Exploring the Educational Potential of Modern Mobile Games

by

Daniel O'Byrne, B.A.(Mod)

Dissertation

Presented to the

University of Dublin, Trinity College

in fulfillment

of the requirements

for the Degree of

Master of Science in Computer Science

University of Dublin, Trinity College

August 2011

Declaration

I, the undersigned, declare that this work has not previously been submitted as an exercise for a degree at this, or any other University, and that unless otherwise stated, is my own work.

Daniel O'Byrne

August 30, 2011

Permission to Lend and/or Copy

I, the undersigned, agree that Trinity College Library may lend or copy this thesis upon request.

Daniel O'Byrne

August 30, 2011

Acknowledgments

I would like to thank my supervisor, Brendan Tangney, for his suggestions and advice throughout the project. I would also like to thank Elizabeth Oldham and all of the student playtesters who helped to evaluate the final two game applications. Finally, I would like to thank Amy Davidson for all of her support, especially towards the end of the project.

Daniel O'Byrne

University of Dublin, Trinity College
August 2011

Exploring the Educational Potential of Modern Mobile Games

Daniel O'Byrne
University of Dublin, Trinity College, 2011

Supervisor: Brendan Tangney

In recent years there have been increasing calls to utilise modern information technology for educational purposes, especially in subjects such as maths where performance figures have been gradually falling. Game-based learning research has indicated there is a huge deal of potential in using modern computer gaming technology for educational purposes [1, 2, 3, 4], yet the educational computer games produced thus far have been almost universally disappointing for a variety of reasons such as poor design, lack of funding, educational unsuitability of certain gaming genres and an inability of these educational games to compete financially with their entertainment counterparts [5, 6, 3].

This research project aims to investigate the suitability of modern mobile platforms, such as the iPhone and iPad, to the development and deployment of educational games.

It was decided to develop a set of educational mathematics games on Apple's iOS platform in order to explore this topic. A generic 2D game framework was first developed which utilised the iOS platform's existing core libraries to provide a wide range of higher level functionality commonly used by modern games. This framework enabled the rapid development of two separate educational mobile games. These two games were each based around a different input method (the touchscreen and the accelerometer) and they each covered a variety of mathematical topics including number theory, inequalities, fractions and algebra.

An exploratory case study was used to investigate both the quality and the educational merit of the two games developed. Data was collected from three sources to give an element of triangulation to the results (feedback from an educational expert, feedback from young players and observations from the researcher). The participants in the case study comprised of an opportunistic sample of children between the ages of eleven and fifteen, and the data collection instruments included interviews and questionnaires administered before and after the participants had played through the games.

The results gathered clearly show that the two games developed were both enjoyable and educational. They also strongly indicated that modern mobile platforms are extremely well suited to the development of educational games, due to both their current technological capabilities and the current realities of the mobile marketplace.

Contents

List of Figures

Chapter 1

Introduction

Computer Games are now one of the major forms of entertainment worldwide, despite only becoming commercially available in the 1970's. Since the early days of games such as 'Pong', computer games have been catering to a player base of ever increasing size and diversity - resulting in an estimated 72% of American Households now playing video games and with the average age of a 'gamer' being 37 [7].

The expectations of these players, and the technical capabilities of gaming machines, have been increasing almost exponentially for years, and players have now come to expect their games to contain elements such as rich, highly detailed 3D worlds, advanced gameplay mechanics and challenging AI. This in turn has resulted in a steady increase in costs related to the development and publication of modern computer games. Developing a game in the seventies generally involved only a few developers and took only a few weeks to complete. Creating a modern computer game today on the other hand, often involves the collaboration of hundreds of highly specialized individuals such as artists, programmers and designers and the process can take anywhere from two to six years.

A more limited version of these trends can be found in the subgenere of 'mobile gaming'. Mobile games are those played on portable handheld devices which generally have much lower technical capabilities when compared to a PC or a home gaming console on account of their portable, battery powered nature. Traditionally the portable market was cornered by machines such as the Nintendo GameBoy, Nintendo DS or Sony PSP and these devices followed a similar increase in technical capabilities over

time mirroring their non-portable equivalents, if on a slightly smaller scale. They also mirrored the marketing and distribution chains of those non-portable machines, resulting in those devices being primarily gaming devices and the games for them being sold at relatively high prices - with the vast majority of the profit being split between the retailer, the publisher and the console manufacturer.

Over the last five years however, there has been a radical shift in the mobile gaming market due to the introduction of modern smartphones, such as the Apple iPhone and those powered by Google's Android platform. Mobile phones have had games developed for them for over a decade, but they were generally of a very poor quality when compared with the games available for handheld consoles of the same time period. Mobile phones also lacked a good marketing and distribution system for their games, which prevented them from generating meaningful sales. The new generation of smartphones are extremely powerful however, and were designed from the start to act as platforms for numerous different types of applications. Furthermore, their increased internet functionality provides a perfect marketing and distribution platform for these applications, such as games, especially when combined with centralised storefronts such as the Apple 'App Store'.

These new smartphones provide a platform capable of running games almost on par with the dedicated handheld gaming consoles. Furthermore, the new storefronts enable developers to deliver their content almost directly to their customers at a price of their own choosing. This removes the retailer and publisher from the equation, leaving the developer a much larger share of the profits. It has also resulted in a huge drop in the price of these games, with the standard price point of gaming applications settling for the moment on 99c, as opposed to the 40 Euros price point of games found on the dedicated handheld gaming consoles like the Nintendo 3DS.

Over the last two years, tablet PC's have also had a serious impact the mobile market - providing even more technical capabilities and larger screens than the new smartphones. Tablet PCs, such as the Apple iPad, are often based on the same operating systems as the latest generations of smartphones which means it is extremely easy to develop games for both simultaneously, or to port games from one to the other. These tablet devices also have access to the same sort of online storefronts as the smartphones. While it is still too early to definitively see what sort of long term effects these devices will have on the games industry, gaming applications are by far the most

popular available and several games have achieved critical success on these new platforms - for example, the game 'Angry Birds' has achieved over 200 million downloads to date.

1.1 Motivation

As the games industry grew, so did the calls to utilise these new, emerging entertainment technologies for educational purposes. Game-based learning research has repeatedly shown that games and gameplay have the potential to provide a superior learning environment [8, 9, 3], yet almost all attempts to produce educational games have been disappointing [6]. In many cases, this stemmed from a lack of sound learning pedagogies used in the design and development of the games, but another potential reason for the lack of success of educational games comes from the pre-existing expectations of players, based on their familiarity with entertainment based games.

Modern computer games are extremely engaging and a lot of fun to play, but many of the most popular genres and gaming tropes aren't necessarily well suited to educational purposes - while first person shooter games can be fun to play for example, it is difficult to use such a simplistic interaction style to deliver a meaningful educational experience[4]. To some degree however, educational games need to compete with their entertainment based counterparts and while it may not be possible for them to be as 'fun' as those games on account of their need to achieve an additional educational goal, they need to come close and they also need to have equivalent production values.

The educational games market, while potentially quite large, has in reality proved to be far smaller than the entertainment market. Poor sales of educational games over the last 30 years has led to a serious decrease in available investment, which in turn has led to a decrease in production quality when compared to entertainment based games. The educational games industry, or 'edutainment' industry, was simply incapable of meeting the huge investments required to make a top quality game once those games started costing millions to produce. Despite these difficulties however, if good educational games could be successfully developed, they could potentially have a huge positive impact on the standards of education afforded to learners.

The emerging trends in the mobile gaming industry could potentially offer a renewed hope for educational games. The production values and levels of fidelity expected of

mobile games, while high, are far lower than their console or PC equivalents. Many of the mobile games being developed today are being developed by small teams of between 2 and 8 people as opposed to the hundreds needed for the production of a full scale console or PC game. Also, the lower price point could enable educational games to reach a far larger market and make them more palatable to players who would generally prefer to get an entertainment title or who might not otherwise risk their money on an unknown entity. This, combined with the higher percentage profit for developers which comes from cutting out retailers and publishers could improve the profitability of educational games.

On a more technical level, the user input methods afforded by the new mobile devices might suit educational games far better than the controller or keyboard and mouse set-ups used by traditional games. Touch screens can theoretically be used to create more intuitive interfaces than can be created using more traditional input devices (especially for younger learners and non-gamers). These interfaces could potentially provide for deeper interaction as they allow users to naturally select and move things very easily. The accelerometers found in many of the new devices is even easier to pick up and use as it enable players to control elements of the game by simply tilting the device in a given direction.

The goal of this project was to investigate the educational potential of these new mobile devices, especially in light of the economic advantages they may present to educational game developers. In order to achieve this, it was decided to develop two educational games from scratch on one of these mobile platforms - in this instance the Apple iOS platform. This involved first developing a basic 2D game framework (which provided a simple API to perform common tasks such as playing a sound or displaying a texture on the screen) and then utilising that framework to construct the two educational games. It was decided to develop two games in order to explore the main two types of input found in modern mobile devices separately - one game was designed to be controlled using the touchscreen, while the other was designed to be controlled using the accelerometer. Developing these games on the iOS platform provided not only an insight into the challenges of mobile game development, but also how it compared and contrasted with development for a traditional console or PC game.

Once completed, these two games were given to an educational expert to get feed-

back on their educational merit. They were also extensively playtested with a number of children between the ages of 11 and 15 years in order to determine their quality, usability and to further evaluate their educational effectiveness.

1.2 This Report

This report will contain a review of the current state of the art in learning theory, game-based learning and 'edutainment' as they relate to this project. Some of the major learning theories and how they relate to game design will also be discussed.

Some of the major design decisions made during the course of this project will then be explained, as will the reasoning behind them. This will be followed by a detailed summary of the implementation and development of both the game framework and the games themselves.

A detailed analysis of the project will then be presented. First the feedback from an educational expert, as well as the results from an exploratory case study, will be explained and analysed. An analysis of the project as a whole will then take place in light of this feedback, focussing on both the educational merit and the overall quality of the two games developed.

The conclusions drawn from these results, as well as the researchers own observations, will then be presented and a brief outline of the future work to be carried out will be given.

Chapter 2

State Of The Art

2.1 Introduction

Education is becoming increasingly important as many nations, including Ireland, attempt to become knowledge based economies [10]. An increased uptake in educational standards, especially in science and technology subjects, is required to drive this shift and maintain such an economy, yet recent trends show attainment in these core skills is dropping [11]. Many theorise that the education system itself, which has remained primarily unchanged since world war two, is flawed - either due to the lack of sound pedagogical principles employed by the system [12] or due to the rise in expectations from a new generation of multimedia learners (or 'digital natives'), who's exposure to modern technology such as computer games have led them to expect a richer learning environment than the traditional classroom situation can provide [13]. Modern educational technology, and modern computer games in particular, are seen by some as potential solutions to this problem, but their impact has thus far been minimal for a wide variety of reasons which will be discussed later in this chapter.

2.2 Educational Theories

In order for any piece of educational software to be truly successful, it must be based on sound learning methodologies [14]. How people learn and how multimedia technology such as computer games can best support learning is an area that has received a great

deal of study in recent years, with a number of unique and often competing theories emerging on how humans learn and what conditions are best for learning.

2.2.1 Behaviourism

Behaviourism considers the reinforcement of positive learning behaviour with reward or punishment and is easily implemented in a game world by rewarding a player for demonstrating proficiency at a particular task - by completing a level for example. While the implementation of reward systems in educational settings has been criticised for reducing students' intrinsic motivation in tasks once such rewards are removed [15], the effectiveness of such reward based motivation has been repeatedly proven [16]

Virtual rewards of varying perceived value can also be used to encourage better behaviour in a player. For example, if a player were tasked to create the number 10, creating the equation 5 + 5 might grant the player 50 points while the equation 2 * 5 might grant them 200 points - since multiplication is a more advanced operation than addition. In addition to this, the use of virtual rewards in a controlled virtual environment, such as a computer game, can be completely controlled by the game designer. This enables them to always be given fairly - removing a major potential issue with the implementation of reward based educational systems in the real world. Virtual rewards also never run out (though poor game design can still lead to a decrease in their perceived value over time).

2.2.2 Cognitivism

Cognitivism is concerned with how best to achieve learning outcomes with consideration towards human memory and how it works. It replaced behaviourism as the dominant learning paradigm in the 1960s and is often concerned with the specifics of how learners form mental models of the world in order to understand it. Topics from this area such as cognitive load [17] and information processing theory are extremely important to the design of all multimedia learning tools, as they can outline the optimum frequency and format for the introduction of new material.

Cognitive theory research has also served to highlight the value of multimedia approaches to learning, through work such as Paivios Dual coding theory [18], which

displayed the effectiveness of using both visual and verbal mediums to transmit information. There has been a significant amount of research carried out in this area recently, most notable by Mayer, who applied Paivio's initial theories to a multimedia learning environment, such as those that can be created by computer games [19].

2.2.3 Constructivism

The educational theory of constructivism proposes that learning is an active, constructive process, and as such the best situation for learning is one in which a learner constructs their own solutions to problems [20, 21]. More recent work has indicated that such learning is even more beneficial when it occurs in a group, where a social component can help to reinforce learning and reflection [22]. Games and other multimedia simulations are often touted as enabling such learning through the exploration of virtual worlds or microworlds [23].

2.3 Game-Based Learning

It has been established that animals develop more rapidly through play [24] and the absence of such play can have a negative impact on the development of children [25]. The concept of play though, has proven very difficult to define - with competing definitions regularly being drawn [26, 27].

Game-based learning is based on the idea that games, through the directed use of 'play', have the potential to provide an excellent learning environment - one far better than what most learners receive today. Games, especially modern computer games, have several strengths which could make them ideal as an educational medium.

Good games are fun. While 'fun', like 'play', has proven difficult for academics to satisfactorily define, the fact that millions of children worldwide play computer games, of their own volition, is a testament to this fact. Games have been found to be motivating and engaging for learners of all ages [8, 9] The traditional classroom situation however, is rarely regarded by children as 'fun', with the experience more often being described as 'boring' by today's learners. Motivation is a key aspect of effective learning and yet a lack of motivation is a major problem in education today. Even if a child is initially motivated, such motivation needs to be sustained through

active involvement, feedback and reflection [9].

Modern computer games are clearly capable of sustaining a player's motivation - with many games taking tens, if not hundreds, of hours to complete. Many of these games are also relatively difficult - yet they manage to encourage players to persevere and attempt to find new solutions to problems long after most would have simply given up in a classroom environment. Computer games also provide a safe environment in which it is okay for the player to fail. In fact it is often expected that players will fail at a task many times before finding the correct solution. This closely mirrors the constructionist theories on learning and contrasts with the traditional classroom environment, where children are expected to always be correct [28].

A great deal of research into game-based learning has been carried out over the past few decades, with many extremely positive results emerging. Simulation games have been successfully used to train advanced skills and tactics by the American military and air force for example [29, 30].

Much of the work in the field aimed at younger learners (primary and second level) has focussed on using commercially available entertainment based games to support existing curricula objectives however, and this has met with varying degrees of success [31, 5]. These studies involve taking off the shelf entertainment games (often strategy games), either as they were originally sold or with some modification, and using them in a school environment. While the results of these studies often indicate learning is taking place, it is often not the specific types of learning that was initially desired [31].The reason for this is that a great deal of time is 'wasted' learning how to play the game itself or focussing on areas of the game that are entertaining but don't contain the desired educational content.

An obvious solution to this problem would be to develop specifically tailored educational games aimed at achieving specific educational goals. However the 'edutainment' games produced thus far have been extremely disappointing for a number of reasons which will be discussed later in the next section.

2.4 Edutainment

The 'edutainment' industry became popular in the early 1990s as a huge influx of educational software was developed which claimed to provide a 'better' way for chil-

dren to learn. Much of this software took the form of computer games, or contained 'game-like-elements', which claimed to make learning more 'fun'. Unfortunately, little to none of this software was developed using sound pedagogical principles, so the educational component tended to consist of drill and test exercises and multiple choice questionnaires poorly disguised with flashy graphics [6]. The gameplay elements of this software were also extremely lacking - with the edutainment games being technologically far behind the blockbuster entertainment games available on the market at the time, and with their gameplay often being repetitive, derivative and boring in nature.

The edutainment industry swiftly collapsed by the start of the new millennium, as consumer interest in educational games lessened. Today, many educational games and brands still exist, but they no longer directly compete with entertainment titles like they once attempted. Instead, the educational games industry has become an almost completely separate entity to the commercial games industry - with the target demographic being parents and educational institutions rather than children (who have proved to be far more likely to opt for an entertainment based game if given the choice). This is a severely limited market compared to entertainment computer games [32].While the graphics have improved, the basic gameplay and experience found in educational games has remained the same for the past two decades with few exceptions.

Despite the seeming failure of educational games, the entertainment based games industry has become incredibly successful in recent years. The games industry is estimated to be woth approximately US$68 Billion globally [33] and is often seen to be one of the main driving forces behind the advancement of commercial graphics hardware. The variety of gameplay experiences have continued to grow and change in tandem with advancements in hardware over the past two decades and computer games have now diversified into a wide variety of genres catering to a wildly diverse global audience.

In theory, the advancement of the technology and techniques of the entertainment games industry should enable for the creation of better, more successful educational games, but there remain a number of significant challenges to the creation of good educational games.

As computer games have grown and evolved, so too have children's expectations of them. What was considered an amazing game ten, or even five years ago, most likely wouldn't do well if newly released onto the market today. The expected graphical and technical fidelity of a modern computer game is enormous, and in order to achieve this

fidelity, so are the development costs. Many modern computer games can take teams of one hundred or more people, two to five years to create and can cost over one hundred million euros to develop. Due to the comparatively small market for educational games, it is unlikely one could raise even a fraction of this level of investment.

Cost isn't the only obstacle faced by educational games however. For every great game released, there are also a number of failures - game design is still considered by many to be more of an art rather than a science [34] and many big budget entertainment titles fail to achieve that hard to define state of 'fun'. Developing a game that is both fun and educational is even harder - while educational elements can be enjoyable, a balancing act usually needs to be played between the amount of educational content and the amount of pure entertainment placed into a game.

Another potential issue faced by educational games is that of input devices and established gameplay mechanics. The main genres and gameplay mechanics found in the games industry have evolved in part to fit the input devices provided by either consoles (gamepads) or the PC (keyboard and mouse). Entertainment games just have to provide a fun experience, so if something doesn't translate well to the medium or can't be achieved easily with the input devices available, its simply isn't used. The most common gameplay elements are often encapsulated in game engines - which can be re-used between games to reduce development time and costs significantly. Educational games on the other hand need to achieve specific educational objectives - which often require completely unique gameplay mechanics. When this requirement is combined with the previously mentioned technological limitations it can severely hamper a development team's design freedom and increase the cost of developing good educational games.

A new industry trend may help alleviate some of these issues however. In 2008 Apple opened the online 'App Store' in support of its increasingly popular iOS platform (used for the iPhone, iPod Touch and iPad). The 'App Store' has proven to be hugely popular amongst both developers and consumers, with over 250,000 apps released and over 5 billion downloads as of September 2010 [35]. As these are portable platforms, with limited hardware capabilities and screensizes, the expected fidelity levels of their games are more easily achieved than games found on the current generation of dedicated gaming hardware.

This has led to the development of gaming applications by small teams, often

numbering less than 10 people. Generally these platforms also offer more natural user interfaces than many existing gaming platforms, with accelerometers and touch screens. These features, combined with the fact that the 'App Store' also provides an extremely low cost route to a growing global marketplace, could enable the development of good educational games to become both viable and profitable.

Chapter 3

Design

This chapter will discuss some of the major design decisions and challenges this project presented. Firstly, the choice of mobile platform on which to construct the game application will be discussed and justified. The need to design a framework on which to base the mobile games will be explained, as will the decision not to use one of the game engines that are already commercially available. The reasoning behind the decision to split the focus of the project between two separate games will then be given, and the basics of each game design and how those designs evolved, will be discussed. Finally, the software development methodology adopted for this project will be discussed, as will the reasons behind its adoption.

3.1 Deciding on a Platform

A number of new operating systems and platforms have emerged to compete for the growing mobile market. Amongst these, the main contenders are the iOS platform from Apple, Google's Android platform and the most recent addition, Microsoft's 'Windows Phone 7' platform. Blackberry and Nokia's Symbian also control major shares of the smartphone markets, however up until very recently the majority of their devices don't have the major technological advantages that the others have, such as touchscreens and accelerometers.

The 'Windows Phone 7' platform is the newest to the market and apps can be created for it using Microsoft's XNA system. XNA has been around since 2006 and

can also be used to create games and applications for the PC and Xbox 360. The code base and libraries used to create applications for the 'Windows Phone 7' platform are almost identical to those used on the PC and Xbox, so there is an existing user base and support structure to assist with development. The 'Windows Phone 7' platform hasn't yet managed to grasp a significant share of the mobile market however[36], and the resultantly small user base prevents most developers from considering it as a viable alternative to Android or iOS.

The Android platform was released by Google in 2007 and since then it has become one of the dominant mobile operating systems on the market. One of the main reasons for this is its open nature - the Android operating system is based on the Linux kernel and its low costs and open source roots have led to a number of third parties using it as an operating system in their own phones. While this has led to the huge adoption of the Android operating system, with analyst expecting its market share to reach 50% over the next year[36], it also presents developers with a number of issues. Android's share of the market is heavily fragmented, with a wide variety of different hardware and potentially conflicting software adding considerable difficulties to the development and testing of new applications. Furthermore, the Android 'app' market is also fragmented, with a variety of competing online marketplaces making it more difficult to successfully commercialise on and market new applications.

Apple's iOS platform, which runs on both the iPhone and the iPad, is currently ranked second in the smartphone market with between 15% and 20% of the market share[36], but it is the clear leader in the tablet market with an estimated 68% of the market share[36]. More importantly however, there is a single unified marketplace for the iOS platform - the Apple App Store. While the vast majority of content on the store is created by third party developers, Apple's control ensures a certain minimum standard of quality for the user experience provided by these applications. Apple also completely controls the hardware on which the iOS platform runs, which means developers can tailor their applications to the hardware's strengths and limitations more easily, and to a greater degree, than they can for Android devices. These advantages, among others, have lead to the iOS platform far outperforming the others in terms of application sales, controlling over 80% of the market revenue [37]. This has led most mobile developers to view the iOS platform as the primary mobile platform despite the Android's superior smartphone market share. While this may change in the future, for

the moment it results in most mobile developers developing for the iOS platform first and then porting their applications to other platforms at a later date.

The iOS platform's libraries and applications are written primarily in objective-C, Apple's own derivative of the C language, though they can mostly also be written in C and C++. While the iOS platform itself is relatively new, having only been released in 2007, many of its libraries and core frameworks are from Apple's Mac operating system which means that the core codebase is actually quite mature and stable. Apples development environment, Xcode, is also very mature and stable, and includes a wide variety of useful tools and support software such as performance analysers and an iOS simulator.

It was decided to develop this project's applications for the iOS platform, primarily for the iPad as the larger screen size opened up additional options in terms of game design. Another advantage of the iOS platform is that the applications code for the iPad and the iPhone are almost identical, making it possible to port from one to the other. iOS development must be done on a Mac, as Apple's development software is designed to only run on their own operating system. While this would normally argue against developing the project's application for the iOS platform, a suitable Mac and first generation iPad were available and provided for the project.

3.2 Building a Generic Game Framework

The iOS platform is extremely advanced when compared to the last generation of mobile operating systems. It's derived from Apple's Mac OS X and has a wide selection of core libraries to aid developers in the creation of new and unique applications. Unfortunately, games are also very complicated and while the iOS libraries enable developers to tap into the huge potential contained within devices like the iPad and the iPhone, it does so at a very low level. Some common application types such as user interfaces, image viewing and on-screen keyboards have high level support, but this functionality is generally too restrictive for use in game applications.

For more advanced graphical applications, like games, the iOS platform supports OpenGLES - a mobile version of the standard cross platform graphics library OpenGL. For audio, OpenAL is supported and the platforms native objective-C language supports a wide range of data types used in the development of modern games. Good

game applications tend to be technically complex however, even if they appear relatively simplistic, and they generally require more advanced functionality than these libraries support 'out of the box', such as loading and displaying textures, displaying fonts, generating particle effects and saving the gamestate when the application closes. Game applications generally have a base engine or framework which supports this more advanced functionality with a high level API.

A number of commercial game engines such as Unity3D[38] and Unreal[39] support the iOS platform, however they tend to be optimised for existing game types such as first person shooters or 2D platformers and can hamper the development of more unique games specifically tailored to educational purposes. Many application developers create their own game engines, either from scratch or based on existing code, and then use these engines as the basis for a number of different games - improving and expanding them to support whatever additional functionality is needed. This was the approach adopted for this project as it would provide the advanced functionality needed, while also enabling the engine to be kept relatively small and lightweight by not including a huge range of features unlikely to be used for the projects own game applications. It would also have the added effect of providing a more thorough insight into the additional challenges facing developers who need to create their own engine from scratch - as would most likely be the case for any developer who wanted to develop a series of educational games.

The design of the project's framework was based on the premise that any educational game would require at least three types of higher level functionality - the ability to play sound, the ability to display a moving texture and the ability to display moving text. This led to the development of Font, Texture and SoundEngine classes. Mobile devices have far more stringent limitations in terms of processing power and memory than PCs or consoles and the iOS platform lacks an automated garbage collector, so memory must be managed manually. This necessitated some form of unified resource manager to manage what assets were stored in memory at any given time, among other issues. The resource manager class was designed with a class level instructor - meaning only one instance of it would ever exist. Having a single unified resource manager, which spanned across multiple levels and screens, helped alleviate a number of potential memory management issues.

Other commonly needed game elements were added to the framework later in de-

velopment and these included a separate particle engine, a number of macros defining useful functions such as random number generator and a number of commonly used vector-math functions such as the dot and cross products.

3.3 Splitting Focus Between Two Games

Many modern mobile devices, including the iPhone and iPad, have multiple different input methods. The two most common methods of interacting with the devices are through the touchscreen or the accelerometer. In most applications the accelerometer is primarily used to simply flip the screen and change its orientation between landscape and portrait modes. Many gaming applications have made more thorough use of the accelerometer however, by using it to take control of in game characters and objects. While the accelerometer is more limiting than the touchscreen in many ways, it is extremely easy and natural to use.

The touchscreen is the main input method used for the vast majority of applications on modern mobile devices. It can be used to select an object or a group of objects, to drag these objects around the screen and even to type (through the use of a virtual keyboard). Like the accelerometer, learning to manipulate the touchscreen is very intuitive and easy to pick up - though applications based on it can be far more complicated than those designed to be controlled by an accelerometer and as such can be harder to use. The touchscreen can also be better suited to 'selecting' from or manipulating a group of objects in a dynamic way than a game controller is, and this is behaviour commonly desired in educational games.

Rather than focus on a single game, it was instead decided to develop two games. One game would be designed to be controlled by the touchscreen while the other would be designed to be controlled by the accelerometer. Basing each game on a different control and input scheme enabled an investigation into which of the two input methods younger learners preferred and found easier to pick up. Developing two different games also lessened the effect the 'game design' would have on the feedback received from educational experts and playtesters towards the end of the project with regards to the educational potential of the platform as a whole.

By first developing a basic framework and then using that in the development of the games, the decision to develop two separate games didn't represent a decision to

double the amount of development work. When coupled with the fact that both games were designed to use very similar artwork and game-mechanics, the second game could be developed in a fraction of the time it took to develop the first.

3.4 Simple Game Design, Open for Expansion

The games found on the new mobile platforms tend to be far shorter and simpler than their non-mobile counterparts for a number of reasons. This is partially due to the smaller design team and development time afforded to mobile titles, but it is also a reflection on the different circumstances users find themselves in when playing them - a console or PC gaming experience can, and often does, involve gaming sessions of several hours in length, while mobile games are designed to be played on the go, while travelling or for short breaks only a few minutes long. To reflect this, the games developed for this project were designed to be simple to just 'pick up and play' and contain levels that could be completed in only a few minutes. Keeping the games simple was also necessary to enable younger players to pick them up quickly.

While the core mechanics needed to be simple, the games themselves needed to be open to expansion in the future. While it wasn't necessary to develop the games to a point where they could be released commercially on the App Store, they needed to accurately reflect fully developed games so they could be properly evaluated. In order to do this, they were designed with future features and expansions in mind, such as additional levels, enemy types and power-ups, which restricted the potential game designs to those that could be fleshed out into fully featured games rather than one off novelties.

The limitations of having a single developer also featured heavily on the game design. Certain elements were intentionally kept simple or added to ease the amount of artwork and development time required. For example, all of the game objects were restricted to spheres, as this simplified not only the collision detection code but also the artwork as textures could simply be re-coloured to represent different objects. Particle engines were also used extensively to provide much of the games graphical appeal. '

3.5 Design of the Touchscreen Game

The first game developed was the one designed around the touchscreen. The educational content was originally envisioned as being basic mathematical number theory, but this was later expanded to also include algebra, fractions and squares.

3.5.1 Game Mechanics

As seen in Figure 3.1, the screen is broken into separate regions during the touchscreen game. At the top of the screen is the menu bar. This allows players to exit to the main menu at any time during a game, displays the current level being player and displays the player's current score. The bottom fifth of the screen contains the numbers and mathematical operators currently available to the player. The rest of the screen is taken up with the 'playfield', which contains the goal star the player must hit to win the level as well as various enemies which move around on set paths blocking the player's progress.

The goal of the levels in the touchscreen game is for the player to reach the star at the top of the screen. There are a number of 'enemies' blocking the players path to the star however, and the player must generally destroy a number of these enemies before it is possible to reach the star. The player can drag, or fling, a number of spheres around the screen and these are used to clear a path. Enemies, the goal star and the player's spheres all have values displayed on them. Enemies and the goal star can only be destroyed by hitting them with a sphere of the same value.

The player can manipulate the value of their spheres by using the mathematical operators they have available in a given level. For example, a player could select a sphere with a value of five, then select the multiplication operator and finally select a second sphere with a value of three. This action would result in a new sphere, with a value of fifteen being spawned and replacing the first operand in the operation (the sphere with the value of five). The sphere used as the second operand in the operation would disapear, to be replaced by a new randomly valued sphere after a few seconds.

The numbers made available to the player are always between one and nine (with the exception of the negative numbers level, in which the numbers have a 50% chance of being negative), so the players are forced to make use of the operators available to the right of their numbers in order to create spheres with larger values. The values

given to newly spawning player spheres are assigned pseudo-randomly (less sevens are spawned as they are harder to work since they are large primes).

While the numbers provided to the player and the values of the enemies all fall within a particular range for each level, they are randomly generated within that range. This gives all of the levels a very high degree of replayability, as players will almost always find themselves having to use and generate different values. It was important that the game support this sort of procedural level content in order to make it suitable as a tool for learners to practice with.

While the player only needs to destroy enough enemies to get to the goal star, points are awarded for every enemy the player destroys, so it is often in the players interest to destroy extra enemies. To counter this however, the player looses points over time - which serves to lend the game a sense of urgency. How many points the player gets per enemy destroyed changes depending on the level, meaning players can be given more time to generate more complicated numbers. The base score values have the player receiving twenty points per enemy they destroy, but they lose one point per second and an additional five points for every enemy they hit in error and fail to destroy. This means players must hit a new enemy every twenty seconds on average, assuming they make no mistakes, to get a score above zero - though most players would want to get as high a score as possible and so would want a much higher average than that.

Seven separate levels were developed for this game and while they mostly have the same enemy layout and behaviour patterns, the mathematical content for each level is different. The enemies in each level are divided up into four rows, each of which is twice the width of the screen. These rows move back and forth across the screen, in a slightly similar vain to those found in the classic space invaders game - except that these rows extend beyond the width of the screen, making the enemies in the centre more valuable as they are on the screen longer than those at the edges. Each row moves in the opposite direction to the row immediately below it, which prevents players from making a permanent, stable path from the bottom of the screen to the star - at least, not without destroying a large amount of enemies in each row.

While the design of the levels developed were loosely inspired by space invaders, the game design as a whole is far deeper than that. A number of alternative level layouts were prototyped and a huge variety of different levels could be created simply by changing the enemy positions and movement patterns. Additionally, the game is

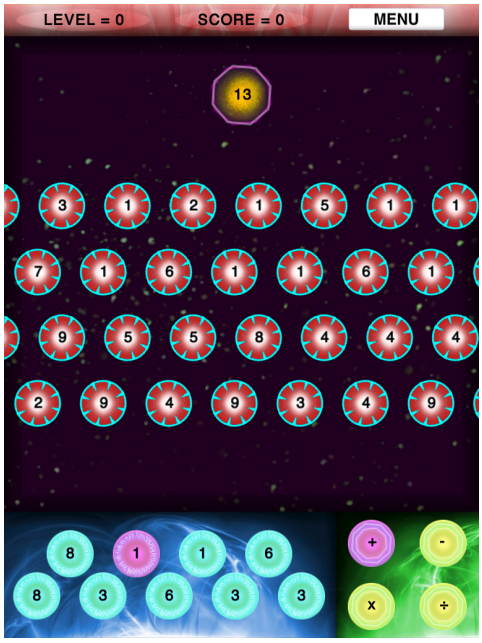

Figure 3.1: Touchscreen Game - 'Practice Level' Screenshot

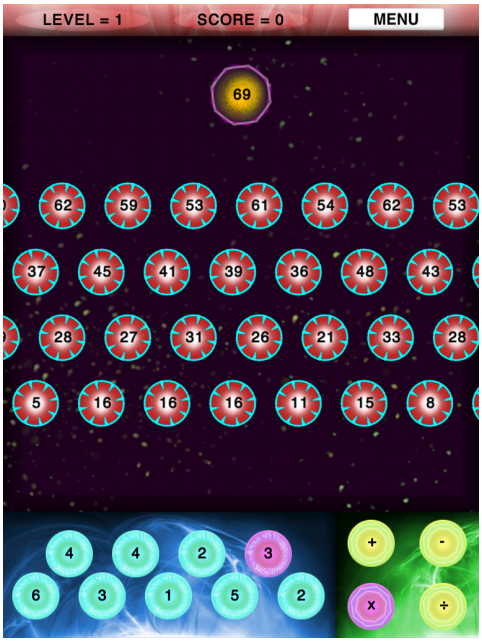

Figure 3.2: Touchscreen Game - 'Basic Numbers Level' Screenshot

open to expansion through different enemy types (such as enemies that can only be destroyed by player spheres created using division or which were created using four or more other spheres), powerups (enabling things like a blast radius when you destroy an enemy, or spheres which destroy enemies with values close to their own rather than having to exactly matching their own value) or indestructible obstacles on the map such as walls, which would limit the range of player movement.

Practice Level

This level, as seen in Figure 3.1, is used to provide a basic introduction to the game. Most of the enemies can be destroyed by the numbers already available to the player, so players don't need to use the operators much and they can afford to make mistakes. The purpose of the level is not to test or improve the players mathematical abilities, but to get them familiar with the games controls - especially if they haven't used a touchscreen device before.

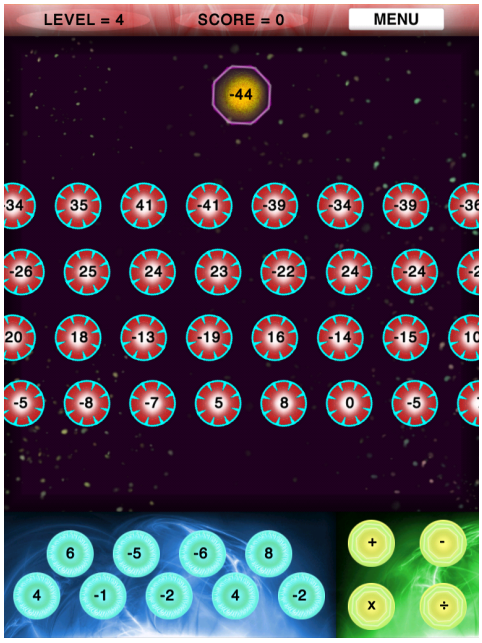

Figure 3.3: Touchscreen Game - 'Negative Numbers Level' Screenshot

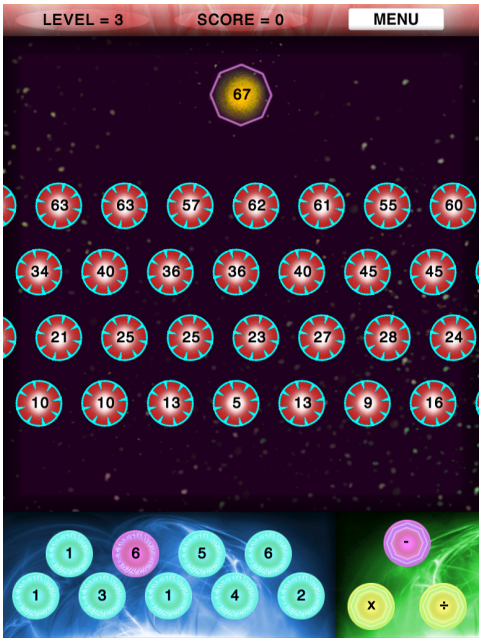

Figure 3.4: Touchscreen Game - 'No-Addition Level' Screenshot

Basic Numbers Level

This level, as seen in Figure 3.2, is the first 'proper' level of the game. Here the enemies' values range from 0 to 64, so the player is forced to make use of the operators in order to finish the level. The values found on the enemies increase successively with each row as the player makes their way up the screen, requiring the player to perform progressively more complex actions as they progress through the level.

Negative Numbers Level

This level, as seen in Figure 3.3, is similar to the previous level with the addition that approximately half of the enemy values are negative. Player spheres, both those the player starts with and those that re-spawn as the player uses them, also have a 50% of being negative.

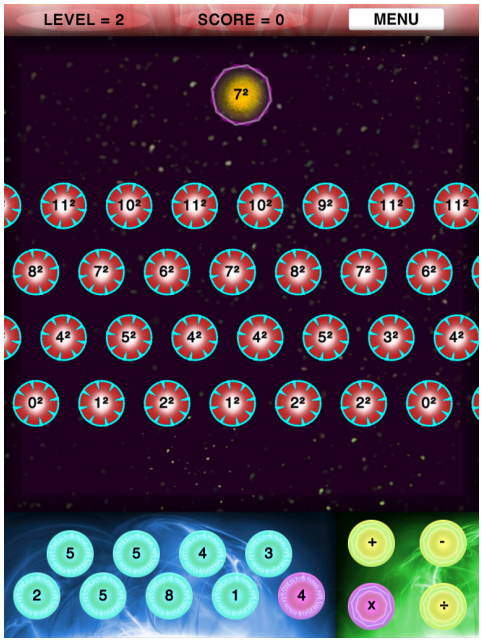

Figure 3.5: Touchscreen Game - 'Squares Level' Screenshot

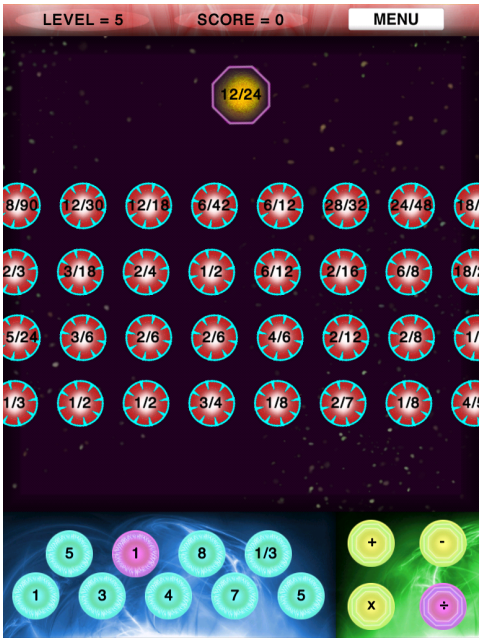

Figure 3.6: Touchscreen Game - 'Fractions Level' Screenshot

No Addition Level

This level, as seen in Figure 3.4, is also similar to the Basic Numbers level, except that the addition operator has been removed. This level was developed after it was observed that players had a tendency to over-rely on addition to get them through the game (Some players would just continually add small numbers together to achieve almost any large number). This level forces players to make more use of the more complicated operators.

Squares Level

In this level, as seen in Figure 3.5, the enemy values are represented as squares (a number to the power of two). This required players to not only be able to generate a given number, but also to understand what squares are.

Fractions Level

In this level, as seen in Figure 3.6, the enemy values are represented as fractions. While the bottom row of enemies is comprised mainly as common fractions in their simplified forms, the fractions found in each row of enemies grow in complexity as players move up the screen. While the fractions found in the upper rows can be quite complex, they can all be simplified down to more basic fractions. Despite the fact that the enemies are all fractions, the player is only given whole numbers to work with - forcing them to create fractions on their own. This requires players not only to use the division operator, but also to grasp that the line in a fraction represents division. Furthermore, when the player creates a fraction, it is automatically reduced to its most simplified form - forcing players to realise that a complex and simple version of a fraction can have the same value (eg. $1/2 == 2/4 == 4/8$). Generating player spheres with fraction values was considered and tested as well, but having to deal with random fractions in that way proved too difficult for most players.

Algebra Level

In this level, as seen in Figure 3.7, the enemy values are represented as simple algebraic equations which the player must solve. The layout of the level is slightly different from the others in that the enemies are much larger and, as a result of that, there are fewer of them. This was necessary in order to fit the equations onto the enemies, as they take up a lot more space that singular numbers or fractions. In order to destroy an enemy in this level, the player must hit it with the value of the unknown variable of the algebraic equation. The unknown variable found in each equation can be represented by a number of different letters, highlighting to player the fact that the character used to represent the unknown variable doesnt matter.

3.5.2 Educational Content and Methodology

As mentioned in the previous section, the educational content for the game was originally envisioned as being basic mathematical number theory, but it was later expanded to also include squares, fractions and algebra. In the context of this project, basic mathematical number theory is taken to be number theory as it relates to young learners in

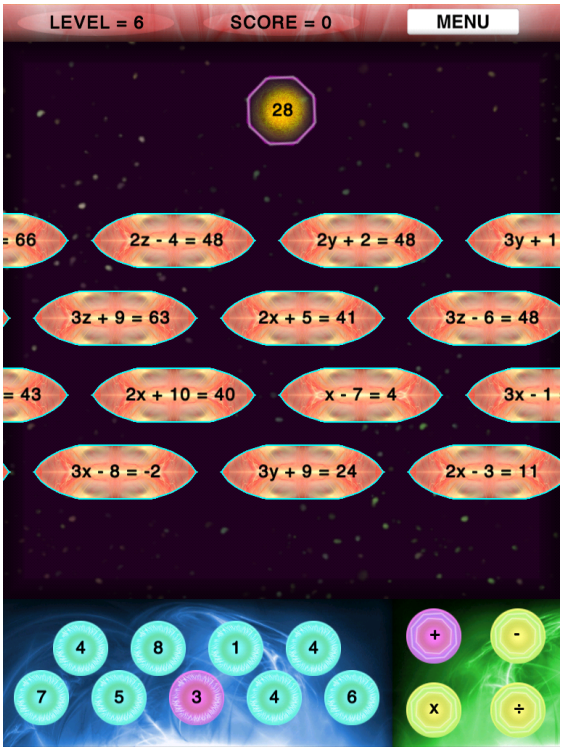

Figure 3.7: Touchscreen Game - 'Algebra Level' Screenshot

late primary or early secondary education - effectively addition, subtraction, multiplication, division and negative numbers. The touchscreen game was designed to provide players with both a motivation and an opportunity to practice, experiment and become more comfortable and familiar with these topics.

The gameplay encourages constructionist learning in that players are not asked what an answer is to a given question, but are rather tasked with creating a particular answer. There are always multiple ways to create a specific number given the numbers and operators provided to the player in the game, especially considering new numbers re-spawn as old ones are used. This allows players the freedom to come up with their own solutions in order to win the game. The game also draws on the behaviourist learning theories in its emphasis on score - with the potential for high scores being used as a motivating factor and the fact that players continually loose points over time acting to discouraging inactivity or hesitation.

At the same time, the score players receive at the end of the level is dependent on

both how many enemies they destroyed and how quickly they did so. This encourages players to not only destroy additional enemies (effectively, generating more specific numbers and getting more mathematical practice), but also to find more 'optimum' solutions to generating these numbers- for example, to generate a large number it is generally quicker to use multiplication once rather than adding several small numbers together)

As the game was designed around generating specific numbers, it lent itself very well to extension in terms of the mathematical content covered. The core gameplay mechanics still revolve primarily around basic number theory, but those numbers can be used to tie into other areas of math such as algebra or even other areas of number theory such as powers. New concepts could be covered by simply changing the format of the enemy values, without requiring any major changes to the gameplay.

3.6 Design of the Accelerometer Game

The second game developed was designed around an accelerometer control scheme. The accelerometer game was designed to be simpler and faster paced than the touch screen game and to place the player in a more reactionary position. As mentioned previously, the accelerometer generally offers less precise control than the touchscreen, but it can be easier to pick up quickly on account of its simplicity (just tilt the device in a specific direction). Development on the accelerometer game only began towards the end of the project, so it was decided to develop only 3 levels, each dealing with a different mathematical area.

3.6.1 Game Mechanics

The screen is broken into separate regions during the accelerometer game, in similar vain to the touchsreen game. At the top of the screen is the menu bar which allows players to exit to the main menu at any time during a game, displays the current level and displays the player's current score. On the bottom right hand side of the screen there is a slider which can be used to adjust the games speed. The bottom of the screen is taken up by a triangular pivot, under which is a value which changes as the game progresses and two inequality signs.

Depending on the level, various values fall from the top of the screen and the player is tasked to make them fall on the correct side of the pivot value. If they are less than or equal to the pivot value, they should fall on the left and if they are greater than the pivot value then they should fall on the right. Players can exert control on the falling values by tilting the device left or right and the degree to which they tilt the device determines the angle to which the values fall (they fall approximately towards the 'ground', so tilting the device at 45 degrees leads to them falling diagonally). The sooner a player can figure out which side of the pivot a value should fall, the gentler they can afford to rotate the device to achieve that outcome. Rotating more gently lessens the potentially negative effect the device's current tilt will have on the next falling value when it appears(new values are timed to appear as soon as the last one reaches the bottom of the screen to encourage this).

Players receive a set amount of points each time they successfully manage to drop a value on the correct side of the pivot. The amount of points they receive is determined jointly by the current level and the speed of the game. The speed of the game can be increased or decreased in real time by using the touch screen slider on the right. This allows the same game to be slowed down to accommodate younger or weaker learners, or to be sped up to provide more of a challenge for older or more advanced learners. The slider was added to the game not only to enable it to appeal to a wider audience, but also to allow players to continuously challenge themselves as they got better at a given level. The faster the game is running, the higher the potential score. As in the touchscreen game, a player's score is reduced steadily over time to give the game a sense of urgency and push players to increase the game's speed. Players also loose points if they drop a value on the wrong side of the pivot point and this is done more to discourage players from simply guessing all of the answers rather than to punish players who make mistakes.

The values falling from the top of the screen are pseudo-randomly generated, so each level provides the player with a slightly different challenge every time it's played. The replayability this offered was necessary not only to allow the game to be used as a practice/study aid, but also because the levels tend to be quite short once a player masters the mathematical content enough to increase the speed. The pivot point values are also pseudo-randomly generated abd the value changes to a newly generated number regularly. After a player drops a value on the correct side of the pivot for a

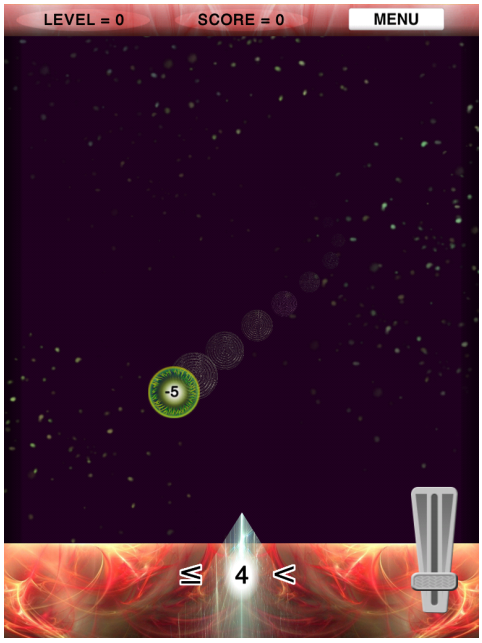

Figure 3.8: Accelerometer Game - 'Basic Numbers Level' Screenshot

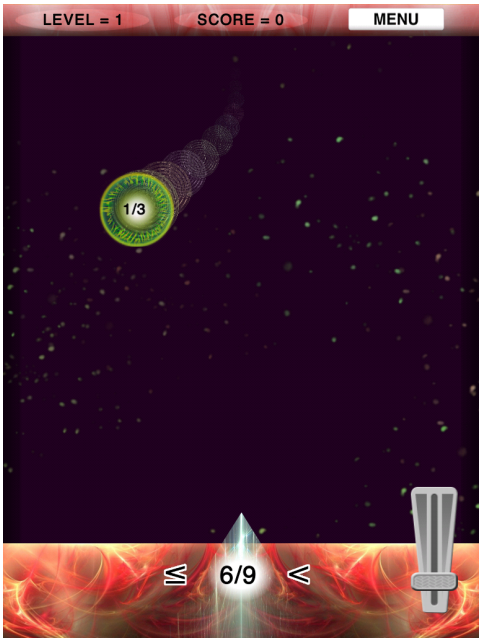

Figure 3.9: Accelerometer Game - 'Fractions Level' Screenshot

third time, the pivot value is reset (along with the counter for the next reset). This was necessary not only to make the game more dynamic and interesting, but also as a counter to poorly selected pivot values by the pseudo-random generator. The pivot values need to be random in order to allow replayability, but if they are too high or too low in comparison to the falling values, then the game becomes too easy.

As with the touchscreen game, the final design of the accelerometer game was chosen as it leaves a lot of room open for future expansion. Multiple falling values could be dropped at the same time, additional pivots could be added (e.g. _ <2< _ <5< _) and special types of falling values could be added to the game such as bombs that need to be destroyed, or values which fell at different speeds. These potential features weren't developed due to the projects time constraints, but they would be relatively simple to add and would provide a variety of different experiences and challenges for users.

Basic Numbers Level

This level, as seen in Figure 3.8, provides a basic introduction to the game. The falling values range between -10 and 10, as do the potential pivot values. Despite the

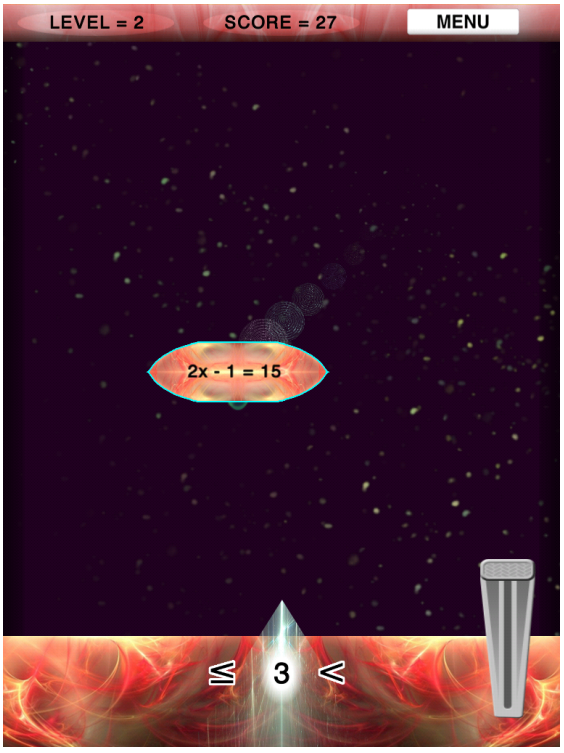

Figure 3.10: Accelerometer Game - 'Algebra Level' Screenshot

relatively low rage of numbers, this level can prove quite challenging as players are forced to evaluate inequalities between positive and negative numbers. This game also acts as an introduction to the accelerometer game, much as the practice level does for the touchscreen game. It was decided that the accelerometer game didn't require a separate practice level as the simplicity of its mechanics and control scheme made it very easy to pick up quickly.

Fractions Level

In this level, as seen in Figure 3.9, both the pivot value and the falling values are fractions. While the range of fractions is kept quite small (nothing smaller than tenths), the fractions are purposely not simplified down into their least complex form. The game aims to help players grasp the relative values of fractions when compared to other fractions and their complex forms.

Algebra Level

In this level, as seen in Figure 3.10, the pivot value reverts back to a simple integer, but the falling values are now simple algebraic equations. Players must drop the falling algebraic equations on the correct side of the pivot point corresponding to the value of the unknown variable. This level moves at a slower pace than the others to reflect the increased difficulty most players have when solving algebra mentally.

3.6.2 Educational Content and Methodology

The educational focus of the accelerometer game is primarily concerned with inequalities. Two of the levels focus on the inequalities of whole numbers, one of which obfuscates the numbers within algebraic equations, while the third level focuses on fractional inequalities. Players must master not only standard inequalities, but also those between positive and negative numbers.

The focus of the accelerometer game was more heavily weighted towards gameplay as opposed to education, in an effort to make the game faster paced. The game expects players to already have at least a rudimentary understanding of inequalities and provides them with an enjoyable practice tool to improve and test their understanding. The game can be extremely fast paced, so audio and visual cues are used to keep the player constantly aware of how well, or poorly, they are doing.

As with the touchscreen game, the accelerometer game draws on the behaviourist learning theories in its emphasis on score - with the potential for high scores being used as a motivating factor and the fact that players continually loose points over time acting to discouraging inactivity or hesitation. The speed toggle also controls a score multiplier, so players are constantly encouraged to push their abilities to get higher scores.

3.7 An Agile Approach

Modern game development is an extremely difficult and complex task - even when the game in question is ‘only’ a mobile application. While developing a game requires a huge quantity of complex programming be completed, it also requires that a large amount of time be put into a wide variety of other tasks such as art asset creation,

sound editing and level designing. Furthermore, unlike many other programming tasks, games aren't complete when they simply 'work' or 'perform to specification'. Once a game is playable, it has to be thoroughly playtested - not only to look for unforeseen bugs and errors, but also to determine what is and isn't 'fun'. This playtesting almost inevitably leads to numerous changes and redesigns, which themselves must then be playtested - making the entire process not only extremely time consuming, but also difficult to accurately schedule and plan for.

The project's nature added even more time constraints than usual to the games' development, as additional time needed to be allocated for extensive testing and analysis once the games were complete. Normally, this wouldn't represent too much of an issue, but only a single iPad device was secured for testing purposes. Having only a single device meant that testing had to take place on an individual basis. The time of year also complicated the testing phase of the project, with the project running primarily through summer - a time when both primary and secondary schools were closed. This prevented the testing from occurring efficiently in a centralised location and instead required that the test subjects be sought out and travelled to individually.

In order to adhere to these time constraints, along with the inherent difficulties involved in scheduling the development of a modern game, it was decided to adopt an agile working methodology inspired loosely on SCRUM[40]. The time available to the project split from the outset between dedicated development time and time reserved for testing and evaluating the games developed. The development time was then split into weekly SCRUM 'sprints', with the goal being that there would be a new, working version of the code that could be playtested at the end of each sprint. The feedback from these playtests would be addressed in the subsequent sprints and new features were prioritised and added according to their ranking on informal weekly task lists.

Chapter 4

Implementation

The previous chapter explained some of the major design decisions made during the course of this project and this chapter will describe the architecture of the resulting application. The design of the view based state machine, which acts as a central control mechanism for the application, will be described first. Then the resource management system and the effects it had on the application's memory management will be discussed. The particle engine, which was developed in an effort to provide a degree of visual polish to the games while reducing the number of high level art assets needed, will also be discussed. Finally, the implementation of each of the two games, and their various components, will be described in detail.

4.1 A View Based Application

The application developed for this project, like many modern games, consists of a number of clearly differentiated 'screens', or 'views'. These include; the main menu, from which players can select which level they wish to start; the tutorial screen, which explains the basic game mechanics to players; and the gamescreens themselves, in which the player can actually play the various levels of the game.

Each of these screens is considered a separate game state, as they each represent very different forms of player interaction. The functionality for each of these screens is encapsulated in separate classes, though they all inherit a basic set of functionality from a 'gamestate' base class, as seen in Figure 4.1. This gamestate class is an abstract

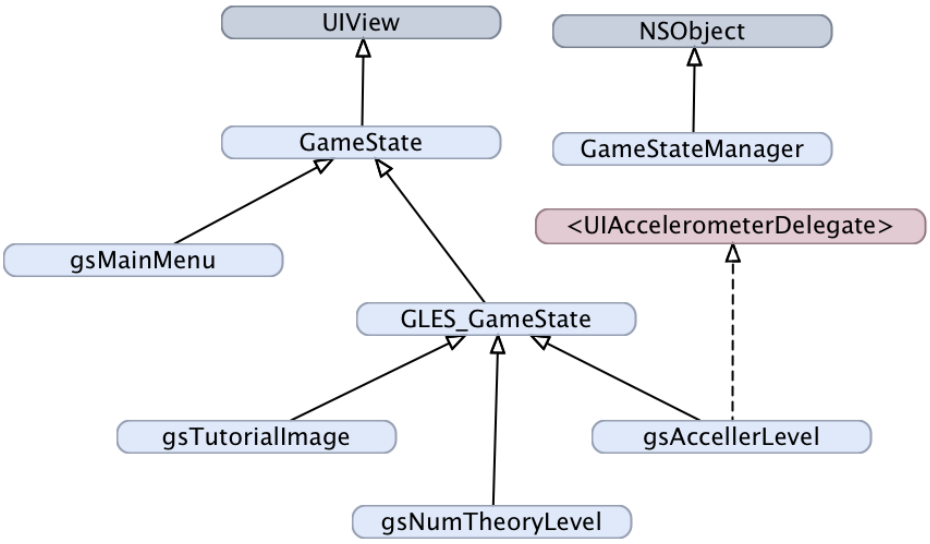

Figure 4.1: Class Diagram of the Framework's View Based Architecture

class, and contains separate update and render methods, which each subclass must implement. These gamestate classes must also implement a 'changeState' method, which releases any system resources they may be using and allows for another gamestate to take over.

These various gamestates are all controlled and managed by a GameStateManager class. This class is essentially a state machine, which controls what gamestate is active at any given time. The active gamestate is sent any relevant input from the user and also controls what is displayed on the screen. This system works quite well on modern mobile devices, where a single application usually controls the full screen and almost all of the systems resources (as opposed to a desktop environment, where there can be multiple applications open at any given time in separate 'windows').

As shown in Figure 4.1, the application consists of a number of separate potential states, or views, each of which is a separate class and which make considerable use of inheritance.

Figure 4.2: Main Menu Screenshot

4.1.1 UIView

UIView is a class from Apple's UIKit framework which defines a rectangular area on the screen and the interfaces for managing content in that area. It is designed to handle the rendering and interaction of any content in its area at runtime. While the UIView class provides only the most basic behaviour itself, it was designed to be subclassed, acting as a basic building block for visual applications.

4.1.2 GameState

The GameState class inherits directly from the UIView class. The GameState class isn't intended to be instantiated, but instead adds a number of necessary functions that all subsequent subclasses must have to enable useful polymorphism. These functions include; Render(), which serves to unify each subclasses drawing function under a single name; Update(), which takes in an increment value called 'delta' and does performs the same funtion for the subclasses' updating functions; and init(), which provides an initialisation function that takes in a pointer to the GameScreenManager class.

4.1.3 gsMainMenu

The gsMainMenu class was designed to act as the central hub for both game applications. To simplify the development process, both games were integrated into a single application and the main menu grants players instant access to any level, from either game, as well as access to each of the games' tutorial screens. Most of the menu's functionality was developed using the iOS platforms interface builder, a visual software development application that enables the creation of interfaces through a graphical user interface. While the functionality behind the menu's various buttons needed to be added programmatically, using interface builder enabled the design and layout of the menu to be developed very quickly. A screenshot of the games main menu can be seen in Figure 4.2.

4.1.4 GLES_GameState

The GLES_GameState class inherits from the GameState class described earlier and adds the functionality necessary to setup and support rendering through openGLES. It adds, and implements, functions such as swapbuffers() and bindlayer() which are commonly used operations for an openGLES render step. It also adds the class method setup2D(), which sets up all of the various environmental functions needed for openGLES (such as defining the camera and blend functions). The GLES_GameState class is not intended to be instantiated, but it adds a lot of extremely useful functionality which is inherited by all of the gamescreens which use openGLES for rendering.

4.1.5 gsTutorialImage

The gsTutorialImage class represents a very simple gameState whose function is to display information about a games' basic mechanic. This information is stored as a PNG image, which takes up the whole screen whenever this gamestate is active. The only functionality the gamestate provides is to change transition back to the mainMenu state whenever the screen receives a touch event. There are two instances of this class in the application, one for each game. The touchscreen game's tutorial can be seen in Figure 4.3 and a accelerometer game's tutorial can be seen in Figure 4.4.

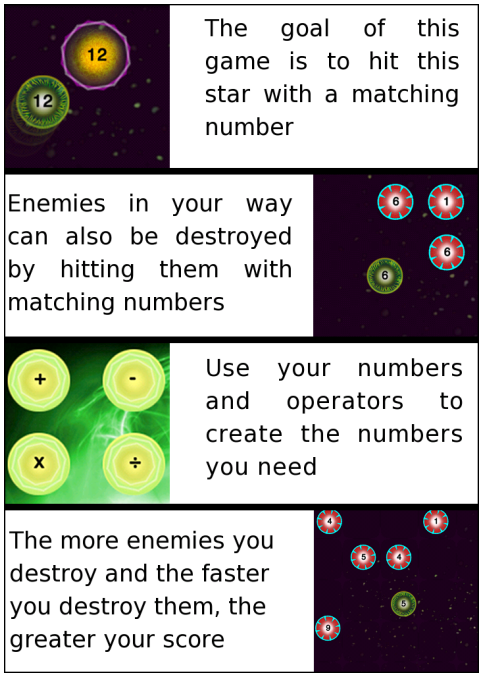

Figure 4.3: Touchscreen Tutorial Screenshot

Figure 4.4: Accelerometer Tutorial Screenshot

4.1.6 gsNumberTheoryLevel

The gsNumberTheoryLevel class represents the primary gamestate for the touchscreen game. It is far larger and more complex than the classes previously discussed as it contains most of the overarching game logic for the touchscreen game. It fully implements the render and update functionality required by the Gamestate class and has a number of its own functions for initialising the game and updating its internal logic. It also implements a number of advanced functions to handle the various touch events which can be generated by player input.

4.1.7 gsAccellerLevel

The gsAccellerLevel class represents the primary gamestate for the accelerometer game. It is quite similar in layout and functionality to the gsNumberTheoryLevel class, although slightly less complex on account of the accelerometer games' relative simplicity when compared to the touchscreen game. This class also inherits from the UIAccelerometerDelegate class, as this is required for it to receive input from the accelerom-

eter.

4.2 The Resource Manager

As was previously explained, the iOS platform provides a huge amount of functionality through its various native libraries, but this functionality is provided at a much lower level than optimal for use in games development. It was clearly necessary to have access to higher level libraries and APIs which built upon the low level functionality provided by these core libraries when developing the projects' two games. In order to reduce development time, it was decided to adapt those readily available from other sources rather than develop them from scratch.

It was decided that the three types of higher level functionality need were the ability to play music and sound effects, the ability to load and display textures and the ability to load a font and display text. Fortunately, this functionality is commonly needed in the development of advanced iOS applications and as such Apple have provided basic coding samples online which contain most of the required functionality. In 2008 Apple provided the development community with a glTexture class, which contained higher level functions for loading and displaying textures using OpenGLES, and a SoundEngine class, which provided higher level functionality for playing sound effects and music using OpenAL. Samples of these classes, along with a glFont class which makes use of the glTexture class in order to render text, are freely distributed for use in the development of iOS applications from 'iPhone Game Development', by Paul Zirkle and Joe Hogue [41].

The functionality provided by these classes was combined into a unified resource manager. This not only made it simpler for the various GameStates to access the higher level functionality, but it also simplified the process of memory management. The Resource Manager was designed a singleton class, so there is only ever one instantiation of it - with subsequent attempts to initialise it resulting in the return of a pointer to the currently initialised version. This allowed the resource manager to manage the devices limited memory resources independently of whichever gamestate was active. Additional functionality was developed to supplement the resource manager as it became necessary, or convenient, during the development process, such as functions to generate specific types of random numbers and functions for advanced mathematical operations such as

the dot and cross product.

4.3 Particle Engines

Players expect a certain minimum standard of visual fidelity in modern games, even mobile games, and this presented a potential problem for the project as there were no additional funds available for the outsourcing of high quality artwork. The solution to this problem, aside from choosing an aesthetically simple visual design, was to utilise particle engines to provide a lot of the games' visual attraction and polish.

The particle engine developed was designed to offer a large degree of variation and options, so that it could fulfil a number of separate roles. These options included the ability to set a particles colour, size, and transparency at both the beginning and end of its lifespan (with the values changing linearly as the particle aged) as well as a particle's starting position, rotation speed and velocity. Particles can also be made from any texture and can use a variety of different blending functions.

One issue with using particle engines is that they can end up using a lot of the systems resources unless they are properly calibrated - especially on a mobile device where there is less processing power available in general. To help alleviate this issue, the particle engine was designed to use point sprites (essentially hardware accelerated billboards that can be textured) rather than regular textures, as they are considerably faster to render. In addition, the rendering is performed through the use of vertex arrays and colour arrays, which allow the rendering of every particle in a given particle engine to be performed using a single call to the OpenGLES API (an extremely expensive action on some iOS devices).

In the accelerometer game, particles are used to provide a tail behind the falling values and to draw the players eye when the pivot value is changed. In the touchscreen game they are used to provide a ripple effect when an enemy is destroyed and an explosion when the player succesfuly reaches the star at the top of the screen. A particle engine also produces the dynamically moving starfield background found in each game.

4.4 The Touchscreen Game

As was previously explained, the touchscreen game is primarily controlled by the gsNumberTheoryLevel class. The functionality of the class can be roughly split into 4 areas - initialisation, updating, rendering and handling touch events (player input).

When the GameStateManager initialises the class, several data structures are created to contain the various game objects used by the class, such as the enemies and player values. Depending on the current level (which is stored as a value in the resource manager and set whenever a player selects a level from the main menu), numbers, operators and enemies are initialised with a variety of potential values. The exact values are randomly generated to provide replayability, but they are generated from within set ranges depending on the level. Four mutable arrays are used to store these game objects; one for the enemy objects; one for the operator objects; one for the number objects; and also one for projectile objects - which is a separate type of gameobject that the player's numbers are turned into once they are dragged, or flicked, up onto the gamefield. Memory is also reserved for the levels various particle engines.

The class' update method is quite compact, since each game object and particle engine has its own update function and handles its own behaviour. In addition, the players input is handled separately by the touch event functions, which are only called when such an event occurs. In addition to calling the update functions of various game objects, the main update function is primarily responsible for removing projectiles that have left the game board and for handling the games collision detection.

Each enemy has its own collision detection function, which takes in a pointer to a projectile, and is responsible for destroying both itself and that projectile if necessary. Since there are generally only a few player projectiles active at any one time, the complexity of the game's collision detection checks are quite simple. The individual enemy objects are grouped into 'enemy rows', which helps to further simplify collision detection complexity by reducing the amount of overall checks needed - further checks for a given projectile aren't necessary once a collision has occurred.

The game objects in the game were designed to all be circles, partially in an effort to simplify any collision detection algorithms, and the game's collision checks were originally accomplished through simple sphere-sphere intersection checks (one of the simplest types of intersection checks to implement). This proved to be insufficient

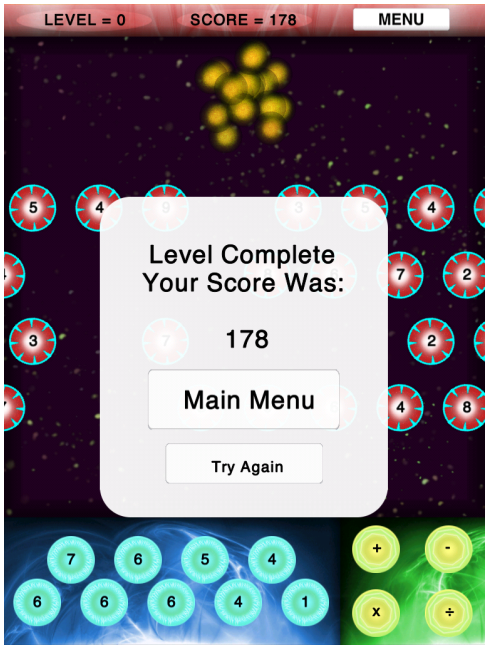

Figure 4.5: Victory Screen Screenshot

however, as specifically targeted testing showed that the game was susceptible to the classic 'bullet through paper' issue - it was possible to 'jump' through entire rows by moving too fast for the updates time step to account for. Adopting a variable time step would have proven too complicated given the development time available, not to mention posing a potential system resource drain, so it was decided to instead adopt a swept sphere approach to collision detection. This involved checking for collisions through time by recording the previous position of each projectile and changing the sphere-sphere collisions checks to line segment-sphere intersection tests.

The gsNumberTheoryLevel class' render function is responsible for rendering the entire game using OpenGLES. This task was simplified considerably by each game object having its own render function. The main render function does contain the code required to render the game's backgrounds however, and the menu bar at the top of the screen. There is also a separate sub-function designed to render a 'victory' screen (see Figure 4.5), for when the player won the game. Whether or not this function is called is controlled by a global 'endGame' variable which is set to true by the destruction of the star at the top of the screen.

The iOS platform supports three basic touch events; touchesBegan, touchesMoved and touchesEnded. The gsNumberTheoryLevel contains handles for each of these events.

The touches began event is triggered when a new touch action is registered by the device (i.e. when the player first touches the screen) and the relevant event handler contains the code necessary to either re-select an active projectile that was previously released, or to select a new number or operator from those available to the player. A state machine is used to keep track of what effect such a selection should have - for example selecting a new number when another number and an operator have already been selected will result in that operator's operation being carried out and a new random number being spawned for the players use. A touchesBegan event can also signify that the player has hit the menu button at the top of the screen and wishes to leave the level.

The touchesMoved and touchesEnded events are both used to manage the player's active projectiles. Each touch event has a unique hash value, which remains the same until the touch ends (the player removes that finger from the game). When a player drags a number or projectile around the screen, that number is paired with the touch's hash value using a dictionary data structure and the projectile's position is updated to match the player's finger's position on the screen. When the touch event is ended, that projectile is released and continues on its current course, allowing the player to 'flick' projectiles, rather than having to always drag them.

4.5 The Accelerometer Game

The implementation of the accelerometer game is intentionally very similar to that of the touchscreen game. As was previously explained, the accelerometer game is primarily controlled by the gsAccellerLevel class and it reuses a lot of the game objects and functionality that was originally developed for the touchscreen game. As with the touchscreen game, the functionality of the gsAccellerLevel class can be roughly split into 4 areas - initialisation, updating, rendering and handling touch events (player input).

When the GameStateManager initialises the class, a mutable array is set up to store the falling values - which are themselves initialised with values according to the

current level. The accelerometer update interval is also set, as are a number of other variables and functions - such as the timer which regularly reduces the players score. The main update function decodes the accelerometer input and then calls the update functions for the various game objects. This accelerometer input is then passed to each active falling value, in order to update its new position relative to the angle the device is currently tilted at. The update function also contains the collision response code, which determines when a value reaches the bottom of the screen, and is responsible for incrementing and decrementing the score and initialising a change in the pivot current value.

As with the touchscreen game, the gsAccellerLevel class' render function is responsible for rendering the entire game using OpenGLES - a process simplified by the fact that each game object has its own render function which can be called upon. The main render function renders the game's background and menu bar, as well as the speed slider found on the right of the screen. There is also a separate sub-function designed to render a 'victory' screen, for when the player has won the game. Whether or not this function is called is controlled by a global 'endGame' variable which is set to tru once the last falling value has reached the bottom of the screen.

While the accelerometer game, as its name suggests, was designed to be controlled primarily by the accelerometer, it does make use of the touch screen for a few types of input. The touchesBegan event is used to allow the player to select the menu button at the top of the screen and to select the speed slider on the right, which grants them control of the game's speed and the associated score multiplier. The 'touchesMoved' event is used to track the movement of the speed slider and update the games current speed accordingly.

Chapter 5

Analysis

This chapter outlines the approach taken in evaluating the two game applications created during this project and the reasoning behind it. The methods of data collection and analysis are described and an overview of the results is given.

5.1 Evaluation Methodology

The game application was evaluated with the aim of assessing its overall quality as well as its effectiveness as an educational tool. With this in mind, feedback was sought from two sources; children of the games' target age and an expert in the field of mathematics education.

As the game applications were designed to provide students with an interesting and engaging way to interact with mathematical topics, rather than to explicitly train them to pass some form of exam, the data collected from both sets of sources was of a qualitative form.

5.2 Expert Review

The educational expert who evaluated the two game applications was Elizabeth Oldham, from the School of Education in Trinity College Dublin. She has been involved in mathematics education for over 40 years and is extremely familiar with not only the relevant educational methodologies and learning theories, but also with existing

multimedia educational tools and games.

5.3 Testing on Students

5.3.1 Sample Selection

An opportunistic sample of children meeting the age criteria was selected to take part in the testing. This sample consisted of six children between the ages of eleven and fifteen, comprising of three boys and three girls. All of the children had started second level education and were either doing higher level mathematics or were in classes not yet been segregated according to their level. Between them, the children all attended different schools.

Ideally a larger selection of students would have been selected; however due to the external time restrictions placed on the project (as explained in Section 3.7), the testing period took place between mid July and mid August - a time when secondary schools were closed for summer holidays. This prevented the more extensive testing which could theoretically have been achieved by approaching a secondary school directly and getting the support and permission of the relevant board of directors, teachers and parents. Testing was also restricted by the fact that only a single iPad device was available for testing purposes.

5.3.2 Ethics

Permission was sought from each participant's parent or guardian before they took part in the testing. It was explained to both the children and their parents that their participation was completely voluntary and that any results would remain anonymous. To avoid any unnecessary pressure, participants were informed that they could stop at any time if they wanted to at the beginning of each session.

5.3.3 Session Procedure

Each participant was given access to the game application in a session generally lasting slightly over an hour (the length of the session being determined by how long it took the participant to finish every level in both games at least once). The game application

was installed on an iPad device which was taken to the most convenient private location available for the participants - usually their own homes. All of the participants completed their sessions individually.

During the session, participants were encouraged to play through the game on their own, with assistance only being given when absolutely necessary. The aim of each session was to play through the levels of each game in a set order (roughly according to each level's respective difficulty).

Before the participants were given access to either game, they were asked to fill out a short questionnaire designed to ascertain their own opinions of mathematics and their confidence in their own abilities with various mathematical topics covered by the games (see Appendix). Once the session was finished, participants were asked to fill out an almost identical questionnaire, the results from which could later be compared with the original questionnaire to ascertain whether or not playing the game had affected the participants opinions or confidences.

The after-session questionnaire also contained a number of additional questions designed to gauge the participants' opinions of the games themselves and to determine their overall quality and usability. This after-session questionnaire was also accompanied by a brief interview, where more probing questions were asked, with the aim of more thoroughly exploring the feedback provided by the participants thus far.

5.4 Expert Feedback

The feedback received from the educational expert regarding the educational merit of the two games was extremely positive. She felt that there was value to presenting educational material in the context of a game as certain game elements, such as high scores and gradually increasing challenge, would provide additional motivation for learners to improve their abilities. She also pointed out that the games would also benefit from their own 'novelty' factor, as learners are not used to explicitly interacting with educational topics such as mathematics within the context of a game.

The touchscreen game was regarded as being the stronger of the two games educationally, both due to its design and the depth to which it enabled the exploration of the mathematical topics it covered. The educational expert liked the constructivist design philosophy found in the game and felt that the way it allowed, and encouraged, players

to experiment with numbers and operators would be of great educational benefit to them.

It was noted that the accelerometer game followed more of a 'drill and test approach' to education than the touchscreen game, in that the game mechanics focussed more on getting the player to supply the correct answers than it did on getting the player to explore the various mathematical topics. The expert pointed out that there was still a place for such an approach in modern education and that the accelerometer game did provide an example of it that was far more enjoyable and novel than most.

The expert also commented that there was a lot of potential for growth in the game applications, especially the touchscreen game, both in terms of the breadth of mathematical topics covered and in the design of the individual levels themselves. The expert stated that she could see both games fitting in quite well with existing mathematical curriculum and that they seemed to be of more educational benefit to players than many of the edutainment games currently available.

5.5 Student Feedback

In this section the feedback received from the students who tested the game application will be summarised and discussed.

5.5.1 Changes in Opinions

As was previously explained, all participants filled out a questionnaire (see Appendix) both before and after playing through the two games and the results of these two questionnaires were compared in an effort to determine what effect, if any, playing through the games had on the participants opinions about maths education and their own mathematical abilities. Among other things, the questionnaires attempted to gauge the participants' confidences at various mathematical topics covered by the games (such as fractions, multiplication and algebra), by asking them how strongly they agreed or disagreed with statements like "I am comfortable working with division". The pre and post session questionnaires also contained more general statements such as "I am good at maths", "I need to practice maths more" and "educational games can aid learning". Any significant changes in the participants' reactions to these statements before and

after playing through the two games clearly shows that their opinions and confidences have been influenced by their experience with the games.

Significant changes were observed in the opinions of all of the participants who took part in the study. Three participants became noticeably more confident in their own ability to work with squares, while one participant was less confident afterwards. Playing the game affected the confidences of all the participants in algebra, with half becoming less confident and half becoming more confident (one became far more confident than before). Two participants displayed less confidence in their own abilities to work with division after playing through the games. The confidence of three participants in regards to working with fractions and working with multiplication also changed after playing through the games - with one participant growing more confident and two growing less confident.

How the confidences of the various participants were affected after playing is less important than the fact that playing the game did affect them. The myriad of changes in opinion after playing through the two games shows that the games enabled the participants to engage with the mathematical topics in a meaningful way.

After playing through the two games, the overall mathematical confidence of two of the participants increased (shown by their feelings towards the statements "I am good at maths" and "maths is difficult"). Two of the participants also felt that educational games were more effective after playing through the two games (as shown by their feelings towards the statement "educational games can help learning")

5.5.2 General Feedback

All of the participants stated that they enjoyed playing both of the games, with half preferring the touchscreen game and half preferring the accelerometer game. Several of the participants clearly recognised the educational value of the games, expressing statements like "I learned a lot playing through that level", and they seemed pleased with the fact that the game allowed them to learn in an enjoyable fashion. All of the participants also stated that they would like the games, or other games like them, to be integrated into traditional maths classes at their schools - or even used for homework. They pointed out that 'practicing with the game was a lot more fun than just doing sums from the maths book over and over again'.

While all of the participants found at least some of the levels difficult due to their mathematical content, they all found the games themselves easy to interact with and play - though many of them needed to play through a level before fully grasping the game mechanics. Several participants, especially those who didn't normally play games, noted that they liked how 'simple' it was to do things in the games. When questioned further about this, they said that they found it very easy to 'just touch and drag things or tilt the iPad'.

All but one of the participants found the scoring system very motivating - making statements like "I wanted to get the best score" and "I liked getting points for getting things right". The participant who wasn't motivated by the score system was the first one to be tested and she stated that she hadn't noticed it until late into the game (The score system is explained in the tutorial screen, but that hadn't been available in the version of the game tested by this participant).

5.5.3 Touchscreen Game Feedback

Half of the participants preferred the touchscreen game to the accelerometer game and this half comprised of the older participants (they were also generally the mathematically stronger participants). The reasons given for preferring it were generally that the game afforded them more time to think and plan things out than the accelerometer game. Most of the participants said they found at least some of the touchscreen levels challenging, and that they enjoyed the challenge. While most of the participants needed to play through the practice level at least once before they grasped the basic game mechanics, they all found the 'controls' very intuitive - several participants commented on how easy it was to just touch something and drag it to where they wanted it.

Most of the negative feedback aimed at the touchscreen game was in relation to the mathematics found in certain levels. Several of the younger participants thought the algebra and fractions level were very difficult, purely on the basis of their mathematical content. All of the participants also found the 'No-Addition Level' (where the addition operator had been removed) quite difficult, which may highlight an over-dependence on the addition operator. Most of the participants had difficulties grasping some of the concepts found in the fractions level - specifically, they didn't realise that fractions

could be made by dividing whole numbers by each other (e.g. 1 divided by 3 is 1/3). Some of the participants also found it difficult to grasp the basic game mechanics from the tutorial screen alone - several of them opted to play through the practice level twice to make sure they fully understood how the game was played.

5.5.4 Accelerometer Game Feedback

Half of the participants preferred the accelerometer game to the touchscreen game and this half comprised of the younger participants. The accelerometer game was praised for its fast pace and 'fun' gameplay. Many of the participants found controlling the game through the accelerometer by tilting the device to be a very novel experience (having little or no previous experience with accelerometer devices), though they generally picked up the basic game mechanics within seconds.

As with the touchscreen game, most of the negative feedback aimed at the accelerometer game was in relation to the mathematics found in the various levels. Many of the older participants found the basic numbers level to be too easy (though they still seemed to enjoy the gameplay). Conversely, most of the younger participants found the algebra level, and in some cases the fractions level, to be too difficult.

5.6 Discussion of Feedback

The feedback received from the educational expert was primarily positive. She saw immediate educational value to the games, as well as additional future potential if they were to be further developed. She praised the educational merit of the touchscreen game in particular, primarily for its constructivist approach to learning, and stated that both games could be integrated well into the current mathematics curriculum.

The feedback received from the participants in the user study was also extremely positive. All of the participants enjoyed playing the games and generally found them very easy to interact with. The changes in participants' opinions with regards to their own mathematical abilities show that the game enabled them to interact with the mathematical topics in a meaningful way. Several participants stated that they 'learned something' while playing through the games, further highlighting the games' educational merit.

Most of the criticism directed towards the games was concerned with the mathematical content of specific levels. Some of the participants, especially the younger ones, felt that the algebra and fraction levels in particular were "very hard". Some of the participants also had difficulty understanding how to play at first, which highlights the need for a more comprehensive tutorial. The main cause of these criticisms can most likely be attributed to a lack of age appropriate gameplay testing before the user study was carried out. Any child who had previously playtested the games couldn't be used in the study without invalidating some of the results, so unforunately the majority of the playtesting that occurred during development was undertaken by volunteers outside of the target age group.

Chapter 6

Conclusions

The previous chapter explained how the games were evaluated and outlined the results from that evaluation. This chapter draws some conclusions from those results and highlights some criticisms of the project. The future work planned for the project is then detailed.

6.1 Results

6.1.1 Was a Useful Educational Game Developed?

The only way to properly demonstrate the educational potential of modern mobile games was to successfully develop and test one, and this project accomplished that. Both of the games developed during the course of this project were shown to be enjoyable and to have educational merit.

The participants in the explorative case study all enjoyed the experience and while they were split as to which of the two games was the most enjoyable, both of the games were received positively. The accelerometer games was praised for its fast pace, while the touchscreen game was appreciated for the challenge it provided.

The educational expert felt that both games had educational merit, especially the touchscreen game, and the feedback and results recieved from the user study supported this opinion. Playing through the games significantly altered the opinions and confidences of all of the participants and many of the participants claimed to have learned something though their interactions with the two games.

The participants all expressed a desire for such games to be integrated into their existing mathematics classes - either to be used in class or for homework and revision purposes. The participants preferred interacting with the various mathematical concepts within the context of the game to the more traditional option of doing sums out on paper and many claimed that the gaming elements like the score and audio cues encouraged them to 'work harder' and to 'get better'. The educational expert pointed out that games like those developed could be integrated quite well within the existing mathematics curriculum and would offer tangible benefits to learners.

6.1.2 The Strengths of Modern Mobile Games

The results and feedback received from the explorative case study, as well as the researcher's own observations during the development of the games, indicate that modern mobile devices offer a number of strengths to the development and deployment of educational games.

On a technical level, the newer generations of mobile devices boast graphical and processing capabilites almost on par with those found in dedicated mobile gaming consoles. These increased technical capabilities, coupled with the impressive software development tools and supporting libraries now available to mobile developers, have allowed for the development of far more complex and interesting mobile games (both educational and entertainment based). As a result of these advances, the mobile games now found on generic mobile devices such as smartphones and tablets rival those found on dedicated mobile gaming consoles such as the Nintendo DS and PSP.

The interface methods available on the mobile platforms may prove to be of particular value to educational games. As the feedback from the case study showed, all of the participants found it extremely easy to interact with the games and come to grips with the game mechanics - even those who generally don't play computer games. This can be partially attributed to the input devices used - pointing at and dragging objects by touching the screen, or even just tilting the device, represent far more intuitive actions than pressing keys on a keyboard or pushing buttons on a controller. As educational games are often aimed at very young students, who may not be very familiar with computer games, interfaces that are easier to intuitively grasp and pick up quickly are obviously of great benefit - the less mental effort expanded on understanding a control

scheme, the more available to focus on the main educational objectives of the game. Touchscreens are also better suited to dynamically selecting objects on a screen compared to a traditional gaming controller, and this is a very commonly desired action in educational games (as seen when choosing numbers and operators in the touchscreen game). While the evidence gathered doesn't conclusively prove the advantages of the mobile input methods, it is extremely suggestive.

The current realities of the mobile market could also prove to suit the development and deployment of educational games quite well. As was previously touched on in Section 2.4 and Section 3.1 of this report, the costs involved in developing a mobile game are orders of magnitude lower than those involved in developing a traditional console or PC title. Mobile development is traditionally performed in small teams and over a short timescale, and this project demonstrated that a single individual can develop a mobile game from scratch in only a few months - in stark contrast to the multi-year development cycles, with team sizes in the hundreds, often required for console and PC development. This lower cost model clearly benefits educational game development, as the levels of investment available to educational games has generally been far lower than that available to entertainment games, since educational games haven't traditionally sold very well. It is worth noting that while 'casual' PC games can also be produced at low costs, they currently lack the centralised online storefronts which make the mobile platforms a viable market.

Mobile applications are generally purchased on an online marketplace, which reduces the need for a publisher and increases the percentage profit share received by the developers. The marketplace model also seems to encourage smaller, lower priced games, which could also favor educational game development as consumers are more likely to take a chance on an unknown property when it costs 99c than they are if it costs 50 Euros. The smaller size and scope of mobile applications would also enable the development of more tightly focused educational games, which could focus on a single educational topic (like multiplication for example) for the entire game, and such games could prove very useful in supplementing existing educational lessons.

6.2 Criticisms

6.2.1 Testing Sample

As was mentioned in the previous chapter, a larger sample size would ideally have been used in the case study. The results taken from such a small sample of children are not conclusive, though the fact that they all came from different schools increases the strength of the sample selection. The close correlation in the feedback gathered amongst the various participants also lends to the strengths of the results. Ideally the game would have also been examined by a larger number of educational experts. In both of these cases, the main factor preventing the use of larger sample sizes was time.

The project was completed within very strict time constraints and the games could only be properly evaluated towards the end of the project, once they had reached a suitable level of development. The evaluation stage of the project had to be completed during July and August, a time when schools were closed for their summer holidays. This, coupled with the fact that only a single iPad device was available for the evaluation, resulted in each participant having to be evaluated individually, usually in their own homes. The individual sessions were also quite time consuming, usually lasting over an hour.

6.2.2 Lack of Gameplay Testing

While the development process of the game involved regular gameplay testing, little of this involved children of the target age group. Any child who tested early builds of the game couldn't take part in the case study, as they could bias the results. As such, most of the gameplay testing involved older children, or the researchers colleagues, and this resulted in the mathematical content found in several of the levels being 'too hard' for the younger participants in the case study.

6.2.3 Scope of the Mathematics

Another potential limitation of this project was the scope of the mathematics covered. The mathematical topics were chosen in part because they were well suited to inclusion in a game context (as shown by the researchers previous work [3]). This doesn't neces-

sarily mean that other topics, both mathematical and otherwise, would also translate well into some form of educational game. By the same token however, some educational topics might be better suited to being used in a game.

6.2.4 Development Scope

One clear limitation of the games developed as part of this project was the fact that they weren't developed fully to the point of release. While it was felt that they were developed sufficiently to investigate the research question, the games are relatively feature poor in comparison to most commercial games - especially in relation to the number of separate levels developed. The main obstacle to further development was of course time - an estimated two months of additional development time would be required to make the games 'feature rich' enough for commercial release, and the project needed to be completed within strict time constraints.

6.3 Future Work

6.3.1 Expand the Existing Game Applications

One obvious area of future work would be to further expand the games developed thus far. The designs of both games allowed for a number of additional features and levels to be added if time allowed for it. Developing both games into fully featured products would prove the potential of the full designs, and it would also allow for more data to be gathered through a longer series of case studies (reflecting the expanded games' increased length).

6.3.2 'App Store' Release

Another potential area of future work, especially if the games were expanded with additional features, would be to release them on the Apple 'App Store'. An in-game questionnaire could be integrated into both games, along with a number of other metric tracking functions, in order to provide feedback from anyone who downloaded and played them. While the feedback gathered from such a method probably wouldn't be as rich as that gathered from a personal interview, the potential sample size would

be enormous - most likely in the thousands. Unfortunately, the process of getting an application certified and released onto the 'App Store' can take several weeks, so it wasn't feasible within the time restrictions imposed on this project.

6.3.3 Larger Scale Case Study

Regardless of whether the games were expanded or released on the 'App Store', there would be a lot of value in conducting a larger scale case study based around the two game applications - possibly in partnership with a school, or group of schools. The larger sample size could provide much stronger results as to the educational value of the two games.

Appendix A

Pre Gameplay Questionnaire

Please consider these statements

I am comfortable working with Squares	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Addition	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Algebra	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am good at maths	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Division	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I find subtraction more difficult than addition	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Fractions	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I find maths difficult	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Multiplication	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I need to practice maths more	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Subtraction	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
Educational games can help learning	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*

Post Gameplay Questionnaire

Please consider these statements

need to practice maths more	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Squares	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Addition	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Algebra	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am good at maths	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Division	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I find subtraction more difficult than addition	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Fractions	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I find maths difficult	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Multiplication	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I need to practice maths more	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I am comfortable working with Subtraction	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
Educational games can help learning	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*

I enjoyed Playing the game	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
I think Playing this game could improve my maths skills	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*

It was easy to learn HOW to play the touch screen levels	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*
It was easy to learn HOW to play the accelerometer (tilt) levels	*Strongly Disagree*	*Disagree*	*Neutral*	*Agree*	*Strongly Agree*

Which type of game was more fun? Touch Screen Game / Accelerometer Game

Why? _____

I found the overall Level of maths in these games was:	Too Easy	Easy	Neutral	Hard	Too Hard

Touch Screen Levels

I found the basic numbers level:	Too Easy	Easy	Neutral	Hard	Too Hard
I found the negative numbers level	Too Easy	Easy	Neutral	Hard	Too Hard
I found the level without addition:	Too Easy	Easy	Neutral	Hard	Too Hard
I found the fractions level:	Too Easy	Easy	Neutral	Hard	Too Hard
I found the algebra level:	Too Easy	Easy	Neutral	Hard	Too Hard

Accelerometer (tilt) Levels

I found the basic numbers level:	Too Easy	Easy	Neutral	Hard	Too Hard
I found the fractions level:	Too Easy	Easy	Neutral	Hard	Too Hard
I found the algebra level	Too Easy	Easy	Neutral	Hard	Too Hard

What part of the game was the Hardest? _____

Why? _____

What part of the game was the Easiest _____

Why? _____

Bibliography

[1] S. Higgins, "The logical zoombinis," *Teaching thinking*, vol. 1, no. 1, 2000.

[2] K. Inkpen, K. Booth, S. Gribble, and M. Klawe, "Give and take: Children collaborating on one computer," in *Conference companion on Human factors in computing systems*, pp. 258–259, ACM, 1995.

[3] D. O'Byrne, "Mathplay - an educaional computer game," *Trinity College Dublin*, 2010.

[4] D. O'Byrne, "Mathplay - lessons learned in the development of an educational computer game," *Irish Symposium on Game Based Learning*, 2011.

[5] S. Egenfeldt-Nielsen, "Beyond edutainment exploring the educational potential of computer games," *future*, 2005.

[6] Z. Okan, "Edutainment: is learning at risk?," *British Journal of Educational Technology*, vol. 34, no. 3, pp. 255–264, 2003.

[7] http://www.theesa.com/facts/index.asp, "Entertainment software association," *Last Visited 15.08.2011.*

[8] J. Kirriemuir and A. McFarlane, "Literature review in games and learning," 2004.

[9] R. Garris, R. Ahlers, and J. Driskell, "Games, motivation, and learning: A research and practice model," *Simulation & gaming*, vol. 33, no. 4, p. 441, 2002.

[10] J. Houghton and P. Sheehan, "A primer on the knowledge economy," 2000.

[11] O. Indicators, "Education at a glance," 2009.

[12] M. Schwebel and J. Raph, *Piaget in the Classroom*. Taylor & Francis, 1973.

[13] M. Prensky, "Digital natives, digital immigrants part 1," *On the horizon*, vol. 9, no. 5, pp. 1–6, 2001.

[14] M. Oliver, "An introduction to the evaluation of learning technology," *Educational Technology & Society*, vol. 3, no. 4, pp. 20–30, 2000.

[15] R. Ryan and E. Deci, "When rewards compete with nature: The undermining of intrinsic motivation and self-regulation.," 2000.

[16] J. Cameron, W. Pierce, K. Banko, and A. Gear, "Achievement-based rewards and intrinsic motivation: A test of cognitive mediators.," *Journal of Educational Psychology*, vol. 97, no. 4, p. 641, 2005.

[17] J. Sweller, J. Van Merrienboer, and F. Paas, "Cognitive architecture and instructional design," *Educational psychology review*, vol. 10, no. 3, pp. 251–296, 1998.

[18] A. Paivio, *Mental representations: A dual coding approach*. Oxford University Press, USA, 1990.

[19] R. Mayer, "Multimedia learning," *Psychology of Learning and Motivation*, vol. 41, pp. 85–139, 2002.

[20] J. Piaget and B. Inhelder, *The child's conception of space*. Psychology Press, 1998.

[21] S. Papert, *Mindstorms: Children, computers, and powerful ideas*. Da Capo Press, 1993.

[22] A. Palincsar, "Social constructivist perspectives on teaching and learning," *Annual review of psychology*, vol. 49, no. 1, pp. 345–375, 1998.

[23] M. Minsky and S. Papert, *Artificial intelligence: Progress report*. Massachusetts Institute of Technology, 1972.

[24] J. Byers, "Biological effects of locomotor play: getting into shape or something more specific," *Animal play: evolutionary, comparative and ecological perspectives*, pp. 205–220, 1998.

[25] B. Bettelheim, "The importance of play," *The Atlantic*, vol. 259, no. 3, pp. 35–46, 1987.

[26] J. Huizinga, *Homo Ludens: A study of the play element in culture*. J. & J. Harper Editions, 1970.

[27] K. Salen and E. Zimmerman, *Rules of play: Game design fundamentals*. The MIT Press, 2004.

[28] E. Medhus, *Raising Children Who Think for Themselves*. Atria Books/Beyond Words, 2001.

[29] E. Page and R. Smith, "Introduction to military training simulation: a guide for discrete event simulationists," in *Proceedings of the 30th conference on Winter simulation*, pp. 53–60, IEEE Computer Society Press, 1998.

[30] M. Prensky, "True believers: Digital game-based learning in the military," *Digital game-based learning*, 2001.

[31] R. Sandford, M. Ulicsak, K. Facer, and T. Rudd, "Teaching with games," *Computer Education-Stafford-Computer Education Group*, vol. 112, p. 12, 2006.

[32] E. Soloway, "No one is making money in educational software," *Communications of the ACM*, vol. 41, no. 2, pp. 11–15, 1998.

[33] PriceWaterhouseCoopers, *Global Entertainment and Media Outlook: 2007-2011*. PricewaterhouseCoopers, 2007.

[34] C. Crawford, *The art of computer game design*. Osborne/McGraw-Hill Berkley, 1984.

[35] http://www.apple.com/pr/products/ipodhistory/, "Apple press info," *Last Visited 9.07.2011*.

[36] Gartner, "Forecast : Mobile communications devices by open operating system, 2007-2014," *Gartner Inc.*, August 2010.

[37] IHS_Inc., "Apple maintains dominance of mobile application store market in 2010," *http://press.ihs.com/press-release/product-design-supply-chain/apple-maintains-dominance-mobile-application-store-market-*, Website last visited on August 28th 2011.

[38] Unity_Technologies, "Unity game engine," *http://unity3d.com/*, Website last visited on August 28th 2011.

[39] Epic_Games_Inc., "Unreal development kit," *http://www.udk.com/mobile*, Website last visited on August 28th 2011.

[40] K. Schwaber, *Agile project management with Scrum*, vol. 7. Microsoft Press Redmond (Washington), 2004.

[41] P. Zirkle and J. Hogue, *IPhone Game Development: Developing 2D & 3D Games in Objective-C*. Oreilly & Associates Inc, 2009.