Networking Issues for 3D Medical Collaborative Virtual Environments: Design and Applications

Paulo V. F. Paiva[*], Liliane dos S. Machado[*], Jauvane C. de Oliveira[†] e Ronei M. de Moraes[*]
[*] *Laboratório de Tecnologias para Ensino Virtual e Estatística (labTEVE)*
Universidade Federal da Paraíba (UFPB), João Pessoa, Paraíba
Emails: paulo.fariaspaiva@gmail.com, liliane@di.ufpb.br, ronei@de.ufpb.br

[†] *Laboratório Nacional de Computação Científica (LNCC)*
Petrópolis, Rio de Janeiro
Email: jauvane@acm.org

***Abstract*—Collaborative Virtual Environments (CVEs) are Virtual Reality (VR) systems and can simulate medical procedures such as surgeries performed by a group of remote users which can interact with one another through the simulation in real-time, with the aid of a network such as the Internet. Therefore, it is important to note that research on CVE-specific networking issues allows us to better understand the requirements for real-time applications. This paper discloses design and implementation issues of a peer-to-peer multicast architecture on the collaborative module of the*CyberMed* VR framework, aiming at providing better network scalability, as well as it shows as results, some developed applications as validation. Finally, we compared the performance between the newly implemented peer-to-peer multicast architecture with the previously existent peer-to-peer unicast. The evaluation of the networking performance was conducted with *OPNET Network Modeler* and *R* statistical software. As result of our experiments, the null hypothesis was rejected (p = 2.2e-16, level of significance = 5%) confirming what we expected about the behavior of *CyberMed's* multicast protocol.**

***Keywords*—virtual reality, collaborative environments, medical training, networked simulation, performance.**

I. Introduction

Collaborative Virtual Environments (CVEs) are Virtual Reality (VR) systems in which multiple users can interact with one another in real-time in order to perform a given task, in a collaborative manner [1], [2][3]. In such systems, the users exploit deeper levels of realism while they perceive remote users actions, such as if they were sharing the same space on the real world [4]. These systems provide immersion sensation to users, while it uses special VR devices which explore multiple human senses (e.g. data gloves, haptics). Medicine has largely benefited from CVEs, which are especially useful for training applications [5], [6].

The CVEs enable the practice of medical procedures by users who can share experiences with a remote tutor or other students. Such applications are also known as VR-based medical simulations and allow users to practice in a real-life scenario, without risking a real-life patient. VR systems may be quite realistic, both visually, when users can see and interact with life-like 3D models of virtual organs in a virtual body, as well as behavior-wise, when such structures react properly when touched, squeezed, cut, pinched, etc. Additionally, a VR simulation allows users to be exposed to both common everyday scenarios, as well as rare conditions/complications, allowing them to gain experience in such circumstances as well [7].

A sub-category of the CVEs, named Collaborative Haptic Virtual Environments (CHVEs), enable users to identify textures and material properties (e.g. stiffness, resistance, weight and elasticity) of virtual objects through the use of haptic devices which explore the human tactile sensation [8], [9]. Considering that CHVEs have a higher network usage, due to the fact that tactile information must be sent in addition to the usual data throughput of non-tactile-oriented CVEs, we can observe how such systems are sensitive to networking issues. It is harder to maintain real-time requirements. The haptic dataflow on these systems must be properly managed, as it has direct impact on the quality of users perceptions during the collaborative interaction. There are some studies on CHVEs performance evaluation which predict a growth in its use over the Internet and emphasize the importance of improving haptic data communications [10].

The use of delivery network protocols such as multicasting allows the partitioning of the CVE, which separates its traffic into different data flows according to a variable demand of clients requisitions, which is considered to be an usual communication approach in multimedia systems. The multicasting protocol has the advantage of providing a better scalability, especially when it is used together with peer-to-peer architectures, which dismisses the overhead imposed by central servers in client-server architectures [10]. The *CyberMed* framework has been created aiming at facilitating the development process of VR based medical simulations [11]. Its collaboration module, named *CybCollaboration* [12], has been recently added with new networking functionalities such as a traffic analyzer and multicasting protocol class that handle group communication, in order to provide better use of network bandwidth. The newer implementation also provides flexibility to users to build CVEs by combining the type of 3D object manipulation (free or blocked), and interactive device desired (haptics, data gloves and magnetic/optical trackers). This work presents the implementation issues of *CyberMeds* collaborative module. It also presents the results of a performance evaluation of a comparative study between unicast and multicast protocols for medical applications, improving the results some found in the literature, as well as some developed applications as initial results. The rest of this work is divided in sections II through

V, respectively addressing an introduction to CVE network architectures, *CyberMeds* collaboration module, as well as its implementation issues and applications, which is followed by final considerations covering performance evaluation through network simulation and statistical analysis. Preliminary results of this research have been reported [12], [13], [14].

II. Correlated Works

This section presents similar studies found in the literature that addresses network protocols and architectures, as well as collaborative virtual environments which are applied for medicine field. VR simulations have been applied to medical applications in different ways, such as: training and education, surgical team training and students evaluation, surgical planning, neuropsychological assessment and rehabilitation, distance diagnosis, simulation of dynamic virtual patients, beyond others. As an example of recent application of CVE to medical education are simulators designed to support surgical team training and students assessment (single and multiple users), such as: 3diTeams (Figure 1-b) [15], SimCEC (Figure 1-a) [7], Virtual ED (Figure 1-c) [16], and others [5].

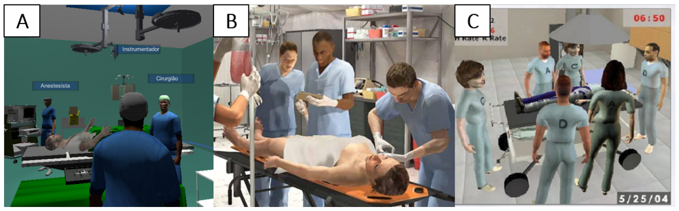


Fig. 1: CVEs applied to surgical team training and evaluation: A) SimCEC B) 3diTeams and C) Virtual Emergency Department

Boukerche et. al. [17] presented a hybrid multicast protocol built with a set of mechanisms and protocols aiming at satisfying CVEs requirements such as scalability, network delay, reliability and synchronization. The authors considered four main protocols: the Synchronous Collaboration Transport Protocol (SCTP), the Selective Reliable Transmission Protocol (SRTP), the Reliable Multicast Transport Protocol (RMTP) and the Scalable Reliable Multicast (SRM) and finally presented a hybrid protocol that is able to satisfy all the CVE requirements. The proposed protocol has a multicast tree architecture in order to avoid the congestion and delay problems while it uses three modes of transmission to ensure reliable data transmission. Implementation issues were discussed within a Brain Tumor Tele-Surgery application case study.

Correia and Pellegrino [18] presented an implementation of application layer multicasting within the AVDNet framework. Their application implements mechanisms for controlling the creation and destruction of groups without using the low level socket API. It also manages users access to the system. The work provides the scalability required to enable that hundreds of users can simultaneously interact. The proposed strategy is implemented as a communication module of the AVDNet framework. Thus, the implementation of real-time interaction and collision detention of objects is allowed, maintaining the synchrony and consistency of the state of the objects. Scalability is achieved by allowing the use of applications developed for any Internet user, regardless of infrastructure conditions. The entire VE was partitioned in different regions which are stored and controlled by different workstations acting as servers of regions.

Therefore clients can communicate with different servers, while moving through the CVE. VEs partitions were also used by Greenhalgh and Benford [19] in a multicast architecture proposed as a general spatial model of interaction in large scale CVEs. In the later architecture the environment was mapped into a hierarchical data structure and managed by various multicast groups. This mechanism is known to reduce overhead imposed by many clients connections, especially in multiuser collaborative virtual environment. In the context of VR frameworks, some of its usual features are: the integration of different interactive devices, including tactile ones; the modeling of complex organs of the human body; and the possibility of remote collaboration. Such comprehensive feature set makes it a challenging task to develop such frameworks. One example of VR framework designed to support collaborative medical simulators is the GiPSi framework, with its GiPSiNet network module [20]. GiPSis features include: tactile communication; modularity through data encapsulation; provides automatic support to a number of surgical simulations; network bandwidth optimization, with automatic handling of network delay, jitter and congestion.

III. Technical Aspects on Collaborative Medical Simulators Design

Its known that VR medical simulators require higher levels of realism in order to provide proper immersion [21]. Much of these requirements imposed by VR medical simulators are caused by the use of complex organs models, object deformation and collision algorithms, which can provide simulation for cutting, suturing and puncture procedures [8]. Considering the real-time requirements of CHVEs, it is necessary to study the network impairments, which can degrade the performance of such applications. Network-wise, the main problems faced by CHVEs are related to network bandwidth, jitter (variation of delay) and data throughput of haptic devices [22], [23]. Moreover, these devices require a higher sampling rate, which is needed due to the high sensitivity of human touch, which increases the problem of data throughput once they are handled in a collaborative setup. Some research reveals that network delay leads to the eventual lack of synchronization among different VE instances spread across a network. Another example is undesirable effects such as the users hand crossing a given object border before the system has an opportunity to react. Additionally, packet loss can reduce the intensity of force-feedback felt by users, as it can also cause abrupt force feedback to be applied.

In this section, we cover topics related to technical aspects related to CHVEs design, network architectures and communication protocols, such as: multicast, synchronism, collaboration approaches, CHVEs architectures [24]. Importantly to observe, that collaborative tasks in CVEs usually occurs in two ways with different names found in literature [25]: a) simultaneous (concurrent or synchronous) collaboration or b) sequential (asynchronous) collaboration. In the first case, remote users operate modifications on the same or different properties of the same virtual objects at real-time, such as when the surgeon and surgical assistants operates on the same operative field or vital organ on the patient. In the second collaborative approach, user actions occur sequentially. For an example, when the anesthetist induces the patient to the state of unconsciousness anesthesia so that the surgeon can start the operation, among other possibilities.

Still on the context of users actions integration, the use of on demand protocols such as Multicast can support the creation of different users groups according to specific tasks performed in a surgery (e.g. only the users belonging to anesthesia team should receive the data related to the anesthesia task). Another possibility would be to send the position data of a vital organ or tissue, as well the forces emitted by a haptic device, just for those members which cooperate performing a procedure on that organ [12]. Multicast operates in one-to-many networking communication architecture, where the source transmits a single data package to a group of interested receivers [17]. With this protocol, a host chooses to join a multicast group by specifying the IP of the group that it wants to receive updates from. Thus, the use of bandwidth resources can be optimized while the data can be delivered on demand.

In early research related to graphic simulators with multiple users, Singhal and Zyda [4], suggest some alternatives to solve the lack of synchronization between different instances of the same VE spread by network , such as: active replication states and prediction algorithms (dead- reckoning). For example, the exact path of medical tools and surgical team members positions can be precisely measured. The active replication it is the most trivial mechanism and more costly, where each host is responsible for sending periodic messages containing the current state of their local copies, to be updated in the remote clients. The dead-reckoning algorithms, in turn, are based on the calculation process of the current position of the given object from a predetermined position, advancing along the time and route. To this end, data from its last known cinematic state are used (e.g. position, velocity, acceleration) [9]. Finally, in the context of tele-haptics, the collaboration may be implemented based on different approaches (Figure 2), according to haptic interaction and feedback:

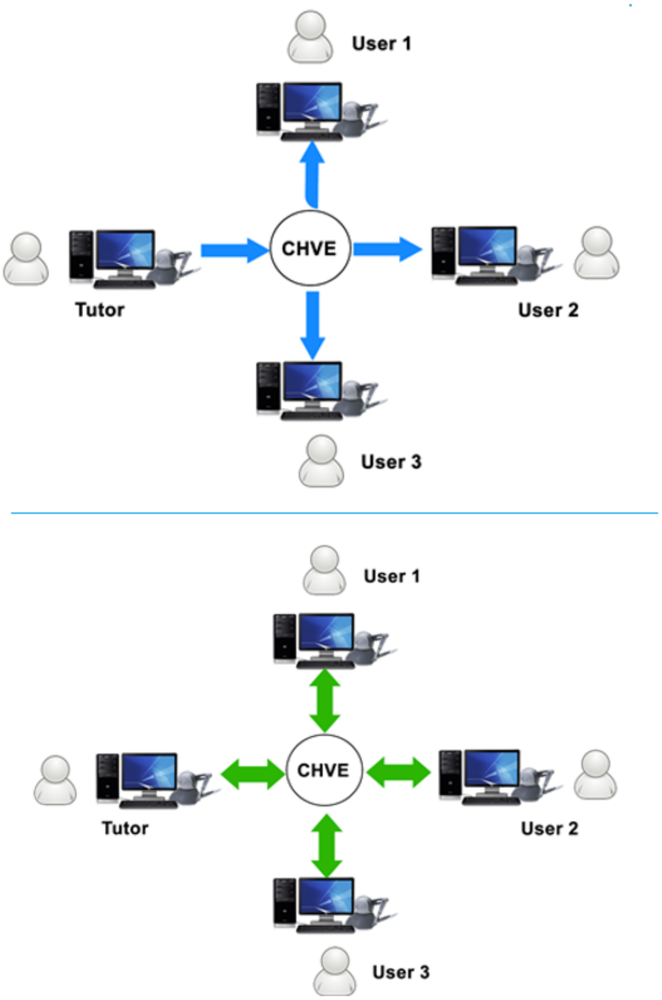


Fig. 2: Architectures of Collaborative Haptic Virtual Environments (CHVEs). Unilateral (top) and bilateral (bottom) tele-haptics

a. Unilateral tele-haptics: A given user owns objects and remote users are able to haptically feel the owner interactions with his/her object.

b. Bilateral tele-haptics: Multiple participants haptically interact with identical or coupled objects and each participant feels one another manipulation in the VE. It is usual in tele-surgery simulators, where a surgery is performed by various surgeons at the same time. In the context of our work, its named as **Concurrent Collaboration**.

c. Tele-mentoring: Allows direct coupling of haptic devices through network architecture such as the Internet. A typical example of such setup is training for surgeons, whereas a trainee can feel a remote expert guiding his/her hand throughout a surgery procedure. We name it **Tutoring Collaboration** .

IV. Collaboration in the CyberMed Framework

The *CyberMed* system is an open source framework designed to support the development of VR-based medical simulations and its written in C++ [26]. The framework allows programmers to build their own VR systems in a higher level of programming, without the need to deal with most of VR-specific coding. The system also allows one to extend it with new modules [12] and it's supported by both Windows ans Linux operating systems. The features provided by *CyberMed* include: monoscopic and stereoscopic visualization; support to common VR devices, such as haptic devices, motion trackers, head-mounted display and data gloves; different algorithms for collision / deformation of virtual objects; an AI module for users assessment [27]; collaboration and network communication, visual programming interface [28], amongst others.

A new feature (implemented through a new data structure) was recently added to the framework, in order to support volumetric 3D objects [11]. Such new feature allowed the development of simulation of medical procedures that required actions such as cutting and suturing virtual organs as it's covered in previous works [29], [14], [12]. The collaboration module, named *CybCollaboration* , has been recently added to its architecture. It allows programmers to build complete medical CVEs with just a few lines of code. Among several functions, the main roles of this module are: a) establish connection between remote users; b) manage the exchanging interactions between all participants through a collaborative protocol; c) support different types of object manipulation; and d) enable users to use common VR devices during a collaborative task. There is two object-handling approaches implemented in CyberMed. In the first approach, all users are free to interact and handle virtual objects simultaneously. In the second approach, only one user at a time is allowed to handle an object while it remains blocked to other users. This is very common in tutoring applications were just the tutor is designated to perform actions in the VE while all remote students follow his/her steps. The first version of *CybCollaboration* was initially implemented with a very simple and non-scalable network setup, with a hard-coded peer-to-peer UDP/IP protocol which did not give the possibility of developers to implement CVEs with flexibility to choose network protocols and interaction modes.

The *CybCollaboration* architecture is divided in three parts according to its specific functions. The core of the module contains operations and information to be used by all the other classes and it is formed by three classes: *CybCollaboration*, *CybCollaborationProperties* and *CybCollaborationIntegrant*. The *CybCollaboration* class is the main class of the module and contains a linked list with information of all participants as well as the implementation of the collaborative protocol which handles the communication between users. The main properties of the collaboration are stored in the class *CybCollaborationProperties*, which includes operations to enable or disable collision and deformation and to choose the desired type of manipulation in virtual objects (e.g. free or blocked).

The *CybCollaborationIntegrant* class, in turn, represents a member of the collaboration and stores information such as name, IP address and a reference to the local device representing the user in the VR world. The second part of the module aims at supporting collaborations with VR interactive devices and its classes: *CybMouseCollaboration*, *CybHapticCollaboration*, *CybGloveCollaboration* and *CybTrackerCollaboration*. It includes the handling of particular implementations for each of these equipments. All these device classes inherits from *CybCollaboration* and implements its abstract method *getMainInterator()* in order to return the specific interator of each device to be used in collaboration [29]. The communication module (named *CybNetwork*) is a set of classes that composes a software layer above the low level Socket API and it is responsible for providing network communication for CyberMed applications.

The relationship of the collaboration module with CybNetwork is established on three classes: *CybUDPServer*, *CybReceiverThread* and *CybNodeAdress*. The first is responsible for creating a node in the network using the UDP protocol. The second acts as a listener of network events containing an operation that allows us to see incoming packets at the socket. The third class stores address information of a particular node, such as the IP address and port number sets, among others. The former implementation of *CybCollaboration* class had an instance of *CybUDPServer* socket through which it communicated via unicast messages. Thus, the module had this particular limitation once this socket was hard-coded leaving no option for the programmer to choose between the transmission protocol (TCP or UDP) or the communication model (unicast, multicast or broadcast) for his/her CVE. In order to allow this module to be more flexible and also to enable the development of larger scale CVEs, also supporting multicast and broadcast protocols, *CybCollaboration* recently passed through modifications which are described in the next sections.

V. Development: Implementation Issues

In order to provide flexibility in choosing the network communication model as well as the inclusion of the multicast protocol, several changes were implemented on the collaboration and network modules of *CyberMed*. First of all, it was necessary to implement a new multicast class which could support the transmission of packets with both unicast and multicast communication models in a peer-to-peer architecture, once the oldest classes were just based on client-server architecture [12]. Such setup was required due to the fact that certain messages had to be sent directly to a given user, as in the case of the acknowledgment message which is always directed to the CVE creator.
The *CybMulticastPeer* class was then created and added to the network module, supporting multicast in a peer-to-peer approach. Therefore, this class aims at simulating a host which belongs to a multicast group, to be able to send and receive data via UDP. *CybMulticastPeer* inherits from *CybUDPServer* allowing it to use the methods of transmission and reception of data which were already defined on the super-class. Thus, creating a *CybMulticastPeer* object, we are automatically instantiating an object type of the super-class *CybUDPServer* forming two sockets: one for sending data to the multicast group and the other for sending via unicast to a particular participant.

Specific operations concerning the multicast protocol, provided by *CyNetwork* module, were added to such sub-class. The network model of *CybCollaboration* is still based on peer-to-peer distributed architecture classes displaced by *CybNetwork* module. However, with the newest implementation, there is more liberty for switching between multicast, unicast and broadcast communication modes as well as TCP and UDP transmission protocols. The changes were made on how *CybCollaboration* instantiates the socket and how it uses *CybNetwork* operations.

It is worth mentioning that this network communication parameter can be assigned in CVE's runtime. That defines the way messages will be distributed. Now, a socket factory class returns instances of all kinds of sockets at runtime. Thus, the *CybNodeFactory* has been created based on the Abstract Factory design pattern [29] enabling the instantiation, the configuration, and the return of sockets according to the type of network required by the user at runtime. The *CybCollaborationProperties* class stores the chosen network communication setup so that the entire module can be aware of it (Figure 3).

The reported setup brings more flexibility to the development of CVEs, through the use of polymorphism mechanisms and design patterns [29]. Nowadays, it is only necessary to assign the desired type of network communication as a parameter in the constructor of the *CybCollaboration* object. Finally, its also developed one traffic analyzer class based on

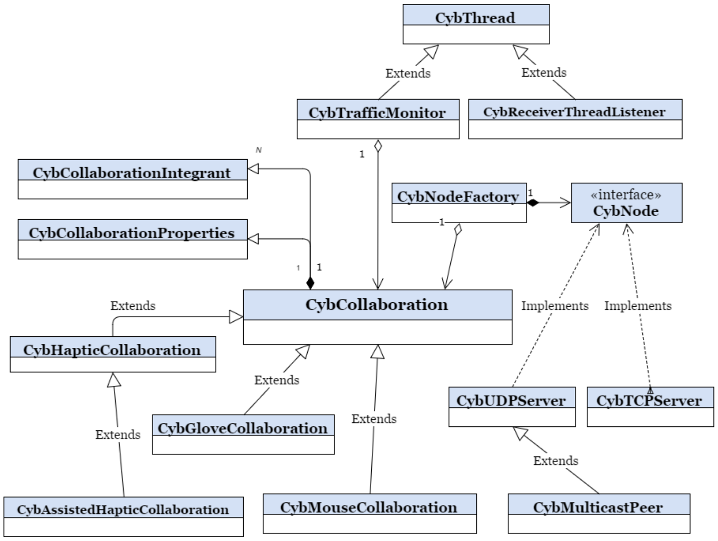


Fig. 3: The *CybNodeFactory* interaction with other networking classes.

the collaborative protocol. The *CybTrafficAnalyzer* class works as one parallel thread (it inherits from the main thread of the framework, *CybThread*) which monitors all messages that are shared by network.

VI. Results: Applications and Performance Evaluation

In this section we have three main objectives: Firstly its presented some CVE applications developed with the aid of *CyberMed* in order to validate its new architecture and to prove how flexible the framework became in the perspective of developers. Second we analyze the performance of multicast and unicast protocols with a statistical non-parametric method (Students t) in order to compare the packet delay levels between two independent samples. The R Statistical Software was used for calculating such tests after generating our samples. Finally, we present the methodology and the results of a performance evaluation process made with *OPNET Modeler (OM)* network simulator. That allowed us to simulate the network traffic behavior of these applications.

A. CVEs Applications Supported by CyberMed

After all modifications, coding with *Cybcollaboration* module became more flexible and intuitive as stated in [14]. More robust applications can be built with few code lines. The module supports CVEs with multiple users collaborating with each other in different communication approaches (one-to-one, one-to-many, many-to-many). Also, CVEs are built by combining the type of object manipulation (blocked or free), the network protocol (udp/ip, tcp/ip or multicast), and the interactive device desired (haptics, data gloves, magnetic/optical tracking) (Figure 4).

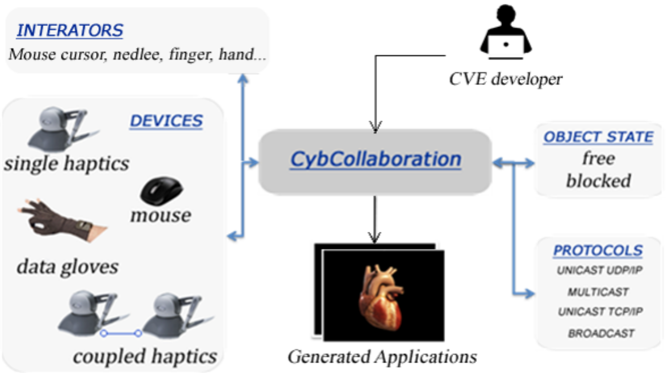


Fig. 4: Integration of CybCollaboration features.

The simplest kind of collaborative application allows users to freely manipulate virtual objects, applying translation and/or rotation with their mouse devices (Figure 5) as well as to use keyboard to apply others transformations (e.g. scale, transparency) and setup stereoscopic visualization. The *CybCollaboration* module also supports the use of data gloves, in remote users interaction within the CVEs. For the development of a collaborative test application, it was used an data glove from the family of 5 sensors (Data Glove 5 Ultra 5DT). In this case, it was necessary to specify how the new remote glove would update the 3D hand model. Once the gloves have five sensors, it was necessary to reuse the value of each sensor as a pivot point for the 3D hand model. So for the same model we have two ways to update it depending on the connected glove.

Another type of collaborative application is what is known as *Tutoring Collaboration* or Tele-Mentoring system [12], implemented by the *CybHapticAssistedCollaboration* class, where one host is designated as a guide and its haptic device guides the remote users movements through their own devices. Such particular way of collaboration is frequently used by systems with educational purposes where the tutor assumes the leading role within the one-to-many communication approach. As a way to illustrate how such applications were developed in an easy way, we present a piece of the source code (both client and server side) of this CHVE. As it can be seen, the framework just receives the necessary parameters, such as IP address and protocol type (Figure 6.A lines 2, 3), to handle all haptic communication in a lower level.

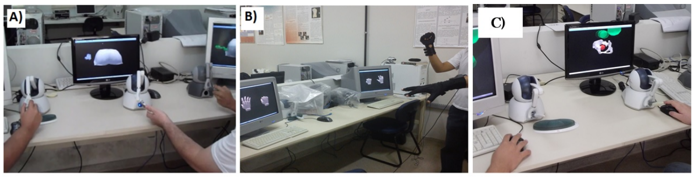
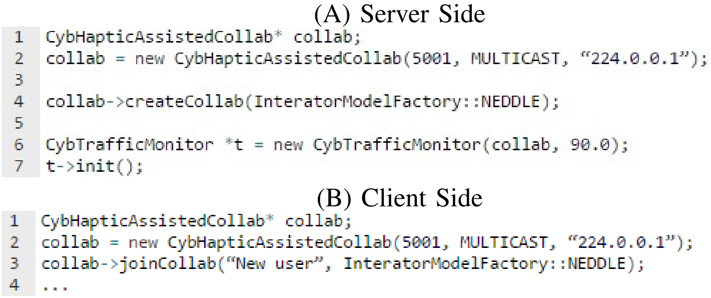


Fig. 5: Tutoring CVE with bidirectional tele-haptics, developed with *CyberMed* . The user at left controls the movements of other remote users.

B. Network Simulation

The networking simulation process enables the representation of protocols, architectures and communication technologies, allowing the study and evaluation of its traffic behavior and overall performance. Therefore, the simulation process is a



(A) Server Side

```
CybHapticAssistedCollab* collab;
collab = new CybHapticAssistedCollab(5001, MULTICAST, “224.0.0.1”);

collab->createCollab(InteratorModelFactory::NEDDLE);

CybTrafficMonitor *t = new CybTrafficMonitor(collab, 90.0);
t->init();
```

(B) Client Side

```
CybHapticAssistedCollab* collab;
collab = new CybHapticAssistedCollab(5001, MULTICAST, “224.0.0.1”);
collab->joinCollab(“New user”, InteratorModelFactory::NEDDLE);
...
```

Fig. 6: Examples of implementing CHVEs with CyberMed after *Cybcollaboration* improvements (A- server initialization, B- client connection).

rather complex methodology [30]. It is necessary to observe the correctness of all input parameters and follow one pre-defined methodology so that the output errors may be minimized. In this work, we performed a simulation of CHVEs developed with *CybCollaboration* aiming at evaluating our collaborative protocol.

The simulation process was developed according to a systematic methodology which involved four sequential phases: (1) the requirements analysis of the simulation, (2) the simulator selection, (3) learning how the simulation tool works and finally (4) the implementation of the network model of *CybCollaboration* , simulation, data collection and analysis. In the first phase, the basic requirements chosen for implementing the simulation of our collaborative protocol were: transmission rates from the haptic device used in our tests, the transmission protocol and its packet format (they have from 18 to 20 bytes of payload, not including the UDP and Ethernet headers) as well as the representation of each message types of our protocol.

Another important issue during the simulation requirements study was to define the network metrics that would be used to evaluate our collaborative protocol, and it was observed in literature that packet delay (ms) and throughput (Mbps) is usually used for this purpose. After defining these metrics, it became important to define the acceptable levels of delay and throughput which wouldnt compromise the CHVE's reliability. These baseline levels to our evaluation metrics were found in literature, more specifically in the studied literature [31], [32], [33] which presents the necessary requirements for supporting haptic data transfer, as well as other kinds of data transfers, such as video and voice. It suggests that packet delay should be in the range of 50 80 ms.

It was then necessary to choose a simulation tool that could allow us to implement the network behavior of the CybCollaboration setup in a very realistic manner. After some evaluation of the options found we chose the OPNET Modeler (OM) [30]. The OM is a simulator of discreet events which allows the creation of network representations, called models, that properly simulate the network behavior of network protocols, architectures and alike. The chosen simulation tool also enables the use of Probability Density Functions (PDFs) for producing and setting up different kinds of data traffic patterns. Ours simulations run for two different scenario sets, considering both unidirectional and bidirectional tele-haptic architecture in order to represent Tutoring Collaborations and Concurrent Collaborations, respectively. The node model, which simulates the behavior of each host within the network, has four sub-units called modules and implements some local CHVE specificities: (1) haptic: this module simulates the behavior of the haptic device while generating its data. It was necessary to define both inactive (off) and active (on) states in which the process may remain, to properly simulate the haptic device maneuvering by users; (2) application: Simulates the protocol of *CybCollaboration* that is responsible for establishing all clients connections, sending, receiving and processing all packets. (3) transmitter: Module that represents a network interface for transmitting packets; (4) receiver: Module that represents a network interface for receiving the incoming packets.

The probability distribution used for transmitting and receiving packets were the Exponential and Poisson distributions, respectively as recommended in the literature [34]. These PDFs are commonly used for emulating communication systems, like network communications, once packet interarrival times (at the receiving nodes) are often random, independent of each other and exponentially distributed. This can be seen in previous works which details our simulations [29]. To evaluate the performance of the CybCollaboration model running within the Internet it was fundamental to generate some background traffic and observe its impact. To simulate this unpredictable behavior imposed by the Internet cloud, we used a background traffic generator host for transmitting random amount of packets with variables sizes (5-10% of CHVEs data flow) through the network. Simulation results were collected for 20, 30, 40, 80 and 100 users (5 runs) interacting in real-time in order to analyze the way the metrics grows.

C. *Statistical Analysis*

The Students T test was also used for comparing the CybCollaborations performance between multicast and unicast protocols. The data used in this analysis was generated by R software [35] which was configured for simulating network scenarios (with 20-100 users) and thus generating throughput and delay means computed values. Its important to note that this statistical analysis has considered just the applications traffic, as occurs in dedicated networks, not computing background traffics as during network simulation experiment.

In the context of our work, we decided to verify how better the scalability is provided by Multicast than by Unicast protocol, being both implemented in CyberMed and without any outter influences. Our hypothesis is that the Multicast protocol will have less delay mean value from the Unicast UDP/IP protocol. The scenarios for the analysis were a CHVE where each user sends approximately 1.000 messages per second with his/her haptic device. Each message has 500 bytes (Ethernet packet size) plus 129 bytes (CyberMeds haptic packet size) once considered that just the haptic cursor position (without force feedback). Before applying the Students T test for comparing the samples mean values, the nature of our data should be analyzed and the Lilliefors normality test was conducted [36]. The Lilliefors test is a modification of the Kolmogorov-Smirnov test and its an EDF omnibus test for the composite hypothesis of normality.

As result of the test for the sample with unicast data, the normality hypothesis was confirmed (D = 0.1306, p-value = 0.000266) and the same hypothesis was rejected for multicast data (D = 0.0653, p = 0.3766). According to Daniel [37], in cases where data comes from a non-normal probability distribution, some mathematical transformations (i.e. square root) can be useful once they do not affect the underlying relationships among variables. Thus, weve normalized our data applying a square root transformation.

D. Simulation and Statistical Analysis Results

The CyberMeds performance for Concurrent Collaboration scenarios remained stable achieving an average packet delay of 2.70 ms, for the scenario with 100 users collaborating. It means that Multicast protocol had almost 97.7% less impact of bandwidth resources than the result of Unicast protocol which presented 22.5 ms as packet delay. Maximum throughput value achieved was 102.96 Mbps for multicast and 10216.79 Mbps for unicast in the scenarios with 100 users. The generated haptic throughput mean of 1.000 Hz can be seen in Figure 8. For the Tutoring Collaboration scenarios, it was achieved 0.03 ms as highest level for packet delay for 100 users collaborating. That means that the Multicast protocol had approximately 97.9% less impact than the result of Unicast in similar scenario, which had 2.55 ms as packet delay.

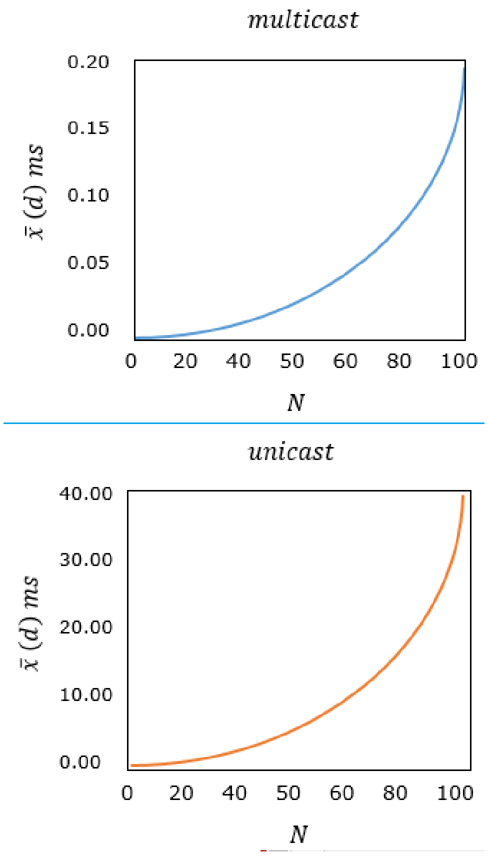


Fig. 7: Unicast and Multicast average delay scores randomly generated by R Statistical Software.

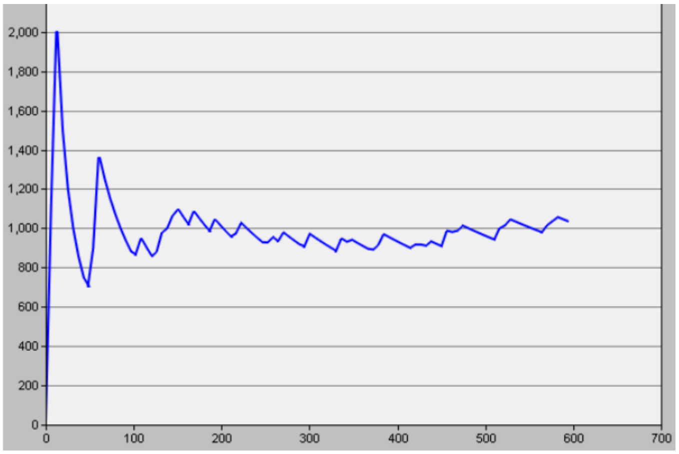


Fig. 8: Simulation of haptic data throughput.

The maximum throughput value achieved was 1.03 Mbps for Multicast and 102.16 Mbps for Unicast in the scenarios with 100 users, as it can be visible on Figure 7. Thus, it was noted that such Tutoring and Concurrent Collaboration scenarios with multicast protocol support have remained within the acceptable limits for end-to-end delay values, providing better scalability and without depreciating the collaboration and the users immersive sensation. As result of the Students T test, the null hypothesis was rejected (p = 2.2e-16) confirming what we expected about the behavior of CyberMeds multicast protocol. Thus, it's stated for these simulated scenarios that the mean of the delay scores of the unicast protocol increases in an exponential manner while multicast grows linearly, providing thus better scalability.

The collaborative VR-based medical simulators, which includes haptic devices, are very sensitive to networking issues once users needs to feel material properties of simulated vital organs, tissues and alike. It is known that VR devices require a high sampling rate, which increases the problem of data throughput. As presented before, the network delay leads to the eventual lack of synchronization and undesirable effects may occur such as the reduction of the intensity of force-feedback felt by users, or the increase of abrupt force feedback. Finally, we've noticed that the new implemented architecture reduced the occurrence of these kind of problems within CyberMed applications. We considered this a important contribution for research in collaborative medical systems, once we've observed how transfer protocols can impact in the experience of final users by delivering haptic data in more effective ways.

VII. Final Considerations and Future Works

It is known that collaborative medical simulators require higher levels of realism, which imposes the use of better technologies and complex systems for enhancing the experience of the users. The research on data transmission within CVEs is important, as it has direct impact on the system performance. Such impairments compromise the realism of a VR medical simulator, especially those devoted to remote surgical setups, which include surgical planning and training of minimally invasive procedures (e.g. video-laparoscopy, endoscopy, etc.), which require greater psycho-motor dexterity from the user [38].

The metrics for quantifying network capabilities in CVEs data communication arent quite well defined and can change according to specific experiments and studies performed. This paper discussed the importance of using better mechanisms to provide better haptic communication for Collaborative Virtual Environments, especially those related to medical simulations. We introduced a collaborative protocol from the *CyberMed* framework, commenting on implementation issues.

In order to obtain better haptic transmissions in the, we chose to implement a peer-to-peer multicast architecture, which lowered the required network throughput. We then presented some applications which were developed as a case study for verifying how the framework behaved with the multicast protocol enabled. It was observed that the requirement to allow a programmer to develop CVEs with few lines of code is still achieved, and the *CybCollaboration* module still remains extensible, allowing one to add further features easily. Then, we presented a comparative performance statistical analysis between the use of multicast and unicast protocol, with the Students T statistical test for comparing the delay samples between unicast and multicast protocols. As result of the Students T test, the null hypothesis was rejected (p = 2.2e-16), which means that delay scores mean of the unicast protocol increases in an exponential manner while multicast grows linearly, providing thus better scalability.

Finally, we introduced a simulation experiment within the OPNET Modeler. It was verified that the traffic of peer-to-peer unicast increases exponentially as the number of users becomes higher, overloading bandwidth as well as imposing higher processing requirements at final hosts. Thus, based on the results obtained and comparing them with the network requirements for haptic transmission found in the literature, we observed that the *CybCollaboration* protocol can satisfactorily support the development of CVE with multicast protocol. As future work, we still expect to implement new mechanisms for achieving even better transmissions requirements, such as packet ordering, as well as to make further evaluation in real network environments.

Acknowledgment

This project is supported by CNPq through the National Institute of Science and Technology Medicine Assisted by Scientific Computing INCT - MACC (CNPq Process Number 465586/2014-7).

References

[1] S. Benford, C. Greenhalgh, T. Rodden, and J. Pycock, “Collaborative virtual environments,” *Communications of the ACM*, vol. 44, no. 7, pp. 79–85, 2001.

[2] J. Broeren, A. Björkdahl, R. Pascher, and M. Rydmark, “Virtual reality and haptics as an assessment device in the postacute phase after stroke,” *CyberPsychology & Behavior*, vol. 5, no. 3, pp. 207–211, 2002.

[3] H. Fuks, A. B. Raposo, M. A. Gerosa, and C. J. P. Lucena, “Do modelo de colaboração 3c à engenharia de groupware,” *Simpósio Brasileiro de Sistemas Multimídia e Web–Webmidia*, pp. 0–8, 2003.

[4] S. Singhal and M. Zyda, *Networked Virtual Environments: Design and Implementation*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1999.

[5] P. Dev and W. L. Heinrichs, “Learning medicine through collaboration and action: collaborative, experiential, networked learning environments,” *Virtual Reality*, vol. 12, no. 4, pp. 215–234, 2008.

[6] F. J. Grajales III, S. Sheps, K. Ho, H. Novak-Lauscher, and G. Eysenbach, “Social media: a review and tutorial of applications in medicine and health care,” *Journal of medical Internet research*, vol. 16, no. 2, p. e13, 2014.

[7] P. V. F. Paiva, L. S. Machado, and T. V. Vieira Batista, “A collaborative and immersive vr simulator for education and assessment of surgical teams,” in *Virtual and Augmented Reality (SVR), 2015 XVII Symposium on*. IEEE, 2015, pp. 176–185.

[8] L. S. Machado and M. K. Zuffo, “Development and evaluation of a simulator of invasive procedures in pediatric bone marrow transplant,” *Medicine Meets Virtual Reality 11: NextMed: Health Horizon*, vol. 94, p. 193, 2003.

[9] Y. Chen and E. S. Liu, “A path-assisted dead reckoning algorithm for distributed virtual environments,” in *2015 IEEE/ACM 19th International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*. IEEE, 2015, pp. 108–111.

[10] K. Zareinia, Y. Maddahi, C. Ng, N. Sepehri, and G. R. Sutherland, “Performance evaluation of haptic hand-controllers in a robot-assisted surgical system,” *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 11, no. 4, pp. 486–501, 2015.

[11] Í. L. Cunha, P. J. Xia, L. S. Machado, T. Restivo, R. M. Moraes, and A. M. Lopes, “Cut and suture support on volumetric models in the cybermed framework,” *Procedia Technology*, vol. 5, pp. 771–776, 2012.

[12] B. R. A. Sales, L. S. Machado, and R. M. Moraes, “Interactive collaboration for virtual reality systems related to medical education and training,” 2011.

[13] P. V. F. Paiva, L. S. Machado, and J. C. Oliveira, “An experimental study on chves performance evaluation,” *Medicine Meets Virtual Reality 19: NextMed*, vol. 173, p. 328, 2012.

[14] ——, “A peer-to-peer multicast architecture for supporting collaborative virtual environments (cves) in medicine,” in *Virtual and Augmented Reality (SVR), 2012 14th Symposium on*. IEEE, 2012, pp. 165–173.

[15] J. M. Taekman, N. Segall, E. Hobbs, and M. Wright, “3diteams–healthcare team training in a virtual environment,” *Anesthesiology*, vol. 107, no. A2145, p. A2145, 2007.

[16] P. Youngblood, P. M. Harter, S. Srivastava, S. Moffett, W. L. Heinrichs, and P. Dev, “Design, development, and evaluation of an online virtual emergency department for training trauma teams,” *Simulation in Healthcare*, vol. 3, no. 3, pp. 146–153, 2008.

[17] A. Boukerche, H. Maamar, and A. Hossain, “A performance evaluation of a hybrid multicast transport protocol for a distributed collaborative virtual simulation of a brain tumor tele-surgery class of applications,” in *Computers and Communications, 2007. ISCC 2007. 12th IEEE Symposium on*. IEEE, 2007, pp. 975–980.

[18] R. C. M. Correia and S. R. M. Pellegrino, “Scalable distributed virtual environments using multicast application level,” in *Computer Supported Cooperative Work in Design, 2005. Proceedings of the Ninth International Conference on*, vol. 1. IEEE, 2005, pp. 140–145.

[19] C. Greenhalgh and S. Benford, “A multicast network architecture for large scale collaborative virtual environments,” in *Multimedia Applications, Services and TechniquesECMAST'97*. Springer, 1997, pp. 113–128.

[20] V. Liberatore, M. C. Çavusoglu, and Q. Cai, “Gipsinet: An open source/open architecture network middleware for surgical simulations,” *Studies in health technology and informatics*, vol. 119, pp. 316–321, 2006.

[21] G. Riva *et al.*, “Applications of virtual environments in medicine,” *Methods of information in medicine*, vol. 42, no. 5, pp. 524–534, 2003.

[22] D. J. Agravante, A. Cherubini, A. Bussy, P. Gergondet, and A. Kheddar, “Collaborative human-humanoid carrying using vision and haptic sensing,” in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 607–612.

[23] J. C. Oliveira and N. D. Georganas, “Velvet: an adaptive hybrid architecture for very large virtual environments,” *Presence: Teleoperators and Virtual Environments*, vol. 12, no. 6, pp. 555–580, 2003.

[24] H. Fisher, “Multicast issues for collaborative virtual environments,” *IEEE Computer Graphics and Applications*, no. 5, pp. 68–75, 2002.

[25] E.-L. Sallnäs, “Effects of communication mode on social presence, virtual presence, and performance in collaborative virtual environments,” *Presence: Teleoper. Virtual Environ.*, vol. 14, no. 4, pp. 434–449, Aug. 2005. [Online]. Available: http://dx.doi.org/10.1162/105474605774785253

[26] L. S. Machado, R. M. Moraes, D. F. Souza, L. C. Souza, and I. Cunha, “A framework for development of virtual reality-based training sim-

ulators," *Studies in Health Technology and Informatics*, vol. 142, pp. 174–176, 2009.

[27] A. D. Santos, R. G. S. Gomes, R. M. Moraes, and L. S. Machado, "A fuzzy logic based assessment tool for vr simulated medical environments," in *Proceedings of Safety, Health and Environment World Congress*, vol. 10, 2014.

[28] A. M. Morais and L. S. Machado, "Cybermedvps: visual programming for development of simulators," *Medicine Meets Virtual Reality 19: NextMed*, vol. 173, pp. 386–388, 2012.

[29] P. V. F. Paiva, L. S. Machado, and J. C. Oliveira, "Simulation and performance evaluation of a distributed haptic virtual environment supported by the cybermed framework," in *Virtual Reality (SVR), 2011 XIII Symposium on*. IEEE, 2011, pp. 188–196.

[30] G. F. Lucio, M. Paredes-Farrera, E. Jammeh, M. Fleury, and M. J. Reed, "Opnet modeler and ns-2: Comparing the accuracy of network simulators for packet-level analysis using a network testbed," *WSEAS Transactions on Computers*, vol. 2, no. 3, pp. 700–707, 2003.

[31] A. Marshall, K. M. Yap, and W. Yu, "Providing qos for networked peers in distributed haptic virtual environments," *Advances in Multimedia*, vol. 2008, 2008.

[32] R. T. Souayed, D. Gaiti, W. Yu, G. Dodds, and A. Marshall, "Experimental study of haptic interaction in distributed virtual environments," in *Proceedings of EuroHaptics*, 2004, pp. 260–266.

[33] K. Yap, A. Marshall, W. Yu, G. Dodds, Q. Gu, and R. Souayed, "Characterising distributed haptic virtual environment network traffic flows," *Network Control and Engineering for QoS, Security and Mobility, IV*, pp. 297–310, 2007.

[34] W. Stallings, "Data and computer communications," *Prentice Hall*, 2005.

[35] M. J. Crawley, *The R book*. John Wiley & Sons, 2012.

[36] H. W. Lilliefors, "On the kolmogorov-smirnov test for normality with mean and variance unknown," *Journal of the American Statistical Association*, vol. 62, no. 318, pp. 399–402, 1967.

[37] W. W. Daniel and W. D. Wayne, "Biostatistics: a foundation for analysis in the health sciences," 1995.

[38] E. Yiannakopoulou, N. Nikiteas, D. Perrea, and C. Tsigris, "Virtual reality simulators and training in laparoscopic surgery," *International Journal of Surgery*, vol. 13, pp. 60–64, 2015.