angle is called the *virtualized* world. Once the real world has been virtualized, graphics techniques can render the event from any viewpoint. The scene description from the transcription angle closest to the viewer's position can be chosen dynamically for rendering by tracking the position and orientation of the viewer. The viewer, wearing a stereo-viewing system, can freely move about in the world and observe it from a viewpoint chosen dynamically at *view time*.

Virtualized reality improves traditional virtual reality. Virtual reality allows viewers to move in a virtual world but lacks fine detail as their worlds are usually artificially created using simplistic CAD models. Virtualized reality, in contrast, starts with a real world and virtualizes it.

There are many applications of virtualized reality. Training can become safer and more effective by enabling the trainee to move about freely in a virtualized environment. A surgery, recorded in a virtualized reality studio, could be revisited by medical students repeatedly, viewing it from positions of their choice. Telerobotics maneuvers can be rehearsed in a virtualized environment that feels every bit as real as the real world. True telepresence could be achieved by performing transcription and view generation in real time. And an entirely new generation of entertainment media can be developed: basketball enthusiasts and broadway aficionados could be given the feeling of watching the event from their preferred seat, or from a seat that changes with the action.

Stereo or image-matching methods, which are the key components in virtualized reality, are well-studied. Precise reconstruction of the whole scene using a large number of cameras is, however, relatively new. Kanade[6] proposed the use of multi-camera stereo using supercomputers for creating 3D models to enrich the virtual world. Rioux, Godin and Blais[16] outlined a procedure to communicate complete 3D information about an object using depth and reflectance. Fuchs and Neuman[3] presented a proposal to achieve telepresence for medical applications. Some initial experiments were conducted at CMU using the video-rate stereo machine[7][9], by the team of UNC, UPenn and CMU[2], and at Tsukuba by Ohta and Satoh[10]. Laveau and Faugeras[8] attempt "view transfer" with uncalibrated cameras using epipolar constraints alone.

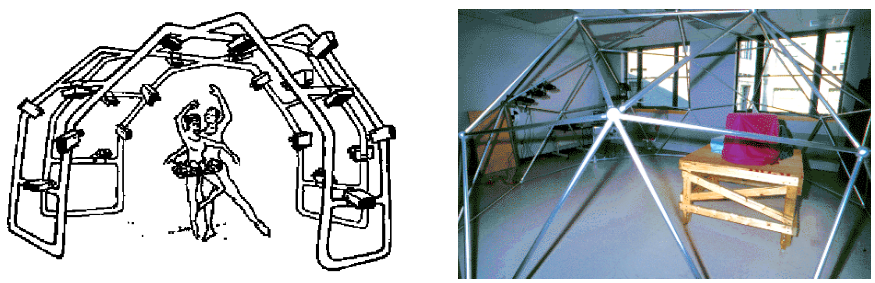
This report introduces the concept of virtualized reality. We present the three stages of creating a virtualized real scene — scene transcription, structure extraction and view generation — in the next three sections. Early examples from the virtualized reality studio are interspersed with the discussion to elucidate the concepts. Our experimental setup at present consists of a 5m dome, 10 cameras and VCRs, a digitizing setup, several workstations and a Silicon Graphics Onyx RE2 graphics workstation.

# 2 Scene Transcription

The central idea of this research is that we can virtualize real-world scenes by capturing *scene description*s — the 3D structure of the scene aligned with its image — from a number of *transcription angles*. The scene can be synthesized from any viewpoint using one or more scene descriptions. The facility to acquire the scene descriptions is called the *virtualized reality studio.* Any such studio should cover the action from all angles. Stereo techniques used to extract the scene structure require images corresponding to precisely the same time instant from every camera to be fed to them in order to accurately recover 3D scene structure. We potentially need to virtualize every frame in video streams containing fast moving events to satisfactorily reproduce the motion. Therefore, the studio should have the capability to record and digitize every frame of each video stream synchronously. We elaborate on the physical studio, the recording setup and the digitizing setup in this section.

## 2.1 The Studio Setup

Figure 1(a) shows the studio we have in mind. Cameras are placed all around the dome, providing views from angles surrounding the scene. Figure 1(b) show the studio we have built using a hemispherical dome, 5 meters in diameter, constructed from nodes of two types and rods of two lengths. We currently have 10 cameras — 2 color cameras and 8 monochrome ones — to transcribe the scene. We typically arrange them in two clusters, each providing a scene description, with the transcription angles given by the color cameras. The cameras are mounted on special L-shaped aluminum brackets that can be clamped on anywhere on the rods.



(a) (b)

Figure 1: The virtualized reality studio. (a) Conceptual. (b) The dome

## 2.2 Synchronous Multi-camera Recording

To synchronously record a set of cameras, a single control signal could be supplied to the cameras to simultaneously acquire images and to the video recording equipment to simultaneously store the images. In order to implement this approach directly in digital recording hardware, the system would need to handle the real-time video streams from each camera. For a single color camera providing 30 images per second, 512x512 pixels per image, 3 color bands per pixel, and 8 bits per color band, the system would need to handle 22.5 MBytes of image data per second. Even if loss-less compression could reduce this bandwidth by a factor of 3, a small, a 10-camera system would need a sustained 75 MBytes per second of bandwidth in addition to real-time image compression. Typical image capture and digital storage systems, however, fall far short of providing this capacity. For example, our current system — a Sun Sparc 20 workstation with a $K^2T$ V300 digitizer — can capture and store only about 750 KBytes per second. Specialized hardware could improve the throughput but at a substantially higher cost.

We developed an off-line system to synchronously record frames from multiple cameras. The cameras are first synchronized to a common sync signal. The output of each camera is time stamped with a common Vertical Interval Time Code (VITC) and recorded on tape using a separate VCR. The tapes are digitized individually off-line using a frame grabber and software that interprets the VITC time code embedded in each field. We can capture all frames of a tape by playing the tape as many times as the speed of the digitizing hardware necessitates. The time code also allows us to correlate the frames across cameras, which is crucial when transcribing moving events. Interested readers can refer to a separate report[17] for more details on the synchronous multi-camera recording and digitizing setup. Figure 2 shows a still frame as seen by six cameras of the virtualizing studio digitized using the above setup.

# 3 Structure Extraction

We use the multi-baseline stereo (MBS) technique [9] to extract the 3D structure from the multi-camera images collected in our virtualized reality studio. Stereo algorithms compute estimates of scene depth from correspondences among images of the scene. The choice of the MBS algorithm was motivated primarily by two factors. First, MBS recovers dense depth maps — that is, a depth estimate corresponding to every pixel in the intensity images — which is needed for image reconstruction. Second, MBS takes advantage of the large number of cameras that we are using for scene transcription to increase precision and reduce errors in depth estimation.

## 3.1 Fundamentals of Multi-Baseline Stereo

To understand the MBS algorithm, consider a multi-camera imaging system in which the imaging planes of the cameras all lie in the same physical plane and in which the cameras have the same focal length $F$. For any two of the cameras, the disparity $d$ (the difference in the positions of corresponding points in the two images) and the distance $z$ to the scene point are related by

$$d = BF\frac{1}{z}$$

where $B$ is the baseline, or distance between the two camera centers. The simplicity of this relation makes clear one very important fact: the precision of the estimated distance increases as the baseline between the cameras increases. In theory, the cameras can be placed as far apart as possible. Practical experience using

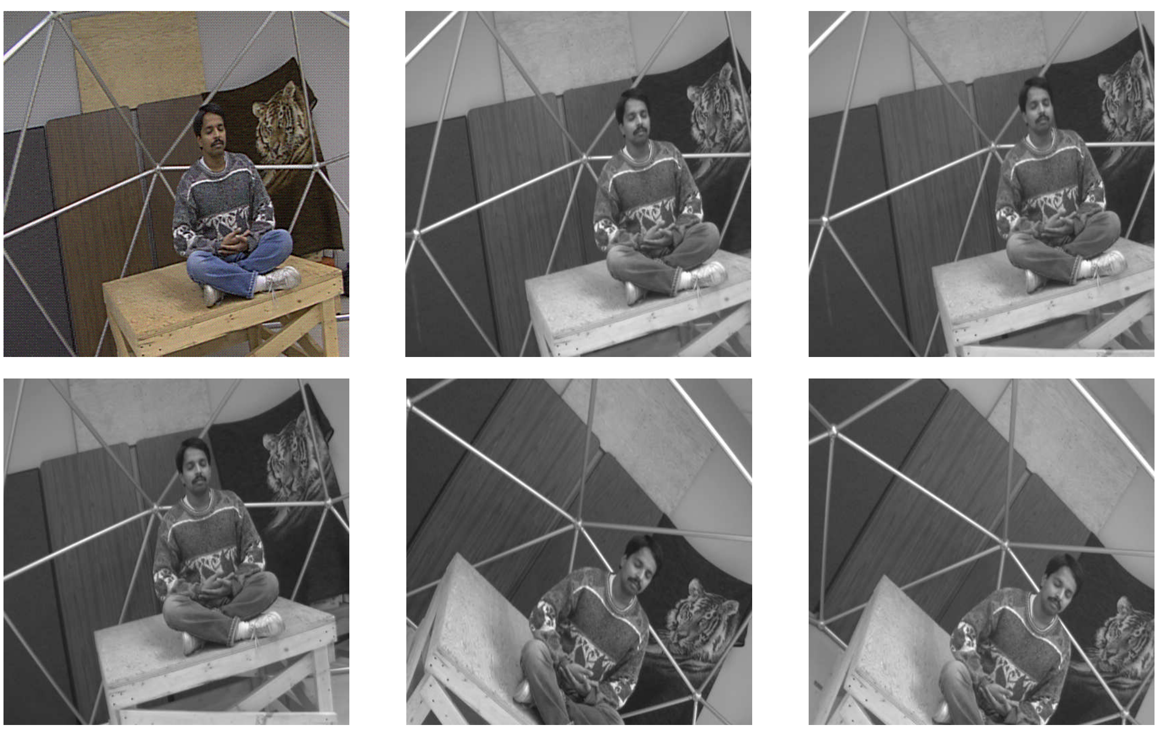


Figure 2: A scene seen by six cameras.

stereo systems reveals, however, that increasing the baseline also increases the likelihood of mismatching points among the images. There is a trade-off between the desires for correct correspondence among images (using narrow baselines) and for precise estimates of scene depth (using wide baselines).

The multi-baseline stereo technique attempts to eliminate this trade-off by simultaneously computing correspondences among pairs of images from multiple cameras with multiple baselines. In order to relate correspondences from multiple image pairs, we rewrite the previous equation as

$$\frac{d}{BF} = \frac{1}{z} = \varsigma$$

which indicates that for any point in the image, the inverse depth ($\varsigma$) is constant since there is only one depth $z$ for that point. If the search for correspondences is computed with respect to $\varsigma$, it should consistently yield a good match at the correct value of $\varsigma$ independently of the baseline $B$. With multiple (more than 2) cameras, correspondences can now be related across camera pairs, since the searching index $\varsigma$ is independent of the baselines. The resulting correspondence search combines the correct correspondence of narrower baselines with the higher precision of wider baselines, and it has been proven that it yields a unique match of high precision [9].

One way to find correspondences between a pair of images is to compare a small window of pixels from one image to corresponding windows in the other image. The correct position of the window in the second image is constrained by the camera geometry to lie along the epipolar line of the position in the first image. The matching process involves shifting the window along this line as a function of $\varsigma$, computing the match error — using normalized correlation or sum of squared differences (SSD) — over the window at each position, and finding the minimum error. The estimate of inverse depth, $\hat{\varsigma}$, is the $\varsigma$ at this minimum.

To demonstrate the advantages of multi-baseline stereo, consider the data presented in Figure 3. Part (a) shows match error as a function of $\varsigma$ for 3 camera pairs. In this set of cameras, we see both of the problems previously discussed: poor localization (in the top curve) for a shorter baseline and false minima (in the bottom curve) for a longer baseline. Applying the multi-baseline stereo algorithm to this data yields the error curve in Figure 3(b). This curve has only the single minimum at the correct location with a sharp profile.

## 3.2 Depth Map Editing

Window-based correspondence searches suffer from a well-known problem: inaccurate depth recovery along depth discontinuities and in regions of low image texture. The recovered depth maps tend to “fatten” or “shrink” objects along depth discontinuities. This phenomena occurs because windows centered near the images of these discontinuities will contain portions of objects at two different depths. When one of these windows is matched to different images, one of two situations will occur. Either the foreground object will occlude the background object so that depth estimates for the background points will incorrectly match to the portion of the foreground in the window, or both the foreground and background regions will remain visible, leading to two likely candidate correspondences. In regions with little texture — that is, of fairly constant intensity — window-based correspondence searches yield highly uncertain estimates of depth. Consider, for example, a stereo image pair with constant intensity in each image. With no intensity variation, any window matches all points equally well, making any depth estimates meaningless.

To address this inaccuracy in depth recovery, we could reduce the window size used during matching, potentially matching individual pixels. This approach reduces the number of pixels effected by depth discontinuities. By doing so, however, we also reduce the amount of image texture contained within the window, increasing the uncertainty of the recovered depth estimate. Conversely, we could increase the size of the window to give more image texture for matching. This action increases the image texture contained in the window, but also increases the area effected by the discontinuities. Optimizing the window size requires trading off the effects of the depth discontinuities with those of the low-texture regions.

In order to work around this trade-off, we have incorporated an interactive depth map editor into our process of structure extraction. Rather than send the MBS-computed depth maps directly on to the next processing stage, we instead manually edit the depth map to correct the errors that occur during automatic processing. While a good window size still helps by reducing the number of errors to be corrected, it is less important in this approach because the user can correct the problems in the depth maps. We are currently exploring modifications to the stereo algorithm in an effort to reduce or eliminate this need for human intervention.

## 3.3 General Camera Configurations

For general camera positions, we perform both intrinsic and extrinsic camera calibration to obtain epipolar line constraints, using either an approach from Szeliski and Kang [13] or one from Tsai [14]. Using the recovered calibration, any point in the 3D coordinate system of the reference camera can be mapped to a point in the 3D coordinate system of any of the other cameras. To find correspondences, we again match a reference region to another image as a function of inverse depth $\varsigma$. To find the position in the second image corresponding to this inverse depth, we convert the reference point and inverse depth into a 3D coordinate, apply the camera-to-camera mapping, and project the converted 3D point into the other image. As with the parallel-camera configuration, the full search is conducted by matching each reference image point to the other images for each possible $\varsigma$. We then add the match error curves from a set of image pairs and search for the minimum of the combined error function. Figure 3 (c) shows the depth map recovered by applying this approach to the input images shown in Figure 2. The depth map has 74 levels for a depth range of 2 meters to 5 meters.

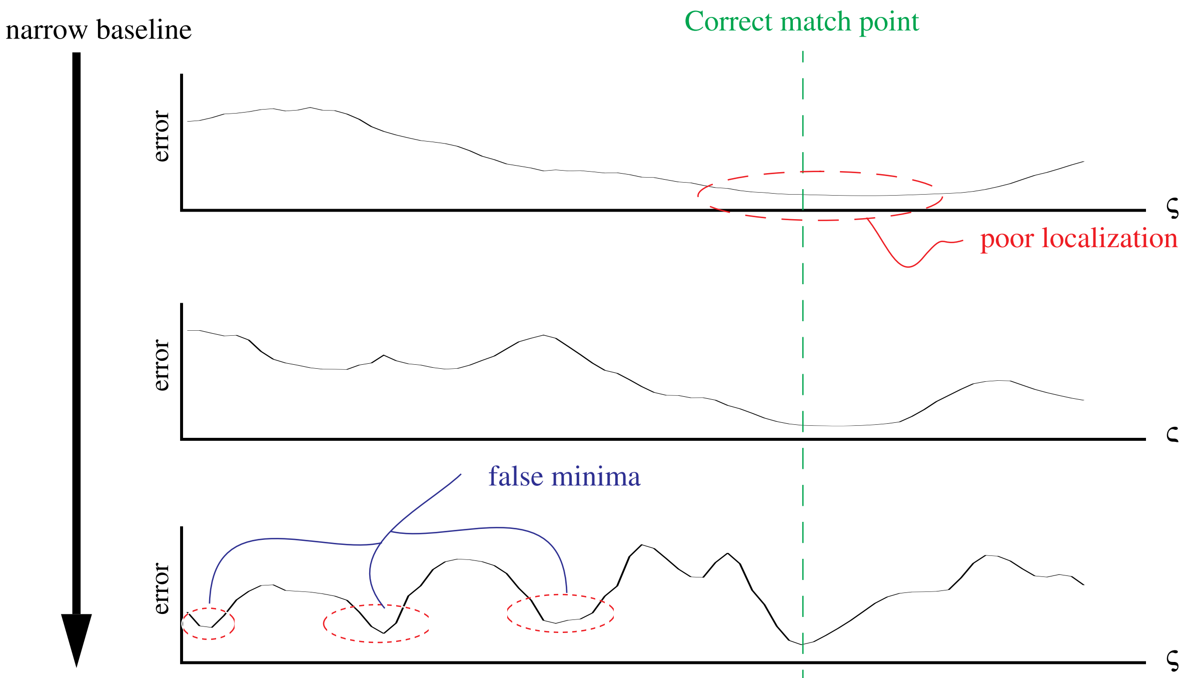
# 4 View Generation

We described how to "virtualize" an event in terms of a number of scene descriptions in the previous sections. The medium of virtualized reality needs to synthesize the scene from arbitrary viewpoints using these scene descriptions. To render the scene from other viewpoints using graphics workstations, we translate the scene description into an object type, such as a polygonal mesh. We texture map an intensity image onto the rendered polygons, generating visually realistic images of the scene. Graphics workstations have specialized hardware to render them quickly. A Silicon Graphics Onyx/RE2 can render close to 1 million texture mapped triangles per second.

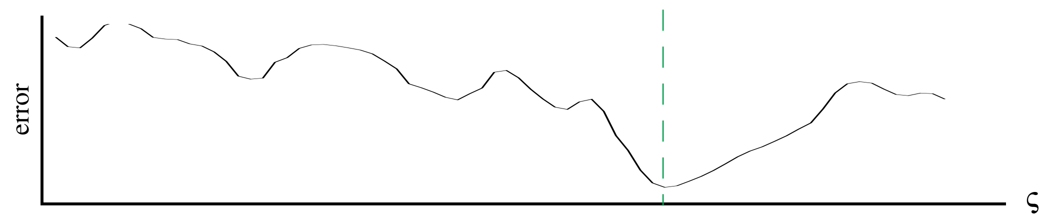
We describe how new views are generated from a single scene description first. The generated view will be lower in quality as the viewpoint gets far from the transcription angle. We discuss how we can use multiple scene descriptions to get realistic rendering from all angles.

## 4.1 Using a Single Scene Description

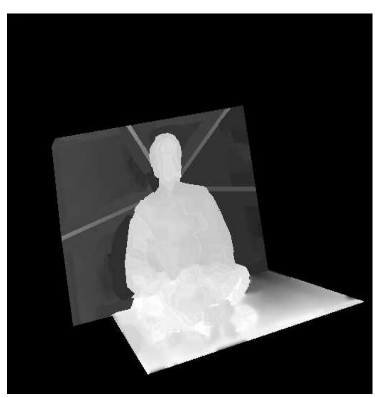
A scene description consists of a depth map providing a dense three dimensional structure of the scene aligned with the intensity map of the scene. The point $(i, j)$ in the depth map gives the distance of the intensity image pixel $(i, j)$ from the camera. We convert the depth map into a triangle mesh and the intensity map to texture to render new views on a graphics workstation. There are two aspects of performing this translation realistically: object definition and occlusion handling.




(a) Match error for camera pairs




(b) Match error for multi-baseline technique



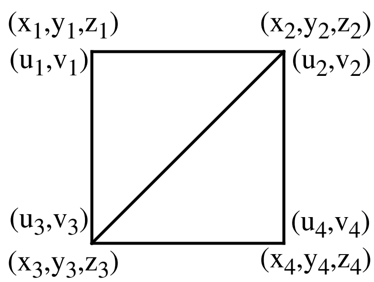
(c) Depth map for images in Figure 2

Figure 3: Results of multi-baseline stereo algorithm.

### 4.1.1 Object Definition

Graphics rendering machines synthesize images of a scene from an arbitrary point of view given a polygonal representation of the scene. Texture mapping pastes an intensity image onto these rendered polygons, generating visually realistic images of the scene from arbitrary view points. We currently generate a triangle mesh from the depth map by converting every $2 \times 2$ section of the depth map into two triangles. Figure 4 illustrates how the mesh is defined. The *(x, y, z)* coordinates of each point in the image are computed from the image coordinates and the depth, using the intrinsic parameters of the imaging system. Each vertex of the triangle also has a texture coordinate from the corresponding intensity image. This simple method results in $2 \times (m-1) \times (n-1)$ triangles for a depth map of size $m \times n$. The number of triangles for the depth map shown in Figure 3 is approximately 200,000. Though this is a large number of triangles, the regularity makes it possible to render them efficiently on graphics workstations.

We reduce the number of triangles in our scene definition by adapting an algorithm developed by Garland and Heckbert that simplifies a general dense elevation/depth map into planar patches[4]. The algorithm computes a triangulation using the smallest number of vertices given a measure for the maximum deviation from the original depth map. The procedure starts with two triangles defined by the outer four vertices. It repeatedly grows the triangle mesh by adding the vertex of maximum deviation and the corresponding triangle edges till the maximum deviation condition is reached. Using this technique, we have reduced mesh size by factors of 20 to 25 on typical scenes without affecting the visual quality of the output.




Triangle 1:

Vertex 1: $(x_1,y_1,z_1)$ texture coordinate: $(u_1/m, v_1/n)$

Vertex 2: $(x_2,y_2,z_2)$ texture coordinate: $(u_2/m, v_2/n)$

Vertex 3: $(x_3,y_3,z_3)$ texture coordinate: $(u_3/m, v_3/n)$

Triangle 2:

Vertex 1: $(x_2,y_2,z_2)$ texture coordinate: $(u_2/m, v_2/n)$

Vertex 2: $(x_3,y_3,z_3)$ texture coordinate: $(u_3/m, v_3/n)$

Vertex 3: $(x_4,y_4,z_4)$ texture coordinate: $(u_4/m, v_4/n)$

Figure 4: Triangle mesh and texture coordinate definition

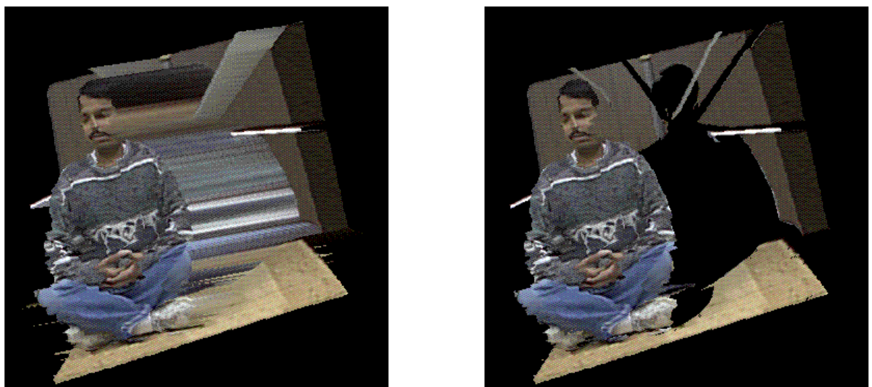
### 4.1.2 Occlusion Handling

The simple rendering technique described above treats the entire depth map as one large surface, connecting pixels across depth discontinuities at object boundaries. This introduces an artificial surface bridging the discontinuity, with the few pixels of texture stretched over the surface. When generating views for angles far from the transcription angle, these surfaces become large and visually unrealistic; in Figure 5(a), for instance, the person and the wall appear to be connected. We therefore delete these artificial surfaces by not rendering the triangles that overlap discontinuities, resulting in “holes” as seen in Figure 5(b). We fill these holes using other scene descriptions as explained in Section 4.2.

### 4.1.3 Experimental Results

Figure 6 shows how the example scene appears from a few different viewpoints when rendered with this technique. While still not completely realistic — we expect to see something behind the man — the visual realism of this method is far superior to the initial approach.

The discussion to this point has focussed on virtualizing a single, static scene. It is also possible to virtualize moving scenes by virtualizing each frame separately. The resulting virtualized reality movie can be played



(a) (b)

Figure 5: (a) View without discontinuity compensation. (b) With compensation

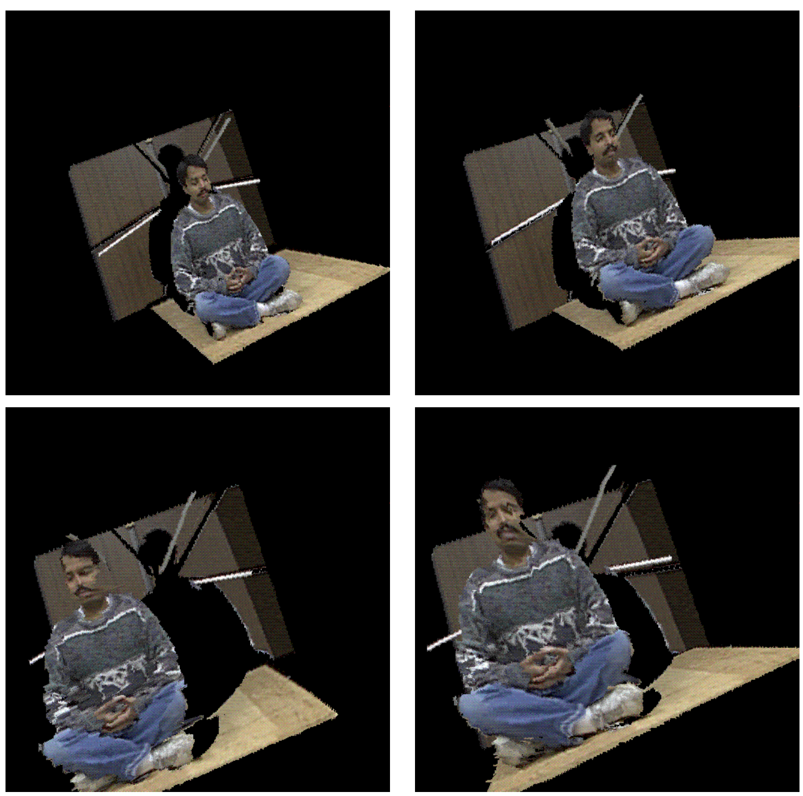


Figure 6: The scene from four alternate viewpoints

with the viewer standing still anywhere in the world by rendering each frame from the viewer's position. The scene can also be observed by a viewer whose movement through the world is independent of the motion in the scene. Figure 7 shows seven frames of a basketball sequence from the reference transcription point and from a synthetically-created moving viewpoint.

(a) Original reference images

(b) Synthesized from a moving viewpoint. The starts left and above the original transcription view and moves to the right.

Figure 7: Seven frames of a basketball sequence

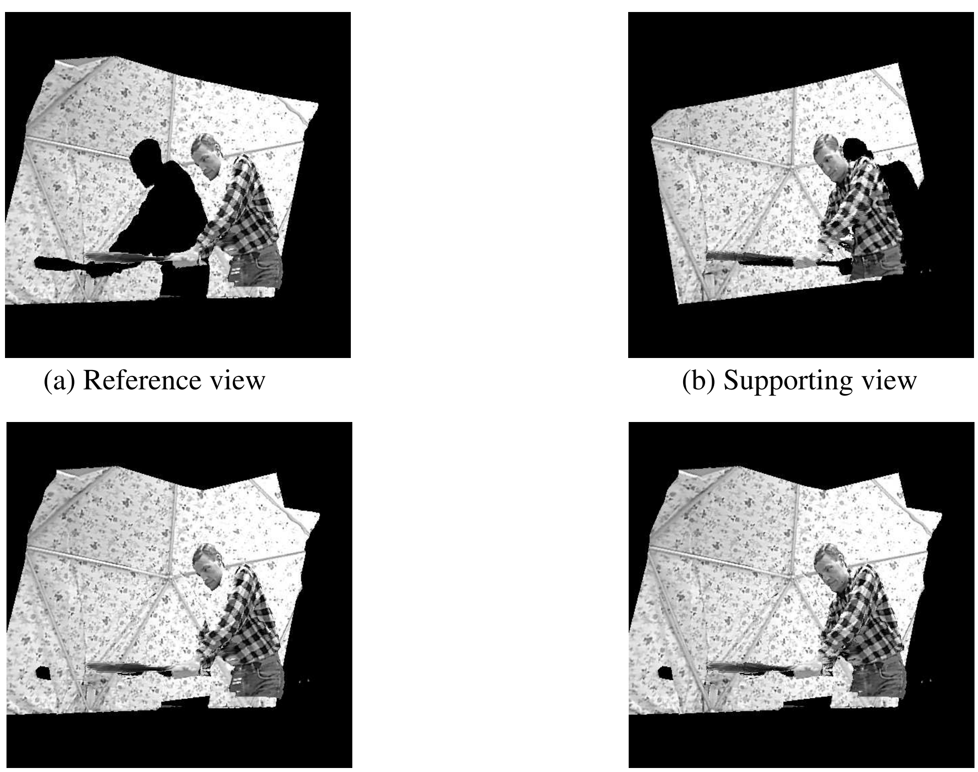
## 4.2 Merging Multiple Scene Descriptions

There are two reasons for combining the scene descriptions from multiple transcription angles while generating new views. First, as discussed in Section 4.1, depth discontinuities appear as holes in views far from the transcription angle when using a single scene description. We should "fill" these holes using a scene description from another transcription angle for which the portion of the scene is not occluded. Second, the intensity image used for texturing gets compressed or stretched when the viewing angle is far from the transcription angle, resulting in poor quality of the synthesized image. If the viewer strays far from the starting position, we should choose the most direct transcription angle for each viewing angle to minimize this degradation.

Filling the gaps requires some thought. In a well calibrated ideal world, the scene descriptions computed from different transcription angles will match one another exactly. In practical systems, however, they will be misaligned somewhat due to the errors in calibration and matching. The triangle vertices that fall on the gap contours will not exactly match in two transcription angles. These factors complicate view merging.

One strategy is to combine the scene descriptions from all transcription angles ahead of time to generate a model of the scene that contains all the necessary detail. Several methods are available to register and model objects from multiple range images[5][11][12][15]. Such a consolidated model attempts to give one grand description of the entire world. We only require the best partial description of the world visible from a particular viewing angle at any time. Such a partial description is likely to be more accurate due to its limited scope; inaccuracies in the recovery of the portion not seen will not affect it. It is likely to be simpler than a consolidated model of the scene, lending easily to real time view generation. The partial description we use consists of a reference scene description from the transcription angle closest to the viewing angle plus one or two supporting ones. The reference description is used for rendering most of the view and the supporting ones are used for filling the gaps.

For merging the views from the reference and the supporting scene descriptions, we have experimented with two strategies: Merging *in image* and merging *while rendering*. In the former, the image for the same viewing angle are rendered using the reference transcription angle and the supporting ones. The holes in the rendering of the reference description are filled directly with the pixel values from the rendering of another. Figure 8(c) shows the results of merging a scene in image using the reference scene description shown in Figure 8(a) and a supporting scene description shown in Figure 8(b). This method holds promise when the transcription angles are fairly densely distributed. Merging while rendering involves rendering using the



(a) Reference view

(b) Supporting view

(c) Merging in image

(d) Merging while rendering

Figure 8: Merging two scene descriptions

reference and supporting transcriptions angles in a common reference frame, using a common Z-buffer. The results of merging the same scenes while rendering is shown in Figure 8(d). This method has filled the holes -- like the right shoulder of the person -- that cannot classified as holes from the reference view.

# 5 The Next Step

We plan to go the whole way to virtualize events from all angles in the studio and let a viewer move about the virtualized world freely in the next step. This requires: (1) covering the dome with cameras to provide views from all directions, (2) extending the software to permit viewing from all angles on the front or on the back, and (3) developing the view generation software to merge multiple views and fill in holes. We divide our plans into five areas: Studio hardware, Stereo program, View generation and merging, Scene enhancement, and Stereoscopic viewing and position sensing.

## 5.1 Studio Hardware

We need more cameras to virtualize a scene from all angles. We currently have ten cameras and the associated recording equipments for the same, divided into two sets. Each set consists of 5 cameras: 1 color camera and 4 monochrome cameras. The color camera is used as the reference camera that provides texture for view generation. We now describe the final, maximum-configuration setup and its requirements and then discuss an intermediate stage that we expect to reach in six months.

### 5.1.1 Fully Equipped Studio

We plan to match the final camera placement to the geometry of the dome. At or above a height of 1.5 meters, the dome has 35 bars connecting 16 nodes. The maximum configuration is to place one full set (5 cameras) at each of these nodes, totalling 5x16 cameras (16 color cameras and 64 monochrome cameras). The off-line implementation of the maximum configuration requires 80 VCRs to record the outputs of each camera, 80 time code inserters to insert VITC time code, a corresponding number of sync lines etc. To update our studio to this level, we require 14 additional color cameras, 56 additional monochrome cameras and 70 additional VCRs and time-code converters. Alternatively, we could place one color camera at each node, and one monochrome camera in the middle of each bar which is shared by the neighboring reference camera. In this case, 29 monochrome cameras and associated equipment can be saved. Using this configuration, there will be 16 transcription angles, one at each node, with the scene descriptions approximately 36 degrees apart. This configuration provides full hemispherical viewing of the scene — i.e. the scene can be viewed from any side and from the top.

### 5.1.2 Intermediate Stage

Before we fully equip the studio, we would like to experiment with various view selection and view merging strategies. We can do this by expanding the number of scene descriptions to 8 in an intermediate stage. This will require 6 additional stereo camera sets — 6 color cameras, 24 monochrome cameras, 30 VCRs and 30 time code converters. With the 8 stereo sets, we will for the first time be able to provide full 360 degree viewing (viewing from any side of the scene) by distributing the sets in two circles around the dome. Since the dome has ten nodes at an approximate height of 1.5 meters, a camera set will be place at every second node at this level, yielding 5 scene descriptions placed 72 degrees apart. A top view of this configuration is given in Figure 10. The other 3 sets will be placed in a circle above the 5-set circle to provide top views of the scene.

Multiple scene descriptions are also used to fill the gaps when the viewing angle differs from the reference angle. Several strategies are possible for this. We have already experimented with some of them. Scene descriptions from non-parallel directions are necessary to do this effectively, whatever the strategy. We would like to research this with a transcription angle arrangement like that shown in Figure 11. This configuration provides a wide range of views, including a top view of the scene.

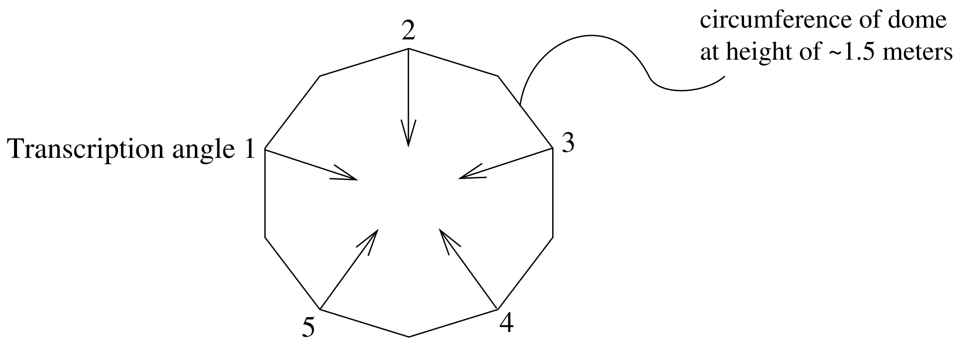



Figure 9: Camera cluster arrangement for 360 degree coverage — top view

### 5.1.3 On-line Capture

We would like to build an on-line image capture system to replace the video tape based system. As a first step, an iWarp supercomputer based image capture system is currently being set up for the studio. This system will have the capability to capture 64 frames each of 16 input channels on-line and hold them in memory before writing them to disk. It could record 2 seconds of action at video rate or longer at reduced rates. The frames are captured live and saved in the memory of the processors till memory runs out. This system has been successfully used by Job Webb at CMU. It is not a scalable system, however, and will never be able to handle the 112 channels (16 * 3 = 48 for the color cameras plus 64 for the monochrome ones) the maximally equipped studio needs.

We are exploring building a PC-farm based scalable on-line image capture system. The basic block of such a system would be a high-speed PC with a video capture board. One video channel of 512x512 pixel images at 8 bits per pixel amounts to 7.5 MBytes per second of data with no compression. (That bandwidth is expected to drop to about 3 MBytes per second with simple loss-less compression.) Current high-performance hard disk arrays (e.g. RAID drives) can support a sustained throughput of 25 MBytes per second. *Imaging Technology* claims that its Color Acquisition Modules (AM-CLR) can sustain 75 MBytes per second digitization. A single Pentium PC with 3 AM-CLR boards connected through a PCI bus (which has a total bandwidth of 132 MBytes per second), and three disk arrays could perform real-time capture and storage of 9 channels of video to disk. Although the effort to develop a single working system would be significant, the

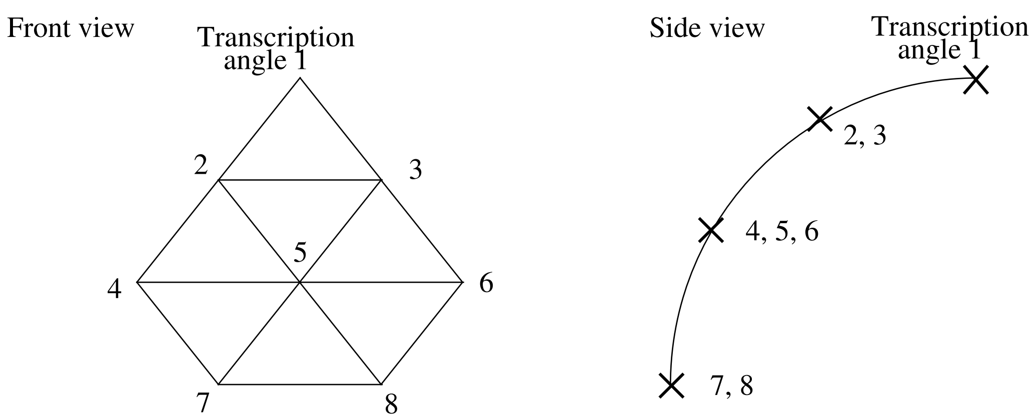



Figure 10: Camera cluster arrangement to study filling of occluded areas
(not to scale)

system is completely scalable simply by replicating the successful prototype. To capture all 112 channels, we will need 13 such PCs to fully equip the virtualized reality studio for on-line capture. VITC time codes can again be used to correlate frames captured using different PCs, but the data will already be in digital form so correlating frames will simply amount to labelling the images already stored on disk, a much faster process than individual playback and digitization of analog video tapes.

## 5.2 Stereo Program

We plan to improve the stereo program in two ways. First, stereo computation requires precise knowledge of each camera's intrinsic parameters (such as focal length) and extrinsic parameters (position and orientation in a coordinate system common to all cameras). These parameters are recovered using traditional calibration techniques. We believe that we can improve the precision of the parameter estimates by using a calibration technique based on epipolar error. Second, because stereo must be computed for every set of cameras, the stereo program should run as fast as possible. The current program takes approximately 45 minutes on a *Sun* Sparc 20 to process a 5-image set with 490x636 images at 50 disparities and for a window size of 11x11 pixels. By distributing the computation over even 10 workstations, for example (we have a 110-computer network of Suns, 18 of which have power comparable to the Sparc 20), we can reduce this time to less than 5 minutes.

## 5.3 View Generation and Merging

### 5.3.1 Selection of the Transcription Angle

The view generation scheme gives good results when the viewing angle is not too far from the transcription angle used to generate new views. Therefore, it is essential to use the best transcription angle for a position and orientation of the viewer. When the transcription centers are distributed on the surface of a sphere, oriented approximately towards the center, the best transcription angle is also the nearest transcription center. This can be determined from a three-dimensional Delaunay triangulation of the transcription points. When the viewer crosses the triangulation boundary, the reference scene description is switched to that of the new region.

### 5.3.2 Filling Gaps

The scene descriptions from the adjacent transcription angles are used to fill the gaps created by occlusions from the reference angle. Our experiment of filling the gaps at the image level using two viewpoints yielded promising results. We propose to use at least two supporting scene descriptions, non-parallel to each other, to fill the gaps properly. We can create scenarios with gaps that cannot be filled for any arrangement of the reference and supporting transcription angles as the problem is related to non-convex covering. We believe we can achieve satisfactory coverage with four supporting scene descriptions, however. We also plan to study the filling of gaps by rendering the object definitions from the supporting scene description angles directly on top of the reference one wherever gaps exist.

## 5.4 Scene Enhancement

There are other avenues towards a better synthetic image to be explored. Texture mapping does not lend itself to lighting and shading variation very easily. Highlights do not move in texture mapped images, creating a less than natural feeling. There is research going on in CMU and other places to extract the reflection model of the surface from the texture image. If the reflection model is known, we can correct the texture so that other reflection and shading models can be used with it. We would like to enhance the virtualized environment by taking advantage of such work.

### 5.5 Stereoscopic View and Position Sensing

For a realistic immersive feeling, stereoscopic display is necessary. We have experimented with the *Crystal Eyes* system for the *Silicon Graphics*. Similar display systems are today available for the PCs, such as the *3D Max*. All they need is separate left and right images. The left and right images need to be changed with the viewer's position to get full 3D feeling. We are currently looking into the commercially available head-tracking systems like that of *Stereo Graphics*.

# 6 Conclusions

We introduced and elaborated on the concept of virtualized reality in this report. It combines techniques from computer vision and computer graphics to *virtualize* a real world event and to let a viewer move about freely in the virtualized world. We also demonstrated the efficacy of virtualized reality using scenes virtualized in our studio to make such movies. It is today possible to virtualize an event such as a surgery and let trainees move about it in a realistic recreation of the surgery in a manner they prefer. We plan to push the training and entertainment applications of virtualized reality in the future.

**Acknowledgments**: We would like to thank Richard Szeliski and Sing Bing Kang for the use of their calibration program. We also acknowledge the use of Reg Wilson's calibration program.

# 7 References

1 M. J. Frankel and J. A. Webb. Design, Implementation and Performance of a Multi-Camera Interactive Video Display System. *Computer Architectures for Machine Perception*. To appear. Como, Italy, 1995.

2 H. Fuchs, G. Bishop, K. Arthur, L. McMillan, R. Bajcsy, S.W. Lee, H. Farid, and T. Kanade. Virtual Space Teleconferencing using a Sea of Cameras, In *Proceedings of the First International Symposium on Medical Robotics and Computer Assisted Surgery*, pp.161-167, 1994

3 H. Fuchs and U. Neuman. A Vision Telepresence for Medical Consultation and other Applications. In *Sixth International Symposium of Robotics Research*, pages 555–571, 1993.

4 M. Garland and P. Heckbert. Terrain Simplification. *In Preparation.*

5 H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle, Piecewise Smooth Surface Reconstruction, *Computer Graphics SIGGRAPH'94*, 295-302, 1994.

6 T. Kanade. User Viewpoint: Putting the Reality into Virtual Reality. *MasPar News*, 2(2), Nov. 1991.

7 T. Kanade. Very Fast 3-D Sensing Hardware. In *Sixth International Symposium of Robotics Research*, pages 185–198, 1993.

8 S. Laveau and O. Faugeras. 3-D Scene Representation as a Collection of Images and Fundamental Matrices, *INRIA Tech Report* 2205, 1994.

9 M. Okutomi and T. Kanade. A multiple-baseline stereo, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):353-363, 1993.

10 K. Satoh and Y. Ohta. Passive Depth Acquisition for 3D Image Displays. In *IEICE Transactions on Information and Systems*, E77-D(9), Sep. 1994.

11 H. Shum, An Integral Approach to Free-form Object Modelling, *Carnegie Mellon University Tech Report* CMU-CS-95-135, 1995.

12 M. Soucy and D. Laurendeau, Multi-Resolution Surface Modelling from Multiple Range Views, *Proceedings of IEEE CVPR'92*, 348-353, 1992.

13 R. Szeliski and S.B. Kang, Recovering 3D Shape and Motion from Image Streams using Non-Linear Least Squares, *Journal of Visual Communication and Image Representation*, 5(1):10-28, 1994

14 R. Tsai, A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses, *IEEE Journal of Robotics and Automation*, RA-3(4):323-344, 1987.

15 G. Turk and M. Levoy, Zippered Polygon Meshes from Range Images, *Computer Graphics SIGGRAPH'94*, 311-318, 1994.

16 M. Rioux, G. Godin, and F. Blais, Datagraphy: The Final Frontier in Communications, In *International Conference on Three Dimensional Media Technology*, 1992.

17 P. J. Narayanan, P. Rander, and T. Kanade, Synchronizing and Capturing Every Frame from Multiple Cameras, Robotics Technical Report, CMU-RI-TR-95-25, 1995.