5rd INTERNATIONAL CONFERENCE ON INTELLIGENT GAMES AND SIMULATION

GAME-ON 2004

EDITED BY

Abdennour El Rhalibi

and

Danny Van Welden

NOVEMBER 25-27, 2004

HET PAND
GHENT, BELGIUM

A Publication of EUROSIS-ETI

Printed in Ghent, Belgium

Cover art was reproduced by kind permission of Larian Studios, Oudenaarde, Belgium

5TH International Conference on Intelligent Games and Simulation

GHENT, BELGIUM

NOVEMBER 25-27, 2004

Organised by

EUROSIS

Liverpool John Moores University

Co-Sponsored by

Atomic Planet

Binary Illusions

University of Bradford

Delft University of Technology

GIGnews.com

Ghent University

The Moves Institute

Simulation First

Sheffield University

Warthog

Hosted by

Ghent University

Ghent, Belgium

EUROSIS-ETI Publication

ISBN: 90-77381-15-5

GAME'ON
2004

PREFACE

Welcome to GameOn 2004.

Recently by the advancement of computers and networks new types of entertainment have been emerging such as video games, entertainment robots, and network games. As these new games have a strong power to change our lives, it is good for people who work in this area to discuss various aspects of entertainment and to promote entertainment related researches, and collaboration between academic and industry.

Computer Entertainment is an exciting topic that crosses over a large variety of different areas such as design and storytelling, human computer interface design, computer music, graphics, software and hardware development, education, psychology and communication. In an era where researchers are becoming increasingly specialized it is refreshing to be able to work and participate in a truly multi-disciplinary field. Attendees at the event should relish the opportunity to discover the links between their research interests and those from a range of different disciplines.

I hope that by attending this conference you would learn about the state-of-the-art of Game Design and Technology and the most recent research trend in this area and would have opportunities to exchange opinions with other participants.

I welcome all participants to this exciting event and hope our interactions will stimulate collaboration that fuels further innovation and career aspiration.

Abdennour El Rhalibi

GameOn 2004 Programme Chair

CONTENTS

CONTENTS

SCIENTIFIC PROGRAMME

GAME
AI

A NOVEL PLATFORM FOR DEVELOPING MUNDANE SKILLS IN ARTIFICIAL PLAYERS

Alasdair Macleod
Computing Department
University of the Highlands and Islands
Lews Castle College
Stornoway
Isle of Lewis, UK
HS2 0XR
Email: Alasdair.Macleod@lews.uhi.ac.uk

KEYWORDS

Perudo, Artificial Intelligence, Games Testbed, Mundane Tasks

ABSTRACT

The use of games platforms for the development of AI is reviewed. Computer games require artificial agents that exhibit human-like behaviour. This is identified with mundane abilities and the dice game of perudo is proposed as a suitable game platform for developing these mundane skills. The basic features of the game are quantitatively reviewed and simple evolutionary learning algorithms are developed and tested.

INTRODUCTION

There is a strong distinction between what might be called 'real-life' artificial intelligence (AI)[1] and the execution of what are deemed intelligent tasks by artificial means. The assumption is often made that an AI system designed for and effective at a specific task should somehow develop features of real-life intelligence. This is not necessarily the case and this paper will consider the characteristics that make real-life AI so challenging, and proposes that artificial systems that exhibit real-life intelligence can be developed and tested within an appropriate games environment.

The topic of artificial intelligence has historically had strong links with game play. Traditionally, the preferred game platform for testing new research ideas in AI was chess because of its clear rule-set, the ability to model the game using minimal memory (important when the subject of artificial intelligence was at a developmental stage), and the perceived strong correlation between skilful play and human intelligence. It was considered a microcosm of real life. However, chess is a perfect information game and is, in principle, solvable by conducting a search of sufficient depth. This can be achieved through the employment of linear programming and computing power alone. Schaeffer (2001) has commented on the close association between the milestones in chess program development and the progression of the state-of-the-art in high performance computing. Chess is actually far from being a microcosm of real life – in real life the dominant element is the unknown and the most prevalent activities are the mundane. It is therefore doubtful that chess can make a significant contribution to real-life AI development and the recent success of the brute force computer programs supports this view[2].

Researchers are well aware of the limitations of perfect information systems and rather than completely abandon game platforms with their advantages of compactness and controllability, alternative games development platforms are being actively considered (Bringsjord and Lally, 1997). Some researchers (Nareyek 2004a, 2004b) have proposed that the best environment for the development of artificial intelligence is not any of the traditional games but massively multi-player Internet games. In this paper, we consider how the clear demand for game agents with human characteristics to enhance the game-playing experience can influence AI research. Driven by a commercial need, the current and future developments in this field can provide a significant boost to AI development.

The complexities of the Internet game environment and sometimes very specific aspects of the agent's role can give rise to ad hoc behaviours lacking the generic properties that give the agent value out with its development environment. It is argued that there is still a very important place for AI development in more controlled environments with limited degrees of freedom where basic AI building blocks can be developed and fully tested[3].

Incomplete knowledge games such as poker and chess have been proposed as suitable test beds for the development of machine intelligence and models created up to now have met with significant success (Billings *et al.* 1998, 2002, 2003). The important characteristics of these games are incomplete knowledge, multiple competing players who must be effectively modelled, and the management of unreliable information (associated with opponent errors and bluffing).

[1] This concept is related to 'strong AI' where the simulation works *like* the human brain and offers insight into the operation of the brain.

[2] The papers from the volume *Proceedings of the AAAI-97 Workshop on Deep Blue vs. Kasparov: The Significance for Artificial Intelligence* (ed. R. Morris), Providence, RI. are relevant, particularly Korf R.E. "Does Deep Blue use AI?", p1; Heath D., Allum D, "The Historical Development of Computer Chess and its impact on Artificial Intelligence", p63.

[3] It is acknowledged that there are doubts that 'Intelligence' can be deconstructed into simpler discrete blocks that exhibit independent functionality.

Of course, there are many popular games that share these characteristics to a greater or lesser extent. A previous paper (Macleod, 2004) proposed that a simplified version of the popular dice game perudo (closely related to liar or liar s dice) is a better choice because the game has less complicated rules whilst retaining a similar level of playing complexity. The complexity arises entirely from human interaction with the result that the value of calculation and reasoning is depressed, but the importance of real-life mundane skills is highly elevated. In this paper, the potential application of perudo to AI research will be quantitatively evaluated.

COMPUTER GAMES

Laird and van Lent (1997) write of the failure to build human-level AI systems, systems that exhibit real -time response, robustness, autonomous intelligent interaction with their environment, planning, communication with natural language, commonsense reasoning, creativity and learning . Many specialised solutions have been developed to solve specific problems (outwith the games field) but it is unclear whether these are suitable components or building blocks for the construction of a human-level AI system. In any case, an application is required to drive forward such a development. They suggest that interactive computer games such a *Doom*, *Quake* and *Myth* are a suitable environment that demands human-level AI.

The AI requirements in such domains are identified by Nayerek (2004a) primarily as the physics of the environment and the achievement of believable responses from non-playing characters (NPCs). It is pointed out that the AI field is highly fragmented by the search for solutions to a range of very specific problems, resulting in narrow intelligence. Academic research is therefore largely inapplicable to computer games. This leaves the games industry to develop in isolation from a very basic start point. It is unclear if any resulting developments will have a strong reciprocal effect on academia.

A good example of this failure of academic research to apply is the programming of NPCs whose specific role it is to assist the player. Sophisticated developments in natural language processing are not used; consequently agents are unable to interact in a natural way because question and answer sequences have to be scripted.

Nayerek (2004b) rallies researchers to embrace the challenge of developing AI in the games domain, to attack some of the huge difficulties introduced by vast interaction possibilities and the need for realistic actuation and interpretation of sensory information. Although test and development platforms such as GameBots (Kaminka *et al.* 2002), which provides an interface into the Unreal Tournament game engine, are available, it is not always possible to access commercial Internet games in this way.

The experience of commercial programmers is consistent with these views. Tomlinson, Davis and Assadourian (2003) describe much of the current implementation of AI in the games industry as smoke and mirrors . While there is a strong desire for academic AI to be applied, it is rarely possible to do so and intelligent behaviour is often attained in an *ad hoc* way through clever scripting and variable response. The point is made that the designer will often exploit psychological factors to achieve believable effects.

Certainly, there is a demand for AI in games and good AI can be a major selling point - and the requirement is human-type mundane agent behaviour, not optimal response and invincibility. This is commercial pressure and is likely to focus attention on what exactly constitutes human behaviour and how it can best be simulated.

The objective in the development of realistic agent behaviour is effectively to impart mundane abilities. A test platform that develops mundane skills is therefore relevant to the games industry.

PERUDO[4]

The game of perudo has not previously been used in AI research, but in this paper we describe its potential. The game is played by between two and six players each of whom start with five standard dice. At the beginning of a round, the dice are shaken in a cup then thrown onto the table hidden under the now upturned cup. The players may look at their own dice after which a bidding cycle begins. The objective of the game is to estimate the quantity of any dice face value when the cups are lifted and all the dice revealed at the end of the round. Ones are wild and cannot be bid. When the dice are revealed at the showdown, the ones take the value of the dice face bid. Once a player has bid, the next player must either increase the quantity of dice or the dice value (or both). If they do not wish to increase the bid, the player may call the previous bid. When this happens, all the dice are revealed. If the quantity bid is not there, the bidder loses a die; if there is at least that number of dice, the caller loses a die. No other players are involved. Dice are re-shaken and a new round starts. The winner is the last player with a die or dice remaining. A typical game start position is shown in Fig. 1.

The basic strategy is complex because of the prevalence of deception. In principle, each bid reveals information about the player s dice but it is difficult to judge whether or not a player is bluffing unless an accurate player model is available; the human player automatically creates and refines opponent models through the course of the game. The difficulty here is to develop a machine simulation that will play with reasonable skill given these complexities.

It is necessary to demonstrate that the game is suitable for AI research. We first propose and test the negative view that machine play based on probability but with sufficient variation to make it unpredictable is equal to or more effective than the intuitive play of the human. If this assumption turns out to be correct then the game is of little value to AI research since it is

[4] The rules of the game have been simplified slightly for this project and the terminology standardised. The FISA organisation (http://www.perudo.org) promote a variant called Santaba. However, the rules and the nomenclature are unnecessarily complex for our purpose. They also offer a mechanism for playing online, but it is unsatisfactory for research purposes.

then unnecessary to simulate the processes followed by the human. However, experience would suggest that this is unlikely to be true. Fixed strategies, even if optimised by game theory, are inflexible and are susceptible to a variety of bluffing strategies. In human play, an accurate opponent model and flexibility are an absolute necessity.

If, as expected, an artificial player operating on probability performs inadequately, the game can then be used to explore a variety of AI techniques, for example using artificial neural networks (ANNs) for learning. The basic idea there is to use a feed-forward network of multi-layer perceptrons with a log-sigmoid activation function and backpropagation training to approximate the characteristic value function of the game. This is not without challenges – what are the appropriate input vectors and how is the error determined? Tesauro (1995) describes these problems in relation to the game of backgammon and on the basis that the 'goal of learning is to generate the optimal action leading to maximal reward' appl ied a variant of backpropagation training called temporal difference (TD) learning where an estimation of value is made at each stage, not just at the end when the game is won or lost. This speeds up learning considerably, especially when used with extended sessions of self-play. The resulting program, TD-gammon proved very successful but the claims for TD learning were later criticised (Pollack, Blair and Land 1996). It was stated that a simple evolutionary model could result in the same parameter optimisation. The experiments of Kotnik and Kalita (2003) appear to back this up. The most significant issue in the success of TD-gammon is now considered to be the learning through self-play.

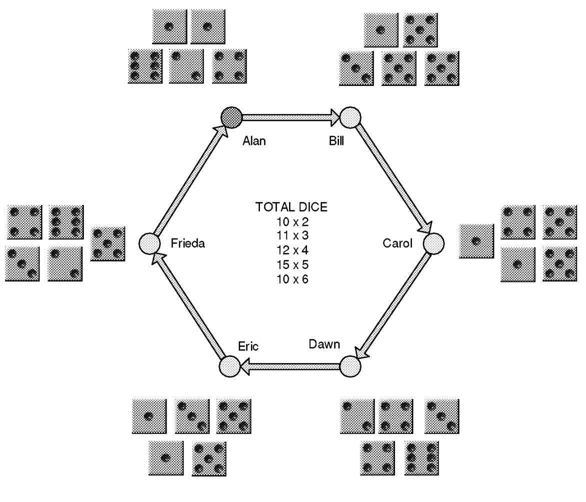


Figure 1: Start of a Typical Six-player Perudo Game - Players see only their own Die

In developing a machine to play perudo, we will as a starting point explore a crude evolutionary scheme to optimise a small number of significant game parameters.

In the absence of expert knowledge, a reasonable strategy is to consider only bids which fall between a minimum and a maximum probability of success, P_{MIN} and P_{MAX}. The selection strategy may simply be to choose randomly between the two thresholds or it may be considerably more complex. Heuristically, the safe limit should be of the order of 0.7 and the risky limit would be about 0.3. The machine will call if the available responses push it beyond the lower threshold.

The machine must alter the distribution function in response to each bid. Some parameters need to be introduced to translate a bid into a sensible alteration of the probability function. In this simple system, the parameters will be global, but they should really be player specific whereupon they become a rudimentary opponent model. A parameter P_{TRUTH} is introduced as a global measure of the probability that the value bid actually reflects a better than average number of the bid die face value in the bidders hand. A reasonable initial value would be 0.7. With each bid the probability distribution table is only altered this fraction of the time. Bids interpreted as a bluff (1 - P_{TRUTH}) are ignored. Finally, a weight parameter W is introduced to quantify the increase in the die face value as a result of a bid interpreted as truthful. An initial estimate of 1 would seem reasonable (it is actually expressed by a smaller number, e.g. 0.2, times the number of dice the bidder has on the table).

The optimal parameter values can be deduced or augmented by applying game theory, and the truth and weight parameters can be modified by a post-mortem analysis of each round. This is certainly this one way to explore the use of AI; however, in this experiment we want to show that the machine can learn the optimal values from game results by adapting the parameters through self-play. This is an example of reinforcement learning.

AI is used to learn the optimal parameters, but the optimised machine does not use AI. We will simply have maximised the use of probability through learning rather than analysis. We would expect the machine to perform badly against humans if perudo is really a game that uses uniquely human abilities, and that AI must be incorporated into the strategy and general play for a strong artificial player is to be created.

IMPLEMENTING A LEARNING SYSTEM

Fundamentally, the decision making is based on probability. In any game, up to 25 dice may be hidden from a player. We can construct a probability function for the success of any bid based only on the hidden dice – the certain knowledge of the contents of the player's own hand is added afterwards. The appropriate function is the cumulative binomial distribution (Weisstein, 2004). The probability of there being n or more of any dice value from a total of N hidden dice is

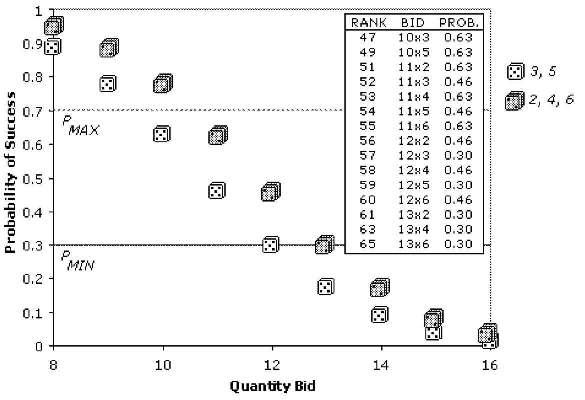
$$\begin{aligned} P(n \mid N) &= \sum_{k=n}^{N} \binom{N}{k} \frac{2^{N-k}}{3^{N}} \\ &= I_{1/3}(n, N-k+1) \end{aligned} \tag{1}$$

where

$$I_x(a,b) = \frac{B(x;a,b)}{B(a,b)} \tag{2}$$

B(a,b) is the beta function and *B(x;a,b)* is the incomplete beta function. *P(n|N)* cannot be described analytically but because the output is discrete and 350 values are sufficient for the game, it is best to pre-calculate the set {(n, N, P(n|N)), $n \in \Im$, N∈ℨ, 0<N≤25, 0≤n≤N} and store as a table.

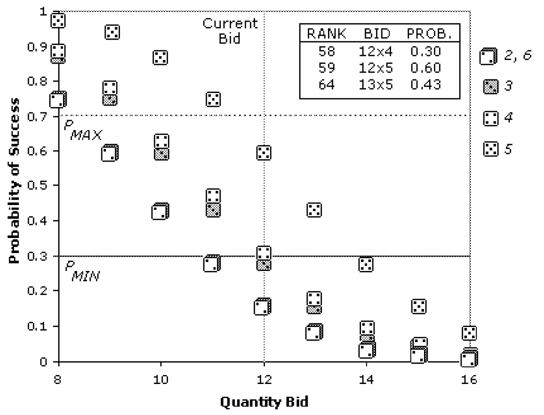
Looking at the position in Fig. 1, Alan's initial probability distribution for the game is shown in Fig. 2 with the ranked bids that fall between the threshold probabilities, P_{MIN} and P_{MAX}.



RANK	BID	PROB.
47	10x3	0.63
49	10x5	0.63
51	11x2	0.63
52	11x3	0.46
53	11x4	0.63
54	11x5	0.46
55	11x6	0.63
56	12x2	0.46
57	12x3	0.30
58	12x4	0.46
59	12x5	0.30
60	12x6	0.46
61	13x2	0.30
63	13x4	0.30
65	13x6	0.30

Figure 2: Alan's Initial Probability Distribution

The ranking is the strength order of each bid beginning from 1x2. Assume Alan bids (at random) 12x4. Bill also has an initial model of the distribution (different to Alan's), but if this bid is accepted as truthful, Bill increases the estimated number of fours in the distribution by 1, but also decreases the estimate of the other values, in this case 0.25 each although this assumption underestimates the strengths of ones. This is implemented by a linear extrapolation of the probabilities. Bill's model is shown in Fig. 3. The inset table shows his available bids (including the current bid).



RANK	BID	PROB.
58	12x4	0.30
59	12x5	0.60
64	13x5	0.43

Figure 3: Bill's Probability Model following Alan's Bid

Bill estimates that a call would have a 0.7 chance of success. However, this immediately puts him in a position to lose a die. 12x5 is safe, but the randomisation will sometimes pick up 13x5. The game continues with Carol probably raising the fives and Dawn calling. The automata strategy thus seems reasonable.

The optimal parameter settings are obtained by first allocating to each player a random value for the four parameters. One epoch of training consists a single game for each permutation of player position (720 games in all for six players[5]). Training is conducted off-line with each 'game' completed in about 0.01 s using a high-spec Pentium 4 PC. At the end of each epoch, the parameter values for the losing players are altered towards those of the winner by a factor α, the leaning rate (0<α<=1). To act as an evolutionary seed, Gaussian noise is added to all the parameters. The experiment is then repeated until the parameter values converge. After convergence, the parameters are given a 'kick' to exclude the possibility of entrapment at a local minimum.

RESULTS

To confirm the game does require skill, six artificial players with various distinct playing characteristics were created and the changes in rating during an extended self-play experiment are shown in Fig. 4.

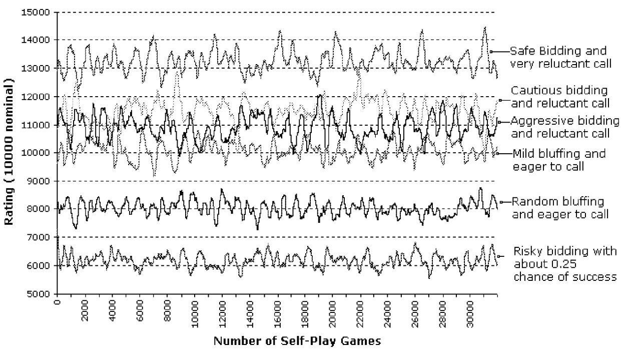


Figure 4: The Performance of Stereotypical Artificial Players.

There is a clear distinction, but there is an inherent variation of ±5%. This reflects the effect of chance in the game – the simple automata are not able to offset this through bluff and extracting accurate information from the opponent bids.

An evolutionary system where the parameters are adjusted towards that of the winner in proportion to the rating difference quickly converges to values of P_{MAX} = 0.85, P_{MIN} = 0.39, P_{TRUTH} = 0.82, and W = 0.08. Fig. 5 shows how these values are sustained over a large number of games even with the addition of ±3% Gaussian noise to each parameter at each epoch.

The results are interesting and indicate that the system allocates little value to each 'correct' bid (probability model is adjusted by the number of dice times W, a lowly value of 0.4 at the start of a game). The system largely ignores the bidding information and the performance is thus strongly affected by the randomness of dice set.

It is no surprise therefore that the initial tests of the optimised model against humans resulted in a poor performance. The opposition quickly identified its weaknesses and were able to deceive the program into calling safe bids.

[5] This is necessary because the rating system does not presently manage the nuances associated with the playing position of the participants. By considering all permutations, the effect is eliminated.

It is possible to extend the parametric model, to explore the best bidding strategy between the bidding limits and to optimise the calling process, but, based on the initial tests, this is unlikely to result in expert play.

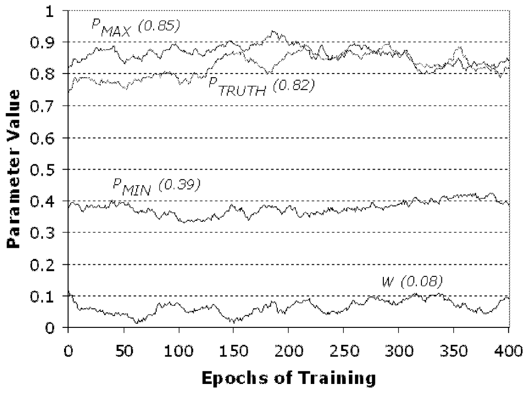


Figure 5: The Effect of 3% Noise on the Optimal Parameter Values over 288,000 Games

CONCLUSION AND FURTHER WORK

Perudo is a very easy game to model and tests so far have shown there is a significant skill level and suggests particularly human characteristics are required for good play. It is therefore suitable for AI research, particularly the exploration of the mundane aspects of human intelligence. Experiments still have to be completed to evaluate the actual level of skill in the game (noise) and the effect of luck[6]. The random factors inherent to the game can be quantified by the rating 'noise' of the best players. The concept of performance noise is useful and can be measured through the RMS level of the noise over the trend line[7]. It is difficult to compare machine with computer at this early stage because of the complexity of human play: good human players change strategy frequently, and tend to play the opponents rather than the dice. Limited calculation is involved and there is little correlation in performance with intellectual ability, however good players play consistently well.

Only limited progress is possible with self-play experiments. For further progress, it is necessary to engage a large number of people to compete with the machines. To this end a web site has been created to enable people to play. The game can be accessed at www.playperudo.com from a browser (from the 1st of December 2004). It is necessary to register in order to obtain a rating – enter your details and a random password will be emailed. The game is realized on the server as a set of global arrays. 33 functions for manipulating the arrays are implemented as ASP files. A user interface for human play or creating automated agents is produced by posting data to these functions in the correct sequence. The game flowcharts and the protocol that artificial players must follow are available on request. Researchers may therefore create their own artificial players and play against humans and other computers. The stable rating system enables the effect of changes to the automata behaviour to be quantified.

ACKNOWLEDGMENTS

The theoretical aspect of this work was supported by the University of the Highlands and Islands Sabbatical Scheme and Lews Castle College. Western Isles Enterprise, Stornoway, Isle of Lewis provided assistance with the practical development of the perudo test platform.

REFERERNCES

Billings D, Papp D, Schaeffer J, Szafron D, 'Poker as a Tes tbed for Machine Intelligence Research", In Advances in Artificial Intell igence (Mercer R. and Neufeld E. eds.), Springer-Verlag, pp 1-15, (1998)
Billings D, Davidson A, Schaeffer J, Szafron D, 'The Cha llenge of Poker", Artificial Intelligence, **134**(1-2), 201 (2002)
Billings D, Burch N, Davidson A, Holte R, Schaeffer J, Schauenberg T, Szafron D, "Approx imating Game-Theoretic Optimal Strategies for Full-scale Poker", Proceedings of the 2003 International Joint Confe rence on Artificial Intelligence (IJCAI-03). (2003)
Bringsjord S., Lally A., 'Chess isn't Tough Enough: Be tter Games for Mind-Machime Competition", *Proceedings of the AAAI-97 Workshop on Deep Blue vs. Kasparov: The Significance for Artificial Intelligence* (ed. R. Morris), Providence, RI, p14 (1997).
Kaminka G A, Veloso M M, Schaffer S, Sollitto C, Adobbati R, Marshall A N, Scholer A, Tejada S, 'GamеBots: A Flexible Test Bed for Multiagent Team Research", Commun ications of the ACM, **45**(1), 43 (2002)
Kotnik C, Kalita J, 'The Significance of Temporl -Difference Learning in Self-Play Training TD-rummy versus EVO-rummy", Pr oceedings of the Twentieth Conference on Machine Learning (1MCL-2003), Washington DC (2003)
Laird J F, van Lent M, 'Human -Level AI's Killer Application: Inte ractive Computer Games", AI Magazine, Summer (2001).
Macleod A, 'Perudo as a Test Platform for Artificial Inte lligence", *To be published.* (2004)
Nareyek A, "AI in Computer Games", ACM Queue 1, **10**, 58-65 (2004a)
Nareyek A, 'Computer Games – Boon or Bane for AI Research?", KI **18**(1), 43(2004b)
Pollack J B, Blair A D, Land M, 'Coevolution of a Bac kgammon Player", Proceedings of the 5th International Conference on Artificial Life (1996)
Schaeffer J, "A gamut of games", AI Magazine, **22**, 29 (2001)
Tesauro G, "Temporal Difference Learning and TD -Gammon", Communications of the ACM, **38**(3), 58 (1995)
Tomlinson S L, Davies A, Assadourian S, 'Working at thi nking about playing or a year in the life of a games AI programmer" in GAME-ON 2003 Eds Medhi Q, Gough N, Natkin S, EUROSIS, GHENT (2003).
Weisstein E W, 'Binomial Distribution", MathWorld – A Wolfram Web Resource.
http://mathworld.wolfram.com/BinomialDistribution.html (2004)

[6] It is believed the significance of chance is reduced (compared to poker and bridge) because all access the same dice set. Certainly, a hand such as five one's give an advantage in that they reveal how the distr ibution is skewed, but deception can weaken these advantages. What is clear is that a player with less dice is strongly disadvantaged and the loss of a single die (along with the accompanying psychological factors) can result in a rapid slide to elimination.
[7] In sport, golf would be a game with high noise; swimming and athletics would have low noise. In high noise sports, unknown players are more likely to be surprise winners.

FORCE NON-PLAYER CHARACTERS TO LEARN

William Tambellini
KYNOGON
12 rue Chaussée d'Antin
75003 Paris
France
E-mail : William.Tambellini@kynogon.com

Cédric Sanza and Yves Duthen
IRIT Laboratory
118 route de Narbonne
31062 Toulouse CEDEX
France
E-mail : sanza@irit.fr

KEYWORDS

Video Games, Non-Player Characters, Behavior learning, High-Level Decision, Evolutionary Algorithms.

ABSTRACT

In commercial video games, there are two kinds of entities regarding Artificial Intelligence : Player Characters (PC) and Non Player Characters (NPC). The first are entities controlled by human players, the second are the ones controlled by the core game engine. In "Multiplayer" or "Shoot Them Up" games, NPC and PC are symmetric, they have a full freedom and a long life time.

We propose here to work on decision making for the top-level layer, the one which decides when and why the NPC will adopt high-level behaviors such as wandering, fleeing, following, shooting, hiding, etc.

Multiplayer and Shoot Them Up games do need efficient high-level decision rules. These rules are difficult, sensitive and time-consuming to write by hand.

Better than that, we propose to force NPC to learn high-level behavior. For this, we use evolutionary algorithms to obtain rules of behavior thanks to an off-line learning session.

INTRODUCTION

In nearly every commercial video game (VG), for personal computer or console platform, there are two kinds of entities regarding Artificial Intelligence : Player Characters (PC) and Non Player Characters (NPC). The first ones are entities controlled by human players with mouse, keyboard and/or joy pad. The second ones are the ones controlled by the core game engine. For example, for car racing games, NPC are the other cars. For strategy games, NPC are the entities the player try to eliminate.

Action & Adventure games propose to the player a mix between action (very often a shooting action) and adventure (a more or less flexible scenario leads the game). In that kind of games, NPC can have several roles regarding the player : they can be partners, tourists, hostages, etc. They are above all enemies for the player.

In this case, NPC enemies have two kinds of situation according to the game style. In scripted titles as (Half-Life 2000) solo mode or (Max Payne 2001), they are present to participate to a written scenario and do not have to interfere the story. Such NPC have a short life time and no liberty of move and decision. In this paper, we do not consider this kind of NPC.

On the contrary, two other game styles provide more complex NPC. The first are "Shoot Them Up" games as (Unreal Tournament 2004) or (Quake III 2001). The second are bots for "Multiplayer" modes as "death match", "team death match" or "capture the flags" modes. In these kinds of games, NPC and PCs are symmetric. NPC have the same role as players, have a full freedom and a longer life time. We propose to work on this kind of NPC. Such NPC do need a high-level decision system, contrary to scripted games. The question is now how to code this system.

NON-PLAYER CHARACTERS BEHAVIOR

VG developers have to tell NPC which behavior to adopt according to the situation. Fig.1 shows a simple example :

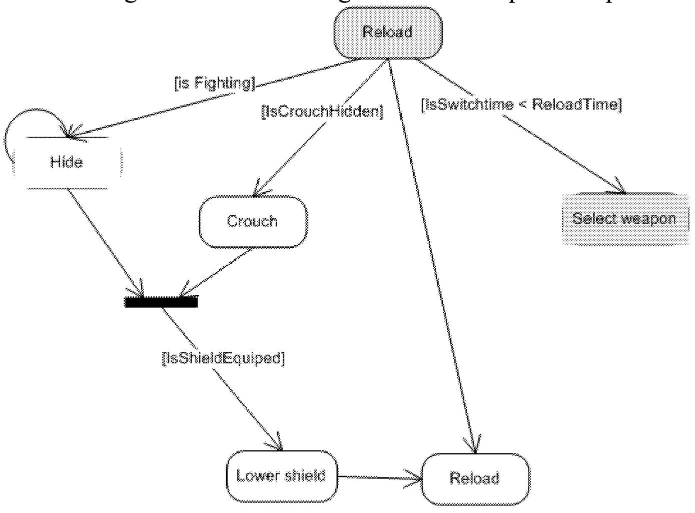


Figure 1 : Example of "Reload" behavior (B.O.S. 2004)

This could be mathematically assimilated to a Finite State Machine (FSM) (Houlette 2003).

Some drawbacks of such techniques are explained in (Manslow 2002) :

- limiting the scope of the problem to a finite set of entities is dangerous ; if the system becomes complex, you can easily forget a state;
- defining a complete FSM is an extremely long process ;
- when system complexity grows up, developers should expect to see an even greater list of events and states to support it. As logic errors are common in programming, event errors are common in FSM. This can easily become a nightmare.

Rather than building the whole base of rules by hand, we propose to force NPC to learn how to play. Then, after the learning session, we could export these rules to an easy formulation language as FSM that game designers could modify. This could improve NPC quality. Precisely, (Tozour 2002) expect that smarter NPC lead to funnier interaction and so, funnier games.

STATE OF THE ART

Learning for Video Games

(Kirby 2003) shows that learning, independently of the used technique, is difficult to be applied in VG.

This issues make appear two different learning tasks. The first is the one made during the game, called **on-line**, and the second is the one made before game commercial release, learning realized by developers during production sessions, called **off-line** learning.

The first one suffers from a maximum of negative critics from VG developers and with a good reason. Mostly all uses of online learning resulted in unplayable VG with bad game plays or made NPC learn just a little or not at all (Manslow 2002). However, (Spronck et al. 2003) shows that this could be useful for dynamic scripting in Role Playing Games.

On the other side, off-line learning techniques are difficult to tune and to use to get the desired behavior. Nevertheless, this kind of learning is safe because it minimizes the appearance of non-desired behaviors during the game. This security can be obtained by letting the game designer delete, select and reinforce the rules of his choice after learning session. Off-line learning also presents advantages to enhance NPC behavior and to find holes in the game engine (Spronck 2002 et al.).

Off-line learning should construct dynamically and automatically a more or less complete and tunable set of rules. It should be much more pleasant for the game designer to use this engine than writing each rule by hand. It allows game designer to have a total control and to keep only few rules if he'd like to.

To provide a such offline learning, the evolutionary algorithm we have chosen is called Classifier Systems (CS).

Classifier Systems and Genetic Algorithms

A Classifier is simply a 'condition-action' rule (Holland et al. 99). A simple example could be : "If strong then attack".

From the point of view of Genetic Algorithms (GA), these two parts represent two chromosomes. Each chromosome contains some genes. Each gene represents a perception or action unit. In our example, "strong" is the only perception gene available in the perception chromosome and it could be boolean ("strong" or "weak") or discreet (strength between 0 and 10).

By consequences, a CS is a system handling a pool of classifiers, i.e. rules. To interact with the environment, here the game engine, CS clients (NPC) send and receive messages from and to the CS. CS will answer by sending the best action chromosome corresponding to the perception according his data base. Then, NPC apply received action chromosome.

At the initialization time, the whole rules are chosen randomly. Then, we let NPC act according to the rules of the CS. When we detect that an NPC has done a good or bad action, we reward or punish the last rules used by this NPC.

To make evolve the whole population, at a chosen moment of the learning session, we have to call a GA for mixing the population of classifiers. This will kill the worst, keep the best, and generally produce better classifiers. (Buckland 2003) and (Buckland 2002) list the main mechanisms needed (selection, crossover, mutation).

Classifier Systems and Video Games

Only few works deal with VG and CS. (Robert et al. 2002) presented one of the first works on using CS for NPC in VG for action selection. However, this work has been particularly used for Massively Multi-players Online Role Playing Games (MMORPG). This context could not be compared with ours. Indeed, NPC in MMORPG usually do not have full freedom of move and are not in the same position as players. However, (Robert et al. 2003) presents an application for multiplayer games for "Capture The Flags" modes. This is interesting particularly for new heuristics issues in VG.

(Demasi et al. 2003) presents works on both online and offline learning, but using GA, not CS. Nevertheless, they build their chromosomes in the same way CS are modeled : <condition><action>. Whereas we propose to work only on pure offline learning, it deals with both learning types even on mixing online and offline evolutions. To finish, it has the same point of view for basic perception (like distance to enemy) and high level behavior (like chase player) as our experiment. All these points can help us with the global issues of this kind of works, but do not help us about pure CS choices.

(Sanza et al. 2000) & (Sanza et al. 2001) present a simulation of a soccer game. The NPC have to decide at each time what to do, on the basis of both an individual fitness value and a collective reward. Each football player is evaluated according to his efficiency for his team.

(Heguy et al. 2002) presents a generic CS for real time task learning. Even if this work mainly deals with the purpose of artificial life, the use of CS in dynamic environment for behavior learning is similar to the pure VG context. It is used here in a virtual basketball game. We find in this work the same idea of collective reward compared to individual reward.

Despite these works, unsuccessful and repeated trials of using GA for VG (Manslow 2002) let consider that the use of these algorithms for pure VG is not obvious. This fact is quite strange since CS have a major advantage : results are readable.

To finish, we do not have to forget that the game designers want to keep control on behavior rules for tuning purpose. This means that the results have to be understandable and modifiable by him. Precisely, CS have the advantage of giving understandable results, contrary to Neural Networks. Hence, CS are well suited to VG for high-level decision. A game designer, after learning, can easily configure, change and extend the base of rules thanks to the understandable formalism of a classifier, 'if … then …', contrary to Neural Network, for example.

EXPERIMENT

The test bed used for this experiment contained (Fig.2 & 3) :
- a world, kind of maze with tunnels and holes ;
- several termites, with perception and action capabilities ;
- a Hunter, with perception and action capabilities ;

Figure 2 : Test bed opposes several termites versus one Hunter

Each entity could have all kind of basic perception capabilities : number of visible termites, visibility and path finder distance to the Hunter or to the furthest and nearest termite.

For this experiment, we use RenderWare Artificial Intelligence (RWAI 2004). RWAI is an Artificial Intelligence development library for VG, belonging to the RenderWare suite (RW 2004). RenderWare is a middleware for VG development, proposing 5 libraries : Graphics for 3D and 2D rendering purpose, Physics, Artificial Intelligence, and Audio.

RWAI separates entity actions by low and high level decisions. The low level deals with entity direction, speed and shooting parameters. The high level decision, called Brain structure, is used for selecting global behavioral agents as Wander, Flee, Attack, etc. This work deals with high level decision only.

Then Hunter and termites have the choice between these behaviors :
- wandering using a random destination point ;
- attacking an entity of their choice ;
- fleeing or hiding from an entity ;

In this experiment, Hunter entity has a hard coded behavior which is:

"If Termite(s) seen then Attack Nearest Termite else Wander"

All the termites are controlled by the same Classifiers System. Figure 4 presents the modeling we used for their classifier representation.

(Sweetser 2003) explains the difficulty to use Evolutionary Algorithms for games, particularly the difficulty of representing possible solutions in genes, chromosomes and genomes, finding a good fitness function and finding well-fitted tuning parameters.

The perception issue deals with the **under** and **upper perception** problem. A termite is in under perception if it does not have enough perception to solve the fitness. For example, if it has only the number of visible termite perceptual gene. A termite is in upper perception if it has too much perception, for example, 10 perceptual genes, whereas the same results can be found with only 4 genes. Major issues in this kind of work are to minimize the number of perception genes without being in under perception state.

Perception chromosome :
<type> Meaning [Min ; Max]
<int> Number of visible termite, [0 ; 10] ;
<int> Number of close termite, [0 ; 10] ;
<int> Hunter visibility, [false ; true] ;

Action chromosome :
<int> action index, [0 ; 4] with
- 0= No Move ;
- 1= Wander ;
- 2= Follow nearest Termite ;
- 3= Attack Player ;

Figure 4 : Representation of a termite classifier

The game rules have been set to provide an inequality between the hits of the termites and of the Hunter. The Hunter kills a termite in only one shot. When a termite hits the Hunter, this last loses only a part of his life. Moreover, the Hunter is faster to hit than the termites. This means that when one termite is in front of the Hunter, it will always die as Hunter is the first to hit. This will oblige the termite to search for a better strategy than : “If Hunter Visible then Attack Hunter”.

For this, we used a simple reward system, which consists of rewarding the last classifiers of the termite which hurts the Hunter.

Learning session is required for letting CS evolve. Then, a test session is necessary to calculate the CS efficiency. Both sessions last 30 minutes.

Considering that the spawning is a way to place entities in the world (position and direction), a random spawning was used for the 12 termites (random places) and a static spawning for the Hunter (always the same place). Number of classifiers was set to 40.

The score of the termites increases when they kill Hunter and the Hunter wins when he kills all the termites. A measure of the global score is equal to the score of the termites divided by the score of the Hunter.

RESULTS

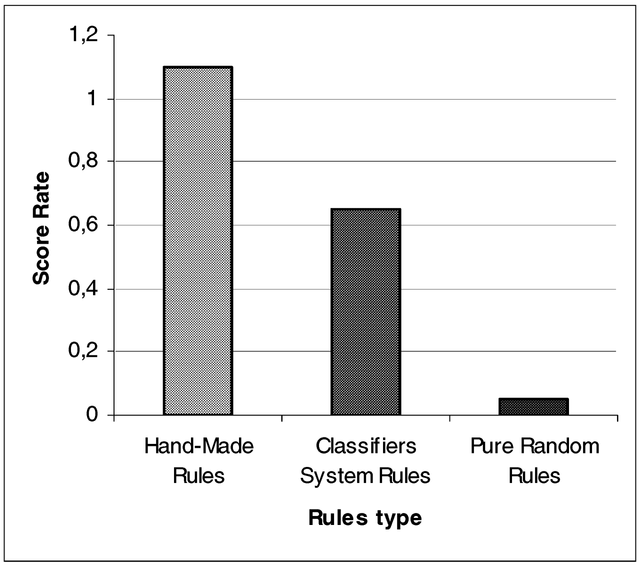


Figure 5 : Mean score rate as a function of the type of rules

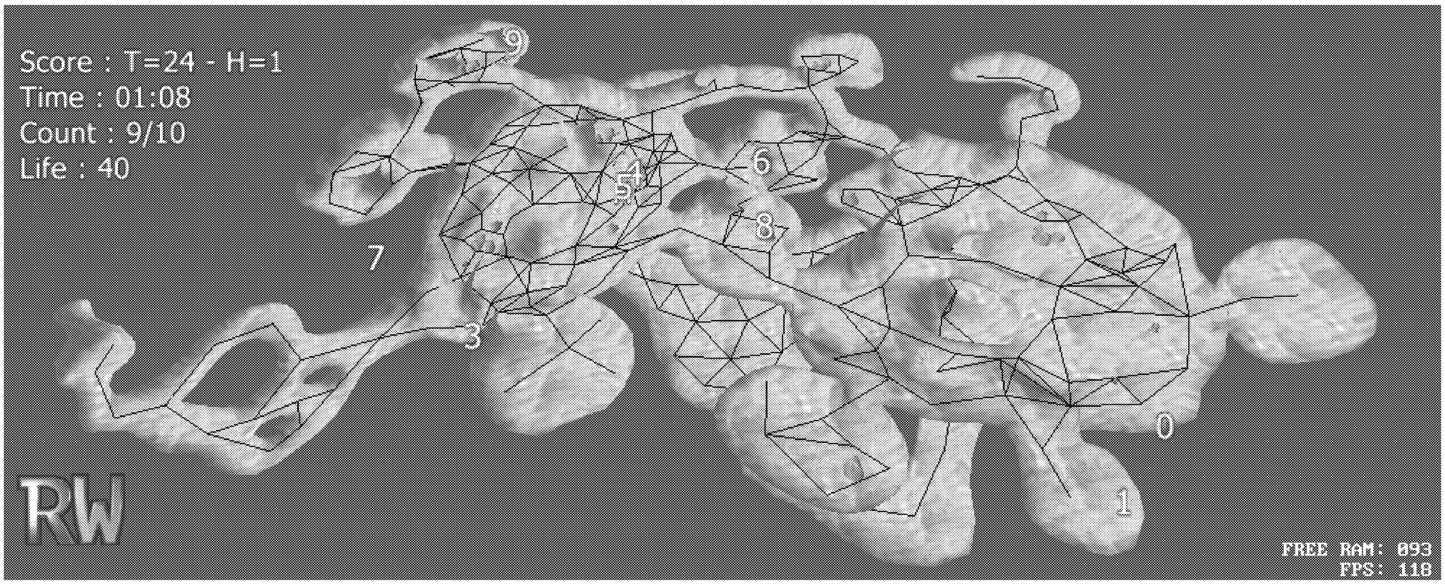


Figure 3 : The map of the experiment with the pathfinder graph as reference point and termites ID in white

A summary of our main result is presented in Fig.5. We see that Hand Made Rules exceed learned ones (mean obtained after 10 testing sessions).

Although, after learning sessions, CS effectively gave efficient rules like these ones :

```
"If    NumberOfVisibleTermite = 5
       and NumberOfClosedTermite = 3
       and HunterVisible = true
       then AttackHunter"

"If    NumberOfVisibleTermite = 3
       and NumberOfClosedTermite = 0
       and HunterVisible = true
       then FollowNearestTermite"
```

We can also see that learning rules are better than pure random rules, showing that the learning session has been useful to find efficient rules even if not perfect.

One of the resulted behavior is a grouping behavior that makes termites group before attacking the Hunter. Indeed, a lonely termite which attacks the Hunter has no chance to hurt it because of the hitting speed of the Hunter.

DISCUSSION

We have shown that using CS for NPC decision systems needs several requirements. The main difficulties are building well-chosen genes, and getting an adequate fitness function using rewards and punishments.

There are probably two main reasons for the results we obtained.

The first one is about “non-deterministic” systems. It seems that the game we used is not deterministic. That means a same action in same conditions does not always result in the same consequence. This is explained by uses of randomness (such as in the “Wander” behavior). The results could be also explained by pure game engine noises like physics between entities, processor unit load, etc. In other words, external parameters make the consequences of the NPC decision vary.

This instability could explain why the CS rules hardly converge.

The second reason is caused by the fitness. Nearly every supervised learning algorithms build their knowledge thanks to an efficient fitness function, i.e. reward or punishment functions. In non-scripted video games, it seems the only reward we can do with CS is when a sub-goal has been achieved by a NPC. GA & CS are generally used with accurate and stable fitness functions. In such video games, we should only have a rare and fuzzy reward.

We have seen that the results of CS learning in a highly non -deterministic environment is quite week.

The next step is now to work on different kinds of CS that could deal with such non-deterministic commercial video games.

CONCLUSION

The use of learning algorithms such as Evolutionary Algorithms could be a useful way to deal with NPC in non-scripted video games. It could reduce the working time in production period and so reduce cost of production. It could too advice game designers about holes in their game engine and increase the intelligence of NPC. Better than doing stronger NPC, we advice to do smarter NPC, which would increase the game fun.

REFERENCES

Buckland M. 2003. "Building Better Genetic Algorithms" In *AI Game Programming Wisdom 2,* (Eds.) S.Rabin, Charles River Media, 649-660.

Buckland M. 2002. *AI Techniques For Game Programming,* (Eds.) Andre LaMothe, Premier Press.

Demasi P., Cruz A. J. 2003 “Online Coevolution for ActionGames.” Int. J. Intell. Games & Simulation 2(2), 80-88

Heguy, O.; C. Sanza; A. Berro; Y. Duthen. 2002. “GXCS: A Generic Classifier System and its application in a Real

Time Cooperative Behavior Simulations“ In *The International Symposium and School on Advanced Distributed System* (ISADS’02), 11-15

Holland, J. H.; L.B. Booker; M. Colombetti; M. Dorigo; D. E. Goldberg; S. Forrest; R. L. Riolo; R. E. Smith; P. L. Lanzi; W. Stolzmann; S. W. Wilson. 1999. "What Is a Learning Classifier System ?" In *Learning Classifier Systems*, 3-32

Houlette, R and D. Fu. 2003. "The Ultimate Guide to FSMs in Games." In *AI Game Programming Wisdom 2,* (Eds) S.Rabin, Charles River Media.

Kirby, N. 2003. "Getting Around the Limits of Machine Learning" In *AI Game Programming Wisdom 2,* (Eds) S.Rabin, Charles River Media, 603-611.

Manslow, J. 2002. "Learning and Adaptation" In *AI Game Programming Wisdom,* (Eds) S.Rabin, Charles River Media, 557-566.

Robert G. and A. Guillot 2003. "MHiCS, A Modular And Hierarchical Classifier Systems Architecture For Bots". In Game-On 2003, Mehdi, Q., Gough, N. and Natkin, S. (Eds). *140-144.*

Robert, G.; Portier, P.; Guillot, A. 2002. "Classifier systems as 'Animat' architectures for action selection in MMORPG". In *Game-on 2002* (Eds) Mehdi, Gough and Cavazza. 121-125.

Sanza, C. ; C. Panatier ; Y. Duthen. 2000. "Communication and Interaction with Learning Agents in Virtual Soccer". In *Proceedings of Virtual Worlds 2000*, (Eds) J.-C. Heudin, 147-158.

Sanza, C., O. Heguy, Y. Duthen. 2001. "Evolution and Cooperation of Virtual Entities with Classifier Systems" In CAS'2001, Eurographic Workshop on Computer Animation and Simulation, Springer Computer Science.

Spronck, P., I. Sprinkhuizen-Kuyper and E. Postma. 2002. “Improving Opponent Intelligence Through Offline Evolutionary Learning.” In *International Journal of Intelligent Games and Simulation* (Eds. N.E. Gough and Q.H. Mehdi), Vol. 2, 20-27.

Spronck, P., I. Sprinkhuizen-Kuyper and E. Postma 2003. “Online Adaptation of Game Opponent AI in Simulation and in Practice” In *Proceedings of the 4th International Conference on Intelligent Games and Simulation (GAME-ON 2003)* (Eds Quasim Mehdi and Norman Gough), EUROSIS, Belgium, 93-100.

Sweetser P. 2003, "How to Build Evolutionary Algorithms for Games", In *AI Game Programming Wisdom 2,* (Eds) S.Rabin, Charles River Media, 627-637.

Tozour P. 2002 "The Evolution of Game AI" In *AI Game Programming Wisdom,* (Eds) S.Rabin, Charles River Media, 3-15.

VIDEO GAMES, SOFTWARE, LIBRARY

(B.O.S 2004) Bet On Soldier, Kylotonn 2004, http://www.betonsoldier.com/

(Half-Life 1999) Half-Life, Sierra 1999, http://www.sierra.com

(Quake III 2001) Quake III Arena, ID Software 2001, http://www.idsoftware.com/

(Max Payne 2001) Max Payne, Rock Star Games 2001, http://www.rockstargames.com/maxpayne/

(RW 2004) RenderWare, Criterion Software, http://www.csl.com, http://www.renderware.com

(RWAI 2004) RenderWare Artificial Intelligence, Kynogon, http://www.kynogon.com

(Unreal Tournament 2004) Unreal Tournament, Epic 2004, http://www.unrealtournament.com/

INTEGRATED ON- AND OFF-LINE COVER FINDING AND EXPLOITATION

Gregory H. Paull
Secret Level Inc.
123 Townsend St., Suite 300
San Francisco, CA 94107
greg@secretlevel.com

Christian J. Darken
MOVES Institute and
Computer Science Department
Naval Postgraduate School
Monterey, CA 93943
cjdarken@nps.edu

KEYWORDS

AI, First-Person Shooter (FPS), Finding Cover

ABSTRACT

Most first-person shooter game AI's are poor at quickly getting out of lines of fire. AI agents that pass up obvious opportunities to hide or take cover can ruin a game's immersiveness. We will present a system that combines the sensor grid algorithm (Darken 2004) with pathnode-based information. This system relies on cover information stored in the path nodes placed throughout the level and performs a focused run-time search in the immediate vicinity of the agent if the node based information is insufficient. This allows it to be both fast and able to react to changes in the environment.

BACKGROUND

Taking cover is a universal human response to threat. However, it is not innate; children must learn to hide. It is also not totally understood; psychologists are still investigating a critical part of hiding, which is what we know of what other people can or cannot see (Kelly et. al.). Nonetheless, nearly everyone is able to quickly and effectively duck to safety when threatened. The use of cover is also not purely defensive in nature. A person can be taught to take advantage of cover when moving to make invisible shifts in their position and to minimize their exposure to danger when shooting.

The ability to use cover effectively is one of the skills that separate the best real players in first-person shooters from the average players. Unfortunately in the current state of gaming it is also one of the ways to distinguish between live players and game agents. Game agents do not use cover effectively. Typical problems include running right by good sources of cover, and failing to consistently take the most direct route to safety.

This paper describes an approach that relies on a combination of data stored in waypoints throughout the level, and a focused dynamic (i.e. run-time) search in the immediate vicinity of the agent when the node data is insufficient. Waypoints used throughout the level for pathfinding contain information that is used by the system to help make more "intelligent" decisions regarding concealment and cover. This information includes data such as appropriate stance to assume, direction in which the cover provides protection, and types of weapons this cover provides protection from. Pre-computed visibility information is also stored in each node that greatly increases run-time performance. If an adequate waypoint is not immediately found the sensor grid algorithm is run to find a safe destination for the agent. This system is both fast and able to react to changes in the geometry of the environment that occur during play. We first describe some related techniques already in the literature. Then, we give a brief overview of the sensor grid algorithm which is described in detail in Darken et. al. 2004. Next, we describe the various types of waypoints used in the system. Finally, we describe a few extensions that could be made to the system in the future.

RELATED WORK

Previous approaches to the hiding problem involve searching a fixed set of potential hiding locations. Often this is the set of waypoints used to plot movement.

Typically, navigation is accomplished in high-resolution shooter games by the use of a set of locations we call "waypoints". An agent gets from point A to point B by moving from A to a nearby waypoint. Then the agent moves from waypoint to waypoint until a waypoint close to B is reached. The waypoint set may be selected by hand, as is typical of games based on the Unreal engine, or the set may be selected by various algorithms (Stout 2000)(Snook 2000). It was early recognized that one key to keeping the computational requirements of searching the waypoint set manageable was to keep it as small as possible (Rabin 2000). Since waypoint infrastructure is so commonly available, it seems only natural to reuse it for determining places to hide (Reece 2003)(Reece 2000)(van der Sterren 2002).

The primary advantage of waypoint-based techniques is ease of implementation and low run-time computational complexity. Unfortunately, the latter benefit is only gained when the set of waypoints is small, and when it is small, the likelihood that the best place to quickly duck out of fire is a waypoint is also small. To see why this is so, consider a map consisting of an open plane with one tall rock sitting in the center. By appropriately placing the observer and the hiding agent, one can make virtually any point the nearest place to hide! An additional difficulty, and one that will become more important in the future, is that a sparse set of potential hiding places fixed in advance is especially vulnerable to becoming invalid in dynamic environments because of vehicle motion, destruction of buildings, and creation of new hiding places such as piles of rubble, to name a few examples. Thus waypoint-based techniques typically result in agents that can behave very counter-intuitively when searching for cover.

In military simulations, space is typically represented by a fine rectangular grid (Reece 2003) (Reece 2000) (Richbourg and Olson 1996). This avoids the difficulties caused by a sparse spatial representation as described above, but at the cost of computational complexity that may be beyond the budget of many games. The memory required to store the grid may also be an issue for very constrained computational platforms, like game consoles.

SENSOR GRID OVERVIEW

The sensor grid approach differs from its predecessors in that the set of possible hiding places is not fixed, but is instead generated dynamically at run-time. This allows it to be relatively dense close to the agent and sparse further out, while keeping the total size of the set small. Thus, this approach has the potential to provide some of the benefit of a large set of potential hiding places while avoiding the computational complexity. Additionally, this approach mirrors the fact that humans can generally perceive nearby opportunities to hide more easily than ones in the distance, and furthermore, the nearer ones are more likely to be useful.

The sensor grid approach takes its name from the fact that the set of potential hiding places that are tested by the algorithm is fixed relative to the agent. It is as if the agent had a collection of observer-detecting sensors fixed with regard to the agent and one another moving wherever the agent moves. A simplified overview of the algorithm is provided in Figure 1.

Figure 1

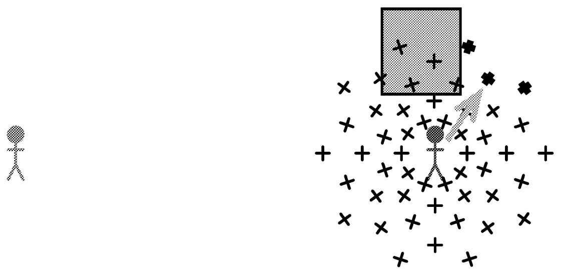


Figure 1: Top-down diagram illustrating the sensor grid approach. The agent (blue) is at right and a single observer (red) is at left. The array of sensors (plus signs) surrounds the agent. A single vision-obstructing object is present (the green square). If a sensor cannot see the enemy, its location is hidden (bold plus signs). The agent chooses the nearest hidden sensor that is accessible (e.g. not inside an object),and moves there (green arrow).

WAYPOINT SYSTEM OVERVIEW

Waypoint systems for pathfinding are quite common in First Person Shooter games. They provide fairly good results when using A*, the usual game industry pathfinding algorithm. However, waypoints can be used for much more than just pathfinding.

We have extended the standard Unreal waypoint system to include not only standard nodes used for pathfinding, but the following as well:

1. Cover nodes
2. Formation nodes
3. Peek-out nodes
4. Tree nodes

Figure 2 depicts a typical cover node setup.

Figure 2

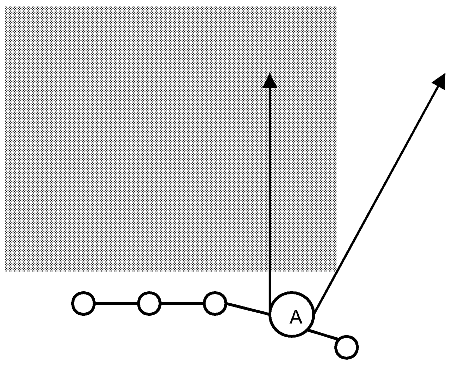


Figure 2: Cover node A is shown with 3 formation nodes to the left and a peek-out node to the right. The 2 handles used to create the angle of coverage are depicted as vectors connected tangentially to the cover node. The angle between these 2 vectors would be the computed angle of coverage for cover node A.

We have also added an initial step to the Unreal waypoint system that is run when paths are built in the Unreal editor. This step pre-computes the visibility amongst all nodes in the level with the exception of formation and peek-out nodes. This provides each node with an easily accessible list of nodes it can "see" at runtime. This optimization greatly reduces the number of rays that need to be cast for line of sight checks which can make standard node-based systems unusable on consoles such as the PlayStation 2.

Cover Nodes

Cover nodes are placed at points in the level which provide adequate cover against some set of weapons. They contain data which provides the agent with an idea of which direction the nodes provides cover towards, what stance to assume for maximum coverage, and what types of weapons it provides cover against.

When a designer places a cover point in the editor he is presented with a circular node that has two vectors we call "handles" connected (see diagram below.) These handles are used by the designer to create the angle of coverage for the given node. By rotating the handles to the desired positions the designer creates an arc. This arc, easily computed using 2D vector algebra, represents the angle that the given node provides cover against. When the designer runs the pre-computed visibility check each node casts a line of sight ray to each other node to see if it is visible. Each node then stores a list of nodes that it can see. Finally, nodes are marked as to whether or not they are in the given nodes angle of coverage. If they do not fall within the angle of coverage they are marked as being dangerous. While this is an N^2 operation it is only run on the entire set of nodes once. After the initial run, only nodes that have been modified will be rebuilt when pre-computed visibility is calculated. It should be noted that cover nodes assume a static environment. They are not designed to work with deformable terrain or other types of dynamic world geometry.

At runtime it is a trivial task to query a given nodes list of dangerous nodes. When being fired upon an agent queries the list of dangerous nodes for the node he currently occupies. If the enemy firing at him occupies a node that is not in the dangerous nodes list the agent simply assumes the correct posture for cover at his present node. If the enemy's node is in the dangerous nodes list the agent has to weigh the cost of either moving to a location that provides cover from the enemy, or returning fire and hoping for the best. If either agent does not occupy a node the node they are closest to can be used for determining cover. Alternatively, a line of sight check could be used. Similar techniques have already been implemented and tested (Liden 2002).

Formation Nodes

Formation nodes are used by fireteams to determine where each member of a fireteam should go when the fireteam leader occupies a cover node. Each cover node has a set of formation nodes associated with it by a level designer using the Unreal editor. A fireteam is a group of 4-5 soldiers who are all under the control of a designated fireteam leader. Only the leader uses the pathfinding system, all other fireteam members have a distance and orientation they maintain from the fireteam leader at all times. When a fireteam leader determines the team needs to take cover he moves to a cover node, and the other members move to the formation nodes that have been associated with the given cover node. Formation nodes are not included in the pre-computed visibility step. They share the same visibility as the cover node they are associated with.

Peek-out Nodes

Peek-out nodes are used by agents to move out from behind cover and return fire at the enemy. During level creation the level designer can associate up to two peek-out nodes with each cover node. Whenever an agent wants to return fire from a cover node he would follow these steps:

1. Lean out from the cover node, cast a ray, and see if the enemy is visible. If he is visible fire, if not proceed to step 2.
2. Check to see if at least one peek-out node is unoccupied. If there is an available peek-out node proceed to step 3.
3. Move a pre-set distance along the path between the cover node and the peek-out node.
4. Cast a ray towards the enemy. If the enemy is visible, fire. If the enemy is not visible return to step 3.

These steps would continue until either the agent had a clear line of sight to the enemy, or he had moved all the way to the peek-out node location and still could not see the enemy. As with formation nodes, peek-out nodes are not included in the pre-computed visibility step.

Tree Nodes

Tree nodes are very similar to cover nodes except they are used specifically around trees found in the level. They function exactly the same as a standard cover node except that they never have any formation or peek-out nodes associated with them. All trees in our current game have a small enough diameter that an agent never has to move out from behind them to return fire. He merely has to lean out in a given direction.

INTEGRATING THE SENSOR GRID AND WAYPOINT SYSTEMS

In a previous paper we discussed how the sensor grid approach was motivated as a replacement for navigating to safety on sparse waypoint graphs. We also discussed integration with waypoints as an extension to the sensor grid system. This combined approach is being implemented in a currently unannounced Unreal engine based first person shooter. While the sensor grid approach is highly effective at finding cover, it relies on numerous line of sight checks for each agent. Casting rays to check for line of sight is a very expensive computation, and with the limited resources inherent to consoles such as the Xbox and PlayStation 2 this approach is not feasible. By combining the sensor grid approach with a waypoint system we have created a system that is both fast and fairly inexpensive, but also yields very realistic results.

The major modification to the pure sensor grid approach is that now, when an agent needs to find cover, all cover nodes within some pre-set distance from the agent's location are checked first. If an adequate cover node is found the agent proceeds to that node. If no nodes are found, then the standard sensor grid algorithm is run. By only running the full sensor grid algorithm when no adequate cover nodes are found we greatly reduce the number of rays cast per frame. The beauty of the system is that it can be throttled by adding more nodes to the waypoint graph. If the system is being used on high end PCs with a lot of cheap memory simply reduce the granularity of the waypoint graph to allow the sensor grid algorithm to run more often. Or, if running the system on a console with limited resources, increase the granularity of the graph to reduce the usage of the sensor grid algorithm. An additional advantage is that even when the sensor grid is used to locate cover, the waypoints can be used to improve pathfinding to the covered point. This improvement is particularly significant in highly constrained environments (e.g. helping navigate through doorways).

EXTENSIONS

Disabling Waypoints in a Dynamic Environment

One disadvantage to using a standard waypoint system for cover is that it cannot handle dynamic environments. For example, if you have cover point A behind a wall, and that wall is destroyed by artillery, point A is no longer a valid cover point. Since waypoint graphs are computed at build time, and are static, there is no way to update the graph and notify agents that point A is no longer usable for cover.

The sensor grid deals with this problem by constantly scanning the environment for cover locations via line of sight checks. Thus, it can easily handle dynamic changes to the environment.

An interesting extension to the system we describe in this paper would be a method of disabling nodes when the environment changes. Going back to the example given above, after the wall is destroyed cover point A would be disabled and excluded from any further pathfinding searches. Now, when the agent is in the area near cover point A he uses the sensor grid algorithm to find adequate cover in the rubble of the former wall.

EXPERIMENTS AND RESULTS

The algorithm was implemented on top of America's Army version 2.0, which uses the Unreal Warfare engine. The core of the sensor-grid code was written in UnrealScript, and is approximately 500 lines in length. The extensions to the waypoint system were primarily made in C++ code, and added about another 1000 lines of code. All tests were carried out on a desktop PC with a Pentium 4 processor, 1 GB of RAM, and a GeForce 5600FX with 256 MB of RAM.

Running the algorithm provided nearly instantaneous results, and no slowdown in gameplay was noticed. Agents were able to successfully find cover behind various types of objects such as trees, rocks, buildings, and vehicles. In addition, agents now successfully traversed doorways which they were unable to do in the sensor-grid only approach.

The one area that can slow the running time of the algorithm down noticeably is the reliance on ray casting for the line-of-sight checks used by the sensor-grid. Ray casting is a very expensive operation in the Unreal engine. When multiple agents are all casting multiple rays each frame in an effort to find cover there is a noticeable drop in the framerate of the game. This is an even more serious problem on consoles such as the PlayStation 2 where the limited amount of memory and CPU resources make it virtually impossible to perform the sensor-grid part of the algorithm in its current form. We are currently working on a scheduling system for ray-casting which will time-slice the process and spread the casting of rays over multiple frames. We hope this scheduling system, used in conjunction with the pre-computed visibility system, will make this a viable cover finding algorithm for console based first-person shooter games.

CONCLUSIONS

We have presented a system that combines the ease of use and quick access to data of a waypoint system with the sensor grid approach and its robustness in dealing with dynamic environments.

The technique we describe is very fast when only node data needs to be queried to discover an adequate cover point. The efficiency at runtime is achieved by pre-computing node visibility at build time, as well as embedding key data in each node. When an adequate node is not discovered the extremely robust sensor grid algorithm is run which is very effective at finding cover quickly.

The sensor grid algorithm can sometimes make mistakes which are described in our previous paper (Darken 2004). Additionally, the waypoint system is subject to the constraint that in order for it to be completely effective, agents must reside on a waypoint anytime they check for cover. The pre-computed visibility relies on the fact that agents are always on a waypoint. They system will work if agents use their closest waypoint for cover calculations, however errors may creep in. To counteract this problem the sensor grid algorithm could be scheduled to run anytime an agent needs cover and he is not on a node.

Computing lines of sight is already a major component of the computational budget devoted to AI for many computer games. We feel this system greatly reduces the need for a large number of line of sight checks, but when necessary can use them to great effect.

ACKNOWLEDGEMENTS

Portions of this work were supported by funds from the Naval Postgraduate School, the Navy Modeling and Simulation Management Office, and the U.S. Army Training and Doctrine Analysis Center, Monterey.

BIOGRAPHY

GREGORY PAULL received his Masters Degree in Computer Science from Boston University in 2002. He is currently pursuing a PhD in AI through the MOVES Institute at the Naval Postgraduate School. He is also a full-time AI programmer at Secret Level Inc., a small computer game company located in San Francisco, CA. He has previously worked at Electronic Arts and Looking Glass Studios.

REFERENCES

Darken, C., Morgan, D., and Paull, G. "Efficient and Dynamic Response to Fire", *Proceedings of the AAAI Workshop on Challenges in Game AI*, 2004.

Darken, C. 2004. "Visibility and Concealment Algorithms for 3D Simulations", *Proceedings of Behavior Representation in Modeling and Simulation (BRIMS) 2004.*

Kelly, J., Beall, A., and Loomis, J. To appear. "Perception of Shared Visual Space: Establishing Common Ground in Real and Virtual Environments", to appear in *Presence.*

Liden, L. 2002 "Strategic and Tactical Reasoning with Waypoints", *AI Game Programming Wisdom,* Charles River Media, pp. 211-220.

Rabin, S. 2000. "A* Speed Optimizations", *Game Programming Gems*, Charles River Media, pp. 272—287.

Reece, D., Dumanoir, P. 2000. "Tactical Movement Planning for Individual Combatants", Proceedings of the 9th Conference on Computer Generated Forces and Behavioral Representation. Available at http://www.sisostds.org.

Reece, D. 2003. "Movement Behavior for Soldier Agents on a Virtual Battlefield." *Presence*, Vol. 12, No. 4, pp. 387—410, August 2003.

Richbourg, R., and Olson, W. 1996. "A Hybrid Expert System that Combines Technologies to Address the Problem of Military Terrain Analysis," *Expert Systems with Applications,* Vol. 11, No. 2, pp. 207—225.

Snook, G. 2000. "Simplified 3D Movement and Pathfinding Using Navigation Meshes", *Game Programming Gems,* Charles River Media, pp. 288—304.

van der Sterren, W. 2002. "Tactical Path-Finding with A*", *Game Programming Gems 3*, Charles River Media, pp. 294—306.

Towards a Fair 'n Square Aimbot – Using Mixtures of Experts to Learn Context Aware Weapon Handling

Christian Bauckhage
Centre for Vision Research
York University
4700 Keele Street, Toronto M3J 1P3, Canada
bauckhag@cs.yorku.ca

Christian Thurau
Applied Computer Science
Bielefeld University
P.O. Box 100131, 33501 Bielefeld, Germany
cthurau@techfak.uni-bielefeld.de

ABSTRACT

Superior weapon handling is the road to success in first person shooter games. On the expert level of gameplay, this demands more than just accurate tracking and targeting of opponents. It also requires knowledge about the behavior and characteristics of different weapons in the player's arsenal. Given enough practice, human players usually master both skills. Usual gamebots, in contrast, can only track and target but lack the sense for the appropriateness of a weapon in a given situation. As a consequence, their performance and behavior appears non-natural. In this paper, we propose to make use of machine learning techniques to realize aiming behavior that is adapted to the spatio-temporal context. We will present a mixture of experts architecture of neural networks which is specialized in the use of several weapons. Experimental result show that this paradigm indeed is able to produce more human-like aiming behavior.

Motivation and Background

The genre of first person shooter games is arguably among the most popular computer game genres[1]. In a FPS game, the player moves through a virtual world (also called a *map*) which he perceives from the first person perspective. Though several tactical variations exist, his main task basically is to battle against other characters on the map.

Virtual combat is carried out by means of weaponry that varies from game to game. However, something all FPS games have in common is that the provided weapons exhibit different characteristics. A typical virtual firearm might be devastating but take long to recharge, another might deliver a high frequent hail of bullets each of which will only have a rather low impact, or there might be an incarnation of a sniper rifle suited for long distance engagement but of minor use for close combat.

As these examples indicate, success in an FPS game may require more than virtual amok run. And indeed, observing professional or semi professional cyber athletes play reveals that they chose their arms according to the game state they encounter. For example, experienced players switch to a lower impact weapon after having hit the opponent with a high impact one. They know that if the opponent survived the first blow, firing another valuable high impact bullet would be a waste. A quick strike with a weapon of lesser strength will also yield the point.

In contrast to human players, artificial agents (also called *gamebots*) lack this level of sophistication. Since their behavior is still largely rule based, the set of variables that determine their weapon handling is comparatively small. While human players typically (and often unconsciously) chose their actions according to a whole zoo of parameters (e.g. the distance to an opponent, the local topography of the map, the current amount of ammunition, or known advantages and disadvantages of a weapon), gamebots often are programmed to have a weapon of choice and to use that weapon regardless of the current context of the game. In order to nevertheless provide a challenge on higher levels of a game, programmers compensate for the lack of bot intelligence by means of supernatural, unerring aiming: the bot reads the opponent's location from an internal variable and therefor simply cannot miss the target.

In this contribution, we will report on an avenue towards more situation awareness for gamebots. Considering the game QUAKE II® by ID software as a practical example, we will follow an idea proposed in an earlier contribution (Bauckhage, Thurau & Sagerer 2003). We will apply neural network architectures to analyze and learn from the data encoded in the network traffic produced by human players. In particular, we will discuss the idea of using a *mixture of experts* to obtain a collection of neural networks that are specialized in the use of several weapon. We shall present experiments which indicate that such an architecture can act as a context sensitive weapon selection mechanism. First, however, we will roughly survey related work in machine learning for commercial multiplayer computer games as well as briefly summerize the theory behind the mixture of experts concept. A summary and an outlook will close this contribution.

[1]According to the game portal Gamespy (08, 2004), 18 of the 20 most popular online games belong to the FPS category; in August 2004, there were more than 60.000 servers with more than 150.000 players online every minute.

Related Work

Currently, we are witnessing an increased academic interest in machine learning for the design of believable computer game characters. One of the boosting factors behind this boom can be concluded from observations reported by authors such as Cass (2002) or Nareyek (2004): on the one hand, up to today, most commercially available games rely on deliberative AI techniques like finite state machines or A^* searches. On the other hand, subsymbolic machine learning as a tool to produce life-like game agents has been largely neglected by the scientific community. However, this situation is about to change.

Recent work by Spronck, Sprinkhuizen-Kuyper & Postma (2003) introduced reinforcement learning to the task of rule selection for agent behavior in a commercially available role playing game. Earlier, the same authors reported on a hybrid coupling of genetic algorithms and neural networks for offline learning in a strategy game (Spronck, Sprinkhuizen-Kuyper & Postma 2002).

Neural networks were also reported to perform satisfiable in the genre of first person shooter games. Given the data contained in the network traffic of a multiplayer game, different architectures of neural networks were applied to learn life-like behavior for artifical characters. This was accomplished on the level of mere *reactive* behavior (Bauckhage et al. 2003, Thurau, Bauckhage & Sagerer 2003) as well as for *strategic* behaviors like goal oriented path computation (Thurau, Bauckhage & Sagerer 2004*a*). More recent work based on biologically inspired modeling and clustering techniques (movement primitives) even indicates that reactive and strategic components of behavior can be generated using a unified framework (Thurau, Bauckhage & Sagerer 2004*b*). All these approaches produce believable behavior for computer game characters. However, subsymbolic solutions for the *tactical* level of gameplay are still missing. According to a widely accepted psychological hierarchy of human behavior (Hollnagel 1994), tactical decisions require more situation awareness than reactive behavior but demand less long-term planning than a strategy. In the following, we will argue that multi-classifier systems like for example mixture of experts architectures may provide an appropriate answer to this problem.

Mixtures of Experts

Given a set of pairs of vectors $\{(\vec{x}^\alpha, \vec{y}^\alpha)\}$ where $\vec{y}^\alpha = f^*(\vec{x}^\alpha)$, a multi layer perceptron can learn a function f that approximates f^* by minimizing the sum of errors

$$E(\mathbf{W}) = \frac{1}{2}\sum_\alpha (\vec{y}^\alpha - f(\vec{x}^\alpha, \mathbf{W}))^2$$

where $\mathbf{W}$ is the weight matrix of the network. For a Gaussian error $(f(\vec{x}^\alpha, \mathbf{W}) - \vec{y}^\alpha)$ this approach will produce the maximum likelihood solution for the parameters w_{ij}. Alas, a single Gaussian mode is an unrealistic assumption for most real data. The mixture of experts (MOE) approach as introduced by Jacobs, Jordan, Nowlan & Hinton (1991) tackles this problem by means of a divide and conquer strategy. Instead of fitting a global function into the training data, it uses a mixture of local experts which are moderated by a gating network. In the following, we will briefly summerize this technique. Readers familiar with the topic should skip to the next section.

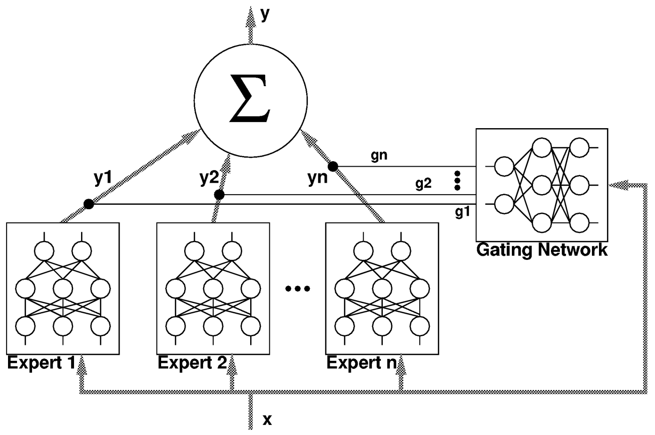


Figure 1: A mixture of experts architecture consists of a set of expert networks and a gating network. Given an input $\vec{x}$, the output $\vec{y}$ is computed as the sum $\vec{y} = \sum_j g_j(\vec{x}) y_j(\vec{x})$.

As shown in Fig. 1, the basic idea of the MOE approach is to compute the vector $\vec{y}$ as a weighted sum of outputs produced by n expert networks. Given a vector $\vec{x}$, the corresponding $\vec{y}$ thus results from

$$\vec{y} = F(\vec{x}, \theta) = \sum_{j=1}^{n} g_j(\vec{x}, \mathbf{V}) f_j(\vec{x}, \mathbf{W}_j)$$

Note that $\theta = (\mathbf{V}, \mathbf{W}_1, \ldots, \mathbf{W}_n)$ denotes the set of all model parameters. It is common to require $\sum_j g_j = 1$ which can be realized by defining the g_j to be soft-max functions of the output layer values s_j of the gating network:

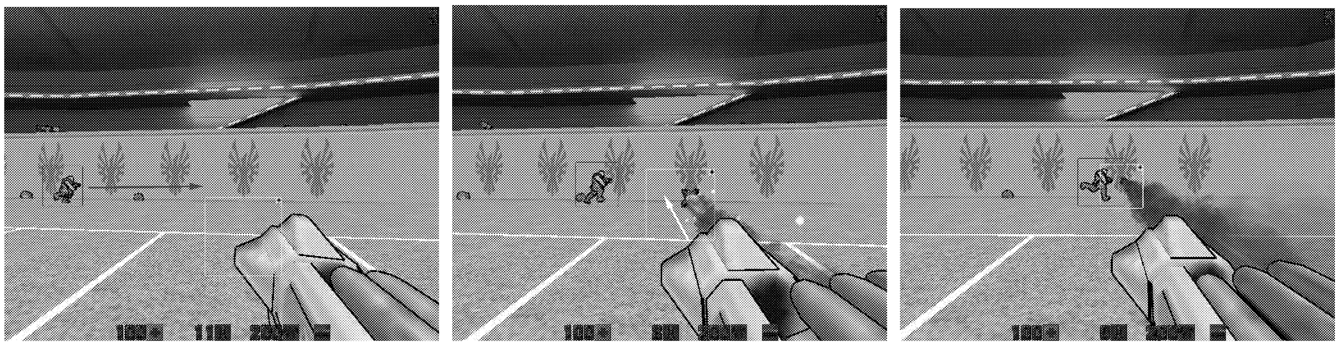
$$g_j(\vec{x}, \mathbf{V}) = \frac{e^{s_j(\vec{x}, \mathbf{V})}}{\sum_k e^{s_k(\vec{x}, \mathbf{V})}}$$

This quite naturally leads to a probability interpretation of the weights g_j.

The naive approach to the estimation of the parameters θ would be to minimize the error

$$E(\theta) = \frac{1}{2}\sum_\alpha (\vec{y}^\alpha - F(\vec{x}^\alpha, \theta))^2$$

by a gradient descend $\nabla_\theta E$. But the MOE model allows for a more elegant solution. Consider the following statistical interpretation: each pair $(\vec{x}, \vec{y})$ is generated by a random process that first chooses $\vec{x}$ according to a given density and then randomly selects an expert according to the probability $g_j(\vec{x})$. The chosen expert j generates a random variable whose mean



(a) Targeting its expected position . . . (b) . . . an sending the missile there . . . (c) . . . will most likely hit the opponent.

Figure 2: Example of experienced handling of the QUAKE II® rocket launcher. In contrast to the 'direct hit' weapons in the arsenal, rockets are considerably slow. This requires to anticipate the opponent's movement and to target its expected rather than its current position. A gamebot will have to reproduce this behavior in order to appear life-like.

is $\vec{y}_j = f_j(\vec{x}, \mathbf{W}_j)$. The vector $\vec{y}$ then is the expected value $\mathbb{E}(\vec{y}_j|\vec{x})$. The probability of a pair $(\vec{x},\vec{y})$ can hence be modeled as

$$\begin{aligned} P(\vec{x},\vec{y}|\theta) &= \sum_j g_j(\vec{x},\mathbf{V}) P(j,\vec{x},\vec{y}|\mathbf{W}) \\ &= \sum_j g_j(\vec{x},\mathbf{V}) \mathcal{N}_j e^{-\frac{1}{2\sigma_j^2}\left(\vec{y}-f_j(\vec{x},\mathbf{W}_j)\right)^2} \end{aligned}$$

where $\mathcal{N}_j$ is a normalization factor. Assuming independence of the pairs in the set $\{(\vec{x}^\alpha,\vec{y}^\alpha)\}$, the set's log-likelihood is thus

$$\begin{aligned} L(\theta) &= \log \prod_\alpha P(\vec{x}^\alpha,\vec{y}^\alpha|\theta) \\ &= \sum_\alpha \log \sum_j g_j(\vec{x}^\alpha,\mathbf{V}) \mathcal{N}_j e^{-\frac{1}{2\sigma_j^2}\left(\vec{y}^\alpha-f_j(\vec{x}^\alpha,\mathbf{W}_j)\right)^2} \end{aligned}$$

This likelihood approach is advantageous because now the familiar expectation maximization algorithm can be used to estimate the optimal set of parameters θ^*. But even a gradient descent $-\nabla_\theta L$ will converge faster and more reliable than in the case of $E(\theta)$ for the surface of $-L(\theta)$ is smoother and comes along with less local minima as Rao, Miller, Rose & Gersho (1997) point out.

Note that a MOE model simultaneously learns how to segment the input space and how to model the mappings from the resulting segments to the output space. In contrast to cluster algorithms like, for instance, k-means, the MOE approach yields a soft partition of the input space. Finally, extensions to hierarchical MOE models are possible where each expert itself represents a mixture of experts.

Experiments

In a first series of experiments, we examined whether a mixture of experts architecture can learn human-like handling of different fire arms in QUAKE II®. We confined this examination to a subset of three weapons and, for the time being, considered the *blaster*, the *rocket launcher* and the *railgun*. Roughly speaking, the railgun is a powerful long-distance weapon with considerable recharge delay. The rocket launcher is a devastating tool in mid-distance combat. Human players tend to avoid its use in close combat because the splash of rockets can cause self harm. The blaster is a handgun with a high bullet frequency suited for close combat where it is most likely to harm opponents.

According to their characteristics, these weapons are typically used in different (spatial) contexts. However, their use does not only differ with respect to the distance to an enemy player; their ballistics vary as well. While the railgun and the blaster are instant hit weapons, the missiles fired by the rocket launcher are rather slow. This requires to add anticipation to the handling of the rocket launcher. As it is exemplified in Fig. 2, instead of directly targeting their opponents, experienced players fire rockets towards a location where their adversaries will most likely appear in a couple of moments.

Given these observations, the question is thus if a multi-classifier architecture can learn to select a weapon according to the distance to an opponent and simultaneously learn to produce appropriate aiming.

Setup

In order to investigate this question, we recorded several demos. In each of these, two players met on a custom map called *stadium*; their use of the three different weapons corresponded to the above characteristics.

situation		# training vectors	# test vectors
blaster	aim	2079	693
	shoot	1572	525
rocket	aim	4104	1369
	shoot	1537	513
railgun	aim	6508	2170
	shoot	1870	624

Table 1: Data used in experiments.

The data that was extracted from this demos in order to train the classifier consisted of pairs $\{\vec{x}^{\alpha}, \vec{y}^{\alpha}\}$ at a frequency of 10Hz where the input vectors $\vec{x}^{\alpha} \in \mathbb{R}^5$ encoded the spatial angle and the distance between the player and its opponent as well as the yaw and pitch angle of the player. The binary output vectors $\vec{y} \in [0,1]^4$ consisted of flags indicating the type of weapon to be used and a flag indicating whether to shoot or not.

Three quarters of the recorded material were used for training, the remaining quarter was for testing. Table 1 summerizes how many vectors were given for each weapon and how many of them corresponded to actual shooting behavior. Note that these figures reflect the recharging times of the considered weaponry. While for the blaster the ratio of shots to observations is approximately $3/4$, for the railgun it is about a mere $2/7$.

We experimented with a MOE architecture of three experts and a gating network. The gate was chosen to have 10 hidden neurons, the experts were tested with 4 and 7 hidden neurons, respectively. Training was done in 5000 epochs using gradient descend or the EM algorithm and applied the Super-SAB method for learning rate control (cf. e.g. (Sarkar 1995)). For the sake of comparison, we also tested an architecture where the three expert networks were pre-trained, one for each weapon, and the gate was subsequently trained to select the most appropriate expert.

Results

All configurations tested in our experiments were able to reproduce the desired behavior. An in-game examination of the resulting network architectures showed bots that switched between the considered weapons according to the handling encoded in the training data. If the opponent was close, the bots selected the blaster, in mid-distance combat they referred to the rocket launcher and for long-distance fights they used the railgun. Also, the bots reproduced the aiming behavior contained in the demos. When using the blaster or the railgun they directly targeted their enemies; when using the rocket launcher they 'predicted' their adversaries trajectory and fired accordingly. Finally, the shooting frequencies characteristic for the different weapons were learned as well.

Table 2 displays some of the figures we obtained from an offline evaluation. Obviously, the architectures with expert networks with 7 hidden neurons perform better even though, compared to the ones with 4 hidden neurons, the gain is small. What is noticeable, is that the difference between training error and test error is small in every case; this backs the above observation of a good reproduction of the behavior encoded in the training material.

training method	pre-trained experts	# hidden neurons	E_{TRAIN}	E_{TEST}
$\nabla_{\theta} L$	no	4	0.01318	0.01280
EM	no	4	0.01332	0.01293
EM	no	7	0.01125	0.01133
EM	yes	7	0.01153	0.01167

Table 2: Experimental results.

In one of the cases considered in our experiments, training via gradient descend produced a slightly better result than the EM parameter optimization. Pre-trained experts performed generally slightly worse than those trained together with the gating network (s. last row of Tab. 2).

To our surprise, none of our experiments resulted in an architecture where there was a clear assignment of the three expert networks to the three different weapons. Rather, our experiment produced a holistic representation of the demo behavior distributed over the three networks.

Discussion

Mixtures of experts provide a means to learn the handling of different virtual weapons from human-generated training data. Simple combinations of small multilayer perceptrons (i.e. 5-4-4 or 5-7-4 network topologies) can learn to switch between weapons if the distance to an opponents changes correspondingly. Simultaneously, they can learn to produce appropriate aiming behavior, that is, to directly target the opponent with an instant hit weapon or to target the expected position of the enemy when using the rocket launcher.

These results are encouraging and hint that MOE approaches may also be able to treat more complex context dependent behaviors. However, these findings come along with a surprise. In none of our experiment did experts for the handling of a single weapon evolve. Instead, the gating networks resulting from the different configurations that were tested always generated soft combinations of the responses of the expert networks.

An explanation for this phenomenon could be that in the data space the handling of the different weapons does not differ as much as it appears from the in-game perspective. An effect like this was already encountered in a different context where we experimented with biologically inspired techniques to learn human-like movement skills (Thurau et al. 2004*b*). If this should be the case in our current experiments as well, the architectures might have learned the principal components or generalized principle components within the abstract data space and produce their output as a linear combination thereof. (Work by Fancourt & Principe (1997) revealed that MOE architectures can accomplish this.) Currently, we are conducting further experiments to verify this assumption or to gain more insight in what else is happening here.

Summary and Outlook

In this contribution we proposed to apply multi-classifier systems to learn different context dependent behaviors for computer game characters. The mixture of experts model provides an approach to the simultaneous training of a set of competing neural networks. Moderated by a gating network, subnetworks will emerge that are specialized in handling different regions of the data space.

For our experiments we used a MOE architecture of three multilayer perceptrons and a nonlinear gating network. Given training data that exemplified the use of three different weapons in the game QUAKE II®, the coupling of the networks indeed resulted in a system with expertise in using the three weapons. As a whole, the architecture learned in which context the considered weapons were most frequently used. It was learned that the *blaster* is usually invoked in situation of close combat where it is fired frequently. Concerning the use of the *rocket launcher*, the architecture produces a distance dependent offset in aiming on an opponent that compensates for the low speed of the missiles. For the *railgun*, in contrast, it learned to directly target opponents in farther distances. Even simple architectures of simple neural networks can thus reproduce human weapon handling. Therefor, the resulting gamebots do not have to rely on superhuman aiming in order to produce engaging gameplay; they are *fair* opponents.

Future work will have to consider larger expert architectures and more complex activities than weapon handling. This will have to include variables like *health* and *armor* which describe the internal state of a game agent and are responsible for more sophisticated tactics and behaviors. It will be interesting to see to what extend the technique applied in this contribution will scale. General experience in neural network research suggest that there might be a boundary. A remedy could be to extend the concept of movement primitive as described in (Thurau et al. 2004*b*) to the idea of *tactics primitives* which may be unified into an architecture similar to the mixture of experts approach.

Acknowledgments

This work was supported by the German Research Foundation (DFG) within the graduate program "Strategies & Optimization of Behavior". We would like to thank our students Marion Wadermann, Klaus Schapp, and Matthias Steinrücken as well as all other participants of the seminar *Neural Quakebots* held at Bielefeld University in the summer term of 2004. Without their interest, diligence, and enthusiasm this contribution would not have been possible. Also, we are deeply grateful for the help of Dirk Raufer. The in-game illustrations in this paper were generated using a visualization tool he developped in his BS. thesis completed at Bielefeld University in 2004.

REFERENCES

Bauckhage, C., Thurau, C. & Sagerer, G. (2003), Learning Human-like Opponent Behavior for Interactive Computer Games, *in* B. Michaelis & G. Krell, eds, 'Pattern Recognition', Vol. 2781 of *LNCS*, Springer, pp. 148–155.

Cass, S. (2002), 'Mind Games', *IEEE Spectrum* pp. 40–44.

Fancourt, C. & Principe, J. (1997), Soft Competitive Principal Component Analysis Using the Mixture of Experts, *in* 'DARPA Image Understanding Workshop', pp. 1071–1075.

Gamespy (08, 2004), http://archive.gamespy.com/stats/index.shtm.

Hollnagel, E. (1994), *Human Reliability Analysis: Context & Control*, Academic Press.

Jacobs, R., Jordan, M., Nowlan, S. & Hinton, G. (1991), 'Adaptive Mixture of Local Experts', *Neural Computation* **3**(1), 79–87.

Nareyek, A. (2004), 'Artificial Intelligence in Computer Games – State of the Art and Future Directions', *ACM Queue* **1**(10), 58–65.

Rao, A., Miller, D., Rose, K. & Gersho, A. (1997), 'Mixture of Experts Regression Modeling by Deterministic Annealing', *IEEE Trans. on Signal Processing* **45**(11), 2811–2820.

Sarkar, D. (1995), 'Methods to Speed Up Error Backpropagation Learning Algorithm', *ACM Computing Surveys* **27**(4), 519–542.

Spronck, P., Sprinkhuizen-Kuyper, I. & Postma, E. (2002), Improving Opponent Intelligence through Machine Learning, *in* 'Proc. GAME-ON', pp. 94–98.

Spronck, P., Sprinkhuizen-Kuyper, I. & Postma, E. (2003), Online Adaptation of Game Opponent AI in Simulation and in Practice, *in* 'Proc. GAME-ON', pp. 93–100.

Thurau, C., Bauckhage, C. & Sagerer, G. (2003), Combining Self Organizing Maps and Multilayer Perceptrons to Learn Bot-Behavior for a Commercial Computer Game, *in* 'Proc. GAME-ON', pp. 119–123.

Thurau, C., Bauckhage, C. & Sagerer, G. (2004*a*), Learning Human-Like Movement Behavior for Computer Games, *in* 'Proc. Int. Conf. on the Simulation of Adaptive Behavior'.

Thurau, C., Bauckhage, C. & Sagerer, G. (2004*b*), Synthesizing Movements for Computer Game Characters, *in* C. Rasmussen, H. Bülthoff, M. Giese & B. Schölkopf, eds, 'Pattern Recognition', Vol. 3175 of *LNCS*, Springer, pp. 179–186.

Agents Based Design for a Peer-to-Peer MMOG Architecture

Abdennour El Rhalibi ; Madjid Merabti
School of Computing and Mathematical sciences
Liverpool John Moores University
Liverpool, UK
a.elrhalibi@livjm.ac.uk ; m.merabti@livjm.ac.uk

KEYWORDS

Peer-to-Peer Architecture; MMOG; Online Gaming; JXTA; protocol.

ABSTRACT

Massively Multiplayer Online Games (MMOGs) are becoming a very important part of computer entertainment business. In this paper we discuss the issues surrounding MMOGs, the limitations in term of network infrastructure, and the lack of simulation environment to study and evaluate network architecture and protocol. We use a peer-to-peer (P2P) based architecture and protocol to provide a more scalable, flexible and robust technology solution than currently used infrastructures. We have conducted the design and implementation of a modular MMOG: 'Time-Prisoners', using a P2P protocol developed in Java and JXTA. The characteristics of P2P overlays enable to organize dynamically, and in transparent way for the users, the group of players according to their locations in the virtual world, and allow to design scalable mechanism to distribute the game state to the players and to maintain the world consistent in case of node failures.

INTRODUCTION

Massively Multiplayer Online Games (MMOGs) (Smed et al. 2002) are one of the most interesting genres of modern computer games. Born out of the internet boom in the mid to late 90's, they have rapidly gained in popularity. The most popular MMOG, Lineage (see, Lineage ref) claims to have over four million active subscribers. MMOGs, as the name suggests, are online games played simultaneously with tens of thousands of players at one time. The games require an internet connection to play. Each player has a copy of the software installed on their machine, which uses the internet connection to connect to a central game server, which in turns keeps all the players up to date with what is occurring in the world. Traditionally, most MMOGs have been Tolkien-esque fantasy role-playing games. While games of this type are still extremely popular, only recently has the full potential of this medium been realized, with games such as Planetside (see Planetside ref), Star Wars Galaxies (see Star Wars Galaxies ref), The Sims Online (see, Sims ref) and EVE (see, Eve ref), appealing to a broad range of player types.

MMOGs nearly always charge a subscription fee to each player in addition to the initial cost of purchasing the game client. This subscription payment is used to cover the costs generated by the game: Customer Service, Patches, Content Updates, Data Storage, Server Maintenance and Bandwidth (Bauer et al. 2002).

The last three make up the largest proportion of the cost. These costs are not directly spent on improving the gameplay for the players, but are a necessity to support the client/server model that forms the backbone of the game. By changing the network topology used to support MMOGs, these costs could be reduced. These savings could be then passed on to the player, greatly reducing the costs, or alternatively spent on developing the game further. The paper discusses the feasibility of using peer-to-peer (P2P) overlays (Knutsson et al. 2004) to support a typical MMOG, as a replacement for the client/server model. The technical details of these two topologies will be discussed, along with the issues. P2P infrastructure provides a better, cheaper, more flexible, robust and scalable technology solution for MMOGs.

The rest of the paper is organized as follows: section 2 provides background information about current MMOG network architectures and P2P overlays, section 3 briefly presents the application we have developed to deploy in a P2P overlays, section 4 introduces the P2P architecture we propose, introduce the software architecture meta-model and a protocol, and discusses the issues and possible solutions in the P2P architecture proposed, section 5 introduce some aspects of our communication system implementation, and finally in section 6 we review the concepts presented in this paper, conclude on the viability of the approach and present the future work.

CURRENT MMOG ARCHITECTURE AND TECHNOLOGIES

In this section we discuss some currently available topologies, which are or could be used for MMOGs. We also discuss some of the P2P overlays available.

Client/Server Topology for MMOG

The client/server network topology is very common in MMOGs (Smed et al., 2002) (Smed et al. 2003). Typically there is a group of "client" machines who want to share information, be they financial reports or game data. Each client connects directly to a server, which deals out information individually to each client as it is requested. This kind of topology commonly used in small-scale multiplayer games such as Half-Life (see Half-life ref). One player chooses to be the server and host the game, then all the other players connect directly to his machine. Whenever a client shoots a gun, moves or performs another action, the data is sent to the server, which calculates the results of that action and forwards the result to all client machines connected.

A single server is fine for small-scale multiplayer games, where the number of players is up to around 64, but a single machine is usually not sufficient to deal with thousands of players synchronously, so typically a MMOG "server" is a group of machines with dedicated responsibilities, as is seen in figure 1. This diagram is only one possible configuration of machines that could make up the server component of a MMOG. Each different machine has a different responsibility to the game. The whole "cluster" of machines operates using grid computing (Wang et al. 2004) methods to dynamically share resources and ensure consistency across the cluster. When a client first attempts to connect to the game, they connect to the login server, which checks the user exists, has a password, and has a paid up account. They may then be forwarded to the patch server, which checks the client version, and can send any game updates. When the patch server has confirmed the client version is up to date, they are connected to a game server and, perhaps, a separate chat server. In this example the chat server handles all player to player communication, regardless of where they are in the virtual gameworld and the game server handles everything else (e.g. physics, trading, combat). In this example, the "game server" is again split into several smaller sub-servers. These sub-servers could, for example, each handle a certain geographical area of the gameworld.

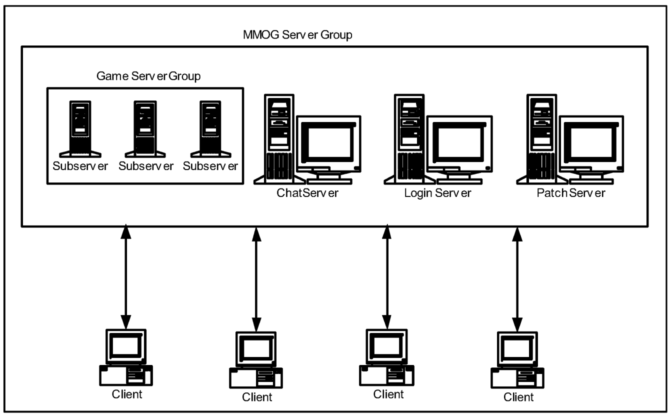


Figure 1. Possible MMOG Server Model

This Server model is one of many possibilities (see ref). Some games split the responsibilities of the servers in different ways, for example having dedicated servers for databases, physics or even Artificial Intelligence. One thing common to nearly all modern MMOGs is that the server group acts as the single server in the basic server/client topology, so they all suffer from the cost overhead of running and supporting so many machines. With many modern games there are even many clusters of servers, sometime referred to as "shards", that allow many distinct copies of the game to coexist. Usually these server clusters are located at different places in the real world, allowing players to use the clusters located near to them geographically, thus reducing the effect of network latency (Bernier et al. 2000).

In client/server infrastructure for MMOGs there are many known issues and solutions related to scalability (Francis et al. 2001), robustness (Knutsson et al. 2004), security and proof-cheating (Smed et al. 2002) (Golle et al. 2001), bandwidth savings (Knutsson et al. 2004), network and transport protocol (Hong et al. 2002), and delay compensation techniques (Bernier et al. 2000). However the solutions usually used are costly and lack of flexibility. For example in the case of the scalability the architecture usually uses server clusters, connected by LANs, or forming a computer grid. Although this architecture scale with the number of players, the server might need to be over-provisioned to handle peaks loads (Francis et al. 2001).

Peer-to-Peer Topology and Overlays

Peer-to-peer (Knutsson et al. 2004) (Kant et al. 2002), or "P2P", networking has become a bit of a technological buzzword in the past few years. It has been popularized by file sharing applications like Kazaa (see Kaza ref), Gnutella (see ref) and Morpheus (see ref), who use the technology to build ad-hoc networks to allow sharing of files.

Figure 2 shows a very basic illustration of how a peer-to-peer network is organized. In essence, each "Node" on the network has exactly the same responsibilities as every other node. It has no requirement for a particular machine to be in the network, and no other node has a requirement for it. This example is simplified, as modern P2P networks do not usually work with this basic a structure. While each node is still capable of the same functions as each other, modern P2P networks are able to organize themselves into more efficient structures.

Peer-to-peer networks have a lot of advantages over networks using the traditional server/client model used by MMOGs. Firstly there is the issue of cost. The network is distributed among the clients, and does not require a central server in order to work. The computation usually done by the server machine is shared instead by the clients. The peer-to-peer network of client machines can be treated as a giant grid computer, where calculations are performed in different parts of the conceptual "whole". This is similar to grid computing projects such as the SETI@Home (see ref) projects. If the entire server infrastructure (or even just a fraction) of a MMOG can be replaced by a grid-like system, a massive saving can be made. The amount of bandwidth used is reduced dramatically. Where before two clients wanting to communicate would be required to do so through the server, effectively doubling the amount of bandwidth required in a P2P network where they can communicate directly.

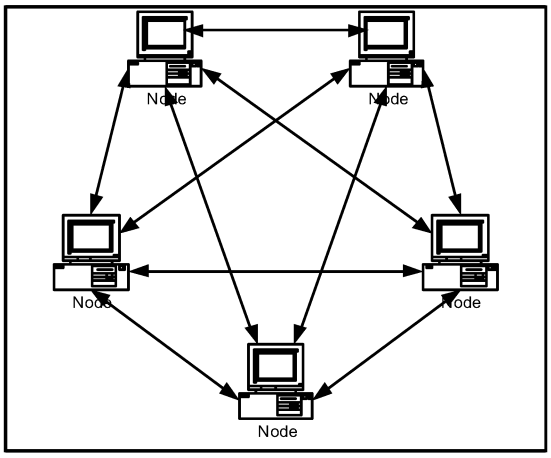


Figure 2. Basic P2P Topology

Latency on the network is also reduced thanks to the elimination of the bottleneck caused by a server handling all information between all clients, regardless of who each client is trying to communicate with.
In the next section we discuss available P2P overlays and their suitability for building multi-platform support for MMOGs.
A number of P2P protocols have been recently devised, including JXTA (see ref), Pastry (Kant et al. 2002), Tapestry (Kant et al. 2002), Chord (Kant et al. 2002), and Can (Kant et al. 2002). They are self-organizing, decentralized systems and provide the functionality of scalable distributed hash table (DHT) (Kant et al. 2002). The systems balance object hosting and query load, transparently reconfigure after node failures, and provide efficient routing queries (Hong et al. 202)(Morse, et al. 1996).

We are particularly interested in JXTA (Kant et al. 2002), which provides a far more abstract language for peer communication than previous P2P protocols, enabling a wider variety of services, devices, and network transports to be used in P2P networks.

JXTA is meant to provide a basic set of services and APIs required for the development of any peer-to-peer application. The architecture of JXTA divides the software into three layers: the JXTA core, JXTA services and JXTA applications (see JXTA ref). The core implements all the basic concepts involved in P2P communication. In particular, it provides the necessary functions for the management of peers and inter-peer message exchange. Furthermore, the core handles peer discovery and monitoring. The JXTA services layer is responsible for generic services that may be required in common P2P situations, such as file sharing and indexing. The JXTA applications layer is reserved for applications developed by the applications developers. It is in this layer that our application will reside.

Peers are organized into centers of interest called peer groups, which segregate the different communities participating in the JXTA network, and provide a way to control the propagation of broadcast traffic, for example.

Peers communicate by sending messages over JXTA pipes. These provide a transport-agnostic abstraction of the message exchange to client applications. Because pipes make use of the underlying transport protocol, whatever it may be, nothing can be assumed about the reliability of messages sent over JXTA pipes. It is, however, possible for developers to design their application so as to ensure reliability by implementing their own scheme, for example using TCP, UDP or even more specialized recent solutions such as GTP (Hong et al. 2002) and event synchronization protocol (Ferretti et al. 2004).

We can see that because of its rich set of P2P functionality, JXTA is eminently suited to our application. Augmented with a judicious reliability design, it provides an excellent starting point for a P2P based MMOG.

Before presenting the P2P architecture and protocol, we discuss the specification of the Game 'Time Prisoners' we have designed and implemented as a MMOG.

' TIME PRISONERS' MMOG SPECIFICATION

In this section we briefly present the specification of the game we have developed as a test-bed to deploy and test our P2P MMOG network over the internet.

The first step in the development of an MMOG is to design the gameplay itself, in both its technical and functional (Bosser, 2004). The game we have developed is 'Time Prisoners'. In figure 3, we can see screen shots depicting some of the regions and levels of the game.

The game-world is composed of 'parallel' worlds representing different donjons/regions and time (medieval, modern-war, etc…) in which the player must complete missions which consists in freeing prisoners, and fighting monsters/guards which wander in the world. Each region/time is composed of several levels, with puzzles to solve (maze to access the prisoners, finding trigger or keys to open levels, etc…) and items to collect (potion, ammunition, keys, etc…).

Players can navigate from region/time to region/time and see their character changes appearance and weapons to match the region/time style. There are two types of NPCs the prisoners and the monsters/guards. The AI controlling the NPCs (prisoners and monsters) is implemented using fuzzy finite-states machines (Buckland, 2002), path-finding (Buckland, 2002), influence mapping (Buckland, 2002), and support Dead Reckoning (Bernier, 2000).

The interactions (implemented as collision events) are numerous between players, players and NPCs, players and items, prisoners and guards, and all the characters with wall and objects. The world is relatively complex in terms of the events generated and the number of world updates which will be required.

The players can start to play in whatever region they want, and when they first join the game their machine will be either attached as client to an existing peer-group playing in the same region/level, or will be assigned the role of the main node (the server) of a new peer-group of future clients which will be playing in the same region/level.

Having briefly described our game application, we present in the next section the P2P architecture and protocol.

A PEER-TO-PEER ARCHITECTURE AND PROTOCOL FOR MMOGS

In section 2, we have mainly been concerned with examining the two extreme topologies available to MMOGs. However there are many hybrid techniques (Smed et al. 2002) that can overcome some of the issues outlined with the 'pure' topologies.

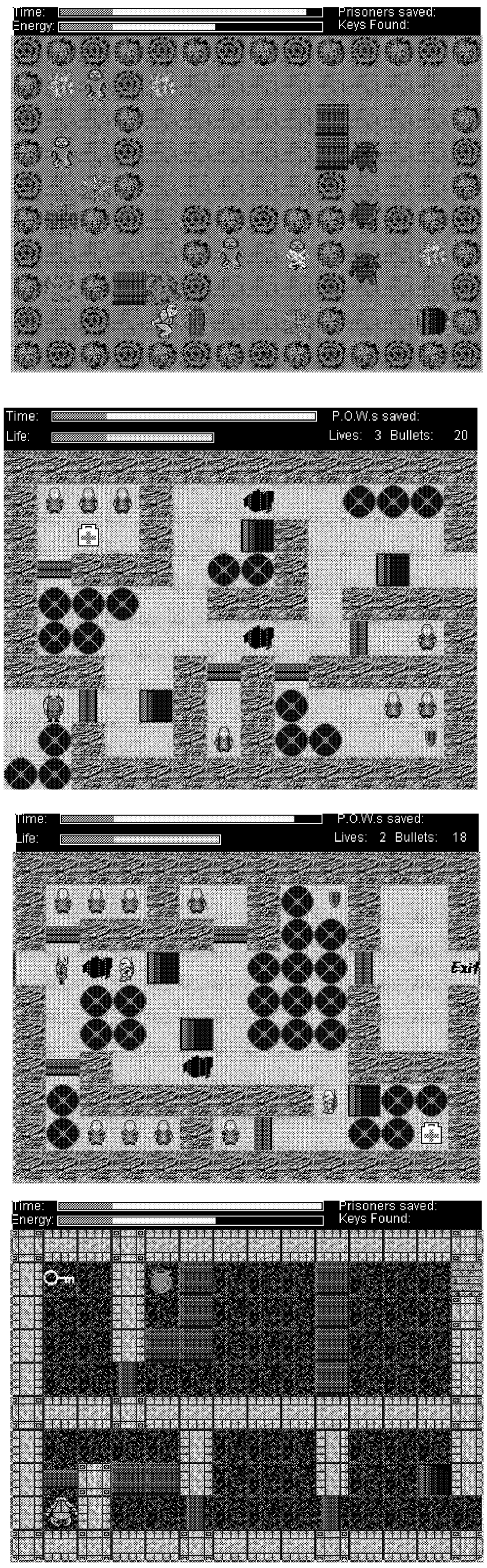


Figure 3. Screen shots of different world and levels in 'Time Prisoners'

A P2P Topology

The topology we propose is a hybrid solution starting with an initial architecture based on a main server, and building-up a P2P topology as the number of players increases (see Figure 4).

As the players connect to the game, the server delegates more and more of its role as game and network/communication manager to the connected player machines, which self-organize in a P2P fashion. The peers still rely on the initial server to join and leave the game, and to help them discover their peer-group if already created and to receive the game data when a new region is required. However all the in-game communications once the players is connected to his peer-group are done in P2P fashion.

The architecture allows the developers to maintain direct control and authority over the players account information, which are more secure when subscriptions and personal details are involved. An example is that of the 'Login' and account management server are kept away from the peer-to-peer network. They are managed into a separate client/server network style that acts as a gateway into the game. This separate network doesn't involve the client/server issues mentioned as it is only a one-off connection for the players. Once a joining player has been checked, he will granted connection to his peer-group and will be managed in a P2P mode until he leaves the game.

The architecture is flexible, robust and dynamic. Several spatial data structures are used to control peers group and their relations in a dynamic way, and to map the peer-groups.

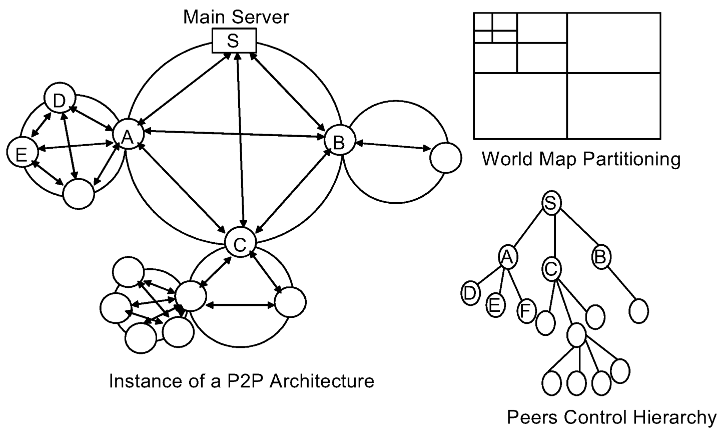


Figure 4. Proposed Architecture

In the following sections we discuss the meta-model architecture and the P2P protocol for joining and leaving the game. This protocol is the more important as regards to the P2P architecture.

Meta-Model Architecture

The first step in our P2P architecture and protocol development is the design of an agent based meta-model architecture. We have defined a high level design based on mobile agent model and suitable for the development of the MMOG and a simulation environment. Our meta-model architecture provides the rules to develop a simulation environment for MMOGs, and to implement an instance of P2P protocol for MMOGs. Using a mobile agent model we can represent all the dynamic and static aspects of our MMOG.
Our agent behavior model consists of three levels of design: the system-level, the host-level, and the agent-level. The system-level design describes an overview of the system and the relationships among hosts in the system. Host-level design uses State-chart to describe the behavior between agents within a host and between hosts, for example communication. Agent-level design uses finite state machines to describe the behavior of a single agent.
We have also incorporated the concepts of agent cloning, host replication, and agent groups within our framework.

To support multi-agent organization, communication and coordination as a P2P infrastructure, we also incorporate the concept of agent groups. Any agent within the agent group may perform multicast or subcast. A multicast communication allows an agent in the group to send a message to all other agents in the group, no matter where the agents are in the system. A subcast allows an agent to send a message to a subset of the group. Agent groups are identified by name, thus an agent may join a group by merely specifying its name.

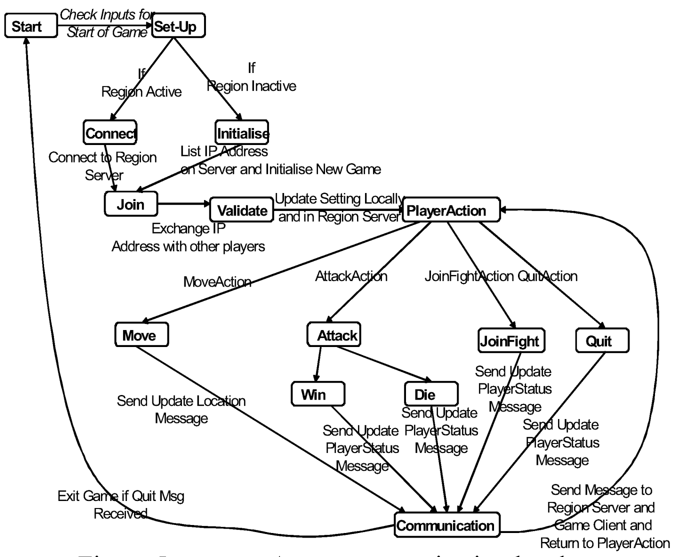

Start
Check Inputs for Start of Game
Set-Up
If Region Active
If Region Inactive
Connect
Initialise
Connect to Region Server
List IP Address on Server and Initialise New Game
Join
Validate
Update Setting Locally and in Region Server
PlayerAction
Exchange IP Address with other players
MoveAction
AttackAction
JoinFightAction
QuitAction
Move
Attack
JoinFight
Quit
Win
Die
Send Update Location Message
Send Update PlayerStatus Message
Send Update PlayerStatus Message
Send Update PlayerStatus Message
Send Update PlayerStatus Message
Exit Game if Quit Msg Received
Communication
Send Message to Region Server and Game Client and Return to PlayerAction

Figure 5. Agent communication level

The overall system behavior can be seen as an emergent property of the concurrent execution of all agents in the system. Our framework enables the understanding of P2P based MMOG design using mobile agents by specifying and simulating behavior on various levels.

HIGH LEVEL DESIGN OF ARCHITECTURE

In this section we will concentrate on the user management aspects of the MMOG system. The requirements for the MMOG system are as follows: There exists only one central database server where all usernames and passwords are stored. The Central Database Server is connected to Region Servers (which are the first players/clients logging to region in the virtual world where there are no other players). Each Region Server handles a different region in the virtual world. Once the Region Server is initiated, to join the virtual world, each user uses a client to connect to the Region Server which represents the particular region. Region Servers notify the Central Database regarding to the user's login information. If a new user logs in, the Central Database will register a new account for the user, otherwise, the user will be checked for authorization. After login, users within close proximity in the shared virtual space need to receive state updates about each other in real-time. The state information of each user should be stored on the Region Server, where the user first created his account.

System-Level Design

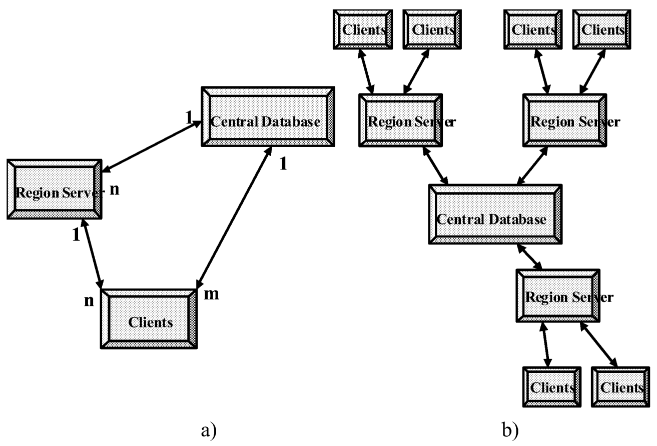
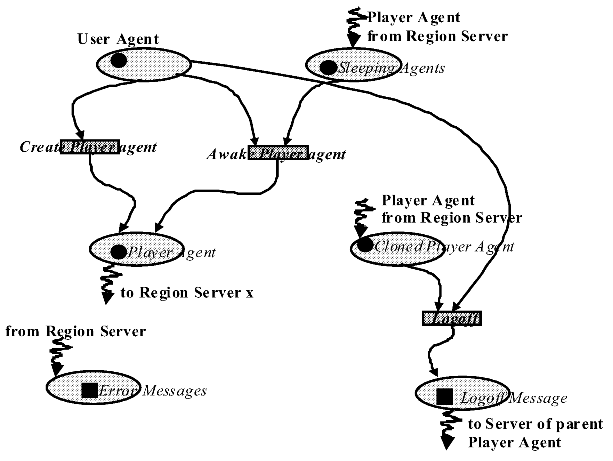


Figure 6. System High level Design of MMOG

To start we need to consider the system as a whole. Thus, we will start by working on the system-level design. In Fig 6.a, we have one central database (unique) along with regional servers (replicable) and clients (replicable). Both regional servers and clients are replicable hosts. A replicable host can have multiple instances of itself. The connections between the hosts are bi-directional system links, which means that the hosts can communicate in both directions. The number and two variables (n, m) describe the relationship between the hosts. In the example, the relationship between the central database to regional server is 1 to n. This signifies that one Central Database Server is connected to multiple Region Servers. Note also that the Central Database Server is unique in the system. Similarly with the regional server and client, one regional server is connected to multiple clients. Fig 7.b is an expanded view of the system-level design which may be more helpful in visualizing the physical configuration of the system.

Host-Level Design



a) Client

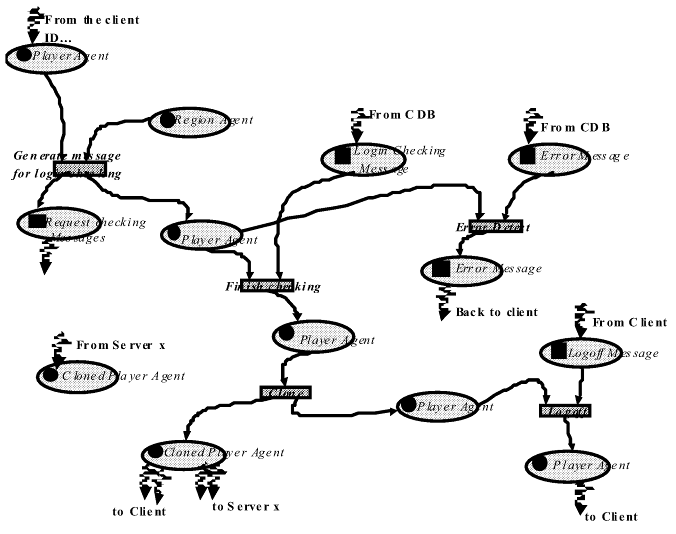
With the system-level view complete, we can now start designing the hosts that were introduced in the system-level design. Host-level design is based on process/event modeling. The order of design of the hosts will not affect the outcome. We will start with the host-level design of the client host.

Client

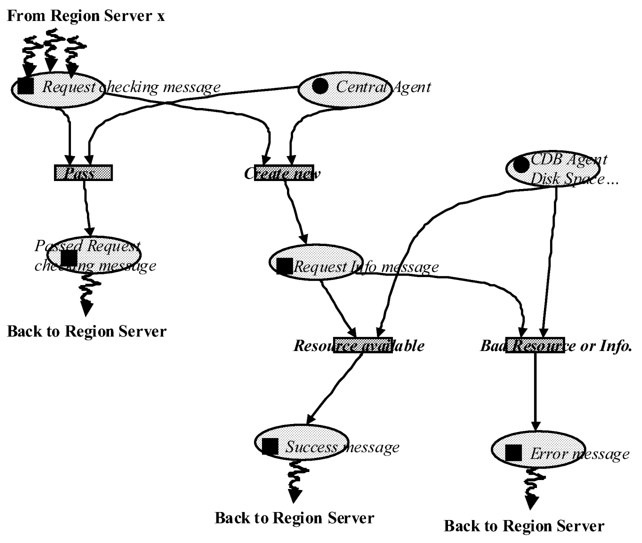
The first step is to determine the agents that may reside on or traverse through this host. We introduce a User Agent to represent each user's account information. In MMOG a user is typically able to create multiple characters (players). Thus, the User Agent should be able to create new Player Agents or wake up existing Player Agents already created by the user. The User Agent should also send out a message to the Region Server upon logoff. We can immediately identify that the User Agent will be static, because it will only process jobs that is related to users on that Region Server.

The next step is determining the transitions and places on the Client Host. There are three events: creating the Player Agent, waking up the Player Agent, and logoff. Fig 7.a shows the relationship among the User agent, the active Player Agents and the sleeping Player Agents. The diagrams show two types of arcs, the arcs going from the place to the transition state and then going from the transition state to a place, these are the process arcs; and the arcs incoming or outgoing from a place, which are the migration arcs. Each migration arc will specify where the agent is going to or coming from. If we follow the path for creating or waking up the Player Agents, there is a migration path to a Region Server with id x. The waked up Player Agent migrates to the appropriate Region Server according to the host ID.

Let us migrate with the Player Agent to the Region Server. A Region Agent is already waiting for the incoming Player Agent. Once a Player Agent has arrived, the Region Agent generates a check up message and the Player Agent waits for the login confirmation from the Central Database Server.



b) Region Server



c) Central Database Server

Figure 7. Host-Level Design of MMOG

Central Database Server

Let us move our point of view to the Central Database Server. In Fig 7.c, as we can see, there are lot migrations coming in from the regional server. The central agent is already waiting on the host to check for the login identification or create a new account. If the identification is correct, the request-checking message will be sent back to the Region Server identifying that the information is correct. Otherwise, if the user tries to create a new account, the Central Database Disk Space Agent will allocate space to record the information for the new user. If the requesting-information is valid and there is enough disk space, the account will be created, otherwise an error message will be created to notify the user that the information is not valid or there is not enough disk space. The messages will be sent back to the Region Server, as shown in Fig 7.b, near the right hand top where the return messages are received. If the incoming message is an error message the error message is forwarded back to client and eventually the user will have to re-enter the information. If the incoming message is a successful message, the waiting Player Agent can now get ready to clone. Each player agent then migrates to different

Region Servers and also to the client where the player agent came from.

The original agent stays at the Region Server for increased security and improved recovery from client crashes.

At this point the Player Agent stays at the regional server until the client decides to logoff. Let us look back to Fig 7.a, we can see the User Agent decides when to logoff.

Once the user decides to quit, the User Agent will generate a logoff message and then the message will be sent to the home Region Server. Follow with the message to Fig 7.b, where the logoff message comes in. The Player Agent now migrates back to the client. In Fig 7.a, the Player Agent comes back from the Region Server and is put to sleep. The player agent may be awakened later when the user decides to login again and reuse the player.

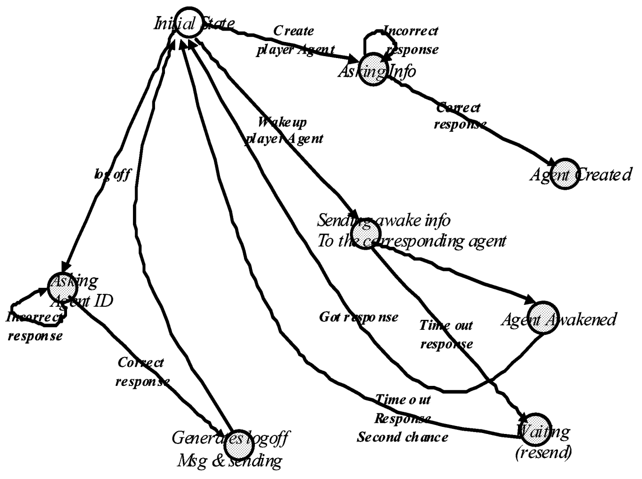


Figure 8. User agent

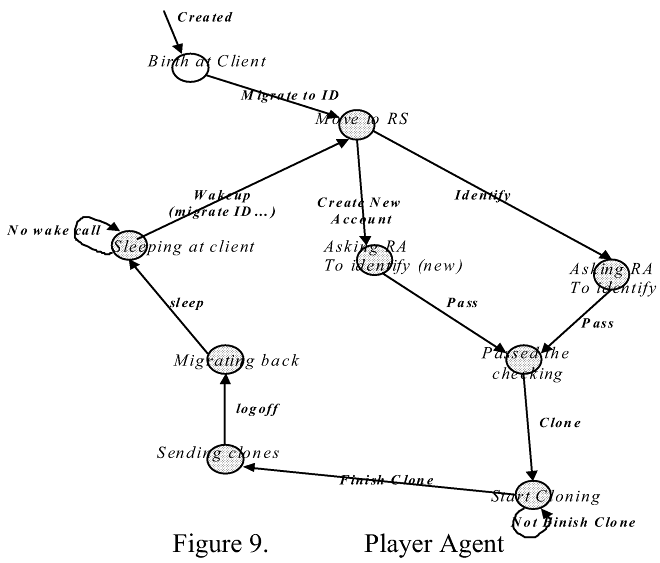


Figure 9. Player Agent

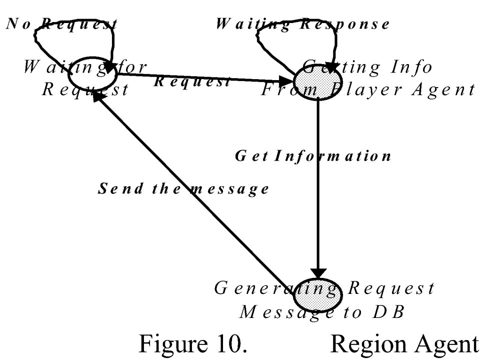


Figure 10. Region Agent

Agent-level design allows us to understand the behavior of individual agents. Let us look into Figures 8 to 12. Fig 8 describes the states of the User Agent. To start, we look at the initial state. If the user decides to create a new Player Agent, the User Agent will then go to the "asking info" state where the user agent will prompt the user for create information. When get the correct response, a Player Agent will be created and the User Agent will go back to the initial state. If the User Agent decides to use the created Player Agent, it will go to the "sending awake info to the corresponding agent" state and follow the path. This comprises the familiar finite state machine (FSM) of the User Agent.

The individual behaviors of the other agents are shown in Figures 9 to 12. Detailed descriptions are not given here; the diagrams are self-explanatory.

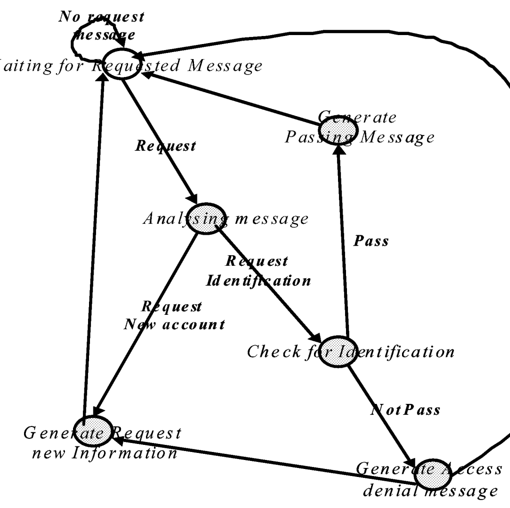


Figure 11. Central Agent

With the combination of these finite state machines at the agent-level (individual behavior), the Process/event model at the host-level (group behavior), and the overall view at the system-level, the complete system design can be seen as a large Process/Event model describing interacting autonomous agents residing in places among a set of hosts.

The meta-model and the high level design of the agents' architecture provide a very suitable framework for MMOGs as dynamic distributed applications, and will be completely described in a future publication.

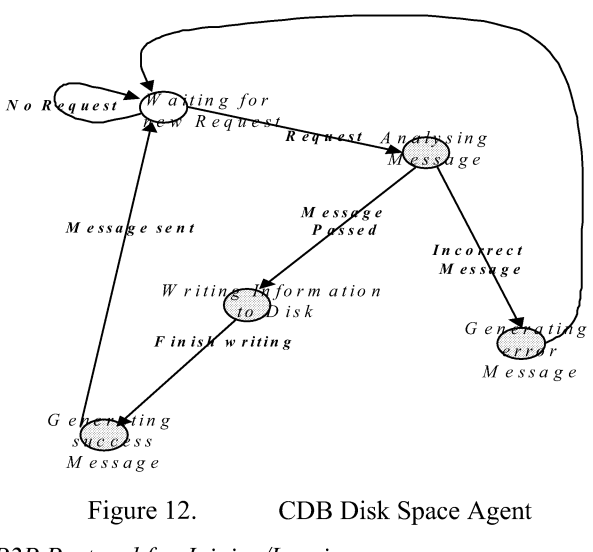


Figure 12. CDB Disk Space Agent

P2P Protocol for Joining/Leaving

In term of architecture build-up the protocol for joining and leaving are the most important. Below we present briefly the main scenarios in the P2P protocol for joining and leaving the game:

- In the initial state there are no players, no peers, only the main game server (world server) maintaining the full game, the players database, the current game data for existing players, etc…

- If a new player joins the game, he will be connected to the world server and assigned the role of new region server for the region he will play on. He will be sent all the data required to maintain the region, and the identity of any other running region server. Any new player joining in the same region will be connected in P2P fashion to the peer-group managed by the region server. The region server will be sent the data of the game, the data of the players joining, and the data of the others existing regions server controlling the others regions.
- If a player moves to another region we have two cases which might occur. Firstly if the region is already controlled by another server region, the new player will join the peer-group associated to this server-region. Secondly if no peer-group for the region exists, the client machine will be assigned the role of server region.
- If a player leaves the game. We have again two cases. If the player is a client in his region peer-group, it will be simply disconnected and the peer-group will be informed. If the leaving player is a region server, the protocol will elect a current peer (client) to become the new region server. The connection with the world server will be re-established. If there are others regions server they will be informed of the disconnection of the leaving region server, and will be given the identity of the new region server. Only the region servers are connected to the world region (i.e. the main server).

Figure 5 shows an example for the agent communication level, part of the communication protocol associated to a peer. Note in particular the states Join and Quit which implement the protocol described in this section.

DISCUSSION

In this section we discuss the issues which might occurs in a P2P network and the solutions we have devised. The issues are related to scalability, network efficiency, data storage, and policing.

Scalability

One of the usual issues with peer-to-peer networks is their scalability. We have designed our topology to have the ability to organize itself efficiently, to reduce the inherent scalability issues. The most basic of this organization is the creation of "Supernodes". The network can identify which nodes could make good supernodes based on how much bandwidth a node has, and how often and for how long it is usually a part of the network. Supernodes connect to one another, and have a collection of normal nodes connected to them. In larger networks, a collection of supernodes could even set up a super-supernode, connected to other super-supernodes and so on. These scaling techniques have been applied to our MMOG to improve efficiency. For example as in figure 6, each supernode includes a subnet of peers based on their locations in the game world.
The diagram does not clearly point out that the "group" and location supernodes would actually be controlled by the nodes within those groups, actually maintained by the player node software itself, but it is how it works.

Network Efficiency

As with the server/client architecture, several methods can be used to make a MMOG more efficiently use its network resources. Most of these techniques can be equally applied to a MMOG running on a peer-to-peer topology. Dead Reckoning (Bernier, 2000) algorithms greatly reduce the amount of object position data required to be sent over the network. Multicasting packets (Francis, 2001) are also used instead of traditional unicasting, thereby reducing the load of network traffic.

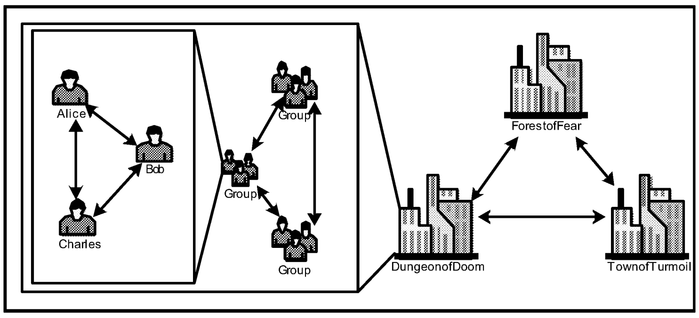


Figure 13. Supernode Hierarchy in P2P MMOG

Data Storage

MMOGs with client/server architecture have a single huge "game state" (Bosser, 2004), which is the data of the game world held usually exclusively by the server. This data is extremely important to the game and access and changes to it are strictly monitored by the server software.

In a game running on a P2P network, the storage of the game data must be distributed among the nodes, and it must store a large amount of redundant (replicated) data, so that the game can function with a minimum number of players.

A way of storing this much data among a collection of nodes hierarchically organized, where information could be required by any node, requires careful designing (Morse et al., 1996).

We believe there is a solution that can be adapted to be used to store data this way based on the Freenet project (see Frenet - ref). In our system each node "donates" an amount of hard disk space to the software when it is installed. When a node wishes to add data to the network, the network finds nodes with available disk space to store the data. When someone wants to access that data, they request it to be sent. Whenever data is requested by a node, that data, or a portion of that data, is replicated to data stores in nodes closer to the node that made the request, thus increasing the availability, redundancy and bandwidth available for that data and anyone who wants it.

Policing

In a peer-to-peer game of any sort, it is important for each node not to trust any other node. Yet in order for the game to function, calculations must be carried out at some point.

Most game events can be reduced to simple transactions between two players, or between a player and the game. In a peer-to-peer MMOG, this transaction requires that neither player has any control whatsoever. This means that no player data (except a reference to that data) may be held on the node owned by that player, and no transactions pertaining to that player may be calculated on that node. Instead, a group of nodes must all perform that transaction, by finding and checking the data on each player, then each node in the group double checks their result with the rest of the group before storing the data. Given that none of the nodes in the transaction can control which nodes are involved in it, this emulates the existence of a "trusted" server as in current MMOGs.

COMMUNICATION SYSTEM IMPLEMENTATION

We have developed our MMOG in Java and use a peer-to-peer architecture, incorporating JXTA as part of the underlying framework. Figure 14 shows part of the game logic and communication system class hierarchies. Figure 15 shows the layered structure of the full application, involving JXTA services and core, JXTA communication interface, the communication system and the game engine.
We are particularly interested in explaining the communication system. The aims of the communication

system are to provide with a way to communicate with other peers over the network. This allows for the passing of information between game peers, such as requesting information about games and updating player information.

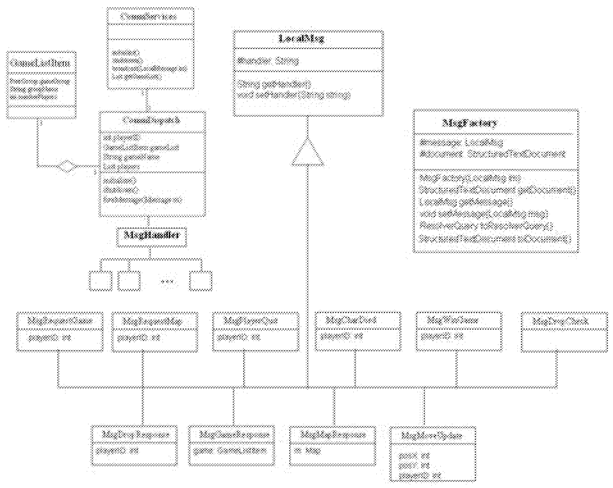
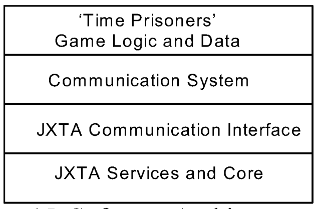


Figure 14. Class Architecture



'Time Prisoners' Game Logic and Data
Communication System
JXTA Communication Interface
JXTA Services and Core

Figure 15. Software Architecture

The communication subsystem provides the following services:

- start a multi-player session game, given the map/region/donjon which will be used
- get a game list
- join a game, given the player to join and an item from the game list
- broadcast a character position update message, given the new position
- broadcast a quit game message
- broadcast a win game message
- broadcast a player died message
- shutdown

The messages that are going to be passed between communication systems are JXTA messages which are XML documents, or binary or both (binary embedded into XML). They will be constructed by using the player state context and, then sent out by the communication system.

CONCLUSION AND FUTURE WORK

Massively Multiplayer Online Games have been rapidly gaining popularity and have proven that there is a big future in online gaming. The subscription price model used by most MMOGs is greatly slowing the uptake in new subscribers, and ways need to be developed in order to reduce the overhead of the server upkeep costs and to pass on a saving to the customer, either in monetary terms or in gameplay enhancement.

MMOGs by their distributed and incremental structure are natural applications for P2P overlays. Games are different from previous P2P applications that focus on the efficient use of idle storage and network bandwidth, including storage systems, content distribution and instant messaging. Games can use the memory and CPU cycles of peers to maintain and share the game state.

Our protocol offers a fully distributed, fault tolerant and scalable solution to MMOG. It is more efficient than usual infrastructure which relies on a set of dedicated servers that agree to share the workload. This protocol may also be extended for mobile networks (Joseph et al. 97) where a peer-group manager is a routing agent responsible for clients in its group. The joining and leaving protocol would remain mostly the same for such an application.

Future work will involve the development of an agent-based MMOG simulator to test 100 thousands of simulated players using different protocols, larger deployment of 'Time Prisoners' to test the P2P infrastructure based on JXTA, and an extensive evaluation of the simulator and the real system for scalability and robustness.

REFERENCES

Bauer D., Rooney S. and Scatton P. 2002. "Network Infrastructure for massively Distributed Games". In Proceedings of *NetGames 2002*, (Braunschweig, Germany, April), pp3-9.

Y.W. Bernier. "Latency compensation techniques methods in client/server in-game protocol design and optimization". *In Proceeding of the Game Developers Conferences*, March 2000.

Anne-Gwenn Bosser, "Massively Multi-player Games: Matching Game Design with Technical Design", *ACM SIGCH Advanced Computer Entertainment Conference, ACE 2004*. NUS, Singapore.

Mat Buckland, "AI Techniques for Game Programming" *Premier Press*, Inc. - ISBN (1-931841-08-X) (October, 2002)

EVE: The Second Genesis, © CCP (*http://www.ccpgames.com/*)

EverQuest, © Sony Online Entertainment (*http://soe.sony.com/*)

Stefano Ferretti, and Marco Roccetti, "A Novel Obsolescence-Based Approach to Event Delivery Synchronisation in Multplayer Games". *International Journal of Intelligent Games and Simulation*, Vol 3 No 1, 2004, pp7-19.

P. Francis, M. Handley, R. Karp, S. Ratnasamy, and S. Shenker. "A Scalable Content-Addressable Network." In *SIGCOMM '01*, August 27-31, 2001, San Diego, California.

Freenet Project – *http://www.freenetproject.org*

Gnutella Protocol Development - *http://rfc-gnutella.sourceforge.net*

Golle, P., and Mironov, I. "Uncheatable Distributed Computations". *Lecture Notes in Computer Science 2020* (2001).

HalfLife, © Sierra (*http://games.sierra.com/games/half-life/*)

Eunsil Hong, Sangheon Pack, Yanghee Choi, Ilkyu Park, Jong-Sung Kim, and Dongil Ko, "Game Transport Protocol: Transport Protocol for Efficient Transmission of Game Event Data," in Proc. *JCCI 2002*, Jeju, Korea, April 2002.

A. Joseph, J. Kubiatowicz, and B. Zhao. "Supporting Rapid Mobility via Locality in an Overlay Network". University of California, Berkeley.

JXTA - *www.jxta.org/project/www/background.html*

K. Kant, R. Iyer and V. Tewari, "A framework for classifying peer-to-peer Technologies", *2nd IEEE/ACM Intl. Symposium on Cluster Computing and the Grid*, May 21-26, 2002, Berlin, Germany.

KazaA - *http://www.kazaa.com/us/index.htm*

Bjorn Knutsson, Honghui Lu, Wei Xu and Bryan Hopkins "Peer-to-Peer Support for Massively Multiplayer Games", *INFOCOM 2004*, March 2004, Hong Kong, China.

Lineage, © NCsoft (http://www.lineage.com/nci)

Maniatis, P., Roussopoulos, M., Giuli, T., Rosenthal, D. S. H., Baker, M., and Muliadi, Y. "Preserving Peer Replicas By Rate-Limited Sampled Voting". *In SOSP* (2003)

K.L. Morse. "Interest Management in Large Scale Distributed Simulations". *Technical Report ICS-TR-96-27*, Universiy of California, Irvine, 1996.

Morpheus - *www.morpheus.com/*

PlanetSide, © Sony Online Entertainment (*http://planetside.station.sony.com/*)

Seti@Home - *http://setiathome.ssl.berkeley.edu/*

The Sims Online, © Electronic Arts (*http://www.eagames.com*)

Smed, Kaukoranta, and Hakonen, "Aspects of Networking in Multiplayer Computer Games", *The Electronic Library*, 20(2):87-97, 2002.

Smed, Kaukoranta, and Hakonen, "Networking and Multiplayer Computer Games--The Story So Far", *International Journal of Intelligent Games & Simulation*, 2(2):101-110, 2003.

Star Wars Galaxies, © Sony Online Entertainment (*http://starwarsgalaxies.station.sony.com/*)

Tianqi Wang, Cho-Li Wang, Francis Lau, "Grid-enabled Multi-server Network Game Architecture," *the 3rd International Conference on Application and Development of Computer Games* (ADCOG 2004), April 26-27 2004, City University of Hong Kong, HKSAR.

DIFFICULTY SCALING OF GAME AI

Pieter Spronck, Ida Sprinkhuizen-Kuyper and Eric Postma
Universiteit Maastricht / IKAT
P.O. Box 616, NL-6200 MD Maastricht, The Netherlands
E-mail: p.spronck@cs.unimaas.nl

KEYWORDS
Gaming, game AI, difficulty scaling, artificial intelligence, machine learning, unsupervised online learning.

ABSTRACT

"Difficulty scaling" is the automatic adaptation of a game, to adapt the challenge a game poses to a human player. In general, a game of which the challenge level matches the skill of the human player (i.e., an "even game") is experienced as more entertaining than a game that is either too easy or too hard. In practice, when difficulty scaling is implemented in a game, it only adapts a few parameters. Even state-of-the-art games do not apply it to game AI, i.e., to the behaviour of computer-controlled opponents in a game. In prior work, we designed a novel online-learning technique called "dynamic scripting", that is able to automatically optimise game AI during game-play. In the present paper, we research to what extent dynamic scripting can be used to adapt game AI in order to elicit an even game. We investigate three difficulty-scaling enhancements to the dynamic scripting technique, namely (1) high-fitness penalising, (2) weight clipping, and (3) top culling. Experimental results indicate that top culling is particularly successful in creating an even game. We conclude that dynamic scripting, using top culling, can enhance the entertainment value of games by scaling the difficulty level of the game AI to the playing skill of the human player.

INTRODUCTION

The quality of commercial computer games (henceforth called "games") is directly related to their entertainment value (Tozour 2002a). The general dissatisfaction of game players with the current quality of artificial intelligence that controls opponents (so-called "game AI") makes them prefer human-controlled opponents (Schaeffer 2001). Improving the quality of game AI is desired in case human-controlled opponents are not available.

Many researchers and game developers consider game AI, in general, to be entertaining when it is difficult to defeat (Buro 2003). Although for strong players that may be true, novice players will not enjoy being overwhelmed by the computer. For novice players, a game is most entertaining when the game is challenging but beatable (Scott 2002). "Difficulty scaling" is the automatic adaptation of a game, to set the challenge the game poses to a human player. When applied to game AI, difficulty scaling usually aims at achieving an "even game", i.e., a game wherein the playing strength of the computer and the human player match.

In complex games, such as Computer RolePlaying Games (CRPGs), the incorporation of advanced game AI is difficult. For these complex games most game-AI developers resort to scripts, i.e., lists of rules that are executed sequentially (Tozour 2002b). AI scripts are generally static, and thus cannot adapt the level of difficulty exhibited by the game AI to appeal to both novice and experienced human players.

In our research, we apply machine-learning techniques to improve the quality of game AI. When machine learning is used to allow opponents to adapt while the game is played, this is referred to as "online learning". Online learning allows the opponents to automatically repair weaknesses in their scripts that are exploited by the human player, and to adapt to changes in human player tactics. Unsupervised online learning is widely disregarded by commercial game developers (Woodcock 2002), even though it has been shown to be feasible for games (Demasi and Cruz 2002).

In prior work, we designed a novel technique called "dynamic scripting" that realises online adaptation of scripted game AI, in particular for complex games (Spronck, Sprinkhuizen-Kuyper, and Postma 2004a). In its original form, dynamic scripting optimises the game-playing strength of game AI. The present research investigates three different enhancements to the dynamic scripting technique that allow it to scale the difficulty level of the game AI to create an even game, namely (1) high-fitness penalising, (2) weight clipping, and (3) top culling.

The outline of the remainder of the paper is as follows. First, it describes dynamic scripting and the three difficulty-scaling enhancements. Then, the results obtained by applying the enhancements to dynamic scripting are presented and discussed. Finally, the paper concludes and points at future work.

DYNAMIC SCRIPTING

Online learning of game AI entails that the AI is adapted while the game is being played. In this section we present dynamic scripting as a technique that is designed specifically for this purpose. A detailed exposition of dynamic scripting is provided by Spronck *et al.* (2004a).

Dynamic scripting is an unsupervised online learning technique for games, that is computationally fast, effective, robust, and efficient (Spronck *et al.* 2004a). It maintains several rulebases, one for each opponent type in the game. The rules in the rulebases are manually designed using domain-specific knowledge. When a new opponent is generated, the rules that comprise the script controlling the opponent are extracted from the rulebase corresponding to the opponent type. The probability that a rule is selected for a script is proportional to the value of the weight associated with the rule. The rulebase adapts by changing the weight values to reflect the success or failure rate of the associated rules in scripts. A priority mechanism can be used to determine rule precedence. The dynamic scripting process is illustrated in figure 1 in the context of a game.

The learning mechanism of dynamic scripting is inspired by reinforcement-learning techniques (Russell and Norvig 2002). It has been adapted for use in games because "regular" reinforcement-learning techniques are not sufficiently efficient for online learning in games (Manslow

2002). In the dynamic-scripting approach, learning proceeds as follows. Upon completion of an encounter, the weights of the rules employed during the encounter are adapted depending on their contribution to the outcome. Rules that lead to success are rewarded with a weight increase, whereas rules that lead to failure are punished with a weight decrease. The remaining rules are updated so that the total of all weights in the rulebase remains unchanged.

Weight values are bounded by a range [$W_{\min}$,$W_{\max}$]. The size of the weight change depends on how well, or how badly, a team member behaved during the encounter. It is determined by a fitness function that rates a team member's performance as a number in the range [0,1]. The fitness function is composed of four indicators of playing strength, namely (1) whether the member's team won or lost, (2) whether the member died or survived, (3) the member's remaining health, and (4) the amount of damage done to the member's enemies. The new weight value is calculated as $W+\Delta W$, where W is the original weight value, and the weight adjustment ΔW is expressed by the following formula:

$$\Delta W = \begin{cases} -P_{\max} \cdot \dfrac{b-F}{b} & \{F < b\} \\ R_{\max} \cdot \dfrac{F-b}{1-b} & \{F \geq b\} \end{cases} \qquad (1)$$

where F is the fitness, b is the break-even value, and $R_{\max}$ and $P_{\max}$ are the maximum reward and maximum penalty respectively. The left graph in figure 2 displays the weight adjustment as a function of the fitness F.

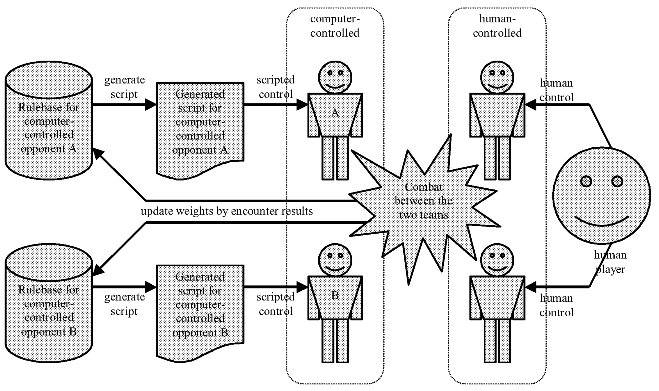

computer-controlled
human-controlled
Rulebase for computer-controlled opponent A
generate script
Generated script for computer-controlled opponent A
scripted control
A
human control
Combat between the two teams
update weights by encounter results
human player
Rulebase for computer-controlled opponent B
generate script
Generated script for computer-controlled opponent B
scripted control
B
human control

Figure 1: The dynamic scripting process. For each computer-controlled opponent a rulebase generates a new script at the start of an encounter. After an encounter is over, the weights in the rulebase are adapted to reflect the results of the fight.

DIFICULTY SCALING

Many games provide a "difficulty setting", i.e., a discrete value that determines how difficult the game will be. The purpose of a difficulty setting is to allow both novice and experienced players to enjoy the appropriate challenge the game offers. Usually the parameter influences opponents' strength and health. Very rarely the parameter influences opponents' tactics. Consequently, even on a "hard" difficulty setting, opponents exhibit inferior behaviour, despite their high physical strength and health.

This section describes how dynamic scripting can be used to create new opponent tactics while scaling the difficulty level of the game AI to the experience level of the human player. Specifically, it describes three different enhancements to the dynamic scripting technique that let opponents learn how to play an even game, namely (1) high-fitness penalising, (2) weight clipping, and (3) top culling.

High-Fitness Penalising

The weight adjustment expressed in formula (1) gives rewards proportional to the fitness value: the higher the fitness, the higher the reward. To elicit mediocre instead of optimal behaviour, the weight adjustment can be changed to give highest rewards to mediocre fitness values, and lower rewards or even penalties to high fitness values. With high-fitness penalising weight adjustment is expressed by formula (1), where F is replaced by F' defined as follows:

$$F' = \begin{cases} \dfrac{F}{p} & \{F \leq p\} \\ \dfrac{(1-F)}{p} & \{F > p\} \end{cases} \qquad (2)$$

where F is the calculated fitness value and p is the reward-peak value, i.e., the fitness value that should get the highest reward. The higher p is set, the more effective opponent behaviour will be. Figure 2 illustrates the weight adjustment as a function of F, with (left) and without (right) high-fitness penalising.

Since the optimal value for p depends on the tactic that the human player uses, we decided to let the value of p adapt to the perceived difficulty level of a game, as follows. Initially p starts at a value p_{init}. After every fight that is lost by the computer, p is increased by a small amount p_{inc}, up to a predefined maximum value $p_{\max}$. After every fight that is won by the computer, p is decreased by a small amount p_{dec}, down to a predefined minimum value $p_{\min}$.

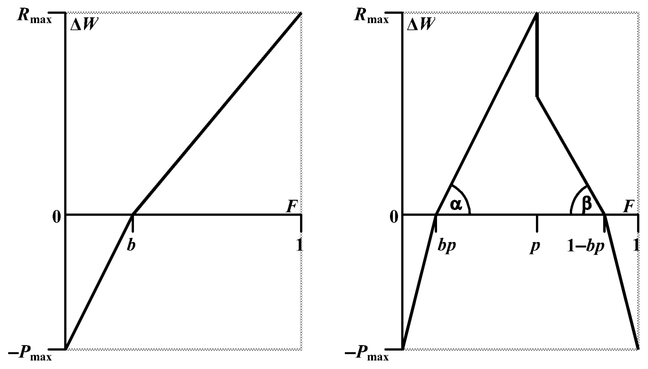


Figure 2: Graphs of the original weight-adjustment formula (left) and the high-fitness-penalising weight-adjustment formula (right). Angles α and β are equal.

Weight Clipping

The maximum weight value $W_{\max}$ determines the maximum level of optimisation a learned tactic can achieve. A high value for $W_{\max}$ allows the weights to grow to large values, so that after a while the most effective rules will almost always be selected. This will result in scripts that are close to optimal. A low value for $W_{\max}$ restricts weights in their growth. This enforces a high diversity in generated scripts, most of which will not be optimal.

Weight clipping automatically changes the value of W_{max}, with the intent to enforce an even game. It aims at having a low value for W_{max} when the computer wins often, and high value for W_{max} when the computer loses often. The implementation is as follows. After the computer won a fight, W_{max} is decreased by W_{dec} per cent (but not lower than the initial weight value W_{init}). After the computer lost a fight, W_{max} is increased by W_{inc} per cent.

Figure 3 illustrates the weight-clipping process and parameters. The shaded bars denote weight values for arbitrary rules on the horizontal axis. Before the weight adjustment, W_{max} changes by W_{inc} or W_{dec} per cent, depending on the outcome of the fight. After the weight adjustment, in figure 3 the weight value for rule 4 is too low, and will be increased to W_{min} (arrow ‘a’), while the weight value for rule 2 is too high, and will be decreased to W_{max} (arrow ‘b’).

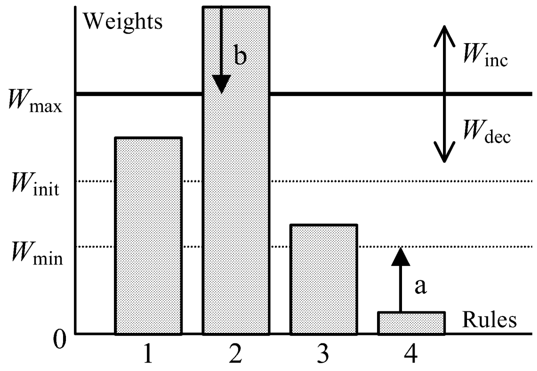


Figure 3: Weight clipping and top culling process and parameters.

Top Culling

Top culling is similar to weight clipping. It employs the same adaptation mechanism for the value of W_{max}. The difference is that top culling allows weights to grow beyond the value of W_{max}. However, rules with a weight greater than W_{max} will not be selected for a generated script. Consequently, when the computer-controlled opponents win often, the most effective rules will have weights that exceed W_{max}, and cannot be selected, and thus the opponents will use weak tactics. Alternatively, when the computer-controlled opponents lose often, rules with high weights will become selectable (again), and the opponents will use strong tactics.

In figure 3, contrary to weight clipping, top culling will leave the value of rule 2 unchanged (the action represented by arrow (b) will not be performed). However, rule 2 will be unavailable for selection, because its value exceeds W_{max}.

EXPERIMENTS

To evaluate the effect of the three difficulty-scaling enhancements to dynamic scripting, we employed a simulation of an encounter of two teams in a complex CRPG, closely resembling the popular BALDUR’S GATE games. We used this environment in earlier research to demonstrate the efficiency of dynamic scripting (Spronck *et al.* 2004a), and to test different measures to reduce the number of outliers that dynamic scripting occasionally generates (Spronck, Sprinkhuizen-Kuyper and Postma 2004b). Our evaluation experiments aimed at assessing the performance of a team controlled by the dynamic scripting technique using a difficulty-scaling enhancement, against a team controlled by static scripts. If the difficulty-scaling enhancements work as intended, dynamic scripting will balance the game so that the number of wins of the dynamic team is roughly equal to the number of losses. In the simulation, we pitted the dynamic team against a static team that uses one of five, manually designed, basic tactics (named “offensive”, “disabling”, “cursing”, “defensive” and “novice”), or one of three composite tactics (named “random party”, “random character” and “consecutive party”).

Of the eight static team’s tactics the most interesting in the present context is the “novice” tactic. This tactic resembles the playing style of a novice BALDUR’S GATE player. While it normally will not be defeated by arbitrarily picking rules from the rulebase, many different tactics exist that can be employed to defeat it, which the dynamic team will quickly discover. Without difficulty-scaling, the dynamic team’s number of wins will greatly exceed its losses.

In our experiments we initialised W_{max}=2000. We set W_{init}=100, W_{min}=0, W_{inc}=10%, W_{dec}=10%, p_{init}=0.7, p_{min}=0.65, p_{max}=0.75, p_{inc}=0.01, p_{dec}=0.01, R_{max}=100, P_{max}=100, and b=0.3. We employed the same fitness function as in previous research (Spronck *et al.* 2004a), andd dynamic scripting with fitness propagation fallback (Spronck *et al.* 2004b).

For each of the tactics, we ran 100 tests in which dynamic scripting was enhanced with each of the three difficulty-scaling enhancements, and, for comparison, also without difficulty-scaling enhancements (“optimisation”). Each test consisted of a sequence of 150 encounters between the dynamic team and the static team. Because in each of the tests the dynamic scripting process starts with a rulebase with all weights equal, the first 50 encounters were used for finding a balance of well-performing weights. We recorded the number of wins of the dynamic team for the last 100 encounters. The results of these tests are displayed in table 1. Histograms for the tests with the “novice” tactic are displayed in figure 4.

To be recognised as an even game, we decided that the average number of wins over all tests must be close to 50. To take into account random fluctuations, in this context “close to 50” means “within the range [45,55]”. In table 1, all cell values indicating an even game are marked in bold font. From table 1 the following four results can be derived.

First, optimisation (dynamic scripting without a difficulty-scaling enhancement) results in wins significantly exceeding losses for all tactics except for the “consecutive party” tactic (with a reliability > 99.9%; Cohen 1995). The “consecutive party” tactic is the most difficult tactic to defeat (Spronck *et al.* 2004a). Note that the fact that, on average, dynamic scripting plays an even game against the “consecutive party” tactic is not because it is unable to consistently defeat this tactic, but because dynamic scripting continues learning after it has reached an optimum. Therefore, it can “forget” what it previously learned, especially against an optimal tactic like the “consecutive party” tactic.

Second, high-fitness penalising performs considerably worse than the other two enhancements. It cannot achieve an even game against six of the eight tactics.

Tactic	Optimisation		Fitness Penalising		Weight Clipping		Top Culling	
	Wins	St.dev.	Wins	St.dev.	Wins	St.dev.	Wins	St.dev.
Offensive	61.2	16.4	**46.0**	15.1	**50.6**	9.4	**46.3**	7.5
Disabling	86.3	10.4	56.6	8.8	67.8	4.5	**52.2**	3.9
Cursing	56.2	11.7	42.8	9.9	**48.4**	6.9	**46.4**	5.6
Defensive	66.1	11.9	39.7	8.2	**52.7**	4.2	**49.2**	3.6
Novice	75.1	13.3	**54.2**	13.3	**53.0**	5.4	**49.8**	3.4
Random Party	55.8	11.3	37.7	6.5	**50.0**	6.9	**47.4**	5.1
Random Character	58.8	9.7	44.0	8.6	**51.8**	5.9	**48.8**	4.1
Consecutive Party	**51.1**	11.8	34.4	8.8	**48.7**	7.7	**45.0**	7.3

Table 1: Experimental results of testing the three difficulty-scaling enhancements to dynamic scripting on eight different tactics, compared to dynamic-scripting optimisation. For each combination the table shows the average number of wins over 100 tests, and the associated standard deviation. The cells marked with a bold font indicate the enhancement to be successful in forcing an even game against the associated tactic.

Third, weight clipping is successful in enforcing an even game against seven out of eight tactics. It does not succeed against the "disabling" tactic. This is caused by the fact that the "disabling" tactic is so easy to defeat, that even a rulebase with all weights equal will, on average, generate a script that defeats this tactic. Weight clipping can never generate a rulebase worse than "all weights equal".

Fourth, top culling is successful in enforcing an even game against all eight tactics.

From the histograms in figure 4 we derive the following result. While all three difficulty-scaling enhancements manage to, on average, enforce an even game against the "novice" tactic, the number of wins in each of the tests is much more "spread out" for the high-fitness-penalising enhancement than for the other two enhancements. This indicates that the high-fitness penalising results in a higher variance of the distribution of won games than the other two enhancements. The top-culling enhancement seems to yield the lowest variance. This is confirmed by an approximate randomisation test (Cohen 1995, section 6.5), which shows that against the "novice" tactic, the variance achieved with top culling is significantly lower than with the other two enhancements (reliability > 99.9%). We observed similar distributions of won games against the other tactics, except that against some of the stronger tactics, a few exceptional outliers occurred with a significantly lower number of won games. The rare outliers were caused by dynamic scripting, occasionally needing more than the first 50 encounters to find well-performing weights against a strong static tactic.

We further validated the results achieved with top culling, by implementing dynamic scripting with the top-culling enhancement in a state-of-the-art computer game, NEVERWINTER NIGHTS (version 1.61), testing it against the game AI implemented by the game developers, with the same experimental procedure as used in the simulation environment. Ten optimisation tests resulted in an average number of wins of 79.4 out of 100, with a standard deviation of 12.7. Ten tests with the top-culling enhancement resulted in an average number of wins of 49.8 out of 100, with a standard deviation of 3.4. Therefore, our simulation results are supported by the NEVERWINTER NIGHTS tests.

DISCUSSION

Of the three different difficulty-scaling enhancements we conclude the top-culling enhancement to be the best choice. It has the following three advantages: (1) it yields results with a very low variance, (2) it is easily implemented, and (3) of the three enhancements, it is the only one that manages to force an even game against inferior tactics.

Obviously, the worst choice is the high-fitness-penalising enhancement. In an attempt to improve high-fitness penalising, we performed some tests with different ranges and adaptation values for the reward-peak value *p*, but these worsened the results. However, we cannot rule out the possibility that with a different fitness function high-fitness penalising will give better results.

An additional possibility with weight clipping and top culling is that they can be used to set a desired win-loss ratio, simply by changing the rates with which the value of W_{max} fluctuates. For instance, by using top culling with W_{dec}=30% instead of 10%, leaving all other parameters the same, after 100 tests against the "novice" tactic, we derived an average number of wins of 35.0 with a standard deviation of 5.6.

In previous research we concluded that dynamic scripting is suitable to be applied in real commercial games to automatically optimise tactics (Spronck *et al.* 2004a). With a difficulty-scaling enhancement, the usefulness of dynamic scripting is improved significantly, since it can now also be used to scale the difficulty level of a game to the experience level of the human player. Notwithstanding this, a difficulty-scaling enhancement should be an optional feature in a game, that can be turned off by the player, for the following two reasons: (1) when confronted with an experienced player, the learning process should aim for optimal tactics without interference from a difficulty-scaling enhancement, and (2) some players will feel that attempts by the computer to force an even game diminishes their accomplishment of defeating the game, so they may prefer not to use it.

CONCLUSIONS AND FUTURE WORK

In this paper we proposed three different enhancements to the dynamic scripting technique that allow scaling the difficulty level of game AI, namely (1) high-fitness penalising, (2) weight clipping, and (3) top culling. From our experiments we conclude that a difficulty-scaling enhancement to dynamic scripting can be used to let the learning process scale the difficulty level of the tactics employed by the computer-controlled opponents to match the game-playing skills of the human player (i.e., to force an even game). Of the three difficulty-scaling enhancements tested, high-fitness penalising was, in general, unsuccessful, but the other two performed well. Top culling gave the best results. We also discovered that both weight clipping and top culling, besides forcing an even game, can be used to set a different win-loss ratio, by tuning a single parameter. We conclude that dynamic scripting, using top culling, can enhance the entertainment value of games by scaling the difficulty level of the game AI to the playing skill of the human player.

In future work, we intend to apply dynamic scripting, including difficulty scaling, in other game types than CRPGs. We will also investigate whether offline machine learning techniques, which can be very effective in designing tactics (Spronck, Sprinkhuizen-Kuyper and Postma 2003), can be used to "invent" completely new rules for the dynamic scripting rulebase. Finally, we aim to investigate the effectiveness and entertainment value of dynamic scripting in games played against actual human players. While such a study requires many subjects and a careful experimental design, the game-play experiences of human players are important to convince game developers to adopt dynamic scripting in their games.

REFERENCES

Buro, M. 2003. "RTS Games as a Test-Bed for Real-Time AI Research." *Proceedings of the 7th Joint Conference on Information Science* (eds. K. Chen *et al.*), pp. 481–484.

Cohen, P.R. 1995. *Empirical Methods for Artificial Intelligence.* MIT Press, Cambridge, MA.

Demasi, P. and A.J. de O. Cruz. 2002. "Online Coevolution for Action Games." *International Journal of Intelligent Games and Simulation* (eds. N.E. Gough and Q.H. Mehdi), Vol. 2, No. 2. University of Wolverhampton and EUROSIS , pp. 80–88.

Manslow, J. 2002. "Learning and Adaptation." *AI Game Programming Wisdom* (ed. S. Rabin). Charles River Media, pp. 557–566.

Russell, S. and P. Norvig. 2002. *Artificial Intelligence: A Modern Approach*, Second Edition. Prentice Hall, Englewood Cliffs, NJ.

Schaeffer, J. 2001. "A Gamut of Games." *AI Magazine*, Vol. 22 No. 3, pp. 29–46.

Scott, B. 2002. "The Illusion of Intelligence." *AI Game Programming Wisdom* (ed. S. Rabin). Charles River Media, pp. 16–20.

Spronck, P., I. Sprinkhuizen-Kuyper and E. Postma. 2003. "Improving Opponent Intelligence Through Offline Evolutionary Learning." *International Journal of Intelligent Games and Simulation* (eds. N.E. Gough and Q.H. Mehdi), Vol. 2, No. 1. University of Wolverhampton and EUROSIS, pp. 20–27.

Spronck, P., I. Sprinkhuizen-Kuyper and E. Postma. 2004a. "Online Adaptation of Game Opponent AI with Dynamic Scripting." *International Journal of Intelligent Games and Simulation* (eds. N.E. Gough and Q.H. Mehdi), Vol. 3, No. 1. University of Wolverhampton and EUROSIS, pp. 45–53.

Spronck, P., I. Sprinkhuizen-Kuyper and E. Postma. 2004b. "Enhancing the Performance of Dynamic Scripting in Computer Games." *Entertainment Computing – ICEC 2004* (ed. M. Rauterberg). LNCS 3166, Springer-Verlag, Berlin, Germany, pp. 296–307.

Tozour, P. 2002a. "The Evolution of Game AI." *AI Game Programming Wisdom* (ed. S. Rabin). Charles River Media, pp. 3–15.

Tozour, P. 2002b. "The Perils of AI Scripting." *AI Game Programming Wisdom* (ed. S. Rabin). Charles River Media, pp. 541–547.

Woodcock, S. 2000. "Game AI: The State of the Industry." *Game Developer Magazine,* August 2000.

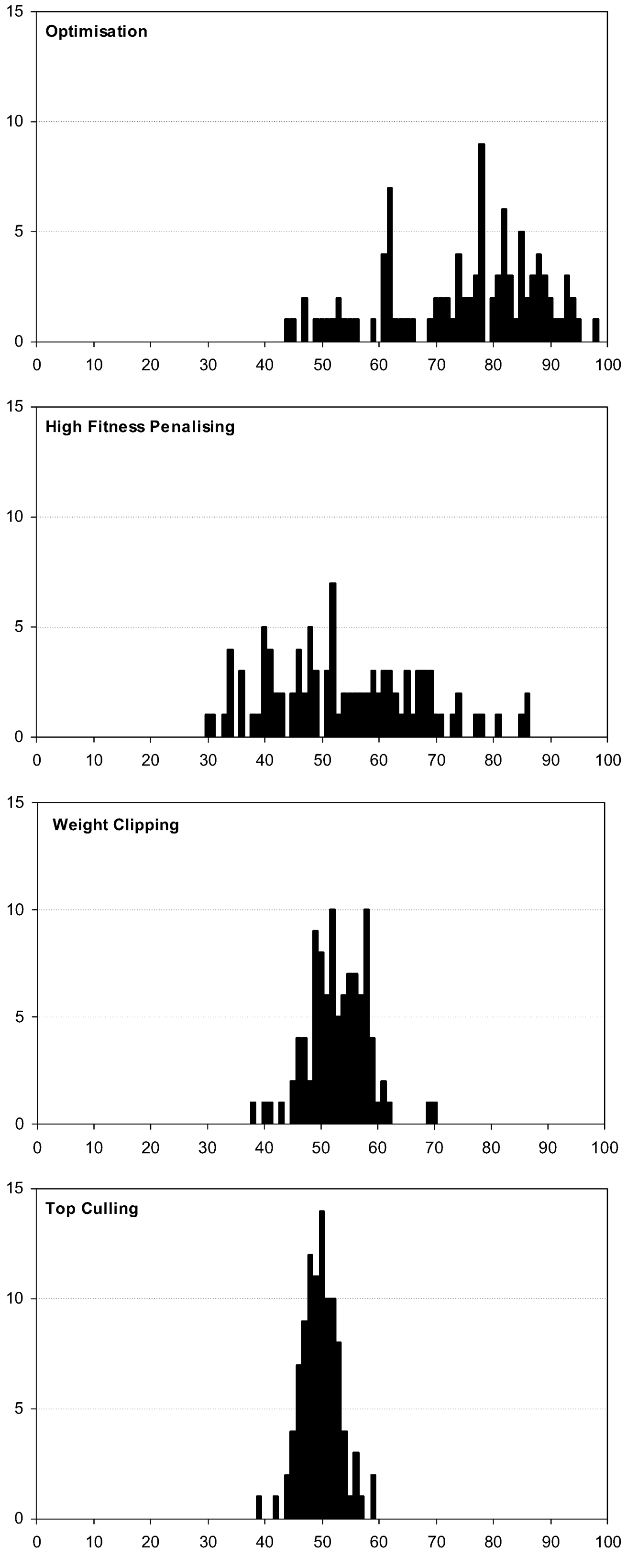


Figure 4: Histograms of the achieved number of wins over 100 tests against the "novice" tactic, using dynamic scripting without (top) and with difficulty-scaling enhancements.

Development of a Cooperative Multiagent System to Facilitate Leadership Roles in Computer Entertainment

Nick Baker and Abdennour El Rhalibi
School of Computing and Mathematical Sciences
Liverpool John Moores University
nick@nbaker.net , a.elrhalibi@livjm.ac.uk

ABSTRACT

This paper contains a synopsis of a Master's level research project (Baker et al. 2004) involving the use of academic artificial intelligence research to create realistic portrayals of leadership roles in computer game entertainment. In response to a perceived lack of depth and realism in the team relationship dynamics of modern gaming, the project developed a human agent architecture, multiagent system, and demonstrative game application. The agent architecture was based partially on research into social psychology, and utilized emotion and belief representations to drive action selection. Agent interaction and relationship development was produced on the basis of the Iterated Prisoner's Dilemma (IPD), through which a team's success came to be determined by its members' choices to cooperate or compete with its leader. A produced game application illustrated the operation of the developed architecture within the context of a political street protest. A set of evaluation scenarios were devised to test the success of the project work within this game application, and ultimately found it to be successful in achieving a modest level of realistic team-based reasoning and interaction.

INTRODUCTION

A common witticism concerning teamwork is that there is no 'I' in team—successfully working as a team requires that its individual members put their own interests aside and commit to each other. While if the idea is most certainly cliché, portrayals of teamwork and leadership roles in computer gaming remain noticeably self-centred. While a number of recent games have featured team or squad-based gameplay, few have challenged players as leaders in a realistic and dynamic way. The emphasis remains on the player as the only member of the team that really matters, as well as the only one really capable of surmounting the challenges ahead. The primary element lacking from such games are both the interpersonal aspects of team work situations and the evolving nature of a team relationship, both among its members and with its leader.

This project sought to take some initial steps towards modelling team relationships and leadership roles more realistically in gaming applications. An oft-referenced presentation given by Clark Gibson and John O'Brien at the 2001 Game Developer's Conference describes team AI as the, "next step in game AI," and that it is, "needed to be competitive," and, "increase realism," (Gibson et al. 2001). While game developers within the industry may be content to build this team AI from scratch, a significant body of research is already available from the agent-based academic AI community. The project work sought to harness the results of such research for this purpose. Its goal was to devise an agent architecture and multiagent system to model the chaotic, evolving nature of teamwork dynamics through representations of emotions, beliefs, and relationships. Previous research efforts into human AI, rule-based system theory, and even the Iterated Prisoner's Dilemma were all incorporated into its development. The successful operation of the produced system was illustrated in the context of a custom-made game application involving a team of protestors staging a political demonstration. The ultimate goal of the project work was not only to make initial steps into realizing better teamwork dynamics, but also to rectify the, "mutual lack of interest between serious game developers and academic AI researchers," by highlighting the suitability of agent-based approaches to AI in gaming (Laird et al. 2000).

LITERATURE REVIEW

Foundational research for the project was conducted within the broad topic areas of social psychology and human agent AI. In addition, current efforts to model team behaviour within the games industry were reviewed and accounted for. Some of the major findings produced by the research are presented in this section.

Social psychology was quickly identified as a primary resource for understanding teamwork and leadership dynamics. A huge body of literature exists concerning how to be an effective leader or manage a successful team, but it was deemed to be too high-level for this initial AI development. At the same time, more specialized modern research into team behaviour was thought to be too specific and nuanced for successful use. The best alternative was found in some of the core theories of social psychology—the branch of human psychology dedicated to understanding how human beings behave in response to each other. Among the topics and theory selected as appropriate for use in the project were conformity, social facilitation and loafing, group norms and cohesiveness, group polarization, and the various factors that influence individual cooperation in group settings. Some of the findings validated common conceptions of teamwork—larger groups elicit more powerful pressures to conform, more cohesive groups generally perform better, and individual behaviour relative to a group is partly determined by personality, past experience, and situational factors. Other findings provided some interesting theoretical backing to the project work. For example, the presence of a single defector in a group decision-making process has been found to reduce the pressure of conformity on the other members by 80 percent (Brehm et al. 2002). Another finding deemed to be interesting was that group cohesiveness, or the strength of a team relationship, can be a negative factor in group decision-making as it can influence members to avoid disagreement and neglect their better judgment in order to maintain group compliance (Brehm et al. 2002). Research into social psychology also aided in the identification of social dilemmas such as the Prisoner's Dilemma, which was later chosen for use in modelling the project's team member interaction.

Perhaps the most important body of research surveyed for the project work was that of human AI—the efforts of a relatively small branch of the academic AI community to devise more and more realistic simulations of human behaviour. The topic area was approached through the examination of three exemplary projects developed for applications in computer animation, gaming, and sociological research simulations. Each of the projects will be briefly reviewed here.

For computer animation the selected project was a knowledge-based human crowd animation system developed by S.R. Musse and D. Thalmann in 1997 (Musse et al. 2004).

The Oz Project was surveyed for an understanding of the state of human AI pursued for applications in computer gaming. The Oz Project's principal 'Tok' agent architecture is pictured in Figure 1. Among the most notable details garnered from its study were its use of a planning system to select appropriate agent actions given both emotional and sensory stimulus, its greater inventory of emotions (including hope, fear, pride, and admiration), and its use of value constructs like standards and attitudes to give its agents individual personality and elicit continual influences on behaviour (Bates et al. 1994).

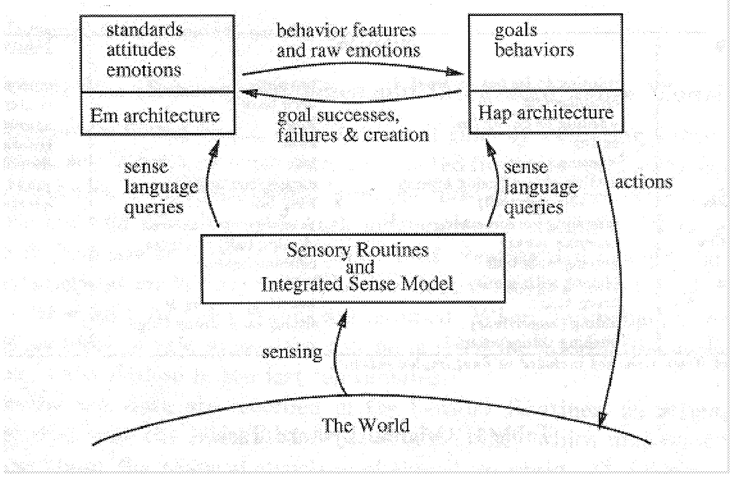


Figure 1: The Oz Project's 'Tok' Agent Architecture (Bates et al. 1994)

The final and arguably most complex of the reviewed projects was that of the Ackoff Center for Advancement of Systems Approaches (ACASA) in developing an integrative agent architecture and multiagent system for purposes in both gaming and social research. The research group's architecture expanded on the Oz Project's dual subsystem model with the use of four subcomponents, designed to give their agents specialized physiological, emotive, cognitive, and motor/expressive reasoning in an interconnected framework. This particular arrangement influenced the project work in its own divided architecture and introduced the importance of stress and physiology on human AI. One particularly interesting aspect of the ACASA's work involved the use of detailed 'concern ontologies' to specify the moral values of a given agent and guide its emotional deliberation—a mechanism the researchers intended to be complex enough to model existing literature such as, "the ten commandments or the Koran for a moral code," or, "military doctrine for action guidance," (Silverman et al. 2002). The ACASA similarly demonstrated the operation of their architecture through the use of a street protest simulation environment.

In addition to surveying academic AI resources for simulating human behaviour the project reviewed the current state of the team AI within the games industry to provide context for its work. The two primary methods highlighting through this research were centralized and decentralized approaches. In the former, game AI developers manage team behaviour through the use of a rigid command hierarchy in which orders flow downward from a group commander to their subordinates and environmental information flows upward (Reynolds et al. 2002). Though strategically sufficient and arguably realistic in modelling teamwork situations in the military, this first method is poorly suited to more dynamic team representations due to its reliance on strict obedience and homogeneous agents. Decentralized approaches, the second primary method, involve an opposite arrangement in which, "interactions between squad members, rather than a single squad leader, determine the squad behaviour. And from this interaction, squad manoeuvres emerge," (Van Der Sterren et al. 2002). Here the emphasis is on emergent team behaviour which develops as individual agents share environmental information and select individual courses of action in response to those of their cooperative team members. The obvious problem with this second approach is that its lack of an explicit leader and the self-interested character of the team agents make concerted, strategic team behaviour much more difficult to achieve. In response to these two approaches the project was able to identify and pursue a middle approach able to balance the structure of hierarchical organization with the versatility of individual member reasoning.

ANALYSIS AND DESIGN

Having reviewed past efforts in developing human and team-based AI, the project moved on to its analysis and design stage. Three principal components required design: (1) the human agent architecture; (2) the multiagent system, and; (3) the demonstrative 3D game application.

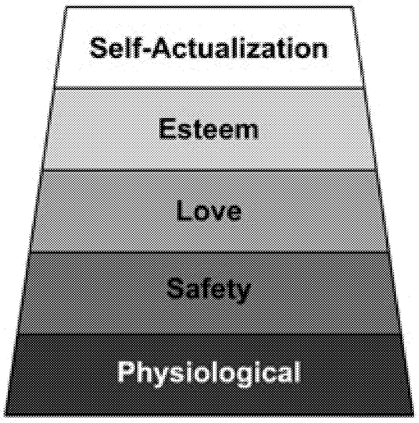


Figure 2: Abraham Maslow's Hierarchy of Needs (Maslow et al. 1954)

The project component requiring the most elaborate design was undoubtedly the human agent architecture. The primary challenge in its design was in integrating a variety of stimulus (including emotions, beliefs, and physical perception) with specialized team-based relationship representations and having the resulting system able to make appropriate behavioural decisions for its agent. The architecture itself came to be modelled after Abraham Maslow's famous 'Hierarchy of Needs' (pictured in Figure 2) due to its own integration of emotion and relationship-based reasoning as well as to its ability to balance both reactive and deliberative intelligence. Agent reasoning was thus designed to take place using a tiered, hierarchical arrangement in which the agent would perform one particular type of reasoning at each level. The resulting emotions or observations from each reasoning level would then be collected by the agent and used in selecting the most appropriate action. A given reasoning operation begins at the bottom level (Physiology) and only advances to the upper levels if the assessment results in non-critical values/observations. Arranged in this way, the agent can quickly obviate more complex reasoning when its lower levels determine it is in immediate physical danger.

In the following section we describe how this model is used as architecture for our team-oriented agent reasoning model.

AGENT REASONING MODEL

Figure 3 illustrates the design for the agent reasoning model. We can see in the left side the agents' default, low-level reasoning; and in the right side the team-based agent reasoning model (Baker et al. 2004).

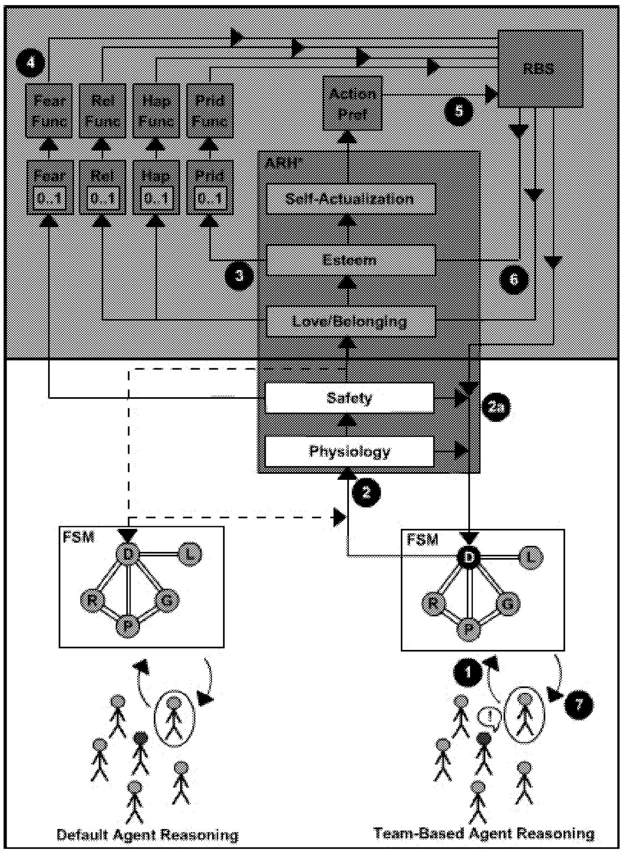


Figure 3: Overview of Team Based Agent Reasoning

Team-Based Agent Reasoning

The opportunity for more elaborate agent reasoning is introduced in the designed system once the team leader selects an action for their team to perform. Figure 3 presents a conceptual illustration of this more complex reasoning process, which is the heart of the project work. Here a number of new visual elements are immediately evident. First, the agent reasoning hierarchy is seen as fully-fleshed out in three new levels: (1) a 'Love/Belonging' level; (2) an 'Esteem' level, and; (3) a 'Self-Actualisation' level. At this point it should be evident that Abraham Maslow's Hierarchy of Needs has had a particular influence on the agent's reasoning design. Other noticeable visual elements are the set of emotion parameters positioned on the upper left hand side of the figure. These are the agent's emotions, represented as numeric values normalized over the range of 0 to 1, and linked to a set of emotion functions. Finally, on the right hand side of the figure is a box signifying the design's rule-based system (RBS), which handles its core Planning and action selection functionality.

All of the components and processing pictured in the grey upper square of the figure should be understood as operating within this RBS, but it is conceptually represented as a separate entity to symbolize its role in making the final action selection for the inquiring agent.

The operation of this more elaborate reasoning design will be demonstrated by walking through the seven-step sequence highlighted on Figure 3. These steps are explained as follows:

(1) The agent looks to the FSM to provide it with an action to perform. The leader of the agent's team has been identified as suggesting an action for the team to perform, and the agent is trying to determine whether it should cooperate or defect and perform its own individual actions. The same reasoning limitations imposed by certain states in the FSM apply, but should the agent be placed into the 'D' state and allowed access to higher level reasoning, the core team-based reasoning of the project can really begin.

(2) The reasoning continues into the agent reasoning hierarchy, where it attempts to scale the various reasoning levels by satisfying their specialized reasoning assessments. The bottom two levels of 'Physiology' and 'Safety' must be surpassed; the project agents are designed to always remain conscious of their physical and environmental conditions in conducting their higher-level reasoning.

(2a) This supplemental step is included to indicate that if either of the two bottom levels of the reasoning hierarchy detect extraneous situations that require immediate response, they can supersede the upper levels and immediately return an action to the agent to perform.

(3) With the agent's physical state secure, reasoning can continue into the upper levels of the hierarchy, which are now managed within the system RBS. The upper levels of the agent reasoning hierarchy are more concerned with processing and updating emotions than recognizing particular environmental conditions. They generally look at previous actions and use recorded stimulus to update the numeric emotion values pictured on the left. The 'Love/Belonging' level is the most important in the hierarchy, as it is where the bulk of the agent's team-based reasoning takes place. It generates two emotions; (1) a relationship strength value ('Rel' in Figure 3) indicating the status of an agent's relationship with their team, and; (2) a happiness value ('Hap' in the figure) that is calibrated relative to the success of the team's collective efforts. The fourth reasoning level, entitled the 'Esteem' level, concerns an agent's self-confidence and generates feelings of pride ('Prid' in Figure 3). It should be noted at this time that the 'Safety' reasoning level generates a fear emotion parameter (provided that reasoning successfully surmounts the level) that can be utilized in higher level reasoning as representative of the danger of the current situation.

(4) Here the normalized emotion values are augmented by a set of agent-specific emotion progression functions. Different agents utilize different functions to model different personality traits. For example, the project provides three functions for the fear emotions interpretation: (1) a 'normal' median function; (2) an enhanced 'coward' function, and; (3) a reduced 'hero' function. These functions allow the basic normalized emotion parameters to be managed in the same manner by the reasoning hierarchy for every agent, while still providing a mechanism for emotional reasoning differentiation. At this step in the sequence the functions are applied and the resulting emotion values are loaded into the RBS in preparation for final action selection reasoning.

(5) The 'Self-Actualisation' level serves mainly as a placeholder in the hierarchy for the final reasoning process. Its main contribution is to load the agent's action preference into the RBS, which is another agent personality detail that will be incorporated into the action selection reasoning. Within the RBS at this step are: (1) the agent's augmented emotional parameters; (2) the agent's action preference, and; (3) the leader's specified action, which is loaded for all of the team agents prior to their team-based reasoning operations. This stimulus is collectively assessed within the RBS at this step, with the final goal of either deciding to cooperate with the leader's suggested action or choosing instead to perform an individual action.

(6) The action selected by the RBS' processing is returned to the agent via the FSM. However, en route the action selection is reviewed by the 'Esteem' and 'Love/Belonging' reasoning levels so that they can better perform their emotional assessments for the next reasoning opportunity. The basic idea is that these levels look at the action and adjust their team relationship, happiness, and pride emotions depending on whether or not the agent ultimately decided to cooperate or whether the final action was consistent with their action preferences or not.
(7) The agent receives their action instruction from the FSM and carries it out amongst the team agents. The reasoning sequence is complete.

Finite States Machine
The actual behaviour of the human agents was designed to be managed through a low-level finite state machine (FSM).

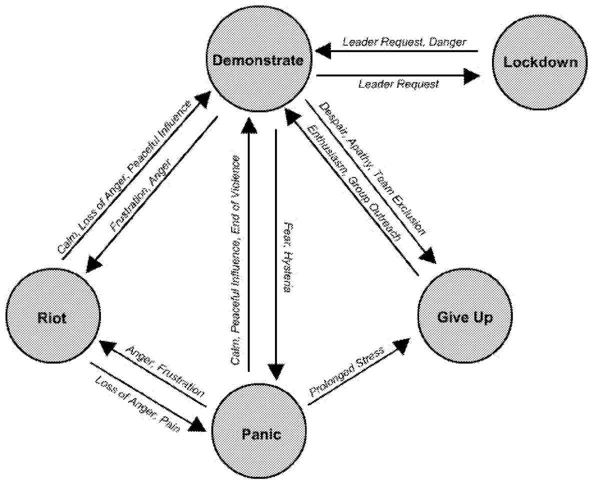


Figure 4: Agent FSM Design

There are a number of reasons for this arrangement. First, the computational complexity of the higher-level reasoning processes was expected to be too significant to carry out for each execution cycle. The FSM provided a means through which a human agent could be placed into a state of action that could continue until its next high-level reasoning opportunity. Second, while the selection of actions by the human agents was intended to be governed through emotional construal, it was thought that this process could be better organized through state representations and transitions.
The FSM for the project's activist agents is pictured in Figure 4.

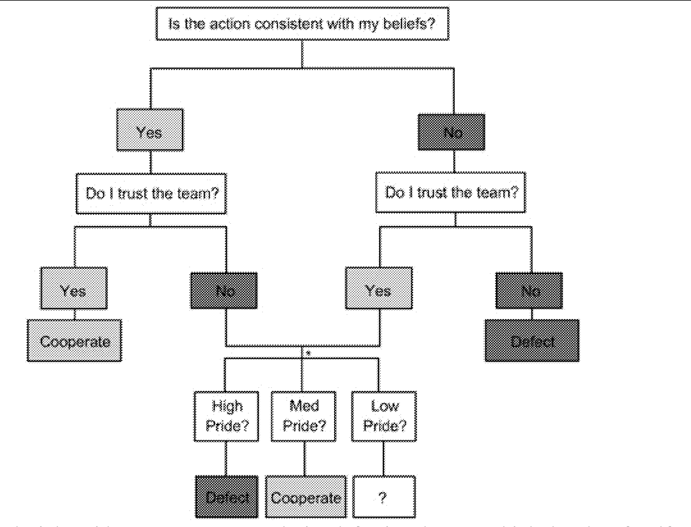
MODEL AGENT INTERACTIONS

Agent interaction and relationship modelling is driven primarily through the use of the Iterated Prisoner's Dilemma (IPD) (Axelrod 1957).

For the project's modelling of team work situations the IPD was chosen for use as an interesting and powerful means of managing agent interaction and team cooperation. Its adaptation for use in understanding team dynamics is fairly simple. Conditions that influence a team member's decision to cooperate with their team include personal and cultural differences, situational factors, and team dynamics. Those same factors are echoed in the project's adaptation of the IPD. The dilemma is posed to the individual team members when the player-controlled leader proposes an action for the team to carry out together. The individual agents must decide whether they should cooperate with the leader's suggestion or defect and ignore their request, deciding instead to act in a manner of their own choosing. The final factor, team dynamics, is modelled through the use of norms—a concept for individual interaction commonly discussed in social psychology as well as in IPD-based research (Axelrod 1957) (Brehm et al. 2002).

Table 1: Leadership and Team AI Design

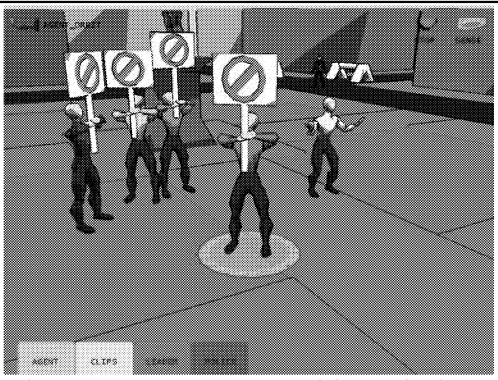
Leader Choice	Effect on Agent A		Effect on Agent B	
	Happiness & Love/Belonging	**Pride**	**Happiness & Love/Belonging**	**Pride**
A's Action*	+ + Team majority participation	+ Increase, personal beliefs validated	+ Team majority participation	- Decrease, Personal beliefs betrayed
	- Team minority participation		- - Team minority participation	
Action*	+ Team majority participation	- Decrease, personal beliefs betrayed	+ + Team majority participation	+ Increase, personal beliefs validated
	-- Team minority participation		- - Team minority participation	
Other Action (careless)	+ Team majority participation	- Decrease, personal beliefs betrayed	+ Team majority participation	- Decrease, sonal beliefs betrayed
	- - Team minority participation		- - Team minority participation	

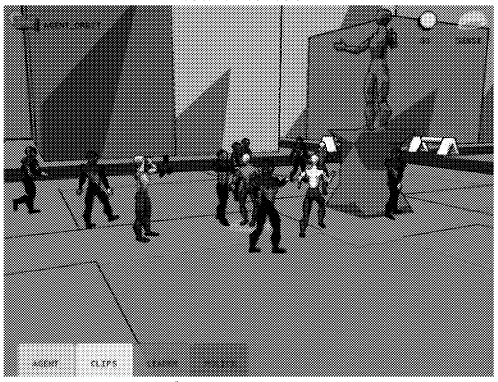


*High pride assessments result in defection because high levels of self-esteem make the agent take the dominant individual action in either situation. Medium pride assessments result in cooperation because the agent is willing to give the team, or a conflicting action, a chance. The result of low pride assessments depends on which is lower, the agent's happiness or its pride; in low pride situations the agent will try to make the choice that bolsters the lower of the two emotional values.

Figure 5: Simplified Team AI Diagram

According to the project's design, being a good leader boils down to selecting the appropriate action for the team to perform given their varied personalities and preferences. Such actions will limit defection in these agents because participating in them will positively impact their happiness as well as their pride assessment. Table 1 and Figure 5 are provided to help illustrate the IPD-based team AI design.


Leader selects action 'Sign' to gain confidence of 'Classical' class team member


The police agents scatter the activist team, using aggressive physical force and causing the agents to respond with evasive actions.

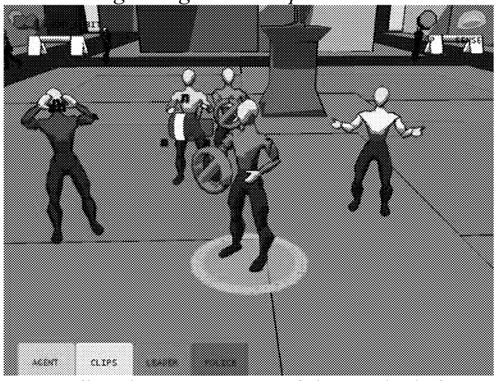

Leader repeatedly selects action 'Vandalize', which does not match any of the team agents' classes and the destructive relationship effects become evident in the reluctance of the majority of the agents to cooperate

Figure 6: "Demonstrate!" Application screenshots

DEMONSTRATE! GAME APPLICATION

The design of the demonstrative 3D game application—seen as the key to illustrating the successful operation of the complex human agent architecture to interested parties—was developed alongside the artificial intelligence it was intended to support (a screenshot of the application developed as test-bed can be seen in Figure 6). Ironically entitled 'Demonstrate!' the game was intended from very early stages in the project's development to involve a political street demonstration setting. The political demonstration setting was seen as providing the potential for a variety of agent personality types and motivations, a vast repertoire of different actions and emotional expression, and an interesting new context for developing teamwork situations and reasoning.

More detailed information about the design and implementation of the model, along with an extensive set of scenarios and evaluation can be found in (Baker et al. 2004), showing the relative success of the model and the agents interaction.

CONCLUSIONS

"The Development of a Cooperative Multiagent System to Facilitate Leadership Roles in Computer Entertainment" project (Baker et al. 2004) set out to investigate agent-based artificial intelligence techniques to simulate realistic team dynamics for use in computer gaming.

The success of the developed AI and its expression within the demonstrative 3D game application are indicative of the many strengths of the work. The constructed reasoning design was successful in achieving a modest level of team-based agent reasoning and behaviour, as demonstrated in the sections on its evaluation (Baker et al. 2004). Leadership and team dynamics remain a complex challenge for artificial intelligence developed within the academic AI community or the games industry. Some initial steps have been made by the project work, but significant challenges lie ahead.

The authors can conceive of a huge number of areas in which the project work can be improved. The developer sincerely hopes that it may prove to be of some merit in influencing new research directions or game designs that recognize the potential that resides in teamwork and leadership as resources for driving future innovation in artificial intelligence.

REFERENCES

Axelrod, Robert, *The Evolution of Cooperation*, Basic Books, Inc., Publishers, New York, NY, USA.

Nick Baker and Abdennour El Rhalibi (2004), *The Development of a Cooperative Multiagent System to Facilitate Leadership Roles in Computer Entertainment*, ACM GDTW 2004 International Conference, Liverpool John Moores University. November 2004.

Bates, J., Loyall, A.B., and Reilly, W.S. "*An Architecture for Action, Emotion and Social Behavior*," *Citeseer*. (1994) Brehm, Sharon S., Kassin, Saul M., and Fein, Steven. *Social Psychology*. Boston, MA, USA: Houghton Mifflin Company, 2002.

Brehm, Sharon S., Kassin, Saul M., and Fein, Steven, (2002), *Social Psychology*, Houghton Mifflin Company, Boston,, USA.

Gibson, Clark, and O'Brien, John. "*The Basics of Team AI*," *Game Developers Conference Proceedings*, 2001. (17 Nov 2003) www.gdconf.com/archives/2001/o'brien.ppt.

Laird, John E and Pottinger, David C. "*Game AI: The State of the Industry, Part Two*" Gamasutra.com. 8 Nov 2000.
http://www.gamasutra.com/features/20001108/laird_03.htm .

Maslow, Abraham H. *Motivation and Personality*. New York, NY, USA: Harper & Row Publishers, Inc., 1954.

Musse, S.R., and Thalmann, D. "*A Model of Human Crowd Behavior Group Inter-Relationship and Collision Detection Analysis*", *Citeseer*. (1997). (19 Jun 2004) http://citeseer.ist.psu.edu/musse97model.html
http://citeseer.ist.psu.edu/bates94architecture.html

Reynolds, J. "*Tactical Team AI Using A Command Hierarchy*", *AI Game Programming Wisdom*, Rubin, S. Hingham, MA, USA: Charles River Media, Inc., Hingham, MA, 2002.

Reilly, W. Scott, and Bates, Joseph, (1992), "*Building Emotional Agents*", Technical Report CMU-CS-92-143, School of Computer Science, Carnegie Mellon University, Pittsburgh, USA.

Silverman, B.G., et al. "*Human Behavior Models for Game-Theoretic Agents: Case of Crowd Tipping*", *Citeseer*. (2002) http://citeseer.nj.nec.com/silverman02human.html

Van Der Sterren, W. "*Squad Tactics: Team AI and Emergent Manuevers*", AI Game Programming Wisdom, Rubin, S. Hingham, MA, USA: Charles River Media, Inc., 2002.

Will Wright (Maxis Software), "*Under the hood of The Sims*", Presentation of The Sims at Northwestern University, http://www.cs.northwestern.edu/~forbus/c95-gd/lectures/The_Sims_Under_the_Hood_files/frame.htm.

AIBO BASED APPLICATIONS

AIBO ROBOT AS A SOCCER AND RESCUE GAME PLAYER

D. Datcu, M. Richert, T. Roberti, W. de Vries, L.J.M. Rothkrantz
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands
DECIS LAB
Delftechpark 24 2628 XH Delft, The Netherlands
E-mail: {D. Datcu, L.J.M. Rothkrantz}@ewi.tudelft.nl

KEYWORDS

Situational AI, Domain knowledge specification and character instruction, Pathfinding.

ABSTRACT

The researches in the field of robotics technologies are set to improve the way intelligent processes accomplish their tasks and interact with the environmental world. An efficient mode for researching and testing the capabilities of the algorithms is putting the robots to play in different coordination games. The current project aims to set up an environment for Sony AIBO robots to play in a soccer game and a rescue game. In both games AIBO is assumed to detect and localize objects such as balls, shouting team players, landmarks and crying victims. The AIBO robot is able to sense the world with different modalities, i.e. sight, hearing and touch. The ultimate challenge of the paper is to fuse data from different communication channels.

INTRODUCTION

The human sensory system perceives processes and fuses multimodal information provided by multiple modalities and enables us to interpret the resulting information with an apparent ease within the current context (time, environment, history). It handles the problems of ambiguity, redundancy and synchronicity in a seamless manner. Multimodal systems represent and manipulate information from different sensor channels at multiple levels of abstraction. Information fusion from sensors plays an important role in combining the actual discovery and the previous knowledge for taking any action.
In playing games, the robotic systems automatically extract meaning from multimodal, raw input data, and conversely produce perceivable information from symbolic abstract level. By setting the robots for playing in games with high difficulty levels, all the involved algorithms and the specific behavior at different interaction levels can be evaluated. During a common game session, the robot's hardware components can be programmed to execute intricate actions. Commonly, the actions are backed by a previous reasoning step that takes into account data from different channels. For the current project, the Sony AIBO robot has been used for the implementation of the research algorithms. Given the wide range of hardware sensors, it represents a suitable choice for the development of a multimodal platform having complex tasks to be accomplished in a game environment.
The fusion can be realized on different levels, at the close to signal data level or at the highest semantic level. Fusion at the basic level has a general character and can be used in many applications. Especially cueing that is the expected sound triggering of visual attention is suited for low-level fusion. Also the fusion of information from spatially distributed sensors is a good candidate for success at the lowest level. Information fusion at the highest level implies context awareness and modeling the patterns of behaviour. In the current project, information is derived from sensors measuring different modalities sound and vision. Our intelligent system must have an awareness of the spatial properties of the environment (locate objects and people, build maps), changes of those properties over time (tracking objects and people) and it must have an awareness of the behavioral aspects (recognize intentions and emotions).

RELATED RESEARCH

The field of multi-sensory data fusion has witnessed a number of advances in the past couple of years, spurred on by advances in signal processing, computational architectures, and hardware. Nevertheless, these advances pertain mainly to the development of different techniques (e.g., Kalman fusion, ANN-based fusion, HMM-based fusion) for multimodal data fusion at a single abstraction level. To date, most multi-sensory data fusion approaches have largely avoided dealing with questions that involve context-awareness and internationality and, in

turn, require the realization of multimodal data fusion at different abstraction levels.
In the past couple of years our research group was already involved in multi-modal fusion projects [M.Pantic, L.J.M.Rothkrantz 2000, 2003]. At the Delft University of Technology has been a project running on the development of a multimodal framework for multimodal data fusion [D.Datcu, L.J.M.Rothkrantz 2004]. [J.C.Martin et al. 1995] describes a basic language for cooperation of modalities in a multimodal application. The event queue mechanisms and the communication between tasks and execution modules are also analyzed. Krahnstoever [N.Krahnstoever 2001] describes a multimodal framework targeted specifically at fusing speech and gesture with output being done on large screen displays. Several applications are described that have been implemented using this framework. The fusion process is not described in great detail, but appears to be optimized for and limited to integration of speech and gesture, using inputs from cameras that track a user's head and hands. The W3C has set up a multimodal framework specifically for the web [J.A.Larson, T.V.Raman 2002]. This does not appear to be something that has actually been implemented by the W3C. Rather, it proposes a set of properties and standards specifically the Extensible Multimodal Annotation Markup Language (EMMA) that a multimodal architecture should adhere to.

AIBO ROBOT

Sony AIBO robot represents a suitable choice for running experiments in the filed of robotics. The processing power of the version ERS-7 of AIBO robot is supported by a 576 MHz 64bit RISC CPU. The built-in Wireless LAN module offers communication capabilities within the network with computers or other robots. All the data processed by the robot are stored on the existent internal memory of 64 MB or on the attached flash memory. The robot is equipped with a color vision camera that is a CMOS Image Sensor capable of capturing images of 350.000 pixels. The sound is handled through incorporated miniature microphones and speaker. The infrared distance sensors provide the robot with data related to obstacles on the path.

GAME CONTEXT AND METHOD

In the case of playing in a football game, the AIBO robot has to perform specific tasks according to the current running context. All the actions are taken according to a finite state machine model. The program contains sets of premises for entering one of the states and rules for specifying the actions. Some of the states may include specific motion actions of the robot's components. The other states are strictly related to reasoning procedures for the given scene.
While playing in a soccer game, every robot has to send information related to its own perception and processing to the other playing robots in its team.
An additional task is to set the AIBO robots to play in the Rescue game. The requirements that exist at the low functional level include the presence of the specific hardware sensors to be connected to each playing robot. Given the data related to the environmental events and the long-time decision previously set, the robot has to take local temporal actions for approaching the target, given a strategy. Each robot has detection rules concerning safety and emergency management activities and possible actions to be taken for each known context. The high-risk activities in the context of the game include those activities associated with accident and emergency management, household, water, swimming, boating, fire, marine, environmental, vehicle use and other activities.

DATA FUSION

The system architecture includes different multimedia data streams in horizontal direction and a hierarchical layered structure in the vertical direction (Figure 1). The parallel data streams are processing paths for sound, video and sensor data. Every stream can operate as an independent unit. The model consists of five layers: filtering and reduction, preprocessor, processor, application and output layer. The first layer includes processings on noise and data reduction on the input raw data from sensors. The preprocessor layer applies smoothing and averaging time processings. The information is further analyzed in the processor layer for deriving hypothesis on the environmental perception of the robot. The application layer tests all hypothesis for consistency and provides the results to the output layer for creating the actual robot's feedback. In both game contexts, the feedback from the robot consists of a set of actions to be executed for achieving the goals given a game strategy.

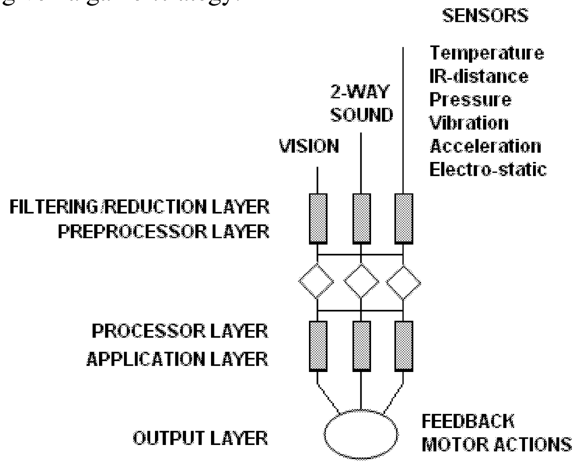


Figure 1. Multimodal fusion

SOUND SOURCE LOCALIZATION

While playing, the AIBO robots try to recover their position in the field by using different techniques. Experimental set-ups for the soccer game can be made to use a ball or other static/moving objects that emit sound at a given frequency. First vision-oriented routines are run to extract useful data of the robot's view. After that step, a second set of procedures based on sound source localization is performed. The goal of the sound localization component is to determine the azimuth of the observed sound source (Figure 2). The procedure consists in finding the horizontal direction expressed as the angular distance between the robot's orientation and the direction of the sound source. The azimuth is determined by using the interaural time delay (ITD). ITD stands for the difference in arrival time of the sound source's signal between the left ear and the right ear. The sound source localization is done only on a number of separate, predefined frequencies so as to be able to discern multiple sound sources from one another and facilitate the distinction between signal and noise.

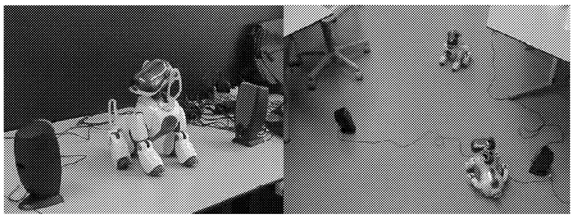


Figure 2. AIBO detecting sound sources

Signal Reliability

A threshold-oriented mechanism is set to prevent sound source localization from being performed on frequencies that contain only background noise. The reliability mechanism includes a Fast Fourier Transform on both the left and right channel of the sampled signal. The reliability threshold *t* is defined as in Formula 1.

$$t = \mu + k \cdot \left(\frac{|f_i^l| + |f_i^r|}{2} - \mu \right)^2 \quad (1)$$

Parameter μ is defined as in Formula 2.

$$\mu = \underset{\forall i}{\text{mean}} \left(|f_i^l| + |f_i^r| \right) \quad (2)$$

The elements in Formula 3 represent the amplitudes of the *i*-th Fourier components of the left and right signal, respectively.

$$|f_i^l| \text{ and } |f_i^r| \quad (3)$$

k is a constant which for has a value of 2, so that the sound from the robot's speaker does not raise the threshold too much, which would cause it to view the signals of other robots as unreliable. The signal corresponding to the *j*-th Fourier component is then considered reliable if and only if the relation in Formula 4 takes places.

$$\left(|f_j^l| > t \right) \vee \left(|f_j^r| > t \right) \quad (4)$$

Performing the FFT on a sampling frame of 16 ms, the *j*-th Fourier component corresponds with a frequency of *j* / 0.016 = *j* · 62.5 Hz.

Calculating the ITD

Calculating the ITD may imply a lot of computation. The fastest way is to perform a FFT on the samples signal and then use that to determine the interaural phase difference. This method is extremely sensitive to reflection and background noise and so a slightly complex algorithm was developed to calculate the ITD. The first stage of processing on the developed algorithm specifies that both, the left and the right signal are run through a filter bank to extract the signals at those frequencies in which there is any interest. The filter bank is filled with band-pass filter of which each central frequency is a multiple of 250 Hz. Every filter in the filter bank is a 128th-order FIR filter with a bandwidth of 62.5 Hz, designed using a 129-point Bartlett window. One beneficial property of this specific type of filter is that it has a magnitude response of –45 dB or less for all frequencies that are multiples of 250 Hz, other than the central frequency (Figure 3).

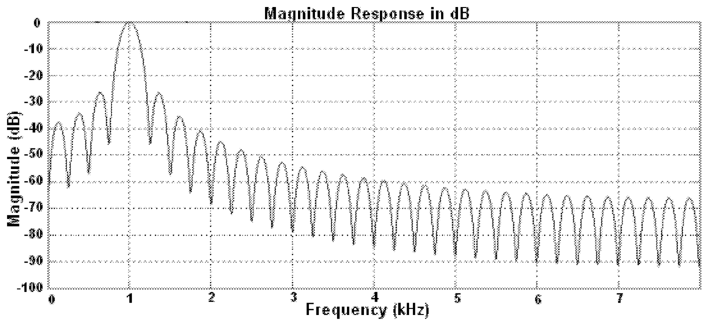


Figure 3. Magnitude Response of Band-pass Filter with 1000 Hz Central Frequency

The next step is to determine which ITD maximizes the cross-correlation between the left and the right signal, for each frequency. In the windowed cross-correlation defined in Formula 5, $l[n]$ and $r[n]$ are the discretely-filtered left and right signal, respectively, and where N_1 and N_2 define a window in time for which the cross-correlation is calculated.

$$\rho_{lr}[d] = \sum_{n=N_1}^{N_2} l[n]r[n-d] \quad (5)$$

The value of the delay d that maximizes $\rho_{lr}[d]$ is the ITD in samples. This is then multiplied by the duration of one sample to acquire the ITD in seconds.

A problem with this cross-correlation method is that for signals of constant amplitude it produces more than one maximum at intervals equal to the signal's frequency. In order to overcome the ambiguity, a triangular modulation was performed since it always produces the greatest difference in amplitude between periods.

Furthermore, when the ITD is larger than one half the period of the signal, the smallest delay which maximizes the cross-correlation is actually smaller than the real ITD. To avoid this issue of wrapping, the signals of frequencies that have wavelengths shorter than half the distance between both ears are not processed.

The Azimuth Model

Determining the azimuth from the observed ITD requires a model that predicts the difference in distance between the left and right ear that the signal

of a sound source on a specific azimuth must travel in order to reach them, respectively.
As testing, the Lord Rayleigh's (John Strutt's) duplex theory has been tested initially. A major problem with Lord Rayleigh's model is that he assumes that the left and right signals arrive at the head in two parallel lines. Unfortunately, this is only the case if the sound source is of the exact same size as the head. In the testing environment, the robot's own speaker has been used (while another robot fulfills the role of observer, Figure 2). The speaker is significantly smaller than the head. Because of this, the paths of the left and right signal are at an angle to each other. This phenomenon becomes even stronger the closer the sound source is positioned to head.

The used model represents the sound source as a point source, since this is the most encountered situation.

$$ITD = \frac{v_s}{r_m}\left((2-\chi)\theta + \chi\sin(\theta)\right) \quad (6)$$

In the approximation given as in Formula 6, v_s equals the speed of sound, r_m equals microphone radius (the distance of either ear to the center of the head), θ is the azimuth, and χ is a function of the distance of the sound source to the head. The parameter χ is always positive and correlates with the distance of the sound source to the head. For computational simplicity in converting the **ITD** to the azimuth, it is assumed that the distance of the sound source to the head is effectively zero, so that χ equals zero (Formula 7).

$$\theta = \frac{r_m}{2v_s} ITD \quad (7)$$

Since the distance of the sound source to the head will in reality never be zero, any θ would seem to lead to a slight overestimation of the **ITD** in this model, and conversely, any **ITD** would lead to an underestimation of the θ. At the current processing stage, the azimuth is confuse by not being clear whether it is really a front angle or merely a mirrored back angle. A deconfusion process is run for exact determination.

Test results on source sound detection

The robot always completely stop the head's movement to safely listen for sound sources (recording azimuths from 10 sampling frames) and only then moves its head to respond. The cycle is repeated, *ad infinitum*. In trying to get a better estimate of the true azimuth of a sound source, a smoothed estimation of the azimuth over time was imposed (Formula 8)

$$\theta_{n+1} = (1-\alpha)\theta_n + \theta \quad (8)$$

The θ is the latest observed, deconfused azimuth, and θ_n is the smoothed azimuth at the *n*-th sampling frame. For the implementation of the algorithm, the value of α was chosen to be equal to 0.1, which gives the last 10 sampling frames an influence of about 65% on the current value of the smoothed azimuth, as can be seen from Figure 4.

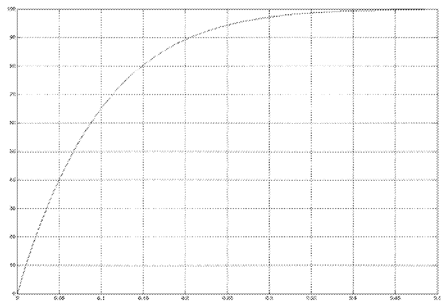


Figure 4. Influence of Last 10 Frames on Smoothed Azimuth

This makes for less fluctuation of azimuth values and also greatly reduces the influence of incorrect azimuths (due to background noise, etc.).

VISION ORIENTATION

In addition to the sound-processing module that is used for determining the location of the objects that emit sound in the scene, the vision module follows a set of commands for extracting the useful graphic information. The tasks consist in recovering the relative position of the ball, the field lines, the goals, the flags, the other players, and the obstacles.
By fusing information from the vision-oriented analysis with results from the sound-oriented analysis, the accuracy of robot perception increases.
For the detection of the ball, the image is searched for candidate positions. The essential criterion for image analysis stands for color and shape-oriented detection. The information concerning the positions and angles of the detected objects are stored relative to the robot.

Ball detection

For balls, upper and lower points on their boundaries are detected during scanning. Points on the border of the image are ignored. During scanning, red pixels below a reasonable number of orange pixels are also treated as orange pixels, because shaded orange often appears as red. Although only a single ball exists in the game, the points are clustered before the actual ball position is detected, because some of them may be outliers for a red robot. If at least three points have been detected, these can be used to calculate the centre of the ball by intersecting the middle perpendiculars. If the three points do not lie on a straight line, the centre of the ball in the image can be calculated, even if the ball is only partially visible. If the ball is not below the horizon or if the camera matrix is not valid because the robot is currently kicking, the distance to the ball is determined from its radius. Otherwise, the distance is determined from the intersection of the ray that starts in the camera and points to the centre of the ball with a plane that is parallel to the field, but on the height of the ball centre. The position on the field is again determined by intersecting the view ray with the field plane in the height of the ball centre. Finally, the position of the ball in field coordinates is projected back into the image, and a disk around this position is sampled for orange pixels.

Flag (field corner) detection

All indications for flags found during scanning the grid are clustered. In each cluster there can actually be indications for different flags, but only if one flag got more indications than the others, it is actually used. The centre of a cluster is used as a starting point for the analysis. It measures the height and the width of a flag. A first approximation of the size of the flag is taken. Two more horizontal lines are scanned in the pink part and if the flag has a yellow or a sky-blue part, two more horizontal lines are also scanned there. To determine the height of the flag, three additional vertical lines are scanned. The leftmost, rightmost, topmost, and lowest points found by these scans determine the size of the flag. Smaller gaps with no color are accepted. This requires the color table to be very accurate for pink, yellow, and sky-blue.

Goal detection

The analysis of the height and the width of a goal are run during the game. The image is scanned for the borders of the goal from the left to the right and from the top bottom. Smaller gaps with unclassified color are accepted. The maximal size in each direction determines the size of the goal.

Robot detection

To determine the indications for other robots, the scan lines are searched for the colors of the tricots of the robots. If a reasonably number of pixels with such a color is found on a scan line, it is distinguished between two cases. If the number of pixels in tricot color found on a scan line is above a certain threshold, it is assumed that the other robot is close. In that case, the upper border of its tricot (ignoring the head) is used to determine the distance to that robot. This is achieved by intersecting the view ray through this pixel with a plane that is parallel to the field, but on the "typical" height of a robot tricot. As the other robot is close, a misjudgment of the "typical" tricot height does not change the result of the distance calculation very much. As a result, the distance to close robots can be determined.

CONCLUSION

The multimodal capabilities present great advantages for robot coordination in playing games. The fusion of information is done at distinct processing levels so as to obtain useful environmental details. A research of a great importance is to combine results from the processing on different type of signal data so as to establish the actions for the next period of time. In the case of AIBO robot there are modules that manage the flow of data from the internal sensors.
The final goal of the project was to make AIBO robots to play soccer game. An additional goal was to program the robots for playing the Rescue game.
In the current paper we tried to research different ways of combining information taken as results of the supported processing modules so as to obtain the best performance for the robot playing in games.
A decisive criterion was to evaluate the individual achievements of the robot and also the performance of the team. We also worked on algorithms for low-level processing of the signal from the sensors.
The results relate to valuable information on common safety and emergency management issues.

REFERENCES

D.Datcu, L.J.M.Rothkrantz: 'A multimodal workbench for automatic surveillance', Euromedia Conference, Hasselt, 2004.

D.L.Hall, J.Llinas: 'Handbook of multisensor data fusion' CRC Press, 2001.

J.A.Larson, T.V.Raman: 'W3C multimodal interaction framework', http://www.w3.org/TR/mmi-framework, W3C Note. December 2002.

J. P. Jansen: 'Looking like humans do. An approach to robust robot vision', T.U.Delft report MMI-2004-11.

M.Pantic and L.J.M.Rothkrantz: 'Towards an affect sensitive Multimodal HCI', Proceedings of the IEEE, Special Issue on Multimodal Human-Computer Interaction (HCI), vol 91, no. 4, 1370-1390, 2003.

M.Pantic, L.J.M.Rothkrantz: 'Automatic analysis of facial expressions: The State of the art', IEEE transactions on Pattern Analysis and Machine Intelligence 22(12): 1424-1445, 2000.

M.Pantic, L.J.M.Rothkrantz: 'Expert system for automatic analysis of facial expression', Image and Vision Computing 18/2000, 881-905.

M. Richert, T. Roberti, W. de Vries: 'Sound source localization using the AIBO ERS-7', T.U.Delft report MMI-2004-10.

N.Krahnstoever, S.Kettebekov, M.Yeasin, and R.Sharma: 'A real-time framework for natural multimodal interaction with large screen displays' In Proc. of Fourth Intl. Conference on Multimodal Interfaces (ICMI 2002), Pittsburgh, PA, USA,October 2002.

AIBO Entertainment Robot
http://www.us.aibo.com/, http://openr.aibo.com

VIDEO BASED INTERFACE OF *AIBO* FOR NATURAL INTERACTIONS

Yoshiaki Akazawa[1], Shigeru Takano[1], Yoshihiro Okada[1,2] and Koichi Niijima[1]

[1]Graduate School of Information Science and Electrical Engineering
Kyushu University
6-1 Kasuga-koen, Kasuga, Fukuoka, 816-8580 JAPAN
{y-aka, takano, okada, niijima}@i.kyushu-u.ac.jp

[2]*Intelligent Cooperation and Control, PRESTO, JST*

KEYWORDS

Entertainment robot, *AIBO*, Motion capture, Motion tracking, Interface

ABSTRACT
This paper proposes a video based interface of *AIBO*, a home entertainment robot produced by *Sony* Corp., for natural interactions with a human being. Recently, many entertainment robots have been developed. A great deal of effort has been made on hardware technology for sensors and actuators of robots. For instance, *AIBO* has many sensors and actuators so that it can take a wide variety of actions. However, *AIBO* cannot still take any attractive interactions to really entertain the human. In this paper, the authors propose a video based interface for natural interactions between *AIBO* and a human being. The authors already have proposed a real-time video based motion capture system [1] and then propose its extended system, a **De**sktop **Mo**tion **Ca**pture system (***DeMoCa***) [2]. Using *DeMoCa*, *AIBO* recognizes the user actions and naturally interacts with.

1. INTRODUTION
One of the main issues in robotics research is the autonomy, that is, the generation of autonomous behaviors in robots. A main goal of entertainment robotics is also the autonomy. For example, there is a four legs autonomous robot "AIBO" produced by *Sony* Corp. [3]. *AIBO* has many input interfaces to sense external stimulus, e.g., a CMOS color camera, touch sensors and microphones. *AIBO* interacts with the user using these interfaces and its various body actions. However, object tracking and shape recognition capability of *AIBO* are very poor. HSV color space is suitable for the color segmentation but *AIBO* outputs video camera images in YUV format. *AIBO* also has a hardware Color Detection Table (CDT) and outputs CDT video images. The object tracking method of *AIBO* is only calculating center positions of the specific color object using CDT images. Practically the CDT needs strict settings so that it is difficult to obtain good results under different lighting conditions. The object shape recognition ability of *AIBO* is limited to recognizing only a few object shapes. Due to these reasons, the tracking functionality of *AIBO* is insufficient to realize many kinds of interactions between *AIBO* and a human being. We already have proposed a real-time video based motion capture system [1] and proposed its extended system, a **De**sktop **Mo**tion **Ca**pture system (***DeMoCa***) [2]. *DeMoCa* uses only one video camera and employs a very simple motion-tracking algorithm based on color and edge distributions to reduce the calculation cost. It is capable of tracking the upper part of human body, e.g., hands, face, etc, and generates their motion data in real time. It also generates the depth value of a focus area, the hand rotation data and the hand shape recognition data.
In this paper, we propose the use of *DeMoCa* as the tracking interface of *AIBO* as shown in Figure 1. We discuss some typical interactions between a real dog and a human being and we describe how we extended *DeMoCa* for enabling *AIBO* to take such interactions. For example, *AIBO* has to extract a hand image of the user and recognize a specific hand shape to take the same action as "PAW" of a real dog.
The remainder of this paper is organized as follows. Section 2 explains *AIBO* specification. Section 3 introduces *DeMoCa*. Section 4 explains interaction examples *AIBO* can perform. Section 5 describes experimental results and performance. Finally, Section 6 concludes the paper.

2. AIBO SPECIFICATION
This section describes the specification of *AIBO*. First of all, we introduce the hardware specification of *AIBO*. Then we briefly explain video images captured by the camera attached to *AIBO*.

2.1 Hardware specification of *AIBO*
The released models of *AIBO* are ERS-100, ERS-200, ERS-300 and ERS-7. Since ERS-7 is the latest model, and has many sensors and a high performance CPU, we use ERS-7 model of *AIBO* and implement its video based interface using *DeMoCa*. Figure 2 shows various sensors attached to *AIBO* (ERS-7) and Table 1 also shows the details of the hardware specification of *AIBO* (ERS-7).

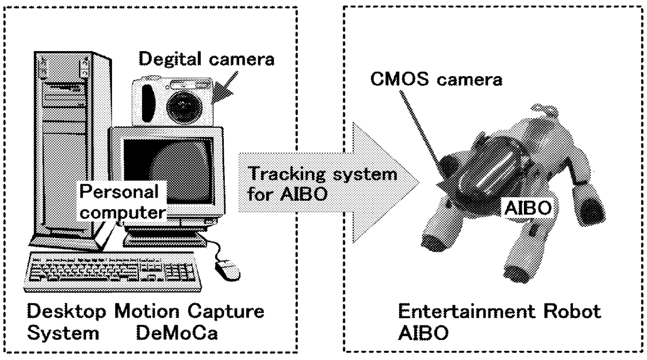


Figure 1: Video based interface of *AIBO* using *DeMoCa*

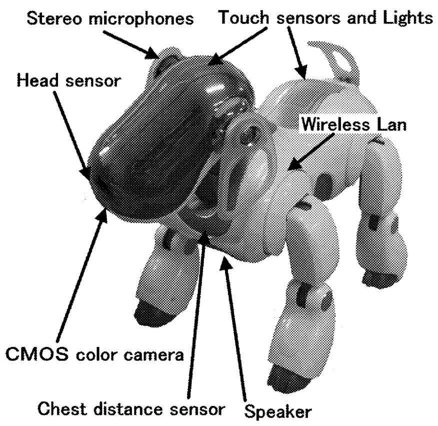


Figure 2: Sensors attached to *AIBO* (ERS-7)

Table 1: specification of ERS-7

CPU	64 bit RISC processor
CPU Clock Speed	576 MHz
Main Memory	64 MB SDRAM
Memory Stick	16 MB
Movable Parts	Mouth - 1 DOF Head - 3 DOF Leg - 3 DOF x 4 legs Ear - 1 DOF x 2 Tail - 2 DOF
Camera	CMOS Image Sensor 350,000 pixels
Audio Input	Miniature Microphones
Audio Output	Miniature Speaker
Built-in Sensors	Temperature Sensor Infrared Distance Sensor Acceleration Sensor Electric Static Sensor Pressure Sensor Vibration Sensor
Wireless LAN Card	IEEE 802.11b

2.2 Video image from *AIBO*

AIBO has a CMOS color camera and outputs video images in YUV format. The component Y is intensity; it is the weighted average of R, G, and B values. The component Y displayed alone creates a grayscale image, used in black-and-white TV sets. The components U and V are chrominance values that can be used with Y to obtain R, G, and B values. As shown in Table 2, the vision stream interface of *AIBO* permits to display YUV images in three resolutions, low, medium and high. *AIBO* has a hardware Color DeTection table (CDT). It means the functionality of hardware-level color segmentation is based on rectangular regions of UV color space. It outputs only medium resolution images. For our video based interface of *AIBO*, we use YUV images of high resolution.

Table 2: Color format

Color format	Color resolution
YUV	208x160(high)
	104x80(medium)
	52x40(low)
CDT	104x80

3. DEMOCA

In this section, we introduce **De**sktop **Mo**tion **Ca**pture system (***DeMoCa***). Originally *DeMoCa* was developed for video-based interactive interfaces of various 3D graphics applications. *DeMoCa* is the extended version of our previous video based motion capture system [2]. In the following subsection, we explain the essential algorithm for motion tracking employed by our previous system [1]. After that, we explain motion data output from *DeMoCa*.

3.1 Motion tracking algorithm of *DeMoCa*

Conventional video based motion capture systems [4] use many video cameras to obtain accurate, desired motion data so they cannot generate motion data in real time because it takes a long time to deal with many video images. Moreover such systems are very expensive and need a large working space [5] so they are not suitable as an input device for a standard PC. To escape these problems, our system uses only one video camera and employs a very simple motion-tracking algorithm based on color and edge distributions. It is capable of tracking the upper part of the body of a person, e.g., hands, face, etc, and generates their motion data in real time. Our system is easily extendable to track the lower part of the body of a person as well as the upper part of the body and to generate more accurate 3D motion data by using two video cameras [1]. In the following, we briefly explain this tracking algorithm.
The motion tracking is performed based on the color information of each specific area of the body. Strictly speaking, the median point of the color information is used as the center of the corresponding focus area as shown in Figure 3. However, practically the color information is insufficient for tracking the motion robustly. For example, the color of skin is uniformly distributed over the arm. So if one wants to track the hand, its color center is influenced by the arm color and it moves to the center of the arm area gradually. Consequently the system will loose the focus area. To compensate this weakness, we employed a new measure involving the edge distribution in addition to the color information. Similar to the color information, the median point of the edges, which are the contour pixels of a focus area, is used to calculate the center of the area. The edge centroid is always located on the upper part of the hand. So the system does not loose the focus area. However, the edge centroid is strongly influenced by the change in the shape of a hand. Therefore, we use weight values for both the color

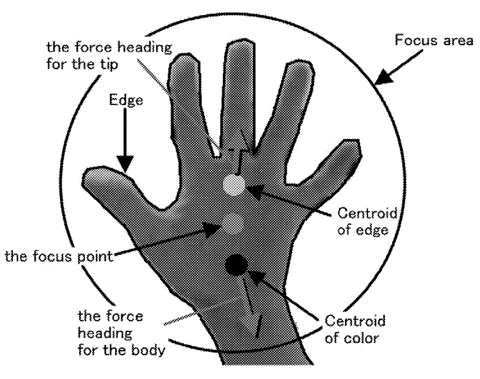


Figure 3: computing a focus point

centroid and the edge centroid. As a result, the focus area becomes stable. The centroid of the focus area is calculated using the following equations.

$$X_p = \frac{w_c X_c + w_e X_e}{w_c + w_e},\ Y_p = \frac{w_c Y_c + w_e Y_e}{w_c + w_e} \quad (1)$$

where X_p and Y_p are the centroid coordinates of the focus area, X_c and Y_c are the centroid coordinates of color distribution, X_e and Y_e are the centroid coordinates of edge distribution, w_e is the weight for the edge distribution and w_c is the weight for the color distribution.

3.2 Motion data

DeMoCa generates x, y location data for each tracking area. This is enough for most applications. Especially when using our motion capture system as a mouse device, this is enough. However, for some cases it is not enough. For example, in a virtual reality application, usually we need 3D position data for manipulating 3D objects. Therefore, we employ another measure concerning the depth.

The depth value is determined by the size of a focus area. The reason is easy to understand because the size of an object far from the camera position is smaller than that of the near one.

For controlling 3D applications, the rotation data is necessary in addition to position data because 3D objects have 6 DOF. *DeMoCa* enables to output the rotation data of the hand besides the x-y position data. First of all, as shown in Figure 4, the system determines the margin area of the focus area to find particular pixels called anchor pixels, which are hand image pixels included in the margin area. Then the hand axis angle can be calculated from the centroid coordinates of the anchor pixels and the centroid coordinates of the focus area using the following equation.

$$Rot = \arctan\left(\frac{Y_c - Y_a}{X_c - X_a}\right) \quad (2)$$

where, Rot is the angle of the hand axis, X_c and Y_c are the centroid coordinates of the focus area, X_a and Y_a are the centroid coordinates of anchor pixels.

Moreover, using this hand axis angle, *DeMoCa* recognizes a couple of hand poses accurately. The system captures the hand image and keeps it as a bitmap as shown in Figure 5 (a). Using the hand axis angle, this bitmap can be normalized through the rotation operation, and then the normalized bitmap like Figure 5 (b) is obtained. Figures 5 (c, d, e) are three bitmaps of typical hand shape images. The system calculates the error between the normalized bitmap of a

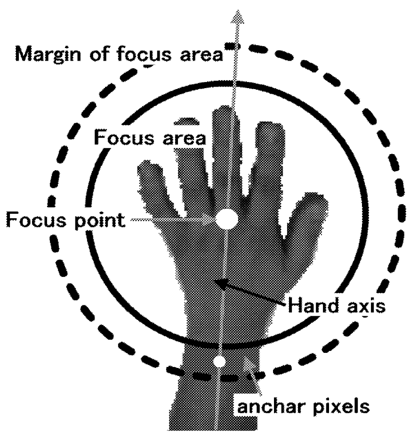


Figure 4: Hand axis from the centroid of anchor pixels to the centroid of a focus

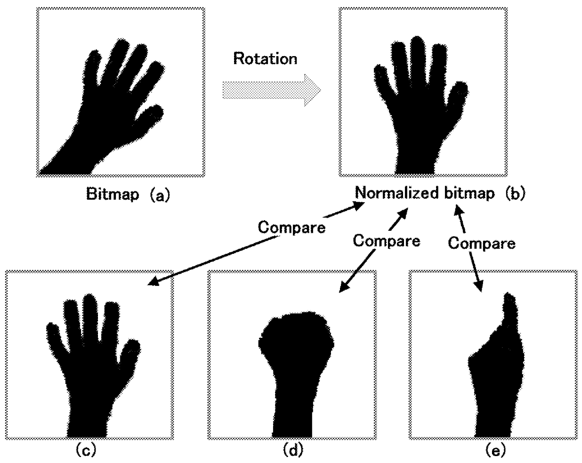


Figure 5: Recognition of three hand shapes

captured hand image and that of a candidate hand shape image. It calculates the error for each candidate hand shape image, and takes as finds the best match the one image that has the minimum error.

4. INTERACTIVE BEHAVIORS OF *AIBO*

In this section, we introduce natural interactions between *AIBO* and a human being. In the following subsection, we explain the role of *DeMoCa* for controlling *AIBO*. After that, we explain some typical interactions between *AIBO* and a human being. In addition, we introduce voice recognition software *Julius* used to recognize the user voice commands.

4.1 Role of *DeMoCa* for controlling *AIBO*

We use *DeMoCa* as a video based interface of *AIBO* for tracking the motions and recognizing the actions of a human being. In the real world, people communicate with a dog using hand gestures and hand signs. So *AIBO* should have a capability to calculate the user hand position and recognize the user hand shape to react like a real dog. To avoid wrong communications, *AIBO* must find whether the user wants to interact with *AIBO*. *AIBO* needs to estimate the distance between *AIBO* itself and the user. For example, to take an action PAW, *AIBO* should track the user hand and recognize it as a paper shape. To communicate with a human being, all the output data of *DeMoCa* except the hand rotation data is required to augment the vision capability of *AIBO* as shown in Figure 6.

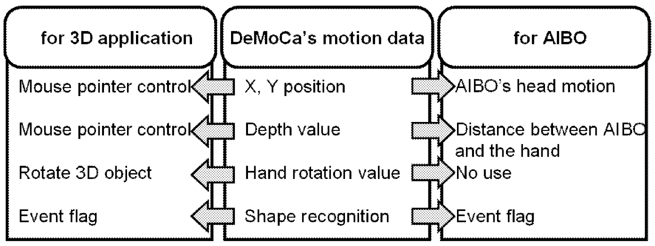


Figure 6: Functions of *DeMoCa*

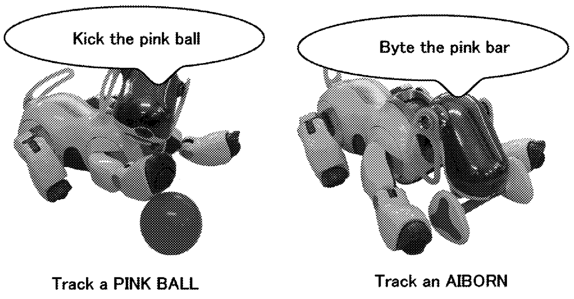


Figure 7: Actions using the standard image

4.2 Interaction examples

AIBO is an entertainment robot and communicates with a human being. However, there are a few interactions based on image processing. For example, as shown in Figure 7, *AIBO* recognizes a pink ball and will kick it. Also it recognizes a pink bar and will bite it. These are not true interactions with a human being. In the real world, a dog can recognize the user commands in the form of voices, body gestures and hand shapes. Through these commands, people communicate with a dog. In the following, we introduce interaction examples of *AIBO*, we already have implemented. These are chosen from interactive behaviors of the dog employed in a campaign dog test in the real world [6].

"PAW"
The user puts his/her right or left hand in front of *AIBO* as shown in Figure 8-A. *AIBO* tracks the user hand and recognizes whether it is the right hand or left one. When the user says "PAW", *AIBO* hears it and begins to paw the user hand. In this case, if the hand the user puts is his/her right hand, *AIBO* paws the user hand using its left forefoot, and vice versa.

"BEG"
The user moves his/her hand in front of *AIBO* as shown in Figure 8-B. *AIBO* recognizes the user hand. When the user says "BEG", *AIBO* hears it and begins to lift its upper body according to the user hand motion.

"TURN"
The user indicates the rotation direction to *AIBO* by his/her index finger as shown in Figure 8-C. *AIBO* recognizes the user hand shape and understands the rotation direction through the user gesture. When the user says "TURN", *AIBO* hears it and begins to rotate along the rotation direction the use indicates.

"BANG"
The user moves his/her hand in front of *AIBO*, and takes an action like "shoot a gun toward *AIBO*" as shown in Figure 8-D. *AIBO* recognizes the user hand shape and action. When the user says "BANG", *AIBO* hears it and lies down.

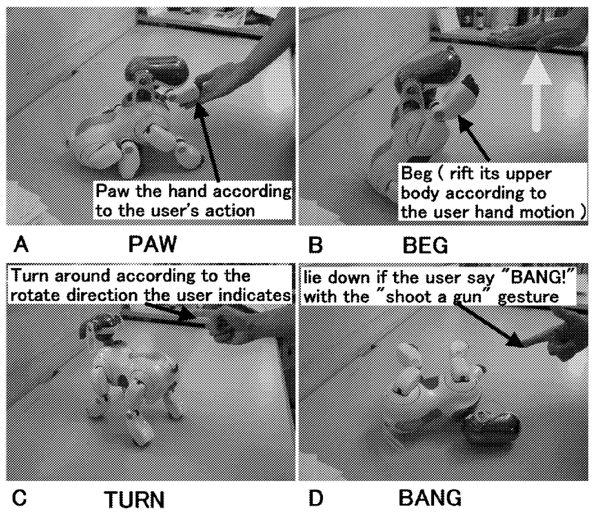

Paw the hand according to the user's action
Beg (rift its upper body according to the user hand motion)
A PAW
B BEG
Turn around according to the rotate direction the user indicates
lie down if the user say "BANG!" with the "shoot a gun" gesture
C TURN
D BANG

Figure 8: Interaction examples

4.3 Voice recognition software *Julius*

If the user wants to communicate with *AIBO* as a pet robot, voice commands are very important. So we employs voice dictation software *Julius* [7] for voice command inputs because *AIBO* Remote Framework [8], a programming toolkit provided by *Sony* Corp., doesn't support the voice recognition functionality of *AIBO*. *Julius* together with its source code is released as the open license software and so it is convenient to develop required functionality for voice command inputs exploiting this software.

5. EXPERIMENTS

5.1 Experimental system

The prototype system was developed using *AIBO* Remote Framework. The hardware configuration for experiments consists of *AIBO*, a standard PC and a standard wireless router as shown in Figure 9. *AIBO* has a wireless LAN card, and *AIBO* outputs video images and audio data to a standard PC via a wireless LAN. Using the video images, *DeMoCa* calculates a hand position and recognizes a hand shape of the user on the PC. *Julius* receives the audio data as the human voice data and outputs dictated text strings. After receiving the motion data and the text data, AIBO Controller determines next *AIBO* action and outputs the corresponding command that will be sent to *AIBO*.
The software architecture consists of five main components, i.e., AIBO server, Virtual AIBO server, AIBO Controller, *DeMoCa* and *Julius*, as shown in Figure 10. AIBO server handles all operation data of *AIBO*, i.e., a rotation angle of each joint, voice data, on/off command for signal lights, etc. AIBO server compresses a YUV video image into a jpeg image to reduce its transmission size/time and transmits it to Virtual AIBO server. AIBO server also transmits audio data in wave file format to AIBO server. It is only that virtual AIBO server that communicates with *AIBO* actually. After receiving the jpeg image from Virtual AIBO server, AIBO Controller expands the file and sends the YUV image to *DeMoCa*. It also outputs the audio data to *Julius*. After that, *DeMoCa* transforms the YUV image into the HSV image.
DeMoCa extracts the human motion from the HSV images by applying the algorithm explained in Sec. 3 in the HSV color space. *Julius* receives the audio data in a wave format

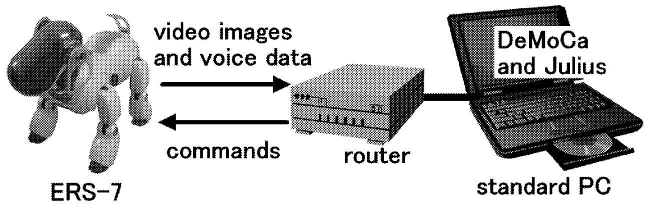


Figure 9: Hardware configuration

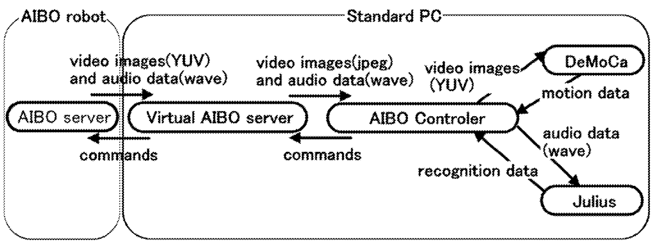


Figure 10: Software architecture

from AIBO Controller. *Julius* dictates from the voice data and returns a word data to AIBO Controller. AIBO Controller compares the word with the user specified command, i.e., "PAW", "BEG", "TURN" and "BANG", and sends the corresponding AIBO command to Virtual AIBO server.

5.2 Cup tracking experiment

Using *AIBO*, we also made a cup tracking experiment. This experiment was an entry of RoboCup Open challenge. In this experiment, we use three blue cups and one yellow ball as shown in Figure 11. At first, the user puts the ball in a cup. *AIBO* understands the cup in which the ball is located. The user touches the touch sensor of *AIBO* to let it start tracking the cup. The user also begins to shuffle the cups. After that, the user touches the touch sensor again to ask *AIBO* stop tracking the cup. Finally *AIBO* indicates the cup, in which the ball is located, by the rotation action of its head.

5.3 Hand gestures and voice commands experiment

We tried to communicate with *AIBO* through interaction examples as shown in Figure 8. *AIBO* understood all of these commands and behaved like a real dog. Combination of video image commands and voice commands avoids wrong interactions. However, to recognize the user voice commands accurately, the user needs a lot of times to make voice dictation software *Julius* learn his/her voice.

5.4 Performance

As for the performance of *DeMoCa*, the sampling rate, when the resolution is 208x160 pixels, is around nine fps on a standard PC (Pentium IV 2.0 GHz, 1.5GB) as shown in Table 3. When *AIBO* outputs a video image to PC, the video image is compressed to the JPEG image so its transmission time is very short.

Table 3: Execution time

Image transmission time from *AIBO* to PC(ms)	51.8
Tracking time on PC (ms)	60.7
Total time (ms)	115.8

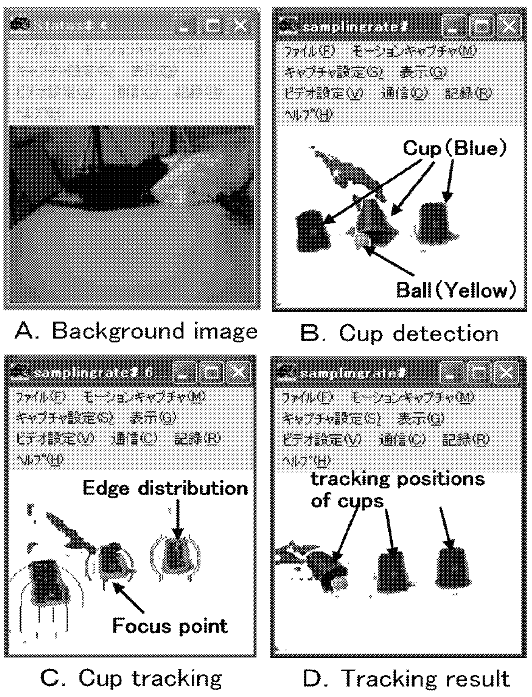


Figure 11: Cup tracking

6. CONCLUDING REMARKS

This paper proposed a video image based interface of *AIBO*, a home entertainment robot produced by *Sony* Corp., for natural interactions with a human being. Using the image recognition with the voice recognition, *AIBO* can understand commands, e.g., "PAW", "TURN" and "BEG", and behaves like a real dog. We used *DeMoCa* as the motion tracking interface to recognize the human motion from video images of the camera of *AIBO*. *DeMoCa* can calculate the user hand position, and recognize the hand shape from the video images. From the several experiments, we clarified the usefulness of *DeMoCa* as the motion tracking interface of *AIBO*.

As a future work, we will develop the voice recognition functionality more suitable for *AIBO* by modifying *Julius*. Furthermore, we will improve *DeMoCa* to recognize the user motion more accurately. And we will also propose more complicated interactions of *AIBO* with a human being.

REFERENCES

[1] Akazawa, Y., Okada, Y. and Niijima, K. 2002. "Real-Time Video Based Motion Capture System Based on Color and Edge Distribution, Proc. of *IEEE* Int. Conf. on Multimedia and Expo, Vol. II, 333-336.

[2] Akazawa, Y., Okada, Y. and Niijima, K. 2002. "Real-Time Video Based Motion Capture System as Intuitive 3D Game Interface", Proc. of Third International Conference on Intelligent Games and Simulation (GAME-ON2002), SCS Publication, pp. 22-28.

[3] http://www.sony.net/Products/aibo/

[4] Gravrila, D. M. 1999. "The Visual Analysis of Human Movement: A Survey." *CVPR*, Vol. 73, 82-98.

[5] Luck, J., Small, D. and Little, C.-Q. 2001. "Real-time Tracking of Articulated Human Models Using a 3D Shape-from-Silhouette Method." Robot Vision 2001, LNCS 1998, 19-26.

[6] http://w-ww.the-kennel-club.org.uk/

[7] http://julius.sourceforge.jp/en/julius.html

[8] http://openr.aibo.com/

GAME ANIMATION AND SIMULATION

INDIVIDUALISED CHARACTER MOTION USING WEIGHTED REAL-TIME INVERSE KINEMATICS

Michael Meredith & Steve Maddock
Department of Computer Science
University of Sheffield
United Kingdom
E-mail: {M.Meredith, S.Maddock}@dcs.shef.ac.uk

KEYWORDS
Character Animation, Character Stylisation, Character Individualisation, Real-time Inverse Kinematics, Weighted IK Chains, Injury Simulation

ABSTRACT

In this paper we present a technique that enhances an inverse kinematics (IK) solver such that when the results are applied to a computer character we can generate a level of individualisation tailored to both the character and the environment, e.g. a walking motion can become 'stiffer' or can be turned into a limping motion. Since the technique is based on an IK solver, we also have the desirable effect of solving retargetting issues when mapping motion data between characters. As the individualisation aspect of our technique is very tightly coupled with the inverse kinematics solver, we can achieve both the individualisation and retargetting of characters in real time.

INTRODUCTION

Computers have long been a tool in the construction of character animation starting with the basic key-framing technique used by artists to the full body motion capture used in many applications today. However, regardless of the format the data is held in, it is general practice to simply play back the animation as a pre-scripted entity in real-time applications. This can lead to many unwanted artefacts such as the visually-incorrect appearance of the character sliding across the ground or character-object penetration.

The term retargeting includes the problem of the character sliding across the ground, which is caused by trying to apply a movement captured on one person to another person of a different size. A solution to this particular problem is to store sets of movements for each individual character, which is essentially an off-line retargeting process, albeit a data intensive one. Instead a more flexible online approach to the general retargeting problem is to use inverse kinematics (see (Watt & Watt 92) for a general introduction to IK) to adapt a single set of motions. The use of inverse kinematics in the field of character animation is not a new idea; however it is only recently being exploited in real-time applications such as games.

In additional to the retargetting properties that come with an IK solver, we have further enhanced the use of our IK technique to incorporate a level of stylisation control over the character that comes at no extra computation cost. We can transform a single base motion to a character of different physiological build, and we can further adapt the motion in real time to simulate the appearance of injuries. Thus a character in a game could receive an injury and change his motion accordingly in real time using our techniques for adapting existing normal motion.

The following section gives a review of current approaches to adapting character motion. Then, after presenting the details of our technique, we demonstrate the principle by applying it to a single walking motion that we adapt to demonstrate both individualisation and injury simulation.

RELATED WORK

There are two general approaches to acquiring base motions for character animations. One way of obtaining the data is to record the motion of a live subject using motion capture technology (mocap) (Vicon 2004), while the other technique requires the motion to be simulated. The latter can itself be further broken down into several different techniques that including keyframing, inverse kinematics (Tolani *et al* 2000, Zhao & Badler 1994) and dynamics (Brogan *et al* 1998) algorithms.

In many cases, it is desirable to adjust motions to meet specific environmental constraints or properties of the computer character. Generally speaking, adaptations are added onto the base motions, or variations of the simulation algorithm are used, rather than creating completely new algorithms to generate such changes. One of the first changes that generally needs to be done is to retarget the motion to a character that may have different dimensions. This is done to eliminate visual artefacts that result from motion mapping and has been successfully tackled in the past with a variety of different techniques include IK (Fedor 2003), spacetime constraints (Gleicher 1998) and dynamics (Hodgins & Pollard 1997). The former techniques tend to be less computational expensive than the latter ones and in particular, the IK algorithm we use in this paper has the ability to perform real-time retargetting in addition to the individualisation we present.

Beyond the task of retargetting characters lies the field of adapting motions to portray more complex stylisation attributes such as physiological build (individualisation) and emotion. In the past, much of the work into introducing different physiological builds into character motions has been based on dynamics and biomechanics (Hodgins *et al* 1995, Komura & Shinagawa 2001). These techniques demonstrate good realism in the results, however this is at the cost of a large computational cost that would not be available in real-time applications which is where our technique demonstrates its potential.

Another area of interest for adding stylisation to base motions is in simulating a level of visible emotion. Various techniques have been used to achieve this goal including the use of Fourier principles (Kraus 2004, Unuma & Takeuchi 1993, Unuma *et al* 1995), energy consumption (Park *et al* 1997) and emotional posturing (Densley & Willis 1997). The latter technique introduces an emotional appearance to the character by constraining joint angles based on the emotional state of the character. Although we do not investigate this in our paper, it would be possible to incorporate this into our work to enhance the individualisation we demonstrate later in this paper, further adding value to our design.

In the following section we describe the technique that allows us to adapt a motion that is retargeted to both the environment and the character and to add extra richness to the motion in the form of individualisation, including injuries. All of this is achieved in real time unlike many of the existing techniques that currently rely on complex dynamics to achieve the same aim.

CHARACTER INDIVIDUALISATION

Our system, *MovingIK SE*, is comprised of three independent modules that communicate accordingly using a level of parameterisation that allows flexible control over the generated motions. The system's modular design is outlined in Figure 1.1.

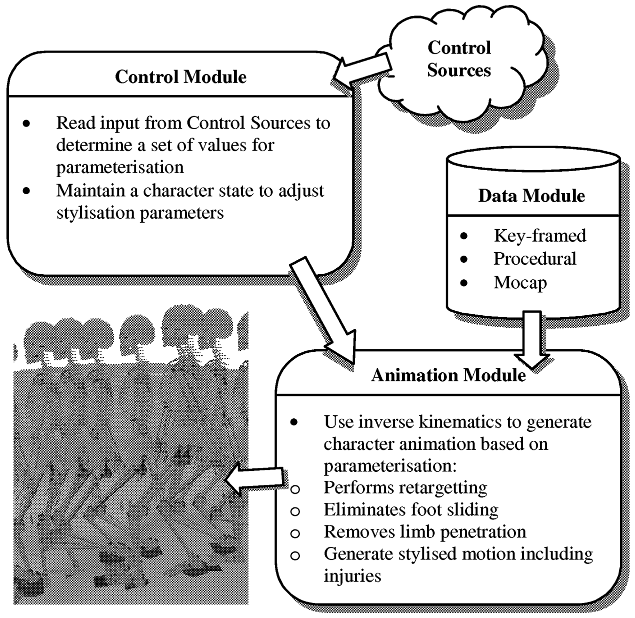

Control Sources
Control Module
Read input from Control Sources to determine a set of values for parameterisation
Maintain a character state to adjust stylisation parameters
Data Module
Key-framed
Procedural
Mocap
Animation Module
Use inverse kinematics to generate character animation based on parameterisation:
Performs retargetting
Eliminates foot sliding
Removes limb penetration
Generate stylised motion including injuries

Figure 1.1: Control Structure of *MovingIK SE*.

The Control Module is the top-level component whose purpose is to generate a set of values for the parameterised motion. The Control Module determines the parameterisation based upon Control Sources that are fed into the module as well as its stylisation state. It is the stylisation state that gives the character its individualisation. The parameterisation of the Control Module, which encapsulates the information required to produce a motion, is passed on to the Animation Module. The Animation Module takes the parameterisation as input and using knowledge about how the motion is performed, which is obtained from the Data Module, it postures the hierarchical structure of the character over time.

For the system we describe in this paper, we will be using the motion of a two-legged humanoid gait. However, with the level of abstraction we have imposed on the system, this can easily be replaced with an alternative type of motion as discussed at the end of the paper. The roles of each of the modules within *MovingIK SE* are discussed in the following subsections.

Control Module

The parameterisation of our system is split into two subgroups. The first subgroup specifies the control parameters of the motion, while the second subgroup influences the behaviour of the inverse kinematics solution used by the Animation Module.

The control parameters of the motion are derived from a user-controlled analogue joystick whose inputs are used to initially determine the stride length, speed, and direction of travel. The Control Module subsequently adjusts these basic motion parameters in order to simulate the character's stylisation state. This, for example, could be to linearly reduce the maximum speed and stride length in order to simulate the fatigue of a character. The way in which the stylisation state affects the parameterisation is discussed later.

The second subgroup of parameters is weighting values that stiffen up joints so they move less compared to surrounding limbs. This gives rise to a basic difference in visual appearance between characters of varying weighted values. These parameters are determined by the stylisation state of the character only. The application of weighted parameters is discussed further under the Animation Module section of this paper.

The stylisation state of the Control Module can be dynamically changed in response to the system's Control Sources. These can take a variety of different forms including responses to environmental events or an AI engine. For our demonstration of producing stylised motions, we invoke the different states by keyboard input.

As well as the basic control parameters and the weighting values, we have control over hip swing parameters for the walking motion we demonstrate. The first of these parameters controls the amount of rotation there is about the vertical axis of the hips. This gives the visual appearance of swaying from side to side, with larger values resulting in the character swaying its hips more. The second hip parameter determines the amount of travel there is along the vertical axis where an increase of this parameter produces a more bouncy looking character.

It is the combination of the control and weight parameters that gives rise to realistic individualisation of the character.

Data Module

The Data Module provides knowledge, in the form of limb Degree-of-Freedom (DOF) values, about how to perform an action to the Animation Module. The output format allows us to model the data using a range of techniques including key-framed data, procedural models and motion-captured data, without affecting the behaviour of the other modules in the system. This allows us to choose the optimal data representation for the given scenario, for example, the use of motion capture data for high detail where storage space is not an issue, compared to procedural models for background characters.

We have based our gait motion data on a mathematical equation which is given in Equation 1.1. A complete cycle ranges from 0 to 3π and the result is the relative height above the ground of the foot. The graphical representation of the procedural stride is illustrated in Figure 1.2.

If $x \le \pi$

$$Y = 1 - \cos(x)$$

else

$$Y = 1 - \cos\left(\frac{\pi + x}{2}\right) \qquad \textbf{(1.1)}$$

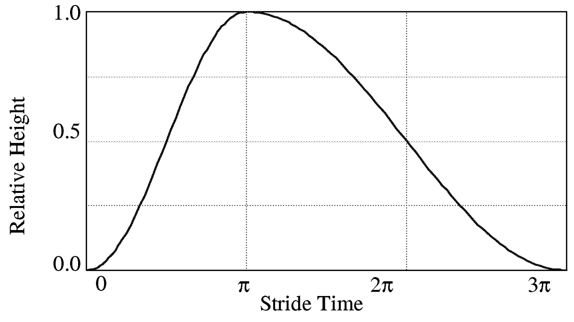


Figure 1.2: Graph of procedural stride of Equation 1.1

Animation Module

Through a combination of information from the control parameters and examination of the surrounding terrain, the Animation Module determines the extents of the motion to perform. Using knowledge about the motion, the end-effector locations are interpolated over time between the extremities and thus produce the retargeted motion. The interpolation follows a modified path whose original is obtained from the Data Module where the end-effectors are positioned along the path through the use of an inverse kinematics solver. This technique allows us to manoeuvre over uneven terrain including climbing and descending steps. The way in which we adjust the base motion for a walking character is briefly outlined in the following subsection.

The use of an inverse kinematics solution to configure the character's structure allows us to precisely position end-effector locations. This has the immediate benefit of eliminating visual problems that are apparent when playing pre-scripted animations. However pre-scripted animations are computational cheap to display therefore to compete we need to keep the resource demand for the IK solution as low as possible. To this end, we utilise a half-Jacobian-based solver (Meredith & Maddock 2004), which allows for a more efficient and quicker solution time for IK chains over the traditional, full Jacobian solver. While the full Jacobian IK solver can operate in real-time, the half-Jacobian has a reduced footprint in terms of processor usage and hence gives a more attractive online animation technique.

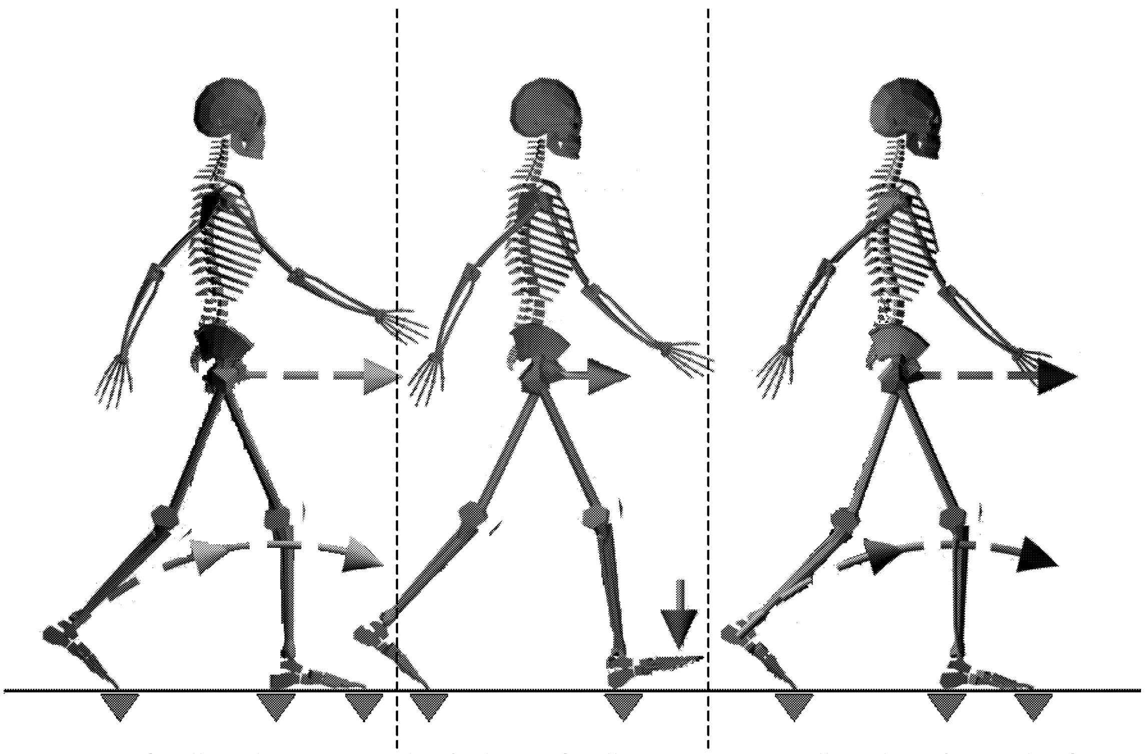
The use of a Jacobian-based inverse kinematics solver has the additional benefit of allowing us to take this algorithm and embed a set of dynamic weighting values. These values are the weighting parameters that the Control Module determines and passes down to this module. The purpose of the weighting values is to give us additional control over the solution that the algorithm produces which visually generates different inter-limb movements for the same end-effector and root nodes positions. In effect, we can make use of the inverse kinematics technique to produce a level of character individualisation at no additional cost to the core algorithm. This technique is further discussed in the *Weighted Inverse Kinematics* subsection.

Changing the Base Motion
There are two cases under which the character can move forward: walking in a straight line or turning. The Control Module uses input from the Control Sources to determine the way the character moves. If there is no sideways movement then the walking forward technique is used otherwise a turning action is executed.

We split a complete walking cycle into two discrete movements. The first is the actual flight of the foot as the character performs a stride, while the second is a post-flight stage that rolls the foot from a heel supporting phase to a

Table 1.1: Illustration of the 2-stage walk cycle where the initial configuration is with the left foot in front and the right foot behind the body.

Stage Description	(a)	(b)
Starting Configuration		
• Left Foot	Both heels and toes are planted on the floor	Toes are planted on the floor
• Right Foot	Toes are planted on the floor	Heel is planted on the floor
Movement	1. Hips move forward, 2. Right heel is advanced forward through the air, 3. Only the left toes remain planted.	1. Hips move forward, 2. Right toes are gravitated towards the floor, 3. Left toes remain planted to the floor



(a) Start of walk cycle (b) Final part of walk (c) Start walk cycle again on other foot

Figure 1.3: Demonstration of the cycles implementing in our system. Each frame represents the start of the cycle with the arrows pointing in the direction of travel the node will take until it reaches the start of the next part of the cycle. The red triangles represent plants of the character's limbs.

complete foot supporting phase. The post-flight phase of the walking cycle increases the realistic-looking nature of the resulting animation and gives us the ability to model the complete foot as opposed to just the heel. An overview of this procedure is outlined in Table 1.1 and Figure 1.3, where a detailed approach of how we control the character is given in (Meredith & Maddock 2004).

Weighted Inverse Kinematics

At the centre of the Animation Module, is an inverse kinematics engine that postures a character using end-effector locations. The algorithm we use is the half-Jacobian-base technique (Meredith & Maddock 2004). Due to the nature of Jacobian-based inverse kinematic solvers, we are able to predictably modify how much different DOFs change when configuring a posture thereby resulting in subtle but individualised results. This, for example, allows us to favour the rate of angle change for the knee over the hip joint.

The Jacobian, J, at the core of the algorithm is determined from the equation of forward kinematics, given by Equation 1.2, where θ represents the set of orientation values for a structure and X is the global position of an end-effector in the hierarchy.

$$X = f(\theta) \qquad \textbf{(1.2)}$$

Taking partial derivatives of Equation 1.2:

$$dX = J(\theta)d\theta \qquad \textbf{(1.3)}$$

where $$J_{ij} = \frac{\partial f_j}{\partial x_i} \qquad \textbf{(1.4)}$$

Rearranging Equation 1.3:

$$d\theta = J^{-1}dX$$

Using these equations, we can describe the complete inverse kinematics solver as follows:

1) *Calculate the difference between the goal position and the actual position of the end-effector:*
$$dX = X_g - X$$
2) *Calculate the Jacobian matrix using the current joint angles: (using Equation 1.4)*
3) *Calculate the pseudo-inverse of the Jacobian:*
$$J^{-1} = J^T(JJ^T)^{-1}$$
4) *Determine the error of the pseudo-inverse*
$$error = \left\|(I - JJ^{-1})dX\right\|$$
5) *If error > e then*
$$dX = dX/2$$
restart at step 4
6) *Calculate the updated values for the joint orientations and use these as the new current values:*
$$\theta = \theta + J^{-1}dX \qquad \textbf{(1.5)}$$
7) *Using forward kinematics determine whether the new joint orientations position the end-effector close enough to the desired absolute location. If the solution is adequate then terminate the algorithm otherwise go back to step 1.*

For our purposes, it is step 6 of the above algorithm that we are interested in which is the stage of the algorithm that updates the joint angles within the hierarchical structure. The algorithm, as illustrated above, will distribute the angle changes needed to meet the desired end-effector location evenly over the chain. However we have rewritten this stage to include a set of dynamic weights that redistribute the contribution each degree of freedom (DOF) has in the

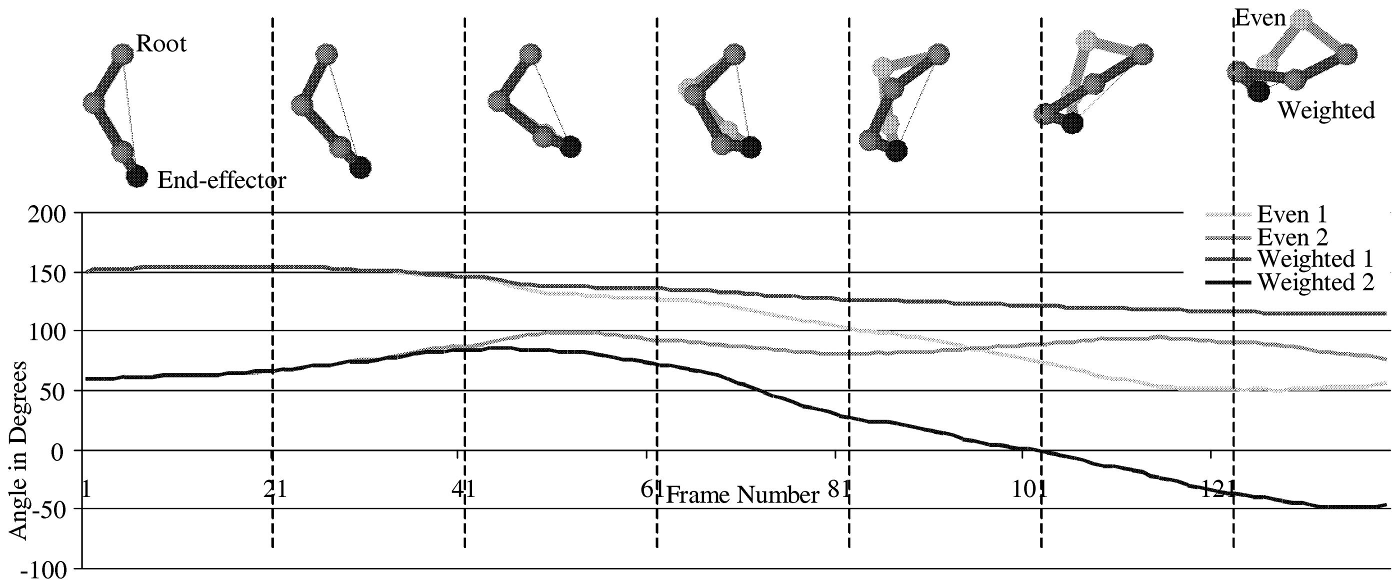


Figure 1.4: Application of weighted IK chains on a simple articulated structure. Top: Comparison of even (lighter colour chain) and weighted (darker colour chain) distribution update of angles – the root node angle has a reduced weighting value over the rest of the chain. Bottom: Graph of the first two angles in the IK chain working from the root node outwards.

resulting motion. We therefore replace Equation 1.5 with Equation 1.6 in our implementation, where *W* is a weighting vector.

$$\theta = \theta + WJ^{-1}dX \qquad \textbf{(1.6)}$$

The weighting vector, *W*, contain real values between 0 and 1 where smaller values result in less significant changes in angle compared to larger values in *W* that correspond to bigger angle changes. This principle is illustrated in Figure 1.4 where the IK solver is applied to a simple hierarchical structure of different weighting values. In Figure 1.4, we have reduced the weighting parameter for the first joint angle, which is at the root, and compared this with an even distribution. As the illustration demonstrates, the joint angle which has a reduced weighting moves less therefore other joints in the chain have to move more to meet the desired end-effector position. This is compared to the evenly-weighted IK chain in which each of the angles involved in the chain are changed relatively equally.

Although we have only applied a weighting change to one of the angles in Figure 1.4, the principle of relatively stiffening up joints within an IK chain equally applies when changing multiple weighting values. However, as it can be seen from the graph of Figure 1.4, reducing a weighting on one joint has the effector of indirectly increasing the weights of the remaining joints because the difference needs to be resolved. This cause and effect result needs to be considered when applying weighting values to an IK chain. We have found through our experiments that as long as these values are specified relative to each other, the results obtained from the solution exhibit the desired properties intended. If the weights are not determined in a relative manner but instead along an absolute scale, the visual results obtained would not necessarily follow that which is expected based on the weighting parameters.

Changing Weight Parameters to Individualise Characters

We harness the property of weighed IK chains in the Animation Module to assist in the production of motions that are individualised to characters based on the stylisation state determined by the Control Module. Although this technique can be applied to a variety of different motions, we will demonstrate how the method lends itself to individualisation where limb masses/muscle tone can be taken into account to determine the weighting values. As an additional effect of individualisation, our technique shows good results when applied to simulating injuries as we demonstrate in the next section.

For the subtle changes involved in individualising a character performing a normal motion, we change the weighting parameters only slightly. This has the effect of simulating limb build within the model. For example, large limbs with low muscle tone would have low weights to simulate a sluggish movement while muscular limbs of the same size would have higher weightings to account for the strength of the muscle.

In order to simulate injuries, as well as adjusting the control parameters, we stiffen up the corresponding limb by decreasing the weighting value associated with it. This reflects the fact that the character changes the movement in the part of their body where a restriction is introduced by the infliction or in order to reduce the resulting pain that would result from using the limb in normal motion.

The results of applying different weighting and control parameters to a single base motion are discussed and illustrated in the next section.

RESULTS & DISCUSSION

To illustrate our technique of applying individualisation characteristics in real-time to characters, we produced a range of different motions by changing only the parameterisation that is determined by the Control Module. The results are obtained by running *MovingIK SE* on a Pentium 4 1.4GHz with a GeForce2 Ultra graphics card. The demonstration animations are achieved in real time running 4 characters simultaneously in response to the same user input.

Figure 1.5(a, b)[1] shows how changing the weighting parameters of a character to produce a slight deviation in the normal walking motion generates an individualised result. To produce the weighted walking motion of Figure 1.5b we decreased the weighting value at the hip joints by 20% and left the rest of the weights the same as used in Figure 1.5a. From the graph in figure 1.5e, the weighted chain knee joint still reaches a maximum and minimum angular measurement as in the evenly distributed walk; however it follows a different path over time to achieve this. The effect of using the weighting reduces the speed with which the hip angle changes therefore the knee angle is changed more to compensate. This can be seen in the middle frames of Figure 1.5a and 1.5b where the character's left knee is further forward in the evenly distributed solution compared to the weighted version for the same point in time.

As the IK chain gets close to being completely extended, as it does at the extents of the walking cycle, the weighted version takes on a similar configuration to that of the evenly distributed one. This is because the possible solutions the IK solver can generate are more tightly packed into a smaller spatial configuration area so the results look similar in either case. Figure 1.5a and 1.5b illustrate this where it can be seen in the first and last set of frames that the configurations are virtually identical.

Despite the solution looking similar at the beginning and end of the cycle, we argue that the differences during the walking phase are enough to demonstrate an individualisation of the character, which is achieved by purely applying weighting vectors to the character. The end configurations could be further diversified if we were to adjust the control parameterisation and skeletal limb lengths. However, in the results we have tried to keep as many parameters constant as possible to demonstrate the potential of weighted IK chains. Furthermore, our Data and Animation Modules are additional contributing factors to the similar looking configurations at the extremities of a cycle because we have specified how a character lands using a two-stage process as shown in Figure 1.3, thereby limiting the possible configuration space.

By changing the weighting parameters alone we are able to generate many individual motions. However, we have found that the most natural-looking motions tend to be linked with low variances in the weight vector used. Larger variances in the weight vector lead to noticeable exaggeration in the resulting motion because joint angles are not updated significantly until there is no choice in order to meet and end-effector location. The motions generated with high-variance weighted vectors could account for normal motion in defined cases however we found it tends to lend itself better to motions that, with a slight adaptation in the other control parameters, simulate the appearance of injuries.

The generation of an injured walking motion is illustrated in Figure 15(c, d), with both the use of weighted and even distribution over the IK chains. In both of these illustrations, the control parameters have been similarly adjusted from the normal walking motion. The control parameters are adjusted for the left stride in order to decrease the maximum flight height by 50%, reduce the stride length by 50% and increase the speed with which the stride is undertaken by 30%.

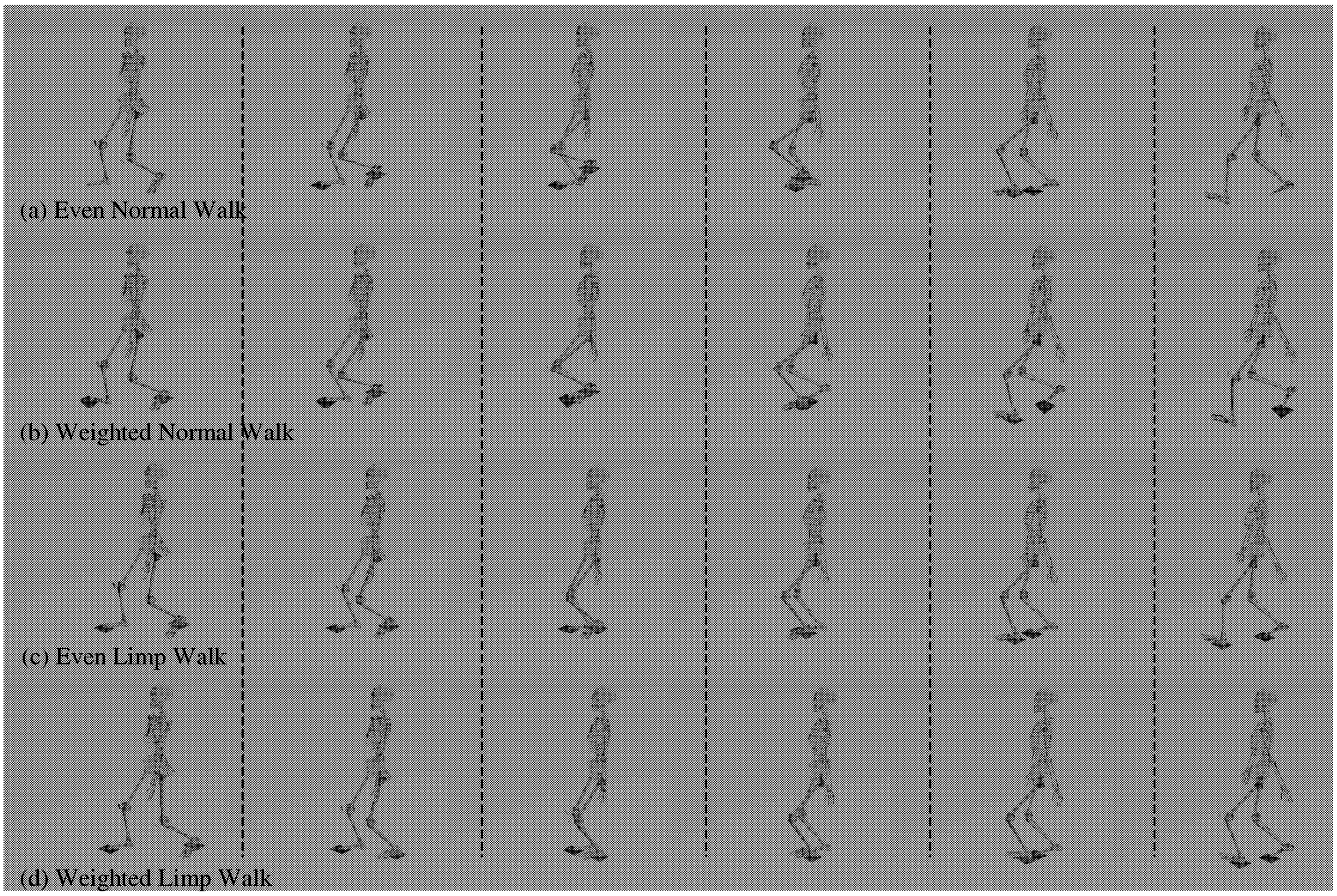
Comparing Figure 1.5a and Figure 1.5c, it is clear that simply adjusting the control parameters is enough to visually change the appearance of the walking motion. This is most visually apparent between the middle frames of the two motions in Figure 1.5. Between these frames, the effect of reducing the flight height of the foot is clear to see, and in the latter frames of the motion it is also visible that the left foot does not travel as far forward as in the normal walking motion. An additional visual effect that is not demonstrated in the stills of Figure 1.5 is that the speed with which the stride cycle is undertaken is faster for the injured leg than for the normal walking motion.

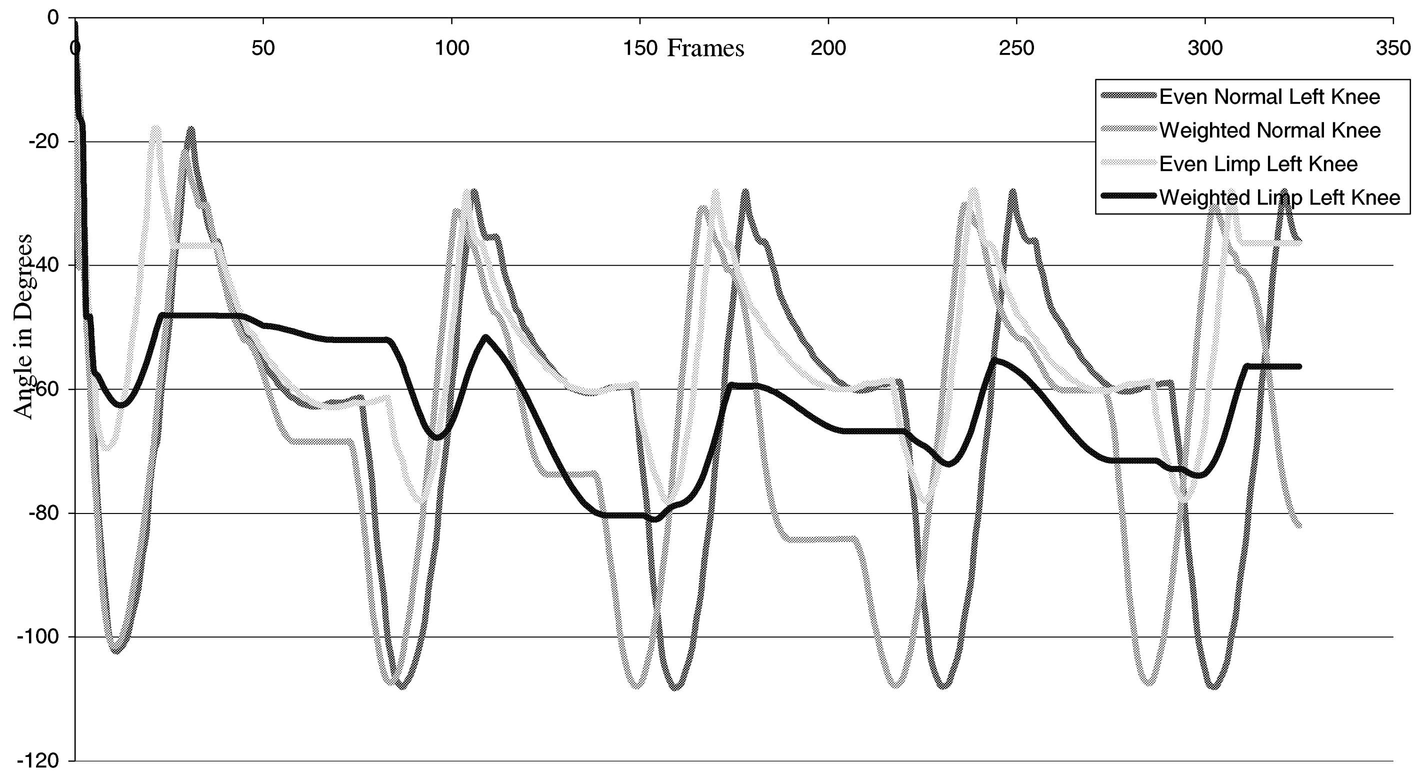
As illustrated, by only adjusting the control parameters of the walking motion we can produce a limping motion. However the realism can be enhanced by additionally adjusting the weighting parameters. This is illustrated in Figure 1.5d. The weight parameters we use to enhance the limp in Figure 1.5d impose a stiffening effect on the left knee, which is achieved by reducing the weighting value for this joint by 90%. As Figure 1.5e illustrates, the effect of applying this weight is to noticeably reduce the amount of movement in the limb compared to the evenly distributed limp motion. This visually translates well into the resulting motion which can be seen when comparing the frames of Figure 1.5c and Figure 1.5d. As these frames show, in order to compensate for the reduced movement in the knee, the hip joint of the weighted motion changes comparatively more than for the evenly distributed one. This is illustrated when comparing the absolute position of the knee joint between the corresponding frames.

The weighting vector used to simulate the motion of Figure 1.5d is tailored to produce an animation that depicts a knee injury. However when we reset the weight values and decrease the weight value associated with the hip joint, we are able to simulate a hip injury. This is achieved by maintaining the control parameters, which like the basic process of individualisation, demonstrates the usefulness of the weighting values in generating subtle differences between base motions thereby customising the resulting motion for a specific stylisation. This makes is possible to determine where the injury is being simulated along the leg.

As the results demonstrate, the additional factor of weighted IK chains provides a good level of differentiation between the changes in joints angles within the character which visually introduces subtle changes for motions that have the same control parameters. This allows us to spawn many motions, each individualised towards a specific character's attributes, at no extra computational cost to the core IK algorithm.

[1] Full animations of the stills of Figure 1.5 are available at http://www.dcs.shef.ac.uk/~mikem/research/ik.html





(e) Graph of left knee joint angle for the walking characters (a)-(d) over 5 strides

Figure 1.5: Demonstration of using weighted chains for individualisation (b) and injury (d) simulation compared to the even distribution of joint changes for the same motions (a) & (c) respectively. Images of (a)-(d) are over a single stride of the left foot.

CONCLUSIONS & FUTURE WORK

The use of real-time inverse kinematics to animate a character has the primary advantage of reducing visual artefacts associated with pre-scripted animations. Through the use of weighted IK chains, we have demonstrated an enhancement to the technique that produces richer visual realism at no extra computational cost. Using weighting values, we have shown that it is possible to take a single representation of a motion and to adjust the motion to different characters, thereby demonstrating a computation ally-cheap mechanism for producing individualised character animations. Taking the technique a step further, we have further shown how the same base representation can be adapted to simulate injury stylisation by adding in discrete visual differences.

From the results, we have found that weighting vectors that have small variance values over their elements produce normal but subtly different motions suitable to the individualisation of character motions. This level of individualisation allows us to control the relative build of the character by assigning comparatively smaller weights to those joints we would expect to change less than others due to muscle structure. This is due to the direct relationship between the amount of angular change performed during the IK algorithm and the weighting values.

Going beyond the basic character individualisation, the use of larger variances within the weighting vectors generates exaggerated motions that we have used to depict an injured motion. The simulation of injuries is enhanced through the use of changes in the control parameters and although this has the effect of fundamentally changing the resulting motion, the weighting values give further control in producing a good looking motion.

We have demonstrated this technique specifically on character gaits. However the idea of adapting the IK result based on weighted chains can apply to any form of posturing using IK. At the heart of this technique, the effect of applying weighting values within the IK chains is to directly affect the rate of change in joint angles. This is a mathematical adjustment on the IK solver itself therefore anything that makes use of the algorithm can utilise this work. Turning to the field of character animation, weighted IK has most potential where computational costs need to be kept minimal but enhanced realism is desirable in the resulting motions.

The next application of this technique is to posture the upper body limbs where it would be reasonable to assume that the same arguments we have presented to demonstrate the application on the legs would also hold for the arms. The application of this algorithm to the upper body would probably produce a much more varied visual result than with the legs because of the increased number of DOFs available to manipulate with the algorithm. An extended area of application that this algorithm would be well suited to is that of performing real-time full body motion retargeting and individualisation.

We acknowledge that this technique will not produce individualisation to a level of realism that dynamics-based techniques can, however we propose that this technique can give a good approximation to the desired results whilst being computational cheaper. Whilst our technique, like any IK solution, is more computationally expensive than directly applying an unadapted, pre-scripted motion to a character, its advantages, namely individualisation and stylisation, are certainly worth appreciating.

REFERENCES

Brogan, D. C., Metoyer, R. A., and Hodgins, J. K., "Dynamically Simulated Characters in Virtual Environments", IEEE Computer Graphics and Applications, Vol. 15, No. 5, p. 58-69, 1998

Densley, D.J., Willis, P.J., "Emotional Posturing: A Method Towards Achieving Emotional Figure Animation", Computer Animation 1997.

Fedor, M., "Application of Inverse Kinematics for Skeleton Manipulation in Real-Time", Computer Graphics and Interactive Techniques, p. 203-212, 2003

Gleicher, M., "Retargetting Motion to New Characters", International Conference on Computer Graphics and Interactive Techniques, p. 33-42, 1998

Hodgins, J. K., Pollard, N. S., "Adapting Simulated Behaviours for New Characters", ACM Siggraph 97, Computer Graphics Proceedings, p. 153-162, 1997

Hodgins, J. K., Wooten, W. L., Brogan, D. C., O'Brien, J. F., "Animating Human Athletics", ACM Siggraph 95, Computer Graphics, p. 71-78, 1995

Komura, T., Shinagawa, Y., "Attaching Physiological Effects to Motion-Captured Data", Graphics Interface 2001, p. 27-36, 2001

Kraus, M., "Human Motion and Emotion Parameterization", Central European Seminar on Computer Graphics, 2004

Meredith, M., Maddock, S., "Real-Time Inverse Kinematics: The Return of the Jacobian", Technical Report No. CS-04-06, Department of Computer Science, The University of Sheffield, 2004

Park, J., Kang, Y. Kim, S., Cho, H., "Expressive Character Animation with Energy Constraints", Compugraphics 97, p. 260-268, 1997

Tolani, D., Goswami, A., Balder, N.I., "Real-Time Inverse Kinematics Techniques for Anthropomorphic Limbs", Graphics Models, Vol. 62, No. 5, p.353-388, 2000

Unuma, M., Takeuchi, R., "Generation of Human Motion with Emotion", Computer Animation 93, p. 77-88, 1993

Unuma, M., Aniyo, K., Takeuchi, R., "Fourier Principles for Emotion-Based Human Figure Animation", Computer Graphics and Interactive Techniques, p. 91-96, 1995.

Vicon Motion Systems Ltd, http:// www.vicon.com, 2004

Watt, A., Watt, M., "Advanced animation and rendering techniques", Addison-Wesley, 1992

Zhao, J., Badler, N. I., "Inverse Kinematics Positioning Using Nonlinear Programming for Highly Articulated Figures", ACM Transactions on Graphics, Vol. 12, No. 4, p. 313-336, 1994

SIMPLIFYING MASSIVE DATASETS WITH COLOR AND TEXTURE IN 3D REAL-TIME GAME ENGINE DEVELOPMENT

Tan Kim Heok[1]
Daut Daman[2]
Abdullah Bade[3]
Mohd Sharizal Sunar[4]
Department of Computer Science
Universiti Teknologi Malaysia
UTM Skudai, 81300, Johor,
Malaysia.
E-mail: tkimheok@hotmail.com[1]
daut@fsksm.utm.my[2]
abade@fsksm.utm.my[3]
shah@fsksm.utm.my[4]

KEYWORDS
3D scalabity, 3D in Game Animation, Applications.

ABSTRACT

Nowadays, the polygonal datasets in many applications are getting larger. These massive datasets has hundreds millions of triangles. This is due to the drastic improvement over scanning technology and increasing complexity in computer simulations. In these datasets, they are not only containing the geometric primitives such as vertex position and vertex normal. Nevertheless, they are also associated with their surface attributes. The surface attributes are important to raise the realism of the object. For instance, the attributes include the color and texture of the triangles. While the performance of graphics hardware also seen a drastic rise in these years, however, displaying massive datasets is still a crucial problem especially in real-time and interactive application such as in game environment. Therefore, automatic simplification on massive datasets while preserving its surface attributes are introduced. In this paper, we present a preliminary approach to simplify the model using modified memory insensitive technique and then organize the data in an octree structure. Conventional vertex clustering technique is used to simplify mesh. At the same time, generalized quadric error metrics is used to preserve the color and texture of the polygonal object. During run-time, the portion of the visible mesh is rendered view-dependently. This approach is fairly simple and fast. In conjunction, the system is demonstrated in game environment.

INTRODUCTION

Computational demanding paradigm like three dimensional interactive applications always require the simulation and display of a virtual environment (VE) at interactive frame rates. Even with the use of powerful graphics workstations, a complex VE can involve a vast amount of computation, inducing a noticeable lag into the system. This lag can severely compromise the display quality.

Therefore, a lot of techniques have been proposed to overcome the delay of the display. It includes motion prediction, fixed update rate, visibility culling, frameless rendering, Galilean antialiasing, level of detail, world subdivision or even employing parallelism (Reddy 1997). Level of detail is certainly a great way in resolving this problem.

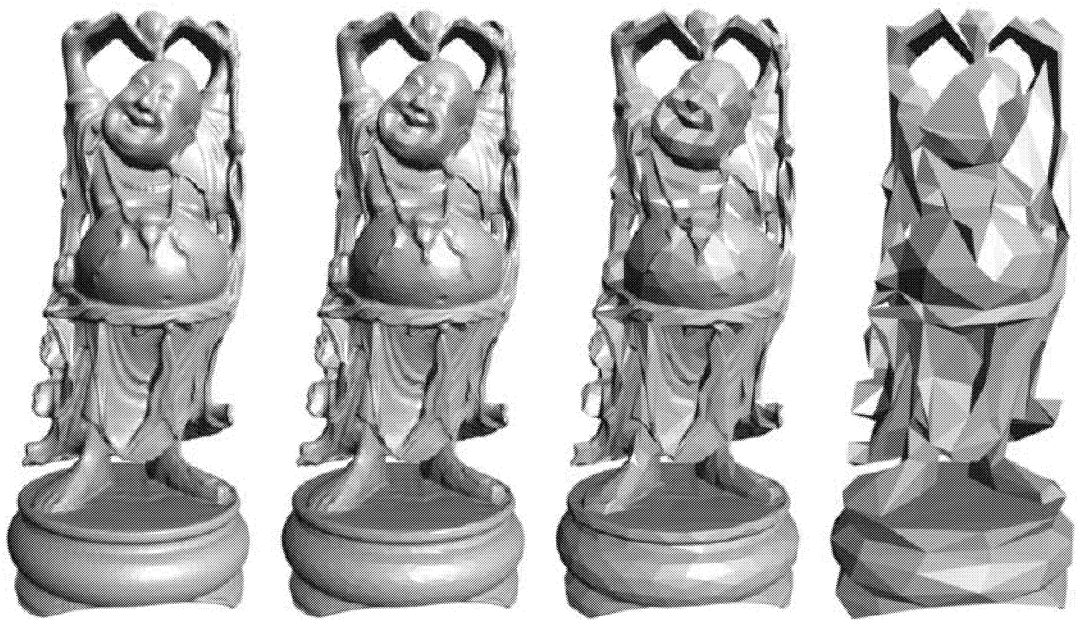
A long time ago, programmers have used Level of Detail (LOD) techniques to improve the performance and quality of their graphics systems. The LOD approach involves retaining a set of representations of each polygonal object, each with different levels of triangle resolution. During the animation rendering stage, objects deemed to be less important are displayed with a low-resolution representation. Where as object of higher importance is displayed with higher details (Figure 1).

Due to the increasing size of datasets, the problem of dealing with the meshes that are apparently larger than the main memory existed. Thus, these data can only be rendered on high end computer system. These massive data are practically impossible to fit in available main memory in desktop personal computer. As a result, it is very cost ineffective and not user friendly.

Suppose this problem can be solved by simplification process, however, the conventional simplification technique need to load the full resolution mesh into main memory in order to perform the task. Consequently, out-of-core approach is invented to overcome this deficiency. Lindstrom (2000) is the first who optimized the secondary memory usage instead of depending on main memory in enormous mesh simplification.

In many computer graphics application, the realism of the virtual environment is very important. Therefore, the preservation on surface attributes is essential during the geometric simplification process. Surface attributes, like normal, curvature, color and texture show the details of the object. Without them, the rendered scene will looked dull and unattractive to user.

In this paper, we present an alternative approach by extending the memory insensitive algorithm (Lindstrom and Silva 2001) to be able to run in real-time according to view-dependent mode. As in memory insensitive technique, the simplification operator is vertex clustering (Rossignac and Borrel 1993) method. The generalized quadric error metrics (Garland and Heckbert 1998) is used to simplify the surfaces in aspect of geometry and color or texture attributes. Then, this data will be input into an octree to construct a multiresolution hierarchy. During runtime, the rendering is generated view-dependently.


Figures 1: Different Level of Detail of Buddha Model

In Section 2, the previous work will be carried out. Next, in Section 3, the framework will be given. Following it, the detail of algorithm will be discussed in flow in different sections, which are memory insensitive algorithm, vertex clustering technique, quadric error metrics, octree construction and view-dependent rendering process. Finally, this work will be summarized and some future works will be covered at the same time.

PREVIOUS WORK

In this section, some previous works in existing simplification operators, error metrics, spatial data structure (tree), level of detail framework and some recent out-of-core approaches. In these methods, attention will draw to vertex clustering simplification method, quadric error metrics, octree structure and view-dependent framework.

The geometric simplifications operators include vertex removal, edge collapse, half-edge collapse, vertex pair contraction (virtual edge), triangle collapse, vertex clustering and face clustering. Vertex removal, which is first introduced by Schroeder *et al.* (1992) performs by remove a vertex and do patching to cover the created hole. Edge collapse (Hoppe 1996) is the highest quality operator that contracts an edge to be a single vertex. Half edge collapse chooses one of the vertex of an edge to be the representative vertex, thus the quality are poorer. Vertex pair contraction enable any pair of vertices to be merged even it is not an edge. In another option, triangle collapse converts 3 vertices into a vertex. Vertex clustering (Rossignac and Borrel 1993) contracts all the vertices in a cell into an optimal vertex. Whereas face clustering merge nearly coplanar faces into large clusters of faces and it is less popular due to its low quality.

From all the simplification operators, vertex clustering is the fastest algorithm. Low and Tan (1997) proposed a slight variation on vertex clustering by using “floating cell” instead of static grid. The most important vertex in a cell is chosen as representative vertex. It offers higher quality result but slower computation time. Luebke and Erikson (1997) use vertex clustering together with octree in their view-dependent refinement.

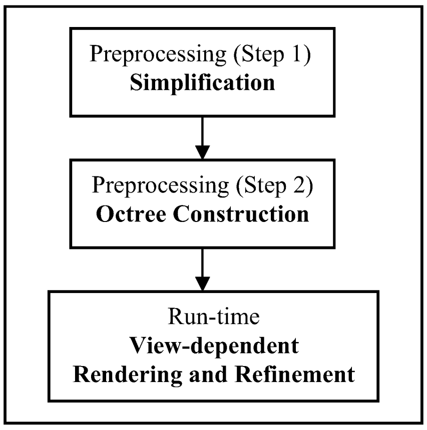
Typically geometric error metric can be done locally or globally. The Hausdorff distance is probably the most well-known metrics for making geometric comparisons between two point sets. This metric is defined in terms of another metric such as the Euclidean distance. Quadric error metrics is based on weighted sums of squared distances (Garland and Heckbert 1997). The distances are measured with respect to a collection of triangle planes associated with each vertex. This technique is fast and good fidelity even for drastic reduction. This work is extended to preserve color and texture attributes (Garland and Heckbert 1998).

Spatial data structures are used to store geometric information. The data or the object can be divided uniformly or adaptively. BSP, quadtree, octree, kd-tree, R-tree are some examples of spatial data structures. BSP divide world using line (2D) or plane (3D). Quadtree is used in 2D environment whilst octree in 3D environment. Shaffer and Garland (2001) use 2 pass in out-of-core approach by first performing uniform clustering like Lindstrom (2000) then adaptive subdivision using BSP to recluster the mesh. OEMM (Cignoni et al. 2002) use octree to partition the mesh

and edge collapse to simplify the mesh. It is slower but higher quality.
There are 4 types level of detail framework, including discrete LOD, continuous LOD, view-dependent LOD and hierarchical LOD. Discrete LOD generate all the static LOD during preprocessing stage but continuous LOD create data structure from which a desired LOD can be extracted during runtime. View-dependent solved problem with large object and can span several resolution over a mesh. However, it's slower. Hierarchical LOD can sit atop of discrete LOD and view-dependent LOD to solve simplification on small object. There are implementations on discrete LOD due to its simplicity like in (Correa 2003). Though, view-dependent LOD and hierarchical LOD are also common nowadays. There's always combination of view-dependent LOD and hierarchical LOD in geometric simplification.
There are some works on for out-of-core simplification for view-dependent refinement. Hoppe (1998) partition terrain in hierarchical format and seams are simplified further. Then, Prince (2000) applied it to arbitrary mesh, but it needs too much of RAM hence slow. Cignoni et al. (2002) extended it using an efficient data structures but it is not reported how this data structure applied to view-dependent refinement. El-Sana and Chiang (2002) also segment the mesh and use edge collapse technique but not using any large model to test their out-of-core technique.
Lindstrom (2000) is the pioneer in out-of-core simplification field. He created a simplification method, called OOCS which is independent of input mesh but and able to simplify extremely large datasets as long as the output size is smaller than the available main memory. Besides, memoryless simplification also introduced. Afterward, enhancement is done on dropping the dependency on output mesh as well in memory insensitive simplification - OOCSx (Lindstrom and Silva 2001). At the same time, many approaches done by previous researchers in simplifying the massive datasets (Borodin et al., 2003; Guthe et al., 2003; Isenburg et al., 2003).

ALGORITHM FRAMEWORK

This paper introduces an approach for end-to-end and out-of-core simplification and view-dependent visualization of large surfaces. Besides, appearance preservation is proposed as well. Here, the arbitrarily large datasets can be visualized by given a sufficient amount of disk space. The work starts from a modified memory insensitive simplification (OOCSx) (Lindstrom and Silva 2001). Then, construction of a multiresolution hierarchy is conducted. Finally, view-dependent rendering on output mesh is carried out during run-time. The framework overview is shown in Figure 2.



Figures 2: Framework Overview

Algorithm starts with a modified memory insensitive simplification (Lindstrom and Silva 2001). This technique eliminates the dependency on main memory size. The input mesh is subdivided into uniform rectilinear grid. Next, vertex clustering simplification (Rossignac and Borrel 1993) is performed on each cell. This simplification operator may introduce non-manifold vertices and edges. Additionally, it may produce less quality output mesh. Nevertheless, it is fast. Hence, it is suitable for our interactive application. During simplification, generalized quadric error metric (Garland and Heckbert 1998) is used to find the optimal representation vertex and calculate the simplify color or texture attribute. These tasks are performed on-disk and therefore must avoid random accesses by using a sequence of external sorts. The output of this stage is a set of triangles and a set of quadric matrices for its vertices.
In building the octree, each triangle is associated with each node in the tree. Each of them represents a grid cell in partitioned mesh and has position and resolution. The triangles file and vertex file are sorted based on the grid cell so that the neighbors are stored together. The generated parent nodes are output sequentially to a temporary file for a certain resolution. It is repeated until all levels in octree are represented. Lastly, we will get a temporary file with single root node. Then, hierarchical structure is created in a single file by linking all the levels from coarsest to most detail level.
In the run-time phrase, the visible nodes are extracted from octree. Each of them is considered as active nodes and the vertex information is loaded into a dynamic data structure. The active nodes are expanded and collapsed based on view-dependent criteria. We make the rendering and refinement stages run in parallel.

SIMPLIFICATION

The main flow of work in OOCSx is similar with vertex clustering (Rossignac and Borrel 1993) simplification process. Just the ways of how to handle data and find the optimal vertex are different.
First, the mesh is subdivided into a uniform rectilinear grid. Because our mesh is in 3 dimensions, therefore, the grid is constrained to have dimension 2^m x 2^m x 2^m. m is how many times we subdivide the space to get the most detail mesh after simplification. The most detail mesh here point to

vertices in the leaf nodes. These leaf nodes will be stored as leaf nodes in (m+1) level in octree later on. The root has cluster ID G(x) = 1. For the kth child of a node's ID is calculated as 8 G(x), + k, where *k*=0, 1, 2, 3, 4, 5, 6, 7.

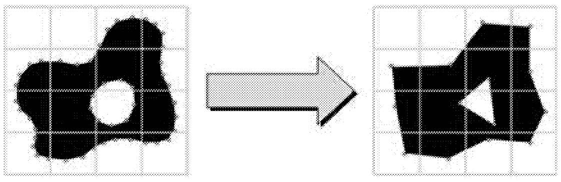
We process the mesh, which is represented as a triangle soup. A triangle soup is a sequence of triplets of vertex coordinates. Steps to perform simplification are similar with memory insensitive algorithm (Lindstrom and Silva 2001) as follows:

- Read one triangle, t = (x^t_1, x^t_2, x^t_3) at one time. Then, compute two orthonormal unit vectors e_1 and e_2 which lie in the plane for t. Determine the grid location G(x^t_i) for each vertex (x^t_i). We generate two files :
  - Plane equation file : store < G(x^t_i), e_1, e_2>for each vertex in each triangle
  - Triangle cluster: store 3 grid location < G(x^t_1), G(x^t_2), G(x^t_3)>(for vertices which fall into 3 different cluster)
- Sort plane equation file using G as primary key using *rsort* (Linderman, 1996).
- Compute quadric q using e_1 and e_2 to get optimal vertices x with its color or texture attribute for each cluster (will be explained in following section). The output is <G(x^t_i), x>.
- Replace cluster IDs in triangle file with corresponding vertices. It is done by creating triangle cluster ID with referring to the file which all the vertices of each triangle.

This stage generated two files for finest resolution in leaf nodes for octree, one is the triangle file. Another one is the cluster ID file associated with its optimal vertex position and its surface attributes.

Vertex Clustering

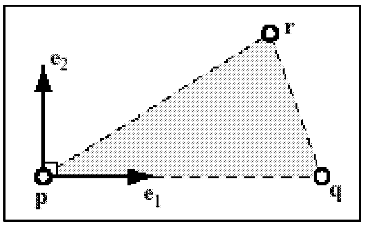
Vertex clustering can also be called spatial clustering as it partitions the space that the surface is embedded into simple convex 3D regions. Because the mesh always represented in Cartesian coordinate system, the easiest way is to do rectilinear space partitioning. Figure 3 show how the vertex clustering work in 2D diagram.



Figures 3: Spatial Clustering Process (Lindstrom 2003)

Generalized Quadric Error Metrics

This generalized quadric (Garland and Heckbert, 1998) created from quadric error metrics (Garland and Heckbert, 1997). This is because original quadric error metrics only handles geometry primitives (vertex position) in simplification. Although it is extended from previous quadric error metric, however, it can't generated by only the normal of the plane for a triangle. Instead, it needs two orthonormal unit vectors e_1 and e_2 to compute quadric error metrics. Let's look at where is the unit vector from in Figure 4.



Figures 4: Orthonormal Vector e1 and e2 define the local frame with origin p for triangle T (Garland and Heckbert 1998)

Consider the triangle T = (p, q, r) and we assume that all properties are linearly interpolated over triangles. If it has color attribute, then p=(p_x, p_y, p_z, p_r, p_g, p_b). If it has texture, then p=(p_x, p_y, p_z, p_s, p_t). To compute e_1 and e_2:

$$e_1 = \frac{q-p}{\|q-p\|} \qquad (1)$$

$$e_2 = \frac{r-p-(e_1 \bullet (r-p))e_1}{\|r-p-(e_1 \bullet (r-p))e_1\|} \qquad (2)$$

Squared distance D^2 of an arbitrary x from T is $D^2=v^TAx +2b^Tx+c$ like in original quadric error metrics where:

$$\begin{aligned} A &= I - e_1 e_1^T - e_1 e_1^T \\ b &= (p \bullet e_1)e_1 + (p \bullet e_2)e_2 - p \\ c &= p \bullet p - (p \bullet e_1)^2 - (p \bullet e_2)^2 \end{aligned} \qquad (3)$$

By solving Ax=b, we get the optimal vertex x with its simplified surface attributes as well. To simplify different type of mesh, refer Table 1 (Garland and Heckbert, 1998).

Table 1: Space Requirement

Model type	Vertex	A	Unique coefficients
Geometry only	$(x\ y\ z)^T$	3x3	10
Geometry+2D texture	$(x\ y\ z\ s\ t)^T$	5x5	21
Geometry+color	$(x\ y\ z\ r\ g\ b)^T$	6x6	28
Geometry+normal	$(x\ y\ z\ a\ b\ c)^T$	6x6	28

OCTREE CONSTRUCTION

In this phrase, we construct a coarse to fine level of detail representation of the mesh in a tree data structure, called octree.

From previous process, we have the simplified mesh in the triangle file. So now, we begin constructing the internal node of the octree. It is done by scanning the triangle file (from previous stage) sequentially. It is needed to generate every levels of octree. As mentioned before, the triangles in level (m+1) is already generated in simplification process. So, we start from level L=m until m=1 (root node). The steps are similar with simplification process with little differences:

- Read one triangle, t = (G(x^t_1), G(x^t_2), G(x^t_3)) at one time. Then, compute two orthonormal unit vectors e_1 and e_2 which lie in the place for t. Determine the grid location G(x^t_i) for each vertex (x^t_i). We generate two files :
  - Plane equation file : store < G(x^t_i), e_1, e_2 >for each vertex in each triangle
  - Triangle cluster: store 3 grid location < G(x^t_1), G(x^t_2), G(x^t_3)>(for vertices which fall into 3 different cluster)
- Sort plane equation file using G as primary key using rsort (Linderman, 1996).
- Compute quadric q using e_1 and e_2 to get optimal vertices x with its color or texture attribute for each cluster (explained in previous section). The output is <G(x^t_i), x, q>.
- Additionally, each child node is recorded its file offset within V_L, and store these with its parents.
- Lastly, we have the triangle file and a file with cluster ID with its optimal vertex for level L.

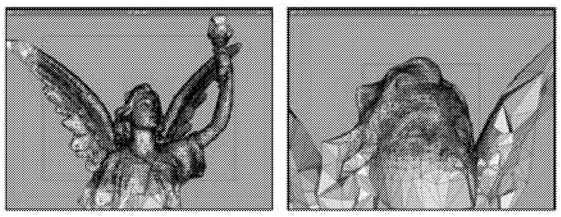
The final step is to link the nodes together in a single hierarchical structure file H using the offsets we save just now. As like other multiresolution methods, we store the multiresolution structure from coarse to fine resolution.

VIEW-DEPENDENT RENDERING AND REFINEMENT

The refinement and rendering are run in two threads. In refinement process, we use Best First search in finding the active nodes. Best first search is a breadth first search with adding heuristics in it. It is because using this search enables the mesh to be updated progressively. Thus, popping effects won't occur. This search also ensures the detail is paged in and added evenly over the visible mesh. These are all benefits from breadth first search. Meanwhile, we control our frame rate using some heuristics.
We begin refinement process by creating the root node into active node. In refinement process, we expand or collapse node. When needs more detail, we add children for the node. Oppositely, collapse when the details are not necessary. However, collapse and expand action only can apply to active nodes which are stored in dynamic data structure in main memory.
During rendering, we test on the boundary of node to the view-frustum planes. If it is visible, expand the nodes. Else, collapse it in coarser form but not totally cull it off. This is to avoid sudden changes in viewing perspective.
If a node is visible, then compare the error threshold based on distance aspect with the node's quadric error. If quadric error is lesser, collapse the node. Else, expand the node.

EXPECTED RESULTS

This proposed framework is expected to handle datasets which is larger than available main memory size on low cost personal computer. At the same time, preserve the surface attributes. The surface attributes here are pointing to color or texture details only. It can generate at least 15 frame rates per second during run-time. Meanwhile, the output is view-dependent. Figure 5 below shows the example of our expected output.



Figures 5: Simplification on Lucy Model. The Portion of the Mesh, which Lie Outside of View Frustum is in Low Resolution

CONCLUSION

We have proposed an outline to simplify the massive datasets, which has millions of polygon and preserve the surface attributes (color and texture) after simplification process. To handling out-of-core datasets, we have modified memory insensitive algorithm (Lindstrom and Silva, 2001) so that can be used in simplifying the mesh in geometry, color and texture aspects. Conventional vertex clustering simplification operator is applied. Even it produced a quite low quality output, but, it is enough for game application. Accuracy is not that vital here like in medical visualization. We have adopted the generalized quadric error metric (Garland and Heckbert, 1998) as the original quadric error metric can't handle surface attributes. This error metric is robust and pretty accurate. To make the data more organized and easier to retrieve during run-time, octree is used. Best first search used here are probable to get a faster solution in tree searching. View-dependent refinement and rendering are run asynchronously.
There's some future works, including performing prefetching to accelerate the data paging and enhance in data handling on-disk. Besides, we can explore in getting a better simplify mesh to make it usable in application which need high accuracy like in medical visualization. We can also extend this application to be run able in network game.

REFERENCES

Borodin, P., Guthe, M. and Klein, R. 2003. Out-of-Core Simplification with Guaranteed Error Tolerance. In *Vision, Modeling and Visualization 2003*, Munich, Germany, 309-316.

Cignoni, P., Montani, C., Rocchini, C. and Scopigno, R. 2002. External Memory Management and Simplification of Huge Meshes. *Visualization and Computer Graphics, IEEE Transactions*, 9(4), 525-537.

Correa, W.T. 2003. *New Techniques for Out-of-Core Visualization of Large Datasets*. Ph.D Thesis, Princeton University.

Garland, M. and Heckbert, P. S. 1997. Surface Simplification Using Quadric Error Metrics. In *Proceedings of SIGGRAPH 97*, ACM Press. Los Angeles, California, Whitted, T. ed., 209-216.

Garland, M. and Heckbert, P. S. 1998. Simplifying Surfaces with Color and Texture Using Quadric Error Metrics. In IEEE Visualization '98, Ebert, D., Hagen, H. and Rushmeier, H. eds., 263-270.

Garland, M. 1999. *Quadric–Based Polygonal Surface Simplification*. Ph.D. Thesis, Carnegie Mellon University.

Guthe, M., Borodin, P. and Klein, R. 2003. Efficient View-Dependent Out-of-Core Visualization. In *Proceeding of The 4th International Conference on Virtual Reality and Its Application in Industry (VRAI'2003)*.

Hoppe, H. 1996. Progressive Mesh. In *Proceeding of SIGGRAPH 96. Computer Graphics Proceedings, Annual Conference Series*, New Orleans, Louisiana, Rushmeier, H. ed., 99-108.

Hoppe, H. 1998. Smooth View-Dependent Level-of-Detail Control and Its Application to Terrain Rendering. In *IEEE Visualization '98*, IEEE, Research Triangle Park, North Carolina, D. Ebert, H. Hagen, and H. Rushmeier, Eds., 35–42.

Isenburg, M., Lindstrom, P., Gumhold, S. and Snoeyink, J. 2003. Large Mesh Simplification using Processing Sequences. *Proceedings of Visualization 2003*, IEEE, Seattle, Washington, 465-472.

Linderman, J., (1996). *rsort* man page, Apr.

Lindstrom, P. and Silva, C. 2001. A Memory Insensitive Technique for Large Model Simplification. In IEEE Visualization 2001, San Diego, CA, 121-126.

Lindstrom, P. 2000. *Model Simplification using Image and Geometry-Based Metrics*. Ph.D Thesis, Georgia Institute of Technology.

Lindstrom, P. 2003. *Out-of-Core Surface Simplification.* California: University of California, Davis*, lectures on "Multiresolution Methods" February 2003.*

Low, K. L. and Tan T. S. 1997. Model Simplification using vertex-clustering. In *1997 ACM Symposium on Interactive 3D Graphics*, ACM SIGGRAPH, Phode Island, Cohen, M. and Zeltzer, D. eds., 75-82.

Luebke, D. and Erikson, C. 1997. View-dependent Simplification of Arbitrary Polygonal Environments. In *Proceedings of SIGGRAPH 97*, ACM Press, Los Angeles, California, T. Whitted, Ed., Computer Graphics Proceedings, Annual Conference Series, 199–208.

Prince, C. 2000. *Progressive Meshes for Large Models of Arbitrary Topology*. Master's Thesis, University of Washington.

Reddy, M. 1997. *Perceptually Modulated Level of Detail for Virtual Environment.* Ph.D Thesis, University of Edinburgh.

Rossignac, J. and Borrel, P. 1993. Multi-resolution 3d Approximations for Rendering Complex Scenes. In *Modeling in Computer Graphics*, Springer-Verlag, Falciendo, B. and Kunii, T. L. eds., 455-465.

Schroeder, W. J., Zarge, J. A. And Lorensen W. E. 1992. Decimation of Triangle Meshes. In *Computer Graphics (Proceeding of SIGGRAPH 92)*, Chicago, Illinois, Catmull, E. E. ed., 65-70.

Shaffer, E. and Garland M. 2001. Efficient Simplification of Massive Meshes. In *12th IEEE Visualization 2001 Conference (VIS 2001)*, San Diego, CA, 127-134.

Adaptive Real-time Control Flight Simulation Using Neural Networks

H. Chen, T.R. Wan and R.A. Earnshaw
School of Informatics
University of Bradford,
Bradford, UK, BD7 1DP
h.chen3@bradford.ac.uk, t.wan@bradford.ac.uk, r.a.earnshaw@bradford.ac.uk

Abstract

Classic autopilot controller design relies on accurate aerodynamic data and needs extensive calculations. In this report, we propose an approach that simplifies the flight controller design process. This proposed architecture contains two parts, which are a dynamic inversion (DI) controller that offers the solution for the approximately linear model and neural network compensators that adaptively cancel the nonlinear inversion errors and un-modelled dynamic errors through offline and online training separately. The algorithm reported in this paper has been applied in a simulation to control a full nonlinear six-degree-of-freedom (DOF) flight model in a 3D virtual environment developed in our work. The simulation results show the usefulness of the neural network based dynamic adaptive control scheme for flight simulations.

Introduction

Today's game industry is flourishing than ever before. The new generation games are expected to offer the user the feeling of highly competitive and the sense of being there. Multiple technologies have been used in the game application, such as artificial intelligence, graphics, etc. Overall, the purpose is to improve the fidelity of the game. Using the first character flight piloting game as an example, the player will not be satisfy to control the aircraft only with mouse or keyboard, instead, force-feed back handle is much popular in this kind of games, which is directly linked to the aircraft control surface. Therefore, complex aircraft models and control technologies have been involved. Microsoft flight simulator 2004 is a good example, which used real parameters of aircraft to interpret the control and motion behaviour of the flights. In this paper, we propose an approach for autopilot controller design. Compared to classic method, this approach does not rely on accurate aerodynamic data and keeps a balance between the simulation fidelity and the simulation speed; therefore, it is supposed to be a good choice for flight simulations in game environments.

The aircraft itself is a multiple-input-multiple-output (MIMO) nonlinear system. The classic design of feedback control systems is based on the use of linear time invariant (LTI) models. In this case, where the parameters of the model change with time, adaptive controls were developed over years to address such situations. The basic idea behind adaptive control is to have a controller which tunes itself to the system being controlled. Traditional adaptive controllers have been proven useful only in the case the time–variations have to be slow. When the system is highly nonlinear, the closed loop system can exhibit undesirable behaviour [1].

Fortunately, the great potential of neural network in control area has been realized. Recent results show that the neural network technique seems to be a very effective tool to control a wide class of complex nonlinear system. Whenever the environment does not allow or does not justify the current controller, the neural network compensators adjust the internal weights to learn a new control strategy. Neural networks offer the advantage of performance improvement through learning by means of parallel and distributed processing and show the great ability while mapping nonlinearity functions. We will give a brief discussion of some of the best known nonlinear control methods that are based on the neural networks. The methods presented here are by no means exhaustive, but they represent part of the state of the art of the methodology in this field.

The most often used neural network controller is utilized as an inverse system model. The inverse model is cascaded with the controlled system such that the neural network produces an identity mapping between the desired response and the controlled system's input [2] [3]. This model needs numerous training data, and the training procedure is tedious. The size of the neural network is considerably large. A hybrid control model was proposed to avoid training the inverse model directly [4]. A fixed gain control box is mating in parallel with the neural network controller. The fixed gain control box is chosen to stabilize the system; the neural network controller is tuned online to improve the performance. Such a strategy only shows limited contribution on overcoming the nonlinearity of the system due to the fact that the states of the system have been ignored while training the neural network. A multiple model based flight control approach was reported in [5]. The proposed method uses the multiple models of the non-linear system for identifications. Each model was a linear approximation in a certain regime. A controller was trained to select the proper model. The drawback of this method is the database of the individual model will be considerable large, when the situation is complex, especially when the system is highly nonlinear. A more detailed and comprehensive survey on neural network control applications may be found in [6].

In this paper, we propose a neural network based adaptive control strategy for flight simulations. This model combines a dynamic inversion (DI) controller and neural network compensators, which is expected to overcome the above drawbacks. This paper is organized as following: The flight simulation methodology is described in the next section, followed by the discussion of the architecture of the adaptive controller and a detailed description of the neural network compensator proposed. Then, the results of a series of numerical simulations are presented. Finally, we will draw the conclusions.

Flight simulation methodology

There are several issues to be addressed when designing a flight simulator. Our target is to offer a platform within 3D virtual environments for flight motion behaviour simulation. A general structure of this system is shown in Figure 1.

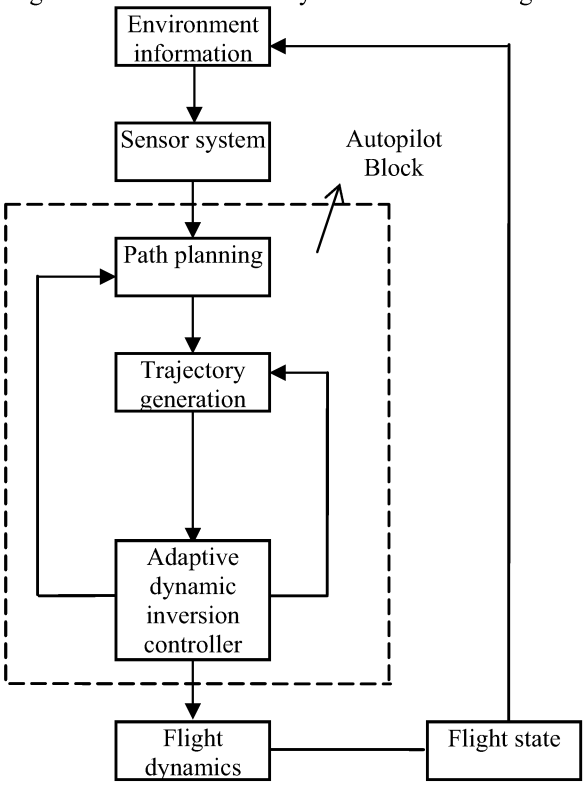


Figure.1. the general structure of the flight simulation system.

The environment information (a terrain for example) is perceived by an attached sensor system. Using the information received, the autopilot unit produces the optimal path solution to the current situation, and then designs a smooth trajectory based on the planned path. The motion control commands are generated according to the trajectory and sent to the adaptive dynamic inversion controller, which adjust the flight control surface and drives the aircraft to fulfil the specific motion. The virtual flight updates its position and continues to perceive the new environment information. Path planning and trajectory generation units receive the feedback from the adaptive dynamic inversion controller, because the aircraft dynamic states have a strong effect on the designed path and trajectory. In this paper we will address the nonlinear flight control problem, which covers the dynamic inversion and the adaptive controller.

A six-degree-of-freedom nonlinear aircraft model has been used in our experiment, which is expected to offer enough complexities to the general flight simulations. A neural network based adaptive dynamic inversion controller is proposed to undertake the duty of motion control. This model combines a dynamic inversion (DI) controller and neural network compensators. The dynamic inversion controller generates the approximate linear model and affords acceptable control inputs, while the neural network is trained to cancel the nonlinear error, and to improve the overall performance. The training process itself can be divided into two stages: one is the offline training to cancel the linear approximation error; the other is the online training to cancel the un-modelled dynamic error. The first stage of the training can be conducted iteratively to guarantee the linear error converges to an acceptable level according to the reasonable system response interval and reduce the calculation intensity, which is very important in both practice application and simulation project. The second stage of the training improves the ability of the system to handle the un-modelled dynamics and un-expectable factors. Instead of compensating the control surface directly; we use this neural network to estimate and compensate the control signal, therefore, it can efficiently avoid undesirable behaviour, such as large transients or a large control signal [1]. The approach we proposed does not strongly rely on the accurate aerodynamic model and the extensive training data. It is especially useful to keep the balance between the simulation fidelity and the simulation speed.

Nonlinear adaptive controller architecture

The design of nonlinear controller for aircraft has been an active area of research for many years. In the 1980's, dynamic inversion, or feedback linearization has been proposed as a practice and viable method for flight control [7]. However, there are two major drawbacks of this approach: one is that it relies on the accuracy of the aerodynamic models, and suffers from intolerance to un-modelled dynamics and noise; the other is that extensive calculation is needed to improve the accuracy of control, which is especially critical when the system is highly nonlinear. Adaptive control is an approach to deal with system uncertainty and nonlinearity. The basic idea is to have a controller which tunes itself according to the system's state being controlled. Therefore, adaptive control is a useful supplement for dynamic inversion control.

We propose a nonlinear adaptive dynamic inversion controller for autopilot design. This model combines a dynamic inversion (DI) controller and neural network adaptive compensators. A general architecture of this controller is shown in Figure 2. A normal PID controller generates a control signal use the error between command and actual output value; the adaptive controller generates the adaptive control input according to the system states, which is expected to compensate the uncertainty and nonlinearity of the system. These two inputs combine together to give the final control signal. The dynamic inversion unit gives the solution according to the control input.

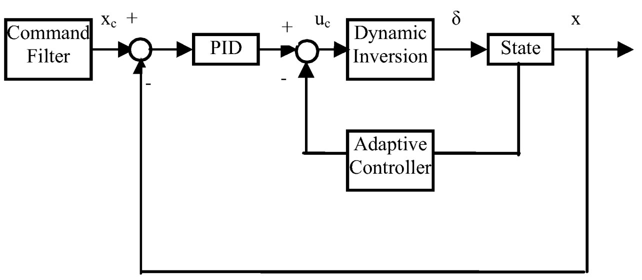


Figure.2. the general architecture of nonlinear adaptive controller

In order to introduce some necessary notation and provide clear insight into the nature of the adaptive control law, the mathematical formulation of the adaptive dynamic inversion scheme is represented as following. Given an n-dimensional nonlinear dynamic system of the form

$$\{\dot{x}_1,...\dot{x}_n\} = f(\{x_1,...x_n,\delta_1,...\delta_m) \qquad (1)$$

where $\{\dot{x}_1(t),...\dot{x}_n(t)\}^T \{x_1(t),...x_n(t)\}^T \in R^n$ are the state variables, $\{\delta_1(t),...\delta_m(t)\}^T \in R^m$ are the control variables and f is a mapping from domain $R^n \times R^m$ into R^n . $\hat{f}(\{x_1,...x_n,\delta_1,...\delta_m)$ is a linear approximation of $f(x_1,...x_n,\delta_1,...\delta_m)$ in certain regimes. Provided that the linear approximation mapping is invertible, the inverse transformation is expressed by

$$\{\hat{\delta}_1,...\hat{\delta}_m\} = \hat{f}^{-1}(x_1,...x_n,\dot{x}_1,...\dot{x}_n) \qquad (2)$$

If the mapping is perfectly known and the system is exactly linear, the system control input δ is the same with $\hat{\delta}$, otherwise, the system which results from applying dynamic inversion can be expressed as follows:

$$\{\dot{x}_1,...\dot{x}_n\} = \hat{f} + \Delta(x_1,...x_n,\dot{x}_1,...\dot{x}_n) \qquad (3)$$

$$\Delta(x_1,..x_n,\dot{x}_1,...\dot{x}_n) = \qquad (4)$$

$$f(x_1,..x_n,\hat{\delta}_1,...\hat{\delta}_m) - \hat{f}(x_1,..x_n,\hat{\delta}_1,...\hat{\delta}_m)$$

and $\Delta : R^n \times R^n \to R^n$ is the nonlinear inversion error. It is also the adaptive factor to the system. Denote X as $\{x_1,...x_n\}^T$. A more accurate control input is given by

$$\{\hat{\delta}_1,...\hat{\delta}_m\} = \hat{f}^{-1}(X, \dot{X} + \Delta) \qquad (5)$$

A simple iterative scheme is employed to compute the adaptive factor until the dynamic inversion error converges to an acceptable or reasonable region. Although the inversion error can not be eliminated or be infinitely small while the time interval is fixed, this iterative adaptive scheme still can greatly improve the control performance and held the inversion error at the lowest level.

Neural network compensator

An adaptive factor can be added to compensate the error generated by the linear approximation. Recent results show that neural network technique has great potentials in the field of controlling a wide class of complex nonlinear system. In our approach, the multi-layer back-propagation neural networks have been trained to work as adaptive compensators.

The training of the neural network can be divided into two stages: the first stage is the offline training to cancel the linear inversion error; the second stage is the online training to cancel the un-modelled dynamics and un-expectable factors. The first stage training guarantees the linear inversion error to converge to an acceptable level and improves the efficiency and performance of the second stage online training. The second stage training improves the ability of the system to handle the un-modelled dynamics and un-expectable factors, which is especially important in the practice applications. It will adjust the weights of the network whenever the environment does not allow or does not justify the current controller. We assume that the adaptive factor has the form of Eq. (4). The system states and control inputs are the inputs to the neural network and the adaptive factor is the target output. Back-propagation algorithm is used to train the neural network to achieve the desired performance. The structure of the neural network is shown below in Figure 3.

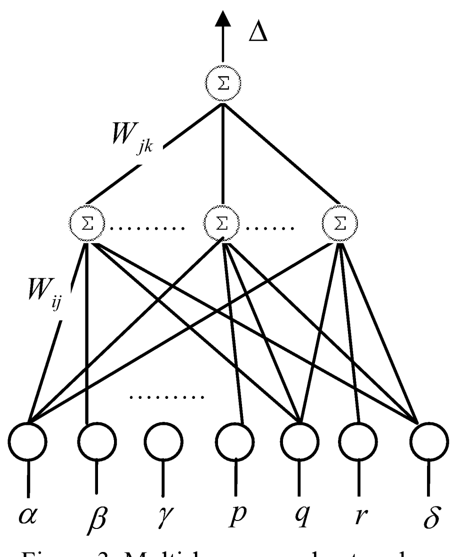


Figure.3. Multi-layer neural network compensator

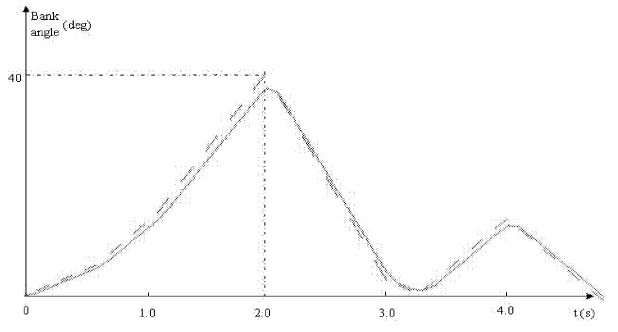
$$u_{ad} = \Delta = \sum_{j=1}^{n} w_{jk}(\sum_{i=1}^{l} w_{ij}x_i) \qquad (6)$$

$$x = \{\alpha \ \beta \ \gamma \ p \ q \ r \ \delta_a \ \delta_e \ \delta_r\}^T \qquad (7)$$

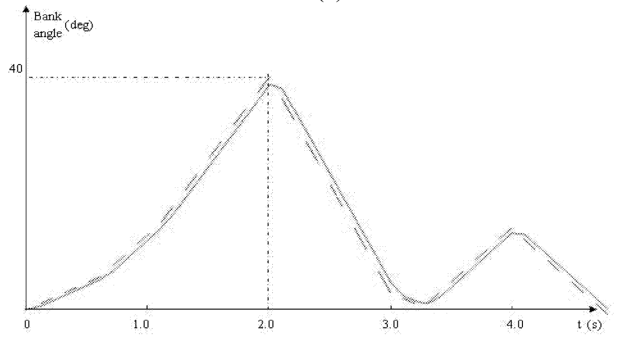
Simulation results

A six-degree-of-freedom (DOF) nonlinear aircraft model has been used in our flight simulation to test the proposed adaptive dynamic inversion controller. The simulation environment is assumed to be at attitude 25,000 ft and Mach=0.6. The physical parameters come from P747. Two sets of simulation results have been presented here. One is the bank-to-turn control, or lateral control. The aircraft rolls following the command while keeping the angle of attack as a constant and slide-angle zero. The other is longitudinal control, the aircraft pitches following the command while keeping both slide-angle and bank-angle zero.

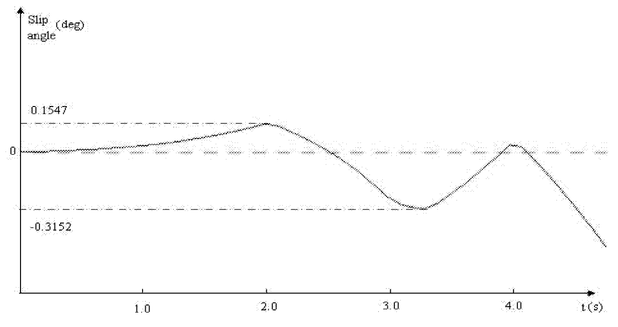
The results of lateral control are shown below in Figure 4. From the simulation results we can draw some interesting conclusions. There is only slight difference between the response of bank angle to adaptive control and non-adaptive control. However, without the adaptive controller, the response of slip angle is twinkling in the area near zero, the amplitude is big. While under the adaptive controller the amplitude has been eliminated considerably. The response of angle of attack diverges according to the command without the adaptive compensation, while the response error has been greatly reduced when the adaptive factor has been introduced into the system.



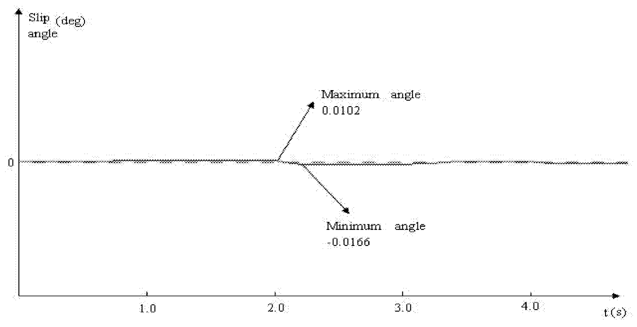
(a)



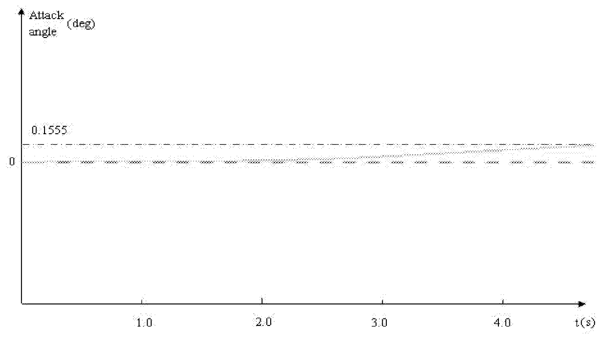
(b)



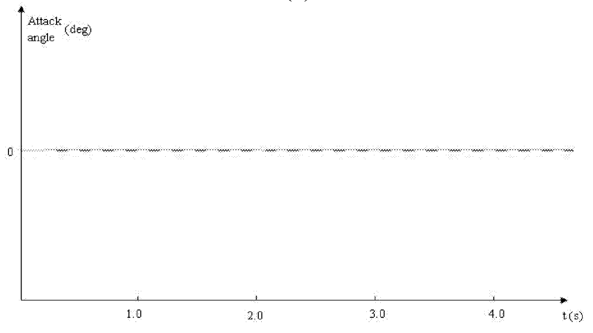
(c)



(d)



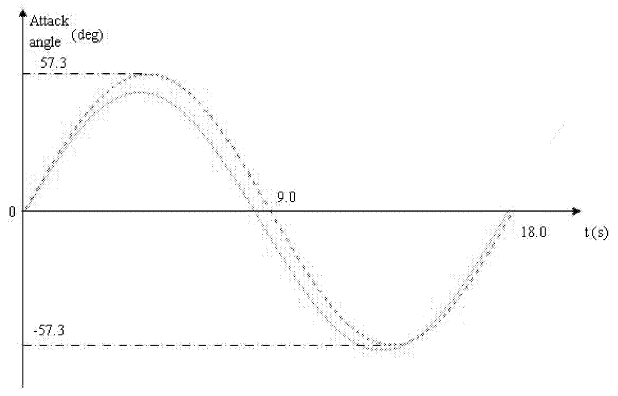
(e)



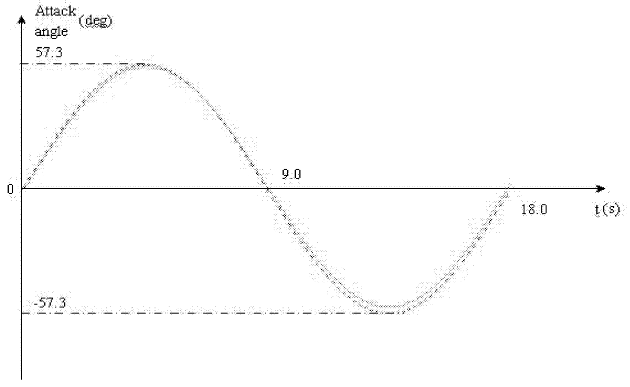
(f)

Figure.4. Lateral control response of 6 DOF aircraft. (a), (c), (e) are the control results without adaptive neural network compensator. (b), (d), (f) are the control results with adaptive neural network compensator. Dotted line is the command; the solid line is the response

The longitudinal control simulation results are shown in Figure 5.



(a)



(b)

Figure.5. Longitudinal control response of 6 DOF aircraft. (a) is the control result without adaptive compensator; (b) is the response with adaptive compensator. Dotted line is the command; the solid line is the response.

From the simulation results, we can see that under the longitudinal control, the system tracks the command much better with an adaptive compensator. In both situations, the adaptive compensator can efficiently improve the control performance. The aircraft model and the proposed control algorithm have been successfully implemented in our 3D flight simulation system. A screen shortcut is shown in Figure 6.

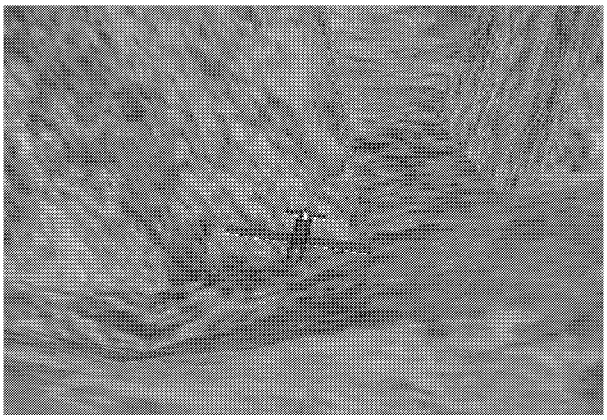


Figure.6. Flights control simulation in 3D environment.

Conclusion

In this paper we have proposed a neural network based nonlinear adaptive controller for autopilot design for aircraft simulation. This approach does not rely on accurate aerodynamic model and extensive simulation data; it can greatly simplify the control design process. This proposed architecture involves two parts: a dynamic inversion (DI) controller that offers the solution for the approximately linear model and a neural network compensator that adaptively cancels the linear approximation errors and un-modelled dynamics errors through offline and online training separately. A six-degree-of-freedom nonlinear aircraft model has been used to test this algorithm. Two sets of simulation results have been presented in this paper. The simulation results shows the proposed adaptive controller works well and can reduce the nonlinear inversion error and improve the control performance efficiently.

References

[1] Daniel E. Miller "A New Approach to Model Reference Adaptive Control", IEEE transactions on automatic control, VOL. 48, NO. 5, May 2003.

[2] Napolitano, M. R. and M. Kincheloe "On-line Learning Neural-Network Controllers for Autopilot Systems". Journal of Guidance, Control and Dynamics, 33(6) 1995.

[3] Cottrell, R. G., T. L. Vincent, and S. H. Sadati "Minimizing interceptor size using neural networks for terminal guidance law synthesis". Journal of Guidance, Control and Dynamics, 19(3), 1996.

[4] Steck, J. E., K. Rokhsaz, and S. P. Shue "Linear and Neural Network Feedback for Flight Control Decoupling". IEEE Trans. Syst. Mag., 16(4) 22-30.

[5]Geetha K. Thampi, Jose C. Principe, Mark A. Motter, JeongHo Cho, Jing Lan, "Multiple Model Based Flight Control Design", Midwest Syposium on Circuits and Systems, Aug, 2002.

[6] Chun Ling Lin and Huai Wen Su, " Intelligent Control Theory in Guidance and Control System Design: an Overview". Proc. Natl. Sci, Counc, ROC(A) VOL.24, No.1, 2000.

[7] Hunt, L, R., Su, R., and Meyer, G., "Global Transformations of Nonlinear Systems," IEEE Transactions on Automatic Control, VOL. 28, 1983, pp. 24-31.

[8] Michael B. McFarland and Anthony J. Calise, "Neural-Adaptive Nonlinear Autopilot Design for An Agile Anti-Air Missle", presented at the AIAA Guidance, Navigation, and Control Conference, San Diego, California, July, 1996.

[9] Joseph M. Cooke, Michael J. Zyda, David R. Pratt and Robert B. McGhee, "NPSNET: Flight Simulation Dynamic Modelling using Quaternion", In Presence, Vol. 1, No. 4, pp 404-420.

VIRTUAL ENVIRONMENTS AND GAME SPACE

DYNAMIC SCENE OCCLUSION CULLING IN 3D VIRTUAL ENVIRONMENTS

Baldeve Paunoo and Daut Daman
Computer Graphics and Multimedia Department
Faculty of Computer Science and Information Technology,
Universiti Teknologi Malaysia.
E-mail: baldeve@msn.com

KEYWORDS
Occlusion culling, dynamic scene portal, DBV.

ABSTRACT

Visibility algorithms have recently regained attention because of the ever increasing size of polygon datasets in a scene and dynamic objects handling makes it impossible to display in real time with conventional approaches. Occlusion culling is one of the key techniques for output-sensitive rendering. Most of the scenes have major static objects and only a few of dynamic objects. Hence, an occlusion culling techniques developed for handling dynamic objects in static scenes using dynamic bounding volume where dynamic objects are wrapped in bounding volumes and inserted into spatial hierarchical data structure as a volume to avoid updating the structure for every dynamic object at each frame is created for each occluded dynamic object by using physical constraint of that object. This algorithm will be implemented using portal occlusion culling which is suitable for indoor and architectural scenes.

INTRODUCTION

Visibility algorithms have recently regained attention [Daniel et al. 2000] because of the ever increasing size of polygon datasets in a scene makes it impossible to display in real time with conventional approaches. Visibility culling aims at omitting invisible geometry before actual hidden surface removal is performed. Many of the researches focus on algorithms for computing tight estimation of visible sets by only drawing visible sets which the subsets of primitives which contributes at least one pixel of the screen. Z-buffer which was used earlier cannot support anymore with the increasing model complexity.

Interactive three-dimensional computer graphic and animation with dynamic objects have gained popularity in 3D game over the last few years. However, their further acceptance is inhibited for dynamic game scenes by the amount of computation they involve [Daniel et al. 2000].

If the game scenes are dynamic, containing moving object, then additional time is needed to update the model to reflect these objects' motions. This make is hard to attain the goal of interactivity in real-time game which requires images to be rendered at rate of about 30 frames per second. Hence, unwanted calculation should be ignored with visibility culling before sending to rendering pipeline.

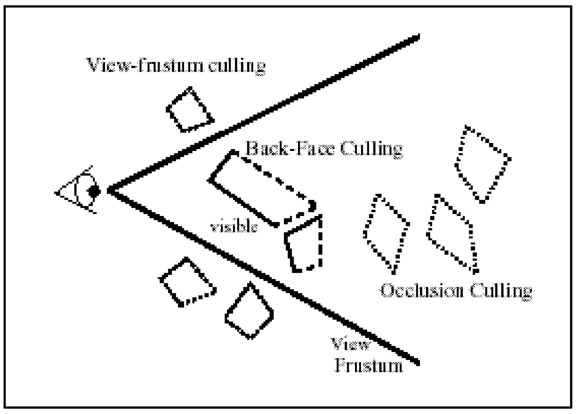


Figure 1: Types of visibility culling [Coorg and Teller 1997].

Visibility culling begins with two conventional techniques which is back-face culling and view-frustum culling [Daniel et al. 2000]. Back-face culling avoids geometry faces that are away from the viewer, while view-frustum culling algorithm omits the rendering geometry that is outside viewing frustum. Three types of culling can be seen in figure 1. Back face culling and view frustum culling are not enough to produce output sensitive graphics. Hence, occlusion culling algorithms were devised. In this review, occlusion culling algorithms will be focused and reveals its advantages and disadvantages.

Occlusion culling also known as visibility culling or output-sensitive visibility calculation [Daniel et al. 2000; Greene et al. 1993] is a special case of visibility calculation. Occlusion culling finds these visible parts without wasting time on the occluded parts, not even to determine that they are occluded. It thus saves the time it would otherwise take to transform the unseen parts to image coordinates, clip and render them. This technique is global as it involves interrelationship among polygon so it is far more complex than back-face and view-frustum culling. Output-sensitive rendering allows unlimited complexity to be rendered at real-time rate compared to hardware acceleration alone [Schrocker 2001].

BACKGROUND RESEARCH

Airey et al. [1991] and Teller and Sequin [1991] proposed an occlusion culling algorithm for static architectural scene and developed foundation for the recent works. This conservative technique need to be pre-computed to calculate potential visible sets or PVS. Their technique is suitable for indoor architectural scenes where rooms inter-connected with another with portal. Their work was further extended by Luebke and Georges [1995]. Instead of region based visibility, they proposed point based visibility. Luebke and Georges perform on the fly visibility calculations using depth traversal of the cells using screen space projection of the portals. However, to develop a dynamic scene, it needs to update hierarchical data structure.

A similar way to look at the occlusion is shadow generated from a light source from the viewpoint. Therefore, Hudson et al. [1997] proposed an approach on dynamically choosing a set of occluders and computing their shadow frusta, which is used for culling the bounding boxes of a hierarchy of objects. However, it still needs to depend on the speed to update hierarchy data structure for dynamic scenes. Britter et al. [1998] improved the method described by Hudson using BSP trees. They combined the shadow frusta of the occluders into an occlusion tree.

N. Greene et al. [1993] used an octree for object precision and a Z-pyramid for image precision. The Z-pyramid is a layered buffer with a different resolution at each level. While the hierarchical Z-buffer algorithm is primarily intended for static scenes, Greene's mentions a few ideas for handling dynamic objects. However, none of them has been implemented or seriously explored. Z-buffer in the hardware need to be modified to allow real-time performance.

A similar method with hierarchical Z-buffer was proposed by H. Zhang [1997]. It decouples the visibility test into an overlap test and a depth test. It also support approximate visibility culling where objects that are visible through only a few pixel can be cull using an opacity threshold. This method need to preprocess a database of potential occluders. For dynamic scenes, both the potential occluder set and hierarchical data structure are omitted and object bounding boxes are used instead. However, due to the omission of hierarchical data structure, this method is not output sensitive for increasing size of dynamic models.

Naylor [1994] proposed an algorithm that performs output-sensitive visibility calculation using BSP trees. A BSP (binary space partitioning) tree can represent the scene itself without additional data structure. The construction of BSP tree is very time consuming but it is only constructed once as preprocessing stage and subsequently used for visibility calculation from many viewpoints.

RESEARCH MOTIVATION

Current occlusion culling algorithms scenes [Coorg and Teller 1997; Naylor 1994; Airey 1991; Teller and Sequin 1991; Luebke and Georges 1995; Zhang 1997] are inappropriate for dynamic because these algorithms highly depend on spatial hierarchical data structures and pre-computation. It takes much longer time [Helin 2003] to construct the spatial hierarchical data structure than to just render the whole scene from any single viewpoint. Therefore the structure is built at preprocessing, under the assumption that all the objects are static. While some of the existing algorithms allow the exploitation of temporal coherence in animation sequences [Sudarsky 1998], these are restricted to walkthrough animations, in which the scene is static and only the viewpoint moves through it. If anything other than the viewpoint moves in the scene, then the data structure used by the occlusion culling algorithm becomes outdated and incorrect images may result. Thus, the only interaction or animation sequences that existing occlusion culling algorithms allow are walk-through or fly-through, where the entire scene is static and only the viewer moves through it [Sudarsky 1998]. It is possible, in principle, to update the data structure to reflect the dynamic objects' movements. However, if this update is not done carefully, it might take far too much time longer than the normal rendering of all the scene objects. In particular, the update should be avoided for dynamic objects which are currently hidden; otherwise the goal of output sensitivity will not be attained. On the other hand, one cannot totally ignore occluded dynamic objects, because they may become visible sometime.

Obviously, it is out of the question to reconstruct the data structure each time an object moves in the scene, as this would be much slower than just displaying everything by the plain Z-buffer algorithm. The existing data structure should be updated for the dynamic objects motions, rather than being initialized from scratch [Sudarsky 1998]. However, if this update is performed for every object movement, then the overall algorithm will not be output-sensitive; it will waste time on updating the structure for occluded dynamic objects as well as for visible ones. To preserve output-sensitivity, the update should only be performed in the visible parts of the data structure. Objects moving in other regions should somehow be ignored. Most of the scenes are static and only few objects are dynamic. Therefore, more efficient algorithm should be developed without trading performance.

What is needed is a mechanism to ignore hidden dynamic objects most of the time, yet be notified when they might no longer be hidden. Such a mechanism can eliminate not only the time it would otherwise take to render hidden dynamic objects or to maintain a spatial data structure according to their motions; it can also save the time to update the hidden objects positions and configurations. This can constitute a great saving in a distributed graphic environment [Earnshaw 1995], where dynamic objects are controlled by remote machines, because the eliminated update messages would have been sent through a relatively slow communication link. An algorithm has to be developed to achieve this avoidance of computations and communications for hidden dynamic objects. The data structure used by an occlusion culling method is updated to reflect the dynamic objects' current positions. The update is not performed for all of the dynamic objects, but only for those that may be visible [Sudarsky 1998]. The culling algorithm itself determines which of the dynamic objects might be seen. The algorithm is based on the observation that the possible movements of dynamic objects may be subject to some known constraints. Such constraints may be imposed, for example, by physical simulation or by a user interface. Given these constraints, and given a static-scene occlusion culling technique, the algorithm adapts the culling technique to dynamic scenes containing multiple dynamic objects.

The dynamic scene occlusion culling algorithm's runtime is linearly affected only by the scene's visible parts, not by hidden parts or by occluded dynamic objects [Daniel et al. 2000]. Furthermore, the algorithm is expected to save unnecessary updates of obscured dynamic objects which can't be seen. It can therefore be used to save update messages in distributed graphical systems, such as multi-user virtual reality environments. This saving can be significant if the eliminated messages would have been sent through a slow communications channel, such as the Internet. Using the proposed algorithm, real time dynamic scene can be produced using normal personal computers.

PROBLEMS HANDLING DYNAMIC OBJECTS

Existing occlusion culling algorithms for dynamic scenes rely on the large complex spatial hierarchical data structures. However, this hierarchical data structures is very time consuming to update dynamic objects in the trees for every frame. Some data structures such as BSP [Ranta 2001] preprocessed for hours before it can be used. While there are some modified spatial hierarchical data structures to handle dynamic objects in real-time, it is waste of processing time to calculate and update dynamic objects which are hidden. Only visible objects should be calculated and ignores the others to preserve output-sensitivity.

Some of the algorithms can use temporal coherence but it is limited to the static scenes. The occlusion culling algorithm that runs on the data structure can report which of the objects it encountered in the last frame displayed, all the other objects were hidden. However, the update of the structure for these unseen objects cannot be simply omitted for the next frame, because they might become visible. Furthermore, if a dynamic object is normally ignored from the moment it is found to be hidden, then it might not be displayed again when it should be. This can happen because the culling algorithm does not traverse the entire data structure, but only its observable part since the object's position in the structure would remain outdated, the algorithm might miss it. In fact, if the object's last updated position remains occluded, then the object will never be seen again.

PROPOSED METHOD : DYNAMIC BOUNDING VOLUMES

If the world is fully static, everything becomes much simpler. First of all, the need for physic and dynamics are all gone. Visibility determination and spatial data structures also become significantly easier. Unfortunately, such constraint is not suitable for most of the virtual environment. However, most of the object in a virtual environment is static. A house does not move but when the window broken, it is dynamic for a few moments. This observation shows that most of the object in the scene is static until there is a certain time or event which causes it to be dynamic for a short period of time.

Therefore a technique developed in this research by introducing dynamic bounding volume (DBV). Dynamic bounding volume is similar to temporal bounding volume (TBV) which was presented by Sudarsky and Gothman [1998]. However, dynamic bounding volume proposed in this research does not restrict the dynamic objects temporally. This method is simpler because it shows the visible object including dynamic objects without validity time. The technique developed in this research avoids wasting time on updating the spatial data structure for hidden dynamic objects, yet circumvents the above-mentioned problems by employing dynamic bounding volumes (DBVs). A DBV is a volume where one or many dynamic objects have the possibility to appear depending on the physic parameter or pre-determined motion in a path. This volume is treated as a single big static object. It will be treated same as other static objects until this volume is tested appear in the scene. "Validity period" also can be used in dynamic bounding volume. DBVs are based on some known constraints on the objects' behaviors.

Dynamic objects are wrapped with bounding volume. This bounding volume can be any shape as long as the dynamic objects are inside the bounding volume. Although the dynamic object may move in uncertain manner, simple object such as bounding box or sphere is recommended to ease occlusion calculation. Dynamic bounding volumes are inserted into spatial hierarchical data structures. This method can be used with most of the spatial hierarchical data structures with some adaptation. This research will use binary space partition (BSP) tree to demonstrate the proposed algorithm.

Occlusion culling algorithm is performed as usual. Hidden dynamic objects are ignored until occlusion culling algorithm determines that the DBV is visible. If dynamic bounding volume is visible, it does not guarantee the dynamic object is visible but there are high possibilities. Depending on the scene complexity, visible bounding volume can be rendered without any occlusion culling or take further step to test the dynamic objects for occlusion. For low complexity scene, occlusion culling in this step does not always improve the processing.

The object should be considered again, because it is no longer guaranteed to be occluded. A priority queue of DBV expiries, similar to event queues used in simulation, notifies when DBVs cease to be valid. As long as expiry dates are chosen with sufficient care most volumes remain invisible throughout most of their validity periods. Thus, in every frame, the majority of the hidden dynamic objects are ignored, and output sensitivity is maintained.

This approach is a variation on the lazy evaluation principle which do not compute anything until it is necessary. Note that the DBVs are calculated on the fly, and that prior knowledge of the objects' trajectories is not essential. Thus compatibility is assured with interactive applications in which this information is not available in advance, such as simulations, games and virtual reality.

Dynamic Bounding Volume Validity Periods

Each bounding volume assigned with a validity period. In general situation, it is best to use an adaptive algorithm to choose the validity periods. If a validity period of the bounding volume expires frequently, it means that the period is too short thus decreasing efficiency. So the next DBV will be assigned a longer period. On the other hand, if DBV becomes visible too frequently, it means that the bounding` volume was too big or the validity period was too long. Thus the next validity period for that bounding volume will be assigned a shorter period.

Priority queue of these dynamic bounding volumes is used to find the expiring ones. In addition, the algorithm must be modified so that it handles DBV and objects instead of objects only. This strategy makes validity periods adapt to their object's behavior and visibility status. Dynamic objects that are fast-moving or stay near visible regions of space of space have relatively short validity periods. Objects in obscured regions have longer periods and are sampled less frequently as time progress.

Portal Rendering with Dynamic Objects

Portal rendering was devised by David P.Luebke and Chris Georges [1995], is an intuitive algorithm ideally suited for visualization of indoor scenery such as those found in architectural models.

Interior of buildings have as a series of interconnected rooms of differing sizes. This intuitive concept is also what forms the basis of portal rendering technology, where it deal with a number of sectors connected to each other by portals. Sectors are three-dimensional volumes defined by a number of polygons, just like any computer 3D model would define room. Each polygon has a material and texture properties associated with it. Some polygons are marked as portal. Portal is a form of interface between the current sector with another sector. Moving from one sector to another is only performed by passing through portals, thus the rest of the world can discarded and concentrate on the new sector.

The actual traversal of this data structures is easily and extremely elegantly performed using a recursive algorithm. First, rendering in the camera sector is initiated using the view port as the clipping volume. Move through each polygon contained in the sector and draw them to the screen. If portal polygon is visible, rendering procedure is called recursively using the portal polygon as the new clipping volume. Thus, all polygons are rendered in the new sector will be confined to the extents of the last portal. In effect, only potentially visible polygons will be sent to the rendering pipeline.

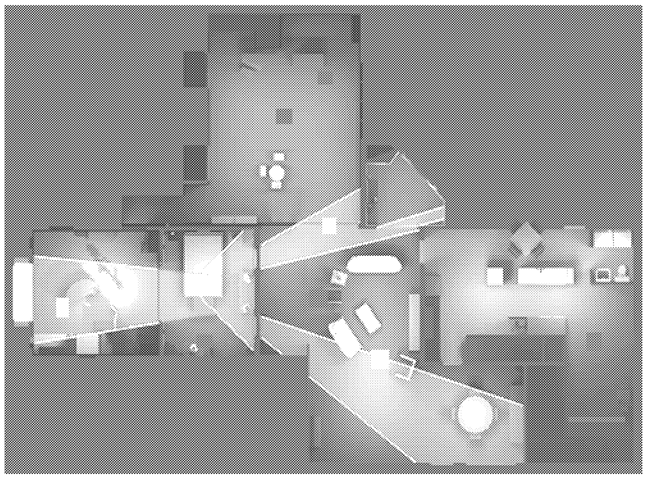


Figure 2: Portal rendering with occluded dynamic objects (red box) and visible dynamic objects (green box).

Even though portal rendering can handle dynamic objects, it still depends on the update of hierarchical data structure [Daniel et al. 2000]. However, data structure such as BSP-tree in not suitable to be updated in real-time [Ranta 2001]. Figure 2 shows the view from a cell to another cell through door and mirror (portal). Green boxes are visible dynamic bounding volume while the red boxes represent occluded dynamic bounding volume. Occluded dynamic objects should be ignored until the validity period of the volume expired or fully or partially visible.

PROPOSED ALGORITHMS

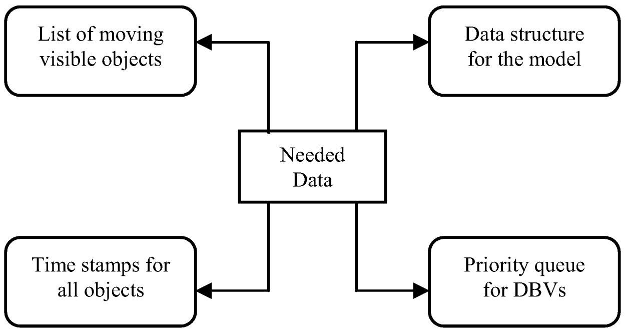

List of moving visible objects
Data structure for the model
Needed Data
Time stamps for all objects
Priority queue for DBVs

Figure 3: Proposed algorithm sub-module chart

Following steps are executed for each frame:

1. If there are any expiring DBVs in the DBV priority queue, they are removed from the queue and their objects are added to the list of moving visible objects.
2. All objects in the list of moving visible objects are calculated for their new positions, and these new positions are updated in the data structure that represents the model
3. Run the visibility algorithm on the data structure as usual. If one encounters a node that contains a DBV, these DBVs are removed from the DBV priority queue and their objects are added to the list of visible objects. Also their new positions are calculated. For each visible object in the node, update its time stamp to the current time.

4. When the algorithm ends, the list of visible objects is updated so that each object that has an old time stamp is removed from the list. Since these objects are now becoming invisible, they are associated with a new DBV that is inserted into the DBV priority queue.

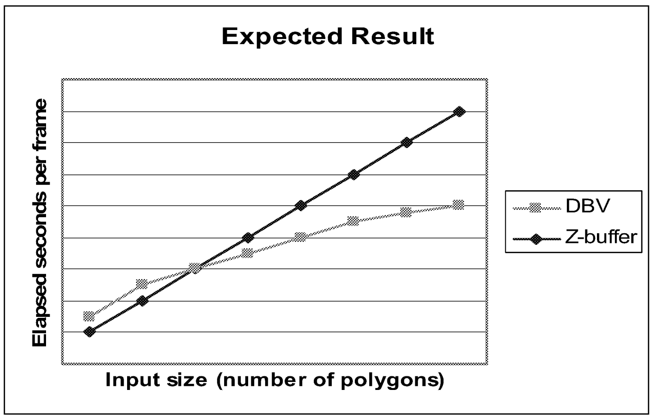


Figure 4: Performance of naïve portal culling with Z-buffer and portal rendering with DBV for dynamic scenes

Figure 4 shows the expected performance of naïve occlusion culling rendering and occlusion culling with dynamic bounding volume for portal with dynamic objects. This shows that with varying number of dynamic objects, frame rate for portal rendering increases linearly as number of polygon increases. In the other hand, the performance of portal rendering with DBV should be as the result of ignoring invisible dynamic objects.

CONCLUSION

In this research, a scheme for occlusion culling in dynamic scenes will be developed. The scheme is based on an existing static-scene occlusion culling technique, and generalizes it to dynamic scenes by updating its underlying spatial data structure according to the movements of dynamic objects. The structure is not updated for every object motion, but only for the visible objects and for a minority of the hidden ones. The dynamic scene occlusion culling algorithm ignores most of the unseen dynamic objects most of the time. Not only is it unnecessary to display these objects and to modify the spatial data structure for their movements, it is also not required to be constantly updated of their current positions.

REFERENCES

Bittner J., Havran V., and Slavik P. 1998. "Hierarchical visibility culling with occlusion trees". In *Proceedings of Computer Graphics International 98.*

Bruce F. Naylor 1994 "Partitioning Tree Image Representation and Generation from 3D Geometric Models." ACM SIGGRAPH.

Coorg S. and Teller S. 1997. "Real-time occlusion culling for models with large occluders." *Proceedings of the 1997 Symposium on Interactive 3D Graphics*, Providence, Rhode Island,. ACM SIGGRAPH

Daniel Coden, Yiorgos Chrysanthou Claudio T Silva, Fredo Durand 2000. "A Survey of Visibility for Walkthrough Applications." ACM Computer Surveys..

Daut Daman and Baldeve Paunoo 2004. "A Review Of Occlusion Culling in Walkthrough Applications". *In Proceedings of The 1st Conference on Telematics System, Services, and Applications 2004 (TSSA2004*).

David Luebkee and Chris Geoges 1995. "Portals and mirrors: Simple, fast evaluation of potentially visible sets". In Pat Hanrahan and JimWinget, editors, *1995 Symposium on Interactive 3D Graphics*, pages 105–106. ACM SIGGRAPH.

Gerald Schrockers 2001. *Visibility Culling for Game Application.* Thesis. Institute for Computer Graphics and Vision, Graz "University of Technology.

John Airey 1991. *Increasing Update Rates in the Building Walkthrough System with Automatic Model-Space Subdivision and Potentially Visible Set Calculations.* PhD thesis, University of North Carolina, Chappel Hill.

Ned Greene, M. Kass and Gary Miller 1993. "Hierarchical z-buffer visibility". *Proceedings of SIGGRAPH 93.*

Oded Sudarsky 2001. *Dynamic Scene Occlusion Culling.* Phd Thesis, Israel Institute of Technology.

Samuel Ranta Eskola 2001. *Binary Space Partioning Trees and Polygon Removal in Real Time 3D Rendering.* PhD. Thesis. Computing Science Department, Uppsala University.

Seth J. Teller and Carlo h Sequin 1991. "Visibility preprocessing for interactive walkthroughs." *Computer Graphics (Proceedings of SIGGRAPH 91)*, 25(4):61–69.

Thomas Hudson, Dinesh Manocha, Jonathan Cohen, Ming Lin, Kenneth Hodd and Hansong Zhang 1997. "Accelerated occlusion culling using shadow frusta." In *Proceesings of ACM Symposium on Computational Geometry.*

Ville Helin 2003. "Hierarchies for Occlusion Culling". *Seminar on Computer Graphic, Helsinki University of Technology.*

Zhang H., Manocha D, Hudson T. And Koff K. E. 1997 "Visibility culling using hierarchical occlusion maps". In *SIGGRAPH 97 Conference Proceedings*, pages 77–88, Los Angeles.

An Architecture For Domain-Independent Collaborative Virtual Environments

Ahmed BinSubaih
Steve Maddock
Daniela Romano
Department of Computer Science
Regent Court, 211 Portobello Street
Sheffield, S1 4DP, UK
Email: a.binsubaih, s.maddock, d.romano@sheff.ac.uk

ABSTRACT

The increase of computing power and its wide availability has raised interest in the use of Collaborative Virtual Environments for training purposes. Nevertheless, many of the currently developed training simulations are inflexible and support only a limited number of scenarios, often limited to a single domain. This work aims to investigate the challenges of separating the domain logic from the system using it. The work presented in this paper proposes to achieve the above by separating the architecture into three parts: domain knowledge, simulation environment and *events space* which links the previous two. The domain knowledge holds the scenario logic or behaviour which dictates how the scenario should run. The simulation environment is where participants meet and interact. The *events space* links the two parts by using the domain knowledge to control the scenario running in the simulation environment. By formulating our system in this manner we attempt to achieve the flexibility pursued and identify and tackle the challenges involved.

Keywords
Collaborative virtual environment (CVE), knowledge-base systems (KB), architecture, events

INTRODUCTION

A number of virtual environments (VEs) and collaborative virtual environments (CVEs) have been developed over the years, looking at such things as training firefighters (Romano 2001), police officers (Williams 1995) and navy personnel (Hamman et al 2001a), and the reconstruction of traffic accidents from textual reports (Akerberg et al 2003). The main issue with such systems is that they are domain dependent, which makes it difficult to reuse their simulation systems on different domains without extensive work. However, this is an expensive and time-consuming process. As identified in (Dachselt 2001) domain specificity in system development leads to highly inflexible applications.

Our work investigates how the disciplines of distributed environments and expert systems can be combined in an attempt to achieve the separation required which we believe leads to a domain-independent architecture. From distributed environments we investigate the suitability of events as a communication mechanism between the simulation environment and the *events space*. From expert systems we examine the applicability of using knowledge-base systems (KB) to store and inference the domain knowledge which is used to control the scenario.

One of the major contributions of this paper is introducing an *events space* which is an intermediary between the scenario logic and the simulation environment. Using such an intermediary should result in decoupling the scenario logic from the simulation environment which confers several advantages:

- The logic and the simulation environment can be modified entirely independently allowing iterative development cycles.
- The decoupling encourages encapsulation and other good object-oriented coding practices.
- It enables interoperability between distinct simulation environments, a practice promoted by the High Level Architecture (HLA) (Smith 1998). Although HLA promotes interoperability, its object management allows the subscription and discovering of remote objects thus violating the space decoupling[1]. Further concerns of the HLA are covered in (Davis and Moeller 1999)
- The three different parts (KB, *events space*, and simulation environment) can be individually tailored to the expertise and computer literacy of their users (domain experts, trainer, and trainees).

Furthermore, the *events space* attempts to achieve the following goals:

- Automatic generation and control of scenarios for training purposes.
- Use the ability of KB systems to provide explanation of solutions to guide trainees during simulation sessions.

The inherent distributed nature of a CVE and the existence of many different techniques for communication, such as the events mechanism with its decoupling ability (Drew et al, Eugster et al 2003), have encouraged us to attempt the separation on a collaborative virtual environment rather than on a VE.

In this paper, we first describe the related work, followed by a detailed presentation of the proposed architecture where we show the different structures and how they communicate to achieve the independence sought. Finally, we illustrate the first prototype developed by showing its ability to run two distinct scenarios: investigating the aftermath of a vehicle accident situation and virtual lecturing.

RELATED WORK

In this section we describe the methods used to structure simulation environments and categorize them with regards to the relationship between them and the scenario logic. Such categorization examines this relationship in terms of the communication approaches employed to control the scenario logic and the flow of the scenario logic from its source to the simulation environment.

We can consider the following different categories based on the relationship between the scenario logic and the simulation environment: applications, virtual development environments, commercial software environments, and KB systems.

Certain applications (Romano 2001) tend to embed the scenario logic inside the simulation environment code and most likely changes to the logic require recompiling the simulation system. Other applications (Akerberg 2003, Hamman et al 2001a, Williams 1995) attempt to give some ability to modify the logic without recompiling, thus eliminating the developers involvement in the scenario creation cycle. This is accomplished by providing ways for the domain experts or trainers to insert the logic into the simulation environment. However, these applications are usually very domain dependent.

[1] Space decoupling (Davis and Moeller 1999)

Some virtual development environments (Cruz-Neira et al 2002, Hawkes and Wray 1998, Shaw et al 1992, Tamberend 2003, Wang et al 1995) are built specifically for the use of developers producing virtual environments. These environments normally ease the development lifecycle by abstracting the low level complexities such as interacting with VR devices. However, some of these development environments are no different than using a programming language in the sense that it places the creation of the link between the logic and the simulation environment in the hands of the developer. Nevertheless, some of these development environments (Hawkes and Wray 1998, Tamberend 2003) encourage flexibility by providing a high level scripting language, thus making it feasible to be used for separating and modifying the scenario logic. One tool (Shaw et al 1992) even provides a higher level of support in the form of a simple script file for creating environments.

The commercial software environments (such as DI-Guy™ and Vega™) address the domain independence in a much better way by allowing the logic to be inserted using a graphical interface or a high-level scripting language. These succeed in achieving domain independence but usually the logic gets formatted in a proprietary format to the specific environment, thus lacking the simulation environment independence. One of the strengths of these tools is their ability to cater for a wide range of users by providing interaction methods of different levels of complexity. Such tools can be used by domain experts or scenario creators using the graphical interfaces provided. The developers also can make use of the API access provided.

KB systems (Hamman et al 2001b, Szarowicz et al 2002) are geared towards separating the logic or knowledge from the system using it. They have an inference engine to deal with retrieving the appropriate results. Furthermore, the separation also allows the modification of the knowledge independently from the simulation environment and more frequently without developer involvement, which means there is no need for recompilation of the simulation.

The drawbacks of some of the previous categories are:

- The embedding of the logic in the simulation environment makes it inflexible to change.
- The logic usually tends to be specific to the simulation environment and requires some work to be able to reuse it in a different simulation environment. This usually makes the logic created limited to a specific simulation environment.

Some of the strengths of the previous categories are:

- Interoperability between different simulation environments.
- The separation of the logic from the simulation environment shown by the KB systems.
- The decoupling accomplished using events mechanisms.
- Providing a high-level scripting language makes the interaction with the simulation environment less complex.

In comparison with the above categorization we propose a new category we call 'simulation services' (similar to web services). This category attempts to combine the best practices from the above categories and also avoid their drawbacks. The main goal of this category is to make the simulation 'brain' run as a service by linking it with the logic from one side and the simulation environment from the other side, therefore advocating independence from both sides. The independence from the simulation environment means that the 'brain' can be reused to service other simulation environments, possibly built using different languages, as long as they conform to common communication protocols.

OUR APPROACH

The simulation environment architecture we pursue puts an intermediary between the scenario logic and the simulation environment in an attempt to be domain and simulation environment independent simultaneously.

The distinguishing factor of the proposed intermediary lies in the way it is modularised and run as a separate service provider, used to automatically generate and service scenario behaviours to a simulation environment. The scenario behaviours are constructed from a knowledge base representing a specific domain. Participants in the simulation environment communicate their status to the intermediary and receive events to decide their course of action. The communication between the simulation and the intermediary in a second stage of development will be achieved through the use of an events service and more specifically the publish/subscribe mechanism.

KB systems have an advantage over conventional algorithmic techniques when solving complex problems. Reasoning to solve complex issues using knowledge (e.g. rule-based systems) is much simpler than reasoning about algorithms which have loops and branches (Hook 2004).

The architecture proposed is composed of three main parts: the KB, the *events space*, and the simulation environment. These need to communicate with each other in order to run the simulation environment. The role of the events mechanisms is to couple the intermediary *events space* to the simulation environment. There are many different events mechanisms such as publish/subscribe, message passing, remote procedure call (RPC), notifications, shared space and message queuing. We chose the publish/subscribe mechanism because it offers full decoupling, as explained in (Eugster et al 2003).

We suggest that using a KB system from one side and publish/subscribe event mechanism on the other side will not only achieve the proposed separation goal but it will also contribute immensely to making the intermediary module, the *events space*, fully decoupled from the simulation environment side.

EVENTS SPACE AS INTERMEDIARY

The conceptual design for the *events space* is shown in Figure 1. The directed arrows show the flow of information amongst the *events space* and the three main entities: KB, entities and relationships, and

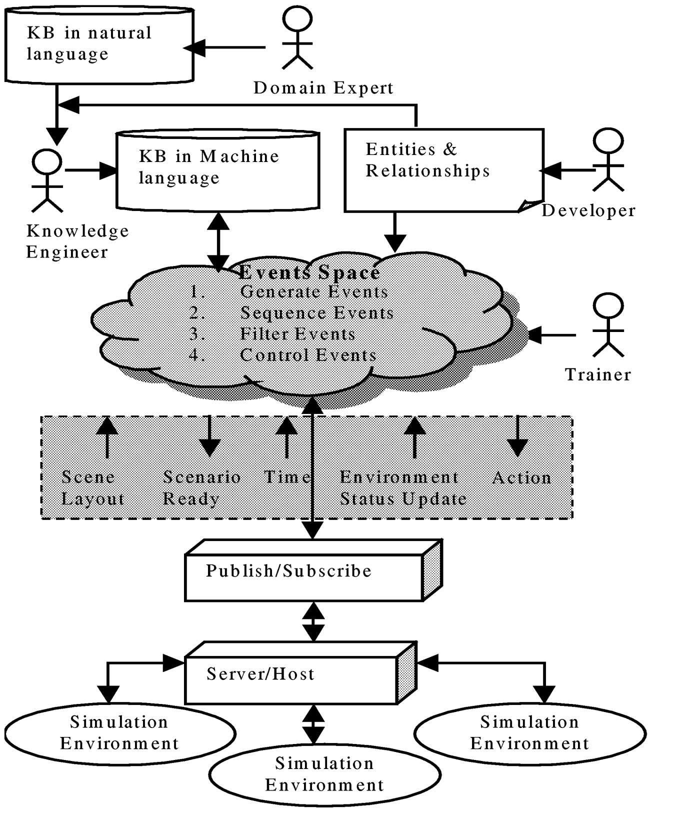


Figure 1. Events space as a link between knowledge base and simulation environments

simulation environment/s. The process of creating a virtual environment will undergo the following steps.

First, the KB is simply information elicited from domain expert/s and represented in a natural language. A Knowledge Engineer then formats such knowledge into a format that is specific to the *events space*. At this point the *events space* can access the knowledge and use it to create and control scenarios.

Second, the simulation environment developer provides the entities and relationships that represent the simulation environment engine capabilities in the form of a class diagram showing the methods, variables, etc. The format that represents the entities and relationships is again specific to the *events space*.

Third, the simulation environment sends the scene layout, time, and environment status and receives a message to state that the scenario is ready and receives events dictating what should happen next.

Once the *events space* receives the KB and entities and relationships in its format, it can then start creating various scenarios. The process of creating these scenarios starts by creating single events, which are filtered into plausible and implausible events based either on human intervention or set criteria (e.g. event duration). The filtered events are then sequenced to make up the scenario. These sequences are passed again through a filter to validate them. On the other end when the simulation environment passes its scene layout to the *events space*, a sequence of possible events from the accepted scenarios is chosen. This sequence must conform to the content of the environment layout. For example, no animation of person in pain is shown unless there is an injured person in the scene layout provided.

The scenario creation process described above can run offline and the results are stored for future use. To start the *events space* the simulation environment needs to send its simulation time and the simulation commences. The role of the *events space* switches to controlling the scenario based on the current time of the simulation and the environment status received from the various simulation environments. The *events space* generates events based on time and/or the occurrence of some behaviour (e.g. collision, user action, time increasing, etc) in one of the simulation environments. The generated events are then passed to the simulation environment.

The trainer is provided with an interface to the *events space* to allow him to filter events, create scenarios, and control scenarios created by the *events space*. The first two are done offline whereas the controlling is done at run-time where the trainer can monitor the training progress and alter the scenario to guide it towards a specific training path.

Inputs

The following sections describe the entities that interact with the *events space* in more detail and provide samples of the format.

Entities and relationships

The entities and relationships module holds a description of how events can take place in the simulation environment (i.e. the capabilities of the simulation environment - these capabilities at low level reflect the functionality of the simulation engine). The *events space* uses this information to form the events that are then passed to the simulation environment to be interpreted and acted upon. The access to the simulation environment is provided by embedding a high-level scripting language that allows run-time access to the classes' properties and methods.

KB

The knowledge base module holds the knowledge of the considered domain acquired from the domain expert by the knowledge engineer. In its natural language form it can be represented using simple rules in the form of 'IF (condition) THEN (action)'. These rules are then translated into the specific format of the *events space* by a knowledge engineer who uses the entities and relationships format guidelines, which describe the different entities, their attributes, and how they are related to each other.

Simulation Environment

The *events space* requires three types of information from the simulation environment: the scene layout, the simulation time, and changes in the environment status. The information that the *events space* sends is an acknowledgement of the readiness of the scenario and messages describing what should happen next in the simulation environment based on the current time and environment status.

The communication between the host machine that services the simulation environments and the *events space* is accomplished through the use of the publish/subscribe event mechanism. This is the only mechanism which offers full decoupling between the *events space* and the server that hosts the simulation environments. In the publish/subscribe mechanism, if two parties want to communicate then one party needs to advertise an event with the publish/subscribe service and the second party to subscribe to the advertised event.

Trainer

A trainer can interact with the *events space* in three ways: he can manually create and modify scenarios, he can filter automatically created scenarios and he can monitor currently running scenarios to

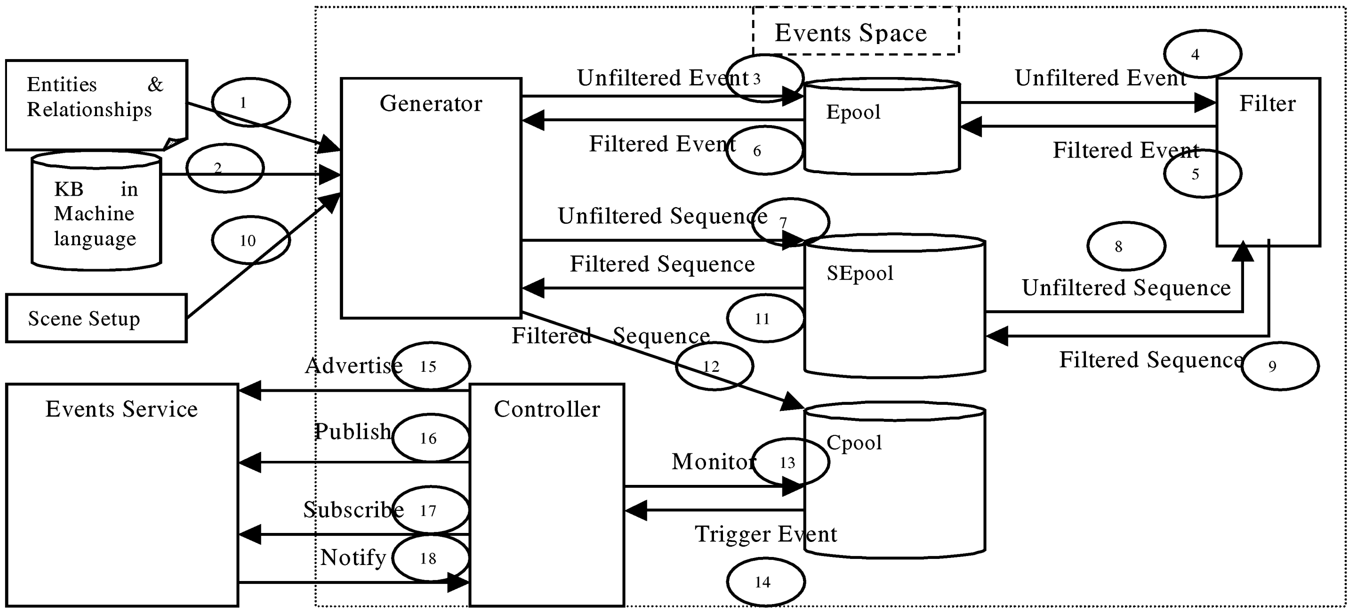


Figure 2: The internal workflow of the events space

redirect the training course to guide the training along different tracks.

Workflow

Figure 2 shows a sequenced workflow of the architecture. The dotted box shows the boundaries of the events space. The workflow can be divided into two main phases. The first phase occurs offline before the simulation environment connects to the events space. The second phase occurs after the simulation environment makes the connection.

The sequenced arrows 1 to 9 mark the first phase where the knowledge base rules are passed in machine language along with the entities and relationships to the generator component. The generator then creates events and stores them in the Epool. After that, these events are filtered by the filter and marked accepted or unaccepted. The accepted events get used by the generator to construct sequences that are placed in the SEpool. Similarly to the events, the filter again reviews these sequences and marks them acceptable or unacceptable. The next step, shown by sequenced arrows 10 to 18, occurs when the simulation sends its scene set-up to the generator to create events specific to the scene provided.

The scene set-up is used to filter sequences, and sequences that match the provided scene are then passed to the Cpool. The Cpool is used by the controller component to service events to the simulation environment.

The controller starts by receiving the synchronised event from the events service which the controller subscribed to. After synchronising the time, the controller starts monitoring the events in the Cpool to search for events consistent with the current time to be fired. The controller also checks if the preconditions of any of the events are satisfied by receiving status events from the event service. Moreover, the controller examines whether triggering an event requires other events to be triggered. To take advantage of the reasoning strength of KB systems, the controller can also be used to provide hints to the trainee by checking the Cpool, SEpool, Epool, and KB in machine language respectively for actions to the current situation.

A FIRST PROTOYPE

In the first prototype of the proposed architecture a human trainer acts as substitute for the proposed *events space*. This prototype has been used to evaluate the architecture's flexibility and suitability for collaborative virtual environments. The aim of this implementation is to simulate the behaviour of the *events space* proposed by having a domain expert doing its tasks (i.e. creating, monitoring, and controlling the simulation behaviour). Furthermore, the architecture should cater for inserting different scene layouts to make it scene independent as well. The trainer, acting as the *events space*, is able to monitor the simulation environment and trigger and control events as he sees appropriate to achieve the goals of the scenario that is running.

The two main goals to be established at this stage to address the flexibility issue are: the ability to provide a method for inserting the behaviour desired into the system without reprogramming and recompiling the application, and the ability of the prototype to allow run-time control of the behaviour inserted.

- ➢ Creating scenario behaviour: allows the trainer to create different scenarios by adding events such as adding animation paths for objects or attaching sounds to objects.
- ➢ Monitoring scenario behaviour: permits the trainer to watch the scenario unfolding by joining the scenario as an invisible participant.
- ➢ Controlling scenario behaviour: allows the trainer to stop or change the course of the simulation behaviour.

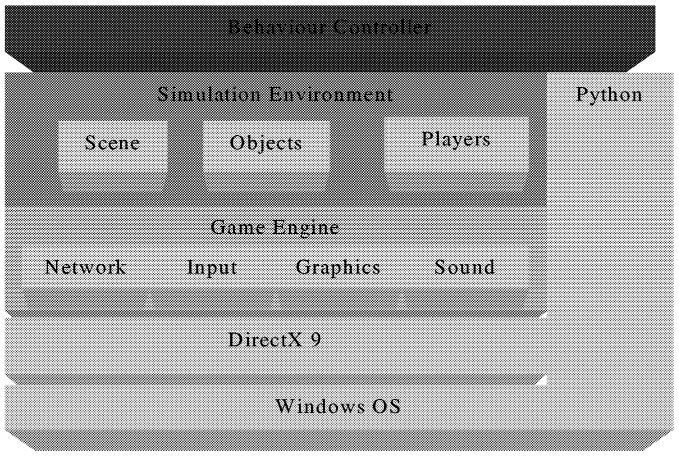


Figure 3. Simulation environment architecture

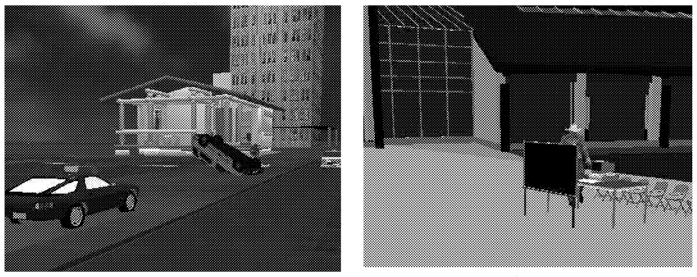


Figure 4. Vehicle accident investigation (a), virtual lecturing (b)

- ➢ Inserting a scene layout: permits the trainer to insert different scenes making the architecture scene independent.

To enable the manipulation of the scenario behaviour, a high-level scripting language has been embedded in the architecture as shown in Figure 3. The language used in our prototype is Python, which allows for easy replication of the game engine classes that are based on an object-oriented approach using C++. Furthermore, the other advantage scripting provides to this prototype is the ability to dynamically load code (Hook 2004) which can be used to insert and control the behaviour. Figure 3 shows how the scripting language is embedded across three architectural layers (DirectX 9.0, simulation engine, and simulation environment), which allows it to access any of them. A game engine has been built on top of DirectX to abstract all of its complexities. The simulation environment lies above the game engine and holds the objects, the players, and the scene settings. The top layer is the behaviour controller which allows run-time access to the simulation environment and allows the trainer to broadcast events inserting or changing the behaviour of any objects, players, or part of the simulation environment.

The scenario creation passes through the following steps: model creation, scene set-up, and scenario configuration. The models are created using a 3D drawing tool; we have used 3D Studio Max 6.0. Optimised models are then exported to the Microsoft DirectX file format, which is then used by the Scenario Creator. Subsequently, the scene layout is created by positioning and orienting the objects in the scene.

The Scenario Creator inserts the objects' information into a database contains five tables, three of which are specific to the scenario (games, players, and scene objects) and the other two hold reusable general data (meshes and groups). The scene tab allows the insertion and positioning of objects in the environment. Titles are assigned to each object to ease their identification. Each object created has also its own unique identification number which is stored in the SceneObjects table.

Two scenarios were deployed on the prototype architecture to show that the scenario logic is no longer embedded in the simulation environment and that it can be inserted and controlled by an outside component (the trainer using an interface). The first scenario aims at training new police officers on how to investigate and deal with the aftermath of a traffic accident. The second scenario is a virtual classroom which allows participants to join a single environment and communicate using voice. Figure 4 shows the two scenarios. Figure 5 shows one of the scenarios running on a Reflex set-up at the University of Sheffield.

The vehicle accident scenario was run on six subjects from Dubai Police who all had the same training background which consists of four years of police academic college and two specialized training courses in vehicle accidents investigation. The six subjects where chosen by the trainer. During the run of the experiment we noticed that all the trainees made good use of the navigation methods and managed to investigate the accident scene. They also made good use of the headset facility in identifying the drivers and communicating with the operation room operator. One of the trainers said that practical training has shown him clearly the trainees' weaknesses which were not obvious to him while conducting the theoretical training.

CONCLUSIONS

The separation of the domain knowledge and the ability to control it at run-time were the two primary objectives of the first prototype. The separation has been achieved by the use of a high-level language (Python) to create a layer on top of the simulation environment to act as an interface. This means that behaviours can be inserted and controlled at run-time. The abstraction achieved by the scripting language made controlling the application behaviour much easier and dynamically loadable at run-time. Moreover, using an existing language rather than building our own reduced the development time and eliminated the need to build a parser and evaluate it. The other advantage of this approach is that it allows easier commands to be added making the use of the simulation engine simpler.

Creating and running two different scenarios on the prototype simulation environment has demonstrated the flexibility of the approach since it managed to detach the domain knowledge from the simulation environment and allow its control at run-time. These results are promising. Proving full flexibility would require more tests to be carried out involving scenarios from different domains. The observation that can be made about the flexibility of the prototype is that the more the simulation environment functionality is exposed through the scripting language the more flexibility is achieved. This means if the full simulation environment functionality is exposed, then the system can be labeled fully flexible according to our description of flexibility, i.e. the ability to insert and control behaviours at run-time.

We have shown that our prototype architecture can handle the running of multiple environments while allowing a trainer to monitor the simulation and trigger events to create the desired scenario. The next stage will replace the trainer by the architecture proposed in Figure 1.

REFERENCES

Akerberg, O., Svensson, H., Schulz, B., Nugues, P. *CarSim: An Automatic 3D Text-to-Scene Conversion System Applied to Road Accident Reports*. Research Notes and Demonstrations Conference Companion, 10th Conference of the European Chapter of the Association of Computational Linguistics, 2003. http://citeseer.nj.nec.com/563862.html

Cruz-Neira, C., Bierbaum, A., Hartling, P., Meinert, K., Just, C. *VR Juggler – An Open Source Platform for Virtual Reality Applications*. AIAA 2002 Aerospace Science Conference, Reno, NV, January 2002.

Dachselt, R. *CONTIGRA Towards a Document-based Approach to 3D Components*. Workshop 'Structured Design of Virtual Environments and 3D-Components' at the ACM Web3D 2001 Symposium.

Davis, W., Moeller, G. *The High Level Architecture: is there a better way*. In Proceeding of the 1999 Winter Simulation Conference

Figure 5. Accident scenario running on Reflex Studio set-up.

Drew, R., Morris, D., Dew, P., Leigh, C.*A System Architecture For Supporting Event Based Interaction And Information Access*.http://citeseer.nj.nec.com/370116.html

Eugster, P., Felber, P., Guerraoui, R., Kermarrec, A. *The many faces of publish/subscribe*. In ACM Computing Surveys (CSUR), volume 35, issue 2 (June 2003), pages: 114 - 131

Hamman, M., Wilkins, D. C., Carbonari, R., Mueller, C. *DC-Train 4.0 Instructor's Manual*. Knowledge Systems Lab Report UIUC-BI-KBS-2001-0040. Beckman Institute, University of Illinois, Urbana-Champaign. November, 2001.

Hamman, M., LeMentec, J. C., Wilkins, D. C., *Design Requirements for DC-Train 4.0*. Knowledge Systems Lab Report UIUC-BI-KBS-2001-0029. Beckman Institute, University of Illinois, Urbana-Champaign. February 2001.

Hawkes, R., Wray, M. *"LivingSpace: A Living Worlds Implementation using an Event-based Architecture"*. HPL-98-181, Extended Enterprise Laboratory, 1998.

Hook, B. *The Secret Life Of Game Scripting*. Feb, 2004 http://bookofhook.com/Article/GameDevelopment/TheSecretLifeofGameScript.html

Romano, D.M. *Features that Enhance the Learning of Collaborative Decision Making Skills under Stress in Virtual Dynamic Environments*. Ph.D.thesis, Computer Based Learning, University of Leeds, UK, August 2001.

Shaw, C. Liang, J, Green, M. Sun, Y. *The Decoupled Simulation Model for Virtual Reality Systems. In* Human Factors in Computing Systems CHI'92 Conference Proceedings, pages 321-328, Monterey, California, May 1992. ACM SIGCHI.

Smith, R. *Essential techniques for military modeling and simulation*. Proceedings of the 30th conference on winter simulation, 1998, pages: 805 - 812 ISBN:0-7803-5134-7

Szarowicz, A., Forte, P., Amiguet-Vercher, J., Gelepithis, P. *Application of Autonomous Agents for Crowd Scene Generation*. 2nd Hellenic Conference on AI SETN-02, vol. 2 April 11-12, Thessaloniki, Greece, 2002

Tamberend, H. *Avocado: A Distributed Virtual Environment Framework*. Ph.D.thesis, University of Bielefeld, 2003.

Wang, Q. Green, M, Shaw, C. *EM – An Environment Manager for Building Networked Virtual Environments*. IEEE Virtual Reality Annual International Symposium (VRAIS 95)*, pages 11-18, Research Triangle Park, North Carolina, March 11-15, 1995, IEEE.*

Williams, R, J. *A Simulation Environment to Support Training for Large Scale Command and Control Tasks*. Ph.D. thesis, School of Computer Studies, University of Leeds, UK, December 1995.

Wright, I. P., Marshall, J. A. R. *RC++: a rule-based language for game AI*. In: Proceedings of the First International Conference on Intelligent Games and Simulation (GAME-ON 2000). SCS Europe BVBA

LOCALLY-ADAPTIVE VIRTUAL ENVIRONMENTS IN PERSISTENT-STATE MULTI-PLAYER GAMES

Marc Lanctot Clark Verbrugge
School of Computer Science
McGill University
Montréal, Canada, H3A 2A7
phone: 1-514-398-7071
fax: 1-514-398-3883
email: marc.lanctot@mail.mcgill.ca, clump@cs.mcgill.ca

KEYWORDS

Computer Games, Adaptation, Virtual Environment, Cellular Automata, Fuzzy Logic, Weather Modeling, Reputation System

ABSTRACT

We present a generic model for adaptation in large scale, persistent state computer games that allows the virtual world to change automatically, with reasonable efficiency. We demonstrate the utility of our technique through two different forms of dynamic common game content: 1) a basic weather cycle that adapts wind, rain and water accumulation to variations and changes in a large-scale terrain, and 2) a simple *reputation* system that allows agents in the virtual world to respond appropriately to a player's actual behaviour in a game.

INTRODUCTION

A basic problem encountered by vendors of large scale, persistent-state gaming environments is how to continuously improve and change the virtual environment so as to maintain player interest, and also reflect the activities of players in the virtual world. In a more generic sense this falls under *content creation* [Mellon, 2003], altering or adding new virtual content to the game. Manual approaches are typically used due to the creative requirements of general content creation and the complexity of determining realistic adaptation results, but impose extra game maintenance costs and administration requirements. Automatic approaches that sensibly alter and tune the game world with minimal human intervention are thus desirable.

We present a generic model for adaptation in computer games that allows the virtual world to change automatically, with reasonable efficiency. We demonstrate the utility of our technique through two different forms of dynamic common game content: 1) a basic weather cycle that adapts wind, rain and water accumulation to variations and changes in a large-scale terrain, and 2) a simple *reputation* system that allows agents in the virtual world to respond appropriately to a player's actual behaviour in a game.

Specific contributions of this work include:

- Design of a general adaptation framework suitable for modeling flow-based properties in game simulations. Our approach is based on cellular automata, ensuring only local information is required at each computation; this allows for reasonable scalability in distributed environments.
- Design and experimental verification of systems for two forms of popular, dynamic game content. We describe a simple, aesthetic and logically consistent adaptive weather model for game worlds, and a game reputation system that can dynamically respond to changing patterns of information dispersal and player behaviour.

Related Work

Virtual environments are of course well studied, and a variety of approaches already exist, though most efficiency and representation improvements concentrate on network and communication optimization [Macedonia et al., 1994] rather than adaptation of the environment itself. Even environments such as Bamboo that allow flexible modification to the environment through hot-pluggable modules [Watsen and Zyda, 1998] do not allow for incremental adaptation of game content.

Adaptation is a traditional target of A.I. research. Here a *virtual environment* is often separated into 2 major components: the static context, and the dynamic agents [Russell and Norvig, 2002]. In the context of computer games, adaptation has been investigated [Spronk et al., 2003], though like most other applications of A.I. [Carmel and Markovitch, 1998] it has been primarily directed at adapting agents (NPCs, game opponents) rather than the environment. Even non-constant, fluctuating environments are usually viewed as the process to react to, rather than the target of adaptation [Haynes and Wainwright, 1995]. Our motivations

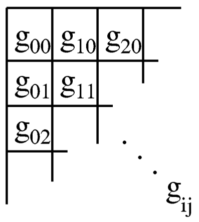


Figure 1: The virtual terrain.

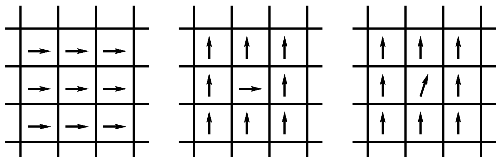


Figure 2: Affect of an update on one grid section (assuming $\gamma = 1$), showing a) before the change b) before the update on the middle grid section c) after the change and update

more closely resemble building an *artificial model* as in done in ALife [Steels, 1994]; we, however, focus on constructing an adaptive environment irrespective of adaptivity of the agents.

AN ABSTRACT MODEL FOR ADAPTATION

Our model is based on a finite 2-dimensional space, the virtual terrain. We assume the space is partitioned into a discrete mapping or *grid*. In the examples below we use the familiar situation of a subset of $\Re^2$ and a square grid G, as illustrated in Figure 1. In general the techniques we use apply equally well to any discrete *metric space* [Burago et al., 2001].

One can imagine that the elements of the grid, or *grid sections* have *properties* depending on the context of the system in which the model is used. The adaptation process aims to modify these properties based on the impact of events that occur in the surrounding sections. The procedure used is similar to the procedures associated with cellular automata: iterative update rules are applied based on properties of neighbouring cells [Gardner, 1970].

A simple example of an application of local property updates is *blurring* or *spatial low-pass/box filtering* in the field of image processing [Baxes, 1994]. Each pixel $p_{x,y}$ in an image has a scalar *intensity* property, $I(p_{x,y})$, and a neighbourhood of nearby pixels $N(p_{x,y})$. To create a blurred image, a new intensity for each point is defined:

$$p'_{x,y} = \frac{I(p_{x,y}) + \sum_{p \in N(p_{x,y})} I(p)}{|N(p_{x,y})| + 1}$$

and a simultaneous update rule is applied: $\forall x, y : p_{x,y} \leftarrow p'_{x,y}$. A good demonstration of the effects of this can be found at [Fisher et al., 2003].

Vector Averaging and Angular Propagation

Grid properties in cellular automata are typically scalar properties. SimCity, for example, is a classic game that relies on cellular automata techniques [Stanford, 1996], associating scalar quantities with grid cells. In SimCity each grid cell may have quantities such as pollution levels, crime rates, land value, and so on. There is, however, no reason to restrict grid properties to scalar values. We define a *discrete vector field* as $V_G : G \longrightarrow \Re^2$, so that for each grid section $g \in G$, there exists an associated vector, $V_G(g)$. The vector at cell $(3, 2)$ will be denoted $\vec{g}_{3,2}$. Note that a 2-dimensional vector can be thought of as a magnitude and angle; when we are interested in just one component of the vector we can reduce it to the scalar case; e.g., a simple angle value $\theta_{3,2}$.

Our technique is analogous to image blurring on vector components. We initially ignore the vector's magnitude and assume it does not change. Each grid section's vector is then modified to have a new angle computed as a weighted average of its own state and neighbouring angles. Suppose we have an average angle $\overline{\theta_g}$ for a grid section g and its neighbourhood. We define a *shift from* $\vec{g'}$ for each neighbour g' as the difference $\overline{\theta_g} - \theta_{g'}$. The total angular change is then some proportion of this shift, for example $\delta_g = \gamma \cdot$ *shift from*$(\vec{g'}, ...)$. The update rule then becomes: $\forall g \in G : \theta_g \leftarrow \theta_g + \delta_g$, applied simultaneously over all grid sections.

To demonstrate this, consider a single grid section surrounded by its *8-neighbourhood*, all of its vectors pointing eastward ($\theta = 0$) with arbitrary magnitude, as seen in Figure 2. Now, if we shift each surrounding vector by 90°, the average will shift by $\Delta\theta = (8/9) * 90°$, so the update will shift the middle vector's angle by $\delta = \gamma\Delta\theta$. Since the middle vector has shifted, upon the next application of the update (the next iteration) it will in turn cause a difference in average of all points for which it is a neighbour. This will cause those grid sections' vectors to update, and so on. As a result, a change in angle propagates through the grid via its neighbouring cells, but loses influence each iteration. By adjusting weights such as γ local turbulence can be damped according to the needs of the system being modeled.

Flow-based Fuzzy Property Update Rules

Non-constant scalar properties on grid sections are modified differently than the vector properties. The vectors on each grid section describe a strength and direction of *flow*. Here, the flow function is defined over a scalar property and computes how much of the property is transferred from a grid cell to the cells in its neighbourhood as a result of the vector property.

We use a *fuzzy* approach [McCuskey, 2000] to computing flow for a more natural flow dispersal. Formally, the flow function consists of n fuzzy components: $z_1, z_2, \cdots, z_n$. Here, z_j is an arbitrary fuzzy membership function $z_x(\vec{g}) \in [0,1]$ which represents the raw influence of that component over a given property. The influences of the components are then scaled so that they represent the local influence in comparison to other influences:

$$f_x(\vec{g}_{i,j}, p_{i,j}) = \frac{z_x(\vec{g}_{i,j}, p_{i,j})}{\sum_{y=1}^{n} z_y(\vec{g}_{i,j}, p_{i,j})}$$

To make this more clear, consider a scenario where the components are associated with the four major cardinal directions: z_N, z_E, z_S, z_W. The amount transfered in each direction is proportional to the corresponding flow influence value f_{dir}. At each iteration, $\Delta p_W = k_p * f_W(\vec{g}_{i,j}, p_{i,j}) * p_{i,j}$ is the amount of $p_{i,j}$ that's displaced westwards, where $0 < k_p <= 1$ is the *rate of transfer*. The simultaneous update rules for this component would then be: $R_1 : p_{i-1,j} \leftarrow p_{i-1,j} + \Delta p_W$ and $R_2 : p_{i,} \leftarrow p_{i,j} - \Delta p_W$. Components for other directions are treated similarly. Note that it is also possible to define hybrid components, formed by the conjunction or disjunction of the fuzzy properties; e.g., $z_{NW} = z_N$ AND z_W. Then the displacement of moisture would be listed as a rule set in a fuzzy controller system as is done in [McCuskey, 2000].

The actual behaviour of the flow depends on the membership functions used; if a system demands a smooth flow, then naturally the membership functions should reflect that. The role of the fuzzy membership functions are to shape the flow. If, for instance we use a "crisp" function, one with a sharply-defined peak such as:

$$z_N = \begin{cases} 1 & \text{if } \pi/2 - \epsilon <= \theta <= \pi/2 + \epsilon; \\ 0 & \text{otherwise.} \end{cases}$$

for small ϵ, then the westward flow will move somewhat discretely. A smoother function like:

$$z_N = \frac{4}{\pi}\sqrt{(\frac{\pi}{4})^2 - (x - \frac{\pi}{2})^2}$$

will lead to a smoother spreading. This will become more clear in the example systems that follow.

EXAMPLE SYSTEMS

Here we describe two systems for adapting game content based on our general model. These provide experimental evidence of the generality of our approach, and are also novel forms of content generation in themselves. Experimental results for these systems are described in the following section.

An Adaptive Weather System

Weather simulation is typically considered a computationally intensive application, largely reserved for supercomputers. In the virtual worlds of computer games, however, physical accuracy is less critical, and much simpler approaches suffice to produce aesthetic, in-game climate effects.

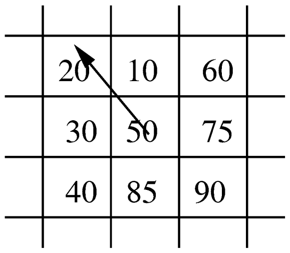


Figure 3: Example gradient vector calculation. Grid cells show local terrain altitudes.

In its simplest form, a weather cycle displaces moisture: water from lakes and seas is carried by wind to cooler locations, where the reduced water capacity of cooler air causes condensation; rain water eventually runs downhill to refill lakes and oceans [Enterprise, 2004]. There are several factors that can affect this process, including altitude and terrain structure, wind, temperature, and so on. We have based our weather system upon the following basic precepts:

1. Wind gathers moisture from bodies of water, and loses water at higher altitudes (cooler temperatures).
2. Water flows downstream.
3. Altitude affects wind patterns.

These basic rules will be transformed into update rules following the model outlined in the previous section. We must however first define another vector property used to describe how water flows downstream.

The *gradient vector* on a grid section is a vector sum composed of differences in scalar properties of surrounding cells. The magnitudes of the vectors are determined by subtracting the terrain *altitude* from the altitude of a neighbouring cell, with corners of the 8-neighbourhood having a weight factor or $\sqrt{2}/2$. The direction of each vector in the sum is given by the position of the neighbour relative to the center. Figure 3 shows an example gradient calculation: $g = 108.64\hat{W} + 120.35\hat{N}$ giving a vector with angle $tan^{-1}(120.35/108.64) = 48°$ north of west.

The gradient vector points to the direction of descent, and its magnitude represents the steepness of the grade. Moisture then flows downstream in the direction of the gradient using fuzzy flow-update rules as described in a previous section.

The inverse gradient points in the direction of ascent, and is used to determine how wind direction is altered by the current terrain. If a gust of wind is pointing

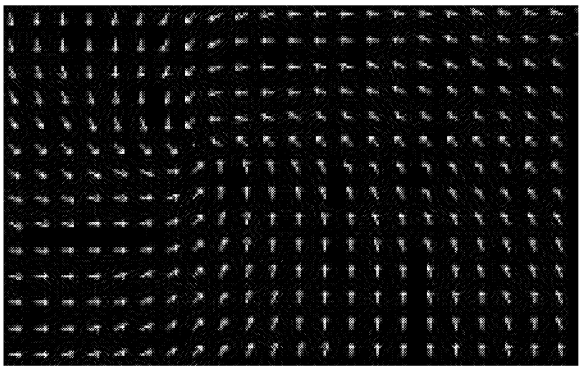


Figure 4: An example tornado.

into a wall, it will instead blow around it. For wind to move around high-altitude obstacles it must therefore be pushed away from the direction of the inverse gradient or, equivalently, towards the direction of the gradient. We also incorporate an inertial factor, to give a smoother flow pattern; we designate the average of the gradient vector and current average wind vector as the vector that will be approached as described in a previous section.

Moisture is displaced by the four independent components that comprise the wind, represented by the cardinal directions: $\vec{g}_N, \vec{g}_E, \vec{g}_S, \vec{g}_W$. The value of each component is calculated by a fuzzy membership function. These values are then normalized, and represent a proportion of the amount of moisture displaced to surrounding grid sections, again as per the fuzzy flow-update method. Similarly, water flow downstream is simulated as a displacement of water downstream proportional to the fuzzy memberships of the gradient.

Interesting weather *events* are also possible. Events can be anything that affects the properties in the system such as earthquakes, tornadoes, tsunamis, etc. We have modeled "tornadoes" as local, non-linear dynamical systems with the stable fixed points at the center. We designed a 2-dimensional dynamical system within a specified sub-grid such that a given point is fixed point in a stable spiral [Strogatz, 2001]. The tornado moves by slightly displacing the fixed point at each iteration and reconstructing the dynamical system around it. The wind in the surrounding cells then adapts to this turbulence using the vector propagation rules found in a previous section. Figure 4 shows a static screenshot of a tornado on a flat terrain.

An Adaptive Reputation System

A player's in-game reputation is often an important component of the game environment, particularly for simulation and role playing games. Player actions that harm or help computer-controlled characters should result in a logically consistent reaction to the player, giving a sense of reality to the game environment. This is necessarily a dynamic property—player reputations need to be constantly updated, and should also ameliorate over time and distance.

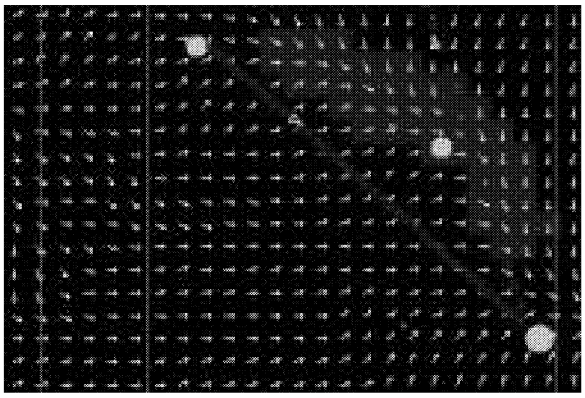
There have been some commercial attempts at creating a general, flexible reputation system, but results have been disappointing [Brockington, 2003]. Our approach was inspired by the *Dungeons & Dragons* reputation system [Collins et al., 2004], which states that as a player progresses his or her reputation will rise by performing "heroic deeds." Symmetrically, of course there should also be the inverse property, to degrade reputation by performing negative actions.

In order to allow reputation to more realistically disperse, we employ a word-of-mouth model to flow reputation events. A game character's reputation is built by the spread of hearsay amongst the populace; flow vectors modeling the communication patterns of the general populace in each grid section are used to describe the direction in which word of a positive or negative action will spread. For our example system we developed a virtual communication terrain which, as in the weather example, is represented as a discrete, adapting vector field. The difference is that the vectors on this vector field do not change with respect to a static value such as the gradient. These vectors are bent towards the sum of agents' velocity vectors currently occupying the corresponding grid cell and averaged out to the surrounding cells as explained in the section on vector averaging and propagation. The same wind model used in our weather system then traces out the flow of reputation information. Trade route popularity and connections in a commercial game could be derived from the relative movements of other player characters. In our case we simulated route popularity by tracking movements of semi-randomized agents moving between cities following smooth curved paths choosing destinations probabilistically based on distance and city size to discover trade routes. Figure 5b shows an example of communication terrain constructed in this manner.

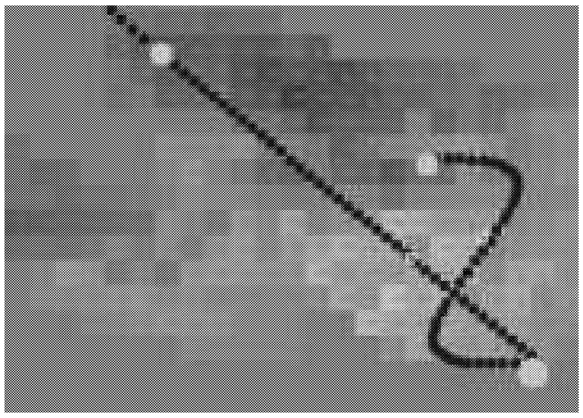
Positive and negative *reputation points* are created on a grid section when an event occurs that would affect someone's reputation: rescuing the princess, killing a commoner, stealing from tavern, *etc.,* and are proportional to the severity of the event. These points are displaced via the flow, and also dissipate at a slow rate. For each point that dissipates on a grid section, the reputation of the player is altered at that location. This process repeats until all the reputation points have dissipated, causing a local alteration in the player's reputation.

EXPERIMENTS AND ANALYSIS OF DATA

We conducted tests on both systems, to assess aesthetic quality, and also to quantitatively ensure the systems were stable and efficient; these are described in Table 1. The 3rd column describes the test data, either



a)



b)

Figure 5: Screenshots of a Reputation test in action. The lines are the paths of the agents, the white triangles are the agents, and the circles are the cities which agents travel to and from. In a) the blue (lighter) smudges are the reputation points spreading. b) shows the communication terrain.

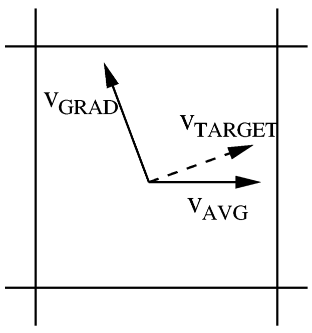


Figure 6: Example of obtaining $\vec{v}_{TARGET}$ when $g = 0.8$.

Test	Type (threads)	Map	GUI	System
1	Weather	N. Korea	yes	UP
2	Weather	Pakistan	yes	UP
3	Weather	Pakistan	no	UP
4	Weather	Pakistan	no	MP
5	Weather (2)	Pakistan	no	MP
6	Weather (4)	Pakistan	no	MP
7	Reputation	62 x 50	no	UP
8	Tests 3 and 7 combined			UP

Table 1: Testing scenarios.

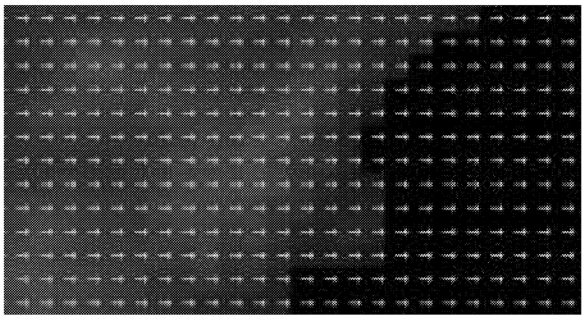
an altitude map of North Korea (48x40), or of Pakistan (62x50), obtained by downsizing more detailed maps from [Rojas et al., 2003]. The fourth column indicates whether the graphical user interface of our testing was present. Some tests were performed on a uniprocessor Pentium 4 1.70Ghz with 528M RAM (*UP* in column 5). Other tests designed to show scalability launch multiple computation threads, each assigned responsibility for a portion of the grid. Every iteration grid calculations are done independently and concurrently by these threads, and synchronized for the simultaneous update. These were performed on a quad-processor AMD Opteron 844 (1.80Ghz) with 3.6G RAM (*MP* in column 5). Every test lasted at most 10000 iterations.

The Weather tests are composed of tests 1–5. In each case all the wind vectors start pointing eastwards with a magnitude of 50, as shown in Figure 7a. This is an initial state far from any final state (Figure 7b), and so represents an extremal test for adaptation. We also included a tuning parameter, an *average bias value* $g = 0.9$. When calculating the vector to approach in the gradient-bending method of the vector averaging and propagation section this constant places bias on either the gradient ($0 <= g < 0.5$), or the current average ($0.5 < g <= 1$). The weighted average of these is the vector that is approached:

$$\vec{v}_{TARGET} = g \cdot \vec{v}_{AVG} + (1 - g)\vec{v}_{GRAD}$$

The update rule then becomes a rule that approaches the target: $R_1 : \theta_{i,j} \leftarrow \theta_{i,j} + \gamma(\theta_{TARGET} - \theta_{i,j})$, as seen in Figure 6. This results in a smoother, if slower adaptation.

The Reputation tests were made up of simulations of agents moving from town to town, as described in a previous section. The probability that an agent would travel from one town to another at any given iteration was set to 0.9. The source city was chosen randomly, weighted by the size of the city (larger cities produce more trade, and hence more communication sources), and the destination city was chosen based on a combined weight determined by city size divided by the square of relative distance (short trips are more common). Reputation events would normally come from player actions; in our tests they are manually generated during testing.


a)

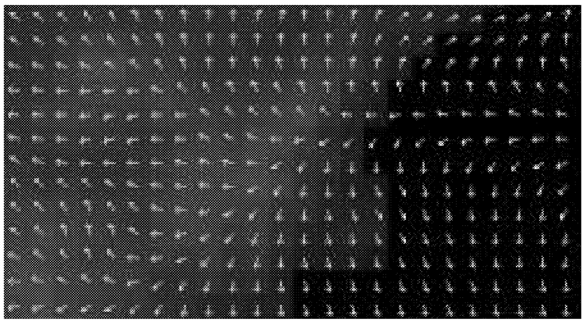

b)

Figure 7: Screenshots of a) before the Weather Test 1 b) end of Weather Test 1.

T_n	$\bar{t}_i(ms)$	$\bar{t}_{i_1}(ms)$	$\bar{t}_{i_2}(ms)$	$\bar{t}_{i_3}(ms)$
1	43.91	29.81	9.6	4.5
2	92.97	64.37	21.5	7.1
3	64.22	44.68	14.34	5.2
4	46.32	28.61	11.61	6.1
5	28.85	16.41	8.3	4.14
6	23.41	12.56	7.5	3.35
7	44.68	34.52	10.12	0.02
8	104.5	74.4	24.74	5.35

Table 2: Performance results for the tests.

Performance Results

Experimental results are summarized in Table 2. In all cases, $\bar{t}_i$ represents the average simulation time per iteration of calculations. In the Weather tests $\bar{t}_{i_1}$, $\bar{t}_{i_2}$, $\bar{t}_{i_3}$ represent the average time spent on the gradient-bending and averaging, moisture spreading, and rain displacement, respectively. In the Reputation tests, these times represents the vector-averaging, reputation point spreading, and the vector-bending due to agents' influence, respectively. In the combined test, these times (except $\bar{t}_i$) are simply added together.

Results are encouraging. Tests 1 and 2 show a linear increase in average iteration cost with respect to grid size: the Pakistan map is 61% larger than the North Korea map, and average iteration time is 58% larger.

Scalability is shown in tests 4, 5, and 6. Average iteration cost drops by a factor of 1.98 using 4 threads for computation. Although this is far from ideal, these results still show that parallel computation is applicable here.

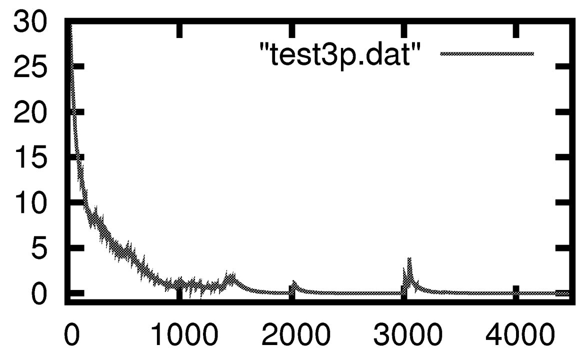


Figure 8: Δ_{mask}(radians) vs. iteration number in Weather test 3.

The value of casting these problems into a generic framework is evident in test 8 in comparison to 3 and 7. By combining the tests we allow not only a shared implementation, but also sharing of resources; this costs somewhat less than doing both tests separately, even though the tests do not use clever collaborative methods. We expect future additions to behave similarly.

Convergence Properties

Since our approach uses an iterative update strategy it is important to know how long it may take for a calculation to converge to appealing and stable results. Convergence can be guaranteed in the finite domain of computer-representable real numbers by ensuring our flow and gradient calculations are *monotonic* functions. This is unfortunately not easy to ensure given the use of complex, stateful (history-sensitive) functions in game simulation. We thus investigate convergence experimentally for our example systems.

Convergence in the Reputation test is not meaningful. Agent behaviour is deliberately non-deterministic in order to produce a complex communication terrain, and actual flow of reputation points is damped, and so trivially converges from any constant number of reputation events.

In the Weather case, we are trying to adapt a quantity (wind) within the system to its surroundings. To measure how this quantity is changing we add up all the proposed change in angles over all grid cells at every iteration and call this the Δ_{mask}; the results for test 3 are shown in Figure 8. In most cases it takes less than 5000 iterations to converge to the point where Δ_{mask} is consistently extremely small ($< 10^{-12}$) or reach a periodic cycle. Note that for certain stability it is required that

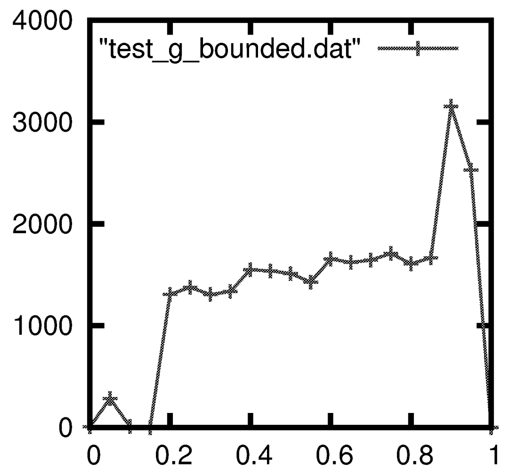


Figure 9: Iterations until convergence vs. g graph, with $\gamma = 0.1$.

we restrict the grid so that neighbours on the grid cell extremes are only those cells closer to the center and not “wrapped around” to the other extreme of the grid, forcing a bounded topology unlike the torus. We are currently studying the causes of instability in the torus topology. Less converged, but quite acceptable results are however achieved much sooner, typically by within 1000 iterations. In Figure 8 random perturbations to terrain altitude are performed on the grid at iterations 2000 and 3000; these small adaptations converge very quickly, in just a couple hundred iterations.

The bias parameter has a direct influence on the time it takes to converge. Figure 9 shows the number of iterations before convergence (variation less than 10^{-12}) for a selection of g values. Convergence in general behaves non-linearly. This is not surprising given the complexity of our system, but it does indicate the importance of experimental validation in the implementation. Note that our examination of slow and non-convergence shows that when it occurs it is typically expressed visually as continuous change localized to just one or two very small areas; the appearance is still overall quite good, and in a continuously adapting situation is difficult to discern.

CONCLUSION AND FUTURE WORK

Environmental adaptation in persistent-state games is a relatively unresearched area. We have presented a general framework to describe such systems that is generic and efficient. We have shown that our approach is suitable for adaptive content management and creation. In order to do so we defined implementations within our framework for two very different forms of content, basic weather simulation, and an in-game player reputation system. Both systems respond to dynamic changes in input data, and we have experimentally demonstrated the feasibility of the technique, both in terms of performance and the ability to converge to good, aesthetic results.

Our approach is of course largely a proof of concept demonstration. The practical value of our technique would be best measured in the context of a commercial, large scale, multiplayer game, though research access to such an environment is difficult to acquire. Even in the context of a prototype, however, performance improvements can be realized; our implementation was designed for easy exploration of ideas more than performance, and a number of optimizations are possible. We’ve proved that the framework is scalable in two different ways: computationally by using concurrency, and content-wise by efficient re-usage of resources for combined systems. An expansion of useful, flow-based content is also of value, and we are investigating the expression of such concepts as adaptive economies, politics, and law in our system.

ACKNOWLEDGMENTS

This research has been supported by Le Fonds Québécois de la Recherche sur la Nature et les Technologies and the McGill Faculty of Graduate Studies.

REFERENCES

[Baxes, 1994] Baxes, G. A. (1994). *Digital Image Processing.* John Wiley & Sons.

[Brockington, 2003] Brockington, M. (2003). Building a reputation system: Hatred, forgiveness, and surrender in Neverwinter Nights. In Alexander, T., editor, *Massively Multiplayer Game Development*, pages 454–563. Charles River Media.

[Burago et al., 2001] Burago, D., Burago, Y., and Ivanov, S. (2001). *A Course in Metric Geometry.* American Mathematical Society.

[Carmel and Markovitch, 1998] Carmel, D. and Markovitch, S. (1998). Model-based learning of interaction strategies in multiagent systems. *Journal of Experimental and Theoretical Artificial Intelligence*, 10(3):309–332.

[Collins et al., 2004] Collins, A., Decker, J., Noonan, D., and Redman, R. (2004). *Unearthed Arcana.* Wizards of the Coast.

[Enterprise, 2004] Enterprise, N. E. S. (2004). GSFC earth science enterprise water and energy cycle. `http://gwec.gsfc.nasa.gov`.

[Fisher et al., 2003] Fisher, R., Perkins, S., Walker, A., and Wolfart, E. (2003). Mean filter. `http://homepages.inf.ed.ac.uk/rbf/HIPR2/mean.htm`.

[Gardner, 1970] Gardner, M. (1970). The fantastic combinations of John Conway's new solitaire game "life". *Scientific American*, pages 120–123.

[Haynes and Wainwright, 1995] Haynes, T. D. and Wainwright, R. L. (1995). A simulation of adaptive agents in hostile environment. In George, K. M., Carroll, J. H., Deaton, E., Oppenheim, D., and Hightower, J., editors, *Proceedings of the 1995 ACM Symposium on Applied Computing*, pages 318–323, Nashville, USA. ACM Press.

[Macedonia et al., 1994] Macedonia, M. R., Zyda, M. J., Pratt, D. R., Barham, P. T., and Zeswitz, S. (1994). NPSNET: A network software architecture for large-scale virtual environment. *Presence*, 3(4):265–287.

[McCuskey, 2000] McCuskey, M. (2000). Fuzzy logic for video games. In DeLoura, M., editor, *Game Programming Gems*, pages 319–329. Charles River Media.

[Mellon, 2003] Mellon, L. (2003). Research opportunities in game development. Tutorial at: PADS'03 Workshop on Parallel and Distributed Simulation.

[Rojas et al., 2003] Rojas, E., Hijmans, R. J., and Guarino, L. (2003). DIVA-GIS. `http://diva.riu.cip.cgiar.org`.

[Russell and Norvig, 2002] Russell, S. and Norvig, P. (2002). *Artificial Intelligence*. Prentice-Hall, 2nd edition.

[Spronk et al., 2003] Spronk, P., Sprinkhuizen-Kuyper, I., and Postma, E. (2003). Online adaptation of game opponent AI in simulation and in practice. In *Proceedings of the 4th International Conference on Intelligent Games and Simulation (GAME-ON 2003)*, pages 93–100.

[Stanford, 1996] Stanford, M. I. (1996). Uses and subversions of SimCity 2000.

[Steels, 1994] Steels, L. (1994). The artificial life roots of artificial intelligence.

[Strogatz, 2001] Strogatz, S. H. (2001). *Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry and Engineering*. Perseus Books Group.

[Watsen and Zyda, 1998] Watsen, K. and Zyda, M. (1998). Bamboo: A portable system for dynamically extensible, real-time, networked, virtual environments. In *IEEE Virtual Reality Annual International Symposium (VRAIS'98)*, pages 252–259, Atlanta, Georgia.

Exploring Game Space – From Mobile Gaming to Location-Based Mixed-Reality Entertainment

Volker Paelke
University of Hannover, IKG
Appelstraße 9a
30167 Hannover, Germany
E-mail: Volker.Paelke@ikg.uni-hannover.de

Christian Reimann, Markus Koch
University of Paderborn, C-LAB
Fürstenallee 11
33102 Paderborn, Germany
E-mail: [reimann; mkoch]@c-lab.de

KEYWORDS
HCI, Mobile Computing, Computer Vision

ABSTRACT

Emerging technologies for positioning and context recognition together with advances in mobile computing and wireless communication technology enable the creation of new styles of games that exploit the mobile context of the user. While spatial aspects and the task of "navigation" have been present in computer games from the beginning this has mostly been in the form of simulation on a fixed display. This paper presents an exploratory study on the use of real world geographic environments for and within games.

INTRODUCTION

As networked mobile devices are becoming pervasive location based services are seen as promising applications for 3G networks like UMTS. Mobile games are one area in which the first practical location based applications are introduced because they are seen as one possible market for early adopters in which the costs can be recuperated despite the technological limitations of current devices and networks as witnessed by applications like SingTel's "Gunslingers", or "Can you see me now" by Blast Theory (Blast Theory (2003, 2004)). Other interesting developments in which novel gaming and interaction concepts are driven by the widespread availability of novel technologies are the use of geographic environments for gaming, e.g. ARQuake (Piekarski, Wayne and Thomas, Bruce, (2002)) and the development of the Geo-Caching community (Geocaching (2004)) and the incorporation of motion in physic space as a means of interaction in the Sony EyeToys games, Siemens Mozzies on the SX1 smartphone and the AR-Soccer application (Reimann, C., Paelke, V., Stichling, D. (2003)).

These developments show that location sensing and video recognition not only enable the exploitation of spatial aspects and geographic environments in existing game concepts but also provide the basis for new gaming concepts if they are combined.

In this paper we explore the resulting design space for games, focusing on experiments with the integrated use of real-world geographic environments in mobile games, using the MobEE (MobileEntertainmentEngine) system.

BACKGROUND

The creation of games with appealing graphics and game-play on a variety of mobile devices ranging from smart phones to PDAs and other portable and wearable computing devices poses a number of challenges. Since entertainment applications are targeted at a diverse user population that will often employ them without previous training or reference to a manual a highly usable interface design is as critical for their success as the narrative content itself.
The mobile context of use imposes additional requirements on entertainment applications: Typically, mobile applications are used for short episodes, possibly as one task among many. Game designers have to ensure that users can enjoy a pleasurable interaction experience under these circumstances that differ significantly from the use of game consoles or PCs.
A common solution - seen in many current mobile games - is to create games that can be completed within a few minutes. It is, however, obvious that there is also interest in mobile games that enable deeper game-playing experiences over an extended period of time, similar to most current console and PC games. As the level of attention that a user can devote to the application may be limited (e.g. when using the device in parallel to other activities) or user interaction may be interrupted by the user's need to attend to some external events in a real-world environment it is necessary to provide appropriate guidance information to enable use without high cognitive effort as well as sufficient status information to enable users to resume their activities after a break. The creation of games with these features requires a platform that provides suitable mechanisms to support intermittent use over an extended period of time and appropriate means to support the creation of complex game content.
Additional limitations and constraints are imposed by the display and interaction capabilities of mobile devices: The visual presentation is typically constraint by the limited resolution of mobile graphic displays (e.g. typical resolutions range from 100*80 pixels for mobile phones to 240*320 for PDAs - compared to mega-pixel displays in desktop environments), the small display size of mobile devices, the limited number of available colors and the limited graphics processing power available. One possible approach that we examine with our system is to use game content representations in multiple media that include the real-world environment as part of the game content in a mixed-reality setup. Using this approach the limited resources of mobile devices can be focused on the game

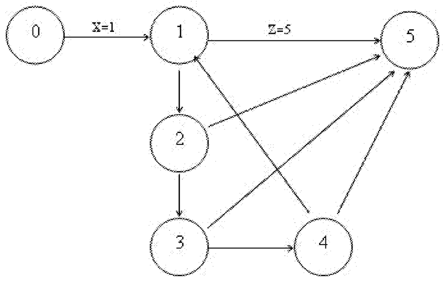
content while the environment provides high-fidelity context for the player.
Similarly, interaction on mobile devices is constraint by the available input modalities. While the use of novel input modalities like video and speech (e.g. ARSoccer in Reimann, C., Paelke, V., Stichling, D. (2003)) can help to reduce the resulting difficulties there are also alternative possibilities that we explore with the MobEE system, namely the use of real world user locomotion and actions as input to the game itself, turning the world around the user into his "game board".

MobEE, THE MOBILE ENTERTAINMENT COMPUTING ENGINE

A game-engine for mobile entertainment computing must address various requirements to be useful for application developers. In addition to support for the creation of the narrative application content the game-engine should also support the run-time presentation of this content on a wide variety of mobile hardware platforms, enabling users to switch between the use of different devices.

Device-Independent Story Structures

To enable the use of an entertainment application across a variety of hardware platforms the story structure that captures the narrative content must be kept is a device independent format. By separating the story structure from its representation, users can be presented with the same storyline and information regardless of the device that they are using. Only the presentation format/media are adapted to the capabilities of the interaction device.
The device-independent representation of the story structure in MobEE is implemented by finite automatons controlled by a variable pool. Each hot-spot accessible by the user changes variables and thereby the states of the automatons. Each hot-spot and automaton-state-change is linked to narrative information such as text, graphics, sound or animation which is displayed to the user upon activation.
Initial experiments with a story structure representation in a single automaton containing all possible states of the storyline proved to be unusable as it was nearly impossible to correct mistakes or adjust the story content or game-play. Even for short and simple stories very large and complex automatons would be required. Additions to the storyline are even harder to handle in a monolithic representation by a single automaton, since every action in the new story segment could affect parts of the old story, resulting in exponential growth of the number states and transitions. The partitioning of a story line into smaller parts that are handled by simple automatons reduces complexity significantly.
Therefore, in the approach that we have adopted for MobEE the storyline is created by an arbitrary number of small automatons, each representing a small segment of the story. Using this segmentation it is easy to debug mistakes made in the game design and to add side-tracks to the story in later design phases. Activating a hot-spot can take effect on all other hot-spots in terms of design, sound and information they provide. By carefully designing the automatons it is possible to represent almost any complex storyline in a format accessible to the MobEE game engine and change it easily afterwards.



Variable Pool	
X	1
Y	5
Z	3

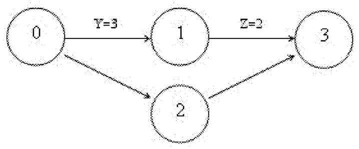


Figure 1: Two automatons communicating over a pool of shared variables

Exchangeable Representations: Text, 2D Graphics, Augmented Reality

Because there are some many different types of displays on mobile devices and an even greater variety of processing power the game engine has to consider all of these differences and take them into account by offering various types of representation of the entertainment application.
A simple output of text can be realized on almost every mobile device regardless of display and processing speed, but for many entertainment applications a text only presentation is not sufficient anymore. Early text-adventures were surely entertaining at their time but in today's world only few users can be entertained by an application based on text-output.
However, because some mobile devices do not provide other means of output text representation is supported by MobEE. Text only output can also be useful on more capable devices in certain situations and is also essential to ensure accessibility for visually impaired users.
Nevertheless, 2D Graphics offers a more attractive representation and is the more suitable presentation modality in many entertainment computing scenarios: The processing power required for 2D graphics is not too high for today's mobile devices and the creatable entertainment for the user is substantial as there are many means of representing a story in 2D graphics: Scrolling images, sprites, 2D animations and even small video clips are all possible.
In its current implementation MobEE supports sprites, scrolling and animations for the 2D representation of entertainment applications. The use of video clips is

limited by the memory capabilities of current mobile devices that are not sufficient to provide enough room for space-consuming videos. Server based video streams could provide an alternative solution but are not yet implemented in the MobEE engine.
3D animation as the state-of-the-art representation for entertainment computing is becoming more interesting for mobile computing as acceleration hardware finds its way into mobile devices. In experiments with current hardware we have found that the limited processing power available on these devices makes the creation of complex 3D representation impracticable. While a server based stream of 3D rendered images could provide a solution we have decided to experiment with a mixed-reality setup to resolve these problems.
In our setup certain real-world locations are linked to game content. If the user is at such a location he can use the camera of his mobile device to capture an image of his surroundings that is then augmented with the graphical game content. The gameplay in mixed-reality mode is similar to that in normal mode as described before. While navigating the user is presented with a map onto which icons representing the "game locations" are added if the user has explored the corresponding part of the game. At a "game-location" the user can interact with the real-environment using the camera on the PDA.
To track the users position in the real world while he is walking around, we use a GPS-sensor (Holux GR-230), which is wirelessly connected to the PDA via Bluetooth. The GPS-sensor sends every second the actual positioning information to the PDA using the NMEA-protocol. To integrate the real-world navigation into the game-engine, an additional navigation-automaton was integrated, which reads the Nmea-data from the GPS-sensor and updates the game with the received information. One problem was the low update-rate of the GPS (1 sec). To overcome this, the navigation automaton updates the position information every 333 milliseconds extrapolating two in-between positions from the last measurements, taking not only the speed and direction but also the acceleration from the last three waypoints into account. At the "game-locations" or hotspots the user interacts with the game more intensively than just navigating. Here he meets NPCs (Non-Player-Characters), solves riddles, fight dragons and so on. So the "game-locations" are used to continue the narration of the game. For such an interaction the data from the GPS is not suitable, as it is not accurate enough and does only provide a very loose tracking of the user's orientation. Therefore we use ARToolKit, a computer-vision fiducial based tracking system for AR-applications. Due to the very limited processor power of a PDA we decided not to use interactive augmented reality, but what could be called "Snapshot-AR": The user takes a single picture with the PDA's camera, which is then analyzed and taken as a static background for the rendering. Since only the augmentation graphics have to be rendered the impact of the hardware constraints are reduced since the user has high-fidelity context information from his real-surroundings.

Context Refresh

As mentioned in the introduction, developers of mobile entertainment applications have to take into account different levels of attention of the user. Unlike on desktop environments the use of mobile devices is often limited to short time intervals (e.g. while waiting). Because there may be longer periods between the uses of a mobile entertainment application it may prove necessary to provide users with a context refresh of his recent activities in the application.
Such a context refresh is realized in MobEE by keeping track of all the users' actions in a file. By doing this it would be easy to show the user all his activities up to the point where he wants to continue the game or application. Since a complete history of all his actions every time he starts the program would be boring or even annoying the context refresh has to select significant actions and display them as the user's memory would do. Unimportant detail and older actions that are no longer relevant can be left out of the context refresh for the user, focusing only on important and more recent details.
Since the application has no way of knowing which detail is important and which one isn't the developer has to provide this information when developing the game or application by applying values to key information that the application might show to the user. Based on this additional information the MobEE engine is able to sort out outdated information simply by looking at the number of facts provided to the user since certain information has been passed and using the importance of the information to calculate the necessity of still showing this fact to the user in the context refresh.
If the application is using storytelling techniques to provide information for the user and the narration is using present time for its presentation it is necessary for the developer to provide the application with the same information placed in simple past for the re-narration.
In respect of the varying intervals between usages of mobile applications the user should be able to skip parts of the context refresh to avoid annoying repetitions when there has been only a short time since the last use of the program. Therefore the MobEE provides the user with a fast forward button to skip unwanted facts or even the whole refresh. If the user has played the game in the mixed-reality mode the context refresh is provided in one of the other (text or 2D graphics) modalities, since the original context is no longer available.

EXAMPLE "Forgotten Valley", A PDA-BASED ADVENTURE

To illustrate how the MobEE game-engine described before can be used in practice we have created a small adventure game. The adventure "Forgotten Valley" demonstrates the capabilities and possibilities that are supported by the abstract concepts encapsulated in MobEE. Starting the adventure the user is offered the opportunity to either start a whole new game or continue a previously played storyline.

By choosing to play a new game he finds his Avatar placed in the middle of a unknown map, not knowing where he is or how he got here. In mixed-reality mode the user can start physically anywhere on the university campus that is our real-world "game board" for "Forgotten Valley".

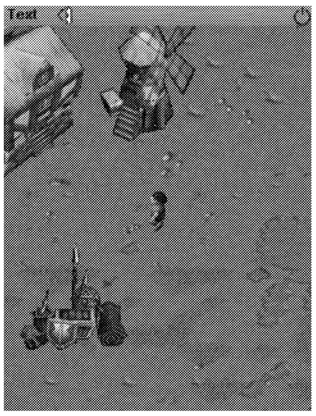


Figure 2: Starting Point

In conventional mode the user can use the pointing device (which can vary between different mobile devices) to move across the map which is scrolling according to the avatars' movements so that the avatar represented by a small person always stays in the centre of the screen. Exploring the surroundings in this manner, the player encounters different places where he may find hints about his whereabouts and how to move on in the game. In mixed-reality mode the user physically walks around on the university campus to discover the places relevant for the game.

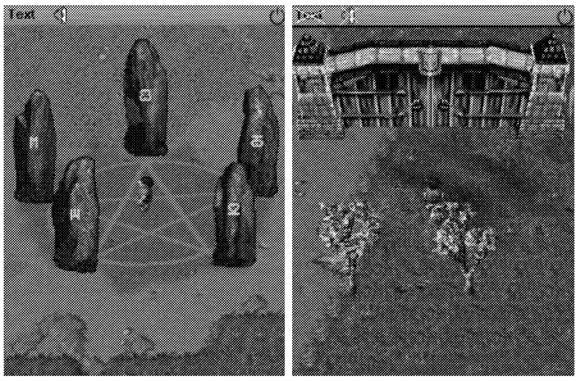


Figure 3: Riddles to solve

The user has to solve several little puzzles and talk to the people populating the valley to eventually find his way out. The behaviour of the non-player-characters as well as the overall logic of the game is realized by several automatons. Figure 4 shows the automaton, that controls the behaviour of the merchant's daughter.

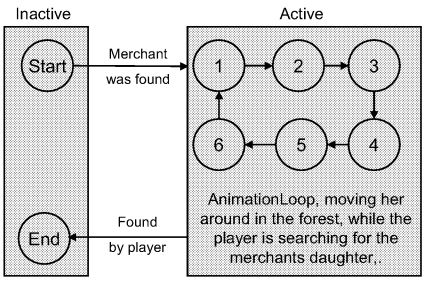

Inactive
Active
Start
Merchant
was found
1
2
3
6
5
4
AnimationLoop, moving her around in the forest, while the player is searching for the merchants daughter,.
End
Found
by player

Figure 4: Behaviour automaton for NPC

The daughter appears first, when the player talks to the merchant, who asks him to rescue his daughter. The automaton switches then to an active state with an animation loop. And back to an inactive state, when the daughter was successfully rescued.
All actions of the user and corresponding "experiences" of his avatar are recorded by the program and saved into a file. The information contained in this reference file is then used as the basis for the context refresh when the user wants to re-enter a previous played game.
When the user chooses to continue a game that he started at an earlier time, he is presented with an automatically generated re-narration of his previous adventures in the game world. The context refresh shows the most important events in the storyline (as specified by the game designer). The context refresh or scenes therein can be skipped by the user by pressing the "fast-forward" button.

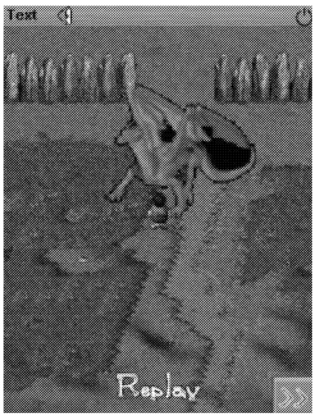


Figure 5: Context Refresh, showing an important part of the story

The game uses background music, spoken parts and written text to tell a story that is designed to be interesting and captivating. Clicking on the menu-bar the user can choose between different combinations of output modalities (e.g. text, graphics, audio, or mixed-reality).
The same adventure can thus be played as a pure text-adventure, as a 2D graphics game or a mixed-reality experience using the same game-engine. To ensure an enjoyable game experience in text-only mode more detailed descriptions of the locations could be added to substitute for the graphics and a linked map in order for the avatar to move around. Using MobEE it is thus

possible to create a game that can be played in several representations: text- 2D graphics and mixed-reality . The basic content can be extended and optimized to ensure an enjoyable experience in all modes.

Mixed-Reality mode

In this section we present initial experiences with the mixed-reality mode of the application. As the hardware platform we use a Pocket PC PDA in combination with a Bluetooth GPS receiver (Holux GR-230) and a plug-in camera (FlyCam).
Figure 6 shows the same riddles as in Figure 3 in the physical environment on the campus.

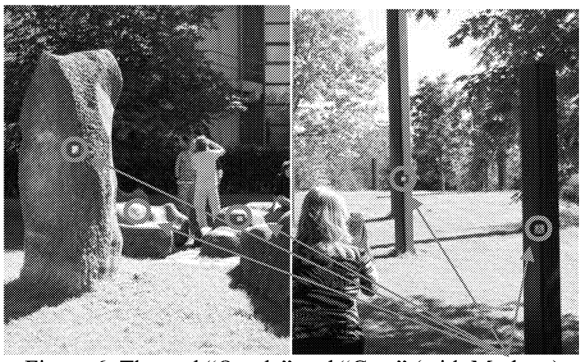


Figure 6: The real "Oracle" and "Gate" (with Markers)

When the user approaches the group of stones (Figure 6, left) the scrolling map on the PDA signals a possible "game-location". When the user takes a picture of one of the markers the "Oracle-Riddle" started similar to the one in the 2D-Version. After a short explanation of the riddle the user has to take pictures on the markers on the stones in the right order to solve the riddle. When he succeeds, additional information is displayed, that tells him about an old dangerous dragon of huge ancient wisdom and the story continues. During the navigation test-users did not use the PDA continuously, but sporadically checked it for new information, spotting for the characteristic markers on the "game-locations".

SUMMARY AND OUTLOOK

We have presented a game-engine for mobile devices that supports the development of entertainment applications with special respect to the mobile context of use. It enables the exchange of content representation to ensure playability on a wide range of mobile devices. The engine also gives the user a short summary if he resumes an interrupted game, to enable intermittent use of mobile entertainment applications containing complex narrative content.
We have also added a mixed-reality mode to the application to explore the use of augmented-graphics presentations and physical locomotion in the interface of games.
While the necessity of markers still restricts the practical playability in a real-world environment our initial experiences have been very positive. For the future we therefore plan to extend our experiments with mixed-reality games, specifically:

- exploitation of enhanced 3D capabilities (Open GL ES) when they become available
- replacement of marker-based position identification
- evaluation of the use of differential GPS and acceleration sensors to enhance location accuracy
- use of electronic maps (e.g. ATKIS) and online authoring tools to enable play groups to adapt location-based games to their own surroundings
- use of spatial databases for game-content management
- exploration of physical group-play activities

In another project the text version of "Forgotten Valley" is currently extended by a speech synthesizer to experiment with the re-integration of blind and visually impaired people into game play.

REFERENCES

AR-PDA (2003): The AR-PDA, a personal digital mobile assistant for virtual and augmented reality, Project-homepage http://www.ar-pda.com

ARToolKit (2002, 2003): ART02 & ART03, First & Second International IEEE Workshop on the Augmented Reality ToolKit, Darmstadt 2002, Tokyo 2003

Blast Theory (2003, 2004): Can You See Me Now, http://www.canyouseemenow.co.uk

Dunlop, M. D. and Brewster, S. A. (Eds.) (2001): Proceedings of Mobile HCI 2001: Third International Workshop on Human Computer Interaction with Mobile Devices, IHM-HCI 2001 Lille, France, September 2001

Gavrila, D.M. (1999): The Visual Analysis of Human Movement: A Survey. In Computer Vision and Image Understanding, Vol.73, no.1, 1999, Academic Press, pp.82-98

Geocaching (2004): Geocaching Homepage http://www.geocaching.com/

Johnson, C. (Ed.) (1998): Proceedings of the First Workshop on Human Computer Interaction with Mobile Devices, GIST Technical Report G98-1, University of Glasgow, Scotland

Nokia: Series 60 Platform: http://www.forum.nokia.com

Piekarski, Wayne and Thomas, Bruce, (2002) ARQuake: The Outdoor Augmented Reality Gaming System, Communications of the ACM,2002 Vol 45. No 1, pp 36-38

Reimann, C., Paelke, V., Stichling, D. (2003): "Computer Vision Based Interaction-Techniques for Mobile Games" in Proceedings of Game-On 2003, London, November

Stichling, D. and Kleinjohann, B. (2002): {CV-SDF} - A model for Real-Time Computer Vision Applications, in proceedings of WACV 2002: IEEE Workshop on Applications of Computer Vision, Orlando, FL, USA, December 2002

Stichling, D. and Kleinjohann, B. (2003): Edge Vectorization for Embedded Real-Time Systems using the {CV-SDF} Model, in proceedings of VI2003: 16th International Conference on Vision Interface, Halifax, Canada, June 2003

Symbian OS (2004): http://www.symbian.com

WMCSA (2002): Workshop on Mobile Computing Systems and Applications, IEEE, http://wmcsa2002.hpl.hp.com/

HAPTIC INTERFACE FOR SPATIAL AWARENESS IN GAMES

Lubo Jankovic[1] and Aineias Martos[2]

[1]InteSys Ltd
University of Birmingham Research Park
Vincent Drive, Edgbaston
Birmingham B15 2SQ, UK
E-mail: L.Jankovic@e-intesys.com

[2]School of Computer Science
The University of Birmingham
Edgbaston, Birmingham B15 2TT, UK
E-mail: eniasmartos@yahoo.gr

KEYWORDS

Spatial awareness, force feedback, haptic interface, virtual environment, tactile perception, games interfaces

ABSTRACT

The paper investigates spatial awareness in a virtual environment using tactile feedback devices in the absence of vision.

The perception mechanism based on vision first detects general and subsequently specific features of the environment, thus making the observer spatially aware of the context.

In the absence of vision, when the observer uses force feedback to explore the environment, spatial awareness cannot be achieved easily. Tactile exploration of the environment starts from a small region in the vicinity of the initial exploration point, and achieves spatial awareness gradually, by widening the region of exploration and building mental maps of relationships of objects in it.

The paper reports on experimental research in which objects diffuse their field of presence into a virtual environment. The user subsequently detects this by a force feedback device, and builds a mental map of the space much accordingly. Application of his method to games interfaces for the blind is discussed.

INTRODUCTION

Initially, force feedback interfaces were used almost exclusively for research labs and more specifically for medical applications, tele-operations and others (Qin, and Kaufmany, 2001; Floyd, 2000). However, the usage of force feedback interfaces has increased in recent years. The reasons for this increase is the progress of technology which in turn brought better performances by relevant devices and at the same time reduced their prices. The technologies which refer to force feedback interfaces are relatively new and it is not an exaggeration to say that they are still in their infancy (Salisbury, Conti, and Barbagli, 2004; Salisbury, 1995).

Only a few papers have been published which address the question how to use haptic interface to benefit blind computer users. Some of these papers deal with game interfaces (Sjostrom, 1999). However, games and interfaces suggested in Sjostrom's work are based on a battleship game and the interface based on three basic concepts: 1) navigation in virtual space; 2) finding an object; and 3) recognizing the object. However, it is important to point out that for blind people to recognize tactual characteristics of an object, prior knowledge plays a significant role (Argyropoulos, 2002).

Using a lattice in combination with the audio feedback suggested by Sjostrom (2001) it is expected that blind users would approach graphical user interface buttons faster and more accurately, recognising shapes more easily than textures. However, further investigation is needed to develop accurate guidelines on how to design haptic interfaces for computer games or other applications useful to blind users. It seems that for the time being there has not been much investigation in this area.

In this research, we will deal with a small area of force feedback interfaces using a Phantom device (Massie and Salisbury, 1994). This device is capable of providing a force feedback which enables the user to handle and to explore a virtual object and its characteristics such as texture and shape.

Until now, and to our knowledge, the usage of the Phantom has been based on simultaneous force feedback and visual feedback received from the visual display (Massie and Salisbury 1994). The main research question which we address in this paper is whether it is possible to use the Phantom effectively without the visual feedback.

This would mean that the virtual environment produced by the Phantom would need to provide the user with more information from the force feedback interface to compensate for the absence of information which would have been provided by the graphic display.

We will also address questions of spatial awareness and how it can be presented in a more realistic manner in virtual environments. This work was motivated by the research carried out by Lahav and Mioduser (2003) and focuses only on haptic feedback. However, we are not attempting to develop an application similar to the application that Lahav and Mioduser (2003) developed, which used complementary audio feedback. Instead, we wish to use force feedback in a similar way but without the audio feedback. Our goal is to investigate spatial awareness by using haptic interfaces only.

The Problem

First, it is important to understand similarities and differences between vision and touch. With exploratory touch, we explore the object starting from a small area that we touch with our fingers and scanning the rest of the object. With vision we initially see the general features of the object and its context in the environment, and then focus on the specific aspects of what we see (Mitsopoulos 2000).

If it is impossible for the user to see the screen and follow the cursor position then an approach towards an object will turn out to be a very difficult task. This is so because the user does not have information about the position of the object and his/her only exploratory tool is actually the fingertip or a stylus of the Phantom. In other words, the user has to randomly search through the whole virtual space to find the object.

Dimensions of the virtual space produced by the Phantom are relatively small, similar to those of a mouse pad or an A4 sheet (Basdogan and Srinivasan 2001). Nevertheless, both the fingertip and the stylus tip are much smaller than the workspace. Hence, the smaller the object is, the harder it is to find it in the virtual space without using vision.

The 3D work space is similar to a glass box and the user is allowed to explore this space only with one of his fingers or with a stylus. As virtual objects which are represented in this scene can be placed anywhere inside the box including the surfaces, the number of the places that one object can occupy is infinite.

If there is more than one object, the user has an additional difficulty: apart from the task of finding the location of the objects, he/she has to find the spatial relationship between them as well in order to successfully navigate from one object to the other.

The only reference points the user has inside the virtual environment are the walls. However, even with this information, the user is unable to construct the mental map of his/her orientation.

If the object is on one of the walls of the virtual environment and the user has identified its location once, then it is much easier for him/her to re-identify it. The role of the walls is crucial because they form a type of orientation in the work space.

The main problem the user faces when exploring the work space is the identification of the object's location. This happens because the only way that the user can "discover" the presence of the object is by collision detection between the Phantom pointer and the object.

However, in some cases the object is not solid and as a result it is possible for the object to be penetrated. This implies that the user will not feel any type of collision. Instead, s/he will enter and explore the inner part of the object, missing all the information provided by its texture, shape and so on (Basdogan and Srinivasan 2001).

Suggested Solution and Research Objectives

As explained in the previous section, the main problem the user faces when exploring the work space is an advance and gradual identification of the object's location without visual feedback.

The force feedback device should be able to provide information to allow the user to perceive the presence of an object in the workspace (Lahav and Mioduser 2003). The forces applied to the user's hand should give him/her the impression that the object attracts or repels the Phantom pointer. This force that the object applies to the phantom pointer has similarities with the gravitational force.

However, when using the information that the Phantom normally provides via the force feedback, even if the user is able to find an object in the virtual space, s/he is unable to construct a mental map for his/her orientation.

A possible solution could be to partition the space, providing the user enough information for his/her orientation (Basdogan and Srinivasan 2001).

For instance, it is possible to develop a 3D lattice inside the work space and objects in the lattice could diffuse their presence outwards into the lattice, thus providing a gradient of increasing or decreasing gravity which could be detected by the Phantom, thus helping the user establish spatial awareness.

This could enable the user learn how to move from one point to another, without losing orientation. Also, it would be possible for the user to find out the spatial relationship between two or more objects in order to navigate from one object to another.

The objective of this work is to develop a software tool that enables to investigate the above assumptions experimentally. We expect to derive useful conclusions on how to represent objects and the environment around them to make it possible for the Phantom to give users the necessary information to compensate for the information that would have been by the visual display as explained above.

If successful, the results of this work will enable blind people to achieve spatial awareness in virtual environments much more easily, and interact with various computer applications, including games, more efficiently.

BACKGROUND

When something is around us and we cannot see it for some reason but we can feel it, it seems that it is possible to perceive objects by intuition. However, this "intuition" has origins in the physics of the world around us. Objects change the local physical environment by their presence: they have a certain temperature which can be felt through radiation, they influence the bouncing of sound, air and smell.

The way that we see is hierarchic (Mitsopoulos 2000). In other words, when we are in front of a group of images, we first see the whole group, and later we focus on particular parts of the group. This process continues until we focus on the most important part for us. Mitsopoulos (2000) used the example below to describe this idea:

"For instance, one may look at a room and perceive a number of objects such as a desk, a chair and a bed. By focusing ones attention on the desk, one may distinguish that there is a desktop lamp, a laptop computer and some stationery equipment. By further looking at the laptop computer, one may see its keyboard, and so on."

If we want to show our room to someone as in the quotation above, but using the haptic sense only, we cannot show him/her everything in the first instance. Instead, we have to do it in sequence. For example, we have to put his or her hands first to the desk and then to more specific objects like laptop and desktop lamp. However, everybody understands weather or not he/she is inside the room using other senses like audition, temperature differences, smell etc. This example clarifies the difference between the vision sense and the haptic sense in the real environment.

The sense of vision depends on the sensitivity of the eye to light whereas haptic is "intrinsically bilateral" (Salisbury and Srinivasan 1997) between us and the environment. The way the haptic works is by exchanging energy between our hands and the object that we touch (Salisbury and Srinivasan 1997). So, we can say here that the "haptic interaction is bidirectional" (Salisbury, Conti, and Barbagli 2004) and the vision is passive. Due to our multimodal interaction with the real world, we receive information from the environment by using several senses in a complementary way to each other. We use at least three senses in our everyday interaction with the real environment (Salisbury and Srinivasan 1997).

In a virtual environment, the number of senses used to perceive objects is substantially reduced. If haptic sense is the only mechanism of perception, we need a lot more information that will compensate for the absence of other senses.

In the next section we describe the Phantom force feedback interface as context for our experimental work.

Phantom Description

In human computer interaction, we often use displays for vision representation, speakers and sound cards for audio representation and the more recently, haptic interfaces, in order to convey information from the computer to the user. In computer graphics we use displays, and similarly in computer haptics we need equivalent devices to make it possible for the user to explore objects (Salisbury, Conti, and Barbagli 2004). Devices that make it possible for the user to interact with the computer by touch are called haptic displays.

They enable the user to feel, at the simplest level, the virtual environment, or more specifically, the objects that this environment contains. The Phantom (Fig. 1) is a haptic display which sits on our desktop and looks like a miniature desktop lamp (Salisbury and Srinivasan 1997).

The Phantom tracks the motion of a stylus or a thimble operated by the user and exerts dynamic forces back on them, simulating surface texture and collision with physical objects.

The virtual environment that it produces is similar to a box the size of the mouse pad or an A4 sheet or a keyboard of a laptop, where the user has the ability to move his/her hand freely, by using either the fingertip or the stylus interface.

In comparison with other devices, the Phantom differs in that it is a 6 degree of freedom device, giving the user the ability to move inside the virtual environment in all directions. As far as the hardware is concerned, the Phantom connects the finger thimble to three brush servo motors via a mechanical arm. The servo motors are connected to two encoders to enable the Phantom to track the position of the users hand and to calculate the forces which should be exerted back on their hand.

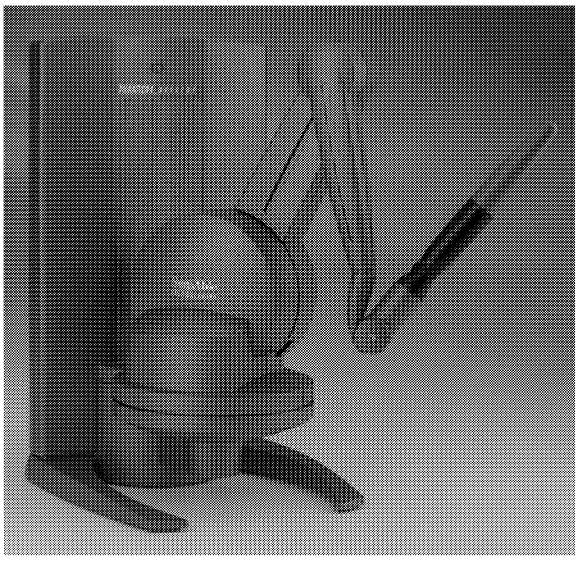


Figure 1: Phantom Force Feedback Interface
(Source: www.sensable.com)

METHOD

The virtual environment used in this work in is the workspace created by the Phantom force feedback device. The Phantom permits the user to interact with virtual objects which are inside this workspace. Users can touch the virtual object with the fingertip of their index finger and because the device applies an explicitly controlled force to the fingertip, it is possible to give the user a haptic sense of texture and shape of the object (Massie and Salisbury, 1994).

Our objective is to add to this environment the ability to find the object in the workspace without feedback from the screen. One way to do it is to create a force which repels the cursor away from the object. Another approach is to create a gravity-like force which attracts the cursor.

Both of these approaches have a similar drawback as application of forces could interfere with the way the Phantom is used. According to its designers, Massie and Salisbury, the Phantom uses a force between 1-1.5 N when the user moves the hand between two sides of the workspace (Massie and Salisbury 1994). Therefore, if we add a force field around objects inside the work space, the usage of Phantom will not be the same.

We therefore divided the 3D workspace into a lattice of 10 divisions in each direction. Due to limitations of the object library supplied with the Phantom, it was not easily possible to implement gradual diffusion of presence of an object. Consequently, the presence of objects could not be represented with a gradual change of resistance when approaching the object.

Instead, a slightly different lattice was designed. This new lattice had discrete points at intersections of the grid, and each point was assigned an attractive force, effectively providing a gravitational field around it. The intensity of these gravitational fields of individual lattice points was proportional to the distance of the points from the object: it was higher nearer the object and lower further away from the object. The number of objects to be detected by the user was limited to three for practical reasons (Fig. 2).

In this way, when searching for an object, the user would experience "stickiness" of the lattice points, and this stickiness would become higher in a closer proximity to the object.

A disadvantage of this approach was that the resistance experienced by the user increased when approaching each lattice point and reduced when departing from the same point. Even though each point had different intensity of the gravitational field depending on the distance from an object, this gave the user the impression of slightly irregular gravitational field, and we were concerned whether this would influence our results.

EXPERIMENTS AND RESULTS

In our experiments we had ten subjects, one of whom was blind. The rest of the subjects were prevented from seeing the screen, by placing the Phantom far from the screen.

The only prior knowledge that the subjects had about the experiments was that the objects were spheres in 3D space and that the task was to find them in the shortest possible time.

Each experiment was conducted in two stages: a) with the lattice and b) without it. Time was recorded and a questionnaire was given to subjects to answer.
The questionnaire was used only in the second experiment, and its goal was to examine if the mental map discussed earlier had been created. To assess this, we asked the subjects whether the spheres ware connected with straight lines; what shape they formed; how the formed shape was placed in the scene etc.

If the subject was able to describe the shape and its position in the scene, we assumed that the spatial map had been created, giving the user the sense of spatial awareness in the virtual environment.

Experiment 1

The objective of this experiment was to establish if the lattice that we described in the "Method" section makes it easier for the user to find an object inside the scene. In this experiment the scene contained only one sphere randomly placed inside it. Each time the program started, the sphere was placed in a different position than in the previous run.

There were 10 sessions of this experiment. In each session the user (subject) was asked to find the sphere with and without the lattice. The sphere was always the same size, and the subject was not allowed to see the screen at any time during the experiment. The search always started from the same position in all sessions, with and without the lattice.

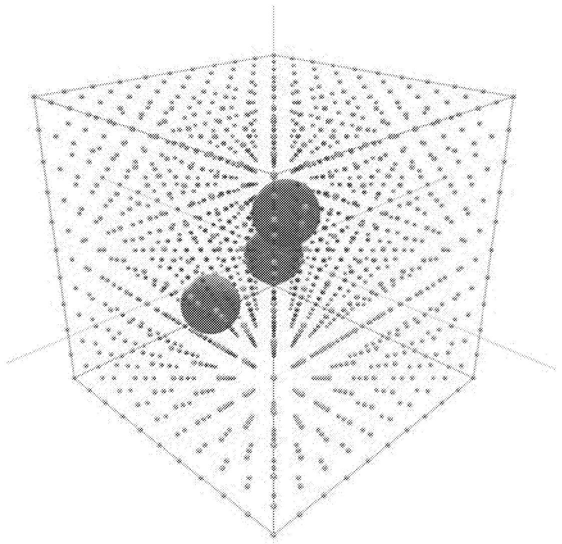
Table 1: Results of Experiment 1: Finding Single Objects in Random Positions in 3D Space

Session	Time with lattice (seconds)	Time without lattice (seconds)
1	56.8	14.8
2	16.3	20.1
3	16.6	198.1
4	29.5	54.8
5	7.0	73.0
6	13.3	22.6
7	12.7	25.3
8	13.0	37.9
9	5.6	63.9
10	5.1	9.4
Average	17.59	51.99

The starting position was always chosen to be the default position of the stylus of the Phantom for two main reasons: a) it was easy to control the scene from this position being in the middle of the scene and b) the subject couldn't have any prior knowledge about the position of the object.

The time to find the sphere was measured until the subject was sure that s/he had found it. It was not enough just to detect the object but also to examine it to be sure that it was the wanted object and the walls of the scene. For these reasons, it was up to the subject to decide when to stop the experiment.

The results of the experiment are shown in Table 1. The results show that on average, much shorter times were needed to find the object with the lattice, and confirmed that the lattice helped the users achieve spatial awareness.


a) complete lattice

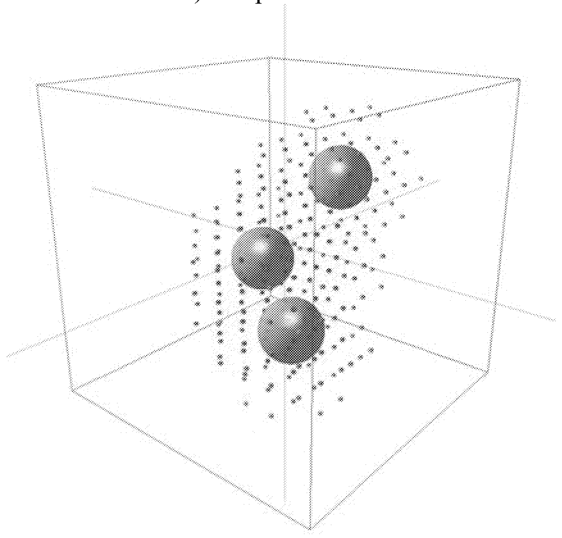

b) partial lattice proportional to the intensity of gravitational field in the grid intersection points

Figure 2: Lattice and Objects in the Second Experiment

Experiment 2

Having established that the lattice helps to develop spatial awareness, we conducted the second experiment to find out whether the lattice can help the user learn spatial relationships and build mental maps.

In this experiment, we used three spheres positioned randomly in the space. Unlike in the first experiment, the positions of spheres were kept constant in the runs with and without lattice with each subject. There were ten sessions of the experiment, each with different subject. The results of the second experiment are shown in Table 2.

These results were initially surprising. It appeared that it took significantly longer time to find objects with the lattice than without. This directly contradicted the results of the first experiment.

Table 2: Results of Experiment 2: Finding Objects in Fixed Positions in 3D Space

Session	Time with lattice (seconds)	Time without lattice (seconds)
1	389	170
2	395	188
3	403	315
4	399	323
5	105	140
6	357	142
7	600	62.4
8	494	600
9	600	136
10	326	177
Average	406.8	225.34

However, there are some important differences between the two experiments which explain the results. Firstly, this experiment had fixed positions of spheres in the runs with and without the lattice. The users were therefore able to learn spatial positions in the run with the lattice and apply this knowledge in the run without the lattice. Secondly, having three spheres in fixed positions in the workspace enabled the users to learn spatial relationships between them even quicker, which influenced the shorter times in the second experiment even further.

The generally longer times in this experiment in comparison with the first experiment were the consequence of having three rather one object to find.

DISCUSSION

Initially, the aim of our research was to determine whether spatial awareness can be increased by using force feedback devices when the vision feedback is not be used. We also attempted to investigate how a mental map could be developed in virtual environments.

The first experiment showed that the spatial awareness significantly improved with the lattice, and that haptic devices can be used for increasing spatial awareness.

The second experiment was slightly more complex, and it attempted to prove that users can develop a mental map inside the virtual space. We believe that this experiment confirmed that the lattice helped this process, as users learnt the spatial relation between the objects in the first stage of the experiment and as result of this, the time to find the objects and their relationships were reduced in the second stage without the lattice.

There are several limitations of this work, which we would like to look into and overcome by further experimentation.

Firstly, the experimental sample was too small to draw general conclusions. More experiments are needed to determine with more confidence whether the lattice helps increase spatial awareness and speeds up the development of mental maps.

Secondly, the lattice used did not provide gradual change of intensity of force fields surrounding the objects. Instead, the presence of objects was modelled by gravitational fields assigned to discrete points at the intersections of the grid. Although the intensity of individual gravitational fields was proportional to the distance from the objects, the result of this was that the user experienced a series of small ripples when approaching the object. We would like to improve on this by making the lattice to generate a much more smooth resistance proportional to distance from objects.

Thirdly, by investigating the records of user observations during this experiment, we found that the lattice confused those subjects who were experienced Phantom users, more than the subjects who had never used it before. It was not always certain that the lattice helped, and we therefore cannot claim that it would provide a universal interface that increases spatial awareness in cases where vision cannot be used. We believe that this was due to the discrete design of the gravitational field in the lattice as explained above.

However, as results of our two experiments to date appear to be conclusive and mutually supporting within these limitations, we believe that further research into this subject is justified.

GAMES AND OTHER APPLICATIONS

If further research proves more conclusively a universal increase of spatial awareness from force-feedback interfaces and application of three-dimensional lattice, then we believe that this can be used to improve games interfaces for blind people. Reversing the notion of touch, from its original specific-to-general nature, to the general-to-specific nature which is analogous to vision, would have a significantly positive impact on the ability of blind people to use not only computer games, but other applications for which spatial awareness is essential to complete a set of tasks.

These other applications include various situations in which vision is not possible.

CONCLUSIONS AND FUTURE WORK

We investigated the role of haptic feedback for the purpose of achieving spatial awareness in a virtual environment, in cases where vision could not be used. The objective of the work was to attempt to reverse the notion of touch, which perceives objects from specific to general features, and make it similar to the notion of vision, which perceives objects from general to specific.

We designed an experimental virtual environment divided into a lattice of 10 x 10 x 10, in which the user had a task of finding between one and three spheres using the Phantom force feedback interface. The intersection points of the lattice were assigned gravitational fields proportional to the distance from the objects, thus giving spatial clues to the user.

The experimental results confirmed that it was easier to find objects with the lattice, and that the lattice also helped the users learn spatial relationships between the objects, thus helping them to build mental maps of the 3D workspace.

We believe that the results of this work can be used to develop games and other interfaces that can help blind users increase spatial awareness and use computers more efficiently.

Future work will focus on eliminating the limitations of the present work in progress, namely by improving the lattice and the experimental sample, and testing the interface in a real application.

REFERENCES

Argyropoulos, V. (2002). An investigation into tactual shape perception and geometrical concepts in students who are blind. PhD thesis, School of Education, University of Birmingham, Birmingham, 52-561 UK.

Basdogan, C. and M. Srinivasan (2001). Haptic rendering in virtual environments.

Floyd, J. (2000). Haptic interaction with three-dimensional bitmapped virtual environments. MIT Artificial Intelligence Laboratory.

Qin, H. and A. Kaufmany (2001). A novel haptics-based interface and sculpting system for physics-based geometric design. Department of Computer Science State University of New York.

Lahav, O. and D. Mioduser (2003). Multisensory virtual environment for supporting blind persons' acquisition of spatial cognitive mapping.

Massie, T. and J. Salisbury (1994, November). The phantom haptic interface: A device for probing virtual objects. DSC 55 (1), 295–300. Proceedings of the ASME Winter Annual Meeting Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, DSC.

Mitsopoulos, E. N. (2000). A Principled Approach to the Design of Auditory Interaction in the Non-Visual User Interface. Phd thesis, University of York Human-Computer Interaction Group, Department of Computer Science.

Salisbury, J. K. and M. A. Srinivasan (1997, September). Phantom-based haptic interaction with virtual objects. IEEE Computer Graphics and Applications 17 (5), 6–10.

Salisbury, K. (1995, Nov). Haptics: The technology of touch. HPCwire 4, 223– 224. HPCwire Special. Nov. 10, 1995.

Salisbury, K., F. Conti, and F. Barbagli (2004, March). Haptic rendering: Introductory concepts. Computer Graphics and Applications 24 (2), 24–32.

Sjostrom, C (2001) "Using Haptics in Computer Interfaces for Blind People", CHI 2001, Seattle, USA. March-April 2001.

ACKNOWLEDGEMENTS

The authors would like to thank Professor Alan Wing, Director of the SYMON lab at the Behavioural Brain Sciences Centre of the School of Psychology at the University of Birmingham, UK for providing experimental facilities for this work, and to Nick Roach, researcher at SYMON for his help with programming the Phantom interface. Their collaboration is gratefully acknowledged.

BIOGRAPHIES

Lubo Jankovic obtained his PhD in Mechanical Engineering from the University of Birmingham. He is the founding Director of InteSys Ltd, Honorary Lecturer at the University of Birmingham, and Senior Lecturer at the UCE. His research interests are in the field of Virtual Reality and Science of Complexity.

Ainieas Martos obtained his first degree from the department of Informatics and Telecommunication at the University of Athens, Greece and his MSc degree in Advanced Computer Science from the University of Birmingham, UK. His research interest is in the area of accessibility using speech recognition and synthesis. He is currently working at the University of Athens as researcher on accessibility for blind pupils in school classrooms.

GAME DESIGN AND EDUCATION

DESIGNING CHALLENGES AND CONFLICTS: A TOOL FOR STRUCTURED IDEA FORMULATION IN COMPUTER GAMES

Stephen Tang
Kolej Tunku Abdul Rahman
School of Arts and Science, Computer Science Division
Jalan Genting Kelang, Setapak
53000 Kuala Lumpur, Malaysia.
tangot@mail.tarc.edu.my

Dr Martin Hanneghan,
Abdennour El Rhalibi
School of Computing and Mathematical Sciences
Liverpool John Moores University
Byrom Street, Liverpool, UK, L3 3AF
m.b.hanneghan@livjm.ac.uk, a.elrhalibi@livjm.ac.uk

KEYWORDS

game design; idea formulation; soft systems modelling; designing challenges and conflict

ABSTRACT

Challenge and conflict are elements that all game designers strive to engineer into their games. Research shows that challenge is what drives a high proportion of games players yet there are few published tools that can be used to assist the game designer in constructing useful challenges and conflict leading many new game designers to resort to the 'tried and trusted' techniques used in previous games and hence limiting the originality of new games. In this paper we apply the Soft Systems Methodology to game design and assess its suitability as a tool for structured idea formulation in games.

INTRODUCTION

Computer games are creative and innovative artefacts that exploit interactivity to form a new genre in entertainment; interactive digital entertainment. Games should not be perceived as application software but rather should be categorised as digital content similar to the entertainment industry. Computing technologies are just the underlying facilitators enabling an exceptionally wide range of creative ideas to be realised.

Creating compelling games evolves from a primary source; creative and innovative ideas. Creativity is a process of exploration and exploitation of knowledge and experience with the aim to generate ideas that are original, novel, useful, relevant and possess adaptive value (Nairne 2003). The ability to visualise the "big picture" and correlates matters requires understanding and time. Though creative thinkers are able to formulate ideas for a given scenario in a timely manner, sometimes it requires numerous cases of trial and error before a truly novel idea can be formulated. In the search of a novel idea, there might be some reasonably creative ideas that the designer may not be able to afford enough time due to cost constraints or impending deadlines. Ironically, the disposal of these ideas may turn out to be a costly waste of intellectual resources (remember that almost all great games will produce a sequel and those lost creative ideas may be needed eventually).

Games development is arguably one of the most demanding and high-risk software projects. Evidence suggests that most of the available game titles in the market today are sequels or licenses (Spector 2003). As development costs become increasingly high, it is a wise option for producers to work on sequels or licensed intellectual property (IP) mitigating risks to the publisher. Though there are strict requirements from originator of the IP and the publisher in developing the content, there can still be room creativity within the scope defined.

Soft Systems Methodology (SSM) aims to improve the areas of social concern by learning about a system through an iterative process with constant debates made to reflect on the real world reducing the whole problem area into smaller and manageable problems to develop a full understanding on the system (Checkland and Scholes 1999). SSM is used to solve difficult and ill-structured problem areas such as the real world social problems where complex interactivities involved.

Computer games share many common characteristics of a social system but they are designed within a controlled environment. Computer games are (often) goal-driven interactive entertainment that incorporates challenges and conflicts to achieve a pleasurable emotional and intellectual engaging simulation (Crawford 2002). While goals within a game can be easily defined within the game, it is often a difficult task to introduce innovative challenges and conflicts that involve interactions. This paper proposes the use of Soft System Methodology as a tool to design challenges and conflicts for computer games. In Section II we look at the roles of challenge and conflict before we take an in depth look at soft systems in Section III. In Section IV we step through the seven stages of an SSM approach and apply this to the Pac-Man game and draw some interesting conclusions in Section V.

CHALLENGE AND CONFLICT IN COMPUTER GAMES

The Need for Challenge in Computer Games

Computer games are a form of play that engages players to solve problems within an imaginary world. While the objective of playing games is to experience fun through the interactions within the imaginary world, games should possess lower level goals and elements that can raise the complexity to achieve these higher level goals. Without such elements, the interactions may seem tedious and eventually introduce boredom to gamers and lead to loss of market. While this hasn't stopped publishers releasing games like this in the past, this practice can eventually detract from the work of great game designers.

According to Crawford, challenges can be explicitly categorised into cerebellar challenges and sensorimotor challenges (Crawford 2002). Cerebellar challenges involve abilities such as spatial reasoning, pattern reasoning, sequential reasoning, numerical reasoning, resource management and social reasoning to decide on the appropriate action in solving problems. Sensorimotor challenges often involve coordination ability between aural, visual and physical senses to achieve a certain task. In computer games, challenges presented often relate to both cerebella challenges and sensorimotor challenges.

Challenges invite the participant to be involved in an action or series of actions that can distinctively justify their superiority in mastering it. According to the Interactive Digital Software Association 2002 consumer survey, 71.4% of gamers play games because they are challenging (IDSA 2002). Overcoming such challenges introduces the sense of accomplishment and satisfaction to gamers that will eventually enhance the fun factor in the game prolonging the hours of game play and increasing the value of the game.

Challenges can also be deadly to game designers when gamers are unable to overcome the challenges posed. Such challenges are not fun and tend to cultivate frustrations toward the game play potentially affecting the 87.3% gamers who play games because they are fun (IDSA 2002). For the purpose of this paper, we shall not provide a discussion on game balance instead restricting our discussion to designing challenges and conflicts.

Conflict Intensifies Challenge

Conflict presents a disagreement that often requires a series of challenges to be overcome in order to resolving it. Unlike challenge, unanswered conflicts will often result in dominance by another opposing force. In addition to initiating challenges of a different dimension, conflicts intensify challenges posed to gamers with a model opposing the gamers' objectives. Conflicts can exist in various dimensions such as physical, verbal, political and economical with different degrees of directness (Crawford 2002). Game designers can often replicate conflict models from the real world into their game to heighten the sense of challenge and thus making the game more fun.

In a social context, the source of conflict evolves from the emotion of living species. Such activating behaviour distinctively builds personality which can discretely separate "allies" from "foes". Thus, devising conflicts within a simulated environment requires modelling of personalities conflicting with the model which gamers represent.

THINKING OF SYSTEMS

A system is a complex matter or set of matters that work together in an organized manner to achieve certain objectives. Take for instance humans; in the study of anatomy, a human system is constructed of seven different systems to support life. It is a complex biological being with organs that work together in an ordered and managed manner with its related parts to enable humans to survive.

The concept of systems not only applies to living organisms, but also to human activities. In an organisation, transforming raw materials into a product, or abilities, skills and knowledge into a service by means of generating income would require an essential set of ordered activities to satisfy the transformation. In the technological era, the concept of systems is widely used as a means to replace the activities done by humans and assign them to machines with the aim to speed up the transformation process and thus increase income generation. To assist in the analysis of systems, a system can consist of many smaller sub-systems, which serve different purposes within the larger context, but where all the sub-systems in combination are working towards the same objective.

Thinking Systemic

Systemic thinking considers the problem as a whole to perceive a clear concept of the "system" context defined or to be defined. Properties, attributes and functions within the system should be identified and links between elements should be defined and organised to ensure it serves a purpose, which will lead to the defined objectives.

"Hard" and "Soft" Systems

In the real world, systems can be either *hard* or *soft*. Hard systems are man-made systems with predictable behaviour which can be engineered to serve some purpose, whereas soft systems tend to inherit uncertainty and are less predictable. While most of the systems known to us are often labelled as hard systems, as science has discretely defined the predictable behaviour in certain situations, Checkland labelled human activity systems as soft systems. Even though sociologists and psychologists have presented studies which educate us about human behaviour, humans remain unpredictable simply because the power of making decisions lies with the will of individuals. Many would argue that humans can be influenced and instructed to perform the required tasks; however we should be reminded that it solely based on the objectives of the individual whether the task will benefit, aid or guide them to achieve the defined objective.

If we analyse the definition of soft systems, computer games do not qualify to be categorised as such. Computer games are a virtual world designed and developed as a simulation for entertainment purposes with well-defined rules and predictable behaviours of each entity within the game world. Therefore computer games are not soft systems.

However if we analyse from the design viewpoint, the initial game idea may qualify to be considered as a soft system. In typical game design activities, the game designer is required to imagine the game, define the way that it works, describe its internal elements and communicate this information to others (Adams and Rollings 2003). In most game design texts, the authors always place emphasis on imagining a game idea as a story and associated mechanics (game rules) and the interactivity (gameplay) e.g. (Adams and Rollings 2003; Crawford 1982; Crawford 2002; Rouse 2001).

Due to the variability of the design task, most of the elements within a game evolve from a single idea, which is then expanded in a logical manner through the process of creative thinking. While many elements within the design remain undefined in the early stage, learning about the *imaginary social system* within the game can aid in designing interactions, conflicts and challenges. Although the imaginary social system within the game can be modelled, there can still be unpredictability within the system that needs to be explicitly addressed so that appropriate conflicts and challenges can be designed to suit the needs of the system. Based on such justifications, we argue that computer games are, in fact, soft systems.

DESIGNING CHALLENGES AND CONFLICTS FOR COMPUTER GAMES USING SOFT SYSTEM METHODOLOGY

Soft Systems Methodology is an approach based on the theory of systemic thinking to study the components of, and relationships with, one another within the system in order to develop understanding about the system. SSM has been used for social system analysis, political system analysis and analysis of intervention. Further expanding the usage of SSM, we will advocate SSM in the process of designing challenges and conflicts in computer games.

Checkland's Formal Systems Model

The Checkland's Formal Systems Model defines a set of properties, which represent human activity systems. Checkland defines: "A system has a purpose(s), its performance can be measured, there is a mechanism for control, it has components, its components are related and interacting, it exist as a part of a wider system(s), it has a boundary, it has its own resources and it has an expectation to be adapted to."

Earlier we introduced the term "imaginary social system" and argued that computer games are soft systems. Checkland's Formal Systems Model only models a system within the universe, not the universe as a system. Games are often placed in an imaginary universe; therefore it is not suitable to model the entire game as a system. We should understand that games consist of three distinct sessions: pre-game session, in-game session and post-game session. The in-game session is usually represented by a collection of levels, which can be organised in a linear, non-linear, or hybrid model containing element of these two. A level within a game can be considered as an imaginary social system because it allows gamers to interact with other characters (be they other human gamers or artificial intelligent avatars). Though there are routes that allow gamers to switch mode from in-game session to pre-game session for adjusting game settings, only the in-game session offers interactions with the imaginary social system.

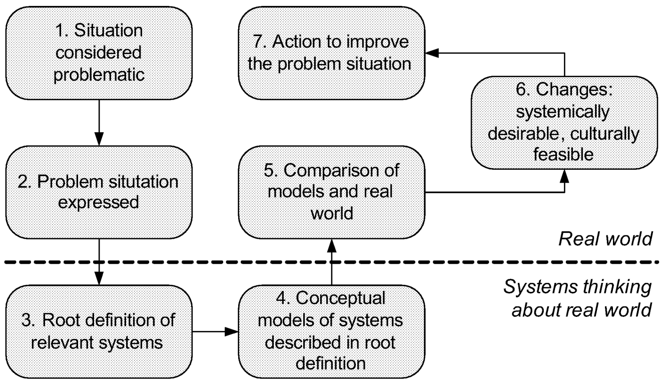
Therefore we can model a level within a game as an imaginary social system and apply SSM to this rather than the entire game. Checkland's Formal Systems Model properties are represented in appropriate game terminology illustrated in Table 1 below.

Table 1: Checkland's Formal System Model in Game Terminology

Formal Systems Model	Game Level
Purpose(s)	Level objective
Performance measurement	Game statistics
Mechanism for control	Game rule
Components	Level objective
Relationships and Interactions within Components	Game play, Conflicts and Challenges
Exist as a sub system	Game
Boundary	Scope of the level
Resources	Game object
Expectation of continuity and adapt or recover from disturbance	Save state

The Seven-Stage Model of SSM

The initial model of SSM was defined as a seven-stage model back in the mid 1970s and is illustrated in Figure 1. The model presents a sequence of tasks, which can aid analysts in understanding the social system by iteratively thinking systemic and engaging in debate on the perceptions of the real world. This in turn should reflect the issues using the concept of system thinking with the aim to propose solutions which can bring about improvement in the social system.



1. Situation considered problematic
7. Action to improve the problem situation
6. Changes: systemically desirable, culturally feasible
2. Problem situtation expressed
5. Comparison of models and real world
Real world
Systems thinking about real world
3. Root definition of relevant systems
4. Conceptual models of systems described in root definition

Figure 1: Checkland's Seven-Stage Model of SSM

In the process of designing games, we can also adopt such a model to improve the game play. In the following sections we will use this model as the framework to design challenges and conflicts within the game, which can enhance the fun aspect of the game.

Stage 1: Consider the problematic situation

Level design requires designers to model the situation considering the properties presented in Table 1. If we perceive game design as system design, levels are similar to modules within a system. With expected inputs and outputs, software engineers are required to design an algorithm to compute the desired output. Similar to the task of a software engineer, a game designer is required to fill in the gaps to enrich the game in any given situation by incorporating interactions and details that can bring the situation into "reality".

In reality, a situation is considered problematic if there are existences of potential threats, which can cause an objective to fail. The challenge in this context is to identify the possible threats and formulate counter measures in order to minimise the impact of the risk of failing. Games are mirrors of the reality where there are objectives (level objectives) to be achieved that will eventually lead to achieving the main objectives (game objectives) defined. Such achievement is made possible if the gamer is able to overcome the challenges and conflicts presented.

While the objective of gamers is to win, the task of game designer is to complicate the route to winning by means of incorporating suitable challenges and conflicts. This may not seems to be a problematic situation because the task of the game designer is to increase complexity of a given situation. Designing levels for games is no different from solving a problem in a given situation. It requires game designer to identify the potential threats for the purpose of entertainment.

Stage 2: Problem situation expressed

Analysis of any given problem situation requires a strong understanding of the given problem area to ensure appropriate solutions can be formulated. Checkland's seven-stage model uses the notion of a Rich Picture™, a pictorial approach to express the problem area by means of developing a through understanding on the area of investigation.

Unlike conventional software engineering diagramming tools such as flowcharts, Data Flow Diagrams (DFDs), Hierarchical Input-Process-Output (HIPO) Chart and, to an extent, Unified Modelling Language (UML), the Rich Picture is not constrained by any symbolic usage in illustration of problem area. Such flexibility allows practitioners to further express the problem situation without being constrained by the limitation of symbols.

In comparison to the conventional approach of problem expression, which usually reported in textual form comprises of words and digits, the Rich Picture allows practitioners to illustrate the relationships between the entities and thus invites the practitioner to think about the possible problems for each of the relationships defined. The approach tends to focus the mind of the practitioner within the scope defined (the relationship) and think of the possible problems that can happen to the relationship analysed in comparison to the real world situations.

In the context of game design, visual representation helps the game designer to visualise the game better than textual descriptions. We'll use the Pac-Man game as an example. The "social system" in this context is rather simple; Mr. Pac-Man is required to traverse the entire maze to collect all the power pills to increase his/her score without being caught by Mr. Ghost. The game idea can be illustrated in Rich Picture form similar to that shown in Figure 2. Game designers then can focus on the relationships between each entity within the game and think about the possible issues that can rise from such relationships (denoted by →) that can cause Mr. Pac-Man not to complete the objective.

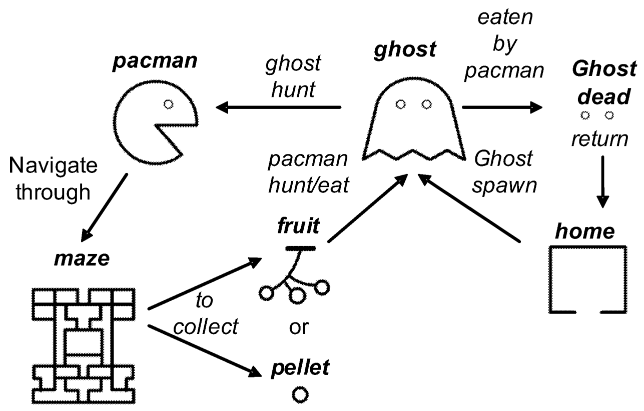


Figure 2: Pac-Man© Game Represented in Rich Picture™

Stage 3: Root definitions of relevant purposeful activity systems

The Rich Picture provides a comprehensive pictorial representation of the problem area. However it can be misinterpreted by others as individual imagination and interpretation could be varied. The Root Definition (RD) provides a short textual description of the purposes and transformation processes of the system to be modelled according to the system's principles (rules within the system being modelled). This RD can minimise the possibility of misconception.

The formulation of a root definition requires the practitioner to think about the transformation process which tends to change the form of input into output. CATWOE analysis (see Table 2) is a technique devised by Checkland to aid the formulation of coherent and comprehensive root definitions by considering the elements within the CATWOE mnemonic; C (Customers), A (Actors), T (Transformation process), W (*Weltanschauung* or worldview), O (Owners), E (Environmental constraints). The technique provides a structure for the practitioner to think about the elements within the CATWOE mnemonic and continuously reasons its relationship with the T (transformation process) and W (Weltanschauung).

Table 2: The CATWOE Mnemonic

CATWOE mnemonic		Description
C	Customers	The victims of beneficiaries of T
A	Actors	Those who would do T
T	Transformation process	The conversion of input to output
W	Weltanschauung	The worldview which makes this T meaningful in context
O	Owner(s)	Those who could stop T
E	Environmental Constraint	Elements outside the systems which it takes as given

Defining root definitions requires SSM practitioners to identify the activity systems within the scope of the problem area by analysing the Rich Picture™. In SSM, there are two kinds of relevant systems; "primary-task system" and "issue-based system". In the context of computer games, a primary-task system is man-made system which can be defined as the activity systems which serve the main objective of the game. Examples from the Pac-Man game are the "Mr Pac-Man collecting fruit or palette" and "Mr Pac-Man navigating through the maze". An issue-based system is defined as a problem area which inherits subjectivities which could not be modelled directly from the real world. Again, examples could be "Mr Pac-Man is hunting Mr Ghost", "Mr Ghost is wandering around", "Mr Ghost is hunting Mr Pac-Man" and "Mr Ghost spawned". Once the relevant systems activity has been identified, it can then be analysed using CATWOE analysis to ensure the root definition is valid and relevant.

In the context of this discussion, we would like to use CATWOE analysis to structure the user's thinking in the given scenario in order to introduce challenges and conflicts to the game. While some of the scenarios identified have the transformation process focused towards the Mr Pac-Man and some towards Mr Ghost, the CATWOE approach remains the same *Weltanschauung*; to raise difficulty for Mr Pac-Man to complete the level or objective(s).

Let's consider the "Mr Ghost is hunting Mr Pac-Man" activity system as an example. The transformation process is defined as "Mr Pac-Man is alive → Mr Pac-Man is dead" with the belief that Mr Pac-Man should be hunted in order to raise the challenge in the collection of pellets and fruits while navigating through the maze. Once the transformation process and *Weltanschauung* are defined, we should then think about the customers, actors, owners and environmental constraints. The CATWOE analysis for the "Mr Ghost is hunting Mr Pac-Man" activity system based on the original game idea by Toru Iwatani is presented in Table 3.

Table 3: CATWOE Analysis for Hunter-Prey Scenario

CATWOE mnemonic		Analysis Made
C	Customers	Mr Pac-Man, Mr Ghost
A	Actors	Mr Ghost
T	Transformation process	Mr Pac-Man is alive → Mr Pac-Man is dead
W	Weltanschauung	The belief that Mr Pac-Man should be hunted in order to raise the challenge in the collection of pellets and fruits while navigating through the maze.
O	Owner(s)	Mr Pac-Man
E	Environmental Constraint	Insufficient time to complete the objective. Complexity of the maze. Placement of fruits on the maze.

The CATWOE analysis has provided sufficient information for naming the activity system to be modelled, in this context, the "Mr Ghost is hunting Mr Pac-Man" activity system being discussed. While there are various approaches in defining the root definition, Checkland suggested the following definition schema: *do **X** by **Y** in order to achieve **Z*** to enhance understanding of the activity system being modelled. Based on the definition schema, the root definition for the activity system being modelled (the "Mr Ghost is hunting Mr Pac-Man" activity system) is as follows;

An intelligent hunting system with Mr Ghost(s) to hunt Mr Pac-Man, by increasing the speed of Mr Ghost(s) as well as the ability to strategise and hunt in a group in order to complicate the process of collection.

In the context of game design, the environmental constraint(s) serve as challenges within the game. Apart from the challenges, the activity system represented in the root definition defines conflict which can introduce a different perspective of challenges to the game.

Stage 4: Conceptual models of systems defined in root definition

In the stages of SSM, practitioners are required to model the system defined in the root definition from the previous stage by building a conceptual model of the activity system. The conceptual model presents the potential activities and its logical dependencies where the activities are represented in a verb-noun phrase using circles (activity) and arrows (logical dependency) as its modelling language. While other forms of conceptual model in system modelling do not limit the number of processes within a particular context (Kendall 1997), Checkland emphasises human memory span following Miller's 7 ± 2 magical number theory (Nairne 2003), limiting the number of activities to be modelled in the conceptual model ranging from 7 to 9 providing an environment to aid human to process and understand the model as a whole rather than a portion of it.

Here is a conceptual model built from the *intelligent hunting system* root definition outlined above:

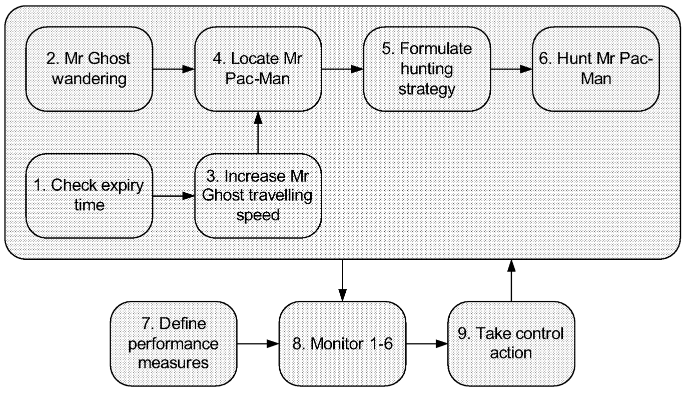


Figure 3: Initial Conceptual Model for Intelligent Hunting System Root Definition

An essential property within a system defined by Checkland is performance measurement which lacks in the conceptual model presented in Figure 3. SSM defines performance measurement in terms of *efficacy*, *efficiency* and *effectiveness* with its well-defined definition presented in Table 4.

Table 4: Three E's for Measurement of Performance

Criteria	Definition
Efficacy	"Does the system work?" — A logical prediction made onto the system determining the operational feasibility of the transformation process.
Efficiency	"The amount of output divided by the amount of resources used" — A cost benefit (not necessary monetary) comparison on the resources used and results obtained from the system.
Effectiveness	"Is the Transformation meeting the longer term aim?" — A judgemental prediction made onto the system determining the success towards achieving the aim(s) defined.

Although the initial conceptual model presented in Figure 3 possesses a general performance control mechanism (activities 7 to 9) to ensure the transformation process is achieved, logical analysis requires the 3 E's to enable SSM practitioner to logically reason the model ensuring the success of the transformation process. While the goal of designing challenges and conflicts is to enhance the fun aspect of playing games, measuring the performance is necessary in order to achieve a well-balanced game play. The revised conceptual model of the *intelligent hunting system* root definition incorporating the 3 E's for measurement of performance is presented in Figure 4.

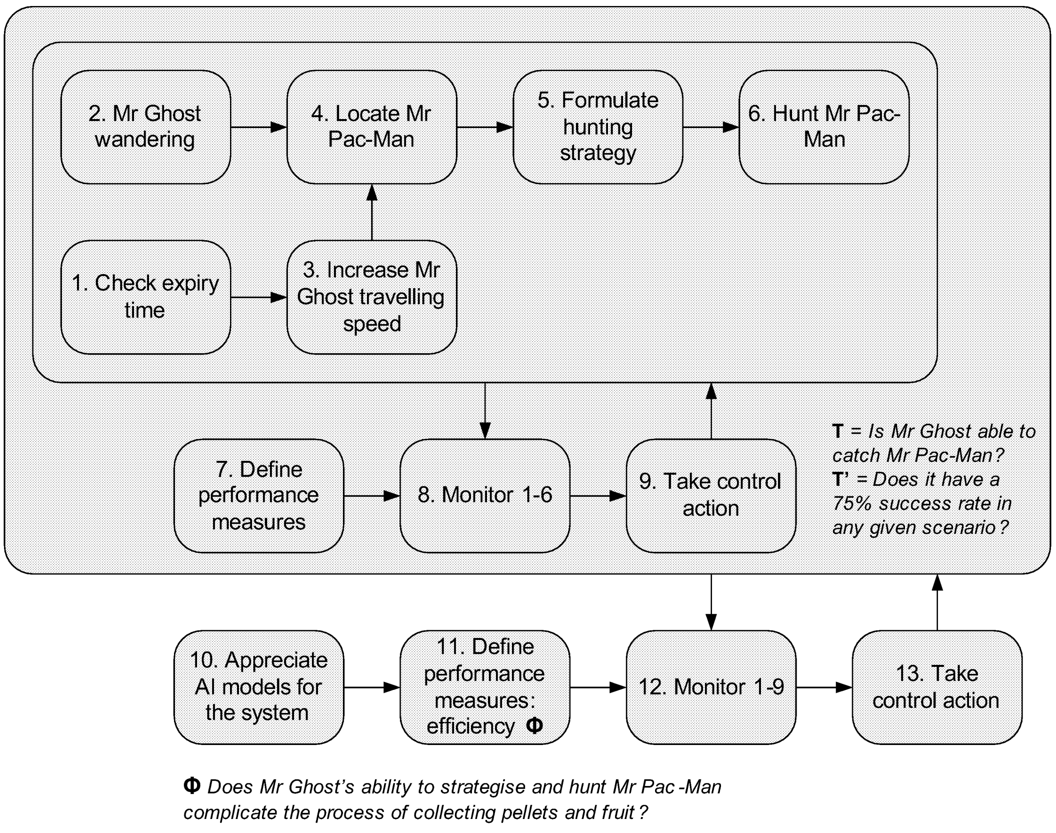

2. Mr Ghost wandering
4. Locate Mr Pac-Man
5. Formulate hunting strategy
6. Hunt Mr Pac-Man
1. Check expiry time
3. Increase Mr Ghost travelling speed
7. Define performance measures
8. Monitor 1-6
9. Take control action
T = *Is Mr Ghost able to catch Mr Pac-Man?*
T' = *Does it have a 75% success rate in any given scenario?*
10. Appreciate AI models for the system
11. Define performance measures: efficiency **Φ**
12. Monitor 1-9
13. Take control action
Φ *Does Mr Ghost's ability to strategise and hunt Mr Pac-Man complicate the process of collecting pellets and fruit?*

Figure 4: Final Conceptual Model for Intelligent Hunting System Root Definition

Considering the task of a game designer, designing conceptual models of the relevant activity systems may seem to overlap with the tasks of a game mechanics programmer who handles the technical components within computer game development relating to artificial intelligence (AI) (Bethke 2003). However, we should understand that conflicts arise from opposing behaviours within the social system. Therefore implementing opposing behaviours to the non-player character introduces conflicts to the game and thus increases the level of challenge to complete the level.

The 3 E's for measurement of performance allows the SSM practitioner to decide the optimal state for an activity system. This can also be used for fine-tuning the efficiency of the AI which has a direct relationship to the difficulty level of the game play.

Stage 5: Comparison of models and real world

The early stages of SSM begin with problem definition followed by a series of logical analyses to gain understanding about the actual problem area and formulate possible solutions. The output from each stage (root definitions, CATWOE analysis, conceptual model and measurement of performance) is then reviewed by comparing the conceptual models developed with the real world situation ensuring the activity systems are thoroughly analysed.

In game design context, not all activity systems modelled can be compared with the real world models as most were fictitious activity systems. However, game designers may exploit such an approach by investigating various models in the real world to see which can offer the best solution. We can then use the real world model as a benchmark to identify the differences in the conceptual model.

Considering the *intelligent hunting system*, game designers can use any hunter-prey scenario as a model to benchmark the conceptual model developed. Examples of some real world models are the wolf-pack hunting model and closed-exit hunting model illustrated in Figure 5 and Figure 6.

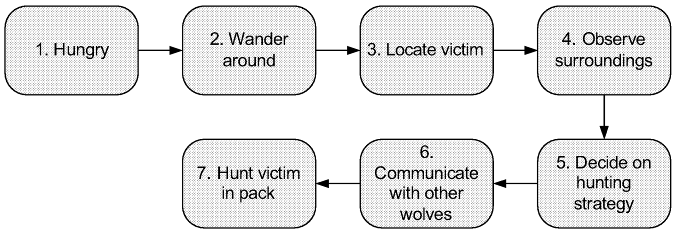


Figure 5: Wolf-Pack Hunting Model

The wolf-pack hunting model basically hunts in a pack led by the leader in an open area where the prey is most vulnerable. Though hunting in a pack can still be viable within complex networks of paths and hedges, the results of such would be different in Pac-Man due to the fact that only one entity can traverse any given space at any one time. This may seem like a group of Ghosts traversing the same path with different arrival time to each checkpoint.

However such a problem can be solved easily with some intelligence programmed into each of the Mr Ghost characters allowing routing decisions to be made whenever there are options available. The efficiency of the algorithm can be further enhanced using the closed-exit model similar to the exhaustive search algorithm with *n*, number of member assigned with individual route (Turban 1992).

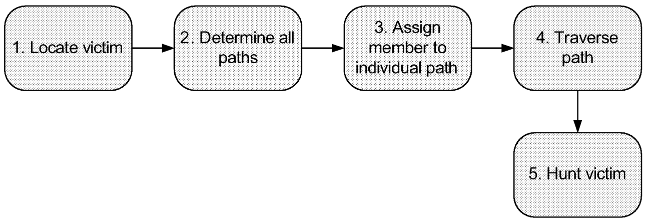


Figure 6: Closed-Exit Hunting Model

Comparison of various hunter-prey scenarios can provide insightful information which can then be used as recommendation for improving the conceptual model. Though this paper only discusses a subset of activity systems identified from the Rich Picture, there are other activity systems to be compared with its related real world model. A list of activity systems identified from Figure 2 with its relevant real world model is presented in Table 5.

Table 5: List of Activity Systems and Relevant Real World Model from Pac-Man

Activity Systems	Real World Model
Mr Pac-Man collecting fruit or pellet	harvesting scenario
Mr Pac-Man navigating through the maze	maze scenario
Mr Pac-Man is hunting Mr Ghost	prey-hunter scenario
Mr Ghost spawned	resurrection scenario

Stage 6: Changes - Systemically desirable, culturally feasible

In the seven-stage model, recommendations collected from stage 5 are used as suggestions for improving the real world situations. This should be based on the logic of the conceptual models developed in stage 4 by considering the systemic desirability and cultural feasibility of such changes proposed. However in the context of game design, the approach is reversed to yield a fun and entertaining simulation by using the real world situation logic to improve the conceptual models. Instead of considering the culturally feasible aspect to reflect on the changes made, we found that it is more logical to consider the technology aspect in this context.

Considering the intelligent hunting system, the conceptual model presented in Figure 4 may seem to be generalised, especially activity 5, *formulate hunting strategy*. The comparisons made may suggest that the closed-exit hunting model may logically seem to be more effective in comparison to the conceptual model while the wolf-pack hunting model is considered theoretically unsuitable in the scenario given. The improved conceptual model for the *intelligent hunting system* is presented in Figure 7.

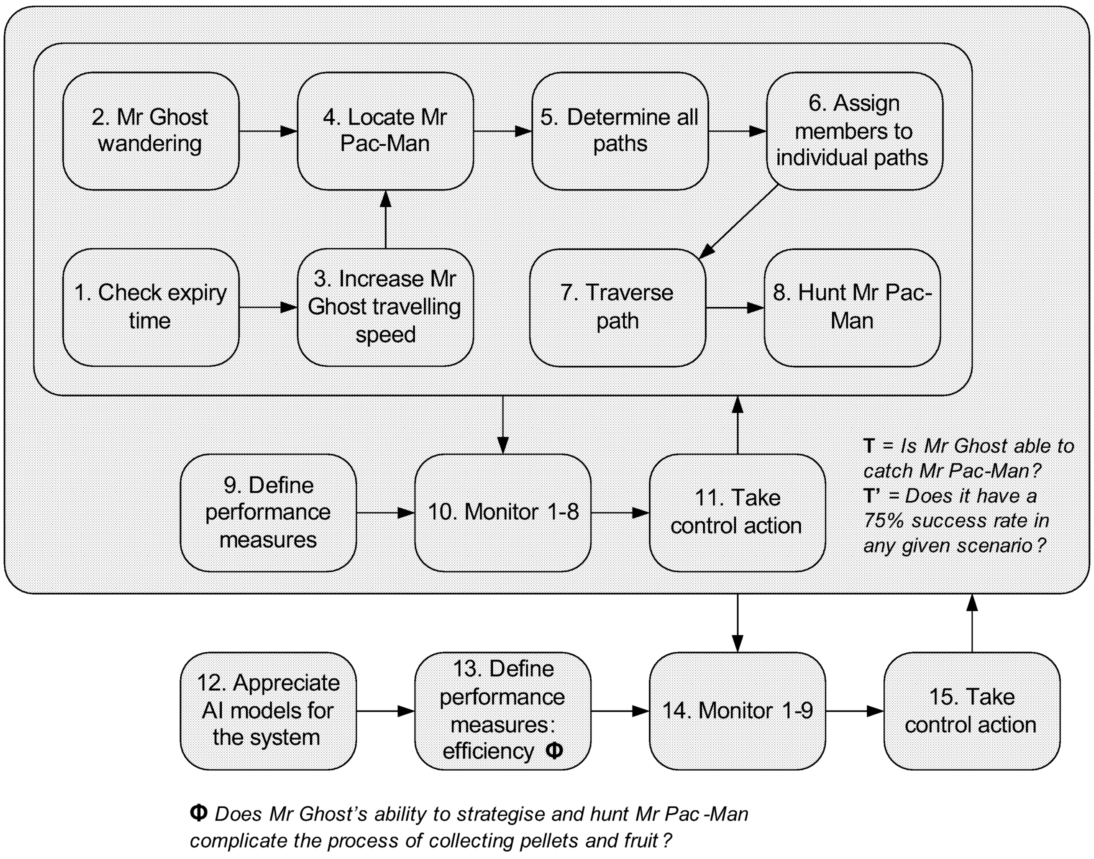

2. Mr Ghost wandering
4. Locate Mr Pac-Man
5. Determine all paths
6. Assign members to individual paths
1. Check expiry time
3. Increase Mr Ghost travelling speed
7. Traverse path
8. Hunt Mr Pac-Man
9. Define performance measures
10. Monitor 1-8
11. Take control action
T = Is Mr Ghost able to catch Mr Pac-Man?
T' = Does it have a 75% success rate in any given scenario?
12. Appreciate AI models for the system
13. Define performance measures: efficiency Φ
14. Monitor 1-9
15. Take control action
Φ Does Mr Ghost's ability to strategise and hunt Mr Pac-Man complicate the process of collecting pellets and fruit?

Figure 7: Improved Conceptual Model for Intelligent Hunting System Root Definition

Stage 7: Action to improve the problem situation

The final stage of SSM simply signifies the end of the methodological approach in developing understanding the problem area in order to propose improvements. In the conventional approach, it may require a few iterations with constant debates to logically analyse the problem area. These iterations aid in developing new perspectives, transitioning from questioning "whats" to "hows" and finally widening the debate and taking the appropriate action.

In the context of designing challenges and conflicts for computer games, the final stage represents the beginning of an exciting development of interactive entertainment. Game designers may undergo a few more cycles of the seven-stage model to gain more information and further refine the proposals of improvement thru logical analysis, prototyping, play testing and finally tweaks that will eventually balanced the challenges and conflicts posed before the design is being finalised.

Summary

Applying SSM in designing challenges and conflicts for computer games is different from conventional SSM practices. The norms may suggest defining a *Weltanschauung* that would improve the initial problem area; however our definitions of *Weltanschauung* propose statements which would complicate the tasks of gamers in achieving their goals and hence introducing challenges and conflicts. This focuses the conceptual mind of game designers to constructively think about the potential threats and later design the obstacles and related social conflicts to provide a set of interactions which defines the game play.

The research into the feasibility of using SSM in designing challenges and conflicts for computer games also introduces the design of appropriate intelligent agents of opposing models from the identified activity systems based on the root definition defined from CATWOE analysis. Though modelling artificial intelligence as part of designing the social conflicts may seem to be unnecessary, however such information can provide a clear representation of behaviours to be implemented later. The measure of efficacy, efficiency and effectiveness of artificial intelligence as part of the methodology can also provide an approximation to the intelligence to be incorporated into the non-player characters which can be tweaked to obtain the desirable level of difficulty to suit the gamers' needs.

The initial comparison mode within the seven-stage model proposed changes to be made onto the problem area by considering systemic and cultural issues was not applicable in our context. Therefore this comparison mode was substituted by appreciating related AI models and made the necessary changes onto the conceptual model based on the scenario. The seven-stage model inherits characteristics of the systematic approach; however the process is iterative thereby positively encouraging constant debate and logical analysis.

CONCLUSIONS

In this paper, we have introduced a methodological systemic approach, Soft Systems Methodology to gain understanding on an ill-structured problem area through an iterative process of logical reasoning. In this context, we exploit the structural and constructive methodology to aid game designers in designing better games by means of introducing appropriate challenges and conflicts within constrained situations where elaboration of an initial game idea is required. Though the methodology may seem to be a cumbersome approach, such a methodology can formally guide game designers to structure creative and innovative thinking thereby reducing the hours of endless search for truly novel ideas to fit into a game design.

Further research in this area is focusing on the design of a generic game design-oriented methodology based on systemic epistemology. We are also investigating the application of other systemic models such as Stafford Beer's Viable Systems Model (Beer 1984) and how this model of organisational behaviour can be applied to computer game worlds.

REFERENCES

Adams, E. and A. Rollings (2003). Andrew Rollings and Ernest Adams on Game Design, New Riders.

Beer, S. (1984). "The Viable Systems Model: Its Provenance, Development, Methodology and Pathology." Journal of the Operational Research Society **35**(1): 7-26.

Bethke, E. (2003). Game Development and Production, Wordware Publishing Inc.

Checkland, P. and J. Scholes (1999). Soft Systems Methodology in Action, John Wiley and Sons Ltd.

Crawford, C. (1982). The Art of Computer Game Design, http://www.vancouver.wsu.edu/fac/peabody/game-book/Coverpage.html.

Crawford, C. (2002). Chris Crawford on Game Design, Prentice Hall PTR.

IDSA (2002). Essential Facts About the Computer and Video Game Industry, Interactive Digital Software Association (www.theesa.com).

Kendall, P. A. (1997). Introduction to Systems Analysis and Design: A Structured Approach, McGraw-Hill Education.

Nairne, J. (2003). Psychology: The Adaptive Mind, Wadsworth.

Rouse, R. (2001). Game Design Theory & Practice, Wordware Publishing Inc.

Spector, W. (2003). Sequels and Adaptations: Design Innovation in a Risk-Averse World. Game Developers Conference (GDC 2003).

Turban, E. (1992). Expert Systems and Applied Artificial Intelligence, Macmillan USA.

ADAPTING MAINSTREAM MULTIMEDIA GAMES FOR SEVERELY VISUALLY IMPAIRED CHILDREN

Dominique Archambault(1), Aurélie Buaud(1,2),
Sylvain Lerebourg(1,3) and Damien Olivier(3)

(1)INSERM U483 / INOVA — Université Pierre et Marie Curie
9, quai Saint Bernard — 75 252 Paris cedex 05 — France

(2)Laboratoire de recherche en Génie des Systèmes Industriels — ENGSI-INPL
8, rue Bastien Lepage — BP 647 — 54 010 Nancy — France

(3)Laboratoire d'Informatique du Havre EA 3219 — Université du Havre
25, rue Philippe Lebon — BP 540 — 76 058 Le Havre cedex — France

Dominique.Archambault@snv.jussieu.fr, Aurelie.Buaud@snv.jussieu.fr,
Sylvain.Lerebourg@univ-lehavre.fr, Damien.Olivier@univ-lehavre.fr

KEYWORDS

Multimodality, Visual Disability, Specific Peripherals, Adaptation.

ABSTRACT

This paper describe several issues that have to be addressed to adapt existing mainstream computer games for severelly visually impaired children. These issues are illustrated with case studies. Most of the work presented here was done within the framework of the TiM project (*Tactile Interactive Multimedia computer games for visually impaired children*).

INTRODUCTION

The mainstream commercial market for computer games and other multimedia products is extremely large and sighted children have a considerable experience of such games. Children who cannot use the ordinary graphical interface, because they are totally blind or because they have a severe visual impairment (sight rated < 0.05), do not have access to this important part of the youth culture.

Most of those games, containing spoken comments, with musical accompaniment and sound effects, offers rich audio contents and interesting scenarii, but they are usually not accessible at all to our group of children since they are based on visual feedback.

Nevertheless it seems very valuable to adapt existing mainstream games for severely visually impaired children. Indeed large amount of multimedia resources can be reused. Another big interest is that it allows visually impaired children to play with the same games than their sighted peers (Hildén and Svensson, 2002).

This paper will describe several very important issues that have to be addressed to adapt existing mainstream computer games for severely visually impaired children. These issues will be illustrated with two case studies.

The first case study concerns the adaptation of a French educational and recreational CD-Roms named *"L'Univers de Pomme d'Api"* (vol. 1, 2 & 3) (Archambault and Burger, 1999; Archambault and Burger, 2000). The main feature of these CD-Roms is the large amount of recorded sounds, like short stories, explanation of words, words said in several languages, simple question game and music. The adapted game was evaluated by 4 blind children in their families, and in 2 special schools for blind children, and then distributed in France through an association.

The second case study concerns the adaptation of the game *"Reader rabbit's Toddler"* which exists in French and in English (Buaud et al., 2002). This game is composed of 9 different activities from which 4 could be adapted. A rich tactile overlay was designed, using pieces of various materials. After testing with several children, a large amount of text were written and recorded to make the game really interesting to the children despite the missing visual information.

This game was adapted and the adaptation evaluated within the framework of the TiM Project (*Tactile Interactive Multimedia computer games for visually impaired children*) (Archambault et al., 2001), which main goal is to offer visually impaired children the possibility to play computer games independently, that is without the assistance of a sighted person. The French version is now about to be distributed and an English version should follow in the next months.

ADAPTATION OF GAMES

One great challenge is to adapt and even to extend the sensory-motor capabilities of computer systems to better match the natural communication channels usable by visually impaired children. Thus, how can we adapt different game situations, designed with specific data fitting visual modalities to other modalities, using audio or tactile channels. In other terms how can we convert multimodal expressions to semantically equivalent expressions, fitting a different set of modalities.

How do visually impaired people usually access to computer applications?

Visually impaired people can only access information via specific modalities, mainly using audio and tactile channels. They use dedicated software, called "screen readers", that allow to render the content of computer screen using speech synthesisers or Braille devices. These software are mainly designed to give access to office applications, like word processor, web navigation or email.
Additionally partially sighted users may be able to use the computer screen via another kind of software allowing specific settings of screen displays that fit their visual possibilities (for instance very large fonts for users with amblyopia; or on the contrary very small fonts for users with a reduced field of vision – like 'tunnel' vision; adapted contrast or colours; no animation for users who need some time to perceive still images).
Ordinary input devices can be used (like standard keyboard, mouses) or specific Braille keyboards. In the case of games we use ordinary joysticks and gamepads, but also tactile boards for children who can hardly use a keyboard (because they are too young or because they have additional disabilities). Special devices like switches or move detectors can also be used.
Most of mainstream computer games do not work with standard screen readers for many reasons. The main reason is that games usually contain a lot of graphical information that is impossible to render automatically using alternative modalities.

Objective of an adaptation

Then to make these games usable by our group or children it is often necessary to find an alternative way of interaction for each game situation, allowing the use of the specific devices. It seems obvious but it is very important that adapted games still be games. Indeed, adults in work situation accept relatively big constraint on usability to be able to use the same software as their sighted work mates and to work on the same documents. This is not the case with children, especially playing. In other terms it is not enough to find a technical way allowing to access to all information needed in the interface, the result must be as interesting and as usable as the original game.
Additionally the visual display allows to give access quickly to a large amount of information which can be dynamical. Usually sighted users select the relevant information at a glance. In the case of games based on fast reaction from the player, trying to give a description of the content of the screen (which is basically the function of screen reader software) would not be efficient enough to allow the player to react fastly enough.
Another constraint is quite important in our case: an adaptation should keep the philosophy or the goal of the original game. It is sometimes possible to use some existing resources to make a game slightly different than the original one, because the goal of the original game is basically visual, but in that case it cannot be called and adaptation of this game. Indeed adapting any kind of content to another modality means to give access to the same content. Then it is often not possible to make a relevant adaptation to all parts of a game.

INTERACTION PROCESSES

The game interaction processes must be modified in order to fit the modalities that can be used by visually impaired children, and especially focussing on their abilities.

Lack of visual information

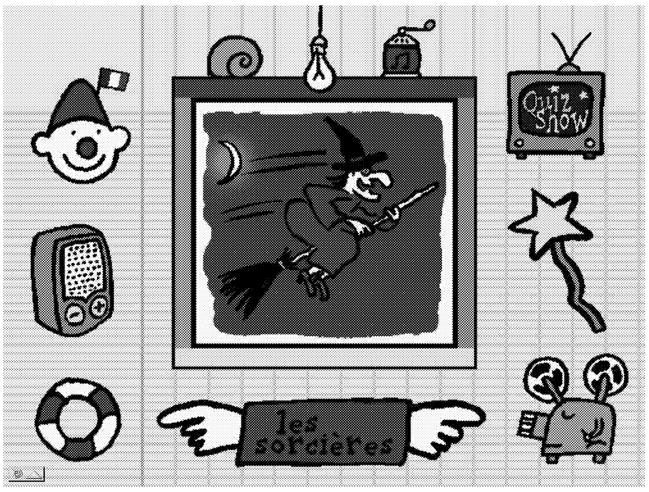
As pointed out earlier, most multimedia games are designed to use visual feedback as the main way of communicating information to the player. This leads to lack of information that have to be replaced using another modality.
A first case is when the feed-back following a success in a part of the game is mainly visual. If these "rewards" are missing, the game might be less interesting for the players. In that case it is necessary to design additional audio feedback so that this missing part is replaced. For instance in the game *"Reader Rabbit"*, new texts have been written and recorded with actors.
In other cases only a part of the original game can usually be reused. Then the content of the game might seem short. In the case of the game *"L'Univers de Pomme d'Api"*, we have put together the adaptation of 3 games of the same series in one game so the volume of the adapted game (that is the average duration necessary to a player to complete the game) is equivalent to one of the original ones.

Modification of the interaction processes

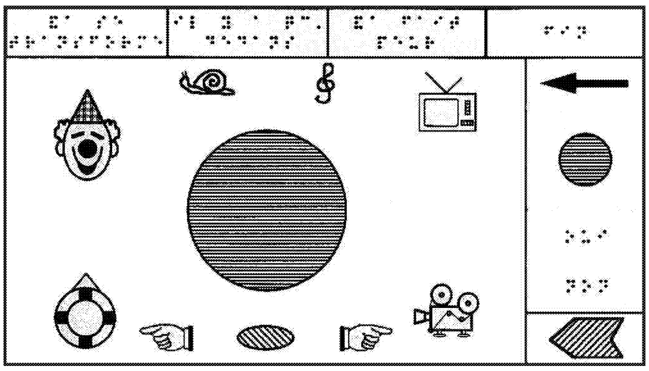
The modification of the interaction processes often leads to the necessity to redesign important parts of the contents. For instance when using tactile boards, it is not relevant to have help messages saying "click on something". First because the action will be press and not click, and for very young player it might be important that the guideline is relevant. Another reason is that tactile board are used together with overlays on which are sticked different objects that are static, that is the objects will always be the same during all the game since on the screen they may change.
Then most of those help messages have to be changed, indeed even messages which are not necessary to change must be recorded with the same voice.

The original graphical interface for the game *"L'Univers de Pomme d'Api"* (see figure 1) has an important static part that was simply reproduced tactily (all the symbols around the central images, they correspond to commands in the game). Nevertheless it was necessary to change all the guidelines recorded voices. Then additional controls are on the tactile overlay. For instance the cells on the top allows to select one of the 3 different games of the series (see above), the right column is used only a part of the game – a quiz.

In the game *"Reader Rabbit"*, the selection between the different activities uses 2 buttons on the tactile sheet, one goes trough the different activity titles (they are said, with an explanation of the game) and the other is used to enter the selected game, which is completely different from the original game on which the title/explanation is given when the mouse is over a symbol on the screen, and the player has to click on this symbol to enter the activity.



(a) The original graphical interface



(b) The tactile adaptation

Figure 1: *"L'Univers de Pomme d'Api"*

Relevance of sounds

The relevance of sounds is a very big concern. We have observed that a lot of sounds are understood very easily thanks to the context, usually provided by the visual output. For instance if a lion is on the screen, even a very bad roaring sound record will be eared as the roaring of a lion. But the same audio file played without visual help cannot be interpreted. In that case there are 2 possibilities.

One is to replace the sound by another, which would be more relevant. In the case of the game *"Reader Rabbit"*, we replaced the sounds of animals by recordings of real animals. But this is not always enough because it is not obvious that a blind child may make a difference between the roaring of a lion and of a tiger. In that case it was relevant because there were no ambiguity between the different animals.

This leads to the second way of handling that problem, which can be combined with the first one. Using other modalities, we have to provide some additional information. This information should be designed in order to provide a context that will give to the child enough clues to recognise the different sounds.

CONCLUSION

Actually in our both case studies it has been necessary to implement new software and in the case of *"Reader Rabbit"* to design new contents. These to games use the sound and some of the graphical resources of the original games but new code had to be implemented.

In the context of the TiM project, a specific API was designed, called *Blindstation* (Sablé and Archambault, 2003a), that allows to develop easily software able to drive the specific devices. In particular a programming library was designed to allow transparent access to various Braille devices via a unified API and a system of driver implementing each device protocol (*libbraille* (Sablé and Archambault, 2003b)). These APIs are available according to the Open-Source model. The games script are implemented with the *Python* language and are using the objects of the *Blindstation*.

After the experience of adapting existing games, we can observe that the cost is higher than developing specific games, because most of the audio resources have actually to be redesigned. But it is very important anyway because it allows blind children to share their feelings and experiences of playing with their sighted peers.

Another option is to design a specific game, reusing only a concept, ensuring that this game will be also interesting for the sighted, in a design for all perspective. For instance we developed a game, called *"Mudsplat"*, which is based on the concept of Arcade shooting games (like *"Space Invaders"*), but this is a different story...

ACKNOWLEDGEMENTS

The **TiM** project is are funded by the European Commission[1], on the program IST 2000 (FP5/IST/Systems and Services for the Citizen/Persons with special needs), under the references IST-2000-25298.
We thank the companies *'Bayard Presse"* and *'Ubi Soft"* for allowing the adaptation of *'L'Univers de Pomme d'Api"*, and the company *'Mindscape"* for allowing the adaptation of the game *'Reader rabbit's Toddler"*.

References

Archambault, D. and Burger, D. (1999). Development Tools for Non Visual Education Applications. In *Proc. CSUN'99 (14th Annual Conference 'Technology and Persons with Disabilities')*, Los Angeles, California, USA.

Archambault, D. and Burger, D. (2000). TIM (Tactile Interactive Multimedia): Development and adaptation of computer games for young blind children. In *Proc. ERCIM WG UI4ALL & i3 Spring Days 2000 Joint workshop, Interactive Learning Environments for Children*, Athens, Greece.

Archambault, D., Burger, D., and Sablé, S. (2001). The TiM Project: Tactile Interactive Multimedia computer games for blind and visually impaired children. In Črt Marinček, Bühler, C., Knops, H., and Andrich, R., editors, *Assistive Technology – Added Value to the Quality of Life, Proceedings of the AAATE'01 Conference, Ljubljana, Slovenia*, pages 359–363, Amsterdam, The Netherlands. IOS Press.

Buaud, A., Svensson, H., Archambault, D., and Burger, D. (2002). Multimedia games for visually impaired children. In Miesenberger, K., Klaus, J., and Zagler, W., editors, *Proc. ICCHP 2002 (International Conference on Computers Helping People with Special Needs)*, volume 2398 of *LNCS*, pages 173 – 180, Linz, Austria. Springer.

Hildén, A. and Svensson, H. (2002). Can All Young Disabled Children Play at the Computer. In Miesenberger, K., Klaus, J., and Zagler, W., editors, *Proc. ICCHP 2002 (International Conference on Computers Helping People with Special Needs)*, volume 2398 of *LNCS*, Linz, Austria. Springer.

Sablé, S. and Archambault, D. (2003a). Blindstation: a Game platform adapted to visually impaired children. In Craddock, G., McCormack, L., Reilly, R., and Knops, H., editors, *Assistive Technology – Shaping the future, Proceedings of the AAATE'03 Conference, Dublin, Ireland*, pages 232 – 236, Amsterdam, The Netherlands. IOS Press.

Sablé, S. and Archambault, D. (2003b). libbraille: a programming library for easy access to Braille displays. In Craddock, G., McCormack, L., Reilly, R., and Knops, H., editors, *Assistive Technology – Shaping the future, Proceedings of the AAATE'03 Conference, Dublin, Ireland*, pages 435 – 440, Amsterdam, The Netherlands. IOS Press.

[1] The contents of this paper is the sole responsibility of the authors an in no way represents the views of the European Commission or its services)

OTHER PAPER

M-LEARNING LOMS

Jeanne Schreurs, Maarten Steegmans, Rachel Moreau, Johan Brouns
Limburgs Universitair Centrum
Universitaire Campus
3590 Diepenbeek.
Belgium.

jeanne.schreurs@luc.ac.be
maarten.steegmans@luc.ac.be
rachel.moreau@luc.ac.be

KEYWORDS

M-Learning, Mobile learning, Learning Management System, Learning Objects Management System.

ABSTRACT

The centralization of the learning objects in a warehouse was the first step in organizing the learning objects independent from the LMS being an important requirement as part of the SCORM standard. The warehouse system facilitates the management of the learning objects (learning material and metadata).
In this paper we describe the development of the extension to our warehouse system. The main goal is to increase the distribution possibilities for learning materials, based on the communication using mobile devices.
M-learning is part of the blended learning solution. That is why it is a challenge for us to develop a system supporting both mobile devices as well as non-mobile devices.
Mainly it is our goal to rebuild the warehouse so that it will still better fit the requirements of e-blended learning courses. Furthermore there will be a distribution of learning objects from the warehouse by the way of both mobile devices and desktop computers. During the development we examined the possibilities and limitations of the different devices to make more flexible learning processes for the students.

1. INTRODUCTION

The organization of knowledge in knowledge objects is a hot topic in knowledge management research. Knowledge objects have been stored in a warehouse and will be managed by a knowledge warehouse management system.

Learning content is organized as learning objects. Learning objects have been defined as learning content items, in which the content document itself is included and the characteristics of the document are formulated as metadata. By this way learning content has been organised and managed in a knowledge warehouse and has been made accessible for learners. The individual learner or the teacher developing a course can select those learning materials fitting their requirements.

For mobile workers, mobile learning or m-learning is becoming popular.
Mobile learning means the provision of education and training courses on wireless devices. PDA's (personal digital wireless devices), palmtops and mobile telephones.
With a Palm handheld, a teacher, student or administrator can do amazing things: take notes, calculate, sketch ideas, collect data, access resources, manage activities and, with the right hardware, even access the Internet wirelessly.
Handheld computers offer unique benefits to students and teachers. Students have a personal, portable device ready-at-hand for individual or collaborative learning activities, wherever they go. Students can use handheld computers to collect data in the field, to learn vocabulary words while waiting to be picked up after soccer practice, or to self-quiz during a long car ride."

Our learning warehouse management system has now additional functions and is open for PDA access by the students

2. MOBILE LEARNING AND MOBILE DEVICES

The centralization of the learning objects in a warehouse was the first step in organizing the learning objects independent from the LMS in which the learning process has been modeled and in which linkages have been created with the learning documents in the warehouse. This is one important requirement as part of the SCORM standard.
The warehouse contains learning objects (learning material + metadata), tutoring documents and student documents.

The learning content of the learning objects has been extracted from external sources or has been created using authoring systems.
The LMS will include a learning process model including linkages with learning objects and tutoring docs.
LearningSpace 5 is the LMS we are using. In the teacher interface we can create the entities of the learning process model. The entities can be linked with the warehouse documents

The advantages of mobile learning refer mostly to the mobile user: a great flexibility, an improved learning schedule is possible, increased productivity during dead moments and just-in-time learning. But also for the non-mobile user is the usage of mobile devices useful in a learning process. For example the supporting and communicative functions of a forum.

On the technological side we have seen some developments who can give a boost to mobile learning. On the one hand there is the upcoming usage of GPRS and UMTS, this makes it possible to send/receive data at a higher speed. On the other hand there is a large increase of functionality and usage of mobile devices. One of the last developments is the support of TCP/IP, http-protocols within WAP2.0. This makes mobile internetting possible and gives access to general webpage formats. Nevertheless they have very small screens; limited memory capacity and the large diversity of mobile devices obstruct a good learning experience.

On the educational field we notice that the learning materials must answer to specific conditions. We will have to get around the technical restrictions so we can create a good learning experience. The usage of video, audio, clear interfaces and divided courses must contribute to this.
Furthermore we will have to adapt the content to the needs of the mobile user. Because he has a very fragmentized time schedule, we will have to be sure that the learning object are not to long. Dividing the knowledge in smaller modules offers a solution.

With a Palm handheld, a teacher, student or administrator can do amazing things: take notes, calculate, sketch ideas, collect data, access resources, manage activities and, with the right hardware, even access the Internet wirelessly.

Twenty-five years of research on desktop computers has shown that, when used appropriately, technology can have a beneficial impact on teaching and learning, The PEP evaluation study found that handheld computers can offer unique benefits to students and teachers. Students can have a personal, portable device ready-at-hand for individual or collaborative learning activities, wherever they go. Students can use handheld computers to collect data in the field, to learn vocabulary words while waiting to be picked up after soccer practice, or to self-quiz during a long car ride.

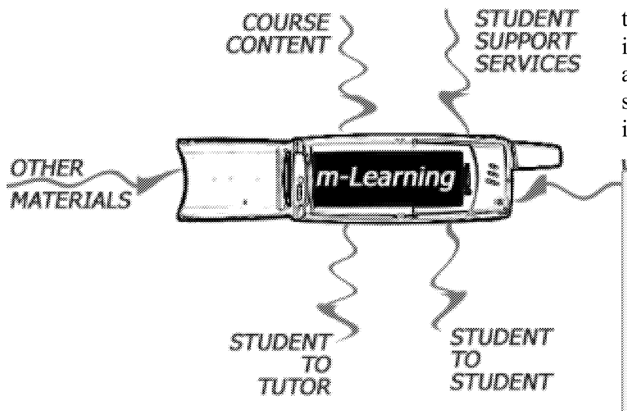


Figure 1: Wireless Virtual Learning Environment of Tomorrow

3. A WAREHOUSE MANAGEMENT SYSTEM TO SUPPORT M-LEARNING

The centralization of the learning objects in a warehouse was the first step in organizing the learning objects independent from the LMS in which the learning process has been modeled and in which linkages have been created with the learning documents in the warehouse. This is one important requirement as part of the SCORM standard.
The warehouse contains learning objects (learning material and metadata), tutoring documents and student documents.
The learning content of the learning objects has been extracted from external sources or has been created using authoring systems.
The LMS will include a learning process model including linkages with learning objects and tutoring docs.
LearningSpace 5 is the LMS we are using. In the teacher interface we can create the entities of the learning process model. The entities can be linked with the warehouse documents

The main objective of this system is to expand the distribution possibilities for learning materials and communication recourses to mobile devices. M-learning is a part of blended learning; this offers a challenge to develop a system not only for mobile devices but also for non-mobile devices.
Mainly it is the intention to extend a warehouse for better support of e-blended courses. Furthermore there will be a distribution of learning objects from the warehouse by the way of mobile devices and desktop computers. During the development we examined the possibilities and limitations of the different devices to make more flexible learning processes for the students.

During the past couple of years there has already been developed a learning management system, this is the foundation for the e-blended learning system. In the current system, the physical storage will be split in the document on the one hand and the metadata on the other hand. The metadata is stored in a data table, while the document is categorized in a hierarchical directory structure.
To increase the flexibility of the learning sessions we have extended the system with another layer. We call this layer the Studentwarehouse, it allows the student to consult information and to share information with others. This is also the layer we made available for mobile learning. The students now have the possibility to consult and share information anywhere and anytime.

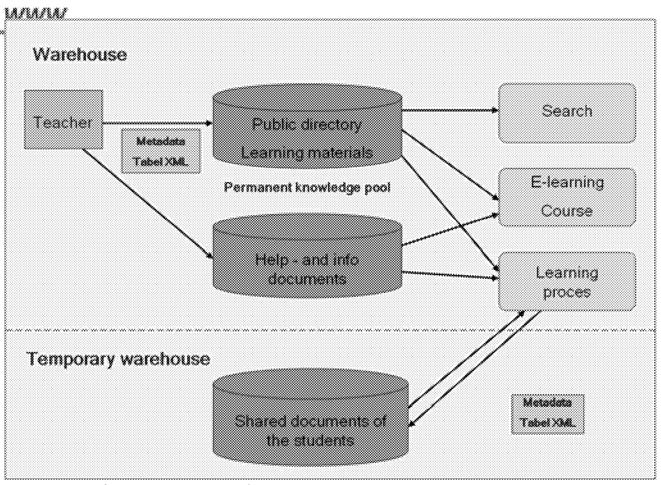


Figure 2: a warehouse management system

3.1. FUNCTIONALITY AND REQUIREMENTS

The intention of this system is mainly to make information accessible for students and by students, regardless of place and time. By making the studentwarehouse available for mobile learning, the need for flexibility can be satisfied.

To increase the flexibility of the student, we have implemented the next parts to the learning environment:

- an upload for documents: the students must have the possibility to upload files trough a webpage. Now the students can use each others learning expertise and knowledge in the learning process. There is also a possibility to add metadata; this makes it easier to consult the documents.
- automatically saving of the document and metadata in the warehouse: the uploaded files must be saved in the warehouse. Depending on the possibilities of the devices, it is safer to convert all the files to a known format (supported by each device).
- The consulting and downloading of the learning objects from each of the different devices.
- Dynamically generation of an document page: if students want to consult or download files, the list of these document must be up to date. This list will be automatically generated with an asp-page and it will be always up to date.

Besides this it is also important to meet the next requirements:

- Flexibility : the student must have access to the learning objects at any place, at any time with every device.
- Easy to use: the whole system must be easy , logical and usable for non-technical people.
- Suitability: We also have to take account of the limitations of each device. The use of large documents is not practical with a mobile device. The applications must be adapted to medium that will be used
- Re-usability: People attach more and more significance to the re-usability of learning materials. So the metadata must be saved in a database. The documents can then be found with a search on the metadata.

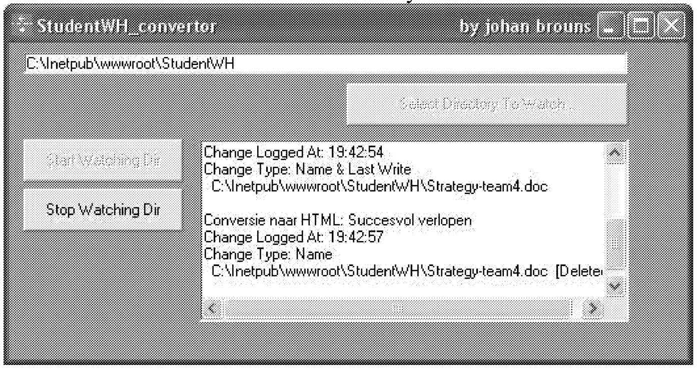
3.2. SYSTEM FUNCTIONS

The system is divided in 7 functions:

- An index page to identify the communicating device: When a student enters the index page, there will be checked witch webbrowser he is using. Since each operating system has its own version of a webbrowser, it will permit us to identify the specific device and to show an altered version of the webpage.

- An upload page for documents and learning objects: The upload of documents in a warehouse makes it possible to share documents with others. Trough a webpage the student can browse to the file that he wants to share, he also has to enter a title and subject and some extra information. This is the metadata that will be saved in a database, so it will be easier to search for the document

- The converting and saving of the documents in the warehouse: the keep the documents readable for each mobile device and also for the re-usability of the learning objects, we will convert them in a definite format. This will be done by an application before the documents are placed in the warehouse. The documents will be converted to the html-format. The application will keep an eye on the directory were the documents will emerge when they are uploaded. When a file is uploaded, the application will convert the document to html and place it in another directory. The original file will also be saved in another directory.



```
C:\Inetpub\wwwroot\StudentWH
```

```
Change Logged At: 19:42:54
Change Type: Name & Last Write
  C:\Inetpub\wwwroot\StudentWH\Strategy-team4.doc

Conversie naar HTML: Succesvol verlopen
Change Logged At: 19:42:57
Change Type: Name
  C:\Inetpub\wwwroot\StudentWH\Strategy-team4.doc  [Delete
```

Figure 3: the converting application

- The dynamic generation of the document page: This page is written in asp, it will watch a directory and every time when the page is called it will make a list of all the files that are present in the directory. So every time you call the page it will be up to date with the latest files.

- The consulting and downloading of documents: We are saving 2 different versions of the uploaded documents. On the one hand we have the original files; on the other hand we have the converted html-files. This makes it necessary to have two document pages, one with the html-files and another with the original files. The page with the html files will be accessible for each device, mobile and non-mobile.

- The forum: The communication between students and the teacher is a very important function within the learning process. A forum is one of these communication forms. Providing that it is well implemented, it will increase the interactivity between the teacher - students and between the students mutually. It allows the student to ask questions, discuss the material, and explain assignments and so one. The usage of a forum is very important for a mobile student. Because he is mostly on the road he can ask his questions on the forum.

- Help function: Because some people have less knowledge of informatics then others. It is really necessary to have a help function. Every part of the Studentwarehouse is explained here. This is of course the same for the mobile user and non-mobile user.

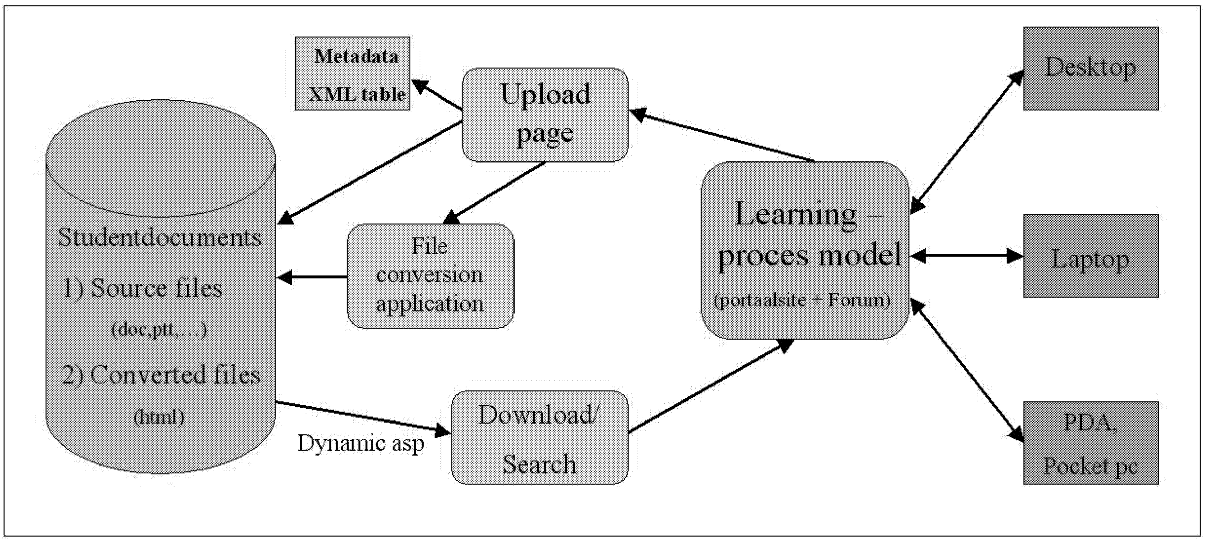


Figure 4: Studentwarehouse layer

CONCLUSIONS

We have extended an existing e-blended learning system with an extra layer, to offer a more flexible and more effective learning process to both mobile and non-mobile students. This by giving them the possibility to consult learning materials, to share and to put a supporting communication applications at there disposal where, when and with each device the student likes.
With this we tried to take into consideration the properties of a well structured learning environment and to come up to the limitations of the different devices. In this project we rather approached m-learning as a new distribution medium than as a new learning format. It's in fact a e-learn-application that's converted to a mobile format. Nevertheless this way of working gives us a larger flexibility, this is why there is an increase of learning possibilities. This form of m-learning is a part of the blended mix that is harmonized with the needs of the mobile student.

The main objective was to make learning material and information accessible for the mobile students and this with every possible device. Because this is no problem for the conventional devices, we focused or attention to the mobile devices. The learning materials of the studentwarehouse are accessible from every device. The uploading and converting of the documents elapses entirely automatic, this makes it possible for the students to easily add documents to the studentwarehouse without the need of a teacher or programmer.

This project shows us that mobile devices can extend a leering environment and increase the flexibility within the learning process.

REFERENCES

Intel, Mobile PCs and Wireless:Business Users Make the Productivity Connection, Intel Corporation, 2002

Leonardo project, From e-learning to m-learning, The book. Projectsite: http://learning.ericsson.net/leonardo/index.html

Rekkedal, T., Trying Out a Learning Environment for Mobile Learners, Nettskolen, 2002

Scheurs, J., R. Moreau, I. Picart, E-learning : New advanced learning models and the limits of our e-learning systems, Euromedia 2002, ISBN 90 77039 05 8, 2002a

Schreurs, J., R. Moreau, I. Picart, Blended learning and live sessions in lifelong learning, ICL Workshop: 2002 Blended learning ,ISBN 3-933146-83-6, 2002b

Schreurs, J., R. Moreau, I.Picart, Building an e-blended learing course using learning objects, 2003

Ultralab, Technology Watch rapport, m-learning project, IST-2000-25270,2002

AUTHOR LISTING

AUTHOR LISTING