Virtual Reality to Improve Remote Control in Presence of Delays

Philippe Le Parc, Etienne Pardo, Amara Touil, Jean Vareille

Virtual Reality to Improve Remote Control in Presence of Delays

Philippe Le Parc, , E. Pardo, A. Touil, J. Vareille
Université Européenne de Bretagne, France
Université de Brest
EA3883 LISyC - Laboratoire d'Informatique des Systèmes Complexes
20 av. Victor Le Gorgeu, BP 809 - 29285 Brest Cedex - France
mail contact : philippe.le-parc@univ-brest.fr

***Abstract*—This paper presents the use of virtual reality in the context of remote control. Virtual reality may be used in a classical manner in order to simulate the behavior of a system, but also in parallel with the real system in order to improve the quality of the control, making it possible to deal with time delays induced by the network. Our proposal may be applied to various fields such as robotics, measurements systems, industrial applications or domotics.**

I. VIRTUAL REALITY AND REMOTE CONTROL

One of the aims of virtual reality is to reproduce real environments and to immerse human beings in order to train them. The built environments have to be as close to reality as possible, in order to give user the feeling of a real situation. Virtual environments generally offer a 3D vision through immersive rooms or through helmet with virtual retinal display. They collect information from the user via sensors and they provide ways to interact with the help of force feedback mechanisms, gloves or other specific devices.

Remote control tends to provide to human beings the feeling of ubiquity, while acting on a remote system with the help of networks and computers, for example in the fields of remote robotics [4], measurements devices [12], home automation systems [9] or industrial web [6]. These systems provide the distant user, information about their state, through sensor values (position, temperature ...), video images or specific messages. These information are transported via networks (from LAN to WAN, with various technologies: Ethernet, Wi-Fi, Bluetooh...) to user displays, generally 2D screens. End user, also, may be able to send commands, through keyboard or mouse clicks, to controlled systems and to see their effects. In such environments, user is supposed to have (or to creat) a mental model of the system he is controlling, in order to understand the provided information and to be able to send the right commands. User is also supposed to be able to take into account time delays due to the network.

In this work, we experiment the use of virtual reality in the context of remote control. One of the first interest is to provide the user a 3D environment of the remote workplace offering him different points of view through virtual cameras. A second interest is to be able to simulate the whole system in order to train future users. But the main advantage of using virtual reality in the field of remote control is to superpose information coming from the real system and information coming from simulation in order to compensate or to anticipate delays induced by the network. This leads to represent on the same environment, different times with a concept of Time Projection. Enriching the user environment will offer a better accuracy and a better quality of experiment for the user.

Section 2 will introduce some usages of virtual reality in the context of remote control. Then in section 3, we will present development framework we used, in order to realize the experiments described in section 4. The latter will also describe in detail the Time Projection Concept we propose. Next section will conclude and open new directions.

II. AUGMENTED AND VIRTUAL REALITY FOR REMOTE CONTROL

Virtual reality has a wide range of applications. A selection is made here to those that, according to us, could be useful for remote control.

- A first field of usage of virtual reality is the simulation of real world(s). A typical example is that of flight simulators, to train pilots to specific situations and to make them learn the behaviours to adopt in critical cases. In such environments, pilots control a virtual plane inside a "real" cabin and get in real time the feeling of being inside a real plane through screens and cabin movements. Other less complicated environments have been developed for training people. [11] describes one for firemen in order to present them a situation in 3D and to offer them a way to choose the right strategy to fight a fire. In this case, users get information through screens and interact with the environment through a keyboard and a mouse.

  In both cases, the real world and users's actions are simulated. Users are immersed in a virtual reality scene, but they are not connected to a real world, which means that their mistakes have no consequence. It's also important to note that the simulated environment reacts in real-time to users's command and that no bias is introduced due to communication delays.
- In the field of remote robotics or measurements devices, users generally get information from the remote system through blackboard and often live video cameras [4].

These cameras are put in specific places (as far as the controlled robot is not a mobile one evolving in a large environment) and are giving views of the ”interesting” parts of the scene. Users take into account these visual information to proceed their tasks.

The experiences we made with students for the control of a 5 Degrees Of Freedom (DOF) robot [5], showed that these information were not sufficient for beginners, because the video camera provides only a 2D vision of a 3D scene. The use of two video cameras (one in front, one at the top) and the gain of experience, improve the control but is not still sufficient to be accurate (see figure 1).

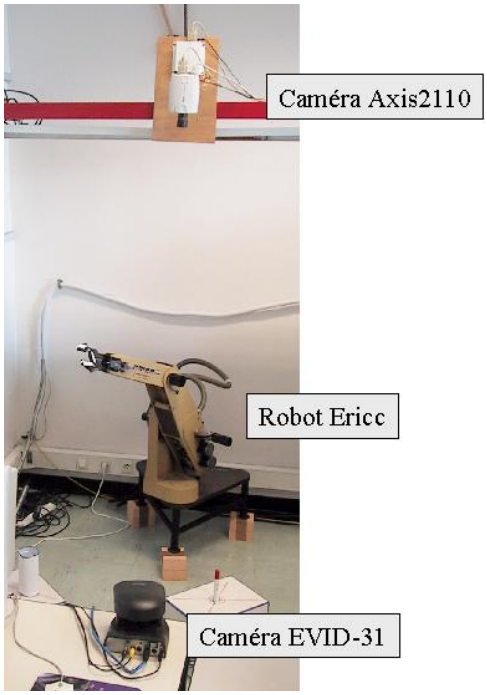

Fig. 1. Testbed

The interest of virtual reality in this case, is to offer different points of view of the scene, as virtual cameras can be place anywhere. Of course, robot's environment has to be known as much as possible. The confrontation of real images and virtual images may help the operator to take the right decisions.

[7] also presents a multimodal interface for controlling a robot arm via the web. The interface presents in two different frames, real images and virtual images to the user. Virtual images come from a prediction module that simulates a virtual robot equivalent to the real one. User can choose to send commands to both robots or only to the virtual one.

- Virtual reality is also used to mix and superpose, real and virtual information on the same display: one speaks then about augmented reality. In the domain of remote robotics, augmented reality has been used for example in Arity [10]. This demonstration platform aims to control a 3 DOF robot, capable to pick and place objects. First of all, it provides a wire frame drawing of the test bed. More it may help user achieving predefined tasks, while giving him virtual guides he has to respect to reach his objectives. Augmented reality is used here to show an environment and to constrain user's actions.

  The augmented reality approach may help in the context of remote control to provide users on one hand more information on the system they want to control and on the other hand to limit their possibilities in order to respect, for example, security constraints.

Previous examples present the interest of using virtual reality in the context of remote control. More, while using a remote system, user has to keep in mind that the network will introduce delays. These can be very small if the network has good properties (in terms of latency) and distances are small. But, in other cases, delays may be longer and the connection may even be lost. In the first case (delays), virtual reality may be used to show the user what will happen in the future while simulating the system's behaviour with some time shift. It may help users to anticipate system reaction. In the second case (lost of connection), the virtual model may inform the user about the status of the ”un-controlled” system and vice-versa. The quality of the system's model will be then important.

We will present this usage in virtual reality in section 4. Section 3 first will introduce the software environment we used for creating our virtual world.

III. DEVELOPMENT FRAMEWORK

The framework used for this work is called ARéVi [1], where A.Ré.Vi. stands for *Atelier de Réalité Virtuelle*, literally ”Virtual Reality Toolkit”. It is a C++ library under the GNU LGPL, for the simulation of autonomous entities and 3D rendering. Its goal is to provide a tool set to simplify the creation of virtual reality applications. Among other features, it provides: 3D objects representation, user interaction, scheduling of activities' entities, and inter-entities communication...

The main technical advantages of ARéVi are its garbage collector, its OpenGL graphic engine, its multi-agent approach and its plug-ins mechanism.

The garbage collector lightens the programmer's workload by managing the memory. Though not as efficient as the latest ”Direct X 10 compliant” graphic engine, ARéVi's graphic engine does its job properly. Particle, transparency, textures, basic shapes... are as much features that it can handle. Moreover, it is quite easy to manage.

Conceived for multi-agent based applications, ARéVi includes some much needed functionality. Communication and delay based scheduled activation are then present. Another interest of ARéVi is that if a functionality came to be missing, it is possible to make it a plug-in to reuse it easily. For instance:

- ArWidget that enables on-screen movable widgets,
- ArAudioVideoRecorder that offers the possibility to create a video file (via ffmpeg) from the screen display,
- hLib that enables rigid skeleton animation,

- ArPhysics that interfaces ARéVi with an external physical engine.

The version of ArPhysics used is based on the Open Dynamics Engine[13]. ODE is an open source physical engine under the BSD Licence. It is aimed to be a high performance library for simulating rigid body dynamics. This is a platform independent, quite simple to use and mature C++ API. ArPhysics implementation is still incomplete, as some functionalities aren't interfaced yet. For instance, cylinders are not fully supported, while boxes and spheres do work fine. So goes for the absence of grip, and some few other parameters. However, almost everything useful for a physical engine functions: gravity, contact and forces computation, and so on.

IV. EXPERIMENTS

The framework presented in this paper is alike those described in [3] and [8]). It is a generic robot simulator, at the exception that the approach is more focused on time apprehension. Here, the goal is not only to compensate delay, but also to try to abstract this constraint.

A. Modelled Robots

For the experiments, two robots have been modelled.

The first one is a 5 DOF robot arm : *Ericc*. Its purpose was to put in place the modelling system, and to work around collision avoidance. The other one is a small mobile one : *Miabot*. The experiments made with this robot, tent to extend the modelling system, and to work on anticipation.

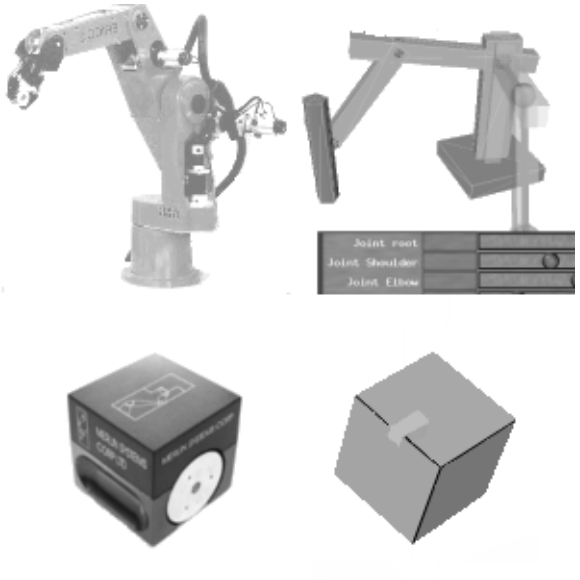

Fig. 2. Real (left) and virtual (right) Ericc (top) and Miabot (bottom) robots

As pictured by Figure 2, virtual models are simpler than real one. ARéVi is able to display more complex image than this. It is a choice to minimize the amount of information displayed: since the same scene might be displayed two to three times, it lowers resources consumption and increases the understanding.

Each robot has been described through an XML file, containing its geometric shape and its characteristics. This file defines whether the robot can move, can be pushed, can rotate on itself, or how it is articulated. It also defines its mass and other physical parameters. If the robot is articulated, the shape describes each joint, with its constraints, properties, and attached parts. The shape is a combination of parts (facultative) in a part used as reference. Each part is either a box, a cylinder, a cone, or a sphere, with properties such as colour, texture, positions...

B. The Time Projection Concept

The main novelty in this work is the *Time Projection Concept*. It relies on the idea that, to improve the quality of the control of a remote system, it is necessary to compensate latency induced by the network. This idea can also be founded in [2] , which describes Predictive Interactive Graphical Interface (PIGI) to control NASA's robot.

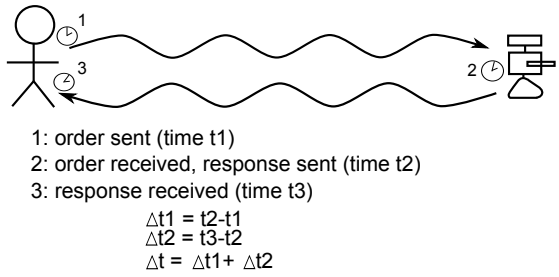

Fig. 3. Transmission delays

Figure 3 represents an user sending an order to a remote robot. The user sends the order at a time *t1*. The order travels to the robot, and reaches it at time *t2*. User at time *t3*, will get information from the robot through video images, sounds or a blackboard.

To "hide" the delay, the test environment may be "virtualized" and orders sent to the real robot can be also send to a virtual one. In this case, the virtual robot will be in advance of $\Delta t1$ time units ($\Delta t1 = t2 - t1$). One can also delay orders sent to the virtual robot of $\Delta t1$ time units, to present user an estimated view of the robot. Moreover, orders may also be played faster in the virtual environment (as example, increasing the movement's speed) to be in advance of a predefined number of time units compared to the real system.

These different views may also be presented to user simultaneously to provide him more information. This need, while building the virtual environment, to use the Time Projection Concept.

The Time Projection Concept tries to give the user benefit from all solutions: fiability of a response driven simulation, coherence of a lag-compensating simulation and predictability of an advance-lag simulation. Instead of simulating a scene (robot & environment), projections are simulated. A projection is a portion of space (the scene) coupled with a clock (the

delay). For instance, a lag compensating simulation is a projection, which clock advances from "lag time" units.

The use of time projection is double. Firstly, the projection is related to a scene at a specific moment. This moment evolves independently from the real time's flow. This means, if the simulation is correct, two projections of the same scene (at different time) must be identical when their clocks are the same (at the same instant). This special relation to the time authorizes to benefits various speed evolution, thus easing the evolution's representation to the user.

Secondly, the simulation should display not a single, but as many as needed projections. Each projection being related to a specific moment, it is possible to display as many moments as wanted. Therefore the simulation can display at the same time several "instants", for example: the known, the estimated and the anticipated situations.

C. Tests

The Time Projection Concept has been used with our two robots, in order to test its interest.

1) Ericc, robot arm: Ericc is a 5 DOF robot arm, with a gripper. At the exception of this last part, not modelled, this robot has been described via an XML file and a few lines of code to exploit the interaction. Each articulation is controllable independently, with respect to its constraints (axes, angle). The control can be made through a sliding-bar widget, to intuitively move the model, or by specifying the order directly (example: *RA1,90* which means *rotating articulation 1 of 90 degrees*).

To control this robot a "move and wait" strategy is used. It means, that from a stopped position, user will specify the next target position. The robot will reach its new position and then stop. Then user will fix a new target and so on.

The user will see on his display a view of the actual robot's position and, with the help of sliders that control joints, will indicate the target position (figures 4A and 5A). The first interest of virtual reality in this case is that the user can easily and precisely indicate the desired position with the help of virtual cameras.

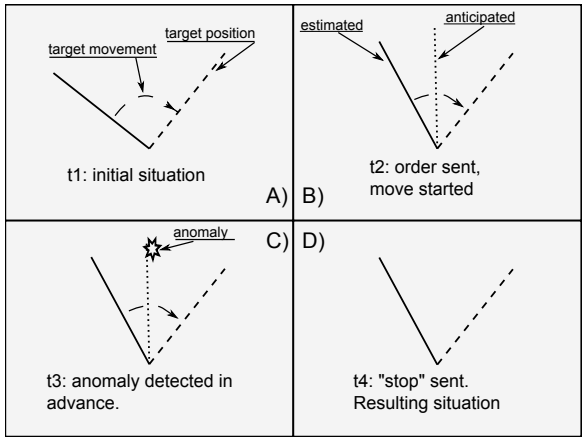

Fig. 4. Ericc test case

Then, while pressing the "start" button, orders will be send to the real robot and to the virtual environment. Using the Time Projection Concept, two virtual robots will be displayed on the screen of the user : a first one that will represent the estimated position of the real robot (time delay = $\Delta t1$, estimated by computing $\Delta t/2$) and a second one that will represent the anticipated position of the real robot (time delay = $\Delta t1$ + θ, where θ is a constant chosen by the user), as shown figures 4B and 5B.

The interest of the estimated position is to give users an idea of what happens really in "actual time": users do not have to wait $\Delta t2$. It also may be fruitfully used if the network connection is broken, to have an idea of what's happen in non-longer controlled environment...

The interest of the anticipated position is to see in advance, if something wrong will or will not happen, as a collision (figures 4C and 5C). Indeed, users will perceive θ time units in advance the position of the robot and is then able to send an emergency stop if necessary (figures 4D and 5D)

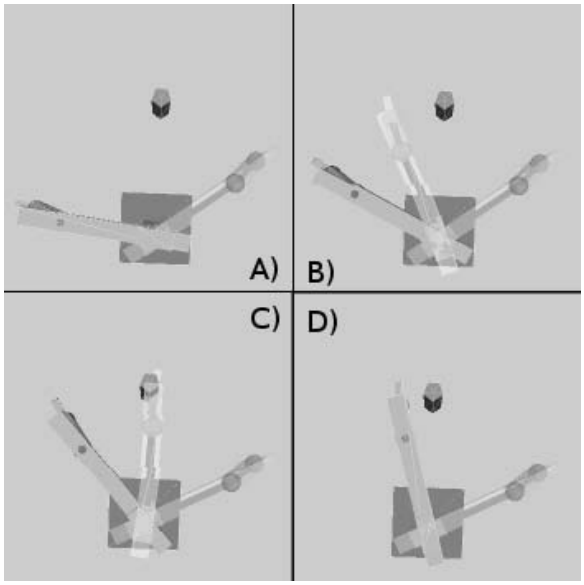

Fig. 5. Estimated, anticipated and target virtual Ericc(s) in Arévi

The "move and wait" strategy may also be used in domotic, where being able to represent an estimated situation and an anticipated situation may certainly be useful. In this case, the value of θ could be adjust to minutes or hours, depending on what effect user want to anticipate.

2) Miabot, mobile robot: Miabot is a small mobile robot moving in a closed area [14]. Users get information on his position with the help of a live video camera. Its shape is basic (a cube), and therefore the virtual model didn't need specific analyses. It is described by an XML, and a few lines of code to enable user interaction. It can move forward or backward and turn left or right. These movements can be ordered by 4 buttons + a stop one. If the mobile is moving while an order to turn is given, the two movements are combined, and reciprocally. If a behaviour (moving or turning) is ordered while already in use, only this order will remain executed.

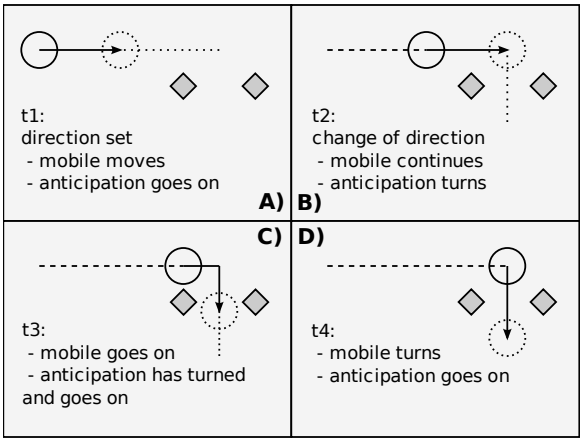

Fig. 6. Miabot's mission

The figure 6 described a mission a user would like to achieve. The robot is moving straight on and it has to turn to pass a door, represented by the two diamonds on the scheme. User has to decide when to press the turn right button which is not obvious if he only gets information from the camera. Parallax and distortion problems are the first pitfalls and of course the delays induced by the network.

Use of virtual reality may easily solve the two first problems while using virtual cameras. For the last problem, Time Projection Concept may be used. User, will be displayed the estimated position of the robot and an anticipated position (figure 7B). The latter is in advance of Δt time units and corresponds to the position where the robot will receive an order if the user sends it now. In fact, user will concentrate on the anticipated virtual robot to control the real one (figure 7C and D).

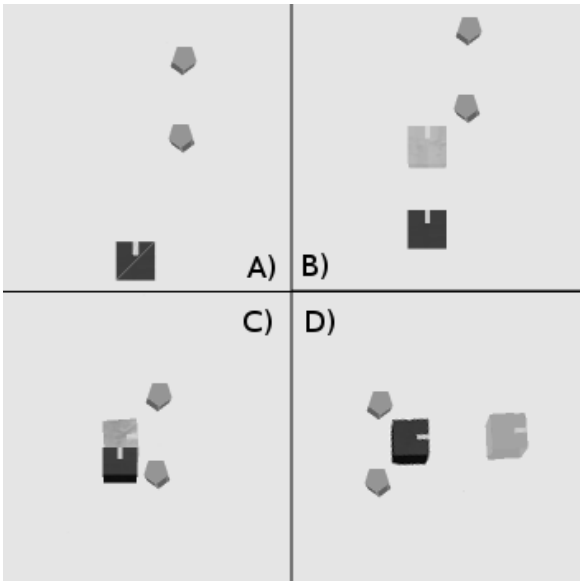

Fig. 7. Virtual Miabot(s) in Arévi

V. CONCLUSION

In this paper, we present the interest of virtual reality for remote control. Virtual reality may help users to have a better view and understanding of the remote system they want to control. It also may be used to compensate delays induced by the networks while using our Time Projection concept.

We present our framework and two realizations done with robots, with two different remote control schemes (move and wait and continuous) and the advantage of using virtual reality in both cases. This work may be applied to other domain, such as measurement systems, as soon as they are remotely operated.

In the future, we also like to improve the interest of our approach with end-users to estimate gains realized while using the Time Projection Concept. We also have to work on the synchronization between the real world and the virtual world, what is rather simple to achieve with a move and wait strategy and much more difficult for a continuous one. We also would like to be able to propose a generic model for all kind of ubiquitous applications.

REFERENCES

[1] CERV - LISyC, 2010. Arevi. [Online : svn.cerv.fr/trac/AreVi/Wiki].
[2] Burridge R and Hambuchen K., 2009. Using prediction to enhance remote robot supervision across time delay. *2009 IEEE/RSJ International Conference on Intelligent Robot and Systems*, St. Louis, USA, pp. 5628 – 5634.
[3] Freund E. and Rossmann J., 1997. How to control a multi-robot system by means of projective virtual reality. *ICAR*.
[4] Goldberg K. and Siegwart R., 2001. Beyond Webcams : an introduction to online robots.*The MIT Press*.
[5] Kaddour el Boudadi L., Vareille J., Le Parc P. and Berrached N., 2007. Remote control on internet, long distance experiment of remote practice works,measurements and results. *International Review on Computers and Software (IRECOS)*, vol. ISSN 1828-6003.
[6] Le Parc P., Touil A. and Vareille J., 2009. A Model-Driven Approach for Building Ubiquitous Applications. *The Third International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies* - UBICOMM 2009 - Malta.
[7] Marin, R. and Sanz, P.J. and Nebot, P. and Wirz, R., 2005. A multimodal interface to control a robot arm via the web: a case study on remote programming. *Industrial Electronics, IEEE Transactions on*, vol. 52, no 6, pp 1506 – 1520.
[8] Michel O., 2004. Cyberbotics ltd - webotstm: Professional mobile robot simulation. *Advanced Robotic System*, vol. 1, no. 1, pp. 39 – 42.
[9] Nokia, 2009. Smart home solution, [Online : smarthomepartnering.com/cms/].
[10] Otmane S., Mallem M., Kheddar A. and Chavand F., 2000. Active virtual guide as an apparatus for augmented reality based telemanipulation system on the internet. *IEEE Computer Society "33rd Annual Simulation Symposium ANSS 2000"*, Wyndham City Center Hotel, Washington D.C., USA, pp. 185 – 191.
[11] Querrec R., Buche C., Maffre E. and Chevaillier P., 2004. Multiagents systems for virtual environment for training : application to fire-fighting. *Advanced Technology for Learning*.
[12] Santos J., Mendonca J. and Martins J.C., 2008. Instrumentation remote control through internet with PHP. *IEEE Conference on Virtual Environments, Human-Computer Interfaces and Measurement Systems, VECIMS 2008*, Istanbul, pp. 41 - 44.
[13] Smith R., 2009. Open dynamics engine. [Online : www.ode.org].
[14] Zhong S., Le Parc P. and Vareille J., 2007. Internet-based teleoperation: a case study. *Fourth International Conference on Informatics in Control, Automation and Robotics (INCINCO'07)*, vol. 1, pp. 267–270.