

Web-based VR Training Simulator for Percutaneous Rhizotomy

Ying Li¹, Ken Brodli¹, Nicholas Phillips²

¹*School of Computer Studies, University of Leeds, Leeds LS2 9JT UK*

²*Department of Neurosurgery, Leeds General Infirmary, UK*

E-mail: {ying,kwb}@scs.leeds.ac.uk

NICKP@ulth.northy.nhs.uk

Abstract. Virtual Reality offers great potential for surgical training - yet is typically limited by the dedicated and expensive equipment required. Web-based VR has the potential to offer a much cheaper alternative, in which simulations of fundamental techniques are downloaded from a server to run within a web browser. The equipment requirement is modest – an Internet-connected PC or small workstation - and the simulation can be accessed worldwide.

In a collaboration between computer scientists and neurosurgeons, we have studied the use of web-based VR to train neurosurgeons in Percutaneous Rhizotomy - a treatment for the intractable facial pain which occurs in trigeminal neuralgia. This involves the insertion of a needle so as to puncture the foramen ovale, and lesion the nerve.

Our simulation uses VRML to provide a 3D visualization environment, but the work immediately exposes a key limitation of VRML for surgical simulation. VRML does not support collision detection between objects - only between viewpoint and object. Thus collision between needle and skull cannot be detected and fed back to the trainee. We have developed a novel solution in which the training simulation has linked views: a normal view, plus a view as seen from the tip of the needle. Collision detection is captured in the needle view, and fed back to the viewer. A happy consequence of this approach has been the chance to aid the trainee with this additional view from needle tip, which helps locate the foramen ovale. The technology to achieve this is Java software communicating with the VRML worlds through the External Authoring Interface (EAI). The training simulator is available on the Web, with accompanying tutorial on its use.

A major advantage of web-based VR is that the techniques generalise to a whole range of surgical simulations. Thus we have been able to use exactly the same approach as described above for neurosurgery, to develop a shoulder arthroscopy simulator - where again collision detection, and the view from the scope, are fundamental.

1. Introduction

Compared to the open intracranial procedure for the surgical treatment of trigeminal neuralgia, in general, all percutaneous techniques have lower incidences of hearing loss, facial palsy, intracranial complications, morbidity, and mortality. Percutaneous Rhizotomy provides the most selective and graded lesion among percutaneous techniques. However, due to the lack of visual cues, this technique is highly dependent on the experience of surgeons. Thus, surgeons require a greater amount of training before it can be adequately performed.

Virtual reality technology is becoming a powerful tool in surgical training as it promises a number of advantages: it allows repeated practice; it enables self-learning of surgical procedures without any time limit, and without risks to patients. Moreover the environment can simulate more than reality: for example, one can provide views inside the human body that are infeasible in practice. This can help students understand better the surgical procedures by allowing a mental visualization of internal structures and their spatial inter-relationship.

Although surgical simulations offer those advantages, they are still not widely used at present. An important reason is that virtual reality remains relatively expensive, and requires special purpose equipment. The success of the World Wide Web (WWW) and its accompanying technologies such as VRML – the Virtual Reality Modelling Language - offers the possibility of performing interactive 3D simulation within the Web environment. This is attractive in terms of cost and accessibility – the simulation will require no more than a VRML-enabled Web browser on a PC – but there will necessarily be limitations on the complexity of the simulation, and in particular what can be achieved through mouse-based interaction.

Web-based virtual reality is an expanding area. A typical Web-based VR application can be decomposed into simulation and presentation processes. The presentation component, handling interaction with the user, necessarily lies on the client-side, but the simulation can be placed either on client- or server-side. The attraction of server-side simulation is that intensive computation (such as physically-based modelling) can be placed on a dedicated server, but then the network link between server and client inhibits interaction. The attraction of client-side simulation is the opportunity for fast interactive response without worrying about network bandwidth, but we are limited by the facilities in VRML. VRML [1] was designed as a simple, general-purpose language for describing interactive 3D worlds, and so, for example, does not support detection of collision between objects (for the good reason that this would impose a major performance penalty).

Real-time interaction is crucial to surgical simulation. The challenge for the computer scientist is to push the limits of Internet technology in order to achieve this in Web-based approaches – either by minimising network traffic in server-side simulations, or increasing the sophistication of what can be achieved in client-side simulation.

Several Web-based surgical simulation systems have been developed. There are examples of both server-side and client-side simulation being deployed. For example, the endovascular training system [2] uses a server-side architecture in order to support physically-based modelling of catheter insertion; for efficiency, geometric models are maintained on both client and server and only incremental changes are transmitted over the network. John and colleagues have created a series of Web-based simulations that are all client-based: ventricular catheterisation [3], pedicle screw insertion [4] and lumbar puncture [5]. In each case, a VRML model of anatomy and instrument is downloaded to the browser and the user is able to manipulate the instruments entirely within the VRML application. Additionally the lumbar puncture application includes simple collision detection.

In this paper we propose a client-based approach to the simulation of the percutaneous rhizotomy procedure. A 3D model of a patient's head is downloaded to the client, and the trainee can carry out a simulation of the procedure using a VRML-enabled browser. Collision detection between the tip of the instrument and the skull is critical to this simulation, and so a novel feature of our work is the extension of VRML, using the Java EAI [6], to achieve fast and accurate collision detection between a point object and other objects in the scene.

2. Purpose

The percutaneous rhizotomy procedure involves the insertion of a needle into the patient's face, and guiding it towards the foramen ovale – which is punctured to allow access to the nerve causing the pain. There is a well-defined procedure for the operation. Three landmarks are placed on the face: the first indicates the point of entry of the needle, and the other two mark the positions of two imaginary planes which intersect in a line through the foramen ovale. The surgeon uses these to aim the needle in the correct direction. In case of a wrong direction, the needle will hit the skull and has to be withdrawn for a further attempt. It is a highly skilled operation requiring extensive training, by observing an experienced neurosurgeon and by closely supervised practice.

The purpose of this study is to provide an alternative exercise for trainees, in which they practise in a virtual environment before tackling the real operation. Our aim was to develop a Web-based application in order to gain the cost and accessibility benefits mentioned above, while maintaining as much realism as possible. A crucial element will be the navigation of the needle, and feedback on whether the foramen ovale has been punctured, or collision with the skull has occurred. We shall aim to take advantage of the virtual environment by providing additional visual cues to help the trainee gain a better mental model of the procedure. Equally there are some elements of the real procedure which we shall not attempt to simulate (such as tissue deformation for example), simply because these details are not crucial to the training.

3. Methods

System architecture

This web-based system is designed using a client-based architecture, where the simulation process is entirely on the client side. The simulator uses the Virtual Reality Modelling Language (VRML) as a modelling tool so that the interactive 3D information can be delivered over the WWW. As shown in Figure 1, the simulation consists of two basic software elements: an interface and a simulation agency. The interface is responsible for the visualization of the interactive 3D objects and communication with the user, whereas the simulation agency is responsible for the state control of the interface, communication between the VRML worlds and mapping the data from one coordinate system into another.

The *interface* is provided by a web browser, where three VRML worlds are embedded into a HTML page. These three VRML worlds are a local viewer, a global viewer and a controller.

The global viewer shows the view from an external viewpoint (the surgeon's eye view). This view represents the objects involved in the surgery: the instruments and the patient's head, and their interactions.

The local viewer shows the view from the position of the tip of the needle in the global viewer. When the needle in the global viewer moves, the local viewer will change its state according to the movement of the needle. This view of course is not available to the surgeon in reality. It is simply to aid the trainee's conceptualisation of the procedure, and help him during training to achieve the correct result. This viewer also plays an important role in the collision detection between the needle and the head in the global viewer, as we shall explain in the next section.

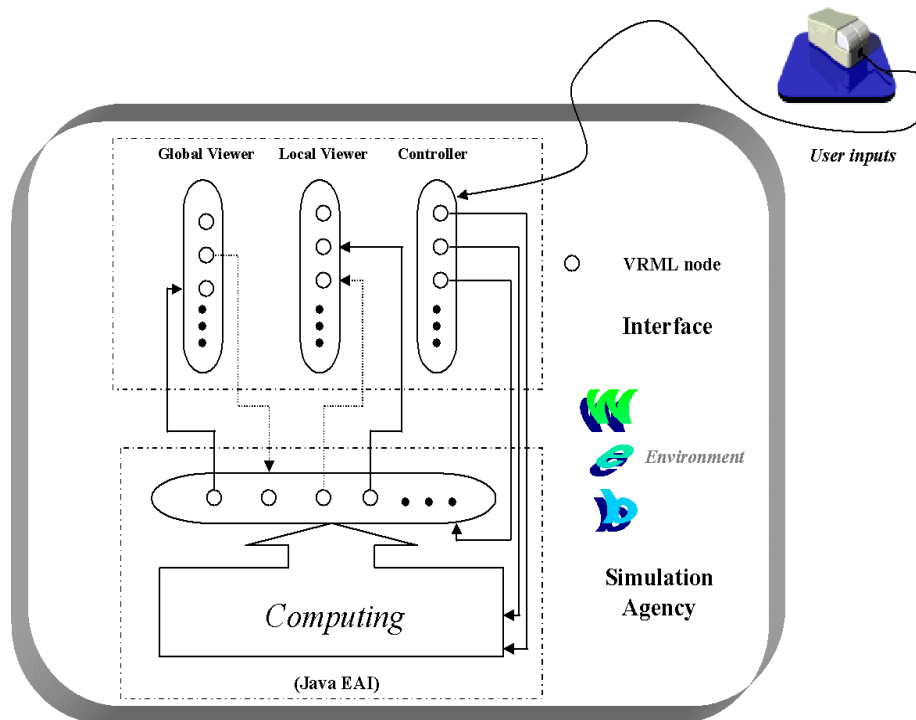


Figure 1. Data flow of the system

The controller displays the widgets, which are used to manipulate the movement of the needle and control the visibility of some features in the global viewer. The user's inputs are mapped into the widgets individually and transformed into the different viewer through the mathematical tools. The communication between these VRML worlds is achieved by the Java External Authoring Interface (EAI).

The *simulation agency* acts as both engine and manager: the engine calculates the movement of the needle in the different co-ordinate systems as a result of the controller interactions; the manager controls the data flow between the different viewers using the Java EAI. Whenever a user triggers a movement from a widget of the controller, it is passed to the simulation agency. The agency then decides where the movement is sent to – a VRML node in the local viewer or the global viewer - and what is the appropriate value to be sent to those viewers. If this movement is in a different co-ordinate system to that in the other viewer, the transformation process is then carried out in the agency. The rotations which are generated from the separate widgets in the controller are accumulated in the agency, and the results passed to the global viewer and the local viewer as appropriate.

Collision detection

Collision detection is a key element of the simulation. The system must recognise, and feedback to the trainee, whether he has successfully punctured the foramen ovale, or whether he has missed the target and hit the skull. Collision detection between polygonal objects is a time-consuming task, involving comparison of each object against all other objects in the scene. For this reason, VRML allows objects to pass undetected through other objects. The only form of collision detection which is supported is collision between viewer and scene (in order to support 'walk-through' applications).

There have been attempts to add collision detection to VRML-based surgical simulations. In El-Khalili's catheter simulation [2], a server-side approach is used, and so

the collision detection can be carried out in an external process on a powerful server. In [5], a client-side approach is taken and VRML augmented with simple collision detection. In that case, the instrument path is known in advance and the object to be hit is unique and relatively simple, and so a prediction of where collision will occur can be made. Here the situation is different: the objects involved in the collision are more complex, and we need to distinguish which object (skull or foramen ovale) we are colliding with. However we have been able to exploit the presence of a needle-eye view in our simulator, and use that for collision detection, in the following way.

In the local viewer, the position of the viewpoint corresponds to the position at the tip of the needle in the global viewer. As we know that VRML supports the collision detection between the viewpoint and the objects in the scene, when the collision happens in the local viewer (between the viewpoint and the objects), the message is fed back to the simulation agency. The agency then passes the message to the global viewer and also signals to the user. The user can decide which action is to be chosen in the meantime. For example, if the needle collides with the skull base the user can check how far it is from the foramen ovale from both local viewer and global viewer. The entry point can then be adjusted in the next practice.

The simulation agency is able to identify collision with the different objects. When it receives the collision message from the local viewer, it gives appropriate signals to the different collisions. In our simulation, for example, the entrance of the needle into the skull base is signalled by a red light. When the user continues inserting the needle, the simulation reverses the motion of the needle so that it appears to bounce back from the skull. On the other hand, the entry of the needle into the foramen ovale is signalled by a green light and a compliment!

Other features

Some additional features are included in the simulator in order to reflect as realistically as possible the real surgical procedure. The trainee begins by marking three anatomical landmarks on the patient's face (see Figure 2). These are then used to construct reference planes which the trainee can display, and compare with a 'correct' set of planes. If a poor set of anatomical landmarks has been chosen, the trainee can start over again. The planes can remain visible to guide the trainee in needle insertion, if wished.

The needle is initially positioned at one of the landmarks. The trainee can use the rotation widgets in turn to orient the needle correctly before insertion. However once insertion has begun, then the system prevents any further rotation – to reflect exactly what happens in reality.

4. Results

Figure 2 shows the Web based training simulation system. The top left is the local viewer, the top right is the controller. The bottom is the global viewer. Although the skull base is a complex structure, the system is still capable of performing accurate and real-time collision detection between the needle and the skull base or the foramen ovale.

This simulation with additional information of how to use the simulation and the description of how the system was developed is available in a collection of Web-based surgical simulations [7].



Figure 2. PRP training simulation system

We have also been able to apply this technique to a shoulder arthroscopy simulator - where again collision detection, and the view from the scope, are fundamental. Arthroscopy consists of inserting a camera probe from one side of the joint, and the tool from the other side, allowing the surgeon to work directly inside the joint. With this simulator, we are able to position the arthroscope by rotating, twisting, or translating. The camera is located at the tip of the needle. One view represents the view from the camera, oriented exactly as the surgeon would see it; the other view gives the surgeon's eye view, and allows direct manipulation of the scope.

5. Conclusions

We have developed the prototype of a Web-based VR training system in the field of neurosurgery, simulating the percutaneous rhizotomy procedure. We use a client-side architecture in which VRML and Java are used to provide both the simulation and presentation. Collision detection between the instrument and patient is critical to this simulation. We provide this by a pair of linked views: one from the eye of the surgeon, one from the eye of the needle. The needle-eye view uses VRML's built-in facility to test for collision between viewpoint and objects, and the collision information is passed to the surgeon's eye view via the Java EAI. The various parts of the procedure – marking anatomical landmarks, orienting the needle and inserting the needle up to point of collision – are all simulated by the application. The simulation is deliberately kept simple in order to run efficiently on modest equipment.

The simulator is available on a public access Web site, and is therefore accessible to anyone with a PC, Internet connection and Web browser with VRML plug-in.

6. References

- [1] Jed Hartman, Josie Wernecke. The VRML 2.0 handbook. Silicon Graphics, Inc. Addison-Wesley Developers Press. 1996.
- [2] Nuha H. El-Khalili, Surgical Training on the World Wide Web, PhD thesis in Computer Studies, University of Leeds, 1999.
- [3] Nigel W. John, Web-based ventricular catheterisation system,
http://synaptic.mvc.mcc.ac.uk/Ventricular_audio.html
- [4] Nigel W. John, Web-based pedicle screw system,
<http://synaptic.mvc.mcc.ac.uk/PedicleScrew.html>
- [5] Nigel W. John, Web-based lumbar puncture system,
<http://synaptic.mvc.mcc.ac.uk/Lumbar.html>
- [6] External Authoring Interface. EAI Working Group, <http://www.vrml.org/WorkingGroups/vrml-eai/>.
- [7] Web-Based Surgical Simulators and Medical Education Tools,
<http://synaptic.mvc.mcc.ac.uk/home.html>