A Methodology for Remote Virtual Interaction in Teleimmersive Environments

Ram Vasudevan, Edgar Lobaton, Gregorij Kurillo, Ruzena Bajcsy
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Hearst Mining Bldg. #475
Berkeley, CA 94720-1764
{ramv,lobaton,gregorij,bajcsy}@eecs.berkeley.edu

Tony Bernardin, Bernd Hamann
Institute for Data Analysis and Visualization (IDAV)
University of California, Davis
One Shields Ave
Davis, CA 95616
{tbernardin, hamann}@ucdavis.edu

Klara Nahrstedt
Department of Computer Science
University of Illinois at Urbana-Champaign
201 N. Goodwin Avenue
Urbana, IL 61801-2302
{klara}@illinois.edu

ABSTRACT

Though the quality of imaging devices, the accuracy of algorithms that construct 3D data, and the hardware available to render such data have all improved, the algorithms available to calibrate, reconstruct, and then visualize such data are difficult to use, extremely noise sensitive, and unreasonably slow. In this paper, we describe a multi-camera system that creates a highly accurate (on the order of a centimeter), 3D reconstruction of an environment in real time (under 30 ms) that allows for remote interaction between users. The paper addresses the aforementioned deficiencies by featuring an overview of the technology and algorithms used to calibrate, reconstruct, and render objects in the system. The algorithm produces partial 3D meshes, instead of dense point clouds, which are combined on the renderer to create a unified model of the environment. The chosen representation of the data allows for high compression ratios for transfer to remote sites. We demonstrate the accuracy and speed of our results on a variety of benchmarks and data collected from our own system.

Categories and Subject Descriptors

I.4 [**Image Processing and Computer Vision**]: Applications

General Terms

Algorithm, Design, Performance

Keywords

Real Time, 3D Video, 3D Teleimmersion, Stereo Reconstruction, Virtual Reality, Human-Computer Interaction, Visualization

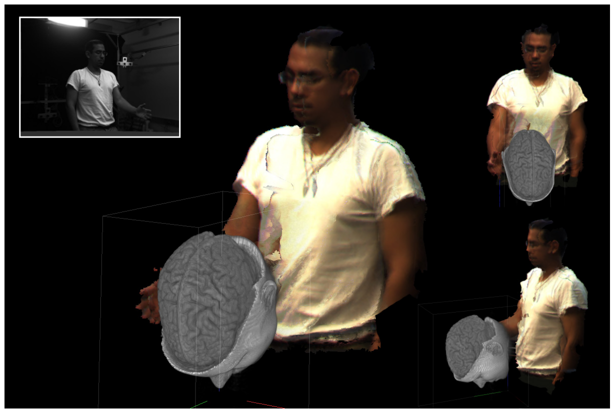


Figure 1: A user inside of the teleimmersion system interacting with an MRI dataset. The top-left image depicts a sample image from a camera and the other images depict the mirrored 3D representation of the user inside of the virtual environment from various views.

1. INTRODUCTION

Though most existing video conferencing systems make some attempt to humanize remote interaction, few are able to provide the desired immersive component of actual face-to-face communication. These systems, which rely on two-dimensional video streams between remote users, fall short of providing the desired immersive component for a number of reasons including not allowing the users to establish eye contact, not placing all users inside the same environment, or not allowing users to jointly interact with synthetic objects. Limited attempts have been made to create a more immersive experience using large displays, gaze preservation through multi-camera capturing systems [20], and matching environments (e.g., tables, chairs) between the remote locations that create the illusion of continuity of the physical space into the screen. Rather than continually improving the hardware of such systems to enhance the immersive experience, telepresence could easily be advanced by creating a shared virtual environment wherein remote users could be jointly rendered.

Teleimmersion is an emerging technology that allows users to collaborate remotely by generating a realistic 3D representation of users in real time and placing them inside a shared virtual space. Such virtual meeting spaces could allow for the possibility of collaborative work on 3D data such as medical data (e.g. MRI, CT scan, 4D ultrasound, 3D x-ray), scientific data, design models (e.g. CAD models, building designs), remote training (e.g. oil rigs, military applications), or remote teaching of physical activities (e.g. rehabilitation, dance). In fact, the utility of the teleimmersive system has been successfully demonstrated in remote dancing applications [19, 25], learning of Tai Chi movements [1, 21], and remote manipulation of virtual objects between users [6, 13].

Traditional immersive virtual reality systems often employ avatars, to represent the human user inside the computer generated environment [10]. Such systems have generated simple avatars by employing markers (e.g. body suit with tracking devices) which restrict the user's movement and provide limited spatial resolution. In contrast to these simple avatars, a full body 3D reconstruction could realistically represent the user's appearance and completely model the dynamics of movement such as facial expressions, chest deformation during breathing, and movement of hair or clothing. Unfortunately, the accurate construction of the 3D data at high frame rates has always been the common shortcoming of teleimmersive systems.

In this paper, we describe a multi-camera system that creates a highly accurate (on the order of a cm), real time (under 30 ms), 360° 3D reconstruction of users. The main contributions of this work are three-fold. First, in section 5, we develop a robust calibration of the multi-camera system and the physical space to ensure the preservation of spatial correspondences between various teleimmersive systems. Second, in section 6, we introduce a novel multiscale representation that allows for highly accurate reconstruction of a scene while allowing for high compressibility of the 3D data produced. Third, in section 7, we describe how the data produced from the multiple cameras can be integrated together to improve the overall visual quality of the reconstruction. The rest of the discussion includes a brief overview of similar work in section 2, and an overall description of the system and its components in sections 3 and 4.

2. RELATED WORK

Several attempts have been made in the past to develop real time 3D reconstruction systems to capture the human body. Most real time approaches fall into one of three categories based upon their computational approach: (1) silhouette based reconstruction, (2) voxel-based methods with space sampling, and (3) image-based reconstruction using dense stereo. In silhouette-based reconstruction, 3D information is obtained via visual hulls that are formed by intersecting generalized cones between a silhouette and the camera center. The voxel-based method, on the other hand, determines depth by sampling a uniform grid of space using color consistency. Finally, the vision-based reconstruction creates dense stereo depth-maps by correlating slightly displaced views of the same scene.

A desktop teleimmersive system based on reconstruction from silhouettes was initially proposed by Baker et al. who used five different views to obtain a 3D model of the user via a visual-hull approach [2]. The system was based on a single PC which performed 3D reconstruction and rendering of the users in a simple virtual meeting room. Though compact, the system showed limited accuracy and speed. A full-body teleimmersive system was introduced by researchers at ETH Zurich who applied silhouette-based 3D reconstruction from several cameras to capture the user inside a CAVE-like environment [6]. Active semi-transparent panels allowed for synchronous image capturing and projection of the virtual environment. Two systems were built to test the collaborative experience of between dislocated users. More recently, Ladikos et al. proposed a 3D reconstruction approach using visual hulls built on CUDA working at 30 frames per second (fps) [15]. Unfortunately, the 3D reconstruction approach employing visual hulls is limited due to its inability to recover concave surfaces. In order to deal with this shortcoming, Hasenfratz et al. presented a voxel-based interactive virtual reality system that worked at 25 fps on 1.6 GHz Pentium 4 processor, but the speed was achieved by explicitly leveraging accuracy for speed; hence, the presented results were unable to accurately reconstruct clothing and facial features [7].

In contrast to these approaches, Kanade et al. designed a full-body capturing system using image-based reconstruction with a large number of distributed cameras to capture human movement in real time (less than a single frame per second); however, no immersive feedback was added for the users (i.e. no shared virtual space was created) [11]. The first teleimmersive system was presented by researchers at the University of Pennsylvania [18]. Their system consisted of several stereo camera triplets used for image-based reconstruction of the upper body, which allowed a local user to communicate to remote users while sitting behind a desk. More recently, Sang et al. presented a teleimmersive framework based on 3D reconstruction from dense-stereo depth maps [9]. Several stereo cameras were used to perform real time (5-7 fps) reconstruction of the scene as a point cloud which was then integrated into a virtual environment.

In recent years, several benchmarks to compare the accuracy of various types of stereo and multiview reconstructions have become available. Scharstein et al. present a thorough overview of a variety of standard algorithms to perform 3D reconstruction (most do not work in real time) and provide a powerful benchmark that has become an industry standard [22, 23]. They even provide a benchmark to compare various multiview algorithms [24]. A brief review of the results of a variety of algorithms on these benchmarks illustrates rather clearly that image based reconstructions, in general, provide far more accurate results than the other two approaches. Image based approaches make fewer explicit assumptions about the object to be imaged and therefore tend to be more accurate. Unfortunately, image based reconstructions tend to be much slower.

Teleimmersive systems, in general, have focused almost entirely on the speed of the system and have not focused on the accuracy of the reconstruction. In this paper, we describe a teleimmersive system that employs image based reconstruction to get accurate results, but does not sacrifice speed in order to arrive at these results. We compare our approach against the various aforementioned benchmarks to illustrate the strength of our method. Moreover, in contrast to most other papers describing image based teleimmersive systems, we focus on developing a method to integrate and blend the generated depth maps from the various views in order to improve the final visual quality.

3. SYSTEM OVERVIEW

The teleimmersion (TI) system presented in this paper allows for collaboration between geographically distributed users by creating a shared virtual environment. All users interact with the virtual environment via their local TI stations (Figure 2). In order to properly model interaction between objects in the shared virtual environment and allow for flexibility in the visualization of these objects, each station must maintain a local copy of the entire virtual space. Model manipulation and post-processing of data can then be performed locally. With these requirements in mind, each station must perform the following three tasks: (a) computation of a 3D reconstruction of local objects, (b) communication of 3D data to other stations, and (c) visualization of the virtual environment with other remote users.

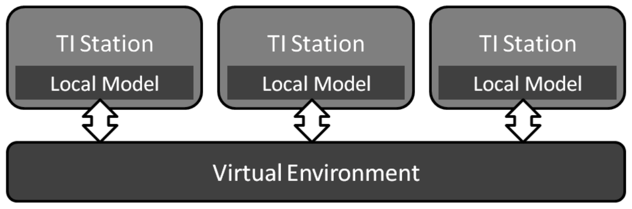


Figure 2: Diagram of the system which depicts the local model maintained by each station and the shared virtual environment.

An appropriate choice of representation for the data should take into consideration the aforementioned tasks. Namely, the system should be able to reconstruct any object that is present at each station. Hence, no priors on the physical models are assumed (i.e. no avatars to be fitted). Since we require real time streaming of 3D data, we must choose a flexible representation that allows for fast and efficient compression. Though each station creates a separate 3D model of an object based on views from multiple camera clusters (see section 4), in order to visualize the objects we must be able to integrate these views, which can be expensive if the representation choice is poor.

With these requirements in mind, we argue that the ideal representation of an object from a view is a mesh as opposed to a set of scattered points without connectivity information. Mesh generation in real time can be a challenging task; nevertheless, in this paper we describe how to construct a mesh rapidly, how the mesh allows us to build dense 3D depth maps, how the mesh allows us to perform efficient data compression, and how to combine the meshes from various views to improve the resulting data quality.

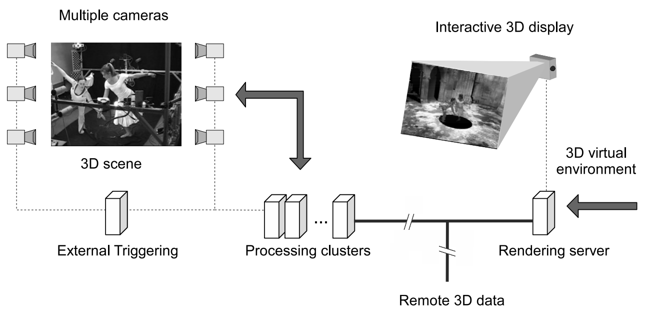


Figure 3: Components of the teleimmersion system. Capturing component is displayed to the left. Data is then processed using a computing cluster. The model is transmitted over the network for display at another teleimmersive station.

4. HARDWARE OVERVIEW

In this section we describe different components of the teleimmersion system (figure 3).

4.1 System Configuration and Cameras

The teleimmersion apparatus consists of 48 Dragonfly cameras (Point Grey Research Inc, Vancouver, Canada), each with resolution 640 × 480 pixels. The cameras are arranged in twelve clusters of three grayscale cameras for stereo reconstruction and a color camera for texture acquisition. Ten of the clusters are equipped with 6 mm lenses while the remaining two clusters have 3.8 mm lenses for wider capture space. The cluster thus cover 360° of the workspace of about 2.0 x 2.0 x 2.5 m^3. The cameras of each cluster are connected through IEEE 1394a (FireWire) interface to a dedicated server which performs image acquisition and stereo reconstruction. The server computers used for the reconstruction have two dual core Intel Xeon 2.33 GHz, 2 GB of memory and 1 Gbps connection to Internet 2. The servers use Windows XP operating system.

4.2 Synchronization

Synchronization of the images across the cameras, which is required for consistent 3D reconstruction from multiple views, is achieved using external triggering. The camera triggering pins are connected to the parallel port of the triggering server which generates a digital signal to initiate capturing of the next image frame while receiving TCP/IP messages from the cluster computers once the reconstruction of the frame has been completed.

4.3 Illumination

The workspace is illuminated by eight 55 W light fixtures with diffusive filters. The filters create diffusive illumination to reduce shadows and specularities which can interfere with stereo reconstruction. Six lights are mounted from the top to illuminate the subject from above and two are located on the floor illuminating the user's lower half of the body.

4.4 Display

The display system consist of a rendering computer with Intel Dual Core CPU, 2.66 GHz, 2 GB of memory, and two NVIDIA GeForce 8800 GTS graphics cards. The renderer can receive compressed 3D data directly from the cluster computers in separate network streams or indirectly through a gateway computer which can also connect to a remote site. The renderer supports different display options, such as single or multiple desktop displays and various passive or active stereo systems. The users can also use different interface devices (e.g. wireless 3D mouse, Wii Remote) to interact with the virtual environment.

5. CALIBRATION

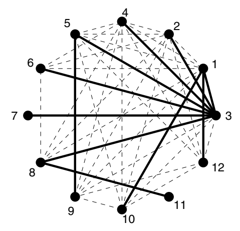
In our multi-camera system, 3D reconstruction is performed by several camera clusters that work independently to acquire partial 3D data of the scene. The cameras must be calibrated with respect to their intrinsic and extrinsic parameters with high accuracy in order to reconstruct an accurate 3D model of the user. In order to guarantee a realistic experience of teleimmersion, the metrics and geometry of the interaction space among remote users must also be properly established. For example, a local participant should perceive spatial relationships with other remote users in a fashion similar to real-life experience.

We approach the calibration of the teleimmersion space in a hierarchical fashion. On the lowest level, we calibrate internal camera parameters using Zhang's method [28] through homography obtained from a checkerboard. Since the cameras of each cluster share a large workspace, we can simultaneously obtain their geometric relationship. Once each cluster is internally calibrated, we can proceed with the external calibration of clusters which returns accurate positions and orientations of each cluster with respect to a reference camera. The external calibration is performed using a virtual calibration object which is generated by two LEDs at a fixed distance. Finally, we perform calibration of the workspace with respect to the display which is used to define the spatial relationship between remote locations. The algorithm and comparison with similar methods is explained in more detail in *[Reference removed for review]*. In this section we briefly outline the steps of the calibration method and provide the results for our setup.

5.1 Cluster Calibration

Calibration of each stereo cluster with four cameras is performed using Zhang's method. A planar checkerboard target is placed in different positions and orientations to generate a set of points for homography calculation. We use the standard pinhole camera model with distortion correction which describes the camera by 8 parameters in total. The algorithm uses a set of known points detected from the corner features of the checkerboard. The camera captures the checkerboard in different positions and orientations while projecting the points to the image plane. A system of linear equations is formed and solved through singular value decomposition. The internal camera parameters (i.e. focal length, optical center and distortion) obtained from the homographies are further optimized using the Levenberg-Marquardt [16] algorithm.

After each camera is calibrated independently, the relative orientation and position of the cameras within the cluster is obtained with respect to the middle camera chosen as the reference. Finally, global optimization is performed for all the cameras in the cluster to further reduce the reprojection error and optimize the geometric parameters. For the calibration of the stereo clusters with 4 cameras, we used a black and white checkerboard with 15 x 10 squares (square size was 40 mm). We typically collect about 15-20 images for each cluster. The calibration software written in C++ using OpenCV and levmar LM algorithm [16] libraries performs the calibration in about one minute. Typical reprojection errors for a cluster are between 0.08 and 0.15 pixels for cameras with the resolution of 640 by 480 pixels.



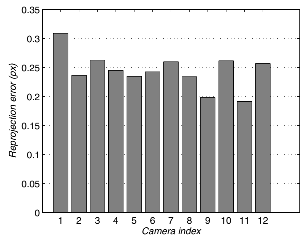


Figure 4: Vision graph generated for 12 stereo clusters with the cluster #3 selected as the reference cluster (left) and typical reprojection error for the external cluster calibration (right).

5.2 External Calibration

In our external calibration method using a virtual calibration object, we combine the idea of vision graphs to calibrate cameras with small overlap. In contrast to other methods [4, 8, 26] our approach resolves Euclidean reconstruction (preserving metric information) and introduces novel parameter reduction in the case of two-point bar calibration which increases the robustness of the calibration [14].

Our external calibration algorithm consists of the following major steps:

(a) image acquisition and sub-pixel marker detection on multiple cameras
(b) composition of adjacency matrix for weighted vision graph describing interconnections between the cameras (e.g. number of common points)
(c) computation of fundamental **F** and essential matrix **E** with RANSAC
(d) essential matrix decomposition into rotation and translation parameters defined up to scale factor λ

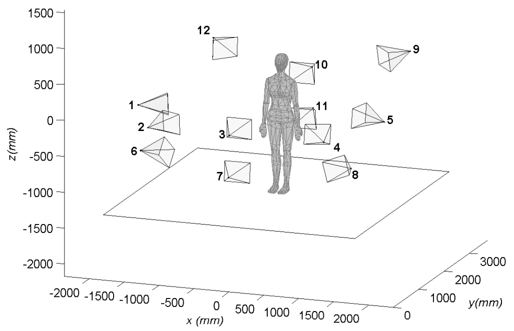


Figure 5: Three-dimensional layout of 12 stereo clusters after the calibration.

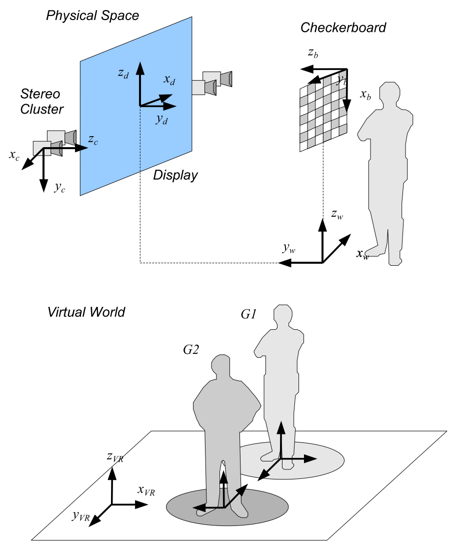


Figure 6: Calibration of the physical space (top). Example of remote users in a virtual space after physical calibration (bottom).

(e) determination of the scale factor λ through triangulation and LM optimization
(f) optimal path search using Dijkstra algorithm
(g) global optimization of the parameters using sparse bundle adjustment (SBA) [16]

Our external calibration approach assumes that at least any two given clusters overlap. Marker detection is implemented using color and ellipse detection in background subtracted images to allow calibration in daylight conditions. Based on captured marker positions, pair wise geometric relationship is established between cameras with large number of common points using fundamental matrix. The transformations can be solved with the metric scale factor since the distance between the LEDs is known. Global calibration is then resolved by constructing a vision graph which models the overlap between the stereo clusters. Weight factor, representing the reciprocal of the number of common points between the camera pairs, is assigned to each connection in the vision graph. Optimal transformation path (i.e. the path with the lowest sum) from each camera to the reference camera is determined. Finally, the parameters are globally optimized using sparse bundle adjustment implementation to further reduce the reprojection errors on each camera.

For external calibration of our 48 camera setup we used a rigid metal bar with two LED markers attached on each end. We chose Luxeon I LED (Philips Lumileds Lighting Company, San Jose, CA), with a brightness of 30.6 lm and 160° emitting angle. The two LEDs were placed on a metal bar at distance of 298 mm. The complete external calibration of 12 stereo clusters with about 3000 3D points, takes between 10-15 seconds on a personal computer with Intel Xeon 3.20 GHz processor and 1 GB of memory. The mean reprojection error between all the cameras is typically between 0.25 and 0.40 pixels with the standard deviation between 0.04 and 0.12 pixels (figure 4). In our setup, these errors result in the cluster position errors of about 0.5% and orientation errors of about 0.1°. Figure 5 shows the results of the external calibration where for clarity only the central camera of each cluster is presented. Camera 3 was chosen as the reference camera since its position and orientation correspond with the floor level.

5.3 Calibration of Physical Space

In the last part of this section, we address geometric calibration between remote locations. We propose a simple method for calibrating the reference cluster (i.e. camera space) to the physical space of each teleimmersion station. By consistently defining the physical coordinates between the remote locations, the geometric correspondence across the sites is well-defined.

The calibration is performed by acquiring one image of the checkerboard placed in the vertical position (figure 6). Since the reference camera has already been internally calibrated, it is possible to determine the exact location of the camera with respect to the checkerboard coordinate system. The physical coordinate system is placed on the floor to retain consistency of coordinates between different locations. If all remote locations are properly calibrated by defining the transformation between the camera and the physical world space, their rendering coordinate systems coincide with the coordinate system defined in the virtual environment.

6. STEREO RECONSTRUCTION

In this section, we briefly describe the construction of the 3D data from pairs of images, the compression of the created data, and the result of combining various reconstructions together in a single environment (figure 7). We focus on comparing our results on the aforementioned benchmarks.

6.1 Construction of the Representation

In this section, we describe how we represent the data and how we build rapid and accurate 3D depth maps. We partition the image domain using Maubach's bisection scheme

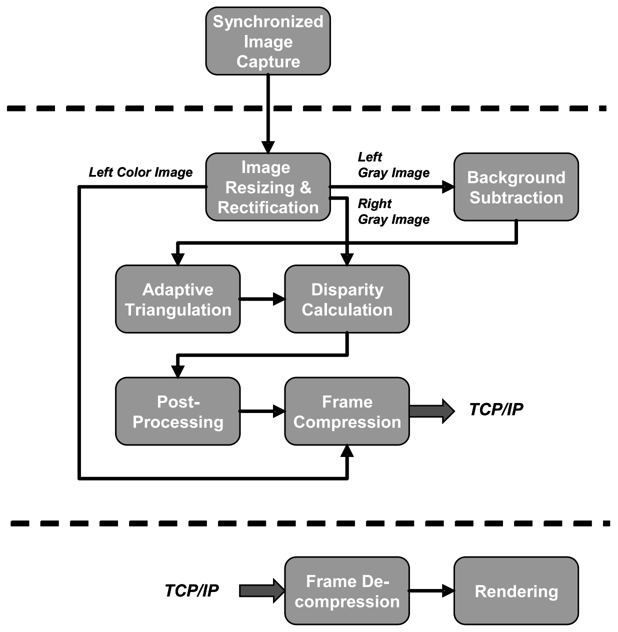


Figure 7: A simplified block diagram of the stereo algorithm for the 3D mesh generation.

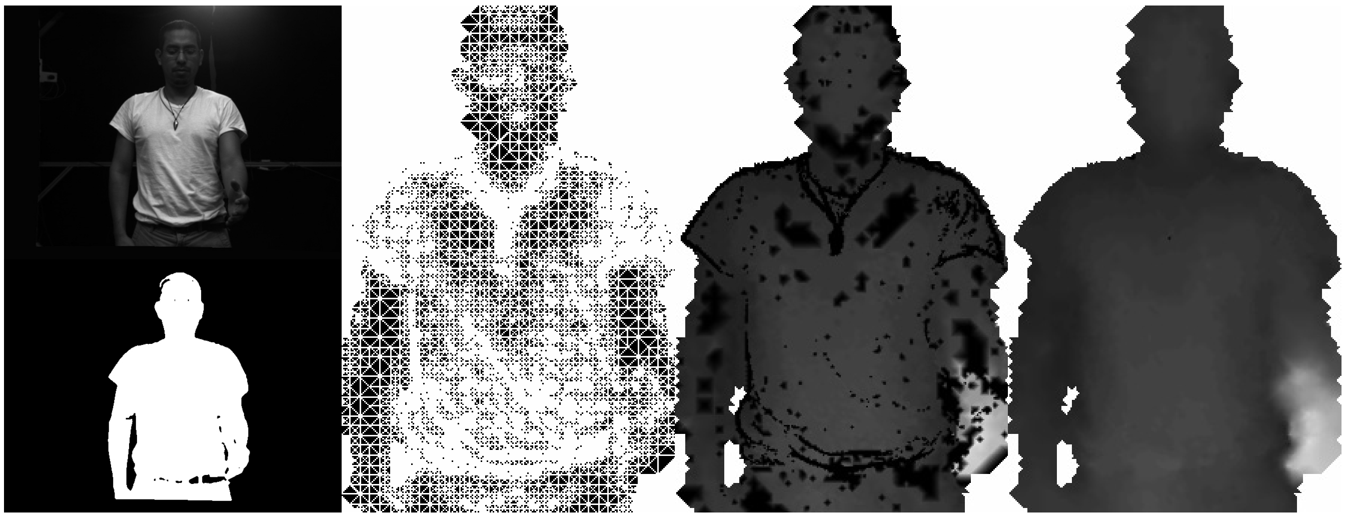


Figure 8: A 320 × 240 image taken from a single camera in a stereo cluster (top-left), a background subtracted image generated for that image (bottom-left), the mesh generated for this image (second image from the left), the pre-processed disparity image (third image from the left), and the post-processed disparity image (right image). Note that lighter gray values indicate that the object in the scene is closer, darker gray indicate that an object in the scene is further away, and black indicates areas of uncertainty.

[17]. Decomposition begins with a coarse representation of the image by right isosceles triangle basis functions. Each triangle is then refined via a simple bisection. Refinement occurs only if a given criterion goes unsatisfied within each triangular region. This criterion, in our system, is the variance of the grayscale image within each triangle. We introduce an additional constraint to allow for the application of a variety of standard Finite Element Methods: we require that there be no floating nodes (i.e. no nodes in the middle of a triangles edge). This type of mesh is referred to as conforming and aids in the development of algorithms to quickly process the depth maps created by our reconstruction algorithm.

6.2 Calculation of Depth

After the construction of this representation of the image, we can calculate the depth at the nodes of the triangles by employing a region by region matching technique. Since window-based stereo aggregation methods implicitly assume that all pixels within the window have similar disparities, they struggle whenever windows straddle depth discontinuities. This results in the infamous foreground fattening effect.

We assume depth varies smoothly within any image segment with homogeneous color. Fortunately, our mesh employs an identical assumption during construction. The size of the image segment generated by the meshing dictates our stereo aggregation window size choice, since all elements in an image segment have similar depth. We do not assume that pixels in the same segment share the same depth, but rather that they lie in a locally fitted surface. This method succeeds in our system for two reasons: first, it improves the robustness of our matching procedure by employing entire regions instead of single points and, second, it reduces the total number of points that must be matched which improves the overall efficiency of the matching procedure.

Finally, since the representation is conformal, the depth map can be post-processed by exploiting standard global optimization procedures such as anisotropic diffusion, which have been proven to improve the overall quality of depth reconstruction [5]. Though these finite element methods generally converge slowly, they are proven to converge rapidly in our representation. In figure 8, we show a sample image from our system (top-left image), the background subtracted image (bottom-left image), the mesh generated for this image (second image from the left), the pre-processed disparity image (third image from the left), and the post-proccessed depth map from that view (the right image).

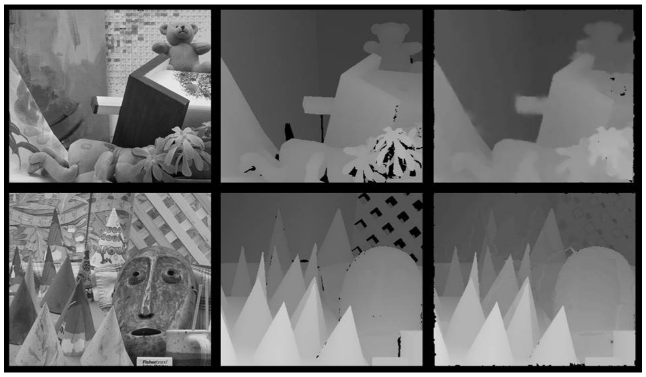


Figure 9: Two images, each of size 450 × 375, from the benchmark developed by Scharstein et al. [23] (left column), the ground truth for these two images produced using a laser range finder (center column), and the output of our stereo algorithm (right column). Note that lighter gray values indicate that the object in the scene is closer, darker gray indicate that an object in the scene is further away, and black indicates areas of uncertainty.

Process	Our's	Wang	Bleyer	Klaus
Teddy 1-Pixel Error	7.15%	8.31%	6.54%	7.06%
Teddy Speed	42.1ms	20s	100s	14s
Cone 1-Pixel Error	7.56%	7.18%	8.62%	7.92%
Cone Speed	53.8ms	20s	100s	25s

Table 1: A quantitative comparison of our algorithm against the top performers on the benchmark developed by Scharstein et al. [23]. The teddy and cone image correspond to the top and bottom rows of figure 9 respectively. Our output was produced with approximately 40,000 triangles in both instances.

At this point, we can compare the effectiveness of our algorithm in calculating disparities on the traditional aforementioned benchmarks. The benchmarks consist of dozens of pictures. The two images that the benchmark has identified as the most difficult are found in the left column of figure 9. The accuracy of the measurements is calculated against a ground truth image, which can be found in the center column of figure 9, generated by a laser range finder. Note that lighter gray values indicate that the object in the scene is closer and darker gray indicates than an object in the scene is further away. Black areas correspond to points where the disparity value is unknown.

The output of stereo algorithms is generally what is called a disparity value for each pixel in the image. The disparity is inversely proportional to the depth, and error, in this domain, is generally calculated by the percentage of pixels that differ in their returned disparity from the ground truth by more than one. Put more clearly, this is approximately the number of pixels that differ in their returned value by more than an order of magnitude greater than a single centimeter. The output of our stereo algorithm on the images found in the left column of figure 9 calculated on two dual core 2.33 GHz machines can be found in the right column of the same figure. A quantitative comparison of our algorithm can be found in table 1. We include the most accurate performers on this benchmark in the same table. Wang et al. employed a dual core 1.6 GHz machine [27], Bleyer et al. employed a 2 GHz Pentium 4 machine [3], and Klaus et al. employed a dual core 2.21 GHz machine [12]. Notice that we arrive at comparable levels of accuracy with the top performers, but our algorithm takes anywhere between three hundred to two thousand times less time to produce an answer.

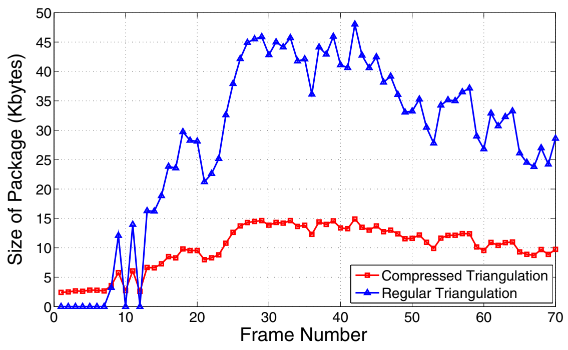


Figure 10: A comparison of package size between the standard and our encoded format. Packages include information on how to generate the representation, RGB values (3 bytes per node), and disparity values (2 bytes per node).

Triangulation	3.83 ms
Disparity	15 ms
Post-Processing	1.78 ms
Decompression	0.78 ms
Total	21.39 ms

Table 2: Average frame rate for a typical image sequence in the TI system on two dual core 2.33 GHz machines obtained using TI stereo pairs each with size 320×240 with approximately 10000 triangles per frame.

6.3 Compression of the Representation

After constructing the 3D data from a single view, the data must be passed to the rendering machine. Sharing the models between different stations requires transmitting the triangulation models in real time. A standard format for transmitting triangulation information consists of transmitting nodal values, and then specifying the triangles based on the node indices. In particular, if we consider RGB values (3 bytes) and depth values (2 bytes) per pixel, we have 5 bytes per node. Each triangle must specify 3 vertices and each vertex requires at least 3 bytes (since there are more than 2^{16} possible nodes in our representation). Hence, there are a total of 9 bytes per triangle. This gives:

$$\begin{aligned}\text{Std. Package Size} &= (\text{No. of Nodes}) \times 5 \\ &\quad + (\text{No. of Triangles}) \times 9.\end{aligned} \tag{1}$$

Given our choice of representation, we can do better. Since our triangulation results from a bisection scheme, it is possible to specify how the representation was generated instead of specifying each node in the triangle. That is, we can specify which triangles are bisected. This encoding scheme for the triangulation yields large gains given enough triangles in the representation. No additional time is required to encode the representation since the information for encoding the representation is generated at the same time the representation is initially computed.

In figure 10, we compare the size of the data package generated over a typical image sequence after background subtraction from the TI system (each image has size 320×240). Typical images in the sequence look similar to the left images in figure 8. In the sequence used for figure 10, no object is present in the field of view of the camera for the first seven frames during which time the compressed package is larger than the standard package, but from then on the compressed package is close to 3 times smaller than the standard package. Notice, each frame is less than 15 Kbytes. In table 2, we include a quantitative description of the average time per frame to generate and decompress the 3D data for the typical image sequence from figure 10 (note, we have not included the first several frames which had no object in the scene while calculating the average time).

6.4 Integration of the Views

After constructing the 3D model from each view, we must project the various views together in the global 3D coordinate frame. Again, we first illustrate the procedure on a benchmark and then data collected from the TI system.

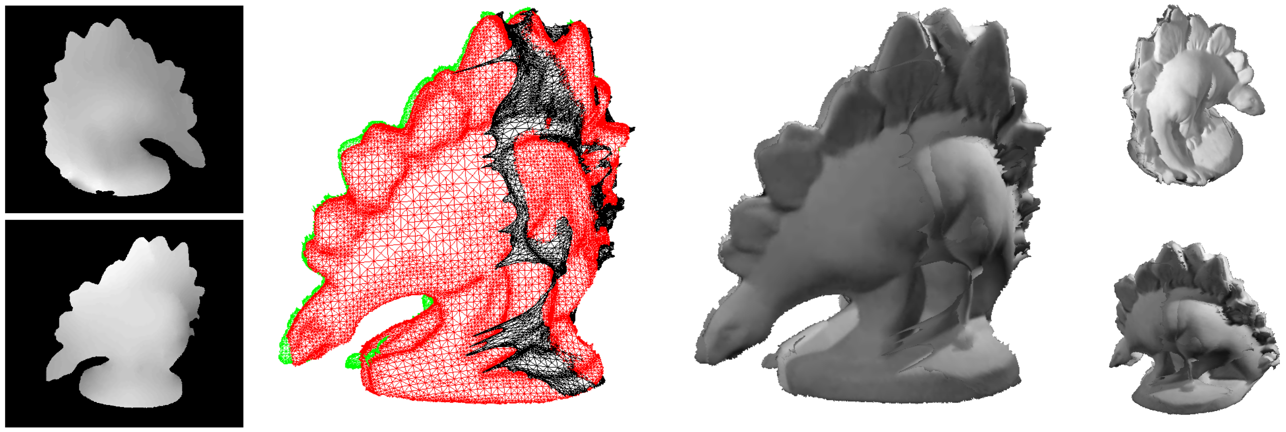


Figure 11: Creating a 3D model: Sample disparity maps obtained from stereo reconstruction (left). Disparity values are used to project the representation into 3D space which becomes patches for the 3D model (middle-left). The model with texture mapping is presented from multiple views (middle-right and right).

We consider a benchmark released by Seitz et al. [24]. The dataset includes a sequence of images of a dinosaur, figure 12, collected at forty eight different positions on an extremely well-calibrated multi-camera system. The system is not only geometrically calibrated (i.e. intrinsic and extrinsic calibration), but is also photometrically calibrated. The result of our algorithm from three different views can be found in figure 11. The left image in the figure corresponds to depth maps produced from two different views. The center image corresponds to the meshes projected with their depth values into a global coordinate system. Notice the meshes are well aligned. Finally, notice the images on the right correspond to a variety of views with the colors from the views projected on top of the mesh.

We repeat the process on two different sets of images (each containing three views) from the teleimmersive system. Figure 13 shows two models constructed for individuals that are not collocated in the same setup. Observe that though the patches are well aligned in Figure 13, they do not perfectly overlap. This occurs in occluded parts of the scene for a particular view; hence the patches that do not overlap occur in areas where the patch's normal does not point in the same direction as the view that generated it. In addition, since the system is not photometrically calibrated different cameras may treat the same color in the scene as different colors in their representation. In order to alleviate both these problems, we rely on the visualization algorithm, which we present below.

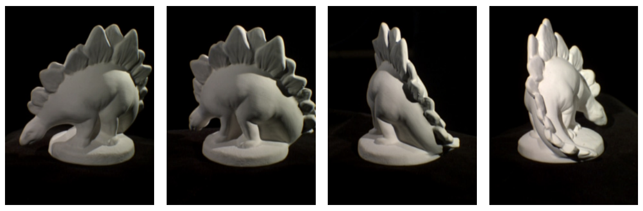


Figure 12: Dinosaur picture samples from an online dataset [24].

7. 3D VIDEO PLAYBACK

The final component of the TI system deals with the visualization of the 3D data streams or 3D video streams, and their integration into the shared virtual environment.

7.1 Display System

In light of the required visual processing, we make use of a fairly state-of-the-art graphics acceleration board (see Sec-

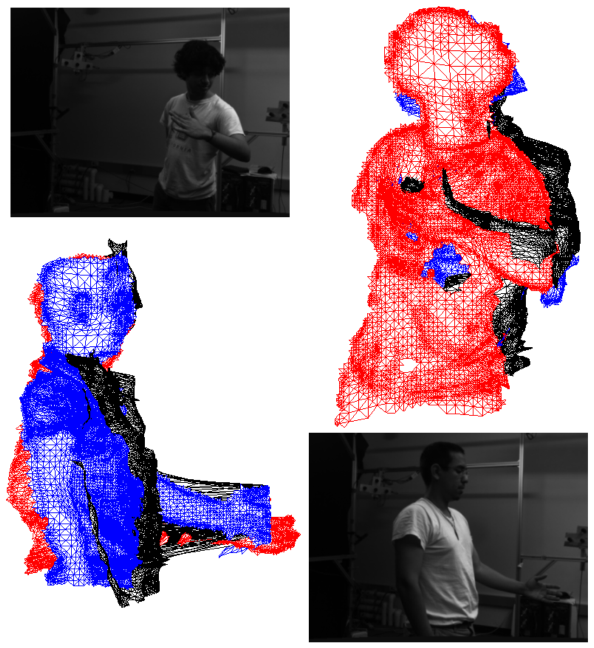


Figure 13: 3D models recovered from our tele-immersion environment: Views of individuals in station where they are interacting with virtual data (top-left and bottom-right), and corresponding models shown as multiple colored meshes (top-right and bottom-left).

tion 4.4) here. The subsystem is the main user-facing component and provides for the connection to remote streams provided by other TI stations. Stream management features are provided for manipulation of the visual representation such as hiding or showing all or parts of streams from remote stations. Users can also freely reposition and reorient the reconstructed 3D objects within the virtual environment. Incoming streams can be recorded and later played back with full control over playback speed, pausing and seeking to random location in the stream. In this way, for example, a complicated dance lesson can be studied at leisure and from arbitrary viewpoints.

In order to provide these capabilities at real time frame rates, the processing of connected streams has been optimized to reduce latency and make use of all computing resources. We achieve this through aggressive parallelization and appropriate buffering to separate the processing tasks. In particular, care has been taken to never stall the rendering process as unsteady frame rates quickly result in user motion sickness when using virtual reality displays.

The display subsystem is responsible for maintaining the local representation of the shared virtual environment and renders the appropriate elements in addition to the 3D video streams. Correspondingly facilities are provided for navigating the environment and manipulating it depending on the collaborative application.

7.2 Visual Representation of Reconstructions

The most challenging task of the display subsystem is the proper rendering of the compressed reconstructions. More specifically, the requirements are first, real time frame rates (60 frames per second are needed for virtual reality displays), and second, a high quality depiction that is visually consistent over the duration of playback.

The representation of the reconstruction provides typical data for which current graphics hardware is optimized. At first glance, the built-in standard z-buffer rasterization would seem sufficient. This is not the case and the playback may not produce well defined surfaces and suffers from temporal popping from tessellations overlapping the same captured surface. We relate these issues to the following characteristics of the data:

1. **Noise** in the reconstruction of a surface produces slight variations for each reconstruction independently. Even assuming a perfect spatial registration between the reconstructions, the noise would produce spot-wise occlusions at overlapping regions, as discussed in the previous section.

2. **Registration** is not perfect and may produce offset reconstructions of overlapping regions of a same captured surface (illustrated in figure 14).

3. **Color** may be slightly mismatched between the capturing clusters because photometric calibration is not done. In combination with the first point, the mismatch may render the inconsistent occlusion visible.

4. **2.5D** reconstructions are generated for each cluster, i.e. the points on the image plane are extruded to the computed depth. The reconstruction provided by a cluster is, thus, sensitive to the orientation of the surface with respect to the cluster's image plane.

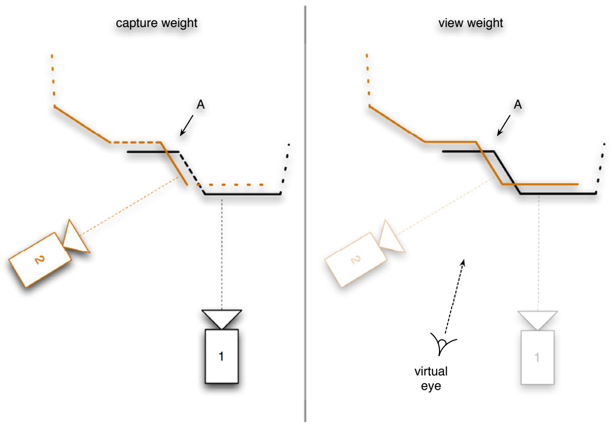


Figure 14: The contribution of each triangle is determined by: Left, how well the corresponding capturing camera is able to resolve that triangle; Right, how much that triangle is likely to contribute to the final image given the current view parameters. Filled lines represent important triangles versus unimportant ones which are depicted as dashed lines.

Our approach to the visual representations addresses these issues by (a) leveraging the built-in functionality of the graphics hardware to rapidly project the triangles and compute screen coverage in real time and (b) composite the individual contributions of all the reconstructions at each screen pixel to obtain better image quality.

In the first phase of the rendering process the individual contributions for each reconstruction is computed with respect to the current view. For this purpose each reconstruction is rendered from the view point into an off-screen buffer using a custom fragment program. For each covered fragment we generate the usual color and depth contribution but additionally compute and store a weight factor for use during compositing. This factor considers the normals of the triangle faces of the reconstruction. We determine the appropriateness of each triangle as illustrated in figure 14.

- **Capture Weight** (figure 14 left): We consider the orientation of each triangle in a reconstruction with respect to the capturing direction of the corresponding camera cluster. In practice, this means that we compute a face normal for each triangle (in a geometry program on the GPU), compute the dot product with the normal of the cluster's image plane, and floor the result to zero. Thus, triangles directly facing the capturing cluster receive a high weight, and as the triangles face away from the camera the weight drops towards zero at the orthogonal orientation. This first weight builds on the employment of many capturing clusters in the environment with many overlapping regions: at overlaps the better resolved triangles amongst the clusters is be favored.

- **View Weight** (figure 14 right): Similar to the capturing weight, we consider the current viewing parameters to determine interesting parts of the reconstruction with respect to the current view. Again the dot

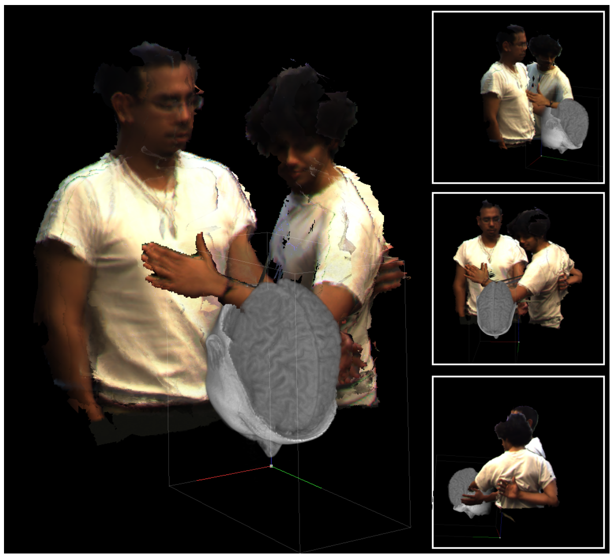


Figure 15: Different views of the visualization generated from the 3D data in figure 13 interacting with the virtual environment.

product is taken, but here it is calculated between the normal of each triangle and the viewing direction.

The two weights are multiplied together to form the final weight of the contribution. In the simplified example illustrated in figure 14, the viewing weight is equal to one at point A, thus, the capturing weight selects the better resolved triangles amongst the two clusters (filled segments). At the edges of the surface though, the triangles are both captured poorly and are therefore uninteresting for the view and have diminished contributions.

In the second phase all the pixels of the screen are triggered and the contributions are composited independently. Care has to be taken not to blend all the stored contributions to retain proper occlusion, e.g. a hand in front of a torso. Our current implementation determines the closest contribution and blends it with those within a user specified depth range. An example of the type of rendering generated for the 3D data from figure 13 can be found in figure 15

8. CONCLUSIONS

In this paper, we described a multi-camera system that creates a highly accurate, 3D reconstruction of an environment in real time. We began by providing an overall systems perspective and then described in detail a calibration, representation, reconstruction, and then visualization methodology that together provide a state of the art teleimmersive system.

The hierarchical calibration approach allows for robust and flexible calibration of multiple cameras with various pairwise overlap. Since the cameras within the cluster are located close to each other, the homography based method is more appropriate as compared to the fundamental matrix approach. On the other hand, for more sparsely positioned stereo clusters fundamental matrix approach yields more accurate results for geometric calibration. Using proposed hierarchical approach, it is convenient to recalibrate only part of the camera system, internally or externally.

The representation and reconstruction of 3D data is simultaneously flexible and accurate which allows for high levels of compressibility and easy visualization. The reconstruction is amongst the top performers on an industry wide bencmark for accuracy and it is easily one of the fastest reconstructions available. Future work will focus on employing the reconstruction to build a single unified model of the 3D environment from the various views as opposed to the current strategy of simply relying on the visualization technique to correct for possible inconsistencies amongst the various views. This would have the added benefit of reducing the final size for the entire 3D environment, which would have the benefit of reducing the overall network bandwidth required for a particular teleimmersive station.

Finally, the visualization technique takes employs the nature of the representation and the viewing direction to build high quality depication that is visually consistent. The technique works rapidly by taking advantage of standard graphics hardware.

Full-body 3D reconstruction of users in real time offers new possibilities for immersive and teleimmersive applications. Within our framework, the users are integrated into the virtual environment and become a part of the model. Different digital effects (e.g., transformations and deformations of rendered images) can be applied in real time to manipulate what is displayed to the remote users. The users can be immersed inside computer generated existing or non-existing environments, such as ancient buildings and future architectural designs to allow interactive exploration. The system could be combined with head-mounted display and head tracking to provide first person immersion. The presented system introduces new opportunities for education (e.g., immersive physics experiments) and training (e.g., emergency and disaster response, military training). The 3D capturing framework presented can also provide data for human motion analysis and modeling. Extracted kinematic parameters could be applied as on-line feedback to a user for training of physical movements (e.g., dancing, physical therapy, and exercise). Finally, there are many applications in social networking and entertainment where the users could interact in real time inside a common virtual environment, such as games supporting physical interaction, interactive music video, and 3D karaoke.

9. ACKNOWLEDGMENTS

This research work was partially funded by NSF grants 0703787 and 0724681 and HP Labs.

10. REFERENCES

[1] J. N. Bailenson, K. Patel, A. Nielsen, R. Bajcsy, S. Jung, and G. Kurillo. The effect of interactivity on learning physical actions in virtual reality. *Media Psychology*, page In Press, 2008.

[2] H. Baker, D. Tanguay, I. Sobel, D. Gelb, M. Gross, W. Culbertson, and T. Malzenbender. The coliseum immersive teleconferencing system. In *Proceedings of International Workshop on Immersive Telepresence, Juan-les-Pins, France*, 2002.

[3] M. Bleyer and M. Gelautz. A layered stereo matching algorithm using image segmentation and global visibility constraints. *ISPRS Journal of Photogrammetry and Remote Sensing*, 59(3):128–150, 2005.

[4] X. Cheng, J. Davis, and P. Slusallek. Wide area camera calibration using virtual calibration objects. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2000)*, 2000.

[5] P. Favaro, S. Osher, S. Soatto, and L. Vese. 3d shape from anistropic diffusion. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, 2003.

[6] M. Gross, S. Würmlin, 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. *ACM Trans. Graph.*, 22(3):819–827, 2003.

[7] J. Hasenfratz, M. Lapierre, and F. Sillion. A real-time system for full-body interaction with virtual worlds. In *Proceedings of Eurographics Symposium on Virtual Environments*, pages 147–156. The Eurographics Association, 2004.

[8] I. Ihrke, L. Ahrenberg, and M. M. Magnor. External camera calibration for synchronized multi-video systems. In *Proceedings of 12th International Conference on Computer Graphics, Visualization and Computer Vision 2004*, volume 12, pages 537–544, Plzen, Czech Republic, February 2004.

[9] S. Jung and R. Bajcsy. A framework for constructing real-time immersive environments for training physical activities. *Journal of Multimedia*, 1(7):9–17, 2006.

[10] P. Kalra, N. Magnenat-Thalman, L. Moccozet, G. Sannier, A. Aubel, and D. Thalman. Real-time animation of realistic virtual humans. *IEEE Computer Graphics and Applications*, 18(25):42–56, 1998.

[11] T. Kanade, P. Rander, S. Vedula, and H. Saito. *Mixed Reality, Merging Real and Virtual Worlds*, chapter Virtualized reality: digitizing a 3D time varying event as is and in real time, pages 41–57. SpringerVerlag, 1999.

[12] A. Klaus, M. Sormann, and K. Karner. Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure. In *International Conference on Pattern Recognition*, volume 2, 2006.

[13] G. Kurillo, R. Bajcsy, K. Nahrstedt, and O. Kreylos. Immersive 3d environment for remote collaboration and training of physical activities. In *Proceedings of IEEE Virtual Reality Conference (VR 2008)*, pages 269–270, Reno, NV, March 8-12, 2008 2008. Accepted for Poster Presentation.

[14] G. Kurillo, Z. Li, and R. Bajcsy. Wide-area external multi-camera calibration using vision graphs and virtual calibration object. In *Proceedings of 2nd ACM/IEEE International Conference on Distributed Smart Cameras (ICDSC Š08)*, Stanford, CA, September 7-11 2008. IEEE.

[15] A. Ladikos, S. Benhimane, and N. Navab. Efficient visual hull computation for real-time 3D reconstruction using CUDA. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, 2008. CVPR Workshops 2008*, pages 1–8, 2008.

[16] M. Lourakis. levmar: Levenberg-marquardt nonlinear least squares algorithms in C/C++. [web page] http://www.ics.forth.gr/∼lourakis/levmar, July 2004.

[17] J. Maubach. Local Bisection Refinement for N-Simplicial Grids Generated by Reflection. *SIAM Journal on Scientific Computing*, 16:210, 1995.

[18] J. Mulligan and K. Daniilidis. Real time trinocular stereo for tele-immersion. In *Proceedings of 2001 International Conference on Image Processing, Thessaloniki, Greece*, pages 959–962, 2001.

[19] K. Nahrstedt, R. Bajcsy, L. Wymore, G. Kurillo, K. Mezur, R. Sheppard, Z. Yang, and W. Wu. Symbiosis of tele-immersive environments with creative choreography. In *ACM Workshop on Supporting Creative Acts Beyond Dissemination, Associated with 6th ACM Creativity and Cognition Conference*, Washington D.C., June 13-15 2007.

[20] D. Nguyen and J. Canny. Multiview: spatially faithful group video conferencing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 799–808. ACM, New York, NY, 2005.

[21] K. Patel, J. N. Bailenson, S. Hack-Jung, R. Diankov, and R. Bajcsy. The effects of fully immersive virtual reality on the learning of physical tasks. In *Proceedings of the 9th Annual International Workshop on Presence, Ohio, USA*, pages 87–94, 2006.

[22] D. Scharstein and R. Szeliski. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. *International Journal of Computer Vision*, 47(1):7–42, 2002.

[23] D. Scharstein, R. Szeliski, and M. Coll. High-accuracy stereo depth maps using structured light. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings*, volume 1, 2003.

[24] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Int. Conf. on Computer Vision and Pattern Recognition*, pages 519–528, 2006.

[25] R. Sheppard, M. Kamali, R. Rivas, M. Tamai, Z. Yang, W. Wu, and K. Nahrstedt. Advancing Interactive Collaborative Mediums Through Tele-Immersive Dance (TED): a Symbiotic Creativity and Design Environments for Art and Computer Science. *ACM Multimedia 2008*, 2008.

[26] T. Svoboda, D. Martinec, and T. Pajdla. A convenient multicamera self-calibration for virtual environments. *Presence*, 14(4):407–422, 2005.

[27] Z. Wang and Z. Zheng. A Region Based Stereo Matching Algorithm Using Cooperative Optimization. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2008: Anchorage), Proceedings. Alaska.*

[28] D. Zhang, Y. Nomura, and S. Fujii. Error analysis and optimization of camera calibration. In *Proceedings of IEEE/RSJ International Workshop on Intelligent Robots and Systems (IROS Š91), Osaka, Japan*, pages 292–296, 1991.