Teaching 3D Computer Game Programming

Maic Masuch and Bert Freudenberg

Institut für Simulation und Graphik
Otto-von-Guericke-Universität Magdeburg, Germany
{maic|bert}@isg.cs.uni-magdeburg.de

Computer games represent a vastly expanding field of application of computer science. Surprisingly, there are only very few degree programs at universities specialising in this area. This article summarizes the experiences teaching 3D game programming for computer scientist at the university of Magdeburg.

1 Academic Game Design

While ten years ago a single game designer simply needed passion, motivation, programming knowledge and a lot of time to develop his own computer game, the production process for modern software title has evolved to a complex software engineering project, developed by a team of several dozen specialists funded by a multi-million dollar investment. In fact, most founders of gaming companies are self made men, as well as most of the staff of a game company. Hardly any artist, programmer or designer does have a specific background in game design. Despite the tremendous growth of the game development industry, there are only very few education facilities offering a curriculum for game developers.

This odd situation, however, is beginning to change. For some time, the number of art schools and multimedia colleges offering game design courses increased. These schools, however, concentrate more on end-user skills. In very recent years, first universities have drawn their attention to game development and quite a number of researchers have focused on game related technologies. This ongoing development can also be found in Germany. There are a few isolated lectures at other universities, but, as far as we know, we are the first university offering a number of lectures on game related topics. In doing so, we concentrate not only on the programming aspects of game development (although we reside at the faculty of computer science), in the first lecture we integrate reflection about computer games with very promising results.

Nevertheless, there is a common hesitation to deal seriously with a subject generally believed to be fun and of trivial content. This cliché is not only shared by colleagues, but even prospective students asked: “Hey, do we get an assignment for playing Quake?”. How can something that is designed solely for recreational purposes be taken seriously for academic research and teaching (funny enough, people from the film industry are never asked this question)? This attitude is especially true for German universities.

So we were confronted with the question, whether computer games are suitable for a curriculum. After three years of enormous success, we are convinced that computer games combine in a very unique way different aspects of computer science: Computer graphics, AI, simulation, user interface design are core components of every game. And more: network techniques, multimedia, databases and many other disciplines take part in game development.

2 Course Notes

At the faculty of computer science of the University of Magdeburg we have given a number of lectures on game-related topics [1]. The two main courses are "Computer games: technology and reflection" (CG I) and "Computer games: algorithms and tools" (CG II), aimed at graduate students of the department of computer science. The topic "Computer Games" developed from a single lecture, over a number of years, resulting in four different courses. The two main courses CG I and CG II are held for the third time. An average of 20 students attended each course until the end, whereas the enrollment had to be limited to 30 participants (the standards have been very high).

2.1 Goals

Our primary goals of the courses have been to impart students a deeper understanding of

- how games work, how they are structured and programmed (algorithms, software, hardware etc.)
- the social impacts of games (psychology of playing, game theory, violence etc.)
- artistic aspects of game design (concept art, game play, interactive story telling etc.)

Furthermore, a significant part of the advanced course CG II is a practical experience in game programming and software engineering. The courses have not been designed to enable students to contribute immediately to games companies, it is one part of the computer science and computational visualistics education[1]. In the medium term it is intended to extend the CG courses to a major subject.

2.2 Lectures on Game Programming

The first lecture, as the title indicates, focuses on two different aspects of computer games: Programming techniques are taught (see the first column of Table 1) as well as there are lessons about the social impacts of computer games (shown in the second column of Table 1). This approach is rather new to computer scientists, as they are more or less exclusively oriented to learn about the technical aspects of their profession. Reflection about the consequences of their technology is an aspect of engineering education that usually decreases the student's motivation drastically. In our courses, however, we experienced quite the contrary. The discussions about the "reflective" themes have been unusual lively. Furthermore, we think that studying the motivation, the potential and the consequences of playing computer games allow a deeper understanding of how to make better games.

At the end of the very first semester our students asked for a continuation of the lectures to steep deeper in the technical details of certain aspects of game programming. We therefore created CG II, where we deal with more advanced technique of graphics, AI and simulation on the algorithmic level. A good compilation of these topics can be found in [4]. This advanced course is accompanied by extensive practical tasks in game programming. Here, students are

[1] Computational Visualistics is a 5-year computer science diploma program introduced at the University of Magdeburg, Germany. It focuses on digital image science (generation and processing) and is accompanied by about 20% of interdisciplinary courses dealing with reflections on the topic of image use (philosophy, psychology, education science etc.)

2D computer graphics	history of computer games
3D computer graphics	psychology of playing
3D engines	power-violence-addiction
character animation	gender-specific aspects
artificial intelligence	interactive story telling
user interface design	game esthetics
network techniques	multi user interactions
sound and music	legal issues

Table 1: Topics of the basic course "Computer games: technology and reflection" (CG I).

asked to form a team and program a small game, applying algorithms and design goals taught in the lecture to their own game.

3 Results

All in all we can conclude that the goals of our courses are met. Now, moving to the fourth year of lectures, we have optimized the course and we can summarize our experiences as extremely positive. The practical task (the teamwork to implement a part of an own computer game) has been moved to the second semester, leaving only a few smaller and more introductory programming tasks to the first term.

The integration to combine technological and social topics succeeded, certainly this is due to the special topic computer games, which integrates so many different aspects. In addition, the advanced course CG II gave us the possibility for a broader variety of game topics in CG I.

All in all students expressed at the end of the term that they not only enjoyed the lectures very much, but also felt to have learned a lot in all discussed topics. The motivation of all participants was very high in both, technical and reflectional aspects. Evidently students have transferred their interest for gaming also to the reflection about gaming [2].

4 Lessons Learned

The quality of student's work was high, if not excellent. However, literally in all cases the teams substantially underestimated the amount of work necessary for the creation of their own game. They fell short on their own milestones by an order of a magnitude. One team, inspired by Umberto Eco's "The Name of the Rose", spent two months modeling a whole medieval monastery as background setting for an action-mystery-thriller. Obviously most students were mislead by their own far too high standards, in strive for an implementation comparable to nowadays full-price game titles, unwilling to turn in an "unfinished" assignment. Learning to cope with restricted resources was a tough, yet worthwhile experience for all teams.

Interestingly, the game projects of the teams evolved over the period of the lecture from rather conventional suggestions (yet-another-Half-Life-Mod) to quite original ideas. The screenshot

below shows a scene from "Project Penguin", a virtual multiplayer snowball fight between two teams (see Figure 1).

In the end we come up with some more fundamental considerations about teaching 3D game programming: To what extend should we use meta-design tools like "Game Studio" in order to compensate the enormous complexity of a game title? Obviously, even two terms are insufficient to develop a complete game from the very idea, over concept art, to graphics and AI programming. Even assembling large teams cannot cope with the lack of artistic skills to create genuine concept art or interactive dramas.

A final question, often posed by colleagues, is whether it would be possible to transfer the motivation of the students, induced by the game topic, to disciplines other than computer science as well. This has to be proven in the future.

Figure 1: Project Penguin: A non-violent snowball-first-person-shooter, an ongoing project resulting from one of the first game programming courses. It is developed by students with the Shark-3D engine [3].

References

[1] Maic Masuch. *Lectures On 3D Game Programming*. `http://isgwww.cs.uni-magdeburg.de/games`, April 2002.

[2] Jörg R.J. Schirra. 'Computer game design': How to motivate engineering students to integrate technology with reflection. In Z.J. Pudlowski, editor, *Proceedings of the 4th Annual Conference of the UNESCO International Centre for Engineering Education, Bangkok*, pages 165–169, Melbourne, Australia, 2001.

[3] Spinor GmbH. *Shark 3D Game Engine*. `http://www.shark3d.com`, 2002.

[4] Alan Watt and Fabio Policarpo. *3D Games: Real-Time Rendering and Software Technology*, volume one. Addison-Wesley, 2001.