

IMECE2005-81873

A DESKTOP NETWORKED HAPTIC VR INTERFACE FOR MECHANICAL ASSEMBLY

Abhishek Seth
Department of Mechanical Engineering
Virtual Reality Applications Center
Iowa State University
Ames, IA 50011
abhiseth@vrac.iastate.edu

Hai-Jun Su
Department of Mechanical Engineering
Virtual Reality Applications Center
Iowa State University
Ames, IA 50011
haijunsu@iastate.edu

Judy M. Vance
Department of Mechanical Engineering
Virtual Reality Applications Center
Iowa State University
Ames, IA 50011
jmvance@vrac.iastate.edu
ASME Fellow

ABSTRACT

This paper presents the development of a PC-based 3D human computer interface for virtual assembly applications. This system is capable of importing complex CAD (Computer Aided Design) models, rendering them in stereo, and implementing haptic force feedback for realistic part interaction in virtual environments. Such an application will facilitate wider acceptance of the use of a VR interface for prototyping assembly tasks. This interface provides both visual and haptic feedback to the user, while allowing assembly tasks to be performed on a desktop virtual environment. The network module has the ability to communicate with multiple VR systems (such as CAVE etc.) at geographically dispersed locations using a non-dedicated network channel. The potential benefits of such a system include identification of assembly issues early in the design process where changes can be made easily, resulting in a more efficient and less costly product design process.

INTRODUCTION

Analyzing and understanding complicated three dimensional (3D) Computer Aided Design (CAD) models using a two-dimensional desktop computer interface has always been a challenge for designers. Designers interact with complex 3D CAD models using two-dimensional devices like a

mouse and a keyboard while viewing the models using a flat computer screen. While this interface allows designers to verify geometric interferences in assemblies, the two-dimensional nature of such an interface makes it difficult to predict issues that arise when an assembly worker is instructed to assemble the parts for the first time.

Virtual Reality (VR) technology bestows the user with different kinds of sensations (visual, haptic, auditory etc.) to create a sense of presence in the virtual world. Jayaram et al. [1] defines the key elements of VR as “a) immersion in a 3D environment through stereoscopic viewing, b) a sense of presence in the environment through tracking of the user and often representing the user in the environment, c) presentation of information of the sense other than vision, and d) realistic behavior of all objects in the virtual environment.”

VR allows the users to interact with CAD models in a 3D environment with the same number of degrees of freedom as the environment in which the actual part exists in. Thus, VR can provide an interface to computers that is better suited for interacting with 3D models[2]. Recent advances in VR technology have enabled users to interact with virtual environments using haptics. Haptics is based on the principle of providing force cues to the user to create a sense of presence in a three dimensional environment. Using haptics, users can feel the difference between soft and hard parts, light and heavy parts and smooth and rough surfaces etc.

Such VR technology can be successfully applied for prototyping assembly operations in virtual environments. Kim and Vance [3] define virtual assembly, as the “ability to assemble CAD models of parts using a three-dimensional immersive, user interface and natural human motion”. Performing assembly/disassembly operations in virtual environments, different assembly sequences can be analyzed and problems in designs can be identified early in the product design process. This will result in shorter product development lifecycles, saving time, effort and money which would be expended in making changes in the later stages of product development. Also prototyping of assembly operations in a virtual environment is less costly than building physical prototypes. However, up until recently, the high cost of VR equipment has restricted widespread acceptance of such useful technologies.

The goal of the work presented here is to develop a low cost VR application that can perform multi-body collision detection and simulate part behavior while performing assembly in a virtual environment. The application is capable of providing haptic feedback and stereo vision to the user for interacting with complex digital models present in the virtual scene. Apart from this, the network communication module enables the application to connect with different types of VR systems (at geographically dispersed location) for demonstration and analysis of assembly sequences using a non-dedicated network channel. Thus, an engineer can conceptualize assembly sequences on his or her workstation and collaborate with people located at a remote VR facility.

BACKGROUND

Several attempts have been made to use VR technology for prototyping mechanical assembly operations in virtual environments. These applications can be described by several distinct features:

- *VR display environment* – desktop systems (with and without stereo viewing), projection screen systems
- *Part interaction methods* – collision detection only, constraint-based interaction modeling, physically based interaction modeling
- *Force feedback* – use of haptic devices
- *Networking multiple VR systems*

Gupta et al. [4, 5] developed a desktop virtual environment called VEDA (Virtual Environment for Design for Assembly) which uses physically based modeling (PBM) for modeling part behavior, dual PHANToM[®] haptic devices for force feedback interaction and auditory and stereo cues to augment part interaction. Coutee et al. [6, 7] developed a similar system for the desktop called HIDRA (Haptic Integrated Dis/Re-assembly Analysis). HIDRA uses the GHOST Software Toolkit from Sensable Technologies and two PHANToM[®] devices for simulating physical behavior of parts in a desktop virtual environment. Both VEDA and HIDRA are somewhat limited because of their inability to adequately handle complex CAD

models. Jayaram, et al. [8-11] have developed VADE (Virtual Assembly Design Environment) for performing virtual assembly. This application advanced the state-of-the-art by providing the ability to directly input and interact with Pro/E CAD files. Two-handed assembly, using CyberGloves, was also developed. Constraint-based methods for modeling part behavior, demonstrated the ability for parts to slide and rotate with respect to each other. Because VADE uses constraint-based interaction methods, reaction forces are not generated when objects collide with each other and therefore, no haptic interface is available. Bullinger et al. [12] developed an assembly planning system at Fraunhofer-Institute for Industrial Engineering (IAO) called VirtualANTHROPOS which uses ANTHROPOS, an anthropometric computer modeling software package, to place a virtual human in the assembly operation. Although, the application used Head Mounted Display (HMD) and Data Glove device for natural part interaction, it lacked in providing haptic feedback to the user.

Fernando[13] at University of Salford developed a virtual assembly application called IPSEAM (Interactive Product Simulation Environment for Assessing Assembly and Maintainability) that uses constraint based geometric modeling for interaction, however simulating part behavior is limited to lower pair joints interactions, such as constraints between surfaces, leaving out constraints involving vertices and edges. Also, there is no force modeling so haptic interaction is not present in the system.

Johnson and Vance [14] developed VEGAS (Virtual Environment for General Assembly), in 2001. Using Voxmap Point Shell (VPS) software from Boeing Corporation, users could assemble full scale models with high polygon counts. Collision detection was implemented, however, the program lacked any part behavior simulation and haptic interaction.

Kim and Vance [3, 15] investigated several collision detection and part behavior algorithms and further modified VEGAS to include physically based modeling to simulate part behavior in virtual environments. Though the application could handle large model data for collision detection and part behaviors, it did not support haptic interaction.

Kim and Vance [16] also developed NHE (Network Haptic Environment) to facilitate collaborative assembly through the internet. A combination of peer-to-peer and server-client architecture is developed to maintain the stability and consistency of the system data. However, the variety of computation capability of each node often causes inconsistency problem which produce unrealistic haptic forces. In addition, each network-node needs a dedicated PC for force rendering as well as a simulation machine for visualization using a projection screen VR system. Using two computers at one network-node made the system even more expensive and unaffordable.

Wan et al. [17] developed a multimodal CAVE-based virtual assembly system called MIVAS (A Multi-Modal Immersive Virtual Assembly System) at Zhejiang University. MIVAS used constraints for simulating part behavior in a

virtual environment. The application performed hand-to-part collision detection using VPS software while part-to-part collision detection was implemented using RAPID. The users can feel the size and shape of digital CAD models using the CyberGrasp haptic device from Immersion Corporation. Since Haptic feedback was only provided in gripping tasks, the application lacked in providing force information when parts collided.

While all of these approaches have advantages and disadvantages, none provide the type of assembly environment that will significantly change the way assembly methods are currently evaluated. Our intent is to develop and evaluate a system that spans various levels of virtual reality hardware from desktop to full immersion to explore how all of these different VR interfaces might be used together to improve the design process.

HARDWARE AND SOFTWARE USED

Software Libraries Used

For providing various functionalities to the application, different publicly and commercially available software development toolkits have been used (Fig. 1). C++ was chosen as the programming language and the open source VR Juggler software toolkit was used for controlling the virtual environment (www.vrjuggler.org). VR Juggler provides a platform for virtual reality application development and allows a user to run a single application on different VR systems easily by just changing a configuration file [18]. The VR Juggler Portable Runtime (VaPoR) library provides an operating system abstraction layer that simplifies the process of creating cross-platform software.

Application Platform	Visualization Toolkit	Haptic Device Control	Network Capability	Collision Detection & PBM
VR Juggler	OpenGL Performer	Open Haptics Toolkit	TCP	VPS

Figure 1: Application Libraries

Voxmap Point Shell (VPS) software from Boeing Corporation was used for collision detection and for simulating realistic part behavior using physically based modeling. Kim and Vance [15] compared several collision detection algorithms and found VPS to be most appropriate for implementing collision detection and physically based modeling for handling arbitrary CAD geometry. VPS represents CAD geometry using voxels which are small cube elements[19]. In VPS, point shell objects are represented by a set of surface point samples and their associated inward pointing surface normal, collectively called a point shell. The environment of voxmap objects is represented by a single cubic occupancy map called a voxmap. Depth of penetration can be calculated when the object's point shell penetrates a voxmap object. Penalty forces are calculated using the depth of penetration and are summed to give a

resultant force and torque. The dynamic rigid body behavior is simulated with Newton-Euler dynamics. OpenGL Performer scene graph library is used for visualization. Using VR Juggler as the platform and C++ as the programming language, the application currently runs on Windows, Linux and Irix platforms. For communication with the haptic devices, Open Haptics Toolkit from Sensable Technologies was used on Windows and Linux platforms. TCP socket programming was used for communicating over the network.

Computer Hardware

The software developed as a result of this research can be used on a wide variety of systems from single-pipe display systems such as head-mounted displays, single projection walls, and projection benches as well as multi-pipe stereo projection environments such as CAVE etc. The main copy of the application runs with the haptic device hooked up to a Windows or Linux workstation. This system can then optionally be networked any other type of VR system running on Windows, Linux or Irix platforms.

Figure 2 shows the desktop hardware configuration of the system. At Virtual Reality Applications Center (VRAC) of Iowa State University, we have tested this system on Windows and Linux workstations. The Windows machine consists of dual 3.6 Giga Hz Intel Xeon processors with 3 Giga Byte RAM while the Linux machine has dual 3.2 Giga Hz Intel Xeon processors with 2 Giga Byte of RAM. Both machines have the PCI Express Nvidia Quadro 4400 graphics card with 512 Mega Byte graphics memory. Active quad-buffered stereo and Crystal Bytes shutter glasses from Stereographics Corporation provide stereo viewing of CAD models and a PHANToM[®] Omni haptic device provides force feedback. The application running on any of these workstations is capable of communicating with the multi-pipe projection screen VR system at VRAC which runs on Irix 6.5 operating system.



Figure 2: Application Hardware Setup (User performing assembly with PHANToM[®] Omni device while viewing parts in stereo using LCD shutter glasses and emitter)

The multi-pipe stereo projection environment at VRAC has a 10 ft. x 10 ft. x 10 ft. room equipped with 6 rear projection surfaces, which serve as the walls, ceiling and floor. The users wear stereo shutter glasses which are synchronized with the computer display to alternate the left and right eye views at a rate of 96 Hz in order to produce stereo images. A magnetic tracking system tracks the user's head, hand, and arm position. A 24-processor SGI Onyx2 Reality Monster supplies the computational power and six InfiniteReality2 graphic pipes, each with 256MB of texture memory manage the graphics output. The processors are 400MHz MIPS R12000's and the computer contains 12 Giga Byte of RAM.

Haptic Devices

The application currently supports PHANToM[®] (Personal Haptic iNterface Mechanism) haptic devices from SensAble Technologies (Fig. 3) We use Open Haptics Toolkit library for communicating with haptic devices on Windows and Linux platforms.



Figure 3: PHANToM[®] Desktop, PHANToM[®] 1.5, PHANToM[®] 3.0 and PHANToM[®] Omni, by SensAble Technologies (Images courtesy of Novint Technologies)

Open Haptics Toolkit is compatible with different models of the PHANToM[®] haptic device. At our VRAC facility, we have PHANToM[®] Desktop, PHANToM[®] 1.5 and PHANToM[®] 3.0. PHANToM[®] Omni is our latest addition to the haptic devices here at VRAC. The Omni model from Sensable technologies provides the most compact and cost-effective solution for haptic rendering. It has a portable design and communicates using IEEE-1394 FireWire port interface.

APPLICATION INFRASTRUCTURE

The application presented here, has four main simulation loops namely, graphics, hand collision, physics and haptics (Fig. 4). The system reads the CAD geometry files and generates haptic and graphic representations of the models during initialization. Once initialization is complete, the application monitors the actions performed by the user and responds to those actions.

Initialization

During the initialization step, the application loads the graphics model file (*.wrl, *.iv, *.3ds, *.pfb etc.) and the haptic model file (*.stl). When a .stl file is loaded, it is parsed and the triangle and normal information is read and stored in a dynamic data structure. During the voxelization step, the set of triangular polygons read from the CAD model file are converted to VPS spatial representation called voxmap. VPS being a pair-wise collision detection algorithm detects collision

between object pairs. The next step is binding all objects to generate the VpsPbmPairType data.

Finally, the initialization step is completed by calculating and storing the physical properties like mass, center of mass, and moment of inertia for each CAD model. The application then loads the hand model file which consists of already voxelized data for the hand model. All this data is required by subsequent VPS calls for simulating realistic physical behavior of the models in the virtual environment.

After the initialization step is complete, the application is launched and ready for interaction by the user. Now, the application monitors the user's actions like grabbing, moving or colliding parts and responds to those actions correspondingly.

Simulation Loops

The graphics loop of the application monitors all input from the mouse or keyboard. It responds to the user's navigation commands and menu events. The physics simulation loop is at the core of the entire application. It is responsible for realistic simulation of part behavior. It performs all computations for collision detection, calculates all reaction forces and computes the final position matrices for the dynamic object for every frame.

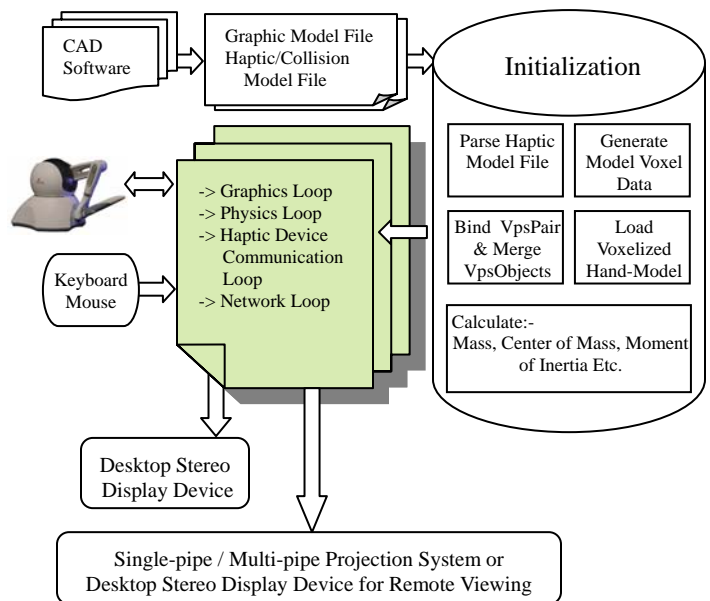


Figure 4: Application Infrastructure

The haptic device communication loop reads the stylus position data and switch state from the haptic device and sends computed force back to the device. The network loop of the application starts a TCP communication link between two copies of the application running concurrently on two different VR systems located at geographically dispersed locations. This mode can be used selectively and the application's role to act as a client or server can be changed using the configuration file.

APPLICATION FLOWCHART

Figure 5 presents the application flowchart. The application starts by reading a configuration file which includes several state flags for turning on/off specific modules. For instance the flag “PBM_ON” controls whether this application should start the physically based modeling thread or not. The flag “HAND_CONTROL” controls whether the virtual hand should be controlled by the local haptic device or through some device on the network. These commands are followed by a list of models to be loaded. For each model, a graphics file (*.pfb, *.vrl, *.vrm, etc.) for visualization and a triangle file (*.stl) for physics based modeling must be provided. The users can also specify an initial position and scale for each model. VPS voxelizes the triangle file to build the necessary model data for PBM computation. Finally, using OpenGL Performer, all graphic objects are stored in a scene graph tree structure.

loop updates hand position and orientation from two possible sources: the haptic device or some device on the network, depending on value of the “HAND_CONTROL” flag. The application detects the collision between the virtual hand and each of the PBM models by the VPS function “VpsIntersect”.

If a collision is detected, and the “grab” button is pushed, the model is attached to the hand by a virtual spring. The physics loop detects the collision detection between the manipulated model and other models which are assumed to be static. Once a collision is detected, the reaction force is returned. This force is sent to the haptics loop. Once all forces exerting on the manipulated model are computed, the physics loop will invoke the rigid body dynamics solver to determine the new model position. This is done by VPS function “VpsPbmEvolve”. The graphics loop receives the new model position and updates visualization. If the manipulated object is released, the application returns back to monitor the hand collision loop.

APPLICATION FEATURES

Applications of VR for simulating assembly tasks have been around for the past decade. Although several methods for virtual assembly have been proposed, issues like platform independent coding, support for different VR systems and handling complicated CAD geometry still need to be addressed. In this paper we have tried to address such issues by adding different features to the application. We have tried to combine stereo vision, haptic feedback, multi VR system support, realistic part interaction and ability to handle arbitrary CAD geometry. In the following section we elaborate the different features of the application developed so far.

Direct Import of CAD Geometry

Object preprocessing steps inhibit direct transfer of CAD data from CAD to CAD-VR interfaces. Additional preprocessing adds time to the product design process and could result in the need for additional model files which could lead original CAD data being lost or transformed. Thus, in order to integrate an application in the design process, direct transfer of CAD data, without additional preprocessing, is desirable. While designing this application, special attention has been given to this problem and standardization has been made possible at every step. The application supports direct transfer of CAD data from standard CAD software to the virtual environment.

The application uses graphic models for visualization and haptic models for performing collision detection and physically based modeling. Thus for each model loaded in the environment, the designer has to export a graphics file (*.vrl, *.iv, etc.) and a collision model file (.stl). For graphics, we can load *.vrl, *.iv, *.3ds, *.pfb and several other generic CAD formats. For collision detection we use standard .stl file format. The *.stl file is parsed and the triangle and normal information is given to VPS functions to generate the voxel representation for collision detection and physically based modeling.

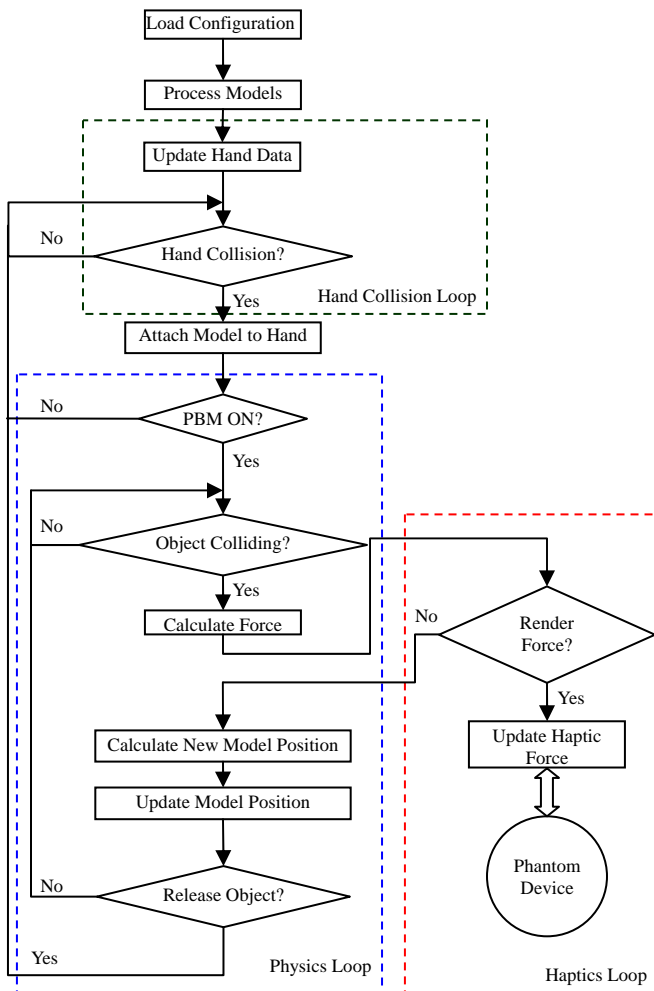


Figure 5: Application Flowchart

Once the initialization step is finished, the application is running in four major loops, graphics, hand collision, physics and haptics. The graphics loop is based on OpenGL Performer with VR Juggler as the base framework. The hand collision

Runtime Configuration Environment

The user can configure the assembly environment using a simple configuration file that the application reads for its initial setup. Using the configuration file, the user can specify the number of models to be loaded and also the location for each model in the virtual environment. The application also provides flexibility to scale the parts. This initial scale and translation are applied to both the graphic and haptic representations of the models while the initial scene is setup.

The application also allows the use of different voxel sizes for different parts in the environment. Large parts which are needed only for collision detection can be voxelized at a coarse scale resulting in large voxels; while parts which have to be assembled with tight tolerances can be voxelized using a small voxel size to make assembly possible. The application's programming structure has been organized into different modules which can be configured to be switched on or off using the configuration file. Thus, the same executable file can be used to run the application in a full or selective feature mode and different modules can be selected to operate using the configuration file.

The Network Module

Consulting with other engineers and taking feedback from other people in the organization, like shop floor workers etc. is an important part of an assembly sequence design process. To fulfill such requirements the application provides a network module that can be activated selectively. When running in the network configuration, the application (running at the workstation with haptic feedback) acts as a Server and communicates with the Client copy running at any geographically dispersed location through a non dedicated network channel. Figure 6 shows operations performed by the Server and Client modules of the application. The Server module of the application runs in full mode, i.e. it loads graphic and haptic models and performs collision detection and physically based modeling, calculates the model's final position and sends the hand and dynamic model's position information to the client.

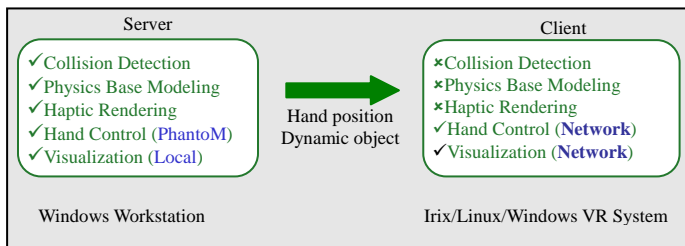


Figure 6: Network Architecture

Since the client does not have any haptic devices, the client module of the application will run in a reduced capability mode where it will only load the graphic models and communicate with the server module to update the hand and dynamic model's position in the virtual environment. All computations for collision detection, physically based modeling and haptic

rendering are done at the server module. The idea is to run the client module for demonstrating the assembly sequence to people at a different geographic location. The VR Juggler based architecture enables the application to run on many types of VR systems (e.g. Desktop, Cave, Power wall etc.) and many operating systems (Linux, Windows or Irix tested so far.) Thus, an engineer can work on his/her workstation and assemble complex CAD models using haptic and visual feedback while the same assembly sequence can be observed and analyzed by the client users in a CAVE, Power Wall or a Desktop system at any other location. This will prove to be very useful for collaborative assembly sequence analysis and demonstration purposes. Figure 7 shows the Client module of the application running in the multi-pipe stereo projection environment at VRAC.

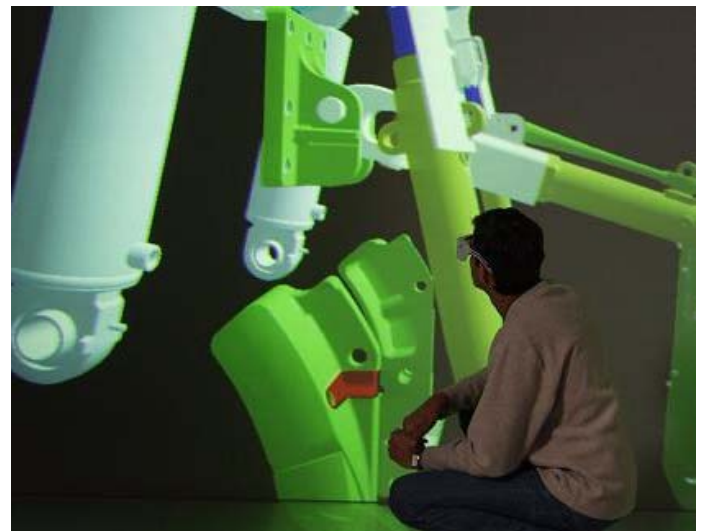


Figure 7: Assembly Demonstration in C6 at VRAC

Record and Play Module

Virtual assembly applications can be used for analyzing and evaluating different assembly sequences. Assembly workers can be brought into virtual environments to perform specific assembly tasks for training and evaluation purposes. Also virtual assembly applications can be used for prototyping collaborative assembly tasks. All of these requirements demand the same assembly sequences to be displayed and analyzed several times in a virtual environment. To accomplish this, a Record and Play module has been developed. By activating the module using the application menu; all assembly sequences that are performed can be recorded. Later the recorded sequence of assembly tasks can be played for demonstration or training purposes.

Stereo Viewing

Analyzing and understanding assembly sequences of complex 3D CAD models using a two dimensional flat computer screen is a difficult task. Stereo views provide a

better interface for interacting with complex 3D geometry. Depth cues provided by stereo viewing help convey the spatial relationships among different parts in a 3D assembly model and enhance the user's understanding of a design. Viewers can perceive distance and spatial relationships between different object components more realistically and accurately than using a two dimensional computer screen.

We use quad-buffered page-flipping mode for creating stereo views on the desktop monitor. In page-flipping, the right- and the left-eye frames are shown alternately on the screen. When the right-eye frame is shown on the screen, the left eye turns dark in the shutter glasses, and vice-versa. In this mode, both the horizontal and vertical resolutions are kept the same, since the frames are displayed one by one on the entire screen providing the best possible stereo view on a desktop monitor.

ASSEMBLY TASK

This application was tested using CAD models of a hitch from Deere & Company. These models were successfully imported and assembled in the desktop virtual environment (Fig. 8). The assembly task was to place the Upper Lift Link End between the two pin-holes of the Cross Member and insert the Pin through the pin-holes of the Cross Member. The part statistics are shown in Table 1.

Model Name	No. of Triangles	No. of Voxels
Hitch	369,901	50,967
Cross Member	17,627	838,356
Upper Lift Link End	6,724	46,453
Pin	1,968	89,332

Table 1: Hitch Model Parts



Figure 8: Parts to be Assembled

For handling such large data-sets, different voxel sizes were used for different parts in the environment. Small parts that need to have tight contact tolerances for assembly (like Upper Lift Link End, Cross Member and Pin) were voxelized with a small voxel size while the large Hitch part was voxelized with a larger voxel size (Fig. 9).

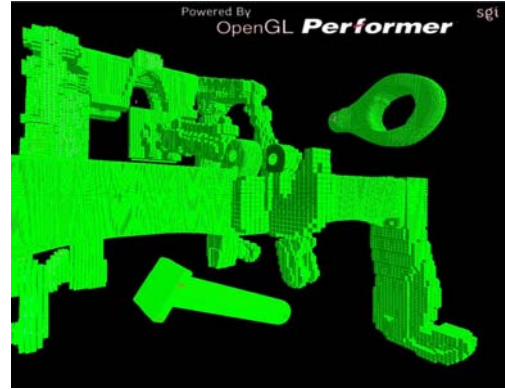


Figure 9: Voxelized View

This method saves memory and facilitates handling of large data-sets for collision detection and PBM while performing assembly. Figure 10 shows the results of the successful assemble of the CAD models using the PHANToM[®] Omni haptic device.

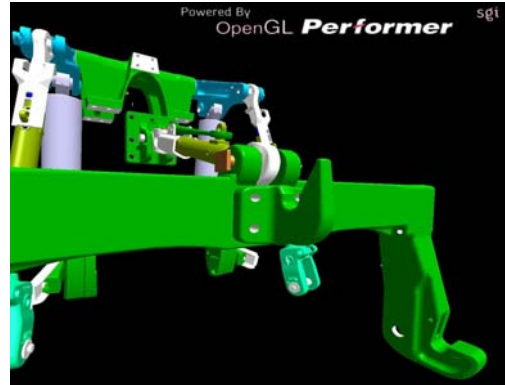


Figure 10: Assembled Parts

CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a low cost solution for performing mechanical assembly in virtual environments. The application presented has been shown to handle complex CAD models consisting of large and small parts while performing assembly in a desktop virtual environment. The application provides real time force feedback and quad-buffered stereo vision to the user for natural and intuitive part interaction when assembling mechanical components.

The unique flexible architecture allows the application running standalone on a low end desktop or high end CAVE systems or in a networked environment where multiple VR systems communicate through internet. Because of this feature, the application has the potential to reach maximum usability in small and large business companies as well as in academia.

As the research progresses, we will be investigating performing dual-handed assembly using two PHANToMs, methods for handling subassemblies, using different voxel sizes in the same part and having a two way communication in the network module to enhance collaborative assembly.

ACKNOWLEDGEMENTS

We are very grateful for the technical assistance of William McNeely of the Boeing Company. This work was funded by Deere & Company.

REFERENCES

- [1] S. Jayaram, Vance, J. M., Gadh, R., Jayaram, U. and Srinivasan, H., "Assessment of VR Technology and its Applications to Engineering Problems," *ASME Journal of Computing and Information Science in Engineering*, vol. 1, pp. 72-83, 2001.
- [2] B. P. Perles, and Vance, J. M., "Interactive Virtual Tools for Manipulating NURBS Surfaces in a Virtual Environment," *Industrial Virtual Reality Symposium Proceedings*, Chicago, IL., 1999.
- [3] C. E. Kim, and Vance, J. M., "Using Vps (Voxmap Pointshell) As The Basis For Interaction in a Virtual Assembly Environment," *ASME Design Engineering Technical Conferences*, (DETC2003/CIE-48297), Chicago, IL., 2003.
- [4] R. Gupta, and Zeltzer, D., "Prototyping and Design for Assembly Analysis using Multimodal Virtual Environments," *Proceedings of ASME Computers in Engineering Conference and the Engineering Database Symposium*, Boston, MA., 1995.
- [5] R. Gupta, Whitney, D., and Zeltzer, D., "Prototyping and Design for Assembly Analysis using Multimodal Virtual Environments," *Computer Aided Design (Special issue on VR in CAD)*, vol. 29, pp. 585-597, 1997.
- [6] A. S. Coutee, McDermott, S.D., and Bras, B., "A Haptic Assembly and Disassembly Simulation Environment and Associated Computational Load Optimization Techniques," *ASME Journal of Computing & Information Science in Engineering*, vol. 1, pp. 113-122, 2001.
- [7] A. S. Coutee, and, Bras, B., "Collision Detection for Virtual Objects in a Haptic Assembly and Disassembly Simulation Environment," *ASME Design Engineering Technical Conference & Computers in Information Engineering* (DETC2002/CIE-34385), Montreal, Canada, 2002.
- [8] S. Jayaram, Jayaram, U., Wang, Y., Tirumali, H., Lyons, K. and, Hart, P., "VADE: A Virtual Assembly Design Environment," *Computer Graphics and Applications*, vol. 19, pp. 44-50, 1999.
- [9] S. Jayaram, Jayaram, U., Wang, Y., and Lyons, K., "CORBA-based Collaboration in a Virtual Assembly Design Environment," *ASME Design Engineering Technical Conferences & Computers in Information Engineering*, (DETC 2000/CIE-14585), Baltimore, MD., 2000.
- [10] U. Jayaram, Tirumali, H. and, Jayaram, S., "A Tool/Part/Human Interaction Model for Assembly in Virtual Environments," *ASME Design Engineering Technical Conferences & Computers in Information Engineering*, (DETC 2000/CIE-14584), Baltimore, MD., 2000.
- [11] F. Taylor, Jayaram, S. and, Jayaram, U., "Functionality to Facilitate Assembly of Heavy Machines in a Virtual Environment," *ASME Design Engineering Technical Conferences & Computers in Information Engineering* (DETC 2000/CIE-14590), Baltimore, MD., 2000.
- [12] H. J. Bullinger, Richer, M., and Seidel, K.-A., "Virtual Assembly Planning," *Human Factors and Ergonomics in Manufacturing*, vol. 10, pp. 331-341, 2000.
- [13] T. Fernando, Marcelino, L., Wimalaratne, P. and, Tan, K., "Interactive Assembly Modeling within a CAVE Environment," *Eurographics-Portuguese Chapter*, pp. 43-49, 2000.
- [14] T. C. Johnson, and, Vance, J.M., "The Use of the Voxmap Pointshell Method of Collision Detection in Virtual Assembly Methods Planning," *ASME Design Engineering Technical Conferences & Computers in Information Engineering* (DETC2001/DAC-21137), Pittsburgh, PA., 2001.
- [15] C. E. Kim, and Vance, J. M., "Collision Detection and Part Interaction Modeling to Facilitate Immersive Virtual Assembly Methods," *ASME Journal of Computing and Information Sciences in Engineering*, vol. 4, pp. 83-90, 2004.
- [16] C. E. Kim, and Vance, J. M., "Development of a Networked Haptic Environment in VR to Facilitate Collaborative Design Using Voxmap Pointshell (VPS) Software," *ASME Design Engineering Technical Conferences & Computers and Information in Engineering Conference* (DETC2004/CIE-57648), Salt Lake City, UT., 2004.
- [17] H. Wan, Gao, S., Peng, Q., Dai, G and Zhang, F., "MIVAS: A Multi-Modal Immersive Virtual Assembly System," *ASME Design Engineering Technical Conferences & Computers in Information Engineering* (DETC 2004/CIE-57660), Salt Lake City, UT., 2004.
- [18] C. Just, A. Bierbaum, A. Baker, and, C. Cruz-Neira., "VR Juggler: A Framework for Virtual Reality Development," *2nd Immersive Projection Technology Workshop (IPT98) CD-ROM*. Ames, IA., 1998.
- [19] W. A. McNeely, Puterbaugh, K. D. and, Troy, J. J., "Six Degree-of-Freedom Haptic Rendering Using Voxel Sampling," *SIGGRAPH '99 Conference Proceedings*, Annual Conference Series, Los Angeles, CA., 1999.