COLLABORATIVE PRODUCT DESIGN AND MANUFACTURING BY WEB-BASED HAPTIC MODELING FOR VIRTUAL AND PHYSICAL PROTOTYPING

Yuan-Shin Lee1* and Shiyong Lin2

1,2Department of Industrial and Systems Engineering
North Carolina State University, Raleigh, NC, U.S.A.
yslee@ncsu.edu

ABSTRACT

In this paper, we present a collaborative product development and prototyping framework by using web-based distributed haptic interfaces along with network flow analysis. This paper presents detailed techniques on Collaborative Virtual Environment (CVE), which is considered as an emerging and promising technique for industrial product development and virtual prototyping. Both geometric and physical analytical modeling of heterogeneous product designs are presented for virtual prototyping and product development. 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 heterogeneous products are involved.

Keywords — Collaborative product development; Virtual and physical prototyping; Design for manufacturing, CAD/CAM, Web-based Haptic interface, Network control.

* Corresponding Author
1 INTRODUCTION

Virtual prototyping techniques have been applied in manufacturing systems ranging from product design, manufacturing process planning, virtual assembly and reverse engineering. Force and torque feedback play a critical role in identifying different materials in design and examining their physical properties in virtual prototyping and product development. To improve the virtual prototyping and product development, various haptic devices were introduced to enhance the realism of Virtual Reality (VR) environments by providing force feedback to the users. Figure 1 shows an example of using haptic device for product design, developed and presented in our earlier works [1, 2, 3]. Haptic interface can provide force and/or torque feedback to users in a virtual design and manufacturing environment. Haptic modeling techniques have been applied in the design and manufacturing systems ranging from product design, virtual prototyping, manufacturing planning and virtual assembly [4]. Force and torque feedback play a critical role in identifying different materials in design and examining their physical properties in virtual prototyping and product development. Force and torque feedback generation and rendering not only depend on the haptic interface but also on virtual objects modeling.

With the rapid progress of haptics technology, traditional visual-only VR systems have been gradually shifted to haptic based VR systems, e.g. haptic virtual sculpting systems and surgical simulators. Users can interact with virtual objects in virtual environments via haptic interfaces. Haptic interfaces facilitate the implementation of high-fidelity VR environments for users, by providing them with tactile senses. The main issue of integrating haptic interfaces into virtual environments is to generate realistic force/torque feedback. Material properties such as stiffness and roughness play a critical role in generating proper force/torque feedback during tool-object interaction. In our earlier work presented in [5, 6], the techniques of using a haptic device in the virtual sculpting and physical prototyping via NC machine tools have been developed. Figure 2 shows an example of haptic-based virtual prototyping and manufacturing planning system, developed in our earlier work [5, 6]. The haptic device has the capability of 6-DOF input and 5-DOF force-torque output in providing the users with 3-DOF force and 2-DOF torque feedback. The developed haptic interface allows direct hardware operation so that it is easier to generate and control the needed force feedback as presented in our earlier works in [1, 3]. As shown in Figure 2, by using the 5-DOF haptic interface device, a user is able to operate a virtual cutter to perform virtual sculpting and prototyping on the volumetric models. In the virtual sculpting process, the cutter is moved to carve the part surface manipulated by the user (see Figure 2). After the virtual sculpting and prototyping process, the multi-axis tool paths can be generated to simulate and verify the machining operations on a virtual CNC machine tool, as shown in Figure 2.
The advent of network communication technology has brought VR applications to a new horizon. Collaborative VR systems have been used in computer games, design and manufacturing. Haptic-based collaborative VR systems with deformable models have great potential in remote surgery training/operation, tele-robotics and other tele-applications. Research on integration and control of haptic interfaces and deformable models in collaborative virtual environments is still in its infancy. Network synchronization and intensive computation caused by haptic rendering and deformable object simulation are the main challenges to be addressed.

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 inconsistency of collaborative virtual systems. In addition, heterogeneous object simulation is notoriously time consuming, which can further deteriorate the stability of collaborative virtual systems.

In this paper, we propose a new method of analytical modeling and web-based haptic force rendering techniques for virtual prototyping and product development. 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 difference between the server and clients. Interpolation and extrapolation approaches derived from Verlet Integration are used to control and synchronize the distributed haptic devices.

2 HYBRID NETWORK ARCHITECTURE FOR THE COLLABORATIVE HAPTIC SYSTEM

In this section, system architecture of the proposed collaborative VR system is discussed. The objective is to solve the instability and inconsistency issues caused by the network communication problems for the proposed collaborative haptic virtual product development system. Several techniques are proposed in this chapter to deal with these problems to support the consistency, scalability and heterogeneity of the system.
As discussed in the Introduction, traditional client/server architecture is not ready for collaborative haptic application since heavy computational burden may deteriorate the performance of the server. The server would 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. However, if some tasks of the server such as collision detection are shifted to clients, computational burden at the server could be significantly reduced and system scalability might be improved.

Figure 3: Architecture of the proposed collaborative haptic virtual prototyping system

In this paper, we present a hybrid architecture extended from client-server architecture, which supports deformable models in haptic collaborative virtual environments. Figure 3 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 follows.

First, tool-object collision detection is performed at the clients (also at the server if the server 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.

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, computational requirements of 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, the haptic force is actually the byproduct of deformation simulation at the server. It can be calculated based on the spring force of contacted mass point by the haptic tool.

The simulation results such as 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 haptic devices also need to be updated to show their current locations. Based on the different roles of the server and clients, the proposed hybrid architecture has following three features:

**Consistency**: deformable object simulation and force calculation are processed at the server. Only one data set is maintained for the collaborative system. System consistency is automatically guaranteed. Clients copy simulation results from the server for their own graphic rendering and collision detection.

**Scalability**: new clients can be added to the collaborative system without bring much computational burden to the server. When a new client with a haptic device is added, only one more external force is added to the deformable object simulation at the server. The collision detection between the new added haptic device and deformable models will be performed at the new client. This will not affect the simulation efficiency of the server.

**Heterogeneity**: different types of haptic devices can be easily integrated into the collaborative system. Clients can have their independent collision detection and haptic render algorithms. They only need to provide haptic device positions and collision points if collision is detected.

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 networks, 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.

### 3 NETWORK DATA TRANSMISSION AND PROTOCOLS

Common problems in network communication include network delay, jitter (variation in the time it takes subsequent data packets to arrive) and packet loss, as illustrated in Figure 4. These problems may cause instability and incoherency of the virtual environment due to the delayed or information lost. In this proposed collaborative haptic VR 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. The data flow between the server and clients is shown in Figure 5. How to transmit these data over the network is determined by characteristics of these data.
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 such as [7].

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.
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 next section.

4 DATA SYNCHRONIZATION FOR ACCURATE COLLABORATION IN MANEUVERS

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 behaviour of users caused by the inaccurate or inconsistent information.

Figure 6: Minimizing network delay by artificial time compensation

Figure 6 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 6, we define the network delay as $\Delta t_{\text{network}}$. The time difference between sender and receiver sides of presenting or using the same packet is defined as $\Delta t_{\text{network}}$. With artificial time compensation, the time difference $\Delta t_{\text{variation}}$ is less than without artificial time compensation.
Generally $\Delta t_{\text{variation}}$ is equal to $\Delta t_{\text{network}}$ as shown in Figure 6(a). To alleviate the delay, an artificial compensation $\Delta t_{\text{artificial}}$ is imposed at the sender, similar to the enforced delay presented in [8]. The sender is enforced to delay $\Delta t_{\text{artificial}}$ 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 6(b). In our earlier work [6], adaptive time compensation is used instead of the fixed compensation. Whenever 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.

In network control, network jitter and data loss is more harmful than pure network delay to the haptic collaborative virtual environment [9]. 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. 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 7, 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 $\Delta t_{\text{artificial}}$ to decrease the time discrepancy between the sender and the receiver as shown in Figures 6 and 7. 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 500 Hz for haptic force rendering.
5 MISSING DATA COMPENSATION BY DATA INTERPOLATION AND 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 7, 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 7 also provides an example of out-of-order packets due to network jitter. P6 is generated and sent earlier 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 [10, 11].

In this paper, Verlet integration method is 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 calculate the missing data compensation, 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:

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

(13)

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

(14)

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 (13) to (14), Verlet position equation can be expressed as follows:

\[
X(t + \Delta t) = 2X(t) - X(t - \Delta t) + \dot{X}(t)\Delta t^2 + O(\Delta t^4)
\]

(15)

In Equation (15), 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 (15) as

\[
X(t) = \frac{1}{2} X(t + \Delta t) + \frac{1}{2} X(t - \Delta t) - \frac{1}{2} \dot{X}(t)\Delta t^2 - O(\Delta t^4)
\]

(16)

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

\[
\dot{X}(t) = \frac{X(t + \Delta t) - X(t - \Delta t)}{2\Delta t} - \frac{X(t - \Delta t) - X(t - 2\Delta t)}{\Delta t}
\]

(17)

Substituting Equation (17) into Equation (16), the unknown \( X(t) \) in interpolation can be interpolated by,
\[ X(t) = \frac{1}{4} X(t + \Delta t) + \frac{5}{4} X(t - \Delta t) - \frac{1}{2} X(t - 2\Delta t) \quad (18) \]

For the extrapolation case, the unknown \( X(t) \) can be estimated by previous three positions as follows,

\[ X(t) = 2X(t - \Delta t) - X(t - 2\Delta t) + \ddot{X}(t - \Delta t)\Delta t^2 + O(\Delta t^4) \quad (19) \]

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

\[ \ddot{X}(t - \Delta t) = \frac{X(t - \Delta t) - X(t - 2\Delta t) - X(t - 2\Delta t) - X(t - 3\Delta t)}{\Delta t} \quad (20) \]

From Equations (19) and (20), the unknown \( X(t) \) can be extrapolated by,

\[ X(t) = 3X(t - \Delta t) - 3X(t - 2\Delta t) + X(t - 3\Delta t) \quad (21) \]

Verlet method offers greater stability than the much simpler Newtonian methods with local 4th order error \( O(\Delta t^4) \), as indicated in Equation (15). The time interval \( \Delta t \) 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.

To estimating the missing data packet, the procedure retrieves the newest data from the local system. Locally generated data can always be acquired easily. If the data are generate remotely and sent to the local system, they are stored in the local buffers. The first thing is to decide whether the retrieved data is the newest one requested by the application. The criterion is the sequence number of the packet. If the packet has the correct sequence number, the packet is used to the application. Due to the network delay or data loss, the requested packet has not arrived in the buffer yet. Then, historical data in the buffer are used to extrapolate or predict the missing packet by Equation (21). On the other hand, if the newer packet than the requested one arrives in advance due to network jittering, interpolation is used to estimate the missing data packet by Equation (18). In the case of network delay or jittering, when delayed or jittered packets arrive late, they are used to replace the estimated ones in the buffer for approximating future missing data packets.

6 SYSTEM IMPLEMENTATION AND APPLICATIONS

The presented techniques and the discussed haptic system have been developed in our lab at North Carolina State University. The presented heterogeneous deformable modeling technique, the haptic controller and the haptic force rendering have been implemented on a dual-CPU workstation with a high-end graphic video card using Visual C++ and OpenGL® library functions. The lab-built 6-DOF haptic force feedback device was developed at the hardware level, which gives us the best flexibility for the haptic application development.

The collaborative VR system allows users to design, manipulate and modify a product design collaboratively at either local or remote site. Figure 8 shows the implemented collaborative haptic VR system for product development. Three users are operating haptic devices to design the same product collaboratively via the network. 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. Users can test product’s physical properties like stiffness and evaluate its functions intuitively [12]. These tasks can be done individually by a single user or collaboratively by several users. Figure 8 shows an example of assembling components via the collaborative haptic network by multiple users. Each user moves a component independently. During the operation, collision forces between components are passed to users via their haptic devices.
In this paper, network emulator software NetDisturb is used to simulate network traffic in a local network. Figure 9 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. For example in Figure 9, Packet 62 (containing haptic positional data) was sent 30ms earlier than Packet 63. Due to the network jitter, Packet 63 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 9. The zig motion of the haptic tool can be easily observed at the client.
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 10 and 11 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 10(a)-(b). Although the two parts are moved towards each other and actually aligned in Figure 10(c), the client still sees they are not aligned as in Figure 10(d). So the client will keep moving up its part. As a result, the alignment does no longer exist as shown in Figure 10(e). Then the server may move up its part again for a new alignment in Figure 10(g). However, the client will notice the unaligned assembly in his/her screen as shown in Figure 10(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 11, 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.
Figure 10: Haptic data jitter and data transmission time with and without network time delay

Figure 11: Haptic tool motion jittering problems
To minimize the time discrepancy between the server and the client, adaptive artificial time compensation is enforced at the server. Figure 12 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 chapter, 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 12, 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.

![Figure 12: Comparison between fixed artificial time compensation and adaptive artificial time compensation](image)

### 7 CONCLUSION

In this paper, a new method of collaborative virtual product development and virtual prototyping framework is presented, which integrates different haptic interfaces and heterogeneous deformable models. A hybrid network architecture extended from client-server architecture is developed to balance the computational burden between clients and the server. Several techniques have been presented to deal with network control problems for collaborative VR systems. Adaptive artificial compensation is used at the sender to compensate the time discrepancy between the sender and the receiver. Interpolation and extrapolation approaches by Verlet integration are used to synchronize the transmitted data. The proposed techniques can be used in collaborative product development, virtual sculpting, virtual assembly, remote surgical simulation and other collaborative VR applications. The presented techniques can be used to improve the effectiveness of the product design, virtual prototyping, product development, and other virtual reality applications.
8 ACKNOWLEDGEMENTS

This work was partially supported by the National Science Foundation (NSF) Grants (CMMI-0553310, CMMI-0800811, IIS-0905505, CMMI-1125872) to North Carolina State University. Their support is greatly appreciated.

9 REFERENCES


