Networked Collaborative Design and Control for Collaborative Product Development Using Haptic Interface

Yuan-Shin Lee, Shiyong Lin and Roger J. Narayan

North Carolina State University, yslee@ncsu.edu

ABSTRACT

This paper presents a collaborative product development and prototyping by using distributed haptic interfaces along with deformable objects modeling. Collaborative Virtual Environment (CVE) is a promising technique for industrial product development and virtual prototyping. Network control problems such as network traffic and network delay in communication have greatly limited collaborative virtual environment applications. The problems become more difficult when high-update-rate haptic interfaces and computation intensive deformable objects modeling are integrated into CVEs for intuitive manipulation and enhanced realism. A hybrid network architecture is proposed to balance the computational burden of haptic rendering and deformable object simulation. Adaptive artificial time compensation is used to reduce the time discrepancy between the server and the client. Interpolation and extrapolation approaches are used to synchronize graphic and haptic data transmitted over the network. The proposed techniques can be used for collaborative product development, virtual assembly, remote product simulation and other collaborative virtual environments where both haptic interfaces and deformable object models are involved.

Keywords: haptic interface, virtual Prototyping, deformable object modeling.
DOI: 10.3722/cadaps.2012.327-343

1 INTRODUCTION

With the advent of haptic technology, traditional visual-only collaborative virtual environments have been gradually transited to haptic-based collaborative virtual environments, such as haptic collaborative virtual sculpting systems, remote surgical simulators, and teleoperation systems [1, 2, 3]. Various haptic devices have been introduced to enhance the realism of virtual reality (VR) environments by providing force feedback to users. A few stand-alone virtual reality systems have been developed for virtual prototyping, virtual assembly, and surgical simulation [3, 4]. Recently, researchers developed collaborative virtual environments based on network communication technology for virtual sculpting and virtual assembly [5, 6]. It is prompted by the trend that manufacturing enterprises have distributed their design centers and manufacturing facilities all over the world to reduce cost. Using collaborative virtual environments, designers and manufacturing technicians can work together to review and analyze the product design and functions without physical presence. Customers can also easily participate in the whole product design and manufacturing cycle and interactively communicate with sales representatives or designers. They can see and even “feel” the virtual products by using the haptic interface over the Internet intuitively. As a result, collaborative virtual environments can significantly improve the product development quality and shorten the product development cycle. Figure 1 shows a five-degree of freedom (DOF) haptic device set up at our laboratory for product development.

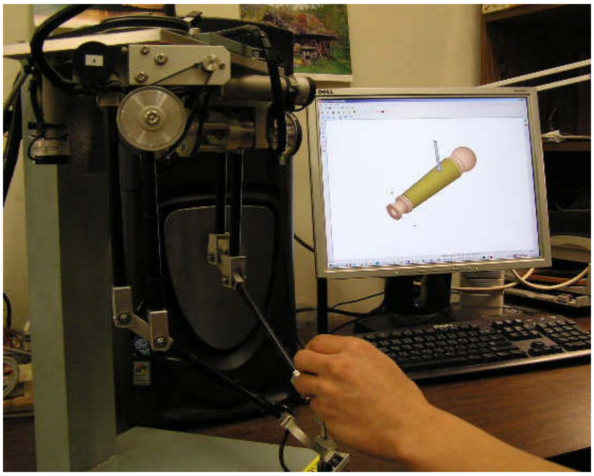


Fig. 1: Lab setup of the 5-DOF haptic device for product design.

Collaborative Virtual Environment (CVE) has been researched and widely used to many areas including Engineering, Sciences and Education. For example, Mejía et al. [7] presented a ‘Collaborative Engineering Environments’ to enable cooperative work. CVE was also applied to Design and Manufacturing industry for Collaborative Product Design (CPD). Huang [8] reported the research on a web-based framework for supporting collaborative product design review. This paper focuses on collaborative product design by virtual sculpting using haptic interfaces. So far, most stand-alone or collaborative virtual sculpting systems considered only rigid objects in the virtual design and manufacturing environments [9, 10]. In fact, soft materials such as foam, rubber, sponge, plastics and fabrics are commonly used in industrial products for decorative, ergonomic and mainly functional purposes. Duriez et al. [11] used deformable objects in virtual prototyping and modeled the snap-in tasks. Rabaetje [2] integrated deformable objects like hoses or pipes into a virtual assembly system. Deformable object models were also used for haptic function evaluation of multi-material part design [12]. Deformable models can represent realistic physical properties of soft materials. Integrating deformable models into virtual environments can create more realistic simulation scenarios. In addition, more accurate force and torque feedback can be generated based on deformable models when haptic tools are used to touch soft objects.

There are a few technical challenges to integrate both haptic interfaces and deformable models into collaborative virtual environments. Haptic interface usually requires high update rate (e.g. 1000 Hz) to guarantee smooth sensation to haptic users. Fast position sampling, updating and force calculation are also demanded. Network problems may cause instability and inconsistence of collaborative virtual systems. In addition, deformable object simulation is notoriously time consuming, which can further deteriorate the stability of collaborative virtual systems.

Techniques are presented in this paper to deal with aforementioned challenges when integrating haptic devices and deformable models into collaborative virtual systems. Extended from client-server architecture, a hybrid network configuration is presented to balance the computation burden and control the collaborative virtual system. Adaptive artificial compensation is applied at the server to alleviate the time discrepancy between the server and clients. Interpolation and extrapolation approaches derived from Verlet Integration are used to control and synchronize the data transmitted over the network.

2 5-DOF HAPTIC INTERFACE AND CONTROL SYSTEM

Haptic interfaces provide a novel way for the interaction in virtual environments. By understanding the mechanism and control principle of the haptic device, one can easily integrate a device into virtual systems. In this paper, a lab-built 5-DOF haptic device shown earlier in Figure 1, along with another two commercially available Phantom haptic devices, is used in our haptic collaborative virtual prototyping framework [13]. Figure 2 shows the data flow of the haptic controller system. Encoders are attached to the motors as well as the probe to record their movements. The encoders can acquire the position and orientation information of the haptic probe. Such information is sent to counter board. Counter board parses the information and sends the processed data to haptic control program. During the system simulation, force and torque of the virtual probe are calculated based on virtual haptic-object interaction. Resultant forces and torques are converted into equivalent torques. Digital signals of equivalent torques are generated and then converted into analog signals via D/A (Digital/Analog) board. Six amplifiers drive six active motors by the equivalent torques. At last, the motors can generate the same force and torque for the haptic probe. With the controller system, the user can feel the calculated force and torque in the virtual environment while holding the haptic probe in the real world. More details of the mechanism control and force feedback analysis can be found in our earlier work in [13].

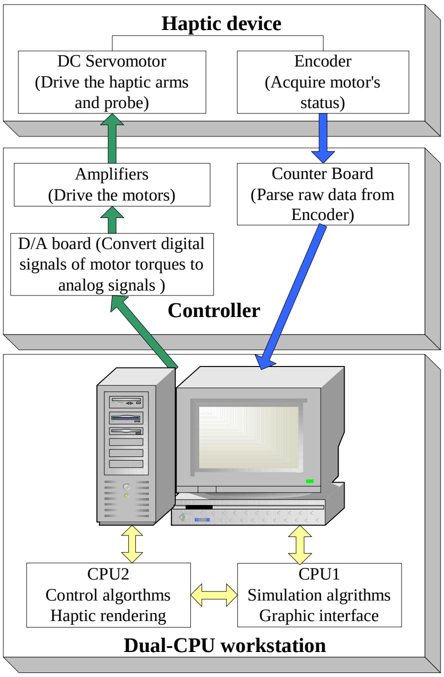


Fig. 2: The controller and signal flow for haptic interface system.

3 MODELING AND SIMULATION OF HETEROGENEOUS DEFORMABLE OBJECTS

Deformable objects modeling and simulation are important components in many virtual systems. To better represent real products, heterogeneous structures and various material properties should be modeled [14]. Figure 3(a) shows an example of a heterogeneous deformable model with multiple materials. Boundary surfaces of each material are first obtained from Computer-Aided Design systems, which include geometric information of internal heterogeneous structures. From boundary surfaces, heterogeneous volumetric mass spring deformable model is generated by a tri-ray snapping algorithm developed in our earlier work in [15]. Interfaces between different materials are preserved in the deformable model, which can be used to explicitly differentiate various materials within one object. To model the deformable objects, a spring-mass modeling is used for the deformation of soft material, as shown in Figure 3(b). By selecting any mass point, all its homogeneous mass points can be chosen simultaneously and material properties like mass can be specified efficiently, which is an advantage of our heterogeneous model. Similarly, we can assign physical properties to springs such as their stiffness.

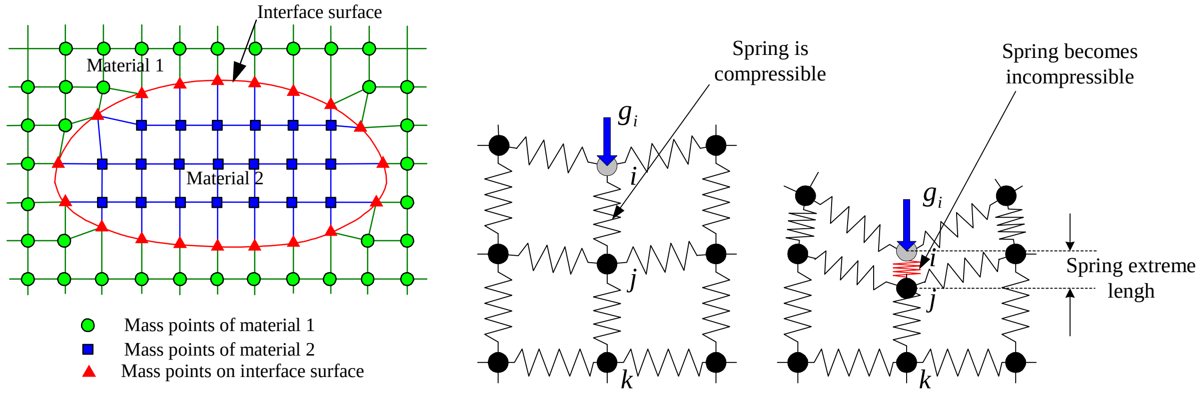
Rigid object simulation is much easier in computation efforts. It is computationally expensive to simulate deformable objects in real-time collaborative virtual systems. Specifically, numerical integration of deformable models and collision detection between haptic tools and deformable models are the main factors that cause the computational difficulty. A rigid object is considered as a single mass point and its simulation can be described as

$$M\ddot{\vec{X}} + D\dot{\vec{X}} = \vec{F} \qquad (1)$$

where $\vec{X}$ is the object displacement; $\vec{F}$ is the external force; M and D are mass and damping parameters. Comparing to rigid objects, each deformable object consists of lots of mass points, as show in Figure 3(b). Each mass point acts like a rigid object. Deformable object simulation can be expressed by

$$M\ddot{\vec{x}}_i + D\dot{\vec{x}}_i = \vec{F} \quad i=1,\ldots,n. \qquad (2)$$

where $\vec{x}_i$ is the displacement of mass point i. A medium size deformable object often consists of several thousand mass points. Although not all mass points are moved by the external force $\vec{F}$ during the simulation, it is common that a few hundred mass points are involved in the computation using Equation (2). Therefore, the computational time of deformable object simulation could reach up to a thousand times longer than that of rigid object simulation. More details of deformable object modelling during force impact can be found in our earlier work presented in [15].



(a) (b)

Fig. 3: (a) Modeling of heterogeneous objects; (b) Modeling of deformable objects.

Collision detection is another time consuming process in deformable object simulation. Figure 4 provides an example of pushing the deformable object surface against internal hard structure. In this paper, a bounding volume hierarchies such as OBB (oriented bounding box) and AABB (axis aligned bounding box) are applied to speed up collision detection [13]. As shown in Figure 4(a), when contact point i is continuously pushed inward, spring ij and jk will reach their extreme length so that they cannot be further pushed down as shown in Figure 4.(b). Further pushing cannot move the tool and object surface any more. Instead, a large collision force is resulted as if the tool directly touches the inside hard structure. This is analogous to pressing a finger against your leg until you feel the hard bone inside. To calculate the collision force f_{in} caused by the hard structure, the tool is assumed to be moved to its virtual location, as the dotted-line tool shown in Figure 4.(b). More details of deformable object collision detection can be found in our earlier work presented in [16].

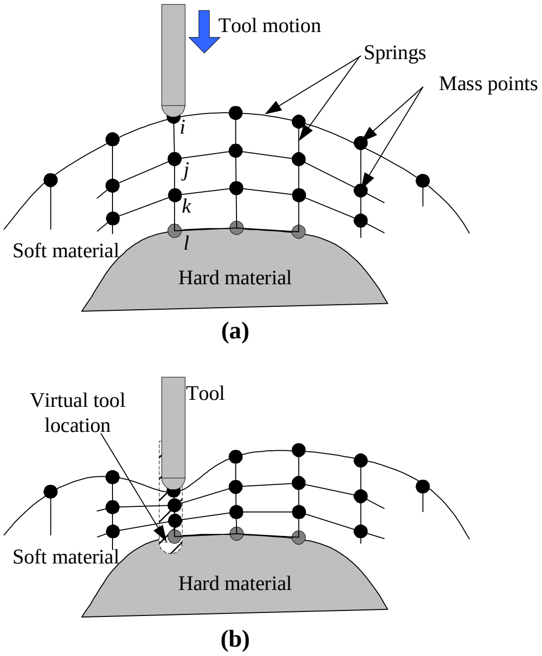


Fig. 4: Deforming objects with soft and hard materials: (a) before deformation; (b) after deformation.

4 ARCHITECTURE AND CONTROL OF THE COLLABORATIVE HAPTIC VIRTUAL ENVIRONMENT

In this section, system architecture of the proposed collaborative virtual product development and virtual prototyping framework is discussed. The objective is to solve the instability and inconsistence issues caused by the network communication problems for the proposed collaborative haptic virtual product development system. Several techniques are proposed in this paper to deal with these problems to support the proposed collaborative virtual product development system.

4.1 System Architecture of the Collaborative System

In client-server architecture, most computation and simulation tasks are usually performed at the server. The main advantage is that only one data copy of the virtual environment is maintained so that data consistency is automatically guaranteed [17]. However, heavy computational burden may deteriorate the performance of the server. For example, if this architecture is applied in our collaborative system, the server will take charge of every time-consuming process such as tool-object collision detection, deformation simulation and force calculation. In addition, this architecture is not well scalable. As more clients join the collaborative system, the system performance may become worse. In peer-to-peer architecture, simulation is performed at every client. Scalability of this architecture is better than client-server architecture. However, the data consistency is very hard to maintain because each client has its own data of the same virtual environment.

Practically, neither client-server nor peer-to-peer architecture can be directly used for collaborative applications, especially when haptic interfaces and deformable models are integrated. Marsh et al [18] presented a roaming-server hybrid architecture based on the two basic network architectures for a distributed virtual prototyping application. Iglesias et al. [19] compared different architectures and suggested that computation was distributed at both client and server sides for their virtual assembly simulation system. However, both researches didn't take into account the complexity of deformable object modeling and simulation.

In this paper, we present a different hybrid architecture extended from client-server architecture, which supports deformable models in haptic collaborative virtual environments. Figure 5 shows the architecture of the haptic-based collaborative product development framework. Deformable objects simulation and force calculation are performed at the server, while tool-object collision detection is accomplished at the client sides. Details of these modules are discussed as following.

First, tool-object collision detection is performed at the clients (also at the server if the serve owns a haptic device). In collision detection, only virtual model surfaces are required for testing. Since surfaces of virtual objects are always available for graphic rendering at the clients, it is reasonable to shift collision detection task to the clients. This approach can decrease the computational burden of the server without causing too much trouble in computation at the clients. If collision occurs at a client side, collision information such as collision point is sent to the server. Then the server uses this information for deformation simulation and force calculation.

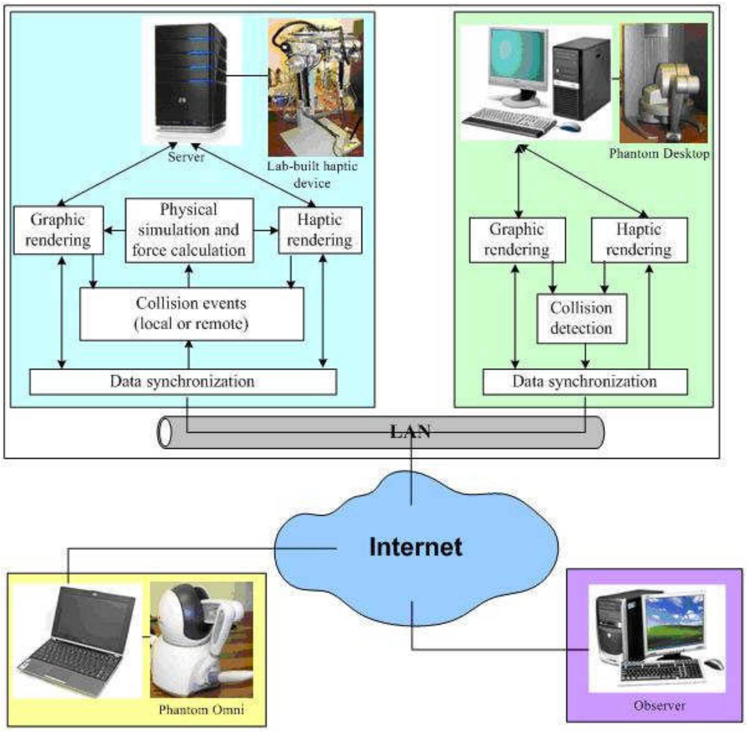


Fig. 5: Collaborative haptic-based product development system.

To maintain the data consistency of the virtual environment, deformable object simulation is only processed at the server. Clients copy simulation results such as new surfaces of deformable objects from the server. Another advantage of performing deformable object simulation at the server is to utilize the computational power of the high-end server. Meanwhile, the computational requirements for clients are minimized so that even a low-end laptop can be used as a client. Therefore, scalability of the whole collaborative system is enhanced. In addition, haptic force calculation is actually the byproduct of deformation simulation at the server. The haptic force is based on the spring force of contacted mass point by the haptic tool.

The simulation results including new surfaces of deformable objects and haptic forces of haptic devices are sent to each client at 30 Hz for geometrical data and 1000 Hz for haptic force data. Because the deformation often occurs locally, only modified surface data are sent to clients. Local data updating can reduce the network workload greatly. When clients receive the updated data, corresponding surfaces of deformable objects are modified. Positions and orientations of all haptic devices also need to be updated to show their current locations.

So far, we only consider how to balance the computation burden of a haptic-based collaborative system with the assumption that data transmitted over the network are sent and received timely and correctly. However, due to network traffic existing in most network, transmitted data may be damaged, lost or out of order. In the following sections, we analyze common network problems and present some solutions to maintain stable and efficient virtual environments.

4.2 Data Transmission and Protocols

In the collaborative virtual environment, common network problems include network delay, jitter (variation in the time it takes subsequent data packets to arrive) and packet loss, as illustrated in Figure 6. These problems may cause instability and incoherency of the virtual environment due to the delayed or information lost. In the proposed collaborative system, several sets of data are transmitted over the network: virtual object surfaces, position and orientation of haptic devices, haptic force data and event data such as tool-object collision information. How to transmit these data over the network is determined by characteristics of these data.

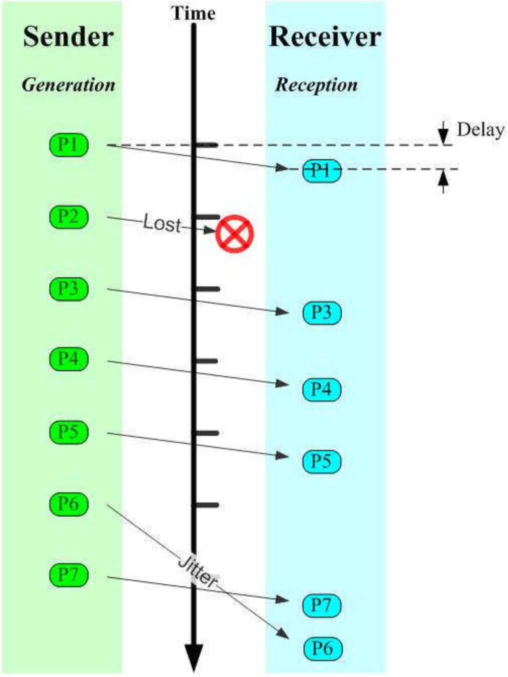


Fig. 6: Problems of network traffic and data lost during network data transmission.

Figure 7 shows the data flow between the server and clients among the networked haptic design stations. The most critical data are event data, which have to be sent accurately. For example, when a haptic device collides or keeps touching with a deformable object, collision information such as the collision point should be reliably delivered to the server and then via the server to all the other users. Therefore, TCP (Transmission Control Protocol) is used for transmitting these data. TCP can guarantee ordered data transmission, retransmission of lost data and discarding the duplicated data. The details of TCP can be easily found in most literature of network protocols.

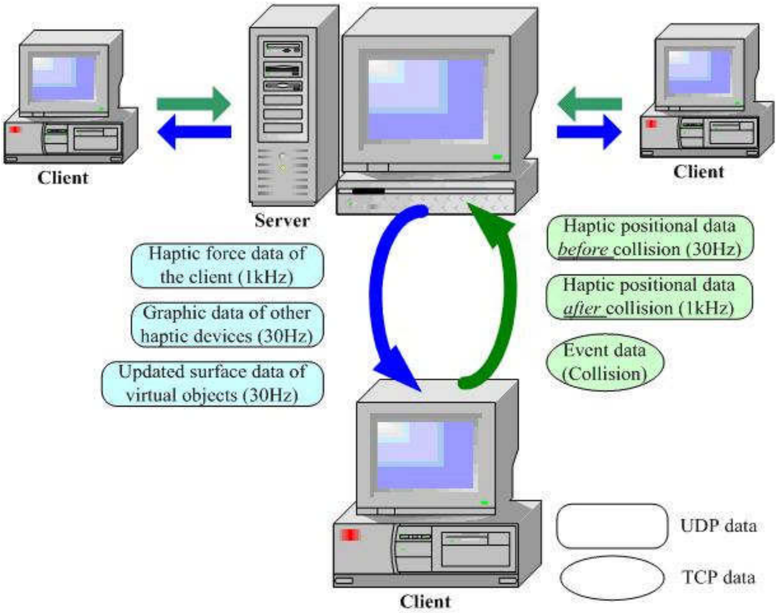


Fig. 7: Data flow between client and server.

For other data, minor transmission delay or errors are acceptable for both graphic and haptic rendering. Such problems can be moderated by approximation approaches like interpolation or extrapolation. Since a large amount of these data need to be transmitted over the network, it is more important to minimize network traffic than to guarantee reliable data transmission. Instead of using TCP, User Datagram Protocol (UDP) is applied because it is faster and more efficient than TCP. UDP delivers the packets without the overhead of checking whether every packet actually arrived or arrived in order. Both geometric data of deformable object surfaces and haptic tools are transmitted at the graphic update rate of 30 Hz using UDP. In addition, haptic force feedback is also sent by UDP at 1000 Hz from the server to each client to guarantee smooth sensation to haptic users. After receivers obtained data packets delivered from senders, these packets cannot be directly used until they are processed by data synchronization, which will be discussed in the coming section.

4.3 Data Synchronization for Accurate Networked Collaborative Object Motions

Due to possible packet delay or loss during the data transmission, received data such as haptic device positions and deformable object surfaces have to be synchronized to maintain continuous and consistent visual and tactile sensation for all users. More importantly, data synchronization can guarantee accurate collaboration among users distributed all over the network. Users should see the same scene at the same time. Proper visual and tactile clue helps them to finish a task collaboratively. Inaccurate shape of virtual objects or location of other haptic devices may lead users to underact or overact on a collaborative task so that the goal of collaboration can never be achieved. In addition, system may suffer instability due to the improper behaviors of users caused by the inaccurate or inconsistent information.

Figure 8 shows a proposed method of minimizing the network delay by using artificial time compensation in the collaborative haptic network control. Time delay is unavoidable during the transmission over any media. Such delay could be as long as a few seconds over the Internet. As shown in Figure 8, we define the network delay as $\Delta t_{network}$. The time difference between sender and receiver sides of presenting or using the same packet is defined as $\Delta t_{variation}$. Normally, $\Delta t_{variation}$ is equal to $\Delta t_{network}$ as shown in Figure 8(a). To alleviate the delay, an artificial compensation $\Delta t_{aritificial}$ is imposed at the sender, similar to the enforced delay presented in [20]. In this paper, the sender is enforced to delay $\Delta t_{aritificial}$ before it can use the data in applications such as displaying object surfaces on the sender's screen. As a result, time discrepancy between the sender and the receiver is minimized as shown in Figure 8(b). In this paper, an adaptive compensation is used instead of the fixed compensation like the one presented in [20]. If the network delay is beyond a threshold value, the current network delay value is sent back to adjust the artificial compensation at the sender. Adaptive artificial can track the changing network status so as to better alleviate the time discrepancy between the sender and receiver.

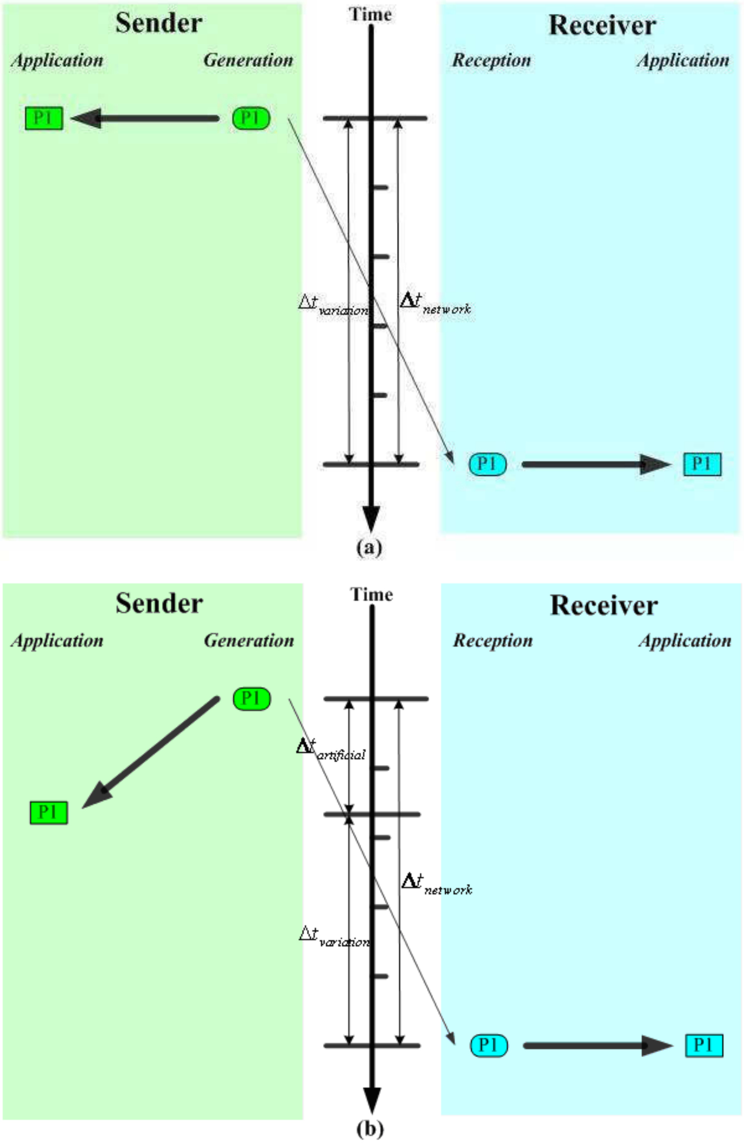


Fig. 8: Minimizing the network delay with artificial time compensation at sender side: (a) time delay without artificial compensation; (b) minimized network delay with artificial time compensation.

In network control, network jitter and data loss is more harmful than pure network delay to the haptic collaborative virtual environment [21]. Data packets arriving out of original order and the loss of data packets may cause serious discontinuity in graphic and haptic rendering. To synchronize lost or disordered data packets, a sequence number is attached to each packet. Figure 9 shows the network data synchronization by interpolation or extrapolation technique used in this paper. Both sender and receiver sides have buffers for every data unit, e.g. the position of a haptic device. Most recent data packets are stored in the buffers. The buffer size is determined by the data estimation algorithm. The larger buffer size uses more memory but can provide more accurate information for estimating lost data. In the Figure 9, we assume the buffer size is three for easy demonstration. Whenever a packet is generated at the sender, it is transmitted to the receiver and at the same time pushed into the sender's buffer. Application at the sender can access this packet after the artificial compensation $\Delta t_{aritificial}$ to decrease the time discrepancy between the sender and the receiver as shown in Figures 8 and 9. At the receiver side, the received packets are put into buffers first. Applications retrieve data packets from buffers at specific update rates, e.g. 30 Hz for graphic display and 1000 Hz for haptic force rendering.

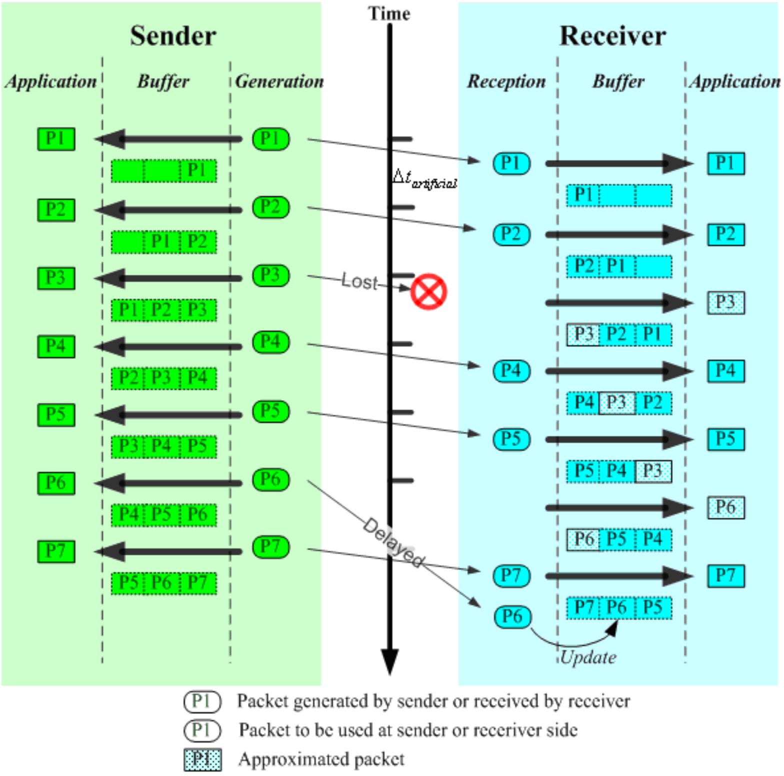


Fig. 9: Data synchronization by interpolation or extrapolation.

Because of network jitter or packet loss, packets may not be available at the time requested by application. Interpolation or extrapolation is used to estimate or predict the unavailable geometric data such as object surfaces or haptic motion locations. As shown in Figure 9, packet P3 is lost during the transmission. If P4 has already arrived at this time, P3 can be interpolated from P2 and P4. Otherwise, extrapolation is used to estimate the data of P3 from P1 and P2 that are available in the buffer. Figure 9 also provides an example of out-of-order packets due to network jitter. P6 is generated and sent earlier than P7. However, due to the network jitter, P6 is received later than P7. When P6 is requested by receiver's application, it is not available. Similarly, if P7 is available, interpolation is used to estimate P6 out of P5 and P7. Otherwise, P6 is predicted from P5 and P4 by extrapolation. After the real P6 is arrived, it replaces the estimated P6. Note that, interpolation is preferred to extrapolation since newer packet that contains recent and more accurate information is used.

The easiest method to estimate unavailable positional data is linear interpolation or extrapolation (prediction). Although it is efficient, this method is not accurate enough to estimate or predict the motion of objects such as the haptic devices. In most cases, objects are not moved at a constant speed. Several interpolation and extrapolation methods have been proposed to estimate the object motion based on the velocity and acceleration of the moving objects, such as Dead-reckoning and Newtonian methods [22, 23]. In our earlier work presented in [15], an interpolation method called Verlet integration method was applied to estimate delayed or lost data during data transmission over network. This method does not require estimating the velocity and it is more accurate than commonly used Newtonian methods. It can be derived directly from Taylor expansion of the object motion. To interpolate and find the missing data, let $X(t)$ be the position of an object at time t. The Taylor expansion at time $t+\Delta t$ and $t-\Delta t$ can be written as follows,

$$X(t+\Delta t) = X(t) + \dot{X}(t)\Delta t + \frac{1}{2}\ddot{X}(t)\Delta t^2 + \frac{1}{6}\dddot{X}(t)\Delta t^3 + O(\Delta t^4) \tag{3}$$

$$X(t-\Delta t) = X(t) - \dot{X}(t)\Delta t + \frac{1}{2}\ddot{X}(t)\Delta t^2 - \frac{1}{6}\dddot{X}(t)\Delta t^3 + O(\Delta t^4) \tag{4}$$

where $\dot{X}(t)$, $\ddot{X}(t)$ and $\dddot{X}(t)$ are the first, second and third derivatives of $X(t)$; $O(\Delta t^4)$ is the error term. Adding Equations (3) to (4), Verlet position equation can be expressed as follows,

$$X(t+\Delta t) = 2X(t) - X(t-\Delta t) + \ddot{X}(t)\Delta t^2 + O(\Delta t^4) \tag{5}$$

In Equation (5), the position at the next time step $t+\Delta t$ can be calculated from the positions at the previous time step $t-\Delta t$ and current time step t, without using the velocity $\dot{X}(t)$. For the interpolation case, we can rewrite Equation (5) as,

$$X(t) = \frac{1}{2}X(t+\Delta t) + \frac{1}{2}X(t-\Delta t) - \frac{1}{2}\ddot{X}(t)\Delta t^2 - O(\Delta t^4) \tag{6}$$

where $\ddot{X}(t)$ is an unknown variable. Since the acceleration of haptic device doesn't change a lot at a very short time under human manipulation, $\ddot{X}(t)$ can be estimated by follows,

$$\ddot{X}(t) \approx \frac{\dfrac{X(t+\Delta t) - X(t-\Delta t)}{2\Delta t} - \dfrac{X(t-\Delta t) - X(t-2\Delta t)}{\Delta t}}{\Delta t} \tag{7}$$

Substituting Equation (7) into Equation (6), the unknown $X(t)$ can be interpolated by,

$$X(t) = \frac{1}{4}X(t+\Delta t) + \frac{5}{4}X(t-\Delta t) - \frac{1}{2}X(t-2\Delta t) \tag{8}$$

For the extrapolation case of the missing data, the unknown $X(t)$ can be estimated by previous three positions as follows,

$$X(t) = 2X(t-\Delta t) - X(t-2\Delta t) + \ddot{X}(t-\Delta t)\Delta t^2 + O(\Delta t^4) \tag{9}$$

Similarly, we estimate $\ddot{X}(t-\Delta t)$ by,

$$\ddot{X}(t-\Delta t) \approx \frac{\dfrac{X(t-\Delta t) - X(t-2\Delta t)}{\Delta t} - \dfrac{X(t-2\Delta t) - X(t-3\Delta t)}{\Delta t}}{\Delta t} \tag{10}$$

From Equations (9) and (10), the unknown $X(t)$ can be extrapolated by,

$$X(t) = 3X(t-\Delta t) - 3X(t-2\Delta t) + X(t-3\Delta t) \tag{11}$$

Verlet method offers greater stability than the much simpler Newtonian methods with local 4th order error $O(\Delta t^4)$, as indicated in Equation (5). The time interval Δt is consistent and small (33 milliseconds for graphic rendering or 1 millisecond for haptic rendering) in this collaborative system. Therefore, this method offers great stability and accuracy. For haptic forces, linear interpolation or extrapolation is used currently. Higher order interpolation or extrapolation may be considered to provide more smooth tactile sensation for haptic device users, which is still under investigation.

5 SYSTEM IMPLEMENTATION AND EXAMPLES

The proposed collaborative haptic virtual product development and virtual prototyping framework has been implemented at our lab at North Carolina State University. Visual C++ and OpenGL® library functions were used in programming the server and clients. Figure 10 shows the implemented collaborative haptic product development system with three users using haptic devices to model the same product development via the network. As shown in Figure 10, a lab-built 5-DOF haptic force feedback device is connected to the server. Two clients with haptic devices have been integrated in our collaborative virtual system, as shown in Figure 10. The client also has a SensAble Phantom Desktop, which can provide 3-DOF force feedback without torque feedback. As shown in Figure 10, 3-DOF SensAble Phantom Omni is used along with this laptop.

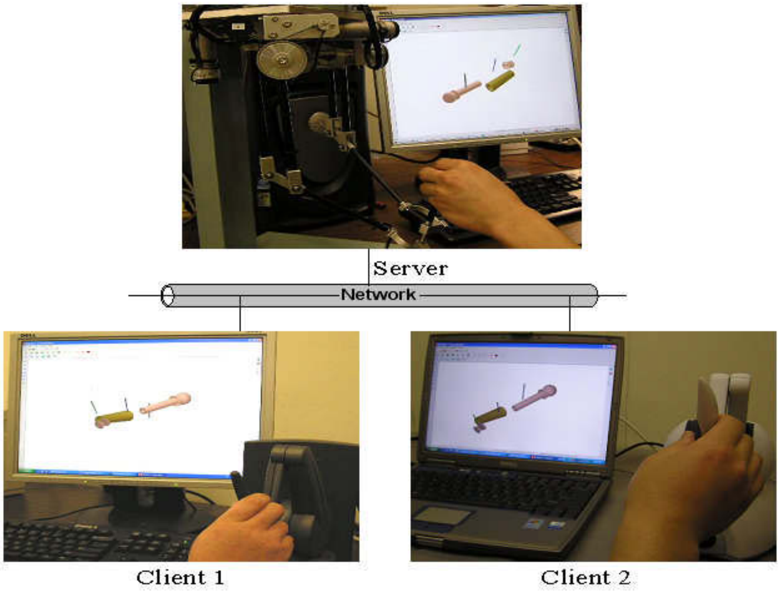
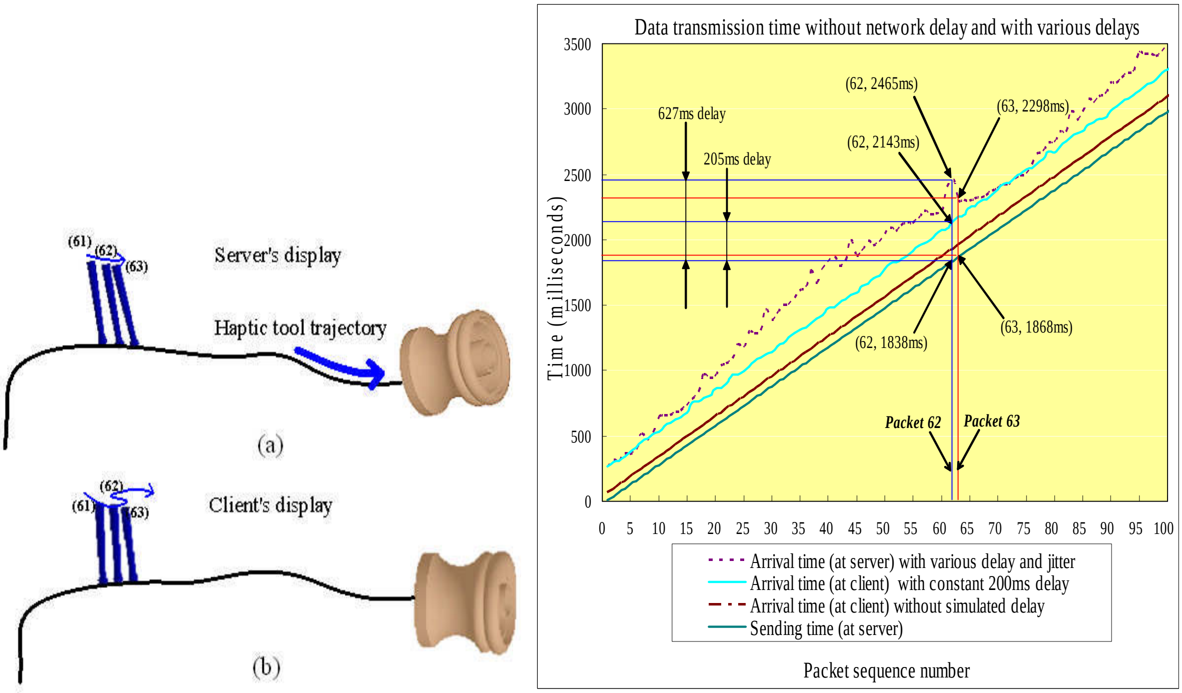


Fig. 10: Collaborative haptic design system via the collaborative network.

Figure 10 shows three users are manipulating a handle assembly collaboratively. The handle assembly consists of a soft rubber layer and another two rigid metal components. Users can move each component of assembly model using their own haptic device and in the meantime watch other users' operations. Since collision detection has been shifted from the server to clients, each client could utilize device-specific library for easy collision detection. For example, SensAble provides a toolkit OpenHaptics that includes efficient collision detection APIs. The data interface between the server and the clients is haptic positions, haptic forces and the collision information. Therefore, the proposed collaborative system can easily integrated different kinds of haptic devices.

The collaborative virtual system allows users to design, manipulate and modify a product design collaboratively at either local or remote site. Users can test (of 'feel' via the haptic force feedback) product's physical properties like stiffness and evaluate its functions intuitively. These tasks can be done individually by a single user or collaboratively by several users.

In this paper, network emulator software NetDisturb is used to simulate network traffic in a local network. Figure 11 shows transmission time for sequential haptic positional data packets sending at 30Hz from the server to a client under various simulated network traffic conditions. A constant 200ms delay is applied to the network. Received data packets are delayed about 200ms (not exactly 200ms due to real network disturbance). In addition, an exponentially distributed network jitter with a mean of 50ms is added along with constant 200ms delay to simulate the network jitter. The network jitter may result in some packets out of order. Figure 11(a) shows the correct sequence of motion at the server (61 -> 62 -> 63). In Figure 11(a), Packet 62 (containing haptic positional data) was sent 30ms earlier than Packet 63. Due to the network jitter and delay, at the receiver side, Packet 63 actually arrives 167ms earlier than Packet 62. Without data synchronization at the receiver, the haptic tool will move at the sequence of (61)-(63)-(62) as shown in Figure 11(b). The zig motion of the haptic tool can be easily observed at the client. Figure 11(c) shows transmission time for sequential haptic positional data packets that indicating the delay and network jitter of the networked haptic motions.



(c)

Fig. 11: Haptic motion jittering: (a) correct sequence at the server site ; (b) disorder at the client site; (c) data transmission time analysis.

As discussed before, it is important to reduce the presentation time discrepancy between the server and the client by using the artificial time compensation at the server site. Figures 12 and 13 compare the collaborative parts alignment process for assembly without and with the artificial time compensation. Without the artificial time compensation, distance between two parts on the server's display is always smaller than that on the client's display at the same time due to the network delay, as shown in Figure 12(a)-(b). Although the two parts are moved towards each other and actually aligned in Figure 12(c), the client still sees they are not aligned as in Figure 12(d). So the client will keep moving up its part. As a result, the alignment does no longer exist as shown in Figure 12(e). Then the server may move up its part again for a new alignment in Figure 12(g). However, the client will notice the unaligned assembly in his/her screen as shown in Figure 12(h). The client may move the part again to destroy the real alignment. This process may repeat a few times, which significantly reduces the assembly efficiency. Instead, using the artificial time compensation can minimize such visual discrepancy between the server and the client. As shown in Figure 13, the scenes on the server and the client are almost the same because of the artificial time compensation. Both users at the server and the client can easily perform the assembly.

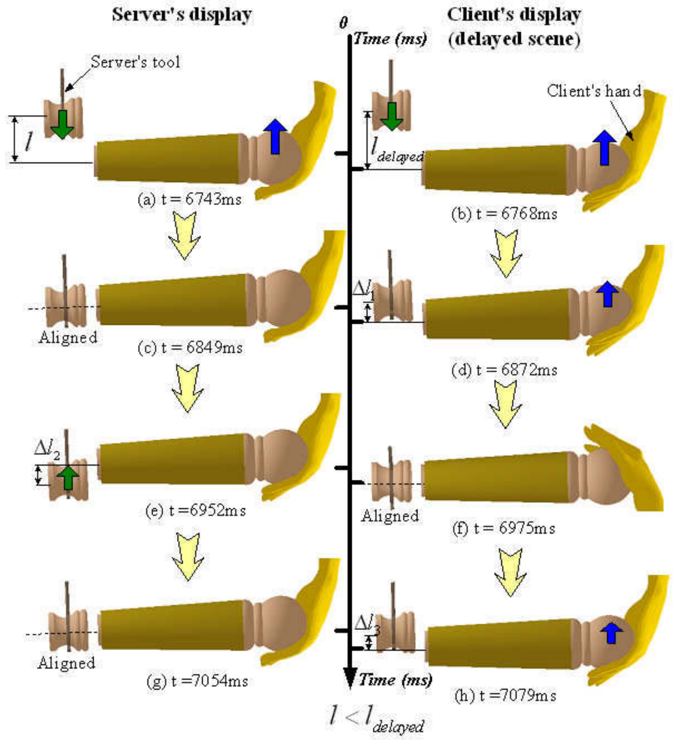


Fig. 12: Parts alignment process without artificial time compensation at the server.

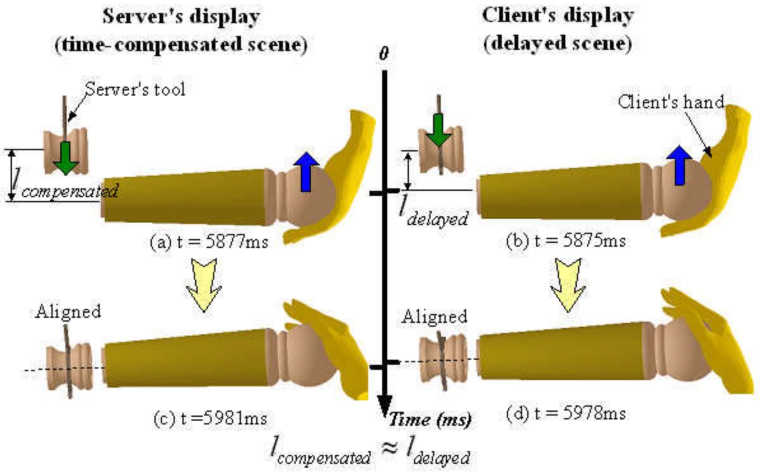


Fig. 13: Parts alignment process with artificial time compensation.

To minimize the time discrepancy between the server and the client, adaptive artificial time compensation is enforced at the server. Figure 14 compares the fixed artificial compensation and adaptive artificial compensation. Fixed artificial compensation is determined by testing the network status before the simulation starts and the compensation value doesn't change throughout the simulation. Adaptive artificial compensation varies according to recent network traffic. In this paper, if the transmission time delay is beyond ±20% of the moving average delay, artificial time compensation value will be replaced by the new time delay. As shown in Figure 14, the adaptive artificial delay reflects the dynamic network status better than the fixed artificial delay. The time difference between the server and clients are greatly reduced using adaptive artificial compensation.

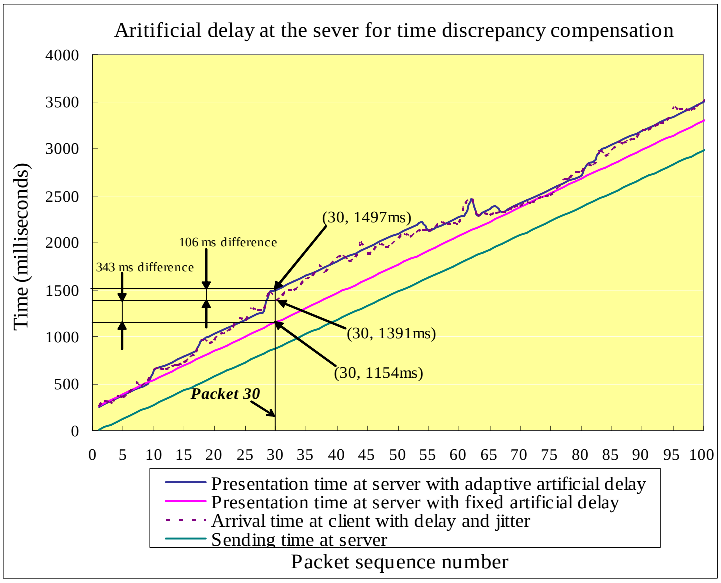


Fig. 14: Comparison between fixed artificial compensation and adaptive artificial compensation.

6 CONCLUSIONS

In this paper, a collaborative virtual product development and virtual prototyping framework is presented, which integrates heterogeneous haptic devices and deformable models. A hybrid network architecture extended from client-server architecture is presented to balance the computational burden between clients and the server. Several techniques have been presented to deal with the network distributed control problems for the collaborative product development system. Adaptive artificial compensation is used at the sender to compensate the time discrepancy between the sender and the receiver. Interpolation and extrapolation approaches are used to synchronize the transmitted data to address the network problems. The proposed techniques can be used in collaborative product development, virtual sculpting, virtual assembly, remote surgical simulation and other collaborative virtual environments.

7 ACKNOWLEDGEMENT

This work was supported by a National Science Foundation (NSF) Grants (CMMI-0800811, IIS-0905505) to North Carolina State University. Their support is greatly appreciated.

8 REFERENCES

[1] Ren, Y.; Lai-yuen, S.; Lee, Y.-S.: Virtual prototyping and manufacturing planning by using tri-dexel models and haptic force feedback, Virtual and Physical Prototyping, 1(1), 2006, 3-18, DOI 10.1080/17452750500283590

[2] Rabaetje, R.: Real-time simulation of deformable objects for assembly simulations, Proceedings of the 4th Australasian user interface conference on User interfaces, 18(1), 2003, 57-64.

[3] Lin, S.; Lee, Y.-S.; Narayan, R.: Heterogeneous deformable object modeling and surgical simulation, Proceedings of the 2007 Industrial Engineering Research (IERC) Conference, Nashville, TN, May 19-23, 2007, 281-286.

[4] Li, Y.; Kaber, D.; Tupler, L.; Lee, Y-S.: Haptic-based Virtual Environment Design and Modeling of Motor Skill Assessment for Brain Injury Patients Rehabilitation, Computer-Aided Design and Application, 8, 2011, DOI: 10.3722/cadaps.2011.149-162

[5] Gunn, C.: Collaborative virtual sculpting with haptic feedback, Virtual Reality, 10(2), 2006, 73-83, DOI: 10.1007/s10055-006-0044-4

[6] Iglesias, R.; Prada, E.; Uribe, A.; Garcia-Alonso, A.; Casado, S.; Gutierrez, T.: Assembly simulation on collaborative haptic virtual environments, Proceedings of 15th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2007, Plzen, Czech Rep., January 29-February 1, 2007, 241-247

[7] Mejía, R.; López, A.; Molina, A.: Experiences in developing collaborative engineering environments: An action research approach, Computers in Industry, 58(4), 2007, 329-346, DOI: 10.1016/j.compind.2006.07.009

[8] Huang, G.Q.: Web-based support for collaborative product design review, Computer in Industry, 48(1), 2002, 71-88, DOI:10.1016/S0166-3615(02)00011-8

[9] Zhu, W.; Lee, Y.-S.: Five-axis pencil-cut planning and virtual prototyping with 5-DOF haptic interface, Computer-Aided Design, 36(13), 2004, 1295-1307, DOI:10.1016/j.cad.2004.01.013

[10] Bordegoni, M.; Cugini, U.: Haptic modeling in the conceptual phases of product design, Virtual Reality, 9(2), 2006, 192-202, DOI: 10.1007/s10055-005-0013-3

[11] Duriez, C.; Andriot, C.; Kheddar, A.: Interactive haptics for virtual prototyping of deformable objects: snap-in tasks case, EuroHaptics, 2003, 159-175.

[12] Yang, Z.Y.; Lian, L.; Chen, Y.H.: Haptic function evaluation of multi-material part design, Computer-Aided Design, 37(7), 2005, 727-736, DOI:10.1016/j.cad.2004.08.010

[13] Lin, S.; Lee, Y.-S.; Narayan, R.: Heterogeneous Material Modeling and Virtual Prototyping with 5-DOF Haptic Force Feedback for Product Development, International Journal of Mechatronics and Manufacturing Systems, 1(1), 2008, 43-67, DOI: 10.1504/IJMMS.2008.018274

[14] Lin, S.; Lee, Y.-S.; Narayan, R.: Snapping Algorithm and heterogeneous bio-tissues modeling for surgical simulation bio-manufacturing, Virtual and Physical Prototyping, 2(2), 2007, 89-101, DOI: 10.1080/17452750701487941

[15] Lin, S.; Narayan, R.; Lee, Y-S.: Hybrid Client-Server Architecture and Control Techniques for Collaborative Product Development Using Haptic Interfaces, Computers in Industry, 61(1), 2010, 83-96, DOI: 10.1016/j.compind.2009.07.004

[16] Larsson, T.; Akenine-Moller, T.: Collision detection for continuously deforming bodies, In Eurographics, 2001, 325-333

[17] Memon, M.A.; Stevens, D.K.; Buttolo, P.; Oboe, R.; Hannaford, B.: Architectures for shared haptic virtual environments, Computers and Graphics, 21(4), July, 1997, 421-429, DOI: 10.1016/S0097-8493(97)00019-8

[18] Marsh, J.; Glencross, M.; Pettifer, S.; Hubbold, R.: A network architecture supporting consistent rich behavior in collaborative interactive applications, IEEE Transactions on Visualization and Computer graphics, 12(3), May/June, 2006, 405-416, DOI: 10.1109/TVCG.2006.40

[19] Iglesias, R.; Casado, S.; Gutierrez, T.; Barbero, J.I.; Sanchez, E.; Alonso, A.G.: Architectures for distributed multimodal virtual environments, Proceedings of the 4th International Conference on Manufacturing Research (ICMR 2006), Liverpool, UK, September, 2006

[20] Boukerche, A.; Shirmohammadi, S.; Hossain, A.: Moderating simulation lag in haptic virtual environments, Proceedings of the 39th Annual Simulation Symposium, Huntsville, Alabama, USA, April 2006, pp. 269-277, DOI: 10.1109/ANSS.2006.31

[21] Lin, S.; Lee, Y.-S.; Narayan, R.: Collaborative haptic interfaces and distributed control for product development and virtual prototyping, (in print), Accepted for publication in Proceedings of 2007 ASME International Manufacturing Science and Engineering Conference, Atlanta, Georgia, USA, October 15-18, 2007, Paper Number (CD): MSEC2007-31214, DOI: 10.1115/MSEC2007-31214
[22] Park, K.S.; Kenyon, R.: Effects of network characteristics on human performance in a collaborative virtual environment, Proceedings of IEEE conference on Virtual Reality (VR'99), Huston, Texas, USA, March 13-17, 1999, 104-111, DOI: 10.1109/VR.1999.756940
[23] Capin, T.K.; Pandzic, I.S.: A dead-reckoning algorithm for virtual human figures, Virtual Reality Annual International Symposium, Albuquerque, New Mexico, USA, May 1-5, 1997, 161-169, DOI: 10.1109/VRAIS.1997.583066
[24] Dyck, J.; Gutwin, C.; Subramanian S.; Fedak, C.: High-performance telepointers, Proceedings of the ACM Conference on Computer-Supported Cooperative Work, Chicago, Illinois, USA, November 6-10, 2004, DOI: 10.1145/1031607.1031636