Creating Cyberworlds: Experiences in Computer Science Education

Mario Gutiérrez, Daniel Thalmann, Frédéric Vexo

Virtual Reality Lab (VRlab)
Swiss Federal Institute of Technology Lausanne (EPFL)
Lausanne Switzerland CH-1015
{Mario.Gutierrez, Daniel.Thalmann, Frederic.Vexo}@epfl.ch

Abstract

This article shows that the creation of cyberworlds (interactive virtual environments) can be an excellent educational tool covering a wide range of computer science and engineering disciplines. We present the experiences gathered during more than one semester of work with computer science students in the framework of semester projects at the Virtual Reality lab of EPFL. Our objective is to share the lessons learnt by both the students and researchers (project advisors). We describe the projects proposed, the methods applied to help the students reach the objectives, the problems faced during the development work and how we managed to solve them.

Keywords: Computer Science Education, Virtual Reality, interactive virtual environments, video games, VR peripheral equipment.

1. Computer Science Education

Computer Science (CS) education is an ever-evolving discipline, just as CS itself. Methods to teach concepts and make students fully understand novel technologies and put them in practice are always challenging tasks. A fair amount of resources can be found in the literature and in the Internet concerning methodologies and strategies to help students assimilate complex concepts such as network programming (e.g. multithreading), data modeling, software engineering, mathematics for computer graphics (e.g. linear algebra), etc. New techniques for computer science courses have been proposed based on methods involving constant variation of teaching styles and the ”surprise factor” (unexpected activities or reactions from the lecturer) [12].

Several learning theories and methods to approach the problem of Computer Science education (CSE) have been proposed. The constructivism is one of the most widely accepted learning theories. It claims that knowledge is constructed by the student in an active way. The construction builds recursively on existing facts, ideas an beliefs. In [3], the author surveys the application of constructivism in the context of CSE. The guidance of the lecturer must take into account previous knowledge, and help the students build their own models and views of the concepts being taught. In this sense, active learning from the students seems to be stimulated through hands-on experiences.

The current trend is having the students ”touch and feel” the new concepts they are supposed to learn. Extensive use of audio-visual simulations during lectures is welcomed, specially in this times when we are all over-exposed to impressive visual effects in films and TV. Moreover, the importance of scientific visualization is rapidly growing and the use of interactive images and animations (computer graphics) is recognized as crucial to understand scientific problems [5]. Using computer graphics during lectures is a good strategy. However, in this article we present our experiences showing that having students create computer graphics (as an essential part of a cyberworld) is even better when it comes to make concepts clearer and introduce new ones.

Concerning the hands-on experience, it is a well-known practice in universities all-over the world to have undergraduate CS students develop research-related projects, or at least have the senior students participate in the preparation/evaluation of CS courses [8]. Other strategies to implement the hands-on approach include the ”Software Factory” [14]: an 8-semester programm simulating a real software organization where CS students put in practice their recently acquired knowledge on software engineering.

Undergraduate CS students have elective courses whose credits are obtained by working on a research-related project. Projects are usually designed to be completed within one semester or one year, depending on the difficulty level. Usually, one-year projects are targeted to senior students and they are mandatory in order to get the engineer diploma.

Projects are usually defined by the research laboratories

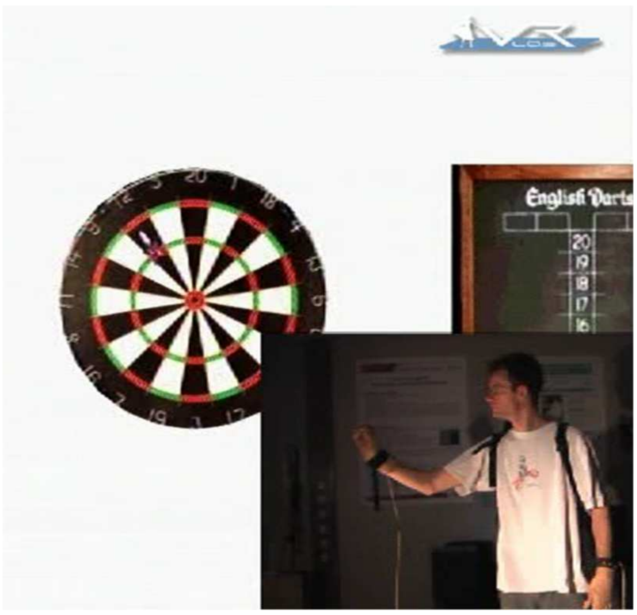

Figure 1. Virtual Darts, using a data-glove and a magnetic tracker.

at the universities. Common practice is to propose projects concerning the development and testing of already defined concepts that must be formally evaluated before applying them in novel research. This way undergraduate students can be introduced to the research world while putting in practice their knowledge in CS.

At VRlab we have noticed this approach has some drawbacks. PhD students are normally in charge of proposing such projects, but sometimes they are not especially motivated to do so, since it usually means a lot of time on supervision. To motivate them their research directors advise them to define projects in such a way that the development work done by the student can be re-used in their own research. As a consequence, undergraduate students end implementing algorithms that look rather abstract and don't represent an interesting activity for them. It is not evident for an undergrad student to understand how such algorithms can be applied in the "real world". Not that the algorithms aren't useful, but the concepts and vision behind them are usually far away from an undergraduate's objectives and motivations.

The result is that less students get interested in doing such projects. Thus, undergraduate lose the opportunity of putting in practice and further develop their knowledge and skills in a research environment. On the other hand, researchers are left without the precious help of developer experts: Usually undergraduate students have fresher knowledge on development techniques and languages, although they still need to mature in other skills, e.g. coding organization and documenting habits.

A year ago we noticed this reduction in the amount of student projects being developed at our lab and decided to try a slightly different approach. The objective was to attract more students and have both sides benefit from the experience: researchers would get some development help to produce attractive research demos and students would learn new skills and knowledge. The idea behind our new strategy was to define projects as very concrete and "funny" applications that could easily show VR in action. We proposed several projects which consisted on creating Cyberworlds: interactive virtual environments, in the form of Virtual Reality games.

The results obtained were very encouraging, last semester we had "full-house" at the laboratory and at the end both undergraduate and PhD students were happy with the results. We also learnt several things in the process. For example, we realized that the CS courses were failing on teaching some basic concepts on network programming, linear algebra and software engineering (or at least the concepts weren't retained long enough in the student's knowledge repertoire). But we were happy to see that the development projects we proposed succeeded on reinforcing such "weaknesses". Moreover, students had valuable experiences concerning the development of important professional-life skills such as working in teams to solve complex engineering problems - e.g. implementing a system for noise-reduction and calibration of a magnetic motion capture equipment.

The rest of the paper will detail our approach of using cyberworlds as educational tools. We will describe the projects we proposed and the methodology we followed with the students. The final part discusses the lessons learnt on both sides (researchers and students) and our plans to further develop this CS education strategies.

2. Cyberworlds as educational tools

Several research works have reported that the use of VR as an educational tool can increase student interests, understanding and creative learning. We can cite for example the Virtual Experiment environments for science education which used a web-based VR platform for middle school science education [11]. A research effort on a similar direction is the distributed virtual environment for children presented in [6], a web-based VR system. Other example using a 3D cyberworld platform to give access to knowledge and contribute to education, this time in the field of archeological research can be found in [7].

The works cited before show the benefits of "using" cyberworlds as educational tools. Our approach is different because we don't "use" them to educate. In our case the

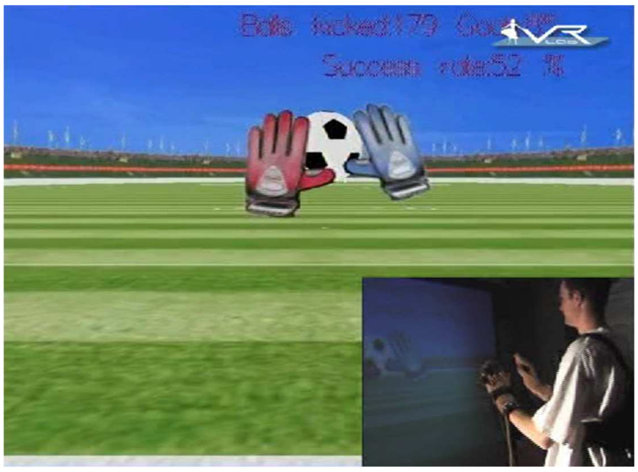

Figure 2. Virtual Goalkeeper, using magnetic trackers on both hands.

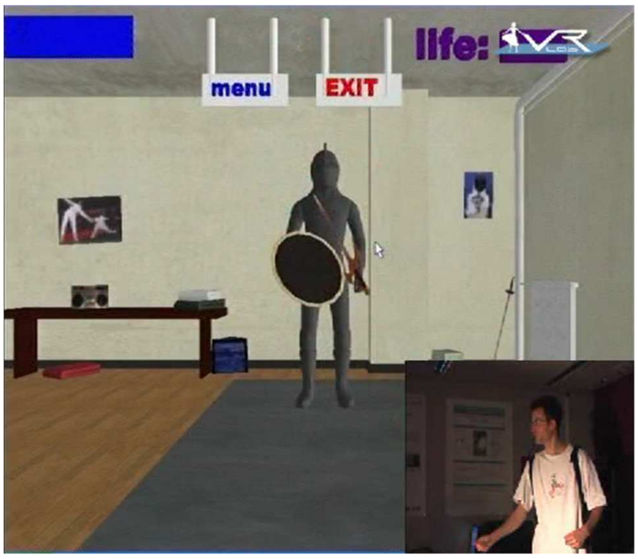

Figure 3. Virtual Fencing, using magnetic trackers.

pedagogical process is based on the creation of the application itself, not on using it as a finished product.

As we mentioned in the introduction, the idea was to apply the hands-on approach, some kind of ”learning by doing” strategy to have CS students put in practice their recently acquired knowledge and skills.

When we asked ourselves the question of how to attract CS students and motivate them to work on research and development projects, we analyzed the kind information and applications that attract them the most. In fact it is easy to realize that we all are exposed to state of the art computer technologies everyday in the form of CGI films and video games. Indeed, it is a well-known fact that the video games industry leads and pushes the advances in computer graphics technology, for both software and hardware. These applications are very appealing. It is no secret that the dream of many CS students is to create a video game.

In fact, video games are complex systems that integrate knowledge from many CS fields: computer graphics, network programming, multi-threading, algorithms optimization, software engineering, etc. The strategy of teaching through video games has been successfully applied before [9] and continues to be useful.

To make the projects more interesting, we didn’t limit ourselves to ”conventional” video games controlled with mouse/keyboard or a joystick.

Since we wanted the development work to be useful for the laboratory, we designed the projects in such a way that they could be used as demos for public dissemination showing VR technologies in action.

Thus, the projects we defined were targeted at exploiting one or more VR peripherals: data gloves, magnetic motion tracking systems, HMD. This added interest to the project, since it is not so hard to find resources in the internet about programming video games. But having access to VR peripherals is not that easy for a student.

Moreover, from previous experience, we realized that even if its difficult to compete with state of the art graphics used in video games. A VR application can be appealing if it uses an uncommon interface. We observed this phenomenon in the VR game we developed last year as part of the celebration of the 150th anniversary of EPFL: ”The Enigma of the Sphinx” [1].

The most attractive feature of the ”Sphinx” demo was the multimodal interface we developed based on the ”magic wand” [4]-a magnetic tracker pointer used in combination with speech recognition. This demonstration had some influence on attracting students to our lab, some of them were intrigued about how to create such VR applications with multimodal interfaces. Indeed, one of the projects we proposed consisted on improving the ”magic wand” interface.

On the research side, we wanted to make tests on some equipment and do some experimentation concerning the noise reduction on the magnetic trackers. We arranged the projects so that students were motivated to work on the problem of calibration and noise-reduction of the magnetic motion tracking system. In order to have a playable application, the tracking data had to be corrected.

In the next section we will describe the projects we proposed and the methodology used to follow the students progress and give them feedback and help when required.

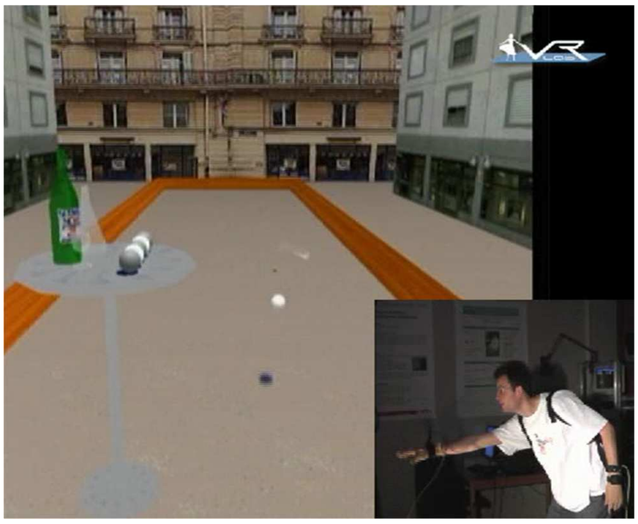

Figure 4. Virtual Petanque, using magnetic trackers on both hands and gesture recognition.

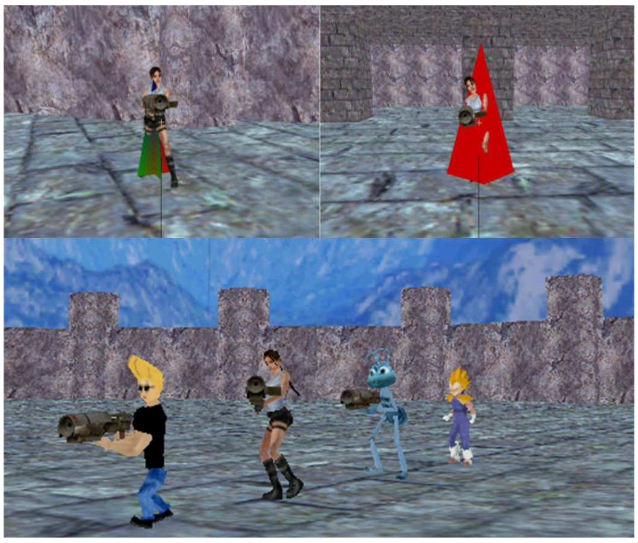

Figure 5. Micro Quake, multiuser network application.

3. Methodology for creating cyberworlds

In this section we will describe the research-development projects we have defined applying the approach of creating a cyberworld using VR peripherals. First of all we will describe the general methodology used to assign the projects, follow the progress and evaluate the results.

The results we are reporting in this article correspond to semester projects carried-on by senior CS students from September 2003 to February 2004. The laboratory presents in a web page the project proposals available per semester. A student can be assigned to a project only after an interview with the research assistant (PhD student) and/or senior researcher responsible of the such project. During the interview the project responsible finds out about the motivations and background knowledge of the student.

Projects are open to students from both the Computer and Communication Sciences Schools. The formers have followed more software engineering oriented courses while the others are more oriented to electronics and telecommunication and sometimes have less strong programming skills. The objective of the interview is to verify whether the student has followed the courses required to develop the project and has authentic motivation to work on it. Sometimes we observed the student was not really eager to work on a given project but was only interested on getting the credits since there were no other project options available for him/her (the projects that interested him/her were already taken). In such cases we strongly recommend to wait for next semester since we know an unmotivated student won't get any benefit from the project and neither the laboratory.

Students have access to the laboratory installations (eight hours a day during working days) where they are allowed to work on PC workstations with OpenGL hardware acceleration and they usually develop using C++ under a Windows environment. Project progress is monitored in a rather informal way, except for the mid-semester and final presentation which are formal and taken into account to assign the final evaluation. In general, the project responsible keeps in touch with the student on a daily or weekly basis since both of them see each other at the lab very frequently. Each student adopted his/her own schedule according to the rest of the courses.

We tried to let the students work on their own and let them come to the project responsible only when there were some technical/scientific problems they were unable to solve. This way we intended to motivate the cooperation between students, the team work and the autonomy. In the next section we will give more details on this subject.

Developing Cyberworlds requires a great amount of coding, several modules must be put in place before an interactive application can run. At VRlab we have developed a software development framework aimed at integrating the different libraries and routines developed by our researchers [10]. However, from previous experiences we realized that confronting undergraduate students to such a system was not a good idea. In fact the framework is a complex system made out of hundreds of classes and with a great variety of configurations and options. Undergraduate students faced to it delivered poor results due to the time invested first to understand and feel at ease with the system. Not enough time was left for the student to work and improve their own work.

To solve this problem we gave them a much lighter ”development framework”, an OpenGL-based set of classes with the basic functions to load some 3D model formats (3DS, MD3, etc.). This way they could start immediately to load or create some 3D models and concentrate on the actual interaction and simulation algorithms required.

The following is a short description of the projects we proposed following the approach of creating cyberworlds. Some projects were done in teams of two students. The final reports and presentation slides are on line at VRlab’s project repository [15](most of them are in french, though).

- **Virtual Darts:** using a data-glove and a magnetic tracker on the wrist. Implement the game of darts, including the a realistic simulation of the darts trajectory, as a function of the speed, position and orientation of the hand when throwing it. See figure 1.
- **Virtual Goal Keeper:** using a pair of magnetic sensors attached to the wrists, simulate a goalkeeper who has to stop penalty shots generated automatically and following parabolic trajectories. See figure 2.
- **Virtual Fencing:** using a single magnetic tracker attached to the wrist, implement a game that lets the user handle a sword and touch some moving objects in the virtual world. See figure 3.
- **Virtual Petanque:** with the aid of a single magnetic tracker and a data-glove, create a virtual world where to play petanque (a french game mainly played in several European countries [13]), with a realistic simulation of the rigid-body dynamics of the balls.
See figure 4.
- **Virtual Micro Quake:** implement a light version of the popular shoot’em all game allowing two or more users to play together through the network. See figure 5.
- **Improving the Magic Wand Interface:** re-implementing from scratch the pointing, and posture recognition algorithms of the magic wand interface [4], in particular using the corrected position information, to improve the pointing. See figure 6.

As can be seen from the descriptions above, all of the projects (with the exception of the ”Micro Quake”, which put more emphasis of the multiuser and networking capabilities) required the use of one or more VR peripherals, all of them required a sensor to track the position and orientation of one or both hands in real-time. This is where we introduced the need for coordinated team work and some more research-related activities. The tracking was to be done with a magnetic sensor (Ascension Flock of Birds (FOB) [2]). The games where intended to be used having the user standing in fron of a large projection screen as shown in figure 6, a semi-immersive virtual environment.

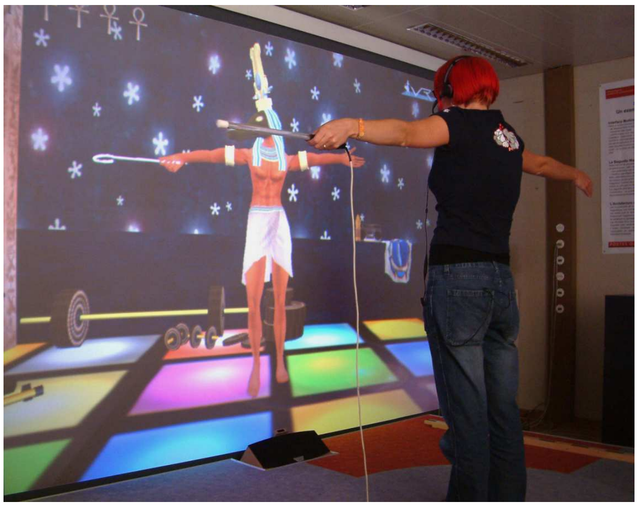
Figure 6. The magic wand with improved position tracking.

The place where the projection screen is located at the lab has several sources of magnetic interference (metallic ceiling, several computers and metallic furniture around). Thus the data coming from the magnetic trackers (position and orientation of each sensor) was noisy, in particular the position information. This problem was reported in the article that presented the ”magic wand” [4]. By that time we avoided using the position information since we had no time to analyze the problem and find a solution for noise reduction and calibration. Then, we gave the whole group of students (with the exception of the ”Micro Quake” team, who didn’t use tracking) the additional task of implementing a solution to the noisy tracking signals. They all had to be motivated to solve the problem, since they required to use both position and orientation information in their applications. If they wanted their project to work, they had to participate in solving the problem shared by all of them.

In the next section we describe our observations and lessons learnt during the six months of project development.

4. Lessons learnt

The first lesson we learnt was that having students work in team should be closely supervised, otherwise they will most likely end letting one or two members of the team do the work for the rest. This is what happened concerning the team task we gave them (noise reduction and calibration of the magnetic tracking system).

At the beginning we had some short plenary meetings where we emphasized the fact that the tracking problem was a common obstacle that all of them should overcome. We clearly stated our intention was to have them all do part of the work. We gave them some clues about the possible alternatives to approach the problem (taking measures at regular intervals and at different times of the day to analyze the noise patterns). And let them free to name a team leader and make a division of work. They started by organizing the measurements, all of them were responsible of measuring at a different day. They used a wooden structure (which had been built some years ago by a PhD student) that allowed for setting a magnetic tracker at preset fixed locations covering the whole working volume.

For the mid-semester presentation, each student was expected to give a report on his/her project status. At that time we realized the tracking problem had been almost solved by only one team. They used multiple polynomial regression and considered the noise as a function of the spatial position of the tracker. They were thus able to correct static errors and neglected dynamic ones.

The group had just let two of their colleagues take the initiative and do the work. The team that solved the problem (two students) got a higher evaluation. The rest of the students were asked to propose a solution, even if it was not implemented, in their final report [15].

On the other hand, as the projects advanced and more problems appeared, the students finally learnt to work in team, helping each others, sharing what they had learnt, exchanging ”tips and tricks” to compile a library, solve a common error, etc. In the end, letting the students work on their own most of the time revealed to be the best way to teach basic skills for the professional life: autonomy, initiative, communicating with unknown people (such as other PhD and undergraduate students at the lab), etc.

Students had several missing skills and knowledge. Concepts of Computer Graphics such as rotation representations (e.g. quaternions) were not understood by most of them when they started their projects. Some of them spent several weeks trying to understand what quaternions were all about and how they worked. They were forced to use them since the orientation information sent by the magnetic sensors used this representation. Some of them came to their project advisor and asked about those strange mathematical entities. After several explanations, and most of all, incommensurable trial and errors in their code, they managed to use the tracking data in their applications. This showed us the concepts that were supposed to be clear during the theoretical courses were not understood in reality. The positive side was that they not only learnt about quaternions, but also learnt to communicate and help each other.

Other minor, but not less worth noting problems were networking, software integration issues. In particular, the team assigned to the ”Micro Quake” had to implement a classical star topology to allow two or more PC clients to share the virtual environment and play together, just as in the real games. They were at first asked to implement a more robust topology, such as a distributed star, but din’t have time to implement it. They spent too much time debugging the star topology application.

The rest of the students faced networking programming as well since they had to retrieve the tracking data from the FOB server, but their task was easier since they had access to a small application that did the work for them. Still they had to implement a socket communication between their application and the small service applet.

They also learnt to work in a structured way, for some of them it was the first time they programmed in C++. The majority had followed a java-based curriculum and had only notions of C, but not C++. We asked them to provide the final version of their application in ”release mode”, having the compiler (MS VisualC) make a more strict validation of memory leaks and so on. Most of them found their applications were not compiling anymore and learnt a good deal of information about the compiler settings, library management and classical problems as memory allocation errors.

Some of them took the initiative and searched in the net for libraries and additional 3D models (they were provided with basic scenarios in and a few 3D characters) and libraries to play sound, etc. They were faced with multithreading programming. Some of them delivered very efficient smooth running applications, while some others presented games with less ”playability” due to an inefficient or absent thread management (e.g. not putting the sound playing routines in a separated thread caused the game to stop from time to time when playing sounds).

Having a concrete project with clear objectives and evaluation criteria: ”either we can play the video game or not”, was motivating for the students. They knew whether their applications filled in the minimum requirements or not. All of the projects were finished in an acceptable way. Some of them were more efficient and implemented extra features that were not in the original specification.

For example, the ”virtual fencing” had three different modalities: playing with the suit of armor as shown in figure 3, blowing up some spheres, or kind of a shot’em all game using different characters that had to be killed with a spear. They all used the same principle: collision detection of a vector whose orientation was controlled by the wrist (simulating a sword, spear).

A particularly interesting achievement of some teams was the implementation of elemental gesture recognition techniques and physics simulation. This was true specially in the case of the ”Virtual Darts” and the ”Virtual Petanque” projects. In which different hand gestures had to be recognized (grasping, throwing, holding, releasing) and used to trigger rigid-body dynamic systems. Other projects such as the ”Virtual Goalkeeper” also required the implementation of basic parabolic motion physics. We didn’t advised the students to use some freely available dynamics engines such as ODE (Open Dynamics Engine: http://ode.org/). And were happy to see they went to search their ”old physics books” and studied the equations and implemented them with satisfactory results.

Finally, everybody was happy to use VR technology and peripherals and learn some of the ”secrets of game development”. They all expressed in their final reports and also in informal discussions their satisfaction for having learnt many things and partially unveiled the mysteries of Virtual Reality and video games.

5. Conclusions

Creating Cyberworlds is a multidisciplinary activity which is accessible with nowadays computer equipment (VR peripherals excepted). This kind of projects are motivating and serve to evaluate and reinforce many skills at the same time, related not only to CS but to many other areas, even in the social planes: teamwork, cooperation, tolerance, etc.

These projects pushed the students to put in practice concepts they would hardly apply otherwise in a single system: networking, multi-threading, software engineering, linear algebra, physics simulation, etc. It also helps to better appreciate and understand current and future technologies: games are not a mystery anymore, they are neither easy to develop.

When the students left the lab at the end of February this year, we are sure they had a different vision of the potential of VR technologies and computer graphics. They had also gained self-confidence (they felt rather satisfied for having created a video game on their own) and reinforced or acquired new technical knowledge.

We are willing to continue our project proposals following the approach of creating cyberworlds. They demonstrated to be an excellent way to integrate virtually all CS and social skills required in the professional context.

The demos have been shown to several visitors from other universities and from the industry, they all have enjoyed playing a little bit with one or more of the video games. This has been of great satisfaction for the students. Many of them have accepted to come back to the lab to proudly show their project to some visiting professor or industrial partner of the lab. Some of the students continued to develop projects at VRlab, now that they have reinforced their CS skills they feel ready to approach more abstract research topics.

6. Acknowledgments

The authors wish to thank Patrick Lemoine for the constant technical support he gave to the students and for providing the ”development framework”. We also thank the students who developed the projects: Alex Mathey (Virtual Darts), Leila Nadia Bekkouche (Virtual Goal Keeper), Sebastien Dupre & Frederic Jaccard (Virtual Fencing), Pascal Cominoli & Yves Pessina (Micro Quake), Gabriel Chevrier & Romain Sauvain (Virtual Petanque), Barbara Yersin (Improving the Magic Wand Interface).

References

[1] T. Abaci, R. de Bondeli, J. Cger, M. Clavien, F. Erol, M. Gutirrez, S. Noverraz, O. Renault, F. Vexo, and D. Thalmann. The enigma of the sphinx. In *Proceedings of the 2003 International Conference on Cyberworlds*, pages 106–113. IEEE Computer Society, 2003.

[2] Ascension Technology Corporation. Flock of birds. http://www.ascension-tech.com/products/flockofbirds.php.

[3] M. Ben-Ari. Constructivism in computer science education. In *Proceedings of the twenty-ninth SIGCSE technical symposium on Computer science education*, pages 257–261. ACM Press, 1998.

[4] J. Ciger, M. Gutierrez, F. Vexo, and D. Thalmann. The magic wand. In *Proceedings of Spring Conference on Computer Graphics 2003*, pages 132–138, Budmerice, Slovak Republic, 2003.

[5] S. Cunningham and A. B. Shiflet. Computer graphics in undergraduate computational science education. In *Proceedings of the 34th SIGCSE technical symposium on Computer science education*, pages 372–375. ACM Press, 2003.

[6] J.-P. Gerval, D.-M. Popovici, and J. Tisseau. Educative distributed virtual environments for children. In *Proceedings of the 2003 International Conference on Cyberworlds*, page 382. IEEE Computer Society, 2003.

[7] D. Green, J. Cosmas, R. Degeest, and M. Waelkens. A distributed universal 3d cyberworld for archaeological research and education. In *Proceedings of the 2003 International Conference on Cyberworlds*, page 458. IEEE Computer Society, 2003.

[8] T. Greening and J. Kay. Undergraduate research experience in computer science education. In *Proceedings of the 7th annual conference on Innovation and technology in computer science education*, pages 151–155. ACM Press, 2002.

[9] T. Pavlidis. Teaching graphics through video games. *SIGGRAPH Comput. Graph.*, 31(3):56–57, 1997.

[10] M. Ponder, G. Papagiannakis, T. Molet, N. Magnenat-Thalmann, and D. Thalmann. Vhd++ development framework: Towards extendible, component based vr/ar simulation engine featuring advanced virtual character technologies. In *Computer Graphics International (CGI) 2003, to appear*, 2003.

[11] Y.-S. Shin. Virtual experiment environments design for science education. In *Proceedings of the 2003 International*

Conference on Cyberworlds, pages 388–395. IEEE Computer Society, 2003.

[12] S. Stamm. Mixed nuts: Atypical classroom techniques for computer science courses. *Crossroads*, 10(4):6–13, 2004.

[13] The World wide Petanque and boule community. http://www.petanque.org/.

[14] J. D. Tvedt, R. Tesoriero, and K. A. Gary. The software factory: combining undergraduate computer science and software engineering education. In *Proceedings of the 23rd international conference on Software engineering*, pages 633–642. IEEE Computer Society, 2001.

[15] Virtual Reality Laboratory (VRlab - EPFL). Student Projects Repository
http://vrlab.epfl.ch/public/STUDENTS_PROJECTS/.