A Computer Graphics pioneer project on computer games

Rafael Bidarra Rogier van Dalen Joris van Zwieten

Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, NL-2628 CD Delft, The Netherlands
http://graphics.tudelft.nl

R.Bidarra@its.tudelft.nl

Abstract

One of the mainsprings currently pushing the development of Computer Graphics techniques lies in the area of computer games. This motivation strongly backed the inclusion of a new project-based course on computer games within the new Computer Science curriculum, recently introduced at Delft University of Technology. This paper describes the most relevant aspects of this pioneer project. It is concluded that the cautious combination of a rather open project assignment with an effective tutoring assistance significantly can raise the level of both knowledge and teamwork skills achieved by the students.

Keywords

Computer Graphics, computer games, object-oriented programming, project-based education

1. INTRODUCTION

In September 2001 Delft University of Technology introduced a new Computer Science curriculum, within the framework of the so-called Bachelor's-Master's structure. It consists of a 3-year Bachelor programme, followed by a 2-year Master programme. The Bachelor programme offers two specialisation variants: Software Technology (ST) and Media and Knowledge Technology (MKT). The variants mainly differ on some specialisation courses, which for MKT are related to multimedia and man-machine interaction.

A novel element in the new curriculum is the emphasis on project work. Each specialisation course (one per semester) is associated with a 'project course'. Project courses typically consist of an open assignment, which is proposed and streamlined in a Project Reader, and supported by a short instruction. These projects have no 'conventional lectures', though a few guest lecturers are usually invited, bringing in some flavour of the 'real world'.

An advantage of this project-based curriculum is that the acquisition of knowledge is strongly motivated by its immediate application in numerous project tasks. In addition, it encourages students to very actively learn to value and promote the teamwork process, instead of focusing exclusively on the final product.

In each year of the MKT curriculum two such 'course pairs' are provided. In the second year, for example, students follow, in the first semester, a specialisation course on Image Processing together with the corresponding project (MKT Project 3). In the second semester, an introductory course on Computer Graphics is taught in conjunction with the MKT Project 4, where Computer Graphics techniques are applied in the development of a computer game.

The MKT Project 4 was completely set up from the outset during the Winter 2002. In particular, the selection of project objectives required careful attention. Eventually, the following four objectives were chosen:

1. to learn to apply Computer Graphics techniques within the graphics-intensive field of computer games;
2. to learn to combine creativity with the effective use of a number of tools provided within a specific framework;
3. to improve OO-programming skills;
4. to improve teamwork skills as a result of the development of a complex software system.

In practice, the above course goals are fairly integrated. A good deal of the work in this project consists of applying basic Computer Graphics techniques and theory, dealt with during the Computer Graphics course. In this task, students make use of a so-called game framework, which provides them with many powerful Computer Graphics tools. These in turn help students to concentrate on the game design tasks, fostering their creativity. All of this is only possible with a good co-ordination of all activities and a close co-operation within the group.

The MKT Project 4 was offered for the first time during the Spring 2003, and had 47 students enrolled. This paper

describes the most relevant aspects of this pioneer project. First, a number of organisational issues are dealt with (Section 2). Second, requirements and global guidelines are presented which had to be fulfilled by the games that were designed and implemented (Section 3). Third, a number of issues related to the development environment are summarised (Section 4). Finally, some results and conclusions are presented (Section 5).

2. PROJECT ORGANISATION

The project was carried out in groups of five or six students, each group being assisted by a tutor. All groups were expected to make their own planning, according to the requirements set on *what deliverables* should be presented around *what time*. This section presents a number of organisational guidelines, given at start up, aimed at streamlining teamwork during the course of the whole project.

2.1. Role of the tutors

Each group had a tutor at its disposal who watched over the course of the development process. The tutor assessed the group's progress relative to the planning, alerted the group for possible problems, and, in general, assisted the group members with suggestions and answers (and often also with questions) whenever this was necessary. During the group meetings, the tutor promoted the participation of all group members, brought in possible issues deserving specific attention, and streamlined the communication and task distribution within the group.

2.2. Teamwork

To yield a good product, the team process needs to be well organised. This was strongly promoted by holding regular and effective group meetings. In such meetings, the following roles were always assigned: the **chairman** (responsible for the process), the **producer** (responsible for the product), and the **reporter** (responsible for taking minutes of the meetings). These roles were rotated amongst the team members so that everyone performed at least once each of these roles. The team members that did not have a specific role at a given meeting were project participants. Although personal preferences and expertise were considered when assigning tasks in the group, every group member was required to do some programming.

To help evaluating individual progress in teamwork skills, there were two reflection moments scheduled for each team member to complete a peer evaluation form. This evaluation questionnaire was useful mainly for the group itself, as it significantly helped the identification of possible trouble sources in the way the group was functioning, thus making it possible to take timely action to overcome them. Peer evaluation was performed via Internet, using an interactive assessment tool which was developed especially for this purpose.

2.3. Project guidelines and time schedule

The project was scheduled for 4 ECTS credits, which sums up to 112 hours per person. This time was divided over the various phases of the project, according to the global schedule in Table 1 below.

Project phases and deliverables

In the **analysis phase** (week 1) the group decided on *what kind of game* was going to be developed. The resulting deliverable was the **specification**, a high-level description of the game, answering the question of **'what?'**. This basically comes down to:

- determining all game elements which fulfil the requirements (see Section 3);
- choosing the secondary aspects of the game, which together determine what the game is going to consist of.

In the **design phase** (weeks 2 and 3) the group decided on *the way* the above specification was going to be realised. The resulting deliverable was the **design report**, a technical plan of the game implementation, answering the question of **'how?'**. This includes, among other things, a decomposition of the system into modules, and a clear documentation of their interfaces. The design report also motivates all choices made to fulfil the requirements. At the end of the design phase, each team member was required to complete the first peer evaluation form. During the first 3 weeks of the project, two guest lectures were held by experts from the game industry: one on 'Design and graphics for a real-time 3D game on a mobile phone', the other on 'Spatial subdivision & real-time visibility determination for rendering and collision detection'.

Table 1 – Global schedule of the project

Week	Phase	deliverables
1	**analysis**	specification
2	**design**	
3		design report; peer evaluation I
4	**implementation**	
5		
6		
7		
8		peer evaluation II
9		computer game
10	**rounding off**	game manual and technical report
11	plenary presentation and group interviews	

In the **implementation phase** (weeks 4 up to 9), the actual game was finally implemented. The resulting deliverable was, of course, the **computer game**. Towards the end of the implementation phase, each team member was required to complete the second peer evaluation form.

In the **rounding off phase** (weeks 10 and 11), the group was required to produce the **final project documenta-**

tion. This consisted of two separate documents: a game manual and a technical report. The game manual describes the game from the perspective of the player, containing a concise description of the main game elements (including: objective, environment, constraints, obstacles and scoring policy) and a table of all key and mouse input commands available to the player. The technical report deals with the technical challenges which the group faced in designing and implementing the game. This document is mainly centred on the results regarding the requirements described in Section 3 (e.g. approach followed and problems found in addressing them, technical solutions adopted in their implementation, possible limitations of these solutions, etc.). In addition, during this phase, there was a plenary session, in which each group was asked to hold a short presentation where the thread of the group's approach was presented. Finally, at the end of this phase, each group had a short interview with the instructor, after which each team member was given his/her final grade.

2.4. Assessment

The project was evaluated on the basis of two marks as follows:

$$\text{final grade} = \frac{\text{product mark} + \text{process mark}}{2}$$

The **product mark** took into account, among other things, the *quality* of the *game* (e.g. inventiveness, coherence) of the *software* (e.g. structure, modularity, clarity, choice of technical solutions) and of the *project documentation*. This mark was the same for all group members.

The **process mark** reflected the *individual contribution* of each group member in the whole development process (e.g. dedication, initiative, performance, leadership). This mark, taking into account the results of the group's peer evaluations, was determined at the end of the course, after the interview with the whole group. Both marks were assigned by the course instructor in consultation with the tutors.

2.5. Communication

The hours scheduled at the project labs for this project did not amount to 112 hours by far. This means that students often had to work on the project in a variety of places, most likely not always the whole group at the same space-time coordinates. So in order to encourage good communication, a number of Internet tools were used.

The BSCW server (Basic Support for Co-operative Work) was primarily used as a project repository. It contained all documents belonging to a group. It was via the BSCW server that group members exchanged files and posted any changes or announcements to the group.

The CVS server (Concurrent Versions System) had an exchange role similar to the BSCW server, but was exclusively aimed at maintaining and exchanging source code. It contained the latest versions of the source code produced by all group members, and it remembered old versions as well.

Blackboard was the place for information about the project, news posts and so on: the gateway through which the course staff communicated with the students.

And last but not least, e-mail, as well as instant messaging, were extensively used to communicate within the groups and with the tutor.

3. GAME GUIDELINES AND REQUIREMENTS

In this section, the generic guidelines are summarised which were provided in the Project Reader to the students, at start up [MKT03]. These also set the minimum requirements which should be fulfilled by their game. It was upon each group to take these as starting points, work them out, and elaborate a complete specification for their own computer game.

3.1. Generic description of the game

The game takes place in a 3D environment, where the player can walk freely (except, of course, that s/he cannot move through any solid objects). Examples of such an environment are: a maze, a jungle, a busy downtown area, a ruin, a cemetery, a submarine, …

The player has to travel across this environment, which is littered with obstacles, constraints and dangers, in order to achieve a specific objective. En route, s/he may be occasionally assisted by precious guidance hints, power-up capabilities, temporary immunity shields, etc. In any case, a score mechanism will always have to be developed, so that the system rewards or punishes 'every' action of the player.

3.2. Objectives

It was up to the group to choose the objective(s) of the game. For example, this could be any (combination) of the following:

- finding and/or collecting objects (as for example in Pacman);
- reaching a specific location (e.g. the way out in a maze);
- surviving as long as possible;
- minimising the duration/distance of the route (no races, though);
- accomplishing some other specific mission…

3.3. Other requirements

A number of requirements have been set in order to (i) guarantee that the project goals would not be left behind, and (ii) compensate for the rather open character of the project task. These requirements can be divided into four different categories: environment, obstacles, constraints and special effects.

Environmental requirements are requirements regarding the *geometry* of the environment in which the game takes place. **Obstacles** are entities that hinder a player from reaching his/her goals, be it actively or passively. **Constraints** are game rules that affect the player. With **special effects** we mean the application of 'special' Computer Graphics techniques. The groups were asked to in-

clude in their specification document which constraints and special effects they were going to implement. In all those choices, the focus should be on Computer Graphics techniques rather than on modelling or aesthetics.

Environment

There are only **two requirements on the environment**:

- the environment must have a number of alternative routes;
- the environment must contain several *decoration objects*, e.g. trees, pillars, statues, painting, flowers and so on. These objects should not be confused with obstacles, they are not necessarily the same. Of course, all decoration objects have to make some sort of sense within the game.

Obstacles

Obstacles can be classified against two axis: dynamic/static and active/reactive. Dynamic obstacles are able to move through the environment, whereas static obstacles are not. Active obstacles can show sensible behaviour (e.g. chasing you around) whereas reactive obstacles only react on stimuli. **The game should contain at least one obstacle for each of the four possible classifications.** They should play a meaningful role within the game.

Constraints

Constraints limit the user in some way. A minimum **list of constraints that should be implemented** in the game includes:

- collision detection (preventing the player from walking through walls and opponents);
- time constraints (e.g. limited oxygen, rising water level, time bomb, temporary antidote, etc.);
- limited knowledge of the environment (e.g., whether that green bottle contains a good or an evil potion);
- limited viewing capabilities (e.g. fog, colour blindness, darkness, invisible obstacles, etc.).

Special effects

This concerns the use of specific Computer Graphics techniques. Students were required to **implement at least two special effects** in their game. Some examples:

- create a realistic mirror in the environment (e.g. by using a second hidden camera);
- create a ‘night vision’ effect (e.g. by rendering the environment using only the green channel of the RGB spectrum);
- use alternative projections or navigation facilities;
- produce lightning flashes;
- create fountains, explosions, rain or whatever by using particle systems.

4. DEVELOPMENT ENVIRONMENT

The implementation work in this project was performed in C++. As a graphics framework, the OGRE rendering engine was chosen [OGRE03]. This open source framework provides a unified interface to graphics cards through OpenGL or DirectX, and works alike on Windows, Linux and MacOS. Its documentation is pretty thorough and can be consulted online, from the OGRE website, where also a rather active user-forum is run.

OGRE provides, amongst other things, scene graph management that lets one define a scene graph, parts of which may be animated or changed in other ways during rendering so that students do not have to make sure the objects are rendered to the screen. It also releases students of the need to code matrices for transformations, although they do need to understand them well enough to manipulate them. OGRE also gives access to some more advanced techniques that were very useful during the project. Lastly it is fully object-oriented, which helps students to get up to speed quickly.

In their modelling tasks, students were encouraged to create their own models. For this, the shareware polygon modeller MilkShape 3D [MilkShape03] has been used, which was originally developed for the game Half Life. It is widely used to model characters for games like Counter Strike, Quake 3, Max Payne and The Sims. As a consequence there are many online tutorials and example models. Today, MilkShape is able to import many different file formats and, even more important, it provides a MilkShape to OGRE exporter. Finally, although MilkShape might not be all powerful like for instance 3DS Max or Maya, its user interface is much easier to learn.

5. RESULTS AND CONCLUSIONS

All nine participating groups successfully finished the project (see the course website [MKT03] for the description and sources of each of the games produced). The students’ success rate was 96%, i.e. 45 out of 47 had a positive final grade.

The MKT4 project, just like all other courses at our faculty, was evaluated at the end of the term. For this, all students were surveyed on the most relevant aspects of its organization. The results of this evaluation (87% of replies) were by and large rather positive.

The most remarkable outcome is the significant degree of realisation for each of the four project goals mentioned in Section 1. Indeed, most students acknowledged having achieved a deep insight on Computer Graphics fundamentals, regarding both basic techniques and their mathematical foundation. Furthermore, work performed during the initial project phases led them to incorporate many original elements in their games, partly encouraged by the excellent functionality provided by the OGRE framework. In particular, five groups came up with very creative and consistent games, with a high level of playability. The limited knowledge of OO-programming at project start-up and, in particular, of the C++ language, was eventually overcome and most groups delivered very attractive

architecture solutions for their games. Last but not least, all groups recognised that only watching over their teamwork process, and specifically over their tight task assignment policy, made it possible to achieve their successful results.

Other results of the survey underline some more concrete, interesting aspects of the project realisation. These are summarised in Table 1, indicating the percentage of students that subscribed to the respective statements.

We can conclude that the new project-based approach introduced at Delft University of Technology has a very high instructive and motivating potential, and that the cautious combination of a rather open project assignment with an effective tutoring assistance significantly can raise the level of both knowledge and teamwork skills achieved by the students.

Table 1 – Summary of survey results

The course design stimulates me to frequent study	78%
The time I dedicated to this project was (much) more that the nominal (of its study credits)	65%
My dedication was (very) great	93%
We were given an interesting assignment	100%
I experienced the powerful capabilities of teamwork	85%
The analysis phase was instructive	20%
The design phase was instructive	44%
The implementation phase was instructive	94%
I am satisfied with the product delivered	87%

6. REFERENCES

[MilkShape03] MilkShape 3D. chUmbaLum sOft, www.milkshape3d.com, as of July 2003

[MKT03] MKT Project 4 website. Delft University of Technology, Faculty of Electrical Engineering, Mathematics and Computer Science, graphics.tudelft.nl/~mkt4, as of July 2003

[OGRE03] OGRE – Object-oriented Graphics Rendering Engine. ogre.sourceforge.net, as of July 2003