A Peer-to-peer Architecture for Collaborative Haptic Assembly

Rosa Iglesias, Sara Casado, Teresa Gutiérrez
LABEIN TECNALIA
48160 Derio-Bizkaia, Spain
{riglesias, sara, tere}@labein.es

Alejandro García-Alonso
University of the Basque Country
20018 Donostia-San Sebastián, Spain
alex.galonso@ehu.es

Kian Meng Yap, Wai Yu, Alan Marshall
Queen's University of Belfast
School of Electrical & Electronic Eng.
BT9 5AH – Belfast, UK
{m.yap, w.yu}@qub.ac.uk, a.marshall@ee.qub.ac.uk

Abstract

Virtual Environments using haptic devices have proved useful for assembly/disassembly simulation of mechanical components. To date most haptic virtual environments are stand-alone. Collaborative Haptic Virtual Environments (CHVEs) are distributed across a number of users via a network, such as the Internet. These present new challenges to the designer, such as consistency of the virtual environments, user-user haptic interaction, and scalability. The system described in this paper considers the CHVEs to be distributed over a packet-switched network such as the Internet. It gives priority to the validation of interactions between objects grasped by users; guarantees consistency across different users' virtual environments. The paper explains the components used and the consistency-maintenance scheme that guarantees the consistency of the virtual scene in the remote nodes. Consistency and force feedback results are also discussed. Results presented show the system maintains a consistent and satisfactory response when network incurs delay or packet jitter.

1. Introduction

There is currently a great deal of interest in the use of haptic devices in Distributed Virtual Environments (DVEs). Moreover, several results from different authors [1, 2] have shown that force feedback enriches human-computer interaction and significantly enhances task performance. This interest, coupled with the availability of new lower cost haptic devices, is enabling a range of new applications. These applications include education and training simulators [3], assistive technology for visually impaired people [4, 5], entertainment and gaming [6], cultural heritage applications [7], as well as industrial design and maintenance [8, 9, 10]. Moreover, the knowledge gained with these environments is leading to develop innovative and new generations of haptic devices.

New distributed interactive applications are also emerging, such as Collaborative Haptic Virtual Environments (CHVEs), or the use of haptic devices in collaborative real-time applications. Many different examples can be found: work in a remote location (tele-operation) [11], haptic guidance for training and teaching (tele-mentoring) [12, 13], and haptic collaboration for other applications.

As in single-user applications, the effects of haptic feedback on a collaborative task performance have been studied. In general the results show that adding force feedback to a collaborative virtual environment (CVE) enhances effectiveness and reduces the time required to complete a task [14, 15]. However there are still major challenges in the development of CHVEs, including: virtual scene synchronization (consistency), the provision of a realistic haptic feedback (quality of force feedback) and scalability. Several attempts have been made to deal with the problem of consistency in DVEs [16]. Additionally, different tests have demonstrated how network impairments (i.e. delay, jitter, message loss and background traffic) affect the force feedback and cause instability [17, 18].

Networked virtual environments require the exchange of information between users. To achieve this there are various communication architectures:

client-server, peer-to-peer or a hybrid mixture. Client-server architectures have the advantage that consistency is more easily achieved [19], however the local representation of the scene is only updated after a round-trip delay to the server; whereas, in peer-to-peer architectures, although ensuring consistency among peers is a challenge, the update delay is only one-way. In CHVEs it is important to minimize the impact of the network on the quality of force feedback; a key factor is therefore to rapidly update the virtual scene for fast force feedback. Furthermore, additional factors must be addressed, such as, reducing the number of messages and parallel computation (i.e. reducing sequential bottlenecks).

Buttolo [19] have addressed the issue of maintaining the consistency of each user's database by limiting interactions to one user at a time. Hespanha [20] uses the concept of object ownership and restricts interactions among users. Marsh [21] proposes a multiple-server architecture with only one active server at a time, which automatically assures the consistency but limits the haptic interaction with the scene to only one user in case of high-delay collaboration. However, the case where several users simultaneously manipulate different objects within the shared virtual scene and may feel other user's interactions has not been considered to date.

So, there is an open challenge: to design an efficient framework, which despite network impairments, allows a truly and realistic interaction where several users haptically interact within the same scene. A peer-to-peer architecture is considered because it addresses the two main challenges: scalability and user-user haptic interaction because peer-to-peer topology may also have better performance in terms of haptic feedback. The goal of this research is to design, implement and evaluate a consistency-maintenance scheme. The framework is validated by a new system that allows two users to simultaneously manipulate the same shared virtual scene by: touching objects, grasping objects and moving them, detecting collisions with other objects or even assembling. This work does not consider that an object can be grasped by more than one user. Some situations present an additional challenge to achieve consistency and realistic force feedback, such as, when a user assembles an object into another one grasped by a different user, or when collisions between grasped objects occur. These types of situations are analyzed in terms of consistency maintenance and haptic feedback.

Despite different network conditions, the preliminary results of the proposed architecture show that synchronization at both sites is achieved, and the quality of force feedback is reliable and stable; however, more refined strategies must be implemented in order to smooth forces in more complex scenes/tasks.

For the remainder of this paper, the terms; object, CAD component or component will be used synonymously throughout the text, and the term position will include both the translation and orientation of an object. Moreover, to improve legibility, assembly will mean both assembly and disassembly.

2. The collaborative system: CHAS

The Collaborative Haptic Assembly Simulator (CHAS) is based on a single-user system but is distributed in its operation. This section explains this single-user system, the actions which the user can perform in CHAS and identifies some design challenges addressed by this research. It also describes the messages required to maintain consistency at both users. The next section presents the consistency maintenance scheme. The final last section analyzes the system and presents results from consistency and usability experiments.

2.1. The single-user system (HAS)

The *Haptic Assembly Simulator* system (HAS) is described in greater detail in [22], a brief description of the system modules are introduced to understand the concepts used in the next sections.

The single-user system, HAS is comprised of an *Assembly Simulator* and a *Haptic Assembly Simulator*. It provides interaction with the virtual scene by means of traditional devices (i.e. keyboard, mouse) or haptic devices, such as, Phantom Premium, Omni (Sensable Technologies [23]) or Grab haptic device (PERCRO [24]). Both simulators are based on ***DAT***um, which is a geometric modeler developed by LABEIN, with a STEP (International Standard for the representation and the exchange of product data between different CAD systems) and VRML translator [25]. ***DAT***um allows the creation of 3D virtual scenes or, for example, automatically imports a 3D CAD component to simulate assembly and maintenance operations. ***DAT***um has been successfully applied to different applications, including haptic audio virtual environments for blind people [5] and interactive virtual prototyping [25].

The objects created or imported into the system can be virtually touched or grasped by the user. The following actions are considered: touch, move grasped object and assemble/ disassemble a CAD component.

Touch is performed when the user has no object grasped. When the user moves the haptic device, a virtual point moves within the virtual scene. The user can touch any object and move along its external surface, detecting its edges and corners. The haptic interaction with a virtual object is based on analysis of the position of the user's finger (provided by the haptic device) with respect to the object, by checking if this position is inside or outside the object. When inside it, the force sent by the haptic device will be proportional to the penetration depth of the pointer into the virtual object and normal to the object surface.

The user can also *grasp* a virtual object. The position of the grasped object is correspondingly updated with the position of haptic device. As the grasped object is moved the system must check for inter-object collisions. If the object in the new position collides with any other object, a collision force feedback is produced in response. In this application all objects are static except the one grasped.

2.1.1. Assembly Simulator. The *Assembly Simulator* allows inter-object collision detection and automatic recognition of potential assembly constraints between a grasped object controlled by a user and the rest of the components of the mechanical assembly within the virtual scene. An object is moved according to the position of the haptic device, mouse or by means of the keyboard. Before applying a new position (translation and/or rotation) to an object, it must be analyzed if the object with the new position collides with any other object of the virtual scene. Part of the collision detection algorithms are based on the RAPID library implemented by the University of North Carolina [26] and others have been implemented for better performance.

During an assembly the system automatically recognizes potential constraints between the grasped component (being moved by the user) and the rest of the components within the virtual scene. Once the system detects a constraint, the movement of the component will be constrained to satisfy the active constraint. During the assembly task, collisions with other models of the virtual scene are also detected [25, 27, 28]. Two generic assembly methods have been implemented:

- Assembly/Disassembly objects along a common axis: pin-hole, pin-multiple holes, hole-pin, pin-pin and hole-hole.
- Assembly/disassembly objects along coincident planar faces.

2.1.2. Haptic Assembly Simulator. This module allows the user to interact with the components of the assembly in a more realistic way, using haptic devices. HAS provides the force feedback according to the user action or interaction mode: touch, grasp, move, collide, assemble or disassemble. The real-time collision detection and the recognition and management of the assembly constraints incur a high computational cost. In order to provide the user with a real feeling of interaction with the scene and maintain the control loop required by haptic devices (1 kHz) it is necessary to use a multi-threaded framework.

Two main threads are considered as shown in Fig.1:

- The haptic thread running at 1 kHz deals with the communication with the haptic device receives the movement requests by the user and calculates the force to be sent to the haptic device.
- The simulation thread is responsible for validating the movement requested by the user (i.e. check if a collision occurs or if a new constraint is active, and constraining the movement if necessary). This thread runs with the highest priority and its rate depends on the complexity of the virtual scene, the user action and CPU processing power.

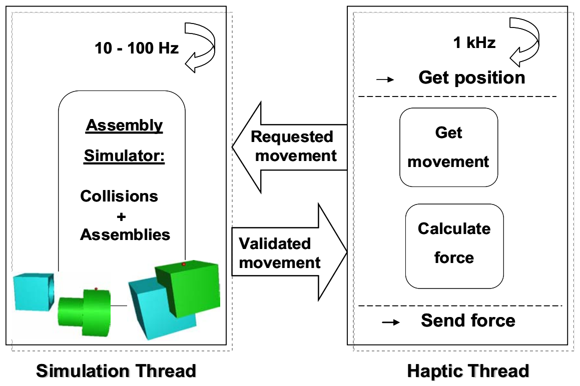


Figure 1. Simulation and haptic threads

Because the haptic thread runs faster than the Assembly Simulator, several strategies must be implemented to avoid brusque changes in the force sent to the user. The haptic thread performs interpolation of haptic rendering when there is no information has come from the Assembly Simulator and when new information has been received.

In order to determine the appropriate force feedback it is also necessary to take into account the user action and the status of the system. For instance, if the system detects an assembly constraint, the constraining feedback force to the movement is calculated based on the constraint conditions. In this way, the system helps the user to align and orient the grasped component to perform an assembly task. This is necessary otherwise this task may be very complex to perform. Three different types of forces are considered for assembly constraints:

- *Sliding forces on a surface* → allows two objects to remain in contact along a common surface.

- *Sliding forces on a line* → allows two objects to remain in contact along a common line.
- *Sliding forces on a point* → allows two objects to remain in contact along a common point.

2.2. Collaborative Haptic Assembly Simulator (CHAS)

CHAS has the same functions as the single-user system. However, in this new system two users may haptically and simultaneously interact within the same virtual environment. In this framework, for a fast haptic interaction response, each user has their local copy of the virtual scene and haptic rendering is performed at each user's computer. Depending on the user interaction with the virtual scene, the following cases are defined:

1. Both users only *touch* objects. No data is exchanged because no object changes its position in the virtual scene.
2. 'User A' grasps and *moves* an object. Moving a grasped object involves interacting with the scene, for instance, move, assemble, collide and so on. Meanwhile 'user B' only performs *touch* actions. Data is sent out from A to B in order to update user B's scene.
3. Each user grasps and *moves* an object, that is, two different objects are moved. In this situation two different cases are defined:

3.a It may happen that two users work independently without disturbing the other user's work, that is, without collision between the two grasped objects (*independent interaction).*

3.b Or it may happen that both users work near each other and both grasped objects collide with each other. In this case, two problems arrive: consistency maintenance and providing an appropriate force feedback (*dependent interaction).*

To maintain consistency the CHVE must avoid each user working with a different representation of the virtual scene. A user action affects the representation of the virtual scene at other users and therefore, one of the main challenges in CHVEs is maintaining a consistent virtual scene in the presence of inevitable network delay, packet jitter or other network conditions.

A multi-threaded solution was employed in the single user system to avoid degradation of the haptic response (the haptic loop).The haptic thread provided smooth forces either when the simulation thread had not received a new valid position available, or when the new position generated brusque force changes. However, in CHAS, even with acceptable network communications the creation of a third loop is required, which adds an additional problem to the goal of achieving a realistic and reliable force feedback. The computation of force feedback is more demanding when users interact with each other, that is, when two grasped objects collide, or a grasped object must be assembled into the object grasped by the other user (dependent interaction, case 3.b). Additionally, network effects can only exacerbate this.

These problems are addressed in two ways: (i) by adapting the system architecture to cope with updated messages and consistency-maintenance, and (ii) developing algorithms for smoothing the haptic force. These algorithms are required, since force feedback is not simply determined by one user interacting with the virtual scene, but by the position of the other user's object (dependent interaction).

Messages are used to inform remote users of required scene updates. At any one instant of time each user knows:

1. the position of its grasped object, from here onwards referred to as: the *local object* grasped by the *local user*, and
2. the current valid position of the other user's grasped object, from here onwards referred to as: the *remote object* (grasped by the *remote user*).

The messages that contain updated states have the following data:

```
{
   local object identifier,      remote object identifier
   local object translation,     remote object translation,
   local object rotation,        remote object rotation,
   local sequence number,        remote sequence number      };
```

Such messages contain information about local and remote objects at each user. The identifiers are used to identify which object of the virtual scene is being moved, whereas the sequence number is used when discarding duplicated or out-of-order messages. The remote object position is the position of remote object at this user.

From the point of view of data transfer the following characteristics were considered to be important: small messages and low message frequencies. The size of the messages sent in each cycle is small (less than 200 bytes). To minimize network traffic, update messages are made only when there is a change in the local grasped object.

3. Consistency-maintenance in CHAS

In this section, the scheme for maintaining consistency in CHAS is explained. This framework permits two different users to interact haptically and simultaneously with the same virtual scene to undertake assemblies and maintenance operations.

3.1. Consistency-maintenance scheme

An important feature of this system is that object movement is validated only once, at the user where it is moved. The position of remote objects is also taken into account in a way that scene consistency is guaranteed.

As explained in Section 2, when a user has grasped an object, it is moved according to the user position (haptic device). This position is sent to the Assembly Simulator which is in charge of validating it. If it does not detect any collision, the object position is updated and haptic thread receives that the movement is valid. A new message is sent out by the simulation thread only when a new local position is validated. In this way the message rate depends on both the frequency of the Assembly Simulator and on the result of the validation (collision or no collision).

In CHAS, the simulation thread must deal with the validation of the current position of the local object and the remote object's position received from the remote user. This is done independently at each user using the same architecture. The first thing to do is to check whether or not any message is pending. The system reads all pending messages and the message with the newest updated position is taken by using the sequence order number. Three different cases are considered within the simulation thread:

A. ***LOCAL MOVEMENT*** "Local moves, remote does not": no remote message was received. As in the single-user case, the *Assembly Simulator* checks for collisions and assemblies with other objects of the virtual scene. In case of no collision, the position of the object is updated and a message containing updated data is sent to the remote user. If a collision occurs, neither position is updated nor is any message sent out to the other user.
B. ***REMOTE MOVEMENT*** "Remote moves, local does not": a new remote position has been received and the local object remains in the same position. Firstly, the remote object is updated with the received position. This remote object position had previously been validated at the remote site with a delayed position of local object and the rest of static objects, so the system must only check collisions and assemblies between both grasped objects. If a collision occurs, the local object position is transformed to the position stored in the message. To summarize, in the case of a collision, both positions are changed to the positions that have been sent by the remote user.
C. ***LOCAL AND REMOTE MOVEMENTS*** In this case, CHAS must validate the local object and remote object positions, which may increase the completion time of simulation thread. In order to increase the efficiency and provide an adequate and reliable force feedback, tasks with low computational cost should be carried out first (i.e. analysis of collision between only local and remote objects).

In case C), as collisions must be detected before the assembly constraints are applied, consistency is evaluated in two steps:

1. Check collisions with remote object. This is done to achieve a fast haptic response when both objects collide. In case of a collision between both grasped objects, this collision information rapidly passes to haptic thread because only both grasped objects are analyzed.
2. If no collision was found in step 1, check collisions with the rest of the objects within the scene. If no collision occurred, then carry on detecting assembly constraints. Assemble computation is first done with remote object (for fast haptic response when both objects are together being assembled).

The following paragraphs describe this strategy in more detail. User A (local user) moves the grasped object A (OA), and user B (remote user) moves the grasped object B (OB).

When the validation cycle starts, the following data is defined:

$\mathbf{P}_{\mathbf{AlastValid}}$:	last valid position of O_A at user A.
$\mathbf{P}_{\mathbf{Anew}}$:	new position for O_A at user A (from HAS). Note that, this position has not been validated yet.
$\mathbf{P}_{\mathbf{Breceived}}$:	new position for O_B (received from B) at user A. Note that, it was validated at user B.
$\mathbf{P}_{\mathbf{Areceived}}$:	one position of O_A received from B. This position corresponds to a previous position of O_A. This position went from A to B and came from B in the same message as $\mathbf{P}_{\mathbf{Breceived}}$ (see message data), so it was a valid position

The simulation thread uses this data to achieve consistency, that is, to select a new pair of valid positions for OA and OB. In general, the new pair will be $\mathbf{P}_{\mathbf{Anew}}$ and $\mathbf{P}_{\mathbf{Breceived}}$, but sometimes this pair is not valid because both objects collide at such positions, so a different pair must be considered. This goal requires the following steps.

Check if OA placed at $\mathbf{P_{Anew}}$ and O_B placed at $\mathbf{P_{Breceived}}$ are colliding or not:

a. In case of collision: check if O_A and O_B collide at positions $\mathbf{P_{AlastValid}}$ and $\mathbf{P_{Breceived}}$.
   1. <u>Collision</u>: Note that the pair ($\mathbf{P_{Areceived}}$, $\mathbf{P_{Breceived}}$) is a valid one because only valid pairs are transmitted. So they are the new valid positions at user A. A new message is sent with this data (CASE 1).
   2. <u>No Collision</u>: The new valid pair is ($\mathbf{P_{AlastValid}}$, $\mathbf{P_{Breceived}}$). No message is sent because $\mathbf{P_{AlastValid}}$ was previously sent (CASE 2).

b. In case of no collision: check if O_A placed at $\mathbf{P_{Anew}}$ collides with any other object.
   1. <u>Collision</u>: check if O_A and O_B collide at positions $\mathbf{P_{AlastValid}}$ and $\mathbf{P_{Breceived}}$: proceed like in case a. When no collision, this object will remain at the last position validated by this user, instead of taking received position by the remote one.
   2. <u>No Collision</u>: the pair ($\mathbf{P_{Anew}}$, $\mathbf{P_{Breceived}}$) is a valid one. A new message is sent with this data (CASE 3).

3.2. CHAS design discussion

Previous research has examined the implementation of HAS with a client-server architecture [10], however CHAS uses a peer-to-peer architecture. The main reason for considering P2P is scalability, although it may also have better networked performance. Haptic interaction is particularly affected by network impairments (i.e. delay, jitter) when using client-server architectures because of the potentially longer round trip times that are incurred with this architecture.

In a peer-to-peer system, the use of update messages is insufficient to support synchronization of a CVE, particularly when several users may be interacting with the virtual scene And additional mechanisms are required to ensure a consistent view of the virtual scene. Mauve *et al*. [16] have addressed this issue by delaying update messages to the slowest network node. In a more recent research [29], the same authors have dealt with this issue by artificially delaying local interaction (increasing consistency) and using a time-warp algorithm (move forward through time to a valid position) to guarantee consistency. However, such mechanisms are more focused on visualization rather than on haptic interaction. Other authors [20] have dealt with this challenge by using the concept of ownership and taking into account the user's network delay. In situations where, a close and dynamic interaction between the users was required, a leader was in charge to prevent discontinuities during the interaction.

In the architecture presented in this paper, messages sent out from a user to the other peers contain information about local and remote objects at each user. They always contain a pair of valid positions, because messages contain only valid pairs validated by the sender. This pair of valid positions sometimes plays an important role to set a valid position: in case of collision between both grasped objects, the local object must be 'rolled back' to such position received in the pair. Instead of storing a series of positions, this architecture uses this 'echo' to obtain a previous valid local position.

In CHAS, while both users move their grasped objects avoiding interferences between them, consistency is guaranteed. But when a collision occurs between the grasped objects, consistency must be assured. The main challenge is to decide which object's position is required to be rolled back to the local object to a consistent state and the result must be rapidly communicated to the other user. The criterion used in this system is based on providing a fast haptic response. This is done in two fast steps (Section 3.1 explains this more formally).

As CHAS only stores the last valid local position, that position is tested for collision. If this one-step local roll back does not provide a valid pair, a different roll back strategy is used. Therefore, positions which are stored in the last message received are used, that is, the positions of the remote object and the local object are updated with the last received positions from the other user. Moreover, another issue is presented: how to cope with a brusque change of the position that may cause large forces that should be avoided.

The experiments and results presented in next section demonstrate that consistency is successfully achieved under different network impairments and force feedback is acceptable with specific network conditions.

4. Experiments and results

The experiments have two goals: (i) to verify that consistency is successfully guaranteed and (ii) to study how positional discrepancies of the same object at both users may affect force feedback when both grasped objects collide (dependent interaction). We define positional discrepancy as the displacement between an object's position at one user's scene and the last received position of the same object (e.g. the distance between $\mathbf{P_{screw}}$ and $\mathbf{P_{screw\ received}}$).

4.1. Assembly virtual environment

In this collaborative virtual environment, each user is presented with the same virtual assembly scene (Fig. 2). This figure consists of different components of a CAD model.

The experimental platform is composed of three co-located computers via a 100Mbps link (Fig. 3). Two computers are each connected to an OMNI device. The other computer was used to simulate the different values of delay and jitter (network emulator software NetDisturb [30]).

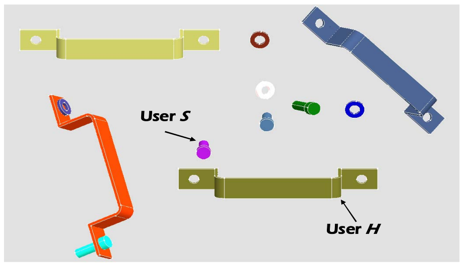


Figure 2. Assembly collaborative VE

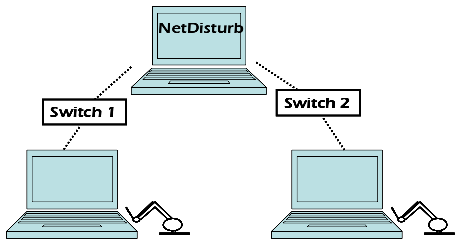


Figure 3. Experimental platform

During the experiments, user S grasped the virtual screw and user H did the same with the handle. Both users took their objects to collide on purpose with other static objects within the virtual scene (independent interaction). After several collisions, both users collide with each others' grasped objects. Finally, user S began the assembly sequence (involving the grasped screw and handle) until the screw is assembled into one of the holes of the handle. The latter two situations imposed dependant interaction and each user felt the corresponding force feedback.

4.2. Results

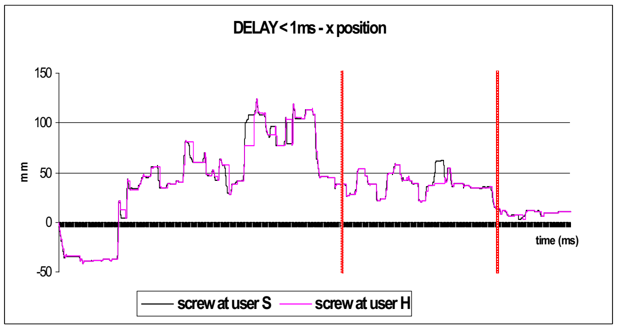
Generally, at any instant of time the screw's position at user S is not equal to the screw's position received in the last message from user H. Such differences in position might be smaller or larger depending mainly on the network conditions. With this measure, the preliminary results show that consistency is guaranteed. The positional discrepancies are analyzed with different network conditions. This allows analysis of the quality of forces, and as a consequence, the design of strategies/algorithms that support an adequate haptic feedback in case of dependent interaction. These forces depend mainly on positional discrepancies which in turn are affected by the network impairments.

4.2.1. Consistency. We consider that consistency between both scenes is achieved when such position discrepancies are corrected along time and position divergence does not tend to continuously increase.

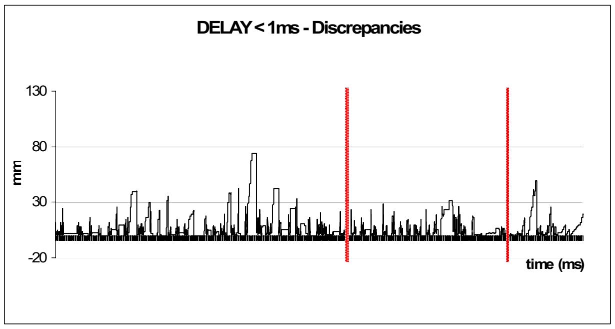
Figure 4 (a, c) illustrates how the synchronization is achieved with different end-to-end network delays: less than 1ms and 100 ms. Each series represents the different positions of the same object (screw) in x axis at both users (to simplify just positions along the x-axis are shown since it was the predominant movement in the undertaken task). These positions were measured and stored at user S once the simulation thread had finished. Therefore, the local updated position of screw was compared with the remote received position. In the experiment the dimension of the scene was 270 W x 180 H x 150 D mm. White vertical lines on each graph correspond to the change from independent interaction to dependent interaction and the last stage denotes when the users were performing the assembly operation.

In Figure 4 (b, d) the positional discrepancy of the screw at both users is shown. As expected, the discrepancies increase with increased network latency, however such differences decrease after a while. We can therefore conclude that consistency is successfully guaranteed in all cases.

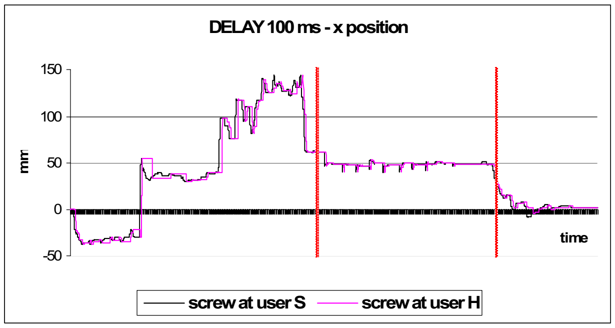
4.2.2. Discrepancy vs. network impairments. In this section, we quantify the position discrepancy as a function of the network conditions. Different experiments were carried out using the same VE explained above. Figure 6 shows the results in case of dependent interaction with different conditions of the network. That is, within the simulation thread when a collision is detected with the other grasped object, the local grasped object is moved to a valid position, the received position from the remote user. After a session with several collisions between both grasped objects, we measured the mean and the maximum values.



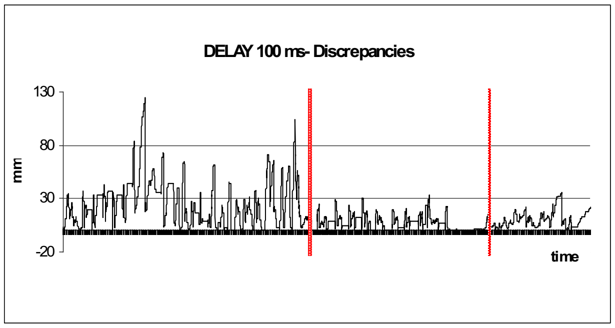
a)



b)

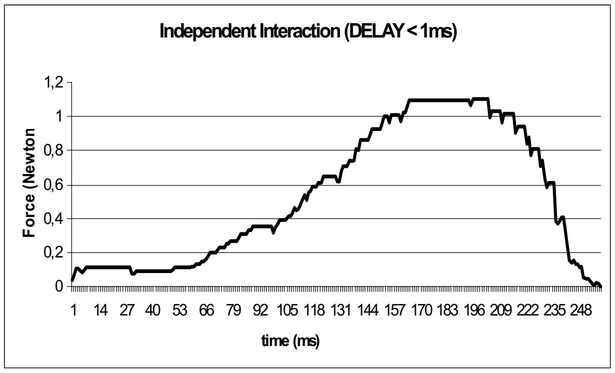


c)

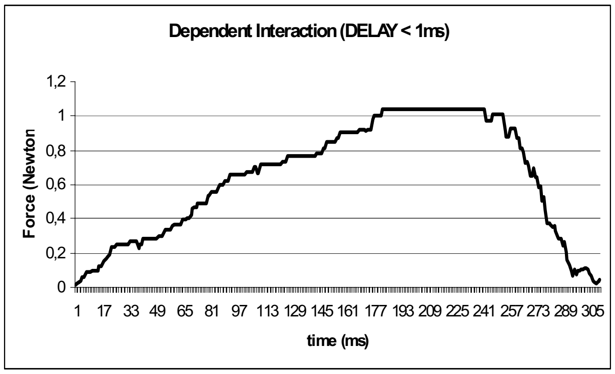


d)

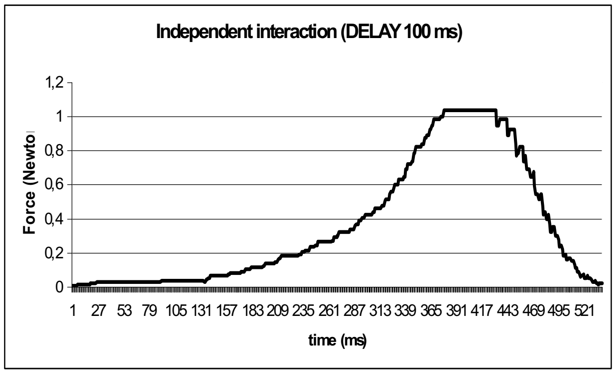
Figure 4. X-position of screw S (left) and positional discrepancies of screw at both users (right)



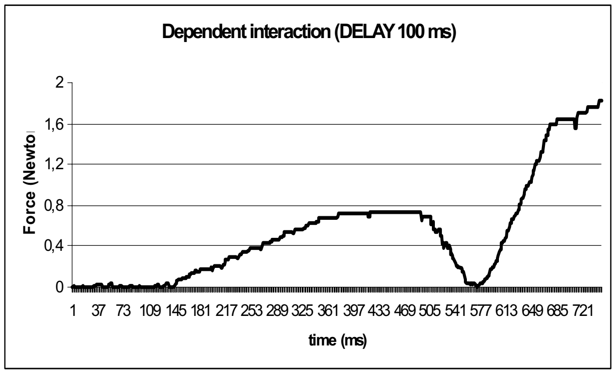
a)



b)

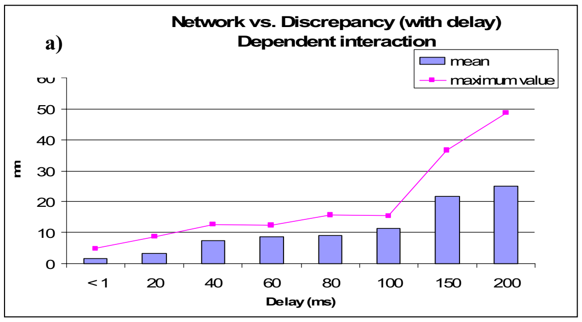


c)



d)

Figure 5. Force magnitude in case of collision



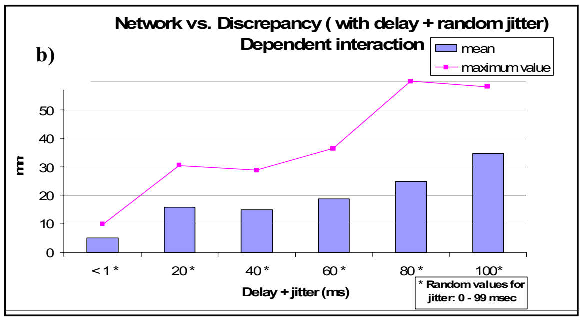


Figure 6. 'Rolled back' discrepancies with different network conditions in case of dependent interaction.

In general, discrepancies are quite unpredictable, because they may depend on: how fast a user moves the object if both users were nearby before the collision occurred, haptic device and so on. In Figure 6.a., it can be observed how discrepancies increase with the end-to-end network delay. As expected, the considered values for jitter and delay (Fig. 6.b.) significantly impact on the displacements (in the worst case, with jitter and delay, the maximum delay may be 200 ms).

4.2.3. Forces. A collision causes a collision force feedback to be sent to the users. In case a user's object collides with any static object (independent interaction), the collision force provided to user *S* at each millisecond is shown in Figure 5.a and c. This collision force feedback is smooth enough and provides a realistic and reliable interaction. Moreover, in case of independent interaction this force is not dependent on the network conditions, as can be observed from the relative similarity between Figure 5.a and c.

On the other hand, a collision with another grasped object (dependent interaction) may cause unstable haptic interaction. This force can be affected by the unpredictable network conditions. The case of 100 ms delay clearly shows that the force is degraded (Fig. 5.d). In case of no delay (Fig. 5.b), haptic interaction was stable and the positional discrepancy when the collision occurred was less than 1 mm; whereas, with 100 ms delay, the positional discrepancy was 8 mm. New algorithms must be implemented to provide a reliable interaction, in these cases where this displacement is significant. Although a case of force degradation has been presented with 100 ms delay, different experiments revealed that this does not always happen. Sometimes this force followed the same pattern as other cases and those situations corresponded with small discrepancies (less than 6 mm). On the other hand, it was observed that the collision time generally increased in case of dependent interaction with the worsening of the network performance.

When user S assembled the screw into one of the handle holes and no collision occurred, the force feedback was the same as in the case of independent interaction, but the assembly constraint might be broken, if the user, who grasped the handle, moved it too far.

5. Conclusions

This paper has described the criteria used to build a collaborative haptic virtual environment (CHVE) for distributed assembly/disassembly simulation over networks. The CHAS architecture has been tested in different network impairments and validated for the different user tasks.

A major challenge in the design of distributed collaborative virtual environments is maintaining consistency. A consistency-maintenance scheme has been developed that gives priority to the analysis of collisions between objects grasped by users. This strategy has produced good results: a user can assemble an object into another object grasped by a different user. As the experiments show, this can be achieved even with certain level of delay and jitter.

CHAS has an advantage in that it does not need to store and manage a history of movements. CHAS has no specific messages to deal with inconsistency because each user manages consistency locally. However, in case of an inconsistent state, the user's scene rolls back to the last consistent state reported by another user.

The framework used in this VE can be extended to a multi-user (i.e. >2) peer-to-peer system. Only dependent interaction (case 3.b) places special limitations for a multi-user system. When dependent interaction involves only two users, the system should behave as reported in this paper. However, it remains to be analyzed, how the architecture behaves when three or more users are involved in dependent interaction.

Acknowledgements

This work has been partly supported by ENACTIVE Interface, European project IST-2002-002114; Gipuzkoako Foru Aldundia project (13/2005) with FEDER funds; and AMIGUNE Basque Government project (2005-2007).

References

[1] M.A. Srinivasan, and C. Basdogan "Haptics in virtual environments: Taxonomy, research status and challenges" Comput. Graph. 21, 4, 1997, 393-404.
[2] G.C. Burdea and P. Coiffet. Virtual Reality Technology. John Wiley and Sons, Inc, 2nd ed. 2003.
[3] C. Basdogan, C. Ho, M.A. Srinivasan, "Virtual Environments for Medical Training: Graphical and Haptic Simulation of Common Bile Duct Exploration", IEEE/ASME Transactions on Mechatronics (special issue on Haptic Displays and App.), Vol. 6, No..3, 2001, 267-285.
[4] W. Yu, R. Kuber, E. Murphy, P. Strain and G. McAllister, "A novel multimodal interface for improving visually impaired people's web accessibility" Virtual Reality Springer-Verlag. 2005.
[5] R. Iglesias, S. Casado, T. Gutiérrez, J.I. Barbero, C.A. Avizzano, S. Marcheschi and M. Bergamasco. "Computer graphics access for blind people through a haptic and audio virtual environment". HAVE 2004 - IEEE International Workshop on Haptic Audio Visual Environments and their Applications ,2004, 13-18.
[6] C. Swindells A. Unden and T. Sang "TorqueBAR: an ungrounded haptic feedback device". Int. Conference on Multimodal interfaces SIGCHI ACM, 2003, 52-59.
[7] M. Bergamasco, A. Frisoli and F. Barbagli. "Haptics technologies and cultural heritage applications", Computer Animation, 2002, 25-32.
[8] M. Bordegoni and U. Cugini "Create free-form digital shapes with hands" International conference on Computer graphics and interactive techniques in Australasia and South East Asia GRAPHITE. 2005, 429-432.
[9] D. Borro, J. Savall, A. Amundarain, J.J. Gil, A. García-Alonso and L. Matey, "A Large Haptic Device for Aircraft Engine Maintainability", IEEE Computer Graphics and Applications, Vol. 24, Nº 6, 2004, 70-74.
[10] R. Iglesias, S. Casado, T. Gutiérrez, A. Carrillo, J.I. Barbero and A. García-Alonso, "Analysing different architectures of a distributed environment for assembly simulation", Virtual Concept, Biarritz (France), 2005, 8-10.
[11] R. Ott, M. Gutierrez, D. Thalmann and F. Vexo, "VR Haptic Interfaces for Teleoperation: an Evaluation Study" IEEE Intelligent Vehicles Symp. IV'05, 2005, 789-794.
[12] D. Wang, L. Ni, M. Rossi and K. Tuer, "Implementation issues for bilateral tele-mentoring applications", Haptic, Audio and Visual Environments and Their Applications, HAVE 2004, 75-79.
[13] D. Feygin, M. Keehner, and R. Tendick, "Haptic guidance: experimental evaluation of a haptic training method for a perceptual motor skill", Haptic Interfaces for Virtual Environment and Teleoperator Systems HAPTICS 2002, 40-47.
[14] E.-L. Sallnäs "Supporting Collaboration in Distributed Environments by Haptic Force Feedback" ACM Transactions on Computer-Human Interaction (ToCHI) Vol. 7, Issue 4, 2000, 461-476.
[15] EE Xiao Shen, J. Zhou, A. El-Saddik and N. D. Georganas, "Architecture and Evaluation of Tele-Haptic Environments", DS-RT 2004, 53-60.
[16] M. Mauve, "Consistency in replicated continuous interactive media", ACM Conference on Computer Supported Cooperative Work (CSCW) Philadelphia, PA, 2000, 181–190.
[17] S. Matsumoto, I. Fukuda, H. Morino, K. Hikichi, K. Sezaki, and Y. Yasuda, "The Influences of network Issues on Haptic Collaboration in Shared Virtual Environments", Fifth Phantom Users' Group Workshop, 2000.
[18] R.T. Souyad, D. Gaiti, W. Yu, G. Dodds, and A.Marshall, "Experimental Study of Haptic Interaction in Distributed Virtual Environments", Proceedings of EuroHaptics, Springer-Verlag, Munich, 2004, 260-266.
[19] P. Buttolo, R. Oboe and B. Hannaford, "Architectures for Shared Haptic Virtual Environments", Computers and Graphics, special issue, 1997.
[20] J. Hespanha, M. McLaughlin, G. Sukhatme, M. Akbarian, R. Garg and W. Zhu. "Haptic Collaboration over the Internet", Fifth Phantom Users' Group Workshop, 2000.
[21] J. Marsh, M. Glencross, S. Pettifer and R. Hubbold, "A network architecture supporting consistent rich behaviour in collaborative interactive applications", IEEE Transactions on visualization and computer graphics, vol. 12, no. 3, May/June 2006.
[22] R. Iglesias, A. Carrillo, S. Casado, T. Gutiérrez and J. I. Barbero. "Virtual Assembly Simulation in a Distributed Haptic Virtual Environment", International conference on Advanced Design and Manufacture, Harbin, China, 2006.
[23] SensAble Technologies http://www.sensable.com/
[24] PERCRO Scuola Superiore Sant'Anna, Italy http://www.percro.org/
[25] T. Gutierrez, J.I. Barbero and A. Eguidazu. "Virtual Assembly and Disassembly". In Intelligent Assembly and Disassembly, Eslovenia, 1998, 21-23.
[26] S. Gottschal, M.C. Lin and D. Manocha, "OBBTree: A Hierarchical Structure for Rapid Interference Detection". Univ. of North Caroline, Chapell Hill.
[27] J.I. Barbero, T. Gutiérrez and A. Eguidazu. "Haptic Virtual Prototypes for Assembly Simulation", Proceeding of the 31st ISATA Conference, Düsseldorf (Germany), 1998, 109- 117.
[28] J.I. Barbero, T. Gutiérrez, A. Alvarez and A.R. Carrillo. "Assembly Simulation Tools: Tolerance Analysis and Haptic Virtual Environment", EAEC European Automotive Congress. Barcelona, 1999.
[29] M. Mauve, J. Vogel, V. Hilt, and W. Effelsberg, "Local-Lag andTimewarp: Providing Consistency for Replicated Continuous Applications," IEEE Trans. Multimedia, vol. 6, no. 1, Feb. 2004, 47-57
[30] ZTI Company. "NetDisturb". http://www.zti-telecom.com/pages/main-ip.htm