Towards Teaching Secondary School Physics in an Immersive 3D Game Environment

Bill Rogers, Dacre Denny, Jonathan Stichbury

The University of Waikato, Hamilton, New Zealand
Bill Rogers: coms0108@cs.waikato.ac.nz

Laboratory exercises are an important part of secondary school physics classes that make an important contribution to student learning. Virtual laboratories have the advantage of allowing experiments that might be too dangerous or too costly in the real world. We present Gary's Lab, an experimental immersive 3D laboratory environment using computer game technology. Our system allows students considerable freedom in constructing apparatus, and running qualitative and quantitative experiments using that apparatus. We argue that the process of constructing experiments in interesting contexts might be expected to help students engage with their lessons, focusing their attention on the apparatus and the methods of measurement used.

***Index Terms*—physics education, computer game, virtual laboratory**

1. INTRODUCTION

Much has been written about using computer game technology for teaching. In the words of Stapleton and Taylor [7] *Computer games seem to captivate the imagination and attention of contemporary teenagers. If only the energy, motivation, fun and exhilaration they enjoy from playing games on their PC, or on consoles ... could be captured in learning physics!* Although the potential has long been appreciated, there has not been the level of success we might have anticipated. The state of the art in 2004 was carefully analyzed by Aitkin [1]. He classifies educational games as 'Data Based', or 'Process Based' and relates the two categories to the instructivist and constructivist educational philosophies respectively. Roughly stated, instructivist educators seek to present information to students, as though their minds are empty and need filling; constructivists take the view that students learn by interacting with their environment and problem solving to extend or change existing knowledge.

Like instructivists, data based games present information and drill students in using it. The game setting, narrative and reward system encourage the player to continue, but often bear little relationship to the educational content. There have been some noteworthy successes in this genre. Math Blasters[3] and Where in the World is Carmen San Diego[2] being two of the best known examples. Math Blasters requires a player to solve problems in arithmetic in order to progress a space alien adventure. Answering geography questions allows a player to find the mysterious hidden Carmen San Diego.

In contrast, Aitken suggests that process based games have the potential to present environments for learners to explore and engage in self motivated learning by solving personally meaningful problems. Such games take the form of simulations in which players have a wide range of choice as to their actions and can observe realistic consequences from those actions. A successful example of a process based game is SimCity[8]. Here, players control allocation of resources in the development of a city. The goal of the game is to have the city grow. The program does not require that this be done in any particular way. It simulates as realistically as possible the various physical and economic processes of a city, and leaves it to the player to work within those constraints. SimCity has been widely used in classrooms. Sadly, success with other simulation games is rare. Aitkin reviews a number of noteworthy failures and concludes that *very few digital games based upon scientific simulations have been made and most of those that have been made are not enjoyable to play*.

What is it about computer games that teenagers so enjoy? Koster[4] argues that the 'fun' is about learning – developing skills and exercising them. For a game to be fun, skills required must be at just the right level; challenging, but at such a level that success is possible. If the challenge is too low, boredom sets in; too great and players will become frustrated. A good background story underpins the goals and objectives of play, letting the player understand what is expected of them. Modern teenagers are also connoisseurs of game technology, appreciative of and having high expectations of the computer graphics and game play. Games that fall short will be quickly rejected.

It is clear that the task of building a successful computer game for teaching is not an easy one. Defining success as the difference between expectation and achievement, the goal of this paper is to work towards more 'successful' educational computer games in two ways. The first is to suggest and demonstrate what we consider to be promising ideas for teaching high school physics. The second is contribute to reducing the expectation people have of such software.

Our project is called Gary's Lab. It provides a laboratory environment for high school physics students, supporting constructive learning. Both qualitative and quantitative experiments have are supported. In a laboratory, students can test the theory they have been given; they learn to use apparatus, and make measurements; they learn about experimental error. Most importantly, abstract concepts become tangible, and they can see that theory will usefully predict experimental outcomes. There are a wide range of choices to be made in setting up laboratory experiments. Sometimes an experiment may be completely pre-built and the student simply asked to run it and observe results. A better option is for the student to assemble or even design their own apparatus. Whilst this may

DOI: 10.5176_2010-2283_1.1.06

seem inefficient in use of the students' time, it actually has value in drawing their attention to aspects of the equipment that they might otherwise overlook. It provides a level of engagement with the task that can only be achieved by spending that time, and being responsible for as much of the process as possible. Building apparatus provides students with the possibility of making mistakes, and also the possibility of trying alternatives. Our goal in the Gary's lab project is to provide a laboratory for teaching which has this 'building' aspect. It is named in recognition of Garry Newman, responsible for a physics sandbox game called Garry's Mod[5], written as a modification for the Source game engine.

In the spirit of reducing expectations, it is firstly not our goal that the software should 'teach' as such. That is the role of the classroom instructor. The purpose of our software is to allow students to test their knowledge of physics in interesting situations. We have not included any assistance with calculation in the software. We expect that students will do calculations themselves, using their knowledge of physics and measurements taken from our simulations. Although software could easily help in this area, having it do so would undermine the development of skills expected in our high school curriculum. Secondly we have no illusions about this being an 'addictive' computer game. We do however hope that we can build Gary's Lab to a high enough technical standard to be acceptable to a modern 'digital native' student. Most importantly we hope to emulate computer games by achieving a good level of 'immersion' for our users. Ideally they will feel that they are participating in a credible environment, enough to help them gain intuition and understanding of the way the physics works – especially of mass, size and speed.

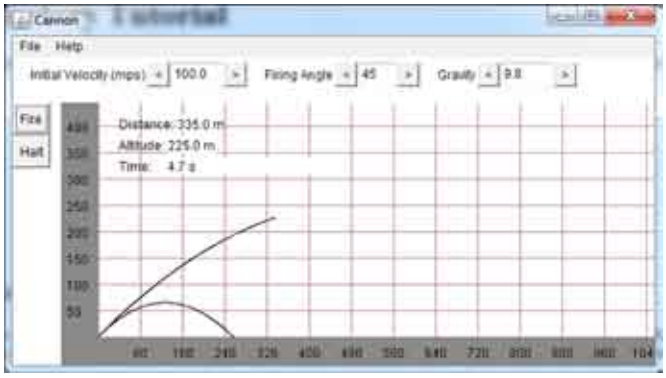

Fig. 1. Abstract Simulation [9].

The idea of using a virtual laboratory is not new. It can permit experiments that are not practical due to scale, safety or cost. From the student's perspective, experiments can be larger and more dramatic. It is also possible to simplify physics in an artificial world. For example, it is possible to have genuinely frictionless surfaces. We have taken great care here though. In our system it is not possible to bypass simulation and directly make sure theory always works perfectly, because we require that experiment variations which a student may choose to do have realistic outcomes.

There is a good deal of existing software that can be used to teach aspects of physics. Typically, software that supports quantitative experiments has largely pre-built setups and tends to be non immersive and abstract. For example, a program that allows experimentation with projectile motion might show equations, allow setting of angle and speed, then show a roughly animated 2D simulation with numbers rolling as it proceeds. Fig. 1 shows an example. There are a number of pieces of software that allow the user to build models, and then watch their behaviour, subject to the laws of 'physics'. Recent examples are "The World of Goo", "Little Big Planet" and of course "Garry's Mod". Each of these is qualitative. It is not possible to use them to trial physics equations.

An underlying assumption of the Gary's Lab project is that a standard game physics engine will simulate physics accurately enough to be used for quantitative experiments. Section II of our paper addresses the issue of physics calculations. Section III describes the sample laboratories we have constructed and some of the experiments they permit. Section IV reflects on the current state of the Gary's Lab project.

2. PHYSICS SIMULATION

The Gary's Lab implementation is built using NVidia's PhysX engine (originally provided by Ageia)[6]. The physics engines used in computer games have not been designed for precise simulation. Rather they are built to operate quickly and satisfy their analogy to the first law of graphics – "If it looks ok, it is ok". In operation, game physics engines may deviate from real physics in a number of ways.

Firstly, calculations are done in finite time steps, as a numerical integration. If the time steps are large, this may generate inaccuracies. Fortunately our system does not require very high precision. Students typically measure sub-metre distances to the nearest millimeter and multi-metre distances to the nearest centimeter; angles to one degree; speed and acceleration to two or at most three significant digits. Most experiments involve features, like friction or air resistance, that are not included in the calculations. As a result it is uncommon to perform an experiment with uncertainty of results better than 1%. We have taken this as our goal – to get physics simulation that is accurate to 1%.

More difficult is the collision system. In a game physics engine, exact collision calculations are expensive. To avoid this cost objects are firstly given a simplified 'physics volume' for collision calculation. This might be an enclosing sphere, cube or maybe a convex hull. Such simplified physics models have proved satisfactory in our experiments. Secondly, the time of collision is not accurate as a result of the finite step integration used for movement. Objects jump from step to step and collision testing is done after each step. In the worst case, if an object is moving very fast, the distance moved in a single time step may be such that collision with a small or thin object may be missed altogether. Usually objects are slightly overlapped (interpenetrating) when a collision is detected. Real objects bend or compress on impact. In a game physics engine, it is possible to simulate deformable objects, but it is more common to treat objects as though they were completely rigid, accepting a small level of interpenetration on collision, and applying a spring force to separate the objects back to the point of contact. This is not physically accurate, but gives a reasonable approximation to compression. With complex shapes, the restoring force may not be applied in quite the right direction, but with simple cubes, spheres and cylinders

the results have proved acceptable in our experiments. So long as movements are not too fast with respect to the sizes of the objects, our results have been accurate enough.

The final problem we have observed with physics calculations occurs when several objects are connected to each other. One example is a rope, simulated as a chain of hinged segments; another is a number of rigid bars connected to form a closed shape. Physics engines compute the forces and motions separately for each rigid component and then iterate to get consistency between components. Just as with interpenetration, restoring forces substitute for unresolved violations of joint constraints. The result can be violent movement or instability. We have tried to choose parameter settings to minimize such effects.

Overall, we have been pleased with the accuracy of simulations. In the same way that real experiments don't give perfect results because of uncontrolled variables and over-simplified physics, our system doesn't give perfect results, just for different reasons. The results, however, are close enough.

3. PROTOTYPE LABORATORIES

Four prototype laboratory environments have been constructed, covering different parts of the kinematic syllabus for the last two years of high school in New Zealand (usually 15 to 17 year old students). The first was Projectile Motion, in which an object was thrown over a canyon; then Rotational Motion, where an airplane circled a mountain tethered by an improbably long rope; Roller, with a large cylinder rolling down a slope; and Ice where frictionless sleds moved in various ways. Ice also served as a surface for a number of rocket propelled devices.

A. Projectile Motion

The player (student) is standing on one side of a canyon. Apparatus available is a triangular pivot base (the fulcrum), a plank, and an anvil. The game objective is to throw crates across the canyon. The quantitative physics learning objective is to use the equations of ballistic motion, to calculate the landing point of the cube after being thrown. The student must first assemble a see-saw from the fulcrum and plank, place the crate to be thrown; and then drop an anvil onto the seesaw to launch the crate. There is room here for qualitative experiments – sliding the plank about on the fulcrum allows test throws with different leverage. The height and position – from which the anvil is dropped can also be varied, as can the position and orientation of the whole apparatus.

It is an attractive feature of a simulation world that the construction process can be used as needed, and 'magic' applied at other times to achieve a desired result. A more elaborate lab setup might have included some mechanism for raising the anvil, but we saw that as a distraction from the task on which we wanted the student to focus. It is useful for them to build the seesaw, because that draws attention to position of the plank and provides an opportunity to vary it. We are also interested in where the anvil falls. Adding and having the user operate a crane would not have helped. Instead the anvil simply has the capability of being raised to a measured height above the point on the ground at which it was placed. Students did not comment on this lack of realism during trials.

Fig. 2. Projectile Motion: Seesaw and crate.

On launch the simulation is paused to give the student a chance to calculate the landing point of the crate. An interesting design problem is to decide on the best way of returning numerical results to the user. Gary's Lab uses two different methods, illustrated in this experiment. Launch speed and angle for the crate are reported automatically as shown in fig 3. As well as the text display, the vector is shown in space above the crate (white 'sail' with red outline). The vector is animated as the crate flies, to help show the nature of the motion. Secondly the system includes measurement tools. In this case a tape measure can be used to position a marker at the expected landing zone. It can be seen in fig 3, connecting two giant map pins. The tape length is shown as bill-boarded text next to the near pin.

Fig. 3. Tape measure and velocity display

As projectile motion is particularly simple, involving no collision detection, aside from the final landing, the game physics engine has no difficulty in accurately simulating the motion. Of course, the virtual world conveniently lacks air resistance, so the physics model of the game is the same as that of a high school physics student.

In summary, the general pattern of this lab exercise is to begin with a construction phase and a time for

qualitative experiments. This is the 'sandbox' part of the experiment. Then a trial is undertaken, for which the system requires the user to make a quantitative prediction – in this case placing a target at the expected landing point. Finally, the system runs its simulation and reports success or failure to the user. The simulation provides a nice period of suspense before the outcome is determined.

When Projectile Motion was trialed in a school, the class teacher decided to run the experiment once using a data projector and had the students in the class make individual predictions. The most popular prediction was entered into the system. Again the suspense time worked well.

Whilst the game style of: build; predict; suspense and result was attractive, it hasn't been applied in all of the other environments. At this stage Gary's Lab is not a complete game system, and there are inconsistencies; it being deemed more valuable to trial as many experiments as possible. Some are game styled, and some work in sand box mode.

B. Rotational Motion

The physics objective here was simply to relate angular velocity, speed, radius and period. However, the game aspects of the lab are the most complex of all the labs yet developed. The setting is of a plane, tethered to a tower at the top of a hill in the centre of an island. In fact, the plane begins on the ground. The player must connect a rope from the plane to the tower, and then fly the plane up to a cruising height. They have control of aircraft pitch allowing them to fly up or down. The game physics controls circular motion – by the mechanism of the rope connected to the plane. There is a game play aspect. Once cruising altitude has been reached the player must keep the plane level and complete a circuit through the yellow 'smoke' rings. The calculation required is to determine the plane's orbital period from its speed and the length of the rope (measured using the tape measure).

Fig. 4. Rotational Motion: Flying through rings

The physics modeling is complex in this exercise. The rope is built as a hinged line of rigid segments. It performs well so long as is it kept tight. The rope segments have some mass, so the rope sags a little. If the rope is allowed to become slack, and the plane suddenly stretches it, the physics system's handling of the stretched connections tends to apply a sudden large force to the plane, flicking it through the air. It doesn't seem unreasonable for something like that to happen with the rope suddenly snapping tight. A real rope would either break or pull the aircraft apart. Even though this behavior might not be ideal, it doesn't lead to the player getting bad numerical results. If they lose control of the plane they don't get any results at all. Their only option is to try again. Considerable effort also went in to keeping the aircraft from tumbling at the end of the rope. To simulate the stabilizing effect of the tail in the physics model, a small force was applied on each frame to keep the plane pitched into its direction of flight.

C. Roller

In this laboratory, a large cylindrical object rolls down a slope. The physics task is to predict its velocity at the base of the slope from knowledge of its radius and the height of the slope. The user interface issue was to provide a device for measuring velocity. After considering radar guns, we settled on something more visual if less realistic. The idea is that the two poles with the white rings (see fig. 5) support a 'force field' that measures the velocity of any object passing through. The result is shown as a vector. This proved to have application in a number of settings.

Experience with the physics engine was good. Provided that the slope is smooth, the cylinder arrives at the bottom with the expected velocity. A rough terrain causes it to lose energy, which is probably realistic.

D. Ice

The final experimental environment was a flat sheet of ice. A sled can slide on the ice – perfectly frictionless. Various experiments were developed. The first involved a constant wind, like that that often flows down from the South Polar Plateau. The first physics problems were to calculate velocity by timing the sled running over a measured distance, and to calculate acceleration by measuring speed change over a measured distance. An extension to the velocity gate was made. Attaching a clock allows the gate to record the time at which an object passes through. In Fig. 5 a sled has just passed from left to right. The clock on the gate is frozen at the moment of velocity measurement.

The most interesting problems in Ice were to do with arranging collisions between sleds. To have two sleds collide, it was necessary to set them in starting positions, start them at the same time with suitable velocities, and most importantly to ensure that they hit head on and accurately aligned.

A good feature of the system had always been that physics was active at most times when the player was working. The only exception was having a simulation pause to ask the user to make a prediction.

At first, getting objects started at the same time seemed to require that physics be frozen. Usability tests showed that this wasn't a good idea. It was hard to remember that, and visually not obvious when, physics wasn't active. To avoid this, the mechanism of having a single sled enter a frozen state when it had its velocity set was adopted, visually denoted by keeping the setting 'vector' visible.

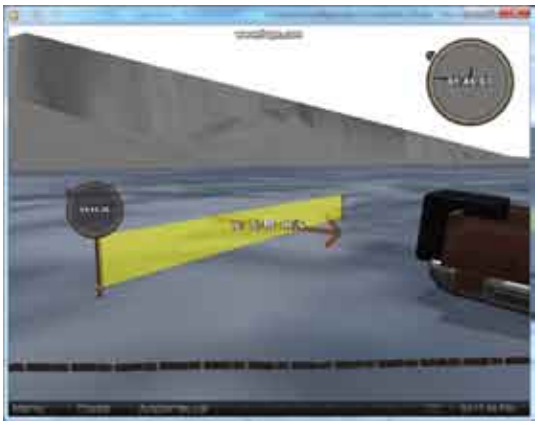

Fig. 5. Velocity gate with clock after sled has passed

The next problem was to allow experiments in which more than one sled starts moving at the same time. We adopted the idea of a plunger, as used to set off explosives, with connections to each item needing activation. Activating the plunger sends a (slow, visible) signal down each wire, guaranteed to reach each destination at the same moment, regardless of the lengths of the wires. To connect to more than two destinations, plungers can be cascaded. Connecting the wires is part of lab construction, focusing the student's attention to the timing issues in an experiment. Fig. 8 shows one plunger wired to a second that is attached to two sleds. A plunger attachment wire is also visible in fig. 6.

The next aspect of setting up collisions was making sure that sleds are properly aligned at the start of an experiment and that they travel in straight lines. Straight line travel was handled nicely by the physics engine. The reason a real sled travels straight is because its runners cannot easily slip sideways as they cut slightly into the ice. In the physics engine that can be simulated by setting anisotropic friction: high for sideways motion and zero for forward/ backward motion. These characteristic of the sled are obvious to the player. If you stand beside a sled and push (by trying to move towards it), nothing will happen. If you stand at an end you can push the sled easily.

Alignment was more troublesome. Getting sleds aligned required some magic – the introduction of apparatus that couldn't reasonably exist in the real world. In Fig. 6, note that there is a block of wood lying on the ice on each side of the sled. In fact these are half of a set of four blocks. The other two are left and right of the other sled. To align two sleds the player must place a pair of blocks about each of the sleds. The blocks are coupled. If one is picked up and moved closer to its partner, the partner simultaneously moves towards the one being held; and the other pair echo their behavior. The two pairs always remain aligned with each other. So, to align two sleds, all that is necessary is to pick up a block and bump it in against the sides of the sleds a few times. Both sleds get pushed into alignment. Afterwards, the blocks can be parked out of the way. The sleds won't creep out of alignment because their anisotropic friction prevents sideways movement.

In a few operations, therefore, sleds can be aligned, given a starting velocity and wired to a starting trigger. A variety of collision and race experiments are possible.

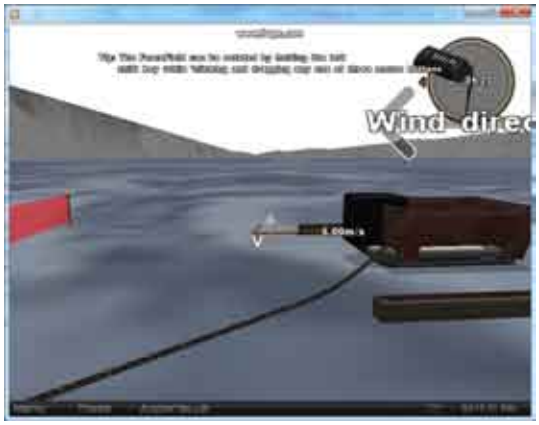

Fig. 6. Sled with physics frozen and velocity setting vector.

With the mechanics of experiment construction out of the way, it is possible to focus on other aspects of collisions. An important characteristic of colliding objects is whether or not their collisions are elastic. It seemed undesirable to have different types of sled for this purpose. Instead, the construction metaphor was employed again. Two kinds of buffer are supplied. The student can modify a sled by attaching one or the other. Black buffers are elastic, like blocks of rubber. Blue buffers are inelastic. We have added a particle effect to suggest that they are crushable containers of water, as are sometimes used in motorway barriers to absorb impact energy. Fig. 7 shows the moment of contact of two blue buffered sleds.

E. Making connections.

It is important to consider the process of connecting objects in 3D space. In Gary's Lab an object can be picked up by looking at it and left clicking (objects flash yellow when in focus, to avoid uncertainty as to which will be picked). The object can then be moved and re-oriented as required. Putting it down in an exact 3D position can be a challenge because it is difficult to judge depth on a two dimensional screen. For simply stacking objects, where very precise location is not critical, shadows provide enough guidance. Reaction by the physics engine to objects touching by accident is avoided by turning off collision detection on the object being moved, making it possible to move it in crowded surroundings, or even out from under another object, without unwanted side effects.

When objects are being connected, however; for example a rocket to a plank or a plank to a fulcrum, high precision placement and orientation is needed. A plank pushed by two rockets has no chance of flying straight unless the two rockets point in the same direction. Placing objects in contact is dangerous as the physics system will react strongly if the user releases an object while it is penetrating another, pushing the objects apart, often 'explosively'.

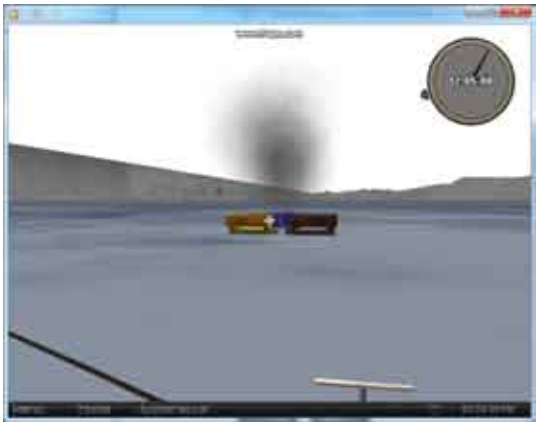

Fig. 7. Inelastic collision, soon after contact.

Fig. 8. Attachment points displayed on plunger.

In Garry's Lab we achieve precise placement by including oriented attachment points as part of object models. These points are highlighted when an attachment process begins. For example, when connecting to a plunger, the plunger displays attachment points as hexagonal rings (see fig 8). Connections are made to attachment points, and are therefore always accurately aligned.

4. CONCLUSION

At its current state of development, Gary's Lab should be viewed as a prototype for ideas in teaching physics using game technology. Our thesis is that a 3D environment is appealing to students and is capable of delivering immersive experiences that have the potential to give students useful insight into the processes of motion and collision. We have established that a standard commercial game physics engine can deliver sufficiently accurate numeric results to satisfy the needs of a high school physics lab in this domain over a good range of activities.

We have conducted limited user testing. The Projectile Lab had a good reception from a high school physics class, and most aspects of the user interface have tested and refactored with laboratory volunteers. It is difficult to establish the merits of educational software more deeply. Ideally we should try to determine whether our software actually leads to students having better understanding of physics. However, it may be more realistic to look at secondary measures. For example, we already have evidence that students like the software. If that meant that they spent more time working with physics than they might otherwise have, then perhaps the software could be said to be worthwhile. Possibly, as the digital native generations spread through the education system, we will need appealing interactive software just to maintain the status quo, as they demand and expect an educational experience which is of the same standard as the entertainment they spend so much time enjoying.

REFERENCES

[1] A. L. Aitken, Playing at Reality: Exploring the Potential of the Digital Game as a Medium for Science Communication, PhD Thesis, ANU, 2004

[2] Broderbund Software, Where in the World is Carmen Sandiego, 1985, described in http://en.wikipedia.org/wiki/Where_in_the_World_Is_Carmen_Sandiego

[3] B. Davidson, Math Blaster, 1987, described in http://en.wikipedia.org/wiki/Math_Blaster

[4] R. Koster, A theory of fun for game design, Paraglyph Press, England, 2004

[5] G. Newman, Garry's Mod, 2006, http://www.garrysmod.com

[6] NVidia, PhysX Technology, http://www.nvidia.com/object/physx_new.html, 2009

[7] A. J. Stapleton, and P. C. Taylor, Physics and Playstation too: Learning Physics with Computer Games, AIP Conference, 2002

[8] Wikipedia, SimCity, retrieved from http://en.wikipedia.org/wiki/SimCity, 21 Feb, 2010

[9] J. L. Zachary, Introduction to Scientific Programming Computational Problem Solving, 1996 (associated web site) http://www.cs.utah.edu/~zachary/isp/applets/Cannon/Cannon.html.

Bill Rogers is a senior lecturer in Computer Science at the University of Waikato in Hamilton, New Zealand. His research interests are HCI, large screen displays and recently games and computer graphics.

Dacre Denny received his B.E. in software engineering in 2008 and has just finished an M.Sc. in computer graphics, both at Waikato University. His art and software can be seen at www.dacredenny.com

Jonathan Stichbury received his B.E. in software engineering at Waikato University in 2008 and now works as a developer at the Hamilton based company SmarTrak which makes GPS based vehicle tracking systems.