

Making Games with LEGO:

Exploring the Uses of an Active LEGO Baseplate for Building Computer Games as a Learning Activity

Candidate: Oliver Peter Moran
Candidate Number: 9724192

MA in Interactive Media,
University of Limerick

Supervisors: Prof. Liam Bannon, Mr Enda O'Donoghue

Submitted to the University of Limerick, September, 2002

Declaration:

Making Games with LEGO: Exploring the Uses of an Active LEGO Baseplate for Building Computer Games as a Learning Activity

Supervisors: Prof. Liam Bannon, Mr Enda O'Donoghue

This project is presented in partial fulfilment of the requirements for the degree of Master of Arts in Interactive Media. It is entirely my own work and has not been submitted to any other university or higher education institution, or for any other academic award in this university. Where use has been made of the work of other people it has been fully acknowledged and fully referenced.

Signature:

Oliver Peter Moran
13 September 2002

For Dad, he taught me to build.

Acknowledgements

Before anyone else I must extend a very special thank-you to Krispin Leydon for his more than generous help and patience. Without his know-how and willingness to talk ideas through – even when I didn't listen and had to learn the hard way – this project would still be a proposal. Likewise I would have got nowhere but for the kindness of James and Noel from the Electronic Engineering Department who slipped my orders onto their books, opened their stores to me and when no-one else could solve the frustrating 'connectors' problem, worked on a solution just because that's the type of guys they are. Thank-you all.

To my sisters, Mum and Dad for not giving up on me despite my bad temperament, and Fiona, especially, for calming me down and showing me that there is always a way out of any logical predicament.

My new friends in the iMedia class of 2002 for being so sound, it's been a great year, thanks lads. My long-standing friends, that I can never do without, for understanding and supporting me through out a year of extreme weirdness on my part, especially to Podz who knows me better than anyone else. Nice one, bud.

Finally, there are some that remain to be thanked in the IDC. First among these are Paul and Eddie, for their confidence that everything is easy in C++ and their readiness to share their know-how about APIs and libraries with any freak that might fall through their door, and my two (why am I the only one with two?) supervisors, Liam and Enda. Liam, who expressed every faith in my ability and Enda, who give me every faith in myself.

Thanks everyone.

Contents

| | | |
|----|---|----|
| 1. | Introduction and Overview | 1 |
| 2. | Comparing the Spectrum and the Console | 6 |
| 3. | Playing Games for Learning | 10 |
| 4. | Criticising Computers in the Classroom | 22 |
| 5. | Designing Computer Games as a Learning Activity | 30 |
| 6. | Designing the Active Baseplate | 40 |
| 7. | Conclusion and Further Development | 56 |
| | References and Bibliography | 59 |

Appendices:

| | | |
|----|--|----|
| A. | Moral Panic and Video Games | 69 |
| B. | Elementary Circuit Design for an Active LEGO Baseplate | 74 |
| C. | Parsing 3DS Max ASCII scene export files to OpenGL | 77 |

List of Figures

| | | |
|-----------|--|----|
| Figure 1: | Prototype Design One ('Two Legs, Two Studs 1') | 48 |
| Figure 2: | Prototype Design One(b) ('One Leg, Two Studs') | 49 |
| Figure 3: | 'Dancing Lights' - How the Baseplate Scans | 51 |
| Figure 4: | Prototype Design Two ('Two Legs, Two Studs 2') | 52 |
| Figure 5: | Prototype Design Three ('Two Legs, One Stud') | 53 |
| Figure 6: | Exploring Stacking Possibilities | 54 |
| Figure 7: | Photograph of final prototype being tested | 74 |
| Figure 8: | Elementary Circuit Design for an Active LEGO Baseplate | 76 |

Abstract

Title: Making Games with LEGO: Exploring the Uses of an Active LEGO Baseplate for Building Computer Games as a Learning Activity

Author: Oliver Peter Moran

The Active LEGO Baseplate that this report explores is a tool to build computer games as a learning experience in the classroom. It uses tokens as representations of the world, pieces of narrative element that are assembled to construct a game in a similar manner to how children partake in pretend-play.

This report examines the social and analytical aspects of its construction with a description of a prototype implementation of it. It discusses a change in relationship between game-players and the games they play over two decades with reference to the machines that they use and offers an examination of the meaning of computer games and game-play itself. This is provided as an essential background to a wider debate. Computer games, it is argued, when the player is empowered to meaningfully play them, may offer a fruitful resource of education and learning. It is put that meaningful play can only happen when the computer-game player can explore the ideas and understandings that are meaningful to her/him in games of their own.

To frame this argument in an applied sense, a criticism of the current widespread use of computers in education is made and from this an argument is put forward as to how computers and the use of computers to make games could be more properly channelled in education. It is from the findings of this debate that the heuristic principles for the design of the Active LEGO Baseplate are drawn.

Introduction and Overview

“Computer games don’t affect kids; I mean if Pac-Man affected us as kids, we’d all be running around in darkened rooms, munching magic pills and listening to repetitive electronic music.”

That statement, made by Kristian Wilson, a Nintendo executive, in 1989 (Blacklock:2001), is infamous for its irony. Popular debate is in uproar about video-games, their inherent dangers and especially the waste of time that they encourage in a child that could be put towards more productive activities (see appendix A). This project, by proposing to employ games at the very heart of institutionalised education, the primary-school classroom, is in danger of scandalising it self. In reply, it has to be stated that for the most part, the depth of criticism made by popular discourse neglects to understand games and properly ignores gamers. What is frustrating about discussing games with anyone except fellow gamers is the displacement of energy caused by having to continually refute the singular perception of games and gamers upheld by an enormous number of people. Being grounded now as it is in ‘common-sense’ and so, imperative to challenge, is devilishly difficult to do so.

Wilson’s statement is worth reappraisal on another point. He is an executive at a company that, challenged possibly only by Atari, is maybe most synonymous with video-gaming. His comment dismisses that computer-games can have any effect, beneficial or otherwise, on its audience. That it was made in 1989, the so-called ‘Summer of Love’ – the iconic height of rave culture – is unfortunate but

what is more important is that his employers, as central as they are to game-culture, manufacture only consoles. In the section that follows, I will argue that the change in game-culture that has been affected by the ascendancy of the console in part disallows the game-player to play with the game. With no other medium is the text so shielded from the audience. If I watch the news and see a story on it that rouses me, I can demonstrate (albeit accepted that few people do) and become a part of the news. If I read a book, I can write my own. Consoles lock the player out of video-games as a medium, they can less play with the messages they are receiving in the same language that it is given to them. This project attempts to return an elemental part of play that that has been lost to the bulk of gamers. It explores how we might learn from games even so far as to include them in our classrooms by examining games and the space occupied by computers in the classroom and exploring what heuristic principles may guide the inclusion of games as a classroom aid based on those discussions.

This report is made in account of a system that, no matter how small it may be as prototype, offers the potential to be developed into a grander system for constructing and exploring understanding through video-games. The accompanying implementation consists of an adapted LEGO baseplate and some accessory bricks with which the child may construct the narrative elements of a game and the spatial arrangement of these at the start of play. Like the child constructing narrative elements from their understandings into imagined play, the bricks, representing narrative tokens, are assembled into game. The baseplate reads what has been constructed on it and allows interaction between tokens to occur in a similar manner to traditional computer-games from a representation of

the construction generated by a connected computer. The tokens and world-space that they inhabit may be animated or audio queues added to it. Characters that are static on the 'real' model might walk or cameras change but, most importantly, relationships created between tokens so as to present an image of the world that the child can explore.

Though the technology to allow this was designed specifically with this project in mind, alone, or with modification and further development, it provides an opportunity to explore other research interests into interaction design, cooperative modelling and data processing. This report will limit discussion of it to only the areas that applies directly here. Though technologically a feat, it is the opinion of this author that technology is only useful in so far as it aids society. Technological endeavour is social endeavour and the worthiness of a technology can only be aptly argued in a socio-cultural/political sense. A technologist will be hindered in her/his understanding of a technology if they do not discuss its construction as a social enterprise and the physical artefact as an abstraction of cultural discourse. This report may then appear abstracted from the tangible project whereas it is, in fact, the project implementation that is an abstraction from the tangible social endeavour that makes it worthy. A later section will discuss at length its technical operation and development but immediately, and in the sections that follow, discussion of the physical artefact will argue its worth as a social technology and in terms of similar technologies that fail as social tools.

The '*Active LEGO Baseplate*' is a name adopted from an investigation into the possibility of a like-system by researchers at MIT (Borovoy and Martin:1994). It is one used here only as a robust working title and not as argued depiction of its nature. As already said, the baseplate uses tokens as representations of ideas to be explored. Depictions of imaginary interactions between characters, institutions and objects can be tested and follow rules and understandings that are to be taught. Computer games already work with rules in such a manner, some to greater affect than others, but do not as readily allow for the creation of contexts and scenarios that would allow a player to exploit as easily their potential as learning aids in this regard.

LEGO, as a constructive tool, offers great affordances that provide an intuitive inference of what constructions can be made with it. LEGO, employed to construct a video-game from narrative blocks, would allow experimentation with permutational interactions between concepts to quickly build and rebuild, in an uninterrupted manner. The child, unlike one creating a game through traditional programming interfaces utilised in the classroom (see Goldstein and Prat:2000 and Tholander:2002), would be less hindered by having to know or learn 'how-to-make-a-game' and could more readily focus her/his attention on the games s/he was making. As a learning aid there are similarities between this system and the play with narrative that research connects to the child at pretend-play (this will be discussed in more detail in the section 'Playing Games for Learning'). The kind of scripted play that an implementation of this sort would bring about is reminiscent of the kind utilised by educators teaching children with special needs, where play is targeted at precise learning activities:

“Special educators often use play intervention methods such as script rehearsal to promote young children’s pretend play abilities, because of the relationships suggested by research between enhanced play skills and enhanced cognitive, social, and language skills.” (Bergen:2001)

This report will not so quickly argue the purposefulness of the baseplate as an educational aid specifically of this sort because to do so would be beyond the learning and experience of the author. However by admission, if the tools were to be adapted as a learning aid, it is this area that he would see as being the most fruitful field of its application. Neglected also by the author is a proper discussion of the nature of video-game software as it affects (or more often fails to affect) quality representative understandings of social interactions. In concluding the report, some comments will be made on this debate but to do so throughout would distract from nature of the artefact that is being demonstrated. However, before discussing these topics we must first look at conventional video-games systems and the set of relationships and values that they encourage in their audience.

Comparing the Spectrum and the Console

In 1972, Rolling Stone magazine visited Stanford University to report on an event that it said would happen “[r]eliably at any night-time moment in North America.”

“Hundreds of computer technicians are effectively out of their bodies, locked in life-or-death space combat computer-projected onto cathode ray tube display screens, for hours at a time, ruining their eyes, numbing their fingers in frenzied mashing of control buttons, joyously slaying their friends and wasting their employers’ valuable computer time.”

The article opened with the brave prediction that, “Ready or not, computers are coming to the people.” More bravely, in the first appendix to the article, it proclaimed, “[A]ll you have to do is read a book on computer programming, and you’re an instant computer scientist.” (Brand:1972)

To understand my motivation for approaching this project as I do, it is necessary that I give a little of myself away by way of a brief personal history. My interest in computing is born out of my childhood experience of using a Sinclair Spectrum and later watching the demise of the particular culture of home-computing to which this machine belonged to, to another. The Spectrum was probably the most infamous (in Europe, at least) of an early breed of home-computers known broadly by the label, ‘the 8-bits’, that like its kin-folk the Amstrad CPC and Commodore 64, differed significantly from the way computers are used in the home today. Though a capable machine for its day, the Spectrum was largely used for entertainment but what made it significant was the relationship that was encouraged between game-players and the games they

played. Using the Spectrum (and likewise other 8-bits) encouraged experimentation and play with the technology they harboured. The machines were operated through BASIC and familiarity with the language, that is still relevant today, was an explicit expectation of their use. A clever integration of otherwise common, low-tech devices such as ordinary audio-cassettes to store data as sound by means of household tape recorders helped lower intimidation levels that technology of its sort might otherwise exude. In a quirky way users became attached to their machines, a attachment that is still obvious ten years on through the dedication of the writers and aficionados devoting appreciable effort to the maintenance of the many website that relate to all things ‘Speccy’ – a word used to describe both the Spectrum machine and the culture that it is at the core of. The intimacy that the machine inspired expressed itself in unexpected ways. Users would quickly learn to pick-out different types of data by ear as they heard it load onto the machine. An experienced user could even tell how long was left for a game to finish loading – a process that could take anything up to fifteen minutes – through familiarity and discernment. For the user of such a computer, the gap between consumer and producer of software was a small one. Both were intimate with the workings of the machine, the 8-bits facilitated home-development and the exchange of home-made games among users. To create ones’ own game was a short step and one that every gamer took at least once.

The early nineties, however, saw a sudden relocation of the manufacture of popular home-computers away from the American/British model that it had previously occupied towards domination by the Japanese and a cultural shift in gaming and popular computing that was not coincidental to it. Video consoles,

though having been popular once before, suddenly resurged and the likes of the Sega Megadrive and SuperNintendo stormed the marketplace routing even their technological equals, the '16-bits' such as the Atari ST and Commodore Amiga. Consoles differed significantly from the previous generation of home-computers by hiding the mechanism of their operation from the user. A differential tone was bred that implicitly stated that – important from a marketing perspective since, as I have just said, equals already existed in the Amiga and ST – games like these were beyond the abilities lowly gamers to make. Cartridge systems held software in read-only blocks of physical memory with the game code hard-written onto the chip in an effort to keep both the development and duplication of software under the thumb of hardware manufacturers, a debacle that is today slugging it out between Microsoft and the makers of 'mod'-chips to disable security protection devices that prevent illegal copies of games being played on propriety consoles and coincidentally locks out the 'home-brew' industry of game-makers from the consoles' innards (Edge:114). It was in the prevention of the popular distribution or manufacture of software that a barrier was erected to the ordinary understanding of what goes on beneath the hood of a gamers' machine. An interesting to note to observe, however, is that that like the Spectrum users listening to the sound of games load, a popular curiosity among PlayStation (PSone) users is to put a game CD into an ordinary stereo and browse its tracks to sometimes find disbundled audio segments from the game and amateurishly surmise that there are so many tracks to so many levels and so forth. Although blank cartridges are not unheard of, today they cost over €200 and require a PC, not the consoles for which they are designed, to employ. Strangely though, early '90s software developers appeared to rebel against the

new tone of hidden technology that was set in the ascendancy of the console, possibly because their own staff may have had their heritage in the slain 8-bit tradition. An odd convention arose whereby games were produced that would allow a gamer the option to browse the software's sound library, hearing the various crashes, beeps, cries and whistles that would be encountered as the game was played. It seemed to reflect back to an era of intimacy once more.

The triumph of the consoles cast in place a dichotomy of game and player, losing in the transition from the '80s to the '90s a sense of understanding of control and creation by the gamer that was associated with the previous generation of home-computing. The area of occupancy of game-players shifted away from familiarity and integration with their machine and the development process to one as audience as consumers.

Motivating this project is the desire to reclaim the expression of a medium annexed in the transitional period of the early nineties and return it, in some sense, to young game-players. Even if the eventual outcome is not practically comparable to my experience of the Spectrum and to the making and sharing games with it, there is a nature of that experience in my argument. What I want to revive in this project is the connection between author and game that makes game-building a worthwhile activity for learning, something the Spectrum encouraged and that contemporary game-culture inhibits.

Playing Games for Learning

Maybe because my father is a carpenter, I am one of the few people I know who never owned LEGO as a child. If I wanted to build something I could pick some pin nails from the floor of the work-shed, take a light-hammer from the kitchen drawer and construct my own little world at the back of the hot-press. With a minimum of physical queues, I could transport myself to any location and partake in an inexhaustive range of activities. I could build a ship, put-up shelves inside a travelling-shop, work open-air at the top floor of a towering skyscraper. My nails could be the size of men and my hammer suitably enormous to drive them into the hard concrete that was the air. As my needs directed, clothes pegs or the cap of an old pen could become a chisel or set of screwdrivers. Characters could be invented, from an over-bearing foreman to a kind-hearted tea lady; health inspectors, policemen, the hot-press was an abundant city of interactions. Anything I needed to explore my games, I could integrate with ingenuity. It was wonderful. What I was experiencing was a later stage of normal childhood development, pretend-play (Beudney:2002).

Pretend-play is an important activity for the child. As children engage in pretend-play, they simultaneously develop receptive and expressive language skills. They utilise mental representations of the larger social world, making sense of it and locating them-selves within it as they grow. Because of the coincidence in time between these development stages, many researchers hypothesise that pretend play and the development of these skills are related (Speech and Therapy Activities:2001 or Bergen:2001). Make-belief (as pretend-

play is called by some authors) offers a real-time, hyper-realistic expression of the world that allows children to test their understandings of it and modify them on the fly. With their imaginations they can create miniature representations of the world within which to practice and learn about social and interpersonal relationships in a virtual model that is so close to the real thing that in their minds it is indistinguishable. Children, as much as their imaginations run rampant, must construct their make-belief worlds as close to real-life as they can. It is essential that their model maps to an established understanding, for it is the understanding that is given to explore. Through their imagined interactions, they can make sense of the knowledge they are gathering, grasp its significance, enabling themselves to meaningfully partake in the social stream that they are becoming a part of.

“What is key about pretend-play is that it is rule-based. Children begin to control and guide their own actions in terms of how they think their ‘mother/fireman/Super Mario’ should behave. Paradoxically, children exert more self-control in imaginative play than they do in real-life when behaviour is less controlled and more reactive or impulsive.”
(Cunningham:2000)

The world that is explored in pretend play is near-tangible. It exists in an organic cyber-space that, though fluid and dynamic, is less than malleable. It is an expression of what the child experiences everyday. Narrative elements are connected to create a meaningful understanding of the world. As such their representation must be true-to-life. The rules of any pretend game are unwritten but implicitly understood. It is in the exploration of these understandings, and the elements that go to create them, that the benefits of pretend play lie. From conversation with those I know that did own LEGO, the worlds they describe

building were similar to ones that I remember visiting at the back of the hot-press and, later, that I would experience when creating similar worlds on the Spectrum.

LEGO, in its explicit form, provides a platform not just for, as is over-stated, the exploration of physical structure, but to create toys for oneself that facilitate pretence in play. The term explicit is used above because some researchers draw a distinction between ‘functional’ and ‘symbolic’ pretend-play (Libby et al.:1998 in Deudney:2002). Functional pretend-play is to use an object resembling its real-world reference ‘as-is’, so for example, to push a toy car along the carpet and make ‘brmmm’ noises is functional play. Symbolic pretend-play involves treating one object or situation as if it were something else, such as pretending a banana is a telephone. LEGO in its implicit form invites a child to partake in symbolic play, too. A single brick can be a car, a clean house can be dirty, the sun can shine down on LEGO-land indoors. However, essential if we are to grasp the relevance of this, what is of value in these activities is not the actual constructions, nor, necessarily, the games that are played. To a certain degree, too (‘a certain degree,’ is said because I do not want to sound too controversial), the imagination involved is not so important, for it too is used like a tool. What is key is the meaning extracted from the play, the making-up of stories, and the validating of them against what is known.

Working with children touched by autism, it is a Catch 22 for researchers to resolve whether the children they observe cannot play because they are isolated or are isolated because they cannot play (Deudney:2002). If our perception of the world is as a narrative construct (Glos and Umaschi:1997) and pretend-play

is a play with the narrative elements that construct it, then when in play we construct our perception. Without play we cannot perceive anything but ourselves.

Unsurprisingly, if pretend-play is a narrative construction, as has been argued, it is argued too that links exist between literacy and pretend-play (Wilford:2000). In this we provide for interesting thought with regard to video-games. Considering the symbolic aspect of pretend-play, Sara Wilford outlines five aspects of pretend-play that share attributes with literacy goals (adapted from Wilford:2000):

- Symbolic processes: To understand that a prop or a person can be symbolise something or someone else in a pretend-play drama underpins the realisation that a written word stands for a spoken word, and that letters, alone or in combination, can represent sounds.
- Language growth (semantic and contextual): Children, in their dramas and discussions, expand their vocabularies and elaborate on the meaning of their words and actions so as to be understood by others.
- Problem solving ability: Children engaged in building a sky-scraper or an airport work through dilemmas, or at later stages construct a game with rules, working at trial and error crucial in creative writing and the prediction and decoding necessary for tackling a challenging text.
- Motivation or disposition to persist: While the term 'practice' can imply dry and meaningless rote learning, motivation or the disposition to persist

turns practice into pleasurable work such as re-reading familiar texts as a bridge to more difficult ones.

- Joyful engagement: Our desire that children will enter all aspects of literacy, the fuel that feeds a lifelong thirst for literacy, that can be seen in children's play through the tone of their language, and the exuberance and zest that they bring to an activity.

What Wilford describes in relation to creative writing, I recall from my play with the Spectrum. The same elemental factors that allow the creation of a spaceship out of cardboard-box are present too in the making of one out of computational semantics as it is in writing a story about one. The same invention and play with narrative that would allow a child to turn a hot-press into a workshop would allow him to believe that `IF playerHasHammer == TRUE` were his tools. The pretend-play of believing that a character created on-screen is as real as one created in LEGO involves as much rulemaking, as much testing, as much learning as any other pretend-play activity. The benefits of pretend-play that Wilford describes are benefits that the making of a computer game imparts, because they are the same activity. The narrative elements, the worldly understandings that go together to make-up a computer game, are the same code that runs make-belief. Turning back onto a section of BASIC and reinventing it as another requires the same fluidity with narrative and meaning that would allow a child to upturn a table, once a stage now a ship. A mash of pixels will be an egg in the same functional sense that a doll is a baby and if `xPos` is the eggs' position on-screen then, in a symbolic play, the wind is `xPos--`.

But what happens to a game if such play with elements as these are lost, as I have argued happened in the transition from the Spectrum-like era to the consoles? Furthermore, what does it mean to this project, if it is about retrieving some of those lost elements. Working closely with a project it is often difficult to see it from the ‘outside’, the perspective that has most worth, being immersed in its inner logic and trials can spoil its sense of location and exaggerate its place among things. I am indebted in this regard to a colleague who at one of my most confused moments told me how he described my project to a stranger in a social setting: “Imagine being able to make your own level of *Quake* out of LEGO! You could build your own castle and run around inside of it firing-off guns in all directions.” While possibly not the impression that it is desired a reader take from this report, the sentiment of his description is apt. It might have been preferred if he had said to imagine knocking together a *Sim*-like city and watch it working or to reconstruct a historical battle but change some aspect of either force. The strength of this project is in the readiness of its potential to create a context in which play may be explored with more freedom video-games.

However, curiously for its detractors (and I, in some respects, am a detractor myself), playing a computer game, and being successful at it, is no easy activity. Often the logical processes required to complete a game are complex. Superficially at least, computer games require problem-solving skills; specifically the ability to solve problems posed by an unknown author and set in an arbitrary context. To be successful at a computer game, a player needs to be abreast not only with the answers to a puzzle but also the logic of the puzzle itself: why was it posed in such a way. More precisely, though I’m not arguing

that this occurs in any disciplined academic sense, the player needs to be able to understand the algorithms that go together to make the gaming environment, that is, the logic of how the game works, the limitations of hardware upon it and how this will effect game-code. The player needs to be able to apply this knowledge to discern the kinds of problems that can be put forward by a games' designers and decide how they will affect otherwise possible solutions for which there may be infinite. A good player will understand the logic of a games' code. Their game-play will be as programmer in reverse.

This understanding of an untaught science is demonstrated by application everyday. The poorest of players are those who allow themselves be in servitude to the game, thinking of it as random, organic thing that they must grapple with and keep under control. Skilful players are able to use their understanding of programming and of the capabilities of technology to win games. As an essential beginning, the most common example of this is for a player to get to grips with the AI algorithm of computer controlled enemies – what route do enemy sentries take, if I shoot and miss will he call for back-up, if back-up comes what search pattern will they use to try to find me, where is the best place for me to hid, so what is my best course of action right now? On to this again is an appreciation of technical application and uses of technologies by the programmer, marked most often by an understanding of the games' engine – how many baddies can appear on screen at once, knowing what objects are important through a discernment by experience that digital artist draw different classes of objects through different rendering techniques.

Recently some games have started to play on this astuteness on the part of the gamer, betraying, like software developers that including audio samples for the player to browse at the turn of the decade between the '80s and '90s, the line drawn in silicon between programmer and player. Spielberg's *Medal of Honour*, for example, contains a 'cheat' whereby the player can see the 3D environment as wire-frame and have enemies numbered above their heads. It is of no advantage to game-play but does intonate an acceptance that gamers understand the environments that they are playing in are little more than computer-generated boxes. Moreover, the numbers over the heads of enemies admits the limitations of the engine to displaying a maximum of five enemy sprites on screen at once, seemingly accepting that players, adjusted to the game, would have an intuitive understanding of this already.

Another important game, Hideo Kojima's *Metal Gear Solid 2*, made an even bolder admission to the arrogance of the computer industry to assume it can 'dumb-up' players. During a surreal moment in one of the later levels, game characters that had previously been carefully developed in a strongly story driven game suddenly pass startling remarks about their own non-existence except in code. They comment on the players understanding of them as being real only in so far as it is necessary to believe in a character so as to be lollod into a story and yet be aware that they are but segments of code in order that the player might read and win the game.

These admissions by game developers are part of a wider up-surge in desire to understand games and game-play in a more academic fashion (though still the

mode is rag-tailed by the larger part of academia). An academic model for understanding games by which they may be criticised in a culturally significant environment would in so doing raise the cultural value of games and the gaming industry. Although this is not an entirely new development – arguments that attempt to understand games as texts have been put forward with some degree of seriousness at least since the mid-eighties (Skirrow:1986) and have gathered momentum since – what is changing is an understanding of a need to criticise games on their own terms rather than trying to merge them with existing theoretical models, especially those of literary and film studies. The swing away from traditional half-way houses into trying to build a shelter for oneself has taken place nearly simultaneously from all interested sides – gamers, game-developers, and interested academics – possibly spurred-on by the fact that games, that once clung onto Hollywood as their most identifiable aesthetic cousin and most in awe technical inspiration, now look set to overtake that economic house later this year in terms of revenue. With academics speakers and academic concerns expressed by developers being heard with a dramatic increase in volume and numbers at the most recent Game Developers’ Conferences in the US and Europe, the movement has been shifted up a gear. Reported by Edge (Edge:114) magazine, Jesper Jull of the University of Copenhagen said, “It is becoming established that a game scholar has to be as knowledgeable about games as a movie scholar is about movies.”

The past application of a mixed theoretical tool-kit failed by drawing too much from little more than superficial similarities between media. Awkward composites between literary theory and technologies that appeared to represent

long-stated abstractions of non-linearity have led to lofty claims that few gamers would agree are representative of their experience of game-play (see Turkle:1984 in Southern:2001). By saying that in play, players write their own narrative, authors have apparently been oblivious, or hid from, the fact that games are written before controllers are taken-up. The text is pressed onto CD. As Julian Kücklick (1999) writes:

“Computer ‘gamer’ are obviously more skilful at deducting the rules of the code from the signs on the screens, and at utilizing the possibilities of manipulations that they are offered ... While Umberto Eco’s concept of the open text is probably old news to gamers, literary scholars seem to forget the achievements of literary theory in over coming the notion of an autonomous text when it comes to applying this concept to computer games.”

And later:

“While it might still make sense to compare adventure games with medieval quest narratives, or action games to certain epic genres, it would be hard to argue that Tetris is an interactive poem.”

That, however, is not to say that *Tetris* is not poetic rather as a colleague reported to James Newman (2002), “[W]hen I play *Tetris*, I am a tetraminoe[sic].” Personal experience of all manner of games substantiates matters. The relationship between the player and game is neither one of reader, participant nor author. If when playing *Tetris* were are tetraminoes, an idea that has intuitive appeal for me, then we are tetraminoes with an astute sense of self. We don’t just know that we are a block, or even what sort of block we are, but we know where we fit amid the stream of blocks that have come and those that are yet to fall. We are tetraminoes touched by a religious transcendentalism.

Peter Nuar (1985a), on a topic that a literary criticism of computer games would see as hardly related, writes that programming computer applications is a theory building activity. In the writing of a computer application, he argues, that a programmer must interpret real life concerns using what theoretical model s/he will. In the using of this application, the user will work within the solution presented by that model (he later expands on that concept to express that if other programmers are to be able to properly adapt an application to changing needs, they must be first understand the original authors model ie. the theory that s/he has stated in code). Is it too great a leap to imagine that computer games may have as much in common with their 'serious' cousins as they would with the cinematic and literary influences on them and that in playing games a player must be able to decode text not written in imagery or narrative in isolation but present in the code game play?

However, in spite of this, I argue that the type of 'fluency' (Resnick:2002) with computers gained from playing games in current circumstances is stagnant. By that I mean that for the generation of gamers that belonged to the Spectrum era, the 'fluency' they gained from playing games could be expressed creatively through making her/his own games. Since the player is now kept apart from the technology whereby games are created, they are prevented from applying, directly through more games, the benefit they get from them. The child enraptured by pretend-play can benefit by drawing their own fancy and learning into the game, however, whereas the game-player may learn the skills to decode another authors' expression, s/he cannot express her/himself through play.

Resnick's 'fluency' is present in the pretend-game, the manner in which it drifts from subject to subject, understanding to struggle, and the manner in which it manipulates itself to the subject consciousness. Contemporary computer game-play, in missing this, is a kind of illiteracy. So long as they maintain removed from the 'fluency' they offer in their design as play, game-players will be illiterate readers. If we are to 'learn' more directly from video games, not just partake in a pedagogic activity akin to television, consuming a liner stream of information, but to engage in learning, take up the elements of a text and play with them, then video games must be adapted to allow this.

Play, as discussed above, is important. That the player of a computer-game can decode the text, and that games even contain a text, is argued also. The project that this report describes attempts to return a union to the two, enabling the player to actually 'play' with a game, reorganising its narrative elements to reconstruct its text so as to express their own understanding. It is argued that just as pretend-play is a learning activity, that a system such as it can, too, be implemented as such. As a natural base for exploring if this is possible, then, as dangerous as it is to experiment with peoples' learning, it is proposed that we test if it is possible to do in the classroom. If it can be made as effective tool as others, then our exploration will have been profitable. Before we can do so, however, we must examine the territory that we are proposing to enter and the legacy that our exploration will follow in order to learn from it ourselves.

Criticising Computers in the Classroom

What I have written so far about the changing relationship between computers and computer users in gaming I sense is a part of a wider set of already altered relationships between users and technology that is to our detriment if we are to employ computing as a creative force in society. What I want to see is a more open, non-directed approach to interpretations of computing and computer use. More relevant to the terms of this project, however, is the need to discuss this drift to ineffectuality as it affects the classroom and learning. Others have written in plenty on education and technology and it is remarkable to read them sound in unison at the failure of recent application of computing in the classroom. Significantly too, in light of earlier argument on the altered state of gaming across the two decades, many writers have noted the stark difference in tone and approach between the first popular steps to educate young children and adolescence in computer science during the late-'70s and '80s, by means of packages like Logo or BASIC, and the approach taken during the so-called 'information age' boom of the '90s (Begel:1997 or Rush:2002 for example). The benefit of teaching specifically understood 'computer skills' such as MS Word sounds unlikely. The lack of a theoretical base and over-attentiveness to the technical application of a transitory technology questions the worthiness of this type of syllabus even on its own terms. MS Word is not a concept in computer science or any other discipline; will this package still be relevant when these children leave school? If it is used as a creative learning tool the problems are compounded again when on top of this are laid the observation that packages such as these are not intended to be used in this way. More often than not

software like this is designed for adults in the business place, as is the computer, the operating system, keyboard, and the mouse.

From my own experience teaching primary level, I saw educators try to integrate packages like MS Word creatively into the curriculum. Very often time on a computer was given as a reward because of the limited collaborative nature of the interface naturally restricted the number of children that could use the facilities available at any one time, and the number of children allowed to use a PC was thus curtailed by the time available to a teacher that s/he could spare from other curricular activities. Children were therefore very excitable about using a computer for schoolwork. A typical scenario for the use of a PC would be if a child had finished a reading along with which they would have had a number of spellings to learn related to the story. As a reward s/he might be given time (and assistance in the case of younger children) with Word to insert a clip-art image and write a little using the words that they had learnt relating to the story. While in theory this may sound all right, in practice it was a frustrating and disappointing exercise for the children to do and for the teacher to watch. The available clip-art never met with the children's expectations and even the most able child found it hard to browse the Word clip-art interface creatively. Inevitably, older children would come away unfulfilled from the exercise, often trying to 'have-another-go-if-they-were-quick-enough-teacher-wouldn't-notice', then astonished when their second attempt left them as empty as their first. For the younger children using the interface was a near comical exercise in concentration, diligence, skill and memory. It was rare to see a young child unsatisfied by their creation if only for the sense of accomplishment in the task,

though they were still constantly disappointed by the inappropriateness of the pictures that they found in the clip-art file.

Even where interface and applicability problems are been surmounted through dedicated learning packages new problems arise and the use of computer technology still does not encourage 'fluency' with computer science or facilitate creative learning. Another personal example from the same group of children is just as telling. The teachers I worked with told me of a number of packages that distracted children from their work by being too 'over-the-top'. I saw one of these in operation. The package was supposed to test spelling and word recognition. A picture showing a scene filled with activity would appear accompanied by a word. The children would have to point and click on the object or activity indicated by the word in the picture. If they were correct a new word would appear and so on until eventually a new picture would be displayed. Unfortunately, the animation and sound effects were so dramatic that the children even when supervised would enter random answers just to witness the packages hyperbolic responses. In order for the child to complete the tasks set by the package, strict supervision was necessary and this even this, as I have said, was often not enough to keep the child's attention focused. Eventually questions were raised as regards why such a package should be used at all when the teacher alone could achieve more beneficial results, with less hassle and more flexibility using traditional paper-copy methods.

In fact, only one computer-aided learning package was regularly used in the school. There was no creative element to this software rather it automated a

standardised spelling programme for children with reading disabilities allowing the teacher to free herself from a mundane aspect of the curriculum, giving her some time to some fulfil light bureaucratic duties while still keeping one eye on the child as the computer ran through the test in a run-of-the-mill manner. This worked well because of the automated nature of this aspect of the spelling programme (programme as in curriculum, not program as in computer program) and the fact that as a special needs teacher she would take children one-to-one rather than in groups whereby the computer facilities would either be full or a new collaborative interface required. The children also enjoyed it, as it was known before starting that there would be no creative element to it so that they were not letdown as in the case of Word. The software gave only a moderately lively response when a question was answered correctly, the interface was clearly laid out, an oversized mouse pointer could be clicked over an appropriately large radio icon if the child wanted to hear the word again, otherwise spellings were entered using the keyboard (and backspace in the case of a mistake) which all the children were very capable of doing effectively. The only occasional problem that sometimes arose was a difficulty telling I from L on the keyboard as the package wrote in all lowercase letters while the keys on the keyboard were printed in all uppercase. The small L on-screen looked like the capital I on the keyboard, however the children themselves were aware of this problem and only occasionally asked for help distinguishing the two. The children also enjoyed printing-out and saving a page that certified that they had got all their spellings right, in fact, the children that used this package were so satisfied by it that they had folders of A4 printout as testimony to their achievements.

While the latter example was successful at its purpose, it is no closer to achieving the “revolutionary” potential of the new technology that Resnick (2002) writes about, that is that computers will not be true learning aids until they are thought of less as “information machines” and more as tools to explore our knowledge creatively. Resnick (2002) argues that if computer technology is to be used beneficially in the classroom it must be seen as more kindred to open-ended learning tools like finger-paint rather than information drivers like television. To associate computers and television as similar media is understandable – both were invented during the 20th century, both employ audio/visual queues – but computers can be much more than that. Television is a closed, directed medium in the strictest sense, finger-paint is anything but. The important work in computers and education to be done in years to come will be to open computers to non-directed activities whereby children can explore a wide area of subjects and ideas gathered from instructive learning so as to be truly educated. His argument is based on the work of Piaget:

“... learning is not a simple matter of information transmission. Teachers cannot simply pour information into the heads of learners; rather, learning is an active process in which people construct new understandings of the world around them through active explorations, experimentation, discussion, and reflection. In short: people don’t get ideas; they make them.” (Resnick:2002)

In this respect, work to explore the usefulness of computer games to education has some theoretical appeal. Games like *SimCity* or *Civilisation* that present an open structure for exploring relatively complex models of urban planning, governance and political history undoubtedly offer a rich resource for discussion and learning if used appropriately and, appropriately or not, the developers of the *Sims* series, Maxis, has produced workbooks and lesson-plans to help teachers

integrate these titles into their curriculum (Squire:2002). What the games industry hope to exploit is the appeal that ‘edutainment’ has for many – kids like fun, make learning fun, kids will like learning – while avoiding the simplistic pedagogical models that were inherent to televised attempts at it. Unfortunately, as with much of the potentially rich research in this area, what study has been undertaken is instigated or sponsored by the gaming industry so it is difficult, even in the most transparent of reports, to discern real benefits from bias. One notable among such studies was undertaken by BECTa (2002), the British Educational Communications and Technology agency – a publicly minded title for an industry interest research group.

The work involved selecting eleven commercially available entertainment software titles, drawing appropriate lesson plans for each and inviting eleven schools to integrate the titles into their curriculum and report on their experiences. Unfortunately, reports were returned for only six titles and BECTa offers an explanation for only two of the missing five (“Two of the schools were unable to complete the research because of competing priorities for pupil time and problems with using the software on school computers.”)

Though a study of six teachers experiences is not generalisable it does make for interesting case study, if more beneficial for understanding criticisms of current game software when put to educational use than for what dubious benefits they may offer – among which are the most boasted ‘ICT skills’, as noted above in relation to MS Word, or ‘Thinking Skills’ that may not be as impressive as imagined. Kurt Squire (2002) explains, “A skilled *Half-Life* player might

develop skills that are useful in playing *Unreal Tournament* (a very similar game), but this does not mean that players necessarily develop generalizable ‘strategic thinking’ or ‘planning’ skills.” It cannot be said that someone playing *Civilisation* will necessarily contextualise their experience of the game with issues of governance and corruption in the 3rd World but if when playing analogies are drawn between real-life events and happenings in the game by someone acting as a teacher then maybe it would be helpful. These difficulties aside and not concerning ourselves with technological and interface issues, potentially the most significant findings of the report indicate the problematic nature of the most important perceived benefit of game software: that they are fun.

“[Teachers] acknowledged that access must be curtailed within the context of the lesson: ‘Pupils very easily lose sight of learning objectives if allowed to use [one element of the game too much], they rapidly become engrossed in the detail.’

“...Some games are difficult to adapt to the time constraints of classroom use. Most games are designed to be absorbing and offer hours of exploration and gameplay, and this can conflict with the focused use needed within lessons.”

Most teachers reported that, even if presently workable within a classroom environment, commercial games needed to be ‘slimmed-down’ if they were to be really beneficial: “Simplification was seen as having more potential than adapting games by adding to them.” Within a broader context of the educational cognitive sciences, this view makes sense. Our understanding of the underlying processes of education is very poor. Essentially, we don’t know how people learn and as such can only act on situations where we at least know that they do. New learning experiences and curricula should work around those known contexts:

“One might hope that research on the fundamentals of leaning would provide guidance for the instructional designer. However, it is also a fact that we know very little about the detailed mechanisms of learning.

“... the more we are concerned with the practical aspects of instructional design the less importance[sic] this ignorance becomes. What matters is that we have some understanding of the situations that promote effective learning even if we don't really understand what is going on in the learners head. It is the engineering rather than the science of learning that is important.” (Hammonds:1993 in Boyle:2002)

The BECTa research, far from discovering new paths to greater learning, involuntarily gives credence to this opinion. Foremost among the ‘key points’ they draw from the study is that: “The role of the teacher in structuring and framing the activity of the learner remains crucial if learning outcomes are to be achieved.” While game software may be a useful tool for education: if it ain’t broke don’t fix it. As Doug Church (cited in Squire:2002) puts it eloquently, would you want to live in a city designed by a person who has only played SimCity?

The resounding answer is ‘no-very-likely’. The utility of orthodox game software to education is deficient in the same manner that the utility of understood computer skills are lacking as learning aids. Neither is fitting to the classroom or parallel classroom activities. They distract from the driven purpose of education because they are not designed to accompany the already established manner in which understanding is communicated. In the next section we will examine how they may do so and anticipate how a system that may allow us to prevent such from happening may be designed.

Designing Computer Games as a Learning Activity

Many undertakings to introduce children to the design of games have been made and research into the usefulness of game programming to children's learning of computer science are abundant (see for example Begel:1996, Goldstein and Pratt:2000 or Tholander:2002). The introduction of Logo as a starting-point for children to learn elementary programming practice is an indisputable success where it is available but speaking to a teacher recently, albeit at secondary level, its failures too are recognisable. While teaching programming languages to children may be worthwhile from the perspective of computer science, how worthwhile it may be in practical terms is less than obvious. The teacher I spoke to worked in a largely (socially) deprived area. Her question was what was the point in teaching a programming language (she was talking about C++) when a majority of her class didn't own a computer and most didn't even know how to work their way around a desktop interface let alone common computer applications. Would it not be better to teach them practical skills such as MS Word so that when they left school they may at least have a chance of competing for the better jobs? A difficult argument to dispute in real life scenarios and not one that I would deny readily in her context.

Difficulties teaching programming languages are even more problematic at primary level, not for the ability of the students or what marketable skills they may learn from it (which are more or less irrelevant at this stage), but for the scarcity of computing resources. Where research into teaching programming at primary-level is done, it is in the majority of cases done so as an extra-curricular

activity with a handful of children. In real terms, teaching children to program is difficult for schools, either because of resistance or because of the impracticality of it to everyday classroom activities.

So what of teaching children to write games? As motioned in the last section, games can be useful, if problematic in themselves, to learning in a classroom environment. Naurs' (1985a) argument that programming, as a human activity, is a creative one where the author expresses their own theory of human endeavour, if extended to games, would mean that games also are expressive of human understanding and theory building. If taken in the context of Resnik's (2002) understanding of Piaget, where learning is not somewhere that people get ideas but make them, then making games would in theory at least, for what we understand of it, be a worthwhile learning activity. In practice this may mean something else.

If learning is an active process in which people construct an understanding of the world then the design of computer-games would allow for the creation of representations of such worldly understandings that would enable children to explore their learning. Games often, if not always, provide a depiction of a world (real or fantasy) imagined by the author that casts the player as an acting spectator able to explore her/his created understanding. The creation of a game necessitates the interrogation of given understandings of how reality is and should be perceived and the formulation of new environments, relationships and epistemologies that are meaningful for whatever reason to its creator. Further to this, the tasks set out in a game must be meaningful to that understanding. The

design of a game has within it the potential to be a useful, and enjoyable, exploration of persons' own learning experience and an expression of her/his understanding of learning.

Working with older children using computers creatively and building on an established knowledge-base whereby creative artefacts (paint, clay, dance) are used in education, Resnik and Rusk (Begel:1997) put forward six advantages that design orientated activities have for education:

- Design activities engage youth as *active participants*, giving them a greater sense of control (and responsibility) over the learning process in contrast to traditional school activities in which teachers aim to 'transmit' new information to the student.
- Design activities encourage *creative problem solving*, avoiding the right vs. wrong dichotomy prevalent in most school mathematics and science activities, suggesting instead that multiple strategies and solutions are possible.
- Design activities can facilitate *personal connections* to knowledge, since designers often develop a sense of ownership (and care) for the products (and ideas) that they design.
- Design activities are often *interdisciplinary*, bringing together concepts from the arts, mathematics, and sciences.
- Design activities promote a *sense of audience*, encouraging youth to consider how other people will use and react to the products they create.

- Design activities provide a context for *reflection and discussion*, enabling youth to gain a deeper sense of understanding of the ideas underlying hands-on activities.

However, as attractive as these benefits may sound, the difficulties involved in overcoming the practical problems posed in the classroom in order to draw them out are serious ones. As a starting point in order to develop a theory of how we may do so, it is imperative that we abandon the orthodox monitor/keyboard/mouse interface and acknowledge that computing facilities in schools are a scarce commodity that need to be open to facilitating the sharing of them among several children at the same time, something that the traditional set-up as a 'personal computer' precludes. In a large respect this requires that we neglect, in building our theory, to think of computing and computer-aided learning as related to computers, the physical boxes that we are scrambling to use, for if, when thinking about computers and how we may be able to employ their computing power as teaching aids, we do so, we will inevitably return to teaching 'computer skills' rather than wider transferable learning experiences. The transitory ICT skills that BECTa give so much significance to are of no broader benefit outside of computers themselves. It is necessary to 'forget' computers in order that we may focus on how computer-aided learning can be made applicable across the curriculum in such a way as to not simply mimic on-screen already-established techniques in those subject areas. Other disciplines can demonstrate their relevance to computer science. It is necessary that 'computers', as a subject, must forget its self in order that it might demonstrate its relevance to other subjects in return.

By removing ourselves physically from the orthodox set-up of a classroom PC but still employing its power, we can too circumvent difficulties it presents and adapt its physical structure to our needs. By way of example, for many curricular activities – art, coming to mind – children will work naturally and mutually beneficially in groups. They will often change quickly from one group to another, learning from each other, a desirable synergy *en masse* for personal development and education. If computers are to prove their worth to such an activity, they must be able to equally adapt themselves as paint and brush do to art to whatever learning context they find themselves in. The orthodox set-up precludes this. It is not mobile, it is a static crate demanding our attention, not fitting to our needs. Peripheral technologies, by name existing at the outer edge of where computers can reach, offer, I believe, a way out of this crux. In what is called peripheral technology here, is not what is meant by any input/output attachment to a computer. The keyboard, though strictly a peripheral, is not. It is commonly understood as an inclusive part of the PC. By a peripheral what is meant is a specifically designed technology that can utilise the computational resources of a computer while avoiding the necessity to remember that the physical artefact we are using relies on a device embodying, as the PC set-up does, the popular cultural understanding of what computers are and what they are used for. We need to avoid this, for if we do not, it will be a parasite to learning.

Two projects to demonstrate this are *Tangible Programming Bricks* (McNerney:2000) and *Triangles* (Gorbet and Orth:1997). Each use peripherals to the neglect of the original computer in order to employ computing technology,

though not the specific ambition of either author, so that it may be applicable across diverse disciplinary fields:

- Timothy Scott McNerney's *Tangible Programming Bricks* (2000) project developed task specific LEGO bricks representing elements of instruction that could be snapped together to be followed through as a linear rule-based logic. The bricks were developed to an extent that enabled a model railway was created with the train instructed to behave as a child determined when it encountered line-signals.
- Mathew G. Gorbeil and Maggie Orth's *Triangles* (1997) project developed a way in which elements of digital information may be associated freely, in the context of this report, most consequently to tell stories. Elements of narrative were represented by magnetised equilateral triangles that could be connected and disconnected tangibly so that in one example, if a fly were connected to a frog, the frog would eat it but if an owl was connected first then it would eat the frog and the fly would survive.

The strength that underlies these projects is that even though the technological application of each relies on a computer to enable it, the practical application is synchronous with the activity of the child. Children learning from the operation of the railway need only to worry about the instructional syntax sitting in the carriages of their model train. Likewise, the child playing with the narrative elements of frog, fly and owl is undisturbed from the abstractions of their story by a mouse or GUI. Similarly, if the construction of a game is going to be a

successful learning activity then the technological how-do must be deleted from the equation. We are not interested, as far as the child's learning is concerned, in how they actually make a game or teaching them how a 'real' computer game is made. What is important are the games that they create and what learning they express through them.

The focus of a peripheral to enable children to create games as a learning activity will then lean on the theoretical interpretation of games rather than the technical knowledge or skills required to make one. This is a central argument because it will impact directly on the motivational reason to design a system that will allow children to do so. In practical terms, it would cursorily appear impossible to build a system, no matter how well designed, that would allow children to create any kind of game, expressing any kind of learning without such a system being so complex as to require substantial learning aside from the what it would be supposed to impart just to use it. What's more, following a postmodern criticism, any system designed, being by Naurs' (1985a) argument, a theory, will impose its own epistemology on the child's learning. The only solution to this is to accept it, for it is not unwelcome.

The two points raised in the last paragraph are intertwined and complementary. They are that any useable peripheral of the kind we are discussing will focus only on the building of a small number, if not just one, type of game (before it will become a system so large so as to require learning itself) and that inherent to the games that it will create will be a fixed epistemological expression. Though this may appear far removed from Resnik's (2002) vision of computers as finger-

paint, it may not be. Education from a sociological perspective, rather than a free exploration of ideas, is the induction of the next generation into already established knowledge-bases (Bilton et al.:1996). Educational practice is not innocent; it is the teaching of children to think like everyone else. Even if in Western liberal societies this includes teaching children to question established orders and to ‘think for themselves’, it is because that is our established order – something that can be traced to the ‘barbarian’ Germanic and Celtic traditions that were freed after the fall of Rome (Strauss:2001) and that have educated generation after generation in Europe, and later America, since. Finger-paint with its liberal and yet functionalist bent, may be more epistemologically loaded than at first perceived.

A peripheral for children to make games as a part of a learning experience, one that will practically limit the extent and meaning of games that it is possible to make with it, is then as practical and as beneficial as the breadth of epistemology it allows the user to play with. In simple terms, such a peripheral will design games of a certain kind to teach a certain lesson.

Imagine a hypothetical system whereby by any game to express anything could be made and the mode of operating this system so intuitive that learning to use it would not impose on the educational focus. Such a system already exists, by analog, in pencil and paper (after an initial mastery of the art) but rarely do we hear teachers instruct their class to write about anything they like, except when the point of the lesson is to learn the tools that they are using. If such a system

(one for making games) existed, and, as above, it did not require any learning to use, its usefulness as an educational tool would require a teacher to set children the task of exploring specified understandings ('Write about Winter', 'Write a story about going to the zoo') and for her/him to grade them on their expressed understandings ('Why was your character wearing shorts in Winter?', 'There were no planes landing at any zoo I've ever visited!'). Such a system would, however, as discussed above appear in practical terms to be forever hypothetical. Undoubtedly, what is developed would require some degree of learning to operate but the balance of an open system, capable of creating an unfettered range of games, against the need to keep the focus of learning on the games created, not the operating of the system, is vital. In fact a more closed system may be more desirable.

The practical problems indicated by teachers, the ones I have spoken to and the ones partaking in the BECTa study – that computer games, if they are to be used in education, should be 'slimmed-down' and made more fitting to classroom use – from a design perspective might support more closed, lesson specific uses for a system. Constraints on classroom time and the need to focus classroom attention on one aspect of learning, even if not palatable from some theoretical perspectives, are real concerns. A closed system, one that focused game-making at learning one lesson or one discipline, might be more robust and beneficial at what it does than a more open, but less focused systems, where time and concentration may lapse. This, however, is not something that can be surmised on paper, it will be necessary to observe such systems in actual use and probably,

even then, I believe, the difference between closed and open systems (while personally leaning towards the former) would depend on what lesson the system would be supposed to impart. Such observations, if they are to take place, will, without meaning to state the obvious, necessitate the construction of some artefact to test. Using the arguments raised in this discussion, the construction of the Active Baseplate, a system that proposes to do what is discussed here, is reviewed in the section to follow.

Designing the Active Baseplate

The design and construction of hardware for this prototype and the software that accompanies it took-place simultaneously and the choice to employ the Active Baseplate to create a 3D arcade-type game, an intuitive decision from the start. In hindsight, this decision appears to have been the most natural and dramatic demonstration of one of the baseplates' potential uses. In being able to create a tangible inanimate world of tokens in real-space that is then represented as an animated, intractable world on-screen, the dynamic nature of a users' understanding of the tokens conceptual meaning will, it is anticipated, be apparent to the onlooker. Though the games that can be created using the software (see the section 'Conclusion and Further Development') and tokens provided in this prototype are very limited, and what learning can be drawn from it, pedestrian, as a demonstration is serves its purpose.

The last section discussed some theoretical considerations to heuristics if creating a system of this nature and purpose, based in a large part on section that preceded it where criticisms were made of the way computers are used in the classroom. The issues raised in these provide a focus for discussion of this project and an argument for its actual implementation. Summarised, the lessons drawn from their argument are that if a project of this nature is to be successful as a learning aid some aspects unique to the design and implementation of a project of this kind should be considered. If it is to be worthwhile in a classroom environment and be suitable across the curriculum, rather than teach specifically understood 'computer' skills, the following must be borne when designing it:

The system should:

- Value the role of the teacher, not compete with her/him for the child's attention, since the teacher is the already established route through which learning in the classroom is mediated and transferred.
- Work alongside established curricular activities in a physical and pedagogical sense rather than replace them or take away from them since these longer-standing traditions have already proved their usefulness.
- Focus learning on a set learning activity for each session so as to respect the fact that classroom time is limited and focused subject by subject concentration is standard educational practice.

In each of the preceding points, it should be noted that we are not trying to revolutionise education but provide an accompanying tool to it.

- Allow an intuitive understanding of its operation in order that it will not become the focus of the learning activity itself in conflict with the lesson that is supposed to teach. Worse again, if the technical implementation of the system is too great a burden for a teacher to utilise, it will be redundant.
- Be adaptable to cooperative use as may be necessary for some learning activities. In nearly all cases this may involve the co-operation of the teacher, but so too will many applications of the tool involve a number of children using the same artefact at one. The

developed system should respect children's co-operative use and the tied resources of computers in the classroom.

- A custom (peripheral) device is preferable to an adaption of the orthodox GUI, as traditional PC interface, in common practice, conflicts with the principles above, be it to a lesser or greater degree in each case.

Having been drawn from previous sections there is no need to argue the logic and worth of these principles any further but rather to apply them to a design and implementation. None the less, in addition to these, a final broader consideration was put onto the project's design: the implemented artefact must be suitable to the everyday physical environment of the classroom, filled as it is with children, limited in space, and so forth. It was this consideration that eventually necessitated the adaption of a LEGO baseplate over technological alternatives.

Working these considerations into an initial concept design to allow freestanding tokens to be placed unrestrained on a set area of table or a special mat was problematic. A starting point was to invent a working model of the game to be created using whatever technical system was chosen. This game was a tangible variation on the long established 'crate and maze'-type. In these games, the player must push, but may not pull, crates onto designated hotspots in order to advance to a higher level. Their ability to do so is hindered by awkwardly placed walls through which neither they nor a crate may pass. The objective of each level is then to solve the process by which each crate can be placed onto a hotspot without getting it trapped against a wall or another crate, the solutions to

which get progressively more difficult with each successive level. This type of game often comes with a level editor so that the users can create their own puzzles, so, initially, it offered an already established tradition within which to work and adequate space within which to build the concepts on which an implementation could be based. As work progressed, however, this game was dropped to one more meaningful to the project. That occurred in an accidental manner while shopping for LEGO.

The *Bob the Builder* branded DUPLO sets that I was using to prototype the final baseplate design came with a brief suggestion for a pretend-play game to be made with each construction. For example, one accompanying a set that contained a model of Bob and bricks depicting a ‘men-at-work’ sign, tools and paving-stones read:

“Bob is busy fixing a hole in the road. He will need his toolkit, drill and a road sign. Can you help him?”

Another containing Wendy and bricks showing wallpaper, a brush and a bucket of glue, suggested:

*“Bob has tried to put up new wallpaper, but he’s made a bit of a mess!
Can you help Wendy fix it?”*

A game that followed these lines of constructionist pretend-play would allow for a more accurate demonstration of the motivating principles of the project and

was eventually what was settled on. Using these bricks, a re-association of them created another game, one where we help Bob paint a wall. Though mundane against standard computer-games, the illustrative connection between the two play activities – one with DUPLO and one with the Active Baseplate – are analogous. The built computer-game presents a similar connection between bricks as did the original DUPLO games. Both embody the ideas of narrative, understanding and play that earlier sections of this report discussed and would allow a demonstration of the baseplate to show the potential uses of it if further tokens were included. However, satisfied as I was with what game could be constructed using the conceptual model I had, the technical doing of its application necessitated the design of a technology that would enable its benefits of to be realised in the context I had demanded of the project.

Some initial ideas for how to do so were obviously impractical if consideration was to be given to the classroom environment. One was to employ an already established system (in the university department) to locate an object on a 2D plane using a web-cam housed beneath a see-through tabletop onto which tokens emitting an infrared signal are placed. The system, in principle, appeared technically able to distinguish one token from another, to tell its locations on the plane and, moreover, if the base of each token consisted of two infra-red ‘legs’ (one ‘left leg’, one ‘right leg’) that cast distinct identifying signals then it would be possible to tell a tokens’ orientation by comparing where each ‘leg’ was ‘standing’ in relation to the other. This method, however, was dismissed. The awkwardness that a unit of this type would present in physical terms, necessarily having to be the size, at least, of a bedside locker, did not convey the same sense

of flexibility that was asked for. A dedicated unit of this type was not thought to be welcome in a classroom. Some safety consideration too ruled it out, as it would have required a glass or Perspex top. But more again, the likely need to periodically recalibrate the camera located inside the unit and to keep the top clear (in both a transparent and physical sense) made it impractical to everyday use.

A similar concept was to use image recognition software to ‘watch’ the construction of a scene in real-time using cameras from above. This method has been utilised successfully by others researching cooperative modelling projects (see Fraunhofer:2002 or Bruns et al.:1999) but was ruled-out too as being impractical for this project. It was proposed to distinguish tokens from each other through brightly coloured strips. Two different colours could be used to distinguish different ‘shoulders’ in a similar way to a tokens’ ‘legs’ above, but again issues pointing to the classroom environment discounted it. The size and nature of an installation of this kind would mean in real application that a teacher would have to specifically, time-consumingly and laboriously set the equipment up each time it was to be used. The learning activity would, thus, be distracted and not exist in parallel with other curricular activities. The likely problem, too, of arms, heads and other limbs blocking the cameras’ view of tokens was also considered, and influential in this idea being dropped as a possible implementation.

The difficulties in implementing the project raised by the examples above contributed to the desire to explore electronic solutions. An initial attempt at

such blended the two styles of answer. One implementation considered employing a set-up that consisted of a flatbed of light-sensitive resistors analysed to source-out the presence of ‘foot-prints’ made by specially shaped tokens standing on top of it. The prospects of actually implementing this solution were unlikely because of the difficulties posed if we were to distinguish shapes, even the crudest of ones, with as much accuracy as would have been necessary and unlikely using a bed with as low a resolution as we had planned it would have been possible to make. However, it was an important concept as the matrix formation of the photo-resistors led to the consideration of a LEGO-ised solution.

Adapting this and previous ideas to a LEGO-ised form, it was proposed to use infrared receivers imbedded into each LEGO stud and blinking infra-red LEDs housed in each block. An LED would communicate to the stud over which it was placed telling it what brick was above it and the stud would in turn communicate this information back to the computer. The software would thus be informed of the position of each token present on the board, its orientation could be deducted through distinct left and right ‘legs’, as above, and as many different tokens would be possible as IR signals could be conceived.

Such a device would be small and robust enough to endure classroom life. Its LEGO-ised form, while a compromise on a unfetter arrangement of tokens, was in itself advantageous because of the affordance it offered to construction. Co-operative work with an active baseplate would be unhindered, restricted only by its size and location in the classroom. It did not hint towards any lesson nor take-away from the role of the teacher. No complex setting-up would be necessary

that might distract from curricular activities. Despite these however, the actual implementation of this solution was never worked-out because a more simplified adaption of it offered more attractive technical attributes while keeping the same practical outcome.

Using the Pico device (a digital voltmeter that connects to a PC), an implementation could be built whereby each stud consisted of semicircular live and ground connectors set at a 45° angle. If a resistor was placed between the connectors to create a circuit then the value of this resistor could be read by sensing the potential difference across the connection. A ‘two legs’ system, like the ones above, could tell orientation so that the position, type and orientation of a brick could be determined (see figure 1). The significant advantages of a system of this sort against the previous one would be that the time needed to scan a board would be greatly decreased. If using the LEGO-ised infrared-LED method whatever device would have been used to scan the base would have had to pause on each stud as it scanned the board in order to give adequate duration for the LED above to communicate its ID. The discernment of what resistor was creating a circuit using this method would be as quick as the technology (chips, Pico and PC) allowed. Welcome too was the opportunity to drop the need to provide power to the bricks (in order that the LEDs may blink) either through on-board batteries or through smart power-giving connections. Like the LEGO-ised infrared-LED method, this solution satisfied all the requirements of the design principles being used.

Prototype Design One (Two Legs, Two Studs 01):

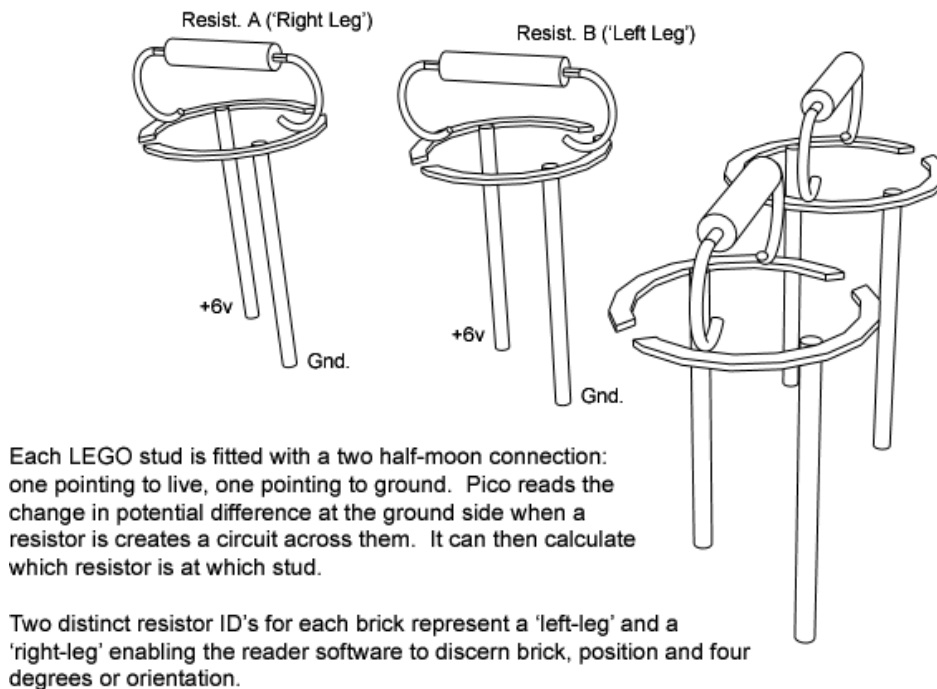


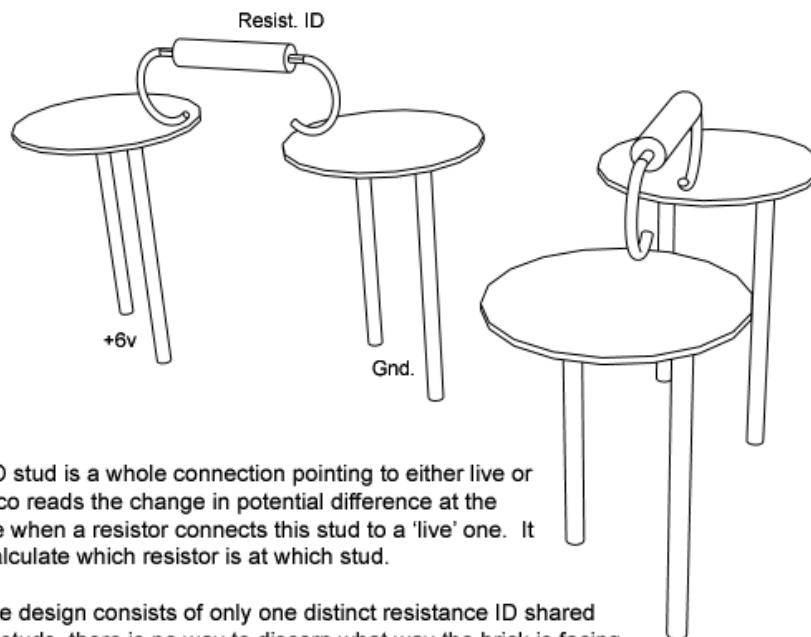
Figure 1: Prototype Design One ('Two Legs, Two Studs 1')

Unknown to me at the time, a similar device had been worked-on by researchers at MIT (Borovoy and Martin:1994) but with significant technological differences that would interrupt a practical application of a project built with it. First, their system was capable only of distinguishing two degrees of rotation whereas I sought to detect four (360° in an unrestricted, non-LEGO-ised, solution) on the intuitively guided principle that orientation mattered in video-games. Thus, their project was able to tell if a brick was facing either left or right but was unable to distinguish the two from each other. In addition the minimum size for a brick capable of being sensed using their system was two by four studs in area. More importantly, however, some difficulties with certain arrangements of bricks meant that various constructions would 'hide' bricks later added or confuse the

interpreting software when it received ambiguous data. These were things I was not willing to tolerate.

The original Avctive Baseplate designed by Borovoy and Martin has been employed to explore children's storytelling where, in a fashion similar to this project, bricks are used to represent narrative pieces (Glos and Umaschi:1997). A brick representing grandmother, for example, would tell a recipe if placed in a specially constructed dolls-house beside the oven. The story was attached to the element by the child using a standard computer interface and re-enacted by her/him in playing the game.

Prototype Design One(b) (One Leg, Two Studs):



Each LEGO stud is a whole connection pointing to either live or ground. Pico reads the change in potential difference at the ground side when a resistor connects this stud to a 'live' one. It can then calculate which resistor is at which stud.

Because the design consists of only one distinct resistance ID shared across two studs, there is no way to discern what way the brick is facing. Moreover the brick cannot accurately be located with much accuracy. We know what brick is present and roughly where it is located, but nothing more.

Figure 2: Prototype Design One(b) ('One Leg, Two Studs')

Despite this success, the actual implementation of the baseplate was beset by difficulties. In the context of the resources available to develop the project, the connections needed to make a ‘one-resistor-per-stud’ solution would have had to have been specially manufactured at prices beyond what were at the disposal of the project. What looked unfortunately likely was a ‘one-leg, two-studs’ solution that if implemented, would not only have the cost every degree of orientation but would have brought about a dramatic deterioration in what scenes it would have been possible to assemble since the exact position of a brick could no longer be determined with certainty (see figure 2). The answer was to employ the ‘dancing lights’ solution.

The baseplate at this point was made up of a checker-board of positive and negative (ground) terminals. The readings for the Pico were taken at the negative terminal. At any negative terminal there were only four possible positive terminals that could form a circuit with it by way of a connecting resistor that fitted across them. The resources of each channel of the Pico device (it has eleven input channels and one digital out-put) were shared across a number of grounded terminals by means of a system of counters and multiplexers that scanned each connection one after another. In the prototype I had built, each channel of the Pico took responsibility for one row of the baseplate and scanned each grounded terminal column by column. There was, in making these statements, a theoretical base that the search for a solution could take advantage of.

What was necessary in order to take a reading was to ensure that whenever a stud was to be read, it was connected to ground while the studs to either side of it were at live. The ‘dancing light’ solution was to alternate which set of terminals on the baseplate were live and which unconnected as the Pico advanced column by column through its scan. Using a tri-state buffer and the multiplexer set-up already in place, the electrical state for each terminal was set to live or disconnected depending on the state of the first bit of a binary counter. The circuit was assembled so that the input channels of Pico were always connected to a terminal that, at a time, was disconnected from live. The input channels were, through a resistor, connected to ground so that when a circuit was formed the potential difference across a stud connected to live by means of a resistor could be measured (see figure 3 and appendix B).

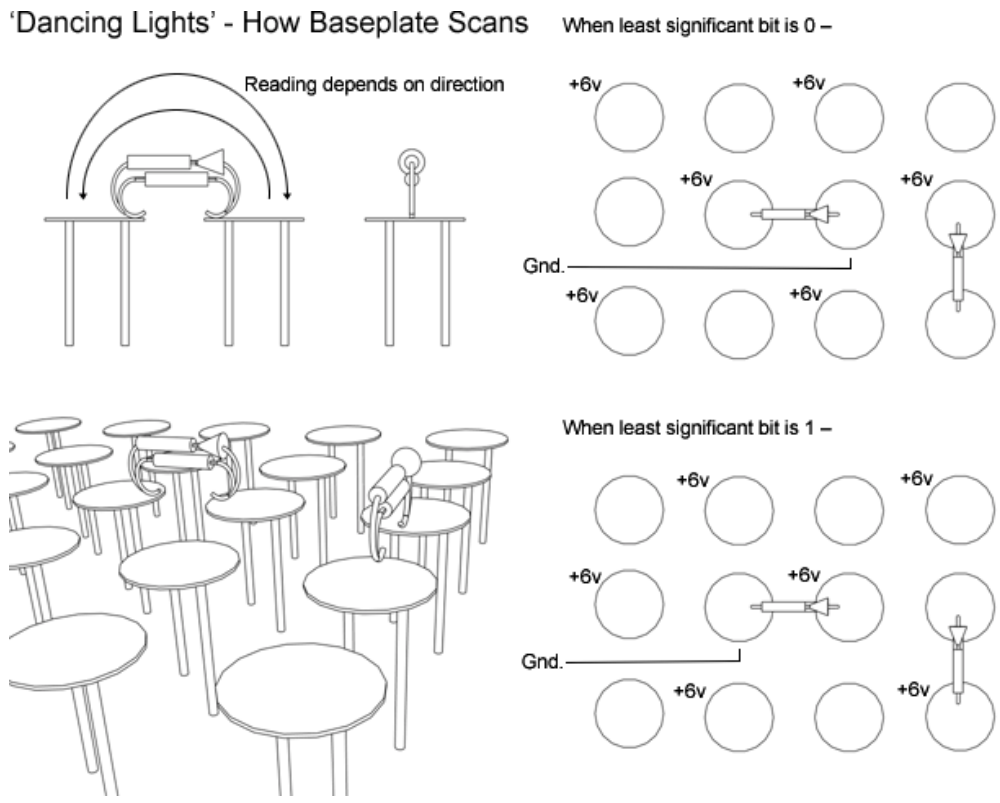
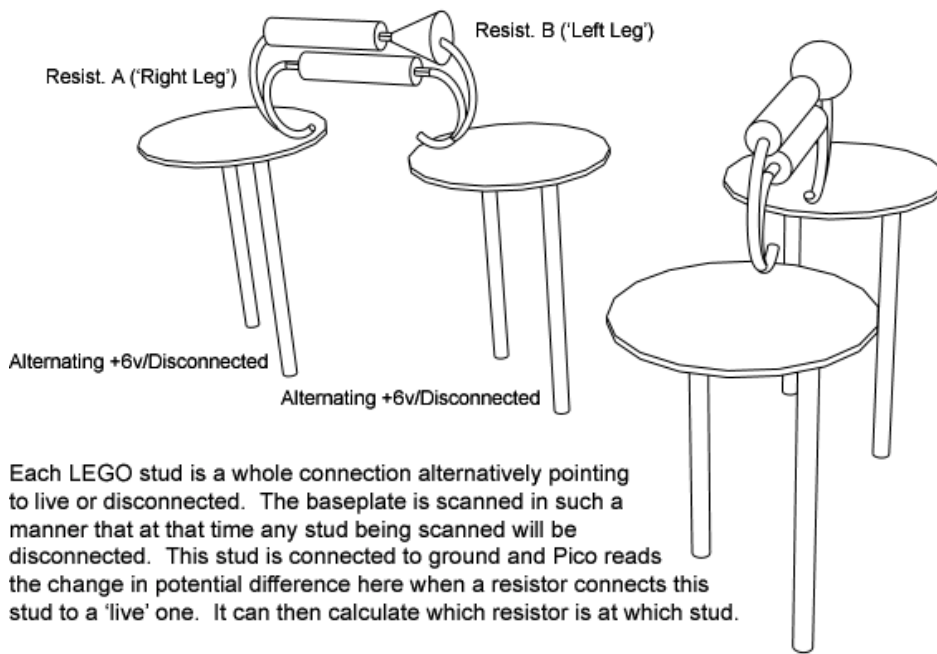


Figure 3: ‘Dancing Lights’ – How the Baseplate Scans

To the in-brick resistor is then added a second resistor and diode. Since one resistance will be measured on one run-through of circuit and another on a second, a ‘two legs’ solution is possible once again, allowing for four degrees of orientation and precise positioning with a minimum two by one area brick (see figure 4). This could, however, be improved again, if the resources that necessitated its invention were available to further develop the device. Given four connections per stud the minimum size of a token could be reduced to just one stud area (see figure 5).

Prototype Design Two (Two Legs, Two Studs 02):



Each LEGO stud is a whole connection alternatively pointing to live or disconnected. The baseplate is scanned in such a manner that at that time any stud being scanned will be disconnected. This stud is connected to ground and Pico reads the change in potential difference here when a resistor connects this stud to a 'live' one. It can then calculate which resistor is at which stud.

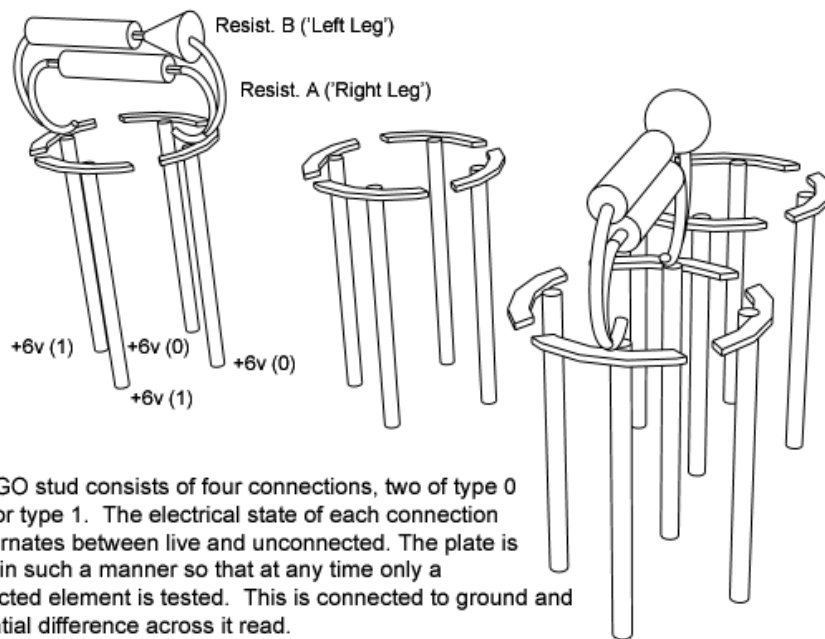
Because of the design what connects two studs, over the reading of adjacent studs two distinct resistance ID's are read for a bricks' 'left-leg' and a 'right-leg' enabling the reader software to discern brick, position and four degrees or orientation.

Figure 4: Prototype Design Two ('Two Legs, Two Studs 2')

There is, in theory, no restriction on the size of the area that the baseplate can cover. The resources of Pico can be shared across as many studs as are required using multi-dimensional arrays of multiplexers. While the connections a brick

makes with the baseplate need only to be one stud in area, the actual size of the brick can be as large as desired. Its position, orientation and identity are known to the processing computer, what appears on screen will be an accurate depiction of the constructed world. However, what is even more exciting for future development is the possibility of stacking.

Prototype Design Three (Two Legs, One Stud):



Each LEGO stud consists of four connections, two of type 0 and two of type 1. The electrical state of each connection -type alternates between live and unconnected. The plate is scanned in such a manner so that at any time only a disconnected element is tested. This is connected to ground and the potential difference across it read.

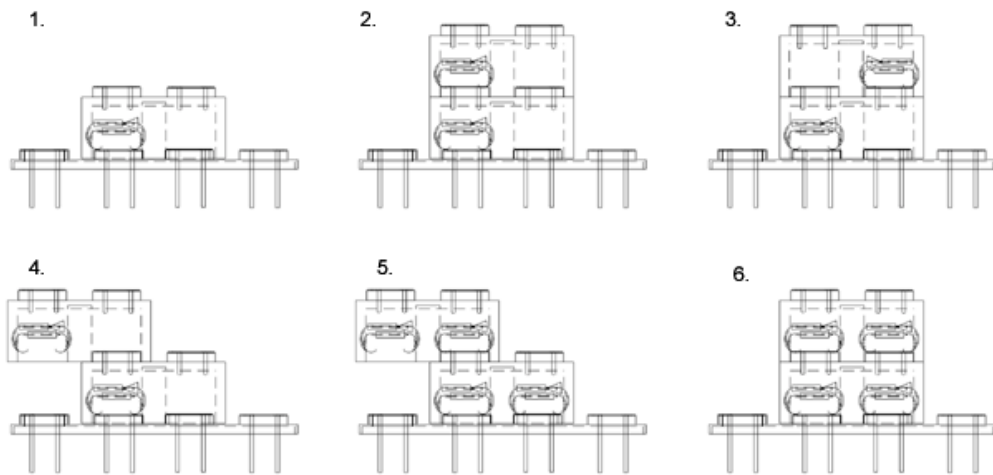
Because of the the design for the inner brick, two values – a 'left leg' and 'right leg' – is read for each stud. This allows the minimum size of a brick to be of just one stud area while retaining the ability to discern brick and position and four degrees of orientation.

Figure 5: Prototype Design Three ('Two Legs, One Stud')

By means of 'hollow' connections that run up through a brick, connecting further bricks on top of it to the baseplate, this implementation could be adapted to depict almost any arrangement of construction. Few restrictions would exist if it were possible to develop a plate and manufacture a brick that made distinct connections with the baseplate (directly or indirectly through connecting bricks below) for each stud area of it. In order for this to be successful, the base-plate

would have to constantly scan for new bricks as they are added during a specific ‘building-mode’. Brick could only be connected to ones already in place on the baseplate ie. they could not be snapped together and then added to the board as a unit. Because of these disadvantages, stacking would not be an feasible attribute to a project of the kind described in this report as it might lead to confusing and frustrating outcomes – children would have to remember to be in ‘building mode’ and to only snap bricks together once earlier ones had been connected to the plate already (see figure 6).

Exploring Stacking Possibilities



1. A single ‘two legs, one stud’ brick adapted to allow stacking. Additional connectors run through to the top from the lower connections.

2. Two such bricks stacked, the top brick connected to the base through the lower brick. The new bricks’ value is calculated as the original resistance is known and the second resistance can be deducted from what resistance must be in parallel to the old to cause the new reading.

3. A variation of example two, the brick is read through a ‘hollow’ connector running up from the first.

4. A stacking problem: the top brick is unconnected.

5. A possible solution: every stud area of a brick is fitted with distinct ID resistors identifying that particular stud, the reader software knows what brick each stud belongs to and where on the brick it is located, so being able to compensate for the ‘missing’ areas.

6. A simple stacking using the model explored in the last example .

Note: bricks can only be assembled on the baseplate.

Figure 6: Exploring Stacking Possibilities

None the less, the possibility of developing this project into a tool to build unrestricted 3D LEGO models on-screen in real-time, or using it to investigate ways of describing abstract data, is an attractive thought and one that should be

investigated. However, before that, aspects of development that relate more directly to this project should be considered. These relate to the construction of meaningful representations of social understanding in adaptable game-modelling through software development. This will be looked at briefly in concluding this report as it was not a part, as accounted for in the introduction, a factor in this prototypes development. Arguments of this kind are emerging only recently from abstracted debate about learning, expression and video-games and need further work and development if they are to be implemented properly in game-development.

Conclusion and Future Development

Whereas the possibilities for stacking bricks that build an virtual 3D model from a real-life one are exciting, if only for the intuitive appeal that it has (hence indicating that there may be a more purposeful application), where real application of technology to accompany a project of this sort are to be developed is in software. The game that accompanies this report, that of organising Bob's workplace, taking hold of him and painting a wall, do present some interesting possibilities. If many tokens, representing many interactions between them could be constructed and made meaningful the possibility of expanding the game, to enlist a multitude of contexts would become more real. In this respect the work of Richard Evans and Thomas Barnet Lamb (Evans and Lamb:2002) is interesting.

They discuss how software could be developed to better emulate human behaviour and organise games (the computational processes behind one) around social activities. In criticising examples of games, they argue that social activities are a part of human life and that games that do not understand them can appear 'dumb':

“The chess computer doesn't understand the place of chess within the social flux – it doesn't understand that chess is a game played for recreation or competition.” (Evans and Lamb:2002)

Likewise, games that attempt to emulate human activities can appear ‘dumb’. That they get it wrong is no wonder, computers are not human, but if, when considering the construction of agents to a game and the relationship between them and a game environment, we do not give due diligence to study the processes and understandings that underlie real-life human society, then clearly whatever characters and situations we invent will be duly devoid of human meaningfulness.

Evans and Lamb are under attack. They say that when we consider how to write characters in a computer game we should consider that their ability to act is as much played by the environment that they find themselves in as it is by the player. This is contrary to much understanding of what character play in computer games is. When we pick up a controller we expect to take control, we don’t expect to find out that we are a part of a wider knowledge and custom base. We don’t want to hear that we live our lives out of our control or of the complex relationship between the objects we interact with and other agents in the game. Such opinion of human activity, however – that the subject alone decides her/his life unaffected by social artifice – is not one that would stand up to much criticism in the social sciences.

More meaningful software, reflecting a more meaningful interpretation of the political, is needed if deeper understandings are to be explored in video-game. A system of the sort described in this report, allowing as it does the player to explore the relationships between narrative elements as tokens, needs a wider scope than most video-games allow if it is to integrate a multitude of tokens and

recreate meaningful interactions between them. What understandings it will enable a player to explore will be useless unless the authors of games to accompany it are willing to apply more thoughtful expressions in the games they create through the writing of a social theory to underlie the actions of agents and props employed in its use similar to what Evans and Lamb propose. If they do not there will be little to explore in them.

References

- Au, W.J., 2002, *Playing Games with Free Speech* in Salon.com. [online]
Available at: http://www.salon.com/tech/feature/2002/05/06/games_as_speech/
(29/8/2002)
- BECTa, 2001, *Computer Games in Education Project*. [online] Available at:
[http://www.becta.org.uk/technology/software/curriculum/computergames/docs/re
port.pdf](http://www.becta.org.uk/technology/software/curriculum/computergames/docs/report.pdf) (29/9/2002)
- Begel, A.B., 1996, *LogoBlocks: A Graphical Programming Language for
Interacting with the World*, Boston: Massachusetts Institute of Technology
- Begel, A.B., 1997, *Bongo: A Kids' Programming Environment for Creating
Video Games on the Web*, Thesis (MSc.), Massachusetts Institute of Technology
- Bergen, D., 2001, *Pretend Play and Young Children's Development*, ERIC
Digest. [online] Available at <http://npin.org/library/2002/n00721/n00721.html>
(12/9/2002)
- Bilton, T. et al, 1996, *Introductory Sociology*, MacMillan: London
- Blacklock, M., 2001, *Computer Game Becomes Violent Stalker* in Limby Limb,
July 13th 2001. [online] Available at

http://www.limbbylimb.co.uk/weekly/july13-01/violent_computer_stalker.html
(13/9/2002)

Borovoy, R. and F. Martin, 1994, *The Active Lego Baseplate Project*, MIT Media Lab. [online] Available at <http://web.media.mit.edu/~fredm/projects/ab/>
(12/9/2002)

Boyle, T., 2002, *Towards a Theoretical Base For Educational Multimedia Design* in Journal of Interactive Media in Education. [online] Available at:
<http://www-jime.open.ac.uk/2002/2/boyle-02-2.pdf> (29/8/2002)

Brand, S., 1972, *Frantic Life and Symbolic Death Among the Computer Bums*, Rolling Stone, 7th December 1972

Bruns, W et al, 1999, *Creating Virtual Worlds with a Graspable User Interface*. [online] Available at: <http://www.artec.uni-bremen.de/field1/rugams/papers/Twente99/TWTL15> (29/8/2002)

Cunningham, J., 2000, *The Importance of Unsupervised Play*. [online] Available at
<http://www.generationyouthissues.fsnet.co.uk/free%20play/The%20Importance%20of%20Unsupervised%20Play.htm> (10/9/2002)

Deudney, C., 2002, *Play and Autism*, National Childrens' Bureau/The National Autistic Society. [online] Available at http://www.ncb.org.uk/library/cpis/cpis_factsheet_autism.pdf (12/9/2002)

Edge Magazine, Issue 114, September 2002

Evans, M. and T.B. Lamb, 2002, *Social Activities: Implementing Wittgenstein* in proceedings from Game Developer Conference 2002. [online] Available at http://www.gamasutra.com/features/20020424/evans_01.htm (29/8/2002)

Fraunhofer Magazine, January 2002

Fuller, A., 2001, *Parents Get Cyber Savvy to Combat Youth Computer Addiction* in New South Wales and Southern Australia, 2001, *Parenting the Teenage Years*, Department of Community Services: Sydney

Glos, J.W. and M. Umaschi, 1997, *Once Upon an Object ... Computationally-Augmented Toys for Storytelling*, MIT Media Lab. [online] Available at http://gn.www.media.mit.edu/groups/gn/publications/jen&marina_iccima_97.pdf (12/9/2002)

Goldstein, R. and D. Pratt, n.d., *Michaels Computer Game*. [online] Available at: http://www.ioe.ac.uk/playground/RESEARCH/papers/open_modelling.pdf (29/8/2002)

Gorbet, M.G. and M. Orth, 1997, *Triangles: Design of a Physical/Digital Construction Kit* in proceedings from DIS '97, March 22—27. [online]

Available at:

http://tangible.media.mit.edu/papers/Triangles_DIS97/Triangles_DIS97.pdf

(29/8/2002)

Harris, J, 2001, *The Effects of Computer Games on Young Children – a Review of the Research*, London: Home Office

Kücklich, J., 1999, *Literary Theory and Computer Games*, Munich: LMU München

McNerney, T.S., 2000, *Tangible Programming Bricks: An Approach to making Programming Accessible to Everyone*, Thesis (MSc.), Massachusetts Institute of Technology

Naur, P., 1985a, *Programming As Theory Building* in Peter Naur (ed.), 1992, *Computing: A Human Activity*, ACM Press: New York

Newman, J, 2002, *The Myth of the Ergodic Videogame* in Game Studies, Vol. 2, No. 1

Resnick, M., 2002, *Rethinking Learning in the Digital Age*. [online] Available at http://www.cid.harvard.edu/cr/pdf/gitrr2002_ch03.pdf (29/8/2002)

Skirrow, G., 1986 *Hellivision: an Analysis of Video Game* in Colin MacCabe (ed.), *High Theory Low Culture*, Manchester: Manchester University Press

Southern, M., 2001, *The Cultural Study of Computer Games*, in proceedings from Game Developer Conference Europe 2001. [online] Available at http://www.igda.org/Endeavors/Articles/msouthern_intro.htm (29/8/2002)

Squire, K., 2002, *Cultural Framing of Computer/Video Games* in Game Studies, Vol. 2, No. 1

Speech and Therapy Activities, May 2001, *The Importance of Pretend Play, How and Why, Work* [online] Available at <http://www.speechtx.com/may2001.htm> (12/9/2002)

Strauss, B.S., 2001, *The Dark Ages Made Lighter* in Cowley, R. (ed.), 2001, *What If?*, Pan Books: London

Tholander, J., n.d., *Children's Use, Interaction and Learning in a Video-Game Construction Environment*. [online] Available at: <http://www.nada.kth.se/hmi/workshop2002/JakobTholander-HMI02.pdf> (29/8/2002)

Wilford, S., 2000, *From Play to Literacy: Implications for the Classroom*, Child Development Institute of Sarah Laurence College. [online] Available at <http://ericps.crc.uiuc.edu/eece/pubs/books/katzsym/wilford.pdf> (12/9/2002)

Bibliography

Aarseth, E., 2002, *The Dungeon and the Ivory Tower: Vive La Difference ou Liaison Dangereuse?* in Game Studies, Volume 2, Issue 1

Andersen, B. et al, 2001, *Games and Stories*. [online] Available at
http://www.staging.dk/pdf/10_MW_KKN_PBA_v2.pdf (29/8/2002)

Au, W.J., 2002, *Triumph of the Mod* in Salon.com. [online] Available at:
<http://www.salon.com/tech/feature/2002/04/16/modding/index.html> (29/8/2002)

Bernstein, C., 1991, *Play It Again Pac-Mac*, *Postmodern Culture*, Vol. 2, No. 1, September 1991

Borovoy, R.D., 1996, *Genuine Object Oriented Programming*, Thesis (MSc.), Massachusetts Institute of Technology

Cassell, J., J.W. Gloss, 1996, *Rosebud: Technical Toys for Storytelling*, Boston: Massachusetts Institute of Technology

Clarke, A., n.d., *New Media Semiotics*. [online] Available at
<http://www.kinonet.com/conferences/cosign2001/pdfs/Organisers.pdf>
(29/8/2002)

Dautenhahn, K., 2000, *Design Issue on Interactive Environments for Children with Autism*, Adaptive Systems Research Group, University of Hertfordshire. [online] Available at <http://web.mit.edu/16.459/www/Dautenhahn.pdf> (12/9/2002)

Du Preez, A., 1995, *Virtual Babes: Gender, Archetypes and Computer Games*, Pretoria: University of South Africa

Elton, M., n.d., *Should Vegetarians Play Video Games?*, Department of Philosophy, University of Stirling. [online] Available at http://www.stir.ac.uk/departments/arts/philosophy/staff/elton/papers_mine/vegetarian.pdf (12/9/2002)

Frasca, G., n.d., *Ludology Meets Narratology*. [online] Available at <http://www.jacaranda.org/frasca/ludology.htm> (29/8/2002)

Frazer, J., E. Sharlin, S. Sutphen, B. Watson, 1999, *Reviving a Tangible Interface Affording 3D Spatial Interaction*, Alberta: University of Alberta

Garitaonandia, C. et al., 1998, *Media Use and the Relationships of Children and Teenagers with their Peer Groups* in European Journal of Communication, Vol. 13, Issue 4

Gotbet, M. et al., 1998, *Triangles: Tangible Interface for Manipulation and Exploration of Digital Information Topography* proceedings from CHI '98, April

18—23. [online] Available at:

http://tangible.media.mit.edu/papers/Triangles_CHI98/Triangles_CHI98.pdf

(29/8/2002)

Herz, J.C., 1997, *Joystick Nation: How Computer Games Ate Our Quarters, Won Our Hearts and Rewired Our Minds*, Los Angeles: Little, Brown

Hornecker, E., n.d., *Graspable Interfaces as Tool for Cooperative Modelling*.

[online] Available at: [http://www.artec.uni-bremen.de/people/Eva/Papers/artec-](http://www.artec.uni-bremen.de/people/Eva/Papers/artec-01-Hornecker_IRIS24.pdf)

01-Hornecker_IRIS24.pdf (29/8/2002)

LeFebvre, J.E., 2001, *Parenting the Preschooler*, Family Living Agent:

Wisconsin

Livingstone, S., 1998, *Mediated Childhoods*, *European Journal of*

Communications, Vol. 13, No. 4, pp.435 – 456

Mortensen, T., 2002, *Playing with Players* in Game Studies, Vol. 2, No. 1

Myers, D., 1991, *Computer Game Semiotics* in Play and Culture, Vol. 4., No. 4.

Naur, P., 1985b, *Intuition in Software Development* in Peter Naur (ed.), 1992,

Computing: A Human Activity, ACM Press: New York

Norman, D., 1990, *The Design of Everyday Things*, Doubleday: New York

Preece, J. et al., 2002, *Interaction Design*, Wiley: New York

Resnik, M et al., 1996, *Piano not Stereos* in Interactions, Vol. 3, No. 6. [online]

Available at:

<http://el.www.media.mit.edu/groups/el/Papers/mres/pianos/pianos.html>

(29/8/2002)

Reynolds, R, 2002, *Playing a Good Game*, IGDA. [online] Available at

http://www.igda.org/Endeavors/Articles/rreynolds_intro.htm (29/8/2002)

Roe, C., 1999, *The EMPTY project (Empirical Modelling Principles for Teaching Youngsters)*, Thesis (MSc.), Massachusetts Institute of Technology

Ryan, C., 1998, *Mediated Childhoods* in European Journal of Communication,

Vol. 13, Issue 4

Ryan, Marie-Laure, 2001, *Beyond Myth and Metaphor – the Case of Narrative in*

Digital Media in Game Studies, Vol. 1, No. 1

Schleiner, A-M in *Interview with Jim McCellan*, Guardian, Thursday April 8th

1999 [online] Available at: <http://www.opensocery.net/jiminterview/>

(29/8/2002)

Small, D.L., 1999, *Rethinking the Book*, Thesis (PhD), Massachusetts Institute of Technology

Szilas, N., 1999, *Interactive Drama on Computer: Beyond Linear Narrative*, Paris. [online] Available from: <http://www-2.cs.cmu.edu/afs/cs/user/michaelm/www/nidocs/Szilas.pdf> (29/8/2002)

Appendix A: Moral Panic and Video Games

Despite the furore that forms the vanguard of popular opinion about video-games and video-game culture little effort has been dedicated to the study of games or gamers under an academic light. It is a subject area that is largely uncharted and compounded by a lingering sense of mystery and myth on the part of academics. Computer games occupy a space in computer science that is defined by their history. First written as technology-for-all demonstrations of elusively high-counting maths and science devices, as was the case for the primordial *Tennis for Two* (Willy Higinbotham:1958) and more established *SpaceWar!* (MIT:1962), as computing sought to establish itself as a distinct discipline early games were shunted to be found loitering among the underground ‘hacker’ community of technicians and solder-hacks tolerated only to the extents of their usefulness, not officially a part of a nascent discipline favoured among military interests and big business patrons. These attributes still characterise the space occupied by gaming today by a majority in both academia and the arts. Computer games’ adolescent years were a season of omission, a misappropriation of ‘serious’ technology. A significant observation would be that video games, even more so than television, are a waste of time.

One recent observation of such can be drawn from the decision of a Louisiana court not to grant freedom of speech protection to video games (Au:2002). Although the case was spoiled by ineffective argument on behalf of the pro-game counsel and looks likely to be over-turned on appeal on account of previous decisions by higher courts in favour of the games industry, the ruling that games

carry “no conveyance of ideas, expression or anything else that could possibly amount to speech” is a popular one. Wide-spread public misgivings with regard to video games occasions a ‘moral panic’ (Southern:2001). Common assertions are that video games inherently lead to aggression, lethargy, aloofness, addiction and falling grades among children, though evidence to support such claims are sparse.

In making this argument, it is not my intention that my opinion be confused with statements to the effect that all video games are beneficial, or even worthwhile, but rather to dispose of some common misconceptions of games and gamers as partaking in a necessarily mindless and/or dangerous activity. Although, as stated already, little sociological research has been undertaken into computer games or game-play, but what little there has neither substantiates nor disputes what is commonly conceived. What is needed above all else is that qualitative research be undertaken into gaming activities in order that at least gamers be understood and from there we may understand games. What work has been undertaken so far has only been made in response to (unfounded) public concern about games, as such it has been reactionary rather than scholarly in nature, seeking to prove an answer rather than wanting to find-out more.

A longitudinal study of 75 self-reported computer game ‘addicts’ between 1989 and 1994 (Shotton:1994) against two control groups found that the ‘addicts’ did better in secondary education, more went on to university and from there into higher ranking-jobs. The study concluded that game-players were “generally highly intelligent, motivated and achieving people but misunderstood.” Other

studies have concluded that computer game-play does not affect academic performance (Creasey and Myers:1986). Although, these studies were made during the Spectrum-like era of gaming (and most significant among my concern would be what class bias in ownership of the machines might affect educational results more) or during the crossover period between home computers and video consoles, I believe their findings deserve restating today.

The lack of real study into this area is confounding, though possibly testimony in absentia to weaknesses that form the bulk of what truth may underlie the 'moral panic' sweeping Western nations. The US Senate has listened to hearings on video games and violence (Edge:114) and the UK Home Office ordered a review of research into the effects of computer games on young children (Harris:2001), both ended inconclusively, by which they mean that the dangers perceived in computer game were not demonstrated to be present. A pamphlet issued to parents of children by the state government of New South Wales encouraged parents to "get cyber savvy to combat youth computer addiction" and summarised that "[t]he good news is, he isn't watching as much television." (Fuller:2001) Parents, alert to the dangers, the authors writes, should ask themselves, "Is my teenager more guarded or secretive than they used to be?" More guarded or secretive than any other teenager, I presume, but such hysteria is not limited to solely political rhetoric.

Laboratory studies undertaken by academics in order to positively prove the link between computer game-play and the dangers associated with it include:

- One study (Schutte et al.:1988 in Harris:2001) to demonstrate that children will mimic what they see character do in video game. Children were instructed to play either a combat or a non-combat computer game and then invited to strike a ‘bobo’ doll, the number of times that they struck it being counted and held determined to be significant. Hardly conclusive since the doll may more properly have been perceived as an extenuation of the game activity, since they were invited to interact with it directly after the game ended, and striking it interpreted in a playful manner not rightly generalised to broader social exchanges.
- Griffiths and Dancaster (1995 in Harris:2001) split their selected study population according to personality-type in order to demonstrate that susceptible players will be roused to aggressive behaviour by video games. They made two groups of their participants: group A, described as, “competitive, achievement orientated, exhibiting time urgency and anger or hostility”, and group B, composed of those having “low levels of competitiveness, time urgency, easy-going, philosophical about life.” , Their test was to monitor the heart-rate and arousal of participants at game-play when encouraged to do well by offering £10 to the participant with the highest score. Unsurprisingly, participants from group A were significantly more aroused by the challenge and thus Griffiths and Dancaster demonstrated their hypothesis.
- More recently, Anderson and Dill (2000 in Squire:2002) sought to demonstrate the dangers of violent games by pitting their subjects against one another, with some participants set a ‘violent’ game (*Wolfenstein 3D*) to play and some a ‘non-violent’ game (*Myst*). The losers of each game

were invited to strike-back at their victors by ‘blasting’ them with sound for as long as they liked. The significant conclusion of this study was that players of the ‘violent’ game blasted their opponents for an average of 6.81 seconds while players of the ‘non-violent’ game only blasted their opponents for an average 6.65 seconds. What transferable reading the authors of this report were able to draw of the difference 0.16 seconds ($\frac{4}{25}$ ^{ths} of a second) is difficult to believe.

The studies given as examples here all suffered from low population numbers, typically 20–30 participants.

Links, too, between video games and crime are tenuous. The only reference to such that I could find being in the UK Home Office review which pointed to a study by Huff and Collinson (1987) of juveniles held at a children’s prison of which 13% said they had stolen to play video games. How significant these thefts were, how often they had happened, and the qualitative nature of them – whether this was the shoplifting of software titles from stores to play at home or the taking of ten-pence pieces from their parents’ bedside lockers to slot into arcade machines – is unstated. Irrespective of this, a study of convicted juveniles cannot be generalised to the players of computer games as a whole and it is not mentioned whether any of the children interviewed were convicted of charges related to whatever criminal activity they undertook that was tangential to playing computer games.

Appendix B: Elementary Circuit Design for an Active LEGO Baseplate

The ‘dancing lights’ element of the Active LEGO base-plate is explained through illustrations in the section ‘Designing the Active Baseplate’, figure 3. What appears in this appendix is a brief overview introducing the circuit design.

The circuit was built on standard breadboards, soldering some wire-ends to the underneath connections of the baseplate for security. The prototype was initially, against better advice, attempted to be assembled using solder and circuit-board. The result was laboriously slow, impossible to debug and a waste of time.

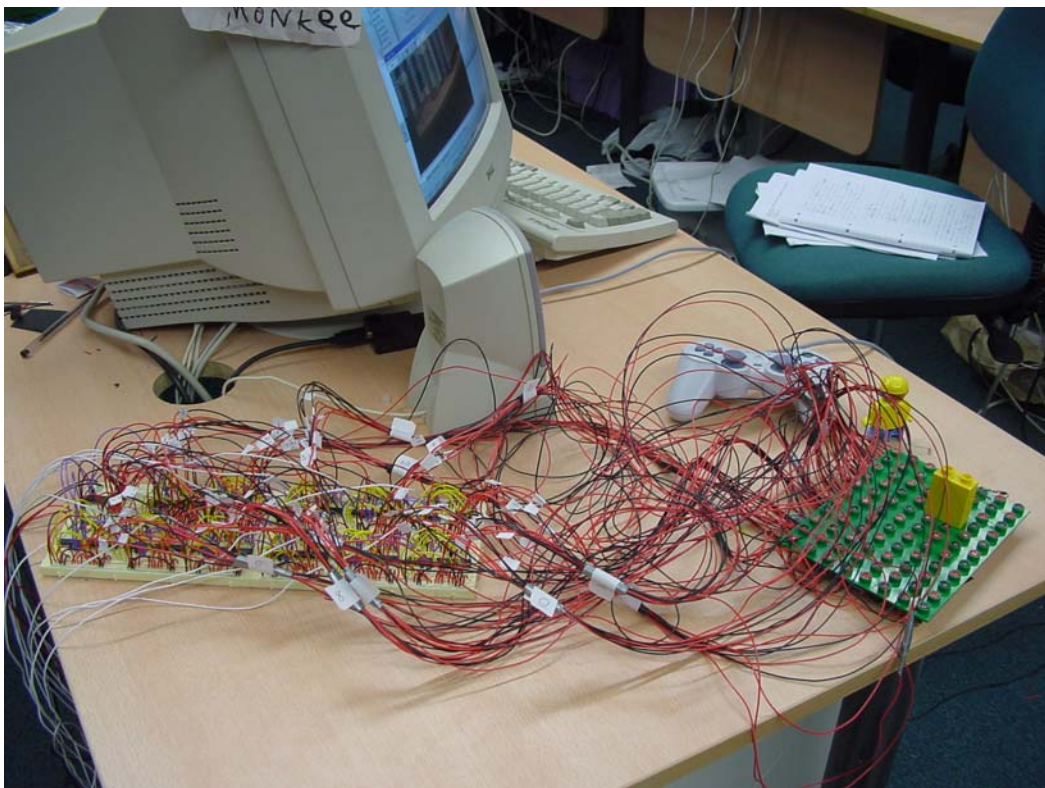


Figure 7: Photograph of final prototype ('Two Legs, Two Studs 2') being tested

The central components used to make this baseplate were (all are CMOS):

- 12-Stage Binary Ripple Counter (CD4020BC)
- Single 8-Channel Analog Multiplexer (DG408)
- Octal Bus Transceiver with 3-State Outputs (SN74LS245)
- Pico Device (ADC 11/22)

Method:

The binary counter advanced by the PC controls an array of multiplexers that share the resources of each channel of the Pico device across a number of studs. A tri-state buffer, controlled by the least significant bit of the counter, alternates every second stud between +5v or unconnected. The multiplexers are set-up so as to ensure that whenever the Pico device is connected to a stud that that stud is disconnected from live. On connecting a stud to Pico, it is simultaneously connected to ground so that an electrical current flows through a resistor put between that stud and a stud connected to +5v. The baseplate is arranged so that when ever a stud is connected to Pico (and ground) that the studs immediately above, below, to the left and to the right are at live. The potential difference across the resistor(s) that creates a circuit is measured by Pico.

A similar design to overleaf is used for all connections.

Appendix C: Parsing 3DS Max ASCII scene export files to OpenGL

The software accompanying this prototype was custom written alongside the development of the core hardware components. C++ was chosen over other development environments, mainly Macromedia Director, for the control, stability and speed that it offered. A second deciding fact was that the author had above other alternatives a more ready theoretical principle as to how the computational processes that interpret and present data collected from the baseplate would be described using that language. Although, he had never undertaken 3D graphics animation or programming in either of the main contending environments, it was thought that the level of learning required in order to present a reasonable demonstration of the project would be broadly the same in both. In the main this was true.

OpenGL was the API worked while using C++. It was chosen above DirectX for its well-stated simplicity and the added benefits of avoiding common application calculations that the GLUT provided. In practice, working with OpenGL was a joy.

However, the decision to use 3DS Max, based as it was on being the authors only large-scale experience with 3D graphics design proved to be the most problematic and time-consuming element of the project in its entirety. On more than one occasion, the difficulties raised by Max cast doubt over the feasibility of completing the project using C++ despite the enormous advantages that that environment presented in every other aspect, including 3D manipulation.

3DS Max was used to design the three-dimensional models and animations that would be the on-screen representation of tokens being worked with while playing with the Active Baseplate. Poser 4 was used to design human characters and animations, these models then being brought into Max for further re-working – mainly in order to reduce their polygon count to a more comfortable number that would animate well in code. Models from Max were then exported as an indigenous ASCII Scene Export (.ase) file. In principle, this facilitated the opportunity to write an application that would automatically parse Max files into C++ (OpenGL) code. The .ase file-type is a near-human description of a Max scene, it can be read as a text file and so the possibility of writing an application to read and translated into something more useful is inviting. Eventually, this is what happened but not before more than eight weeks of frustration has passed.

The cause of this frustration was Max's preference to re-invent standard descriptions of 3D space as arbitrary matrix transformations. This has some aesthetic worth in mathematical theory, but close to none in real terms. What is more troublesome is the desire that Max has to intermix these re-inventions with more widely understood conventions without reference to which description of space is in use at any one time.

Max, as a rule, describes points in space through a little used system. The standard way is to mean by a three-dimensional array (x, y, z) that the point one is describing is x distance to the right, y distance up and z distance back. Max swaps the latter two and inverts the standard z. So when Max describes the point

(x, y, z), what is indicated is x distance to the right, y distance forward and z distance up. This deviation is well documented and, though troublesome for the unsuspecting, easily overcome. However, by changing the matrix, this is, changing what is meant by up, forwards or to the right, a problem more difficult to overcome is created.

That difficulty is caused not by the maths involved but by the inconsistency of the description we receive of the changes occurring in 3D space. In the .ase format, these transformations are denoted by the references:

```
TM_ROW0 number number number  
TM_ROW1 number number number  
TM_ROW2 number number number  
TM_ROW3 number number number
```

and

```
TM_POS number number number  
TM_ROTAXIS number number number  
TM_ROTANGLE number number number
```

In a file, these descriptions of an axis transformation (TM) occur one after another and describe the same thing. That the same thing is denoted twice is confusing enough. However, if it is considered that what they describe is irrelevant to the larger bulk of the file, needlessly affecting only a minor part but none the less essential to the purposeful utility of the whole, then the infirmary that their obscure presence incites will be obvious. Every part of an .ase file-type uses the standard matrix (what we understand as up, down, left, right) except for the part furthest from the reference to any other and bearing no indication that this anomalous section of code might incur upon its meaning. The part that it is

relevant to, is the section that describes how light should cast off the three-dimensional objects in a scene.

Normals are imagined lines (commonly described as one unit long, unitary normals) that stand perpendicular to a surface. Their purpose in 3D graphics is to cleverly allow objects that are in fact no more than an angular, flat body of polygons appear rounded by causing light to bounce off it as if it were so.

Being unable to understanding why an objects' normals were 'not working properly', several attempts were undertaken to 'fix' the problem, eventually leading to success. The first of these was to use face normals calculated independently of Max. These are simple calculations based on a given polygon that returns what a unitary normal to it would have if the face were treated as flat. Using these, which drew hard, unconvincing graphics on-screen, the relationship between the calculated face normals and those offered by Max were explored. It was summarised that all the given face normals of a polygon were off-set against the calculated face normals by the same angle and rotation. An assumption was made that the true normals (what we were interested in retracting from Max, the ones describing a polygon as a true, rounded object) were off by the same degree. This proved to be the case.

For the most part this technique, transforming the described true normals by the same amount that described face normals were off against the calculated ones, was successful. However, depending as the calculation did on cosine in order to remedy a solution, so to speak, some situations were left unsolvable (cosine

approaches infinity as it angle nears $\pm 180^\circ$). However, success at this procedure led to the exploration of the aforementioned TM references. Eventually, code for the parser recognised the meaning of these parts and adapted Max normal description as was appropriate. The section of code that does so is transcribed below:

```

void polygon::sortNormals()
{
    // [ a b c d ] (TM_ROW0)
    // [ e f g h ] (TM_ROW1)
    // [ i j k l ] (TM_ROW2)
    // [ m n o p ] (TM_ROW3)
    //
    // multiplied by a column vector (x, y, z, w)T (which may
represent
    // either a vector or a point) yields the vector:
    //
    // (ax + ey + iz + mw, bx + fy + jz + nw, cx + gy + kz +
ow, dx + hy + lz + pw).

    int i;
    float x, y, z;

    for ( i = 0; i < MESH_FACENORMAL_LIST.size(); i++ ) {
        x = MESH_FACENORMAL_LIST[i].getX();
        y = MESH_FACENORMAL_LIST[i].getY();
        z = MESH_FACENORMAL_LIST[i].getZ();

        MESH_FACENORMAL_LIST[i].setX(    TM_ROW0.getX()*x    +
TM_ROW1.getX()*y + TM_ROW2.getX()*z );
        MESH_FACENORMAL_LIST[i].setY(    TM_ROW0.getY()*x    +
TM_ROW1.getY()*y + TM_ROW2.getY()*z );
        MESH_FACENORMAL_LIST[i].setZ(    TM_ROW0.getZ()*x    +
TM_ROW1.getZ()*y + TM_ROW2.getZ()*z );
    }

    for ( i = 0; i < MESH_VERTEXNORMAL_LIST.size(); i++ ) {
        x = MESH_VERTEXNORMAL_LIST[i].getX();
        y = MESH_VERTEXNORMAL_LIST[i].getY();
        z = MESH_VERTEXNORMAL_LIST[i].getZ();

        MESH_VERTEXNORMAL_LIST[i].setX(    TM_ROW0.getX()*x    +
TM_ROW1.getX()*y + TM_ROW2.getX()*z );
        MESH_VERTEXNORMAL_LIST[i].setY(    TM_ROW0.getY()*x    +
TM_ROW1.getY()*y + TM_ROW2.getY()*z );
        MESH_VERTEXNORMAL_LIST[i].setZ(    TM_ROW0.getZ()*x    +
TM_ROW1.getZ()*y + TM_ROW2.getZ()*z );
    }
}

```

To the best of this authors' knowledge, of the existing software to convert the .ase file-type to OpenGL, certainly among independent developers, this is the only one to be successful at overcoming this problem. Documentation to accompany the .ase file-type is non-existent. What is available is only what a few disparate people can surmise from exploring their own exports. Below is what the explorations into parsing .ase for this project has revealed. What is apparently missing from it is a description of animation as the .ase code to handle this generates elaborates descriptions of tweens rather than more prosaic and usable 'cell-frame'-type animations. Investigating animation alone would be worthwhile endeavour, though tiresome and difficult, but more needs to be openly documented before such an undertaking can be made. The descriptions that follow are incomplete and refer only to elements that are relevant to this project and as .ase relates to OpenGL. They are listed as they would occur in context.

As a reference, `number`, below, refers to a single number whole or decimal. `x`, `y`, and `z` refer to points in three-dimensional space. `rgb` refers to a three part value for colour (given in red, green and blue) where 0 indicates a complete absence and 1 complete saturation. `string` is used to denote an alphanumeric string enclosed in double quotes. Other references to data-types will be explained as they arise.

| | |
|---|--|
| <code>*3DSMAX_ASCIIEXPORT number</code> | The number here refers to the version of the ASCII Scene Export generator. |
| <code>*COMMENT string</code> | Describes the version once more along with the day, |

| | |
|-------------------------------|--|
| | time and date that the export was made. |
| *SCENE | Denotes the start of a section describing broad technical attributes of the scene. |
| *SCENE_FILENAME string | The file-name of the file from which this export was made. |
| *SCENE_FIRSTFRAME number | The first frame of the scene (usually 0). |
| *SCENE_LASTFRAME number | As above but pointing to the last frame of the scene (not necessarily the last frame that animation takes place in). |
| *SCENE_FRAMESPEED number | Frame-speed in frames per second. |
| *SCENE_TICKERSPEED number | Refers to Max's own time keeping system. |
| *SCENE_BACKGROUND_STATIC rgb | The main background colour of the scene. |
| *SCENE_AMBIENT_STATIC rgb | The ambient colour of the scene. |
| *MATERIAL_LIST | Indicates the start of a section describing an array of materials used in the scene. |
| *MATERIAL_COUNT number | The number of materials present in the scene (the size of the array). |
| *MATERIAL number | Indicates the start of a list of elements belonging to the MATERIAL_LIST array. |
| *MATERIAL_NAME string | The name of this material. |
| *MATERIAL_AMBIENT rgb | The ambient colour of the material. |
| *MATERIAL_DIFFUSE rgb | The diffuse colour of the material. |
| *MATERIAL_SPECULAR rgb | The specular colour of the material. |
| *MATERIAL_SHINE number | The shininess of the material. |
| *MATERIAL_TRANSPARANCY number | The alpha channel value of material. In OpenGL only applicable to MATERIAL_DIFFUSE. |
| *MAP_NAME string | The name of the bitmap used to texture this material. |
| *BITMAP | The full path location to the bitmap file that this material uses. |

| | |
|---|---|
| *GEOMOBJECT | Indicates the start of the description of a 3D object. Each element of the scene is treated as a separated GEOMOBJECT. |
| *NODE_NAME string | The name given in Max to this object. |
| *NODE_TM | Indicates the start of a section dealing with the matrix transformation for this object. (*NODE_NAME is repeated needlessly once again at the start of this section.) |
| *TM_ROW0 x y z *TM_ROW1 x y z *TM_ROW2 x y z *TM_ROW3 x y z | The matrix which vector will be multiplied normal by. Max by preference uses 3x4 matrices. The third row of this matrix (TM_ROW3) is irrelevant in practice. |
| *TM_POS x y z *TM_AXIS x y z *TM_ROTANGLE number *TM_SCALE x y z | The same information as above described in a different way – the position, direction and angle by which to tilt the matrix and how much to scale it by. |
| *MESH | Indicates the start of a section describing the polygon elements of the 3D object. |
| *MESH_NUMVERTEX number | The number of vertices in an array of vertices to follow. |
| *MESH_NUMFACES number | As above, the number of faces in an array of face description that follow. |
| *MESH_VERTEX_LIST | Indicates the start of a list of vertices as elements of an array. |
| *MESH_VERTEX number x y z | A vertex. number is the index given to this array element, and x y z is its location in 3D space (unaffected by the reference to a change in the matrix above). |
| *MESH_FACE_LIST | Indicates the start of a section that lists faces by describing the elements in an array of vertices, above, that go up to make them. |

| | |
|----------------------------------|--|
| *MESH_FACE number a b c ab bc ca | A description of one face of the GEOMOBJECT. number refers to the place this description has in the list of faces. a b c refer to which elements of the array of vertices are vertices of this polygon. ab bc ba refer to the edges of this polygon that do not form visible edges (Max describes all polygons as triangles but some graphics cards may be faster drawing quadrilaterals, etc. depending on circumstance). |
| *MESH_MTLID number | The element of the array of descriptions below that tells how a bitmap texture is to be applied to this face. |
| *MESH_NUMTVERTEX number | The number of elements in an array of vertices on the texture bitmap that will be used to describe its application to a specific face. |
| *MESH_TVERTLIST | Indicates the start of a list that describes each element of an array of vertices on the texture bitmap. |
| *MESH_TVERTLIST number x y z | An element in the array of vertices on the texture bitmap that will be mapped onto a face. number is the index of this element in the array, x y z refer to the location of the vector this element points to on the 2D texture. z will, in all but the rarest of exceptions, be 0 since we are dealing with a 2D texture. |
| *MESH_NUMTFACES number | Like *MESH_NUMTVERTEX this is the number of elements in an array where each element describes an arrangement for a texture to be mapped onto a face. |
| *MESH_TFACELIST | Indicates the start of a list of elements describing how a texture is to be mapped onto a face. |
| *MESH_TFACE number a b c | An element of the array of descriptions. number refers to this elements' index, a b c refer to the texture vertex descriptions that are mapped onto the a |

| | |
|---------------------------------|---|
| | b c vertices of a face referring to this element as its MESH_MTLID. |
| *MESH_NORMALS | Indicates the start of a description of normals to the faces of this object. |
| *MESH_FACENORMAL number x y z | A face normal to the face with the same index as number (it will come in the same place in the MESH_FACE_LIST as this description appears here). x y z refer to the vector normals' location in space – this will be affected by the matrix transformation. |
| *MESH_VERTEXNORMAL number x y z | A description of a true normal to the object. number refers to the index of the vertex that should refer to this (MESH_VERTEXNORMAL comes in threes, they are the a b c vertex of the face that their corresponding MESH_FACENORMAL is normal to). x y z are this vector normals' position in space – this will be affected by the matrix transformation. |
| *MATERIAL_REF number | Indicates the element of the materials' array that should be applied to this object. |