Your Paper

You

January 14, 2015

Abstract

Your abstract.

1 3-Hand manipulation

1.1 Introduction

This paper presents a tool that allows to 2 or 3 users to manipulate virtual the same virtual object simultaneously. The 6 freedom degrees can be reproduced.

1.2 State of the art

Multi-User interaction tools already exist, but they are limited and not representative of the real world interactions. Some aren't symmetric: the DOF are distributed over the users, like "trackball". Others don't allow one of the DOF (the rotation around the axis defined by two handling hands) or at least don't reproduce it in a realistic way, like "SkeweR". By the way, the good ways to combine two interactions change from an application to another (ex : adding, compute the mean).

The determination of the sixth DOF is impossible because of the number of contact points, which is only 2. Therefore, it's impossible to generate forces able to make the object rotate around the defined axis.

1.3 Solution

The simple solution is to define a third point of handling. Now we have a plane and therefore a possible change of orientation. This rotation is computed as the combination of two (sufficient) rotations.

1.4 Critics

The adding of a third point for a third hand in order to provide 6DOF is not realistic because in the real life, it is possible to use the 6DOF with only to hands or even only one. The reason is that there are several contact points between a hand and an object, e.g the fingers. At least a more realistic interaction paradigm would have defined four contact points (for two times two hands), and even more. In that case, the limit is the non-deformability of the objects.

2 Exchange of avatars

2.1 Introduction

This paper presents and compares several ways to represent an exchange of avatars, for the actors populating the virtual environment to be aware of the exchange. Two cases are considered: if the user to be notified is involved in the exchange or not. One of the conclusions of the experiments is that it's not necessary to perform a visual feedback of the exchange even if it's more efficient.

2.2 State of the art

Exchange of avatar already has a lot of applications. In video games, a player can enhance his game experience by changing of character. In collaborative training, switching roles allows a trainee to understand the work as a whole and reinforces the team cohesion. Other researches studied the virtual body takeover sensation, which must be taken into account when you change your virtual body. However, the perception of the exchange in itself has not be studied before this paper.

2.3 Solution

2.3.1 Formalisation

As it was already explained, two points of view are considered: the point of view of a user witnessing the exchange and the one of a user experiencing it. For both cases, the interest of being aware of the exchange is obvious, since all users are likely to interact with each others and as a consequence being able to know which user is using which avatar.

For each exchange, the user experience was evaluated through two criteria:

Understanding: how difficult it is for the user to be aware of the exchange, in terms of time, feelings and errors (how many time a user refers to an avatar A believing it is owned by the user 1 whereas it is owned by the user 2).

Perception: how pleasant the exchange is for the user.

2.3.2 Evaluation

Witnessing exchange test

General description: a user is placed in a virtual environment in which he has to perform some tasks. Sometimes he must call a specific expert among several. The experts regularly switch their avatars. Therefore, the user can make mistakes and call an expert instead of another. The goal of the experience is to find the best metaphor to represent a switching of avatars. The owners have to be identifiable during the switching, whatever the metaphor. In this experience, they are identified by a color.

Tested parameters: 3 metaphors representing the switching of two avatar owners ; *Flickering*, *Ghost* and *Popup notification*. *Flickering* consists of making each avatar texture blink, changing temporarily its color to the

one representing the new owner. *Ghost* consists of making appear a ghost coloured according to its owner next to each avatar before translating to the new avatar. *Popup notification* consists of displaying a notification pop-up associating each avatars to its new color.

Data: 54 users, heterogeneous regarding VR knowledge. This group was divided into three parts. Each part was assigned to one of the three metaphors.

Evaluation criteria: ranking of the three exchange metaphors over 10 criteria that we can group into two types: subjective pleasantness and objective performance (completion time of a task and error rate when identifying who is who).

Results: the ranking analysis led to the conclusion that in term of pleasantness and understanding, *Ghost* is the best metaphor, a bit better than *Flickering* and far better than *Popup notification*. On the contrary, *Popup notification* led to the best mean of completion time. The three metaphors led to equivalent error rates and general ease of use.

Triggering exchange test

General description: a user is placed in a virtual environment, controlling an avatar with a first person point of view. He changes his avatar with another one. The same system of color than for the first test is used.

Tested parameters: 5 metaphors are used. Each combine a camera motion (*straight*, *curved* or *none*) and a visual effect (*ghost* or *none*): (*straight*, *ghost*), (*straight*, *none*), (*curved*, *ghost*), (*curved*, *none*) and (*none*, *none*). The motion *straight* means that the camera directly moves from the point of view of the original avatar to the new one. With the *curved* motion, the camera starts and ends at the same points but takes a third person point of view while moving. The *none* motion means that the point of view of the user changes instantaneously.

Data: 42 users, heterogeneous regarding VR knowledge. They all saw and compared the 5 metaphors.

Evaluation criteria: ranking of the metaphors with two by two comparisons, and with marks given for 5 criteria related to the understanding degree and the pleasantness.

Results: (*none*, *none*) is the best choice because it is easy to understand and simple. The *straight* motions are preferred to the *curved* ones, because they represent well the travel from an avatar to another. The *ghost* visual effect is also appreciated for the same reason. The worst metaphor is (*curved*, *none*) because of the lack of information it carries and the disturbing camera motion.

2.4 Critics and proposals

After having seen that the simplest metaphors are the more efficient but not necessarily the preferred ones, an idea of enhancement would be to combine in one metaphor the simplicity and the pleasantness. A solution is for instance

a popup notification which show the exchange with a schematic view and a minimalist video effect, in order to well represent the fact that there is an exchange. In other terms, the idea is to adapt *Ghost* in a schematic view, displayed in a popup.

This proposal is only adapted to the exchange witnessing, because the third person point of view is the least rated type of metaphor for the exchange triggering. For this problem, it is obvious that the most representative method are the camera motions *straight* and *none*, so it is difficult to do better.

Nevertheless, the *curved* motion should have been adapted for the *none* visual effect. Indeed, the video at `http://youtu.be/Yu4YpGzKXes` shows that the motion stops for an instant in the middle of the trajectory and then resumes. Combined with *ghost*, it allows the user to actually see the translating ghosts. However, the stop was kept for the *none* visual effect. The result is a disturbing pause in the motion, hardly understandable. One can assume that the rate would be better without this effect.

A last critic can be made on the test questions. There are only slight differences between some of them, like the criteria "the metaphor is easy to understand" and "the metaphor is easy to use". Others are difficult to interpret, precisely because the answering person can give them the meaning he wants. For instance, the criterion "this metaphor is a good idea" is too general and is most likely related to the other answers.

3 Show-Through Techniques

3.1 Introduction

This paper presents solutions to the problem of showing hidden objects in a virtual collaborative environment. In the real world, one can walk and reach a point of view from which he is able to see the object. Nevertheless, in some situations, it is necessary to stand close to another person and so violate social protocols (each person have a zone around him where he won't accept the presence of the others). The proposed solutions consists of making obstacles disappear and so show through them.

3.2 State of the art

Several tools were designed in order to improve the "vision power" of a virtual environment actor, like HMDs. However, they don't allow to see well the other actors so these tools are not interesting for the study, which focus on (good) collaborative environments. As the object localization problem require to be in a 3D context with several view points in our case, simple projection screen (with one point of view) are not good tools. Therefore, the chosen solution was a projection tool with glasses displaying several points of view. A virtual ray of light was used in order to point objects.

3.3 Solution

3.3.1 Show-Through techniques

Various show-through techniques don't respect the integrity of the user experience, because the scene structure has to be modified, or because the perception of the environment is disturbed. The two that were selected for the experiments were *Transparency* and *Cutaway*, because they only affect the alpha of the object (i.e the transparency). The first one only make the pointed object become transparent around the pointer (the ray of light) and the second one make the zone become invisible. A solution can be to use a mush larger ray coming from above, evnetually with a spatialized sound pitch.

3.3.2 Experiments

General description: a presenter point a sequence of virtual objects (hidden from the observer). The observer has to localize them in the same order, to ensure that he understood where they were. The practical case used in the experience was the localization of pieces in the engine compartment of a car.

Tested parameters: the techniques that allow to see a hidden object: *Transparency*, *Cutaway* and *None*. In the last case, the user has to move around the scene to see the pointed object. The authors want to compare these techniques.

Data: 24 users, heterogeneous regarding VR knowledge.

Evaluation criteria: the methods are ranked over two types of criteria. The first ones are related to the duration of the retrieval task and to the covered distance. The other ones are evaluated subjective feelings of the users, regarding the understanding of the environment and the comfort, taking into account the "private zone" (or proxemics) respect.

Results: global efficiency was mainly due to the training effect (taken into account in the experiment design). There is very few influence due to the technique because the users learn how to optimize their performances in all cases. Obviously, walking a bit more is required for the *None* technique, but it doesn't affect the efficiency. It is a thing important to note, because it proves that all the techniques are good at giving a good perception and assimilation of the 3D environment. According to the users, the show-through techniques are better for comfort and collaborative tasks and the annoyance due to close positions are confirmed.

3.4 Critics and proposals

All the tchniques tested here are efficient because the "trainer" is visible. It is obvious for *None*. For *Transparency* and *Cutaway*, the trainer is also a visual indication because the trainee know that he faces the transformed zone. If the trainer is invisible (e.g he is not physically present and is not represented by an avatar), there is need for a metaphor indicating what zone was transformed.

4 #SEVEN

4.1 Introduction

This paper presents a sensor effector model allowing to execute complex scenarios in collaborative virtual environments. These scenarios can be made of parallel action executions or even branched execution, *i.e* performing an action can lead to trigger the execution of several actions in parallel and the triggering of an action can require the ending of several actions. The tasks can be parametrizable to be either very precisely defined or very opened. The environment behaviour must also be described. As this type of application is made for collaborative training, each user must have a defined role and there is to be a team organisation. There is also an expert who defines the tasks and their organisation.

4.2 State of the art

Scenario model systems exist. They often use graph-like representations, like Petri nets. Several aspects defined in the previous section are covered by these tools, but separately (some tools focus on how to model the "task performing", others on defining complex scenarios). Two important aspects, the "training driving" and the collaborative aspect are poorly handled in general, regarding the needs described above.

4.3 Solution

Petri nets are used in order to model the scenario proceedings. The events represent transitions in the scenario and places can be in fact sub-scenarios (and therefore Petri nets themselves). Such places transfer received tokens to their initial places and when these tokens reach the ending places, they are "merged" and transferred to the event following the "meta" place.

Each event of the Petri nets system is connected to a sensor which listen to the VE (Virtual Environment) and fires the event when each upstream place hold a token and a specific condition related to the VE is verified. In the same way, each event is connected to an effector which triggers the event resulting effect on the VE.

Each token holds a set of actor identifiers. This set can be changed through a related event execution. Therefore, actors assigned to an interaction (*i.e*, an event) can be known and are definable in the scenario.

Each actor is associated to a set of assignments which define the actions he can perform. Effectors can remove (resp. add) some assignations from (resp. to) an actor.

4.4 Critics and proposals

Three cases should be considered:

1. There is no scenario at all: the users can do what they want to. Nothing is ordered, there is not a special combination of events which lead to a wanted result. An example is a VE that the users can explore and where they can manipulate some objects, only for the pleasure to explore it.

2. There is an underlying scenario but nothing in order to manage its structure. The user has to perform a sequence of tasks in order to achieve a goal but the system and possibilities don't evolve with the achievement steps.

3. The scenario is well defined and controlled with an adequated structure that allows much more complexity. The tool presented there is an example.

One of the interests of the second type of VR applications is to evaluate the user with respect to his respect of the underlying scenario. An example is an application in which the user is ordered by a trainer to perform a sequence of tasks, letting him the possibility to perform them in a wrong order or to give up to the more difficult ones, and evaluating his closeness to the good sequence.

#SEVEN is adapted to all these types of applications. The case of super-users and simple-users can be modeled thanks to projections on task asignements (a super-user acts on a Petri Net allowing him to do everything at any time).

However, the definition of role is not adapted to the case of a trainee requesting his trainer to perform a demonstration, i.e to perform a task only in order to show how to do without making the system evolve. Indeed, the trainee and the trainer have the same action possibilities at the same time, but the results of their action won't be the same. It is possible to avoid the problem by defining an equivalent to each trainee's action which is attributed to the trainer and said *"demonstrative"*. The trainer and the trainee don't share any asignement and can evolve in their own Petri Net.

5 Communication and Awareness

5.1 Introduction

This paper presents a set of tools which goal is to have a better awareness of what is happening in a CVE, in order to collaborate better and communicate with other users in an efficient way. Indeed, the typical action loop is decomposed into event perception, decision making and action performing. Most of the work done so far focus on this loop. However, awareness and communication (in order to get more awareness) weren't very studied, probably because they are considered as a tool that helps each step of the loop. Therefore, this paper presents general solutions to these problems.

5.2 State of the art

5.2.1 Awareness

Awareness is the support of information in CVEs. People can communicate their awareness to the others, so information can be shared. Awareness is therefore an entity distributed among the group of users. It can be "produced" by the VE and the users with the usage of metaphors, audio and video signals. However, the brought information has to remain simple for the users not to be lost.

There are several types of awareness.

Awareness of the others The most important type of awareness in CVEs is the awareness of the other actors. One has not only to be aware of the spatial localization of the other users, but also of their behaviour and current capacity to interact with the VE. For example, it is desirable to know if a collaborator is physically able to reach a required resource before asking him to bring it. It implies to know what others can see or what they are doing in real time. Several means were studied in order to access this information, like the embodiment of another avatar as a witness or small figures with symbolised actions.

Awareness of the environment The objects of the environment can bring interesting information about their history and state (ex: pointed by a user, in motion, involved in an action). This information helps to interact with the VE accordingly to its state. It's also a mean to share information between consecutive users of a same object.

Awareness of coordinating actions If two users handle the same object or have to collaborate in order to perform a task, it is important to be aware of what the other is doing, so that inconsistent coordination of actions can be avoided. The used metaphors play on color codes, informing about the state of a handled object or a specific performed action, like rotate an object. In this last case, the action can be represented as a link between the user and the object, which color changes in according with what the users intends to do (push, pull, etc...).

Awareness of the physical world A simple problem related to VR is that users evolve in the real world with the illusion of being in a possibly unbounded virtual world. As a consequence, collisions with walls or even with others users can happen. The main idea in order to solve this problem is to make appear dissuading virtual obstacles or warnings when a user is close to have an "accident".

Awareness of the network delays The most often faced technical problem in collaborative applications is the inconsistency due to network delays (and so lose of the coordination). It is impossible to avoid this type of problem, so the least the engine can do is to freeze inconsistent objects of the CVE, while the inconsistency is not solved. Some information about the inconsistent state of the object can be displayed in form of echo objects that represents the incompatible states. Whatever the metaphor, information overload must be avoided since it doesn't help to understand the problem at all.

5.2.2 Communication

There are 4 types of communication means: audio, visual, non verbal, and text. It is recommended to use several of them together in order to be complete. Nevertheless, communication means should be chosen in accordance to the application needs and the devices restrictions.

Audio communication is a typical example in which it is better to use a complementary visual communication means. Indeed, it avoids disturbing situations

in which users don't know how to interpret a silence (pause, system failure, delay ?). If correctly associated with a visual feedback, this means has a lot of advantages: it is simple, it can improve the immersive effect if the spatialization is well done and can be used as a command.

Non verbal communication refers to gestures and facial expressions. Its interest is obvious, but the complexity of representation too. In fact, allowing to express through realistic non verbal communication is intuitive for the users and the perception-decision-action process is very quick. Simpler ways to express feelings (like smileys), require less complexity but much more time to actually perform (there is to open a menu, chose the most appropriate smiley and "click" it). Depending of the application, it is necessary to compromise between the two approaches.

Visual communication is in fact a kind of signalisation system. Depending on the needs of the application, some metaphors can be used in order to give specific information, like the field of view of a user, or the state of a key object. The most important thing is to ensure that all users understand the signalisation system.

Text annotations are easily storable. However, they require a support, like a tablet or a virtual object. Sometimes, the written information has to be editable, what adds complexity for the support objects. Speech recognition systems have also their drawbacks, like the obvious need to be efficient. There is also the need to lock support objects for the exclusive usage of its owner (like a status menu in a video game for example), while allowing the other users to understand what is happening. In other words, text based communication bring a lot of constraints that must be taken into account.

6 A general framework for collaborative manipulation in Virtual Environments

6.1 Introduction

The paper presents a framework which goal is to enable collaborative manipulation in VE. Collaboration levels are to be defined.

6.2 Levels of collaboration

Level 1 Representation of the users (e.g avatars).

Level 2 The users can interact with the objects of the scene individually. This level can be divided into two sub-levels:

1. The interactions are constrained: users can do some actions but with some limitations, e.g users can move some objects but not all and can't create or destruct these objects.

2. The interactions are not constrained: users have a kind of unlimited power with the scene, e.g they can desctruct and create objects and walls, as well as make them translate.

Level 3 Two or more users can interact with the same object at the same time. One more time, this level can be subdivided into to sub-levels:

1. The cooperating users can perform idependant actions on the same object, e.g one can change the color of an object while the other one changes its location.

2. The cooperating users can perform codependant actions on the same object, e.g the two users changes the color at the same time the color of an object. In that case, the system preserves the integrity of the object by using a synchronisation mechanism.

6.3 Cooperation metaphors

There are at least three types of metaphors making collaboration easier:

1. Activity metaphor: informations about the users showing what activity they are performing (e.g walking, grabbing). The activity metaphors have to be visible for all the participants.

2. Informative metaphor: when the system of one user has failures or if an action is impossible (e.g go beyond the limits of the map) informations have to be provided to the concerned user.

3. Action metaphor: when an action in collaboration between two users is performed, each of them have to be aware of the parameters related to the other's implication in the task, e.g the direction and intensity of the strength he is using for the collaborative moving of a heavy object.

6.4 Combining inputs

There are two possibilities in order to preserve an object consistency on all the computers:

1. Replicate the object: a master object is chosen and then when a change occurs copies are distributed on the network.

2. Replicate the calculation: the physical models are all designed in such a way that all computers compute the same transformations with respect to a given combined interaction.

6.5 Coordination of the system

Two constraints must be checked in order to guarantee the system viability:

Complete order on sent data All the replications of an object must receive manipulation data in the same order. Indeed, the same code will produce the same results if the data is input has the same order. It's up to the model of the object to guarantee this ordering constraint.

Global agreement on simultaneity Given two actions performed by two different users, all the object replications must "agree" on whether they are simultaneous or not. The solution is that users transparently subscribe or unsubscribe to the object to manipulate (note that this information must be synchronized on all the copies of the object). Then, the object itself "decides" on wheter two or more actions are simultaneous and send this number to all its copies, that wait for all the actions to be transmitted. Some actions don't cause any change (e.g holding an object without moving), so the system send a special value in order to inform that a "passive action occured".

6.6 Critics

1. Using more than two or three objects seems to be unfeasible, because of the high colision detection time.

2. The use of undeterministic functions in order to compute the object transformations invalidates the assumption that sending data in the same order produces the same effects on all the copies. It is important when an action too complex to model by laws of physics occurs, e.g the throwing of a coin. In that case, it is better to update the object copies.

3. The paper doesn't address the concept of a super-user with powers going beyond laws of pysics (level 2.2). The problem is not so trivial if some users are "simple mortals" while others are "super-men". **Action metaphors** are to be defined for super-actions which are suddens and that anihilate all the simple-users efforts, like a teleportation of the object.

4. Some users might be intentionally invisible. In this case, the metaphors have to be changed.

5. Users could control results ore another user performance. In this case, models have to be made for the users too (and not only the objects).

6. A user could control another user, at least partially. In that case the users too have to subscribe or unsubscribe the super-users trying to control them.

More generally, The asymetric collaboration implies to re-think the metaphors.