ON THE USE OF COMPUTER GAME ENGINES IN PHYSICS, MATHEMATICS AND COMPUTER SCIENCE EDUCATION

Price C. B.

(United Kingdom, Worcester)

There is a clear and apparent downwards trend of recruitment to courses at University and Secondary School levels within the U.K. within the fields of Physics, Mathematics and Computer Science. This trend is worrying, since it suggests that the backbone of our engineering and manufacturing base is under threat. Manufacturing, commerce and productivity are all based upon a solid understanding of the physical world which traditionally has been provided by natural sciences and mathematics. Meanwhile, our youth seems to be consumed by a desire (which may be justifiable) to engage with "Computer Games"; indeed, the games industry is one of the fastest-growing commercial activities within the U.K. In this paper we propose that the locus of physics, mathematics and computer science education is poised to shift from the "classical" approach of laboratory experimentation and "theoretical" paper-bound exercises to an embodiment within the development of realistic computer games and immersive environments. The construction of virtual worlds is engaging: Work with our undergraduate students at Worcester, and with pupils from local secondary schools has shown that both students and pupils are able to appreciate, and learn those principles of mathematics and physics which are relevant to the construction of virtual worlds according to their own desires to design. Within a "Games Programming" module at the University of Worcester, we have exposed our students to the fundamental concepts of mathematics and physics, but situated within the context of developing a realistic game.

1. INTRODUCTION. The downward trend in recruitment to science and engineering degree courses in the UK is continuing. This is espe-

cially true of Computer Science. There has been a recent shift in recruitment away from Natural Sciences to Psychology, but the latest shift seems to be into "Media Studies". While this may reflect, appropriately, the shift of the contemporary UK culture, it remains worrying, since a generation ignorant of mathematics, science and engineering, may well prove to be critically harmful to the UK manufacturing base and ultimately our economy. At the University of Worcester, we have amassed great experience in pedagogy, and recognize the need to adapt our courses to the needs of incoming students, while maintaining an honest desire to teach the "fundamentals" required for life.

Therefore, to attempt to enthuse our students into a desire to engage intellectually with the concepts and principles of physics, maths, and engineering, we have taken one of *their* beloved media, the medium of "computer games" and have crafted some activities within the three-dimensional world, provided by computer games, to provide then with a valid "learning experience" of Physics and Engineering. In other words we have *embedded* the learning of Physics and Engineering principles within computer games.

This paper is structured as follows. Following this introduction, in section 2 we report on previous uses of "game engines" in education and training. Section 3 details our own project, and describes the materials we have developed. An evaluation of these materials is presented in Section 4, and Section 5 presents a conclusion.

But what is a "game engine", what is a "computer game"? The name "Computer Game" is loaded; it suggests the blood and gore associated with the "first person shooter" games so ubiquitous, and loved. But the same "Game" System Development Kit" (SDK) software which produces these (perhaps) morally-questionable games is also able to support the production of "non-violent" virtual reality experiences, which we shall refer to as "Immersive Environments" (IEs) where the focus of attention is on an interactive experience within a 3D virtual world.

A "Game Engine" - What's this? It's a collection of coded modules that handle 3D rendering, interaction with the players (including network connections) and code which provides a faithful rendition

of the *physics* of the virtual world. Of course, for a game engine to be acceptable, there must be a *high fidelity* of this rendition. Stated simply, the virtual world must be realistic! The structure of a typical game engine is shown in Fig.1

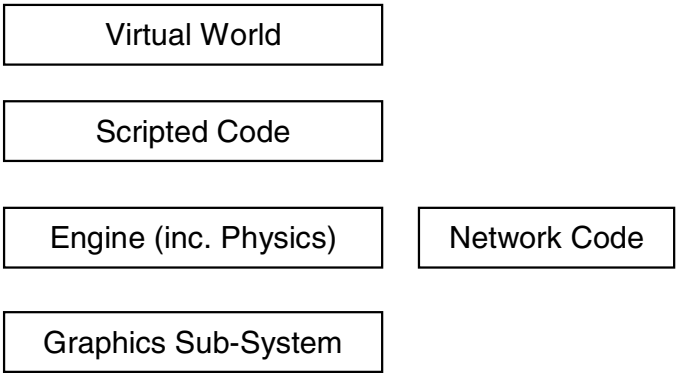

Figure 1. Typical Game Engine structure. At the top level is the "Virtual World" build by the developer, using a pure graphical (no coding) interface. Underneath is a "Scripted Code" level where the developer may write Java-like OOP code to provide intelligence to game objects. The kernel of the engine contains the physics, and the networking subsystems (accessible to the developer) and the graphics sub-system which his highly-optimized proprietary code not visible to the developer.

We have chosen to use the Unreal Tournament game engine for our research. This is a mainstream computer game environment which is sold with an integrated level editor ("UnrealEd") and for which there are various ancillary software tools; "Wotgreal" provides a scripting IDE, "KAT" (Karma Authoring Tool) allows easy specification of the physics of game objects. Unreal Tournament can accept 3D content produced by industry-standard software such as Maya and 3DStudio Max. These factors motivated our choice of engine. But also significant was the Object-Oriented nature of the *scripting code*. This Java-like syntax opens up the possibility of teaching OOP in Java using the Unreal engine (an object of a future research project).

2. EMERGING DEPLOYMENT OF GAME ENGINES. Within the last few years academic and scientific applications build around game engines have started to appear. Recent project have included the study of AI techniques [5], the generation of synthetic characters [9], and the creation of Virtual Reality displays [4]. There have also been several large-scale projects, such as the production of "Urban Search and Rescue" (USAR) simulations [1] and a game-based simulation for emergency response to disasters[6]. The USAR project at the University of Pittsburgh, has developed out of the "Robocup" initiative, and aims to provide a high-fidelity simulation of robots deployed to help in a search and rescue scenario. Researchers have deployed the Unreal game-engine and have constructed detailed levels based on the NIST reference arenas. The simulation is based on actual physical robots which could be deployed in a real situation. Another application is "UnrealTriage", a simulation of a response to an aircraft crash involving multiple casualties. The authors also chose to use Unreal Tournament [6]. There has been a recognition of the potential use of game-engines for such large-scale simulations for some time. This can be found specifically in military simulations used in training, such as "America's Army" and "Full Spectrum Command" [8]. Indeed in 1997, the US National Research Council identified those characteristics provided by game engines which could be of direct use in producing military simulations; computer-generated characters, human modelling, low-cost graphics hardware, networking.[7].

Despite these "large-scale" projects, there has not been any attempt to utilize the in-built physics engine to actually *teach physics*. Nor has there been any work on the utilization of the 2D game world to *teach mathematics*. And neither has there been any work which investigates how to synthesize virtual worlds which may be used in *engineering education*. In this paper, we take a small step to start this ball rolling.

There is also a small, but developing body of work which has attempted to incorporate the paradigm of "Qualitative Physics" into game environments to produce realistic simulations. Qualitative physics attempts to abstract the physical laws of interactions in the real

world into "chunks" of inter-relational dynamic information, which are expressed at a level above the differential equations of dynamic systems. The reader is directed to the work of Forbus[3] for a detailed exposition. Cavazza et al., have deployed the qualitative physics paradigm within Unreal Tournament to generate_useful simulations [2]. This work is very interesting, but we nevertheless question the *need* for a simplification from *quantitative* to *qualitative* physics, given the fact that our game engines already contain a highly efficient *quantitative* physics engine. Our intention in this research was therefore clear; to discover to what extent the in-built *Karma-Physics* engine could be used to generate learning materials for students of physics, maths and engineering.

Table 1. Summary of game levels produced either by the Tutor (T) or students (S).

Topic	Tutor (T) or Student (S)
Gravity and Collisions including Rigid-body dynamics	T
Energy Levels visualized with interacting balls	T
Investigation into Momentum	T
Simple Pendulum	T
Newton's Cradle	S
Diatomic Molecule	T
Simple Harmonic motion	T
Normal Modes of Oscillation	S
Coupled Pendulae	T
Solitons	S
Finite State machine	S
Potential Hill (Harmonic potential)	T
Potential Hill (An-harmonic potential)	T
Electron Gun Potential surface	T

3. OUR PROJECT – USING GAME ENGINES IN EDUCATION. We propose to harness the power and expressivity of a game-SDK to pro-

duce learning and teaching materials, and have made some initial experiments to this goal. There are two threads of action in these experiments; (i) the production of learning materials by the Tutor which are then given to our students, (ii) asking our students to reflect upon a physics or engineering problem, and to develop a game level to investigate this problem. The materials produced are listed in Table 1, and are available from the author.

Each of the above activities was constructed as the union of (i) a functioning Unreal Tournament game level and (ii) supporting worksheet material. No Tutor input was provided. Students were presented with this material and required to work autonomously. After each session, they were required to fill in an evaluation questionnaire. In this section we outline the activities which were constructed and indicate the rationale for the choice of activity.

3.1. Gravity and Collisions. This level consists of a single room containing some spheres and telephone handsets. They are located need the ceiling of the room. When the level starts, the objects fall under gravity and bounce and interact according to the laws of rigid-body dynamics. Students are asked to make several observations, record them and compare them with their everyday experience of a similar situation. They are then free to add other objects to the level, or additional rooms. They can add water volumes and experiment with buoyancy.

3.2. Energy Levels and Colliding Balls. The intention here is to present the concept of energy level, activation energies, etc. There are two rooms connected by a passage. One room is lower and the floors slope down in this direction. The top room starts off containing balls, which bounce around and gravitate to the lower room, although collisions in this lower room result in the expulsion of an occasional ball into the upper room. This visualizes the concept of activation energy. The height difference between the rooms is taken as an experimental parameter

3.3. Conservation of Momentum. This level contains two collision scenarios. First two spheres, and second two plates. The user can assign masses and initial locations to the objects to investigate collisions. The initial heading of the objects can be changed so that glancing as well as head-on collisions can be investigated.

3.4. Simple Pendulum and Newton's Cradle. Two activities which first explore the motion of a simple pendulum where the user can specify the length of the string. The Newton's Cradle activity consists of four pendulae where one starts off with an initial displacement.

3.5. Oscillating Systems. Here there are three activities. A mass-spring simple harmonic oscillator is presented where the mass and spring stiffness may be changed. Then a diatomic molecule is presented, two masses connected by a spring, but the whole molecule is allowed freedom to rotate and bounce in space. The user can interact by firing a gun at the molecule. Finally, normal modes of oscillation are investigated using a system of two masses and three springs.

3.6. Coupled Pendulae and Solitons. The first activity involves experimenting with two pendulae coupled with a spring. Lengths and stiffness may be changed. This system is replicated to produce a chain of coupled pendulae which is a good arena to investigate the concept of solitons

3.7. Finite State Machines. A finite state machine which computes the parity of a series of binary numbers input into the system is presented. This example shows how instructional messages may be incorporated into the game level as "display boards" with the room, (see Fig.2). The user interacts with the machine by the use of "trigger" actors placed in the room. Following this demonstration, the user is asked to design and implement other machines.

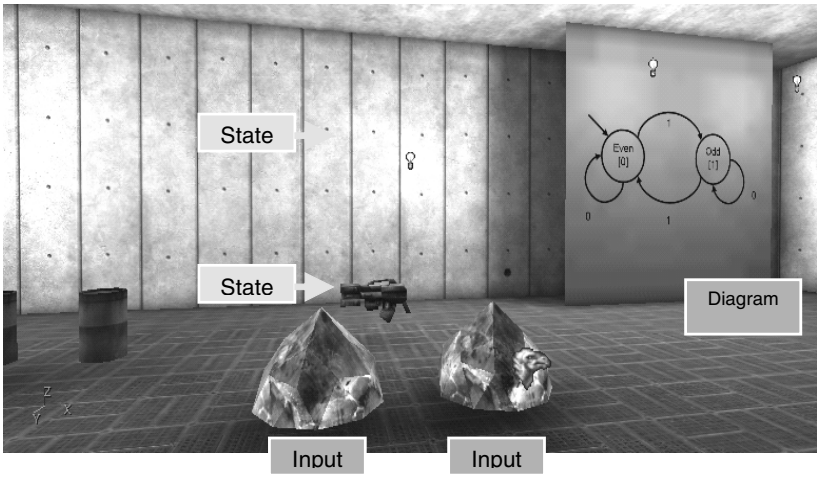

Figure 2. Finite State Machine calculating the parity on inputs onto the two crystals. The gun rises and falls to indicate the parity. A diagram explaining the machine is included top-right.

3.8. Potential Hills. This is one of the most interesting examples of how a game engine or immersive environment can enliven "theoretical" material, the motion of particles in potential fields. The idea is to import a potential surface into a game engine so that the student then can walk around this surface, investigating it. We can also let spheres roll on this surface and observe their motion. This is easily done since in Unreal Tournament, terrain can be imported as a "level map" where the height of a particular surface is specified by the pixel values of a 2D grey-scale image. To test out the possibilities of *literally* exploring potential hills we constructed a harmonic x^2 potential and a non-harmonic potential $(1-e^{-\alpha x^2})/(1-e^{-\alpha})$. Balls rolling on a harmonic potential should display *isochronous* motion, the period of oscillation is independent on the amplitude of the oscillation. Of course balls rolling on this anharmonic potential (see Fig.3) will tend to show a decrease in period of oscillation. So we constructed a level with a harmonic potential and a level with an an-harmonic potential

and placed balls at different amplitudes on the hills. The balls behaved as expected, the physics engine worked. But all players of these levels expressed a surprise that they found the potential hill terrain exceedingly difficult to understand. There were several spheres rolling around and even in the case of the harmonic potential, several students reported that they found it hard to really "see" the isochronous behaviour. This was probably due to phase differences in the motion of the spheres. We could not remove these from the simulations.

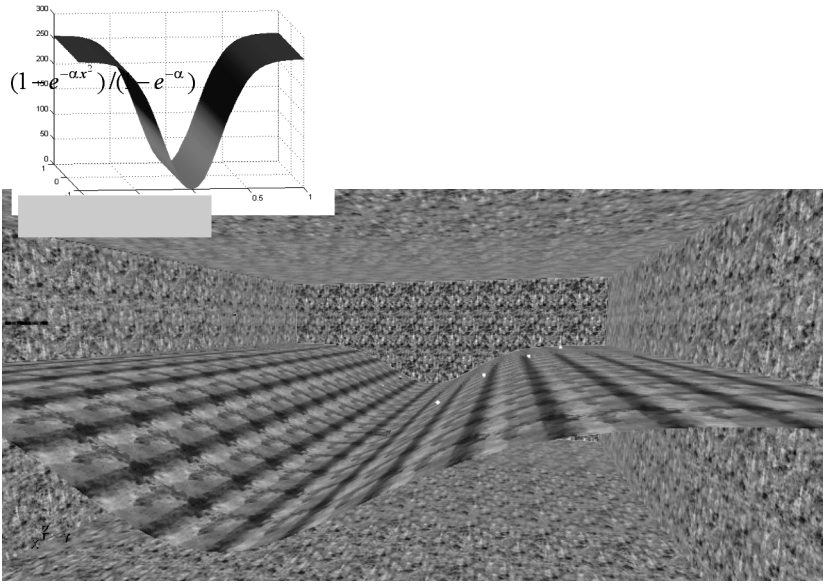

Figure 3. Anharmonic gunction generated in Matlab (inset top-left) and inserted into Unreal Tournament as a curface on which four balls have been placed and where the player is able to walk around and interact with these balls.

The development process flow started from a Matlab definition of a function of two variables, through to the generation of a greyscale image which could be imported into Unreal Tournament.

A Matlab "meshgrid" is declared and a function on this grid is defined. This function is then rescaled (0-255) and finally written to an image (using the Matlab "imwrite()" function.) This image is then imported into Unreal as a "heightmap" in the terrain dialogue. So we move from mathematical function to images to height maps! Exemplar matlab m-files are available from the author.

Table 2. Evaluation of the levels produced. Scores given on a scale of 1 (low) to 5 (high). "Ease of production" is the Tutor's estimate on how easy it was to produce the learning material. "5" indicates that not much time or effort was required. "Fidelity" expresses the Tutor's evaluation of how close the level actually reflected "real" physics ("5" being "High-Fidelity"). "Learning experience" is the students' assessment of how much they learned from the activity.

Topic	Ease of Production	Fidelity	Learning Experience
Gravity and Collisions including Rigid-body dynamics	5	5	5
Energy Levels visualized with interacting balls	5	5	5
Investigation into Momentum	5	4	4
Simple Pendulum	5	5	5
Newton's Cradle	5	1	1
Diatomic Molecule	5	4	3
Simple Harmonic motion	5	4	5
Normal Modes of Oscillation	5	1	1
Coupled Pendulae	5	3	3
Solitons	5	1	1
Finite State machine	5	5	5
Potential Hill (Harmonic potential)	5	4	5
Potential Hill (An-harmonic potential)	5	4	5
Electron Gun Potential surface	5	4	5
Snake	5	5	5

3.9. Electron Gun Potential Field Simulation. The potential hill describing the flow of electrons though the parts of a simple electron gun was constructed with Matlab's "pdetool" exported an converted to a gray-scale

4. EVALUATION OF OUR APPROACH. The evaluation of our experimental project was anecdotal. The Tutors and students who developed these materials were asked two questions: (i) How easy was it to produce the "level" to demonstrate a particular concept, (ii) What was the "fidelity" of the resulting experience, in other words, how well the "level" produced represented and displayed *quantitative* or *believable* physics? Also, the students who were given these "levels" as modules of learning (with supporting texts) were asked to evaluate their "learning experience", simply how much they felt they had learned about a particular concept. The results are shown in Table.2

There is an evident correlation here between the developers' assessment of *fidelity* and the students' *learning experience*. Students were clearly engaging in a process of bringing the "level" experience (the "experimental") into a relationship with the "theory" (provided by associated textual material). Where there was disagreement, the students indicated that they were not learning. They voiced the opinion that there may be a software limitation in those "levels" which did "not appear to work".

The Tutor assessment of *fidelity* is also important. Our *a priori* reading of the Unreal game engine led us to believe that the *Karma Physics* engine would provide high-fidelity representations of the true physics. Clearly this is not always the case. Further investigations are required to understand the reasons why. This may be difficult, due to the inaccessibility of the SDK-*Karma Physics* engine interface, and may be overtaken by the new release of Unreal (2007) which integrates a different physics engine. While the *Karma Physics* documentation suggests that high-fidelity simulation is possible, our experiences with Unreal indicates that this may not always be the case. We suspect the problem lies in the *interface* between UnrealEd and the physics engine. It proved difficult to obtain satisfactory quantitative

results for momentum conservation experiments. While these simulations showed *qualitative* potential, they would not stand up to numerical scrutiny. The potential hill experiments seemed to be confounded by the inability of the physics engine to cope with zero damping. Also, these experiments had difficulty in locating more than a few spheres.

The final evaluation is however positive. Students at our University, who volunteered to test the levels, and children from local schools, all expressed satisfaction and interest at this mode of learning physics. They all indicated a wish to know how to program the game-engine to develop other concepts in physics and engineering. They found the ability to add rooms and objects, as well as simply changing the parameters of objects provided to rewarding.

5. CONCLUSIONS. This paper records our justification for using a games engine to teach Physics Maths and Engineering. We feel that the preliminary results obtained are encouraging, and that it is worthwhile to take this project forward.

REFERENCES

1. S. Carpin, J. Wang, M. Lewis, A. Birk, A. Jacoff., High fidelity tools for rescue robotics: results and perspectives. Robocup (2005): Robot Soccer World Cup IX, Springer, Lecture Notes in Artificial Intelligence (to appear).
2. Cavazza, M., Hartley, S., Lugrin JL., Le Bras, M., Qualitative physics in virtual environments,. Proceedings of the 9 international conference of user interfaces (2004)
3. Forbus, K.D., Using qualitative physics to create articulate educational software. IEEE Expert May.June 1997.
4. Jacobson, J., Hwang, Z., Unreal tournament for immersive interactive theatre. Communications of the ACM 45 (2002)
5. Laird, J., Research in human-level ai using computer games. Communications of the ACM 45 (2003)
6. McGrath, D., Hill, D., UnrealTriage: A Game-Based Simulation for Emergency Response. The Huntsville Simulation Conference (2004)

7. National Research Council Modeling and Simulation: Linking entertainment and defence. National Academy Press (1997).
8. Wray, R., Laird, J.E., Nuxoll,A., Stokes,D., Kerfoot, A., Synthetic adversaries for urban combat training. Proc. 2004 Innovative Applications of Artificial Intelligence, San Jose, CA.
9. Young, M., Riedl, M., Towards an architecture for intelligent control of narrative in interactive virtual worlds. ACM Conference on Intelligent User Interfaces (2003)