Simultaneous Object Manipulation in Cooperative Virtual Environments

Abstract

Cooperative manipulation refers to the simultaneous manipulation of a virtual object by multiple users in an immersive virtual environment (VE). We present techniques for cooperative manipulation based on existing single-user techniques. We discuss methods of combining simultaneous user actions, based on the separation of degrees of freedom between two users, and the awareness tools used to provide the necessary knowledge of partner activities during the cooperative interaction process. We also present a framework for supporting the development of collaborative manipulation techniques. Our framework is based on a Collaborative Metaphor concept that defines rules to combine user interaction techniques. Finally, we describe an evaluation of cooperative manipulation. Results indicate that in certain situations, cooperative manipulation is more efficient and usable than single-user manipulation.

1. Introduction

Some object manipulation tasks in immersive virtual environments (VEs) are difficult for a single user to perform with typical 3D interaction techniques. One example is when a user, using a Ray-casting technique, has to place an object far from its current position. Another example is the manipulation of an object through a narrow opening. This problem can be illustrated by the situation where it is necessary to move a couch through a door or a window. In this case, if we place a user on each side of the door, the task can be performed more easily because they can both advise each other and perform cooperative movements they are not able to perform alone. Some problems of this type can be addressed without cooperative manipulation; that is, by simply allowing one user to advise the partner. For this situation existing architectures are sufficient to support the collaboration. If, however, it is necessary or desired that more than one user be able to act at the same time on the same object, new interaction techniques and support tools need to be developed.

Our work is focused on how to support cooperative interaction and how to modify existing interaction techniques to fulfill the needs of cooperative tasks. To support the development of such techniques we have built a framework that allows us to explore various ways to separate degrees of freedom and to provide awareness for two users performing a cooperative manipulation task. We also aim to make the transition between a single-user and a collaborative task seamless and natural, without any sort of explicit command or discontinuity in the interactive process, thus preserving the sense of immersion in the VE.

We base the design of our interaction techniques on the concept of a *Collaborative Metaphor*: a set of rules that define how to combine and extend single-user interaction techniques in order to allow cooperative manipulation. We noticed that the state-of-the-art in single-user object manipulation was in so-called "magic" interaction techniques, based on the concept of mapping the user's motions in some novel way to the degrees of freedom (DOFs) of the object. We also noticed that cooperative manipulation techniques were based on "natural" interaction (simulation of the forces that each user applies to the virtual object). Simply combining these two approaches would create a discontinuity when users transitioned from single-user to cooperative manipulation.

Based on this observation, our work strives to show that magic interaction techniques can also be used efficiently in cooperative manipulation, in the sense that each user could control a certain subset of the DOFs associated with an object.

2. Characterization of Collaborative Manipulation

The original motivation for this work lies in the fact that certain VE manipulation tasks are more difficult when performed by a single user. These difficulties can be related to the interaction technique being used or to the task itself.

2.1.1 Difficulties related to the interaction technique in use

Interaction techniques facilitate or complicate object manipulation in various ways. When using the ray-casting technique (Mine, 1995), for instance, some rotations are difficult because this technique

does not afford rotation around the vertical axis. To perform a task that involves this kind of rotation, a technique like HOMER (Bowman, 1997) would certainly be a better option. Users of HOMER, however, have difficulty with certain types of object translation.

Another possible solution is to allow the user to navigate to a better position to perform the rotation. However, if the environment presents too many obstacles, like walls or other objects, the navigation may be difficult also. In addition, navigation introduces an additional level of complexity to the interactive process because the constant switches between navigation and manipulation create cognitive load and break the natural pace of operation (Mine, 1997; Smith, 1999).

Another example of the limitations of interaction techniques is presented in Figure 1, in which a user U1 needs to move an object O from position A to position B without touching the obstacles. If the available interaction technique is the direct manipulation with the hand, then U1 will have to navigate to move the object. This will create additional difficulties, for U1 will have to release and grab the object many times in order to avoid the obstacles along the way. If HOMER, ray-casting or Go-Go are used, navigation will not be necessary, but the translation parallel to the horizontal axis will not be easy to accomplish.

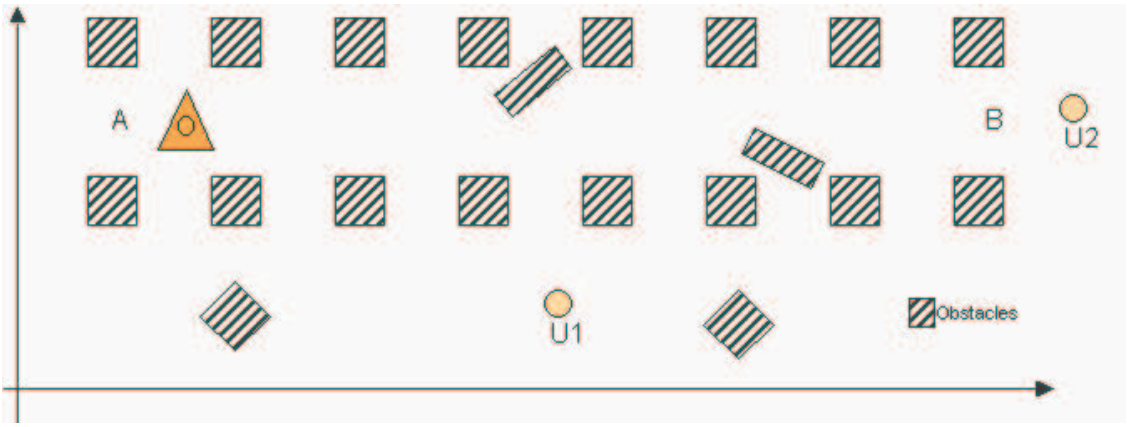


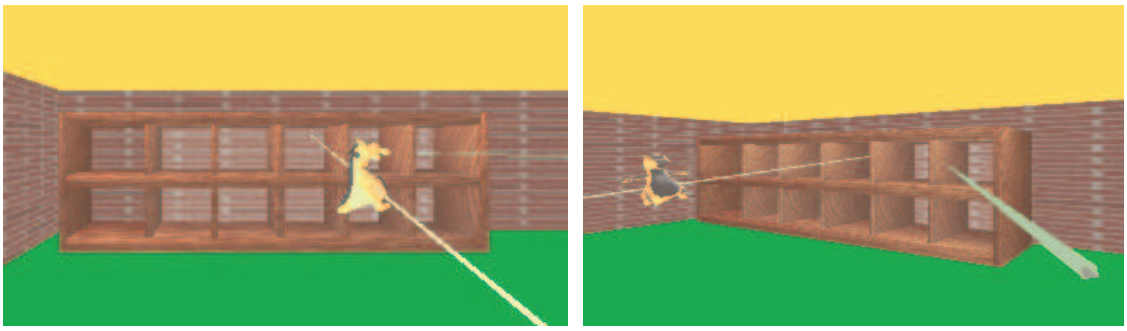
Figure 1 – Object translation with obstacles

In this situation, a second user U2 next to point B may be able to help by “sliding” the object along the ray.

2.1.2 Difficulties related to the task to be executed

Another motivation for cooperative manipulation comes from situations where the position (the object is distant from users or just partially visible) and/or the shape of the object complicate its positioning and orientation.

For instance, if a user has to place an object inside a shelf that is directly in front of him, as in Figure 2a, horizontal and vertical positioning are simple. However, this user cannot easily determine the depth of the object for proper placement. A second user, shown in figure 2b, can more easily perceive the depth and so help the first user to perform the task.



(a) (b)

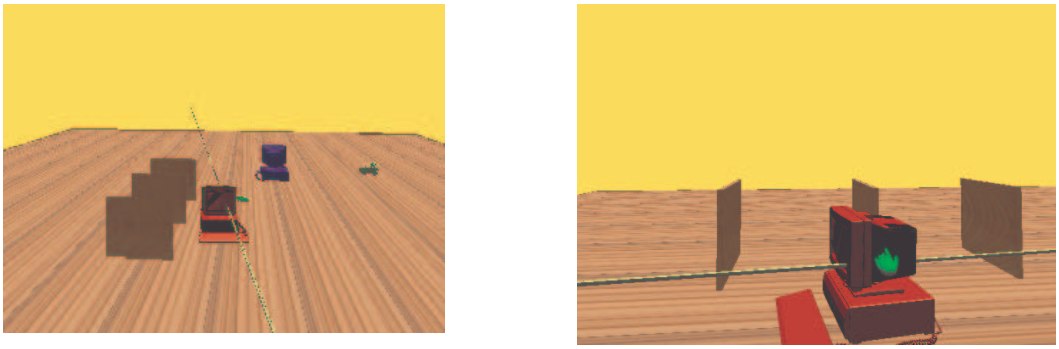
Figure 2 - User without the notion of distance of the object up to its final position

Another example involves the movement of a relatively large object through small spaces, such as moving of a couch through a door (Figure 3). This task can become rather complex (regardless of the interaction technique being used), especially if on the other side of the door there is an obstacle which can not be seen by the user who is manipulating the object. A second user, placed on the other side of the wall, can help do this task. This task is similar to the "piano movers task" studied by Ruddle and his colleagues (Ruddle, 2001).

2D scene 3D scene

Figure 3 – Movement of objects between obstacles

The manipulation of remote (distant) objects, which has been a focus of some prior work (Bowman, 1997; Mulder, 1998), is another example where cooperative manipulation can make the task's execution easier. In Figure 4, for example, if user U1 has to place a computer between the panels he will have difficulties because he cannot clearly see the target location. In this case, a second user (U2) placed in a different position, can help user U1 to find the proper position of the object.



User U1's view User U2's view

Figure 4 - Manipulation of distant objects

2.2 Approaches for supporting cooperative manipulation

In most of today's collaborative virtual environments (CVEs) like NPSNET (Macedonia, 1994), MASSIVE (Greenhalgh, 1995), Bamboo (Watson, 1998), DIVE (Frécon, 1998), RAVEL (Kessler, 1998), AVOCADO (Goebel, 1999) and Urbi et Orbi (Fabre, 2000), the *simultaneous manipulation* of the same object by multiple users is avoided. In these systems, the object receives a single command that is *chosen* from among many simultaneous commands applied to the object. Figure 5 shows a diagram modeling non-simultaneous manipulation. Through an interaction technique a user executes an action that is converted (by the local metaphor) into a command to be sent to the object. A second user performs a different action on the same object. The commands are received by a *selector* that decides which of them must be applied to the object.

True cooperative manipulation has only been the focus of a few research efforts. Most of these systems have used force feedback devices so that each user senses the actions of the other (Basdogan, 2000; Sallnäs, 2002). The manipulation process used in these systems is schematically

demonstrated in Figure 6, where it can be observed that the commands generated by each user are *combined*, producing a new command to be applied to each local copy of the object.

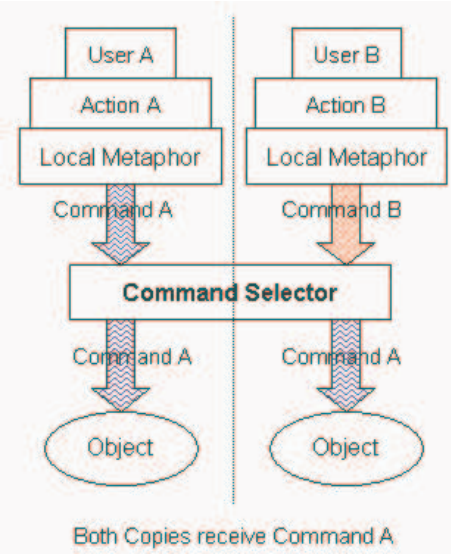


Figure 5 - Command selection architecture

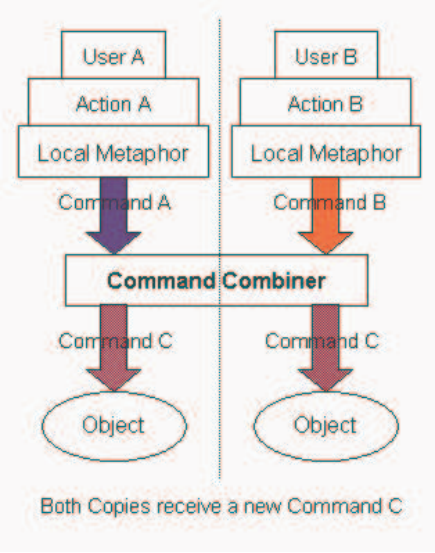


Figure 6 – Command combination architecture

Margery (Margery, 1999) presents an architecture to allow cooperative manipulation without the use of force feedback devices. This system is restricted to a non-immersive environment, and the commands that can be applied to the objects are vectors defining direction, orientation, intensity and the point of application of a force upon the object. This work, then, is based on the simulation of real-world cooperative manipulation.

More recent work by Ruddle (Ruddle, 2002; Ruddle, 2001) presents the concept of *rules of interaction* to support symmetric and asymmetric manipulation. This work is especially concerned with the maneuvering of large objects in cluttered VEs. In symmetric manipulation the object can only be moved if the users manipulate it in exactly the same way, while in asymmetric manipulation the object moves according to some aggregate of all users' manipulation. This work, however, also uses only natural manipulation, and does not consider magic interaction techniques.

3. Collaborative Metaphor - Combining Interaction Techniques

The analysis above has led us to the development of a novel interaction model in which two users act in a simultaneous way upon the same object.

Our approach *combines individual interactive metaphors* instead of simply combining force vectors, creating an extension of the single-user interaction techniques commonly used in immersive VEs.

3.1 Single-user Manipulation Techniques

An interactive metaphor defines a mapping between the user's actions and their effects on the object. Figure 7 shows that using a ray casting metaphor, the rotation of the user's pointer will move the object as if it were attached to the pointer. On the other hand, using HOMER, the same rotation will be mapped to the object's rotation around its own axis.

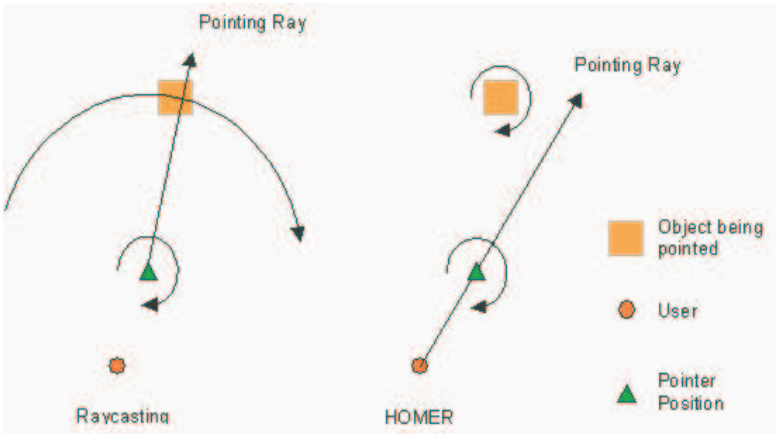


Figure 7 - Different mappings of the same user's actions

In this paper, to model an interaction technique, we use Bowman's methodology (Bowman, 1999), which divides manipulation into four distinct phases: *selection*, *attachment*, *position* and *release*. Table 1 shows the meaning of each phase.

Table 1 – Phases for an interactive manipulation technique

Phase	Description
Selection	Specifies the method used for indicating an object to be manipulated.
Attachment	Specifies what happens in the moment that the object is captured by the user (or linked to its pointer)
Position	Specifies how the user's and the pointer's movements affect the object's movement
Release	Specifies what happens in the moment that the object is released by the user

The use of this decomposition facilitates the combination of interaction techniques because each phase can be treated separately. It is worth mentioning that all the interaction between user and object in the VE is done through a pointer controlled by the user. The shape and function of this pointer depend on the individual interactive metaphor.

3.2 Collaborative Metaphor

Based on the decomposition presented above, we define the concept of **Collaborative Metaphor**. It includes:

a) How to *combine* actions in each phase of the interactive process when users are collaborating (section 3.3), and

b) What kind of *awareness* must be generated in order to help the users understand each phase of the collaborative interaction (section 0).

We also consider the following issues in the design of our cooperative manipulation techniques:

a) **Evolution**: Building cooperative techniques as natural extensions of existing single-user techniques, in order to take advantage of prior user knowledge,

b) **Transition:** Moving between a single-user and a collaborative task in a seamless and natural way without any sort of explicit command or discontinuity in the interactive process, preserving the sense of immersion in the virtual environment, and

c) **Code reuse:** The subdivision of the interaction technique into well-defined phases, allowing the designer to modify only the necessary parts of the single-user techniques to define a new collaborative technique.

3.3 Interactive metaphor phase combination

In this section we examine how to combine each phase of two or more interaction techniques to support simultaneous interaction.

3.3.1 Combination of the selection phase

In the selection phase the collaborative activity begins. From the interaction technique point of view, the way in which an object is selected does not change whether the interaction is individual or collaborative. This is because simultaneous manipulation does not take place until both users confirm the selection of the same object. The way one user selects an object does not depend on whether or not his partner is manipulating this object. This property helps in the learning of the collaborative technique, because if the user already knows how to select an object with his individual interaction technique, he will not need to learn anything else to select the object for cooperative work.

3.3.2 Combination of the attachment phase

At the attachment of an object to a user's pointer, it is first necessary to verify whether the object is being manipulated by another user. If it is not, then single-user manipulation proceeds normally. A message should also be sent to the partner, letting him know that one of the objects has just been attached to another user.

If another user is already manipulating the object, it is necessary to check which DOFs can be controlled by each one, and set up functions to map each user's actions to the object based on these DOFs.

3.3.3 Combination of the positioning phase

The process of positioning an object in a simultaneous manipulation is based on the pointer's movement. If at each rendering cycle the local control system receives information related to the partner's pointer, it can, based on the collaborative metaphor rules, *locally* perform the proper interpretation of this information and apply the resulting commands to the object. This strategy eliminates the need for sending explicit commands related to the simultaneous manipulation situation through the network.

3.3.4 Combination of the release phase

When an object is released, we should determine whether or not there is another user manipulating the object. If there is not, the functions that map from pointer's movements to commands should be disabled and a message sent to the partner. From then on the interactive process goes back to the selection phase.

If a second user is manipulating the same object, he must be notified that his partner has released the object. In our system, upon receiving the notification message *he automatically releases the object.* This way both users return to the selection phase and can restart the interactive process. In the first versions of our system, when a user received a message saying that his partner had released the object, he started to manipulate the object individually. This was not effective because the mapping rules of the movements were unexpectedly and involuntarily modified. From then on the user was able to control all the DOFs that were allowed by his individual interaction metaphor, without any notice whatsoever or possibility for controlling/reversing the situation. This almost always caused an undesired modification in the object placement just obtained with the simultaneous interaction. After some trials, we noticed that the users began to synchronize the

release of the object, trying to avoid undesired modifications in the object's position and orientation. The automatic double release allows a smooth transition from a collaborative to an individual activity.

3.4 Awareness for Simultaneous Manipulation

In this section we present the features related to *awareness* generation in each phase of the collaborative interaction process.

3.4.1 Awareness in the selection phase

While the user chooses the object he wants to manipulate, it is essential that his partner know what is going on. This awareness will serve as a support to the interactive process. The *pointer representation* is used to allow a user to visualize what his partner is pointing to, and also to enable him to indicate an object he wants to manipulate or reference. Using such pointers, dialogues based on *dietic references* (Bolt, 1980), such as the one in Figure 8, can take place in a CVE.

User 1:	- No, this is not the one! Please, get the object that is in front of this one I am pointing at.
User 2:	- Which one? This or this one?

Figure 8 – Dialogue supported by pointing

We can also use the shape or color of the pointer to allow the user to predict the interactive capabilities of his partner.

In our system, when a user points to an object, that object takes on the color of the user's pointer.

During selection it is also necessary to provide awareness of two more states that can occur in collaborative environments. When one user has already attached an object to his pointer and, at the same time, the partner points to the object, we display the object using a third, different color. When both users, simultaneously point to the same object we use a less saturated version of this color.

3.4.2 Awareness in the attachment phase

The attachment phase is a transition between the state in which the object is free and the state in which it is controlled by one or two users. During this transition two events occur, one related to the

object and another related to the user's pointer. The object is highlighted somehow to signal that it is attached to a particular pointer. The pointer shape is also modified according to the interaction technique that is being used.

In our system, if only one user performs the attachment, the object goes back to its original color. In our first implementation, the object kept the pointer's color with slightly greater intensity. Often, however, the users did not realize that the attachment had taken place, and they frequently complained that the original color would help in choosing the position/orientation of the object.

In a collaborative situation, when one user attaches to an object that is already attached to another user, the pointers for both users should be modified so that they represent which DOFs can be manipulated by each of them.

In our system, three different representations were used for three types of DOFs: rotation, translation and sliding along a pointing ray, also called "reeling". To demonstrate that a user can translate an object, the pointer turns into a set of one to three arrows, each of them representing an axis along which the object can be moved. Figure 9 shows some examples of pointers for translation. On the left, we can see the configuration of a pointer that allows a user to move the object only horizontally (plane XZ), and on the right another pointer that tells the user he can only move the object along Y axis. For rotation, the pointer turns into small disks that define the axes around which the user can rotate the object. Figure 10 shows two examples of pointers: the one on the left shows the user can only rotate the object around Z-axis, while the one on the right indicates that all rotations are possible.

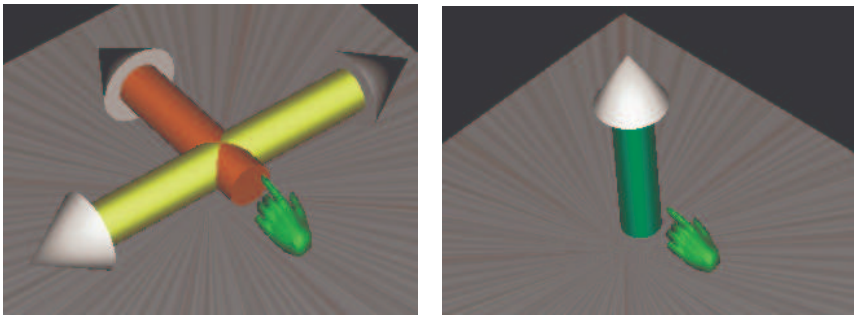


Figure 9 - Examples of translation pointers

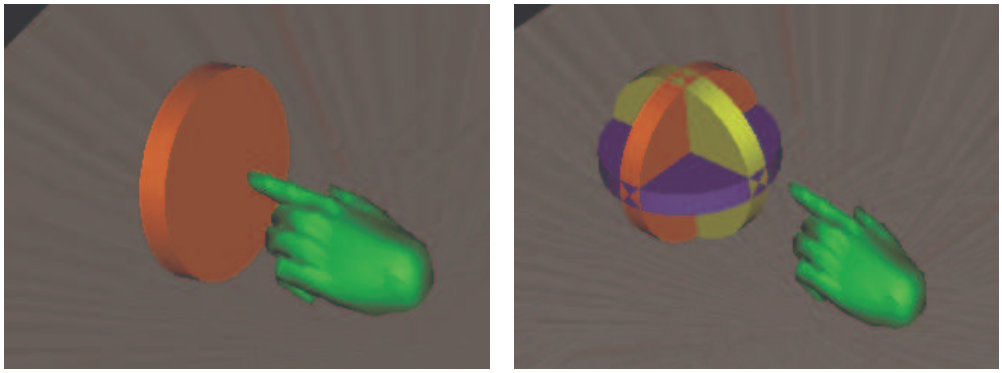
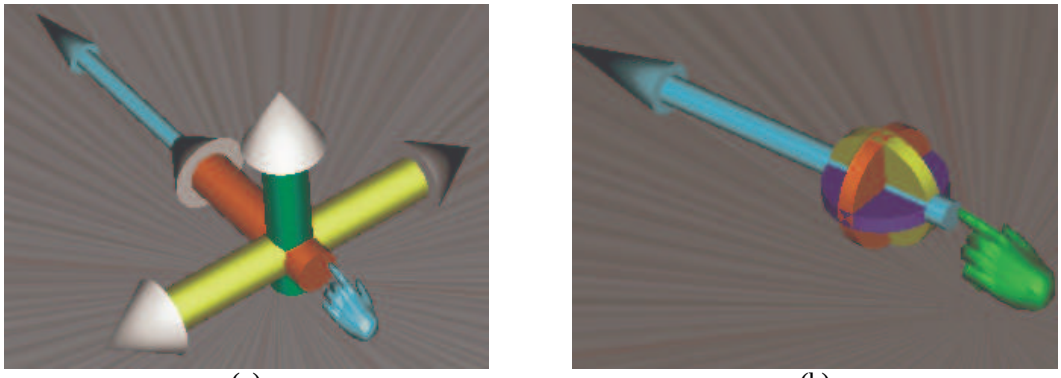


Figure 10 - Examples of rotation pointers

In order to provide to the user the notion he can slide an object along a ray, a longer arrow was introduced in the pointer representation. This arrow can be displayed in the same color as his own pointer or his partner's color. In the first case, the color indicates the user can slide the object along his own pointer and, in the second case, that it is possible for him to slide the object along his partner's pointer. Figure 11 shows this awareness tool combined with translation and rotation pointers.



(a) (b)

Figure 11 - Examples of combined pointers

It is possible to do any combination of the three types of pointers, indicating all the DOFs that a user can control for an object.

3.4.3 Awareness in the positioning phase

During the collaborative positioning phase the object is manipulated according to the rules of the collaborative metaphor, without any special awareness information.

3.4.4 Awareness in the release phase

From the awareness point of view, the releasing phase reconfigures the pointers back to their original state, according to the individual interaction metaphor rules.

3.4.5 Representation of the user's body

The graphical representation of the user's body in a CVE supports body-relative positioning. This feature allows partners to point to elements in a natural way based on common conventions used during collaboration in real environments. We might hear, for example, the sentence: "Take the lamp that is in the box to your left and place it between us, within the reach of my right hand."

An avatar should also represent the user's head position and orientation. This allows other users to understand the gaze direction of the partner.

4. Software Architecture

In order to support the methodology presented in the previous section we have developed a software architecture that provides: (a) the independence of the individual techniques, (b) the exchange of messages among partners, (c) the generation of awareness and (d) the combination of commands. The architecture is presented in Figure 12.

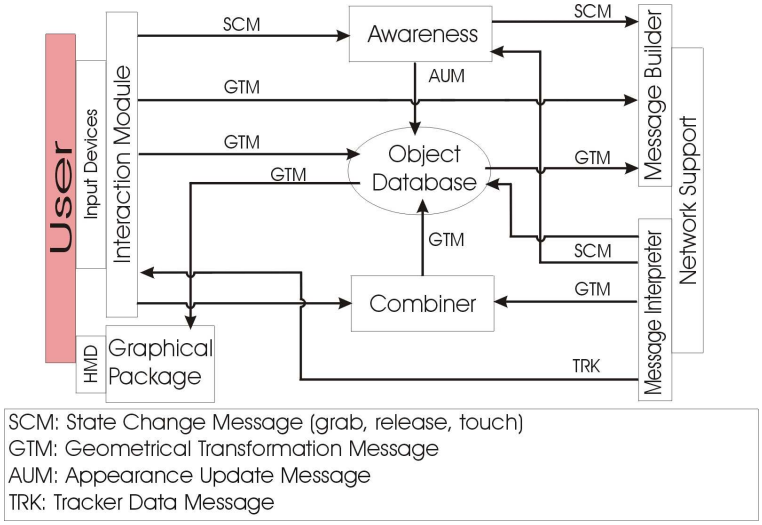


Figure 12 –Architecture for cooperative manipulation based on the Collaborative Metaphor

The *Interaction Module* is responsible for mapping the pointer movements and commands generated by a user into transformations to be applied to the virtual object. This mapping is based

on the individual (or collaborative) interactive metaphor's specification. The *Input Device* module reads the pointer movements.

Implemented as a single module, the *Graphic System* and the *Object Repository* generate the image of the VE that is displayed to the user. In this work, the VE is made up of a set of geometric objects rendered using the *Simple Virtual Environment (SVE)* library (Kessler, 2000). The geometric data that define the VE are replicated on each the machines taking part in the collaboration, in order to reduce network traffic.

The *Command Combiner* is activated when a simultaneous interaction is established and it receives messages about the user's pointer position from the *Interaction Module*, and messages about the position of the partner's pointer from the *Message Interpreter*. Based on the Collaborative Metaphor rules that it implements, this module takes the received messages and, in every rendering cycle, selects which DOFs will be used from each user to update the position of the object that is being cooperatively manipulated. After generating a new transformation, the Combiner sends a message to the Object Repository in order to update the object position.

The *Awareness Generator* is the module responsible for updating the colors of the objects when the pointers are touching them, and it is also responsible for modifying the pointers' shapes whenever a collaborative manipulation situation is established or finished. This module receives messages from the *Interaction Module* that originate from the interpretation of the local user's movements and also from the *Message Interpreter*.

The *Message Interpreter* receives the messages coming from the partner and decides to which local module they should be sent.

Table 2 shows the set of existing messages, their meaning and the module to which the Interpreter sends them.

Table 2 – Messages received by the Messages' Interpreter

Message	Local destination module(s)	Meaning
UPDATE Position	Combiner/Object Database	The object was moved by the partner
TRACKER data	Interaction	Tracker device data
GRAB event	Combiner	An object was attached by the user
RELEASE event	Combiner	An object was released by the user
TOUCH event	Awareness	An object was touched by the user
UNTOUCH event	Awareness	An object is not being touched anymore

The *Message Generator* processes the messages received from the local modules and sends them to the partner. The *Network Support* module is responsible for sending and receiving the messages between the partners. This module is built on the TCP/IP protocol in order to ensure the proper synchronization between the environments and the consistency of the data that travel through the nodes.

5. User Studies

In order to evaluate the use of our techniques for performing cooperative manipulation tasks in CVEs, we developed three VEs that allow two users to perform both simultaneous and non-simultaneous collaborative tasks. Our goal was to find specific situations where cooperative manipulation can lead to easier and more efficient task execution.

Each VE evaluates one combination of two single-user techniques. To choose the interaction techniques, both single-user and collaborative pilot studies were conducted. In these studies, "expert" VE users tried various interaction technique combinations and expressed their opinion about the best choices to perform each task.

For the collaborative techniques, we based the separation of DOFs on the task to be performed, not to prove that those configurations are the best possible choices, but to demonstrate that the use of simultaneous interaction techniques can be more efficient than two users working in parallel using single-user interaction techniques.

In our studies, each user wore a tracked head-mounted display (HMD) and held a tracked pointer that allowed him to interact in the VE. The two machines were connected through their Ethernet interfaces in a peer-to-peer configuration, at 10 Mbits/s.

5.1 Case study with object displacement and orientation

The first VE was designed to simulate a classroom in which two users (in opposite corners of the room) are to place computers on the desks. Figure 13 shows the view of one of the users.

The task was to place four computers on four desks in the middle of the room, in such a way that the computers had their screens facing the opposite side of the white board. This task involves both object movement and orientation.

Figure 13 –Virtual Environment for Task 1

For the individual execution of this task we chose the HOMER technique, because the task required two basic actions that are easy to perform with this technique: selection (and manipulation) of distant objects and rotation of objects around their local coordinate axes. After a pilot study, we decided to allow the user to slide the selected object along its pointing ray so that he could bring it closer or push it away from his hand (the indirect HOMER technique (Bowman, 1997)).

The collaborative technique chosen for the simultaneous manipulation allowed one of the users to control the object's position and the other user to control the object's rotations. We chose this technique because we could clearly see the existence of two steps: one when the object is moved from its initial position to the desk where it will be placed and another when, by means of small

rotations, the object is placed in its final position. The control of the sliding function was disabled in the collaborative technique.

Each pair of users performed the task in a non-simultaneous condition (each user used the individual technique, and the users divided the task between them), and a simultaneous condition (the two users were allowed to use the cooperative manipulation technique). This is a strict test of the power of cooperative manipulation, because indirect HOMER has been shown to be appropriate and efficient for this type of task (Bowman, 1999).

The experiment was conducted using ten pairs of users. Our pilot studies showed no effect of the order of the two conditions. Therefore, in the experiment we always asked the pairs to use the individual technique first, since the collaborative technique assumes that users already know the individual technique. Table 3 shows the time taken by each pair to complete the task in the two conditions.

Table 3 – Comparison of results for the performance of Task 1 with and without simultaneous collaboration

Pair	Without simultaneous manipulation	With simultaneous manipulation	Difference	% Difference
1	06:15	03:20	02:55	47%
2	03:29	03:20	00:09	4%
3	09:37	06:41	02:56	31%
4	09:41	07:30	02:11	23%
5	03:50	01:36	02:14	58%
6	07:10	04:30	02:40	37%
7	06:40	04:05	02:35	39%
8	08:50	06:10	02:40	30%
9	07:30	04:40	02:50	38%
10	04:30	04:10	00:20	7%
Mean	**06:45**	**04:36**	**02:09**	**32%**
STD Deviation	**2:16**	**1:46**		

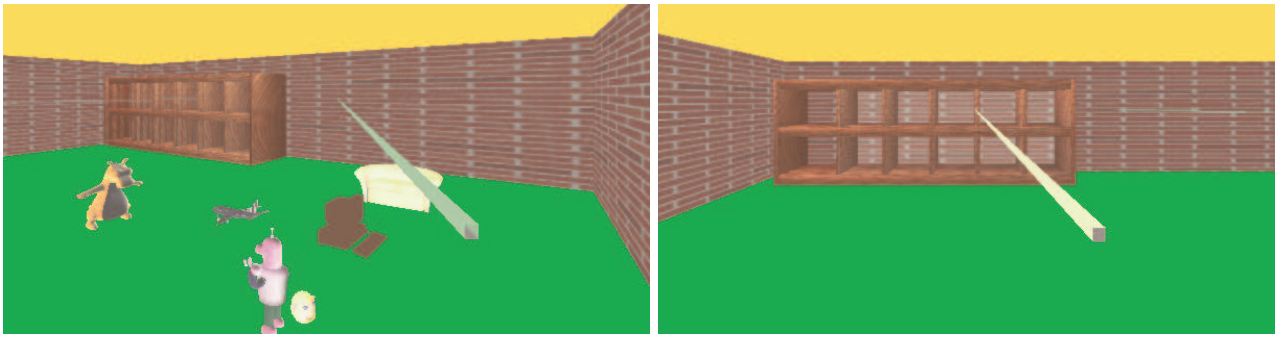
On average, the task time for the simultaneous condition was two minutes and nine seconds less than in the non-simultaneous condition.

A t-test analysis indicated that this difference was highly significant ($p < 0.0001$).

5.2 Case study with large movements and object fitting

The second task designed to evaluate the use of simultaneous manipulation consisted of placing objects inside some divisions of a shelf. Figure 14 shows what both users could see in the VE.

For performing this task using the individual techniques the users used ray-casting with the sliding feature. At first we tested the HOMER technique, but we decided not to use it in this task because it did not represent any significant advantage in the interaction process, and it was in fact more difficult, because it has a more complex control when the user needs to perform large movements on the selected object.



User U1 view User U2 view

Figure 14 – Scenario for the shelf task

At the beginning of the experiment the objects were put next to user U1 and far away from the shelf. This user selected the desired object and put it next to the other user (U2), placing it in front of the shelf. At this point, user U2 was able to select and orient the object as wished and could start moving it towards the shelf.

Because of the distance between user U2 and the shelf, depth perception was a problem when placing the objects. U1 could then give U2 advice to help him slide the object along the ray.

For performing the simultaneous manipulation for this task, we chose to configure the collaborative metaphor in such a way that the user placed in front of the shelf (U2) was able to control the objects' translation, leaving the sliding and rotation of the object for U1, who was close to the objects' initial position. U1 did not control the movement of the object along his own ray, but along U2's ray. We have called this type of control *remote sliding*. This way the user in front of the

shelves needed only to point into the cell where he wanted to place the object, while the other user controlled the insertion of the object into the shelf.

The experiment was again conducted using ten pairs of users. Table 4 shows the resulting data from the individual and simultaneous manipulation, considering the time for finishing this task

Table 4 – Experiment 2 results comparison

Pair	Without simultaneous manipulation	With simultaneous manipulation	Difference	% Difference
11	04:00	02:45	01:15	31%
12	05:21	02:30	02:51	53%
13	08:02	04:55	03:07	39%
14	13:00	03:27	09:33	73%
15	05:15	02:38	02:37	50%
16	08:30	05:00	03:30	41%
17	09:00	04:05	04:55	55%
18	08:20	06:30	01:50	22%
19	04:30	02:50	01:40	37%
20	06:00	04:40	01:20	22%
Mean	**07:12**	**03:56**	**03:16**	**42%**
STD Deviation	**02:43**	**01:20**		

A t-test indicated that the difference in conditions was again highly significant ($p<0.001$).

5.3 Case Study with movement in cluttered environment

The third experiment asked users to move a couch through an aisle full of columns and other obstacles. A user (U1) was at one end of the aisle (Figure 15) and the other one (U2) was outside, on the side of the aisle. For doing this task using an individual interaction technique, based on our pilot study, we again chose HOMER.

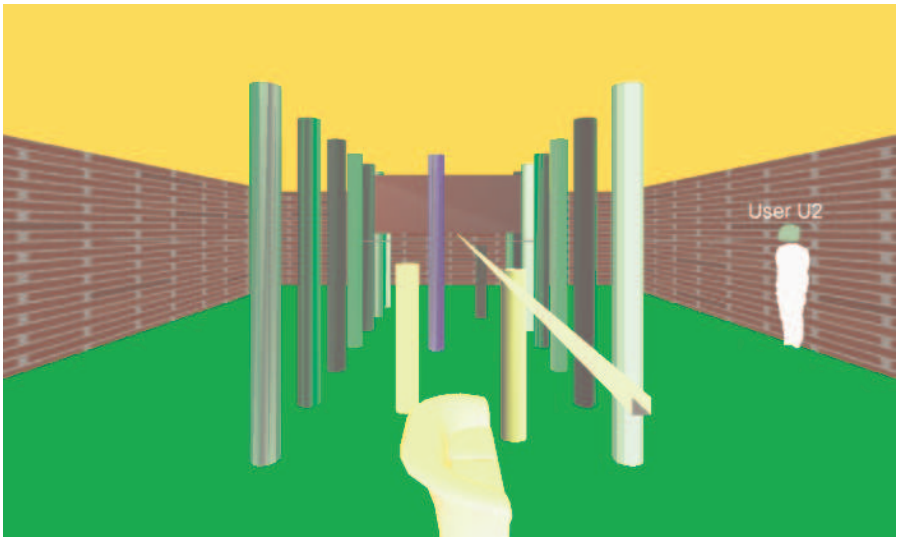


Figure 15 – Scenario for experiment 3 (User U1 view)

For doing the task using the individual technique the users instinctively used a strategy in which U1 started the object's manipulation and, sliding it along its ray, placed it inside the aisle. Upon finding an obstacle, this user released the object, which was then selected by the partner (U2) who avoided the obstacle and released the object, giving control back to U1.

The collaborative technique chosen for this task was configured to allow user U1 to control the object translations and user U2 the rotations and the remote slide.

The experiment was conducted using ten pairs of users. Table 5 shows the resulting data from the individual and simultaneous manipulation, considering the time for finishing this task.

Table 5 –Experiment 3 results comparison

Pair	Without simultaneous manipulation	With simultaneous manipulation	Difference	% Difference
21	02:05	00:50	01:15	60%
22	03:00	01:40	01:20	44%
23	03:20	02:40	00:40	20%
24	02:40	01:05	01:35	59%
25	02:40	01:40	01:00	38%
26	04:20	04:00	00:20	8%
27	02:45	01:55	00:50	30%
28	03:20	02:00	01:20	40%
29	03:10	02:30	00:40	21%
30	03:00	02:10	00:50	28%
Mean	**03:02**	**02:03**	**00:59**	**35%**
STD Deviation	**00:35**	**00:53**		

A t-test indicated that the 59-second difference between the means was highly significant ($p<0.00001$).

6. Discussion

The experiments have allowed us to evaluate both the architecture and different methods of combining individual techniques. It has also allowed us to evaluate the basic premise for the work, that certain tasks are more easily performed using simultaneous manipulation during collaboration when compared to methods of non-simultaneous manipulation (in which the users work in parallel).

Concerning the design of collaborative techniques, the experiments allowed us to verify different alternatives for separating the DOFs that each user can control.

Several configurations of position DOFs were tested. The most common was the one in which a user could move the object in the horizontal plane while his partner controlled just the object's height. This technique was useful in cases where the users had to move objects among obstacles that were not all the same height. In such cases, while one user moved the object forward, backward and sideways, the other one simply lifted or lowered the object to avoid the obstacles. A similar configuration was important for controlling the movement of distant objects, particularly when one of the users could not clearly see the final position of the object. In the second experiment, for instance, the user in front of the shelf could not clearly see how much the object had to be moved forward or backward, in order to be correctly fit in one of the available spaces. The partner's simultaneous manipulation allowed the correction in the final positioning phase.

The techniques that allowed one user to slide the object along his partner's ray also provided a greater control over small adjustments in the final position of the object. One of the users could point to where the object should be moved while the other controlled the sliding itself. This technique was particularly useful in those cases where the user who controlled the direction of the ray had a good view of the trajectory to be followed by the object, but was too distant from its final position. This technique is also applicable to other interaction techniques that use a ray as a pointer.

Finally, the experiments allow us to assert quite confidently that for many tasks in which the insertion of a collaborator (with non-simultaneous manipulation) benefits the task execution, the use of simultaneous manipulation provides an even greater benefit.

7. Conclusion and Future Work

Our architecture and techniques based on the Collaborative Metaphor make several novel contributions to the field of collaborative VEs, including:

- Allowing the use of magic interaction techniques in simultaneous manipulation processes,
- Allowing simultaneous manipulation in immersive environments, and
- Simultaneous manipulation without the need for force feedback devices.

We were concerned in the beginning of this study with the *context change* between the individual and collaborative activities (and how this would affect the users), a recurrent problem in collaborative environments. Separating the interactive metaphors into distinct phases made it possible to control the effect of the users' actions and to prevent the interference of the activity of one user into the other, regardless of the phase of the interaction.

Our architecture allows an easy configuration of the Collaborative Metaphor, if it is based on techniques such as ray-casting, HOMER or Simple Virtual Hand. In these cases, the configuration of the Collaborative Metaphor is done simply by changing a configuration file that defines the interaction technique and the DOFs controlled by each user during the cooperation. To include an individual technique that is totally different from the ones already implemented, we simply need to change the *Interaction Module* that interprets the movements of each user.

Our collaborative techniques were implemented with minimum changes in the individual techniques. An important point to make is that the designation of the DOFs that each user will control is done *a priori*, in the configuration of the collaborative technique itself. The possible dynamic and immersive configuration of who controls each DOF is left for future work. The main

problem in this case is how to build an immersive interface to perform such a configuration procedure.

Finally, since the architecture does not limit the number of users that can participate in a collaborative interaction session, we plan in the future to evaluate the usability of these techniques with more than two users.

References

Basdogan, C., Ho, C., Srinivasan, M. A., and Slater, M. (2000). An Experimental Study on the Role of Touch in Shared Virtual Environments. *ACM Transactions on Computer-Human Interaction.* New York v.7, n.4, p. 443-460.

Bolt, R. Put-that-there: Voice and gesture at the graphic interface. (1980). *Computer Graphics*, v. 14, n. 3, p. 262-270.

Bowman, D. and L.F. Hodges. (1997). An Evaluation of Techniques for Grabbing and Manipulating Remote Objects in Immersive Virtual Environments. *Proceedings of Symposium on Interactive 3d Graphics*, p. 35-38.

Bowman, D., & Hodges, L. (1999). Formalizing the design, evaluation, and application of interaction techniques for immersive virtual environments. *The Journal of Visual Languages and Computing*, v. 10, n.1, p. 37–53.

Fabre Y., Pitel G. and Verna D. (2000). Urbi et Orbi: Unusual Design and Implementation Choices for Distributed Virtual Environments. *Proceedings of International Conference on Virtual Systems and Multimedia* (VSMM'2000), 714-724.

Frécon, E. Stenius, M. (1998). DIVE: A scalable network architecture for distributed virtual environments. *Distributed Systems Engineering Journal*, v. 5, n. 3, p. 91–100.

Goebel, M. (1999). Digital Storytelling – Creating Interactive Illusions with Avocado. *Proceedings of International Conference on Artificial Reality and Telexistence (ICAT'99)*, p. 9-22.

Greenhalgh, C. Benford, S. (1995). Massive: A Virtual Reality System for Tele-conferencing. *ACM Transactions on Computer Human Interfaces*, v. 2, n. 3, p. 239-261.

Macedonia, M., Zyda, M., Pratt, D. and Barham, P., (1995). Exploiting reality with multicast groups: A network architecture for large-scale virtual environments. *Proceedings of Ieee Virtual Reality Annual International Symposium (VRAIS'95)*, p.15-19.

Mine, M.; Brooks F.; Sequin, C. (1997). Moving Objects in Space: Exploiting Proprioception in Virtual-Environment Interaction. *Proceedings of the 1996 ACM Conference on Graphics (SIGGRAPH'97)*. New York:ACM p. 19-26.

Mulder, J. (1998). Remote Object Translation Methods for Immersive Virtual Environments. *Proceeding of Virtual Environments Conference & 4th Eurographics Workshop (EGVE'98)*, p. 80-89.

Ruddle, R. A., Savage, J. C.; Jones, D. M. (2001) Movement in Cluttered Virtual Environments. Presence, Vol. 10, No. 5, October 2001, 511–524.

Ruddle, R. A., Savage, J. C.; Jones, D. M. (2002). Symmetric and asymmetric action integration during cooperative object manipulation in virtual environments. *ACM Transactions on Computer-Human Interaction, 9*, p. 285-308, 2002.

Sallnäs, E-L. (2002). Collaboration in Multimodal Virtual Worlds: Comparing Touch Text. Available at http://www.nada.kth.se/iplab/knowhow/KnowHow_Lota.pdf.

Smith, S. and Duke, D. (1999). Virtual environments as hybrid systems. *Proceedings of the 17th Eurographics Annual Conference* (*Eurographics'99)*, p. 169-178.

Watson, K.; Zyda, M. (1998). Bamboo - a portable system for dynamically extensible, real time, networked, virtual environments. Proceedings of *IEEE Virtual Reality Annual International Symposium (VRAIS'98)*, p. 252-260.