# TOWARDS STATISTICAL CLIENT PREDICTION – ANALYSIS OF USER BEHAVIOUR IN DISTRIBUTED INTERACTIVE MEDIA

Aaron McCoy*, Declan Delaney^, Dr. Seamus McLoone* and Dr. Tomás Ward*
*Department of Electronic Engineering, ^Department of Computer Science
National University of Ireland Maynooth
Maynooth, Co. Kildare
Ireland
E-mail: amccoy@eeng.may.ie

**KEYWORDS**

Distributed interactive media, Torque game engine, Time-series data analysis, Behavioural modeling, Statistical client prediction, Networked multiplayer games, Dead-reckoning.

**ABSTRACT**

Distributed interactive media such as networked multiplayer computer games offer users the opportunity to interact and share experiences within a virtual environment. More often than not, these interactions are required to be performed in real-time, a constraint which poses problems given the underlying network capabilities used to transmit information. In these real-time distributed systems, the amount of information that needs to be shared between participants in order to maintain complete game-state fidelity is too large. As a result, trade-offs must be made over what information requirements are necessary to maintain a level of consistency that will provide adequate quality of interaction for the users. One possible solution to this problem is the use of statistical modeling techniques that attempt to capture the individual behaviour of system users. These models can then be used to predict the likely future behaviour for the users, thus reducing the shared information requirements. In this paper we present some preliminary analysis of the behaviour of users within a distributed interactive application, with a goal towards future work of attempting to develop and incorporate statistical models of user behaviour for the purpose described above.

**INTRODUCTION**

Distributed interactive media (DIM) such as networked multiplayer games are prone to quality-of-interaction and scalability problems as a consequence of non-ideal communication infrastructure characteristics such as network latency and bandwidth. This well-known problem is generally dealt with through careful entity state update procedures that filter the game-state based on criteria such as client relevancy or state changes. An example of the former is area-of-interest techniques (Singhal 1999) while examples of the latter are delta-compression (Van Hook et al. 1994) and dead-reckoning (IEEE 1993). In addition QoS techniques are sometimes used to ensure that all participating clients are given sufficient network resources to meet quality-of-interaction criteria (Internet2 2004). The communication of game state changes is the key issue in all of the above and the games industry-standard techniques are based chiefly on variants of dead-reckoning, which is an example of an entity state extrapolation mechanism. Rather than updating entities over the network once per simulation loop (which we will refer to as the game loop) all clients in the DIM maintain a local model, usually a linear extrapolation, of entity dynamics. This model is only updated when the client responsible for the entity determines that the difference between the true entity state and that of the model as used by all other clients has deviated by some pre-defined threshold amount. Only this update then needs to be transmitted to all the participating clients hence reducing the number of packets required to maintain a tolerable fidelity across the DIM.

Such a technique obviously helps reduce bandwidth requirements and therefore aids in scalability. More subtly it also aids in the reduction of communication latency, one of the key factors in maintaining a high quality of interaction for the user. This is apparent if we look at the individual components which make up latency in distributed interactive media for any particular link between two participating clients $i$ and $j$ :

$$\mathrm{T}_{ij} = \tau_c + \frac{K_{ij}}{B_{ij}} + O_{ij} \qquad (1)$$

where K represents the generation rate of state information during the global gameloop in bits per second and B is the bandwidth of the link to the particular client in question. $\tau_c$ represents the physical propagation delay. O represents all other processing overheads. Obviously through an increase in link bandwidth or a reduction in K the latency can be reduced. It is through such information rate reduction techniques such as in dead-reckoning that latency problems can be dealt with in DIM.

Recently in an attempt to further improve the power of entity state update mechanisms using the concept of state extrapolation, a technique known as the hybrid model approach has been proposed (Delaney et al. 2003). This concept is very powerful and yet quite simple. In this paradigm entity state are extrapolated based on a combination of low order short term extrapolation and longer term statistical inference. Essentially the technique relies on extrapolating state changes based on previous examples of state behaviour in similar circumstances. In the absence of good information on typical state changes for an entity, the model is switched to simple low order extrapolation as in dead-reckoning. By switching between the two models, hence the term hybrid model, entity states can be extrapolated further than currently possible under dead-reckoning and other entity state extrapolation schemes. Further, in the absence of good statistical information, long-

term heuristic models can be used in lieu as provided by the DIM designer. Consequently the technique has superior performance to dead-reckoning alone as demonstrated in (Delaney et al. 2003). However for the technique to realize its potential in the field of DIM it is imperative that techniques and methods should be developed to first of all recognize, and second of all represent, such statistical models. All work in this area so far has concentrated on demonstrating the concept for fixed statistical spatial models that naturally arise out of static navigation tasks in typical DIAs (Marshall 2004). We are currently studying the possibility of automatic recognition of statistically similar behaviour among entity dynamics in important classes of DIM.

In a previous paper we have shown that patterns of behaviour emerge for human-human interaction in such DIM (McCoy 2004). However, in an attempt to bridge the more solid statistical categorization that arose out of fixed spatial models with such patterns, an intermediate study has been conducted in which the reactive coupling between human agents has been loosened through the investigation of interaction between human users and finite-state machine-driven agents (BOTS). This paper reports on work done on this area so far in exploring such human user behaviour in these multiplayer computer games. It is hoped that through the analysis of such behaviour that categorization can be determined with the goal in mind that statistically similar patterns of behaviour can be recognized and ultimately exploited to predict future behaviour in order to pre-empt gamestate changes ahead of time and so reduce latency problems in such DIM through both update packet rate reduction and perhaps pre-emptive transmission.

## TEST ENVIRONMENT AND TEST SCENARIOS

We use the Torque Game Engine from GarageGames (Marshall 2004) to construct simple test environments and test scenarios. This allows us to perform experiments in a relatively controlled manner, and provides us with the ability of recording information. During a user's interaction within the test environment, various data is time-stamped and collected in a log file for subsequent analysis. Firstly, 3-dimensional positional data for each user is recorded at regular sampling intervals (at the rate of at least 1 Hertz, but usually higher), allowing us to reconstruct a user's positional state over time with respect to that of another user and any events that occur. This positional data is represented as time-series datasets for analysis, examples of which can be seen in the results and data analysis section. Secondly, direct interface control interactions (i.e. keyboard and mouse button presses) performed by the user to control their player onscreen are time-stamped and recorded whenever they occur (this consists of primarily movement and weapon firing instructions). This allows us to reconstruct a user's control sequence of actions with respect to any events that occur. Given the limited complexity of our test environment described below, the only events which we are concerned with here are both weapon firing events and disabled events (where one user is disabled by weapons fire from another), and these are recorded and time-stamped when they occur.

The test environment that we used for the experiments reported in this paper is a simple enclosed environment consisting of several building and tower like structures along with sets of trees and rocks. These provide some visual stimulation for the users and also helps obscure their view in certain areas, forcing them to move about. The experiments are performed in the style of a First-Person Shooter (FPS) game, whereby users interact with the environment as though they were looking through the eyes of their player (see Figure 1). FPS games are one of the most popular genres of games currently in the market, and their networked multiplayer capabilities make them on obvious choice for research into distributed interactive systems in general.


**Figure 1: In-game screenshot of the test environment.**

### Test Scenario 1

This was a simple test scenario set up with the intention of analyzing the behaviour of a user towards a pseudo-dynamic goal. At the beginning of the game, the user is spawned randomly inside one of the buildings. In addition, a computer controlled opponent (typically referred to as a 'bot') is spawned randomly at one of the predefined pathnodes that were placed beforehand throughout the environment. These pathnodes join together to form a path network, and this path network is traversed by the bot pathnode-by-pathnode (a representation of this path network can be seen below in Figure 2). The bot is set on a looping run so that all it does is constantly follow the path over and over again – no artificial 'thinking' or reacting occurs for the bot. The goal for the user in this case is to disable the bot a specified number of times using their weapon before the specified time-limit runs out. The user is encouraged to score as high as possible over a number of runs such that more typical focused behaviour is exhibited rather than carefree wandering through the environment. This is so as to replicate the typical conditions under which FPS games operate. Typical behavourial patterns we would expect to see should be target seeking, pursuit and firing action. Such classes of behaviour are useful labels (albeit not always clearly distinguishable) and have been applied with some success in our previous work (McCoy 2004).


**Figure 2: Path network used for navigation by the bot.**

### Test Scenario 2

This test scenario was set up with the intention of analyzing the user's behaviour towards a bot that had limited reactive capabilities towards their own actions, namely the ability to fire it's own weapon and damage the user. It was set up in exactly the same manner as described for test scenario 1, with the exception that the bot is given a sensor field that can be used to detect the presence of the user. If the user comes within the bot's sensor field and within direct line-of-sight (LOS) of the bot, the bot is instructed to fire it's weapon in the direction of the user. If the user goes out of LOS of the bot or out of it's sensor field radius, then the bot is instructed to stop firing it's weapon. The movement capability of the bot is fixed as in test scenario 1, meaning that it never deviates it's position off of the path network on which it navigates through the environment, regardless of whether it is currently firing it's weapon at the user or not. Again, the goal for the user in this case is to disable the bot a specified number of times before the time-limit is reached. Unlike test scenario 1 however, it was now possible for the bot to win the game by disabling the user the specified number of times. The motivation for this test scenario is to determine if any resultant defensive or evasive behaviour can be recognized in addition to those behaviours stated for the previous scenario.

## RESULTS AND DATA ANALYSIS

In this section, we shall present a specific selection of results taken from data that was collected for users of varying 'expertise' with regard to not only this particular test environment, but also with computer games in general and FPS specific games. Each user was asked to play a number of successive games in each test scenario, and they were given a specific score that they had to reach to complete the game.

### Test Scenario 1

Figures 3-5 below present state data over time for several different users of varying ability. In each figure, the first 3 subplots present positional data for the user (Player 1) and the bot (Player 2), and are broken into x, y and z coordinates and plotted separately. Overlaid on each of these are solid vertical lines representing each point in time where the user disabled the bot, and these lines allow us to segment the data into partitions. The final subplot within each figure uses solid vertical lines to represent each point in time where the user fired their weapon. In Figures 9-11, we have plotted the relative position of the user with respect to the bot (i.e. taking the bot's position as the origin for each time-step). Each of the data partitions are plotted individually, with a dark circle representing the starting position of the user, and a dark 'x' marking their end position (i.e. when they disabled the bot).

From visual inspection of Figure 3 (novice user), we can see several areas where the positional data of the user and the bot seem to correlate highly, with one essentially 'following' the other one. This corresponds to a pursuit strategy, where the user has found the bot and is now pursuing them with the intention of disabling them (from time 50 to 100 for instance). It is interesting to note that this is less pronounced in the data for the advanced and expert user. The primary reason for this is that these users tend to wait in a central area and let the bot come to them rather than engage in pursuit, due to the fact that they often quickly learn the fixed movement pattern of the bot along the path network. As a result, their movement patterns tend to display significantly less variation than that of the novice user. Also, they tend to take less time disabling the bot than less experienced users. This is evident from the shooting events seen in each figure, where we see that the novice user tends to fire their weapon in large spreads, while the more advanced users fire shorter, accurate bursts that tend to disable the bot quickly, as evidenced by the correlation between the shooting events and disabled events. Obviously if the bot had learning behaviour or was controlled by another expert user such a fixed strategy would only provide short-term results, as Player 2 would soon adapt to deal with this. Indeed it is this constant interplay between various strategies in the human-human case that makes such online games both enjoyable for the user and difficult to predict for researchers (McCoy 2004). Another point of interest is the correlation of the distance between user and bot, the firing events of the user, and the bot being disabled. In most cases, we can see the positions converging close together right before the bot is disabled, indicating the user moving closer towards the bot. This is particularly evident from inspection of Figures 9-11, where we can see the convergence of the user's trajectories relative to the bot (where each trajectory starts as a dark circle and works its way towards a dark 'x' marker). This agrees with our intuitive notion that in order to disable the bot quicker, we should get closer to them (particularly in this case where the bot does not fire back). We also notice the variance in the trajectories that appear under different circumstances. For instance, long-winding trajectories typically represent some random or wandering type behaviour for a user, whereas shorter more convergent trajectories would often represent attack-type behaviour. Of interest are the trajectories that appear to both diverge and subsequently converge rather sharply. These often represent cases where a user has engaged the bot but not disabled it, and having learned the bots predictable movement strategy, chooses not to pursue it but instead chooses to wait in a suitable area for ambush.

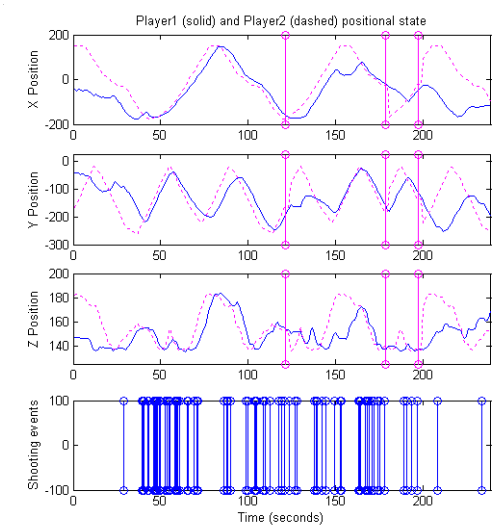


**Figure 3: State data over time for a novice user.**

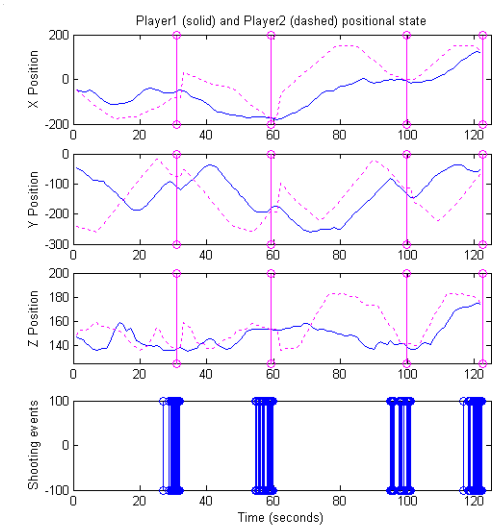


**Figure 4: State data over time for an advanced user.**

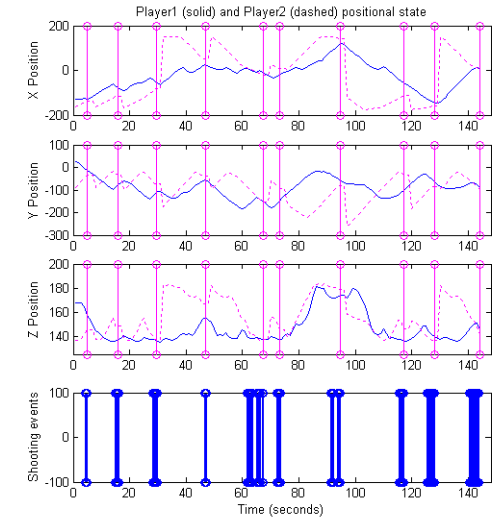


**Figure 5: State data over time for an expert user.**

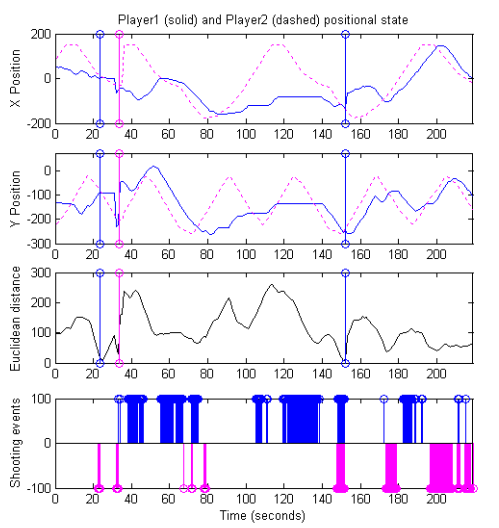


**Figure 6: State data over time for a novice user.**

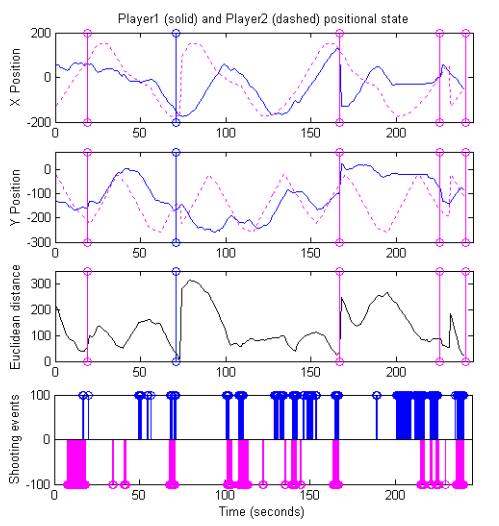


**Figure 7: State data over time for an advanced user.**

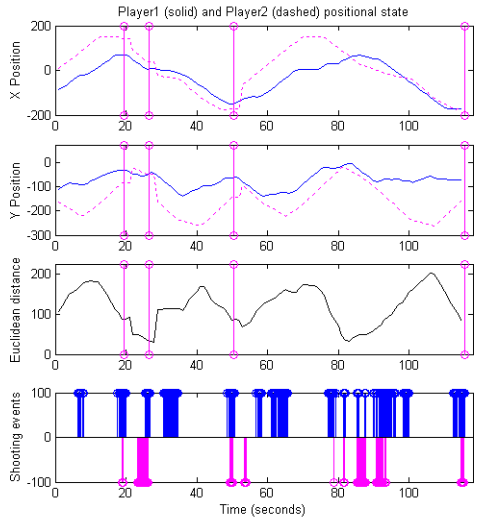


**Figure 8: State data over time for an expert user.**

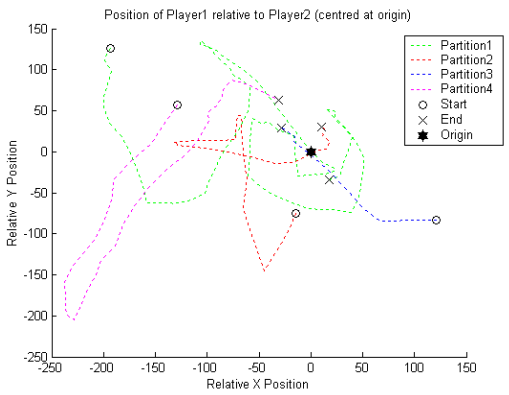


**Figure 9: Trajectory for novice user relative to bot.**

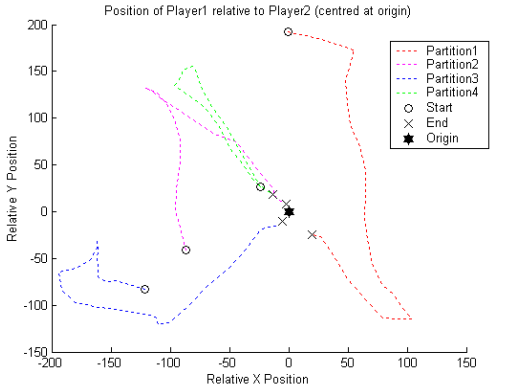


**Figure 10: Trajectory for advanced user relative to bot.**

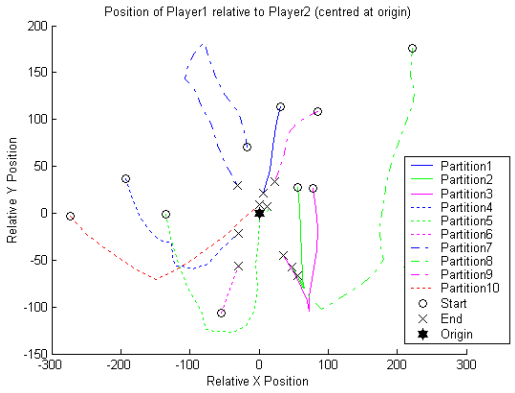


**Figure 11: Trajectory for expert user relative to bot.**

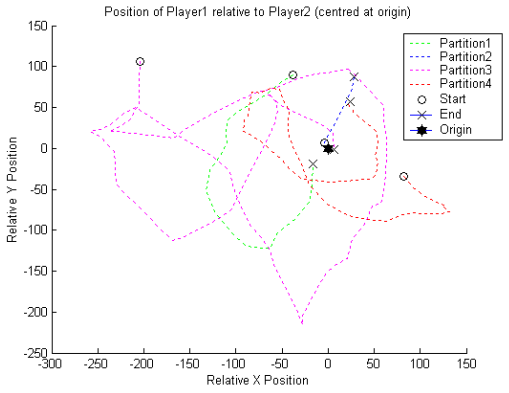


**Figure 12: Trajectory for novice user relative to bot.**

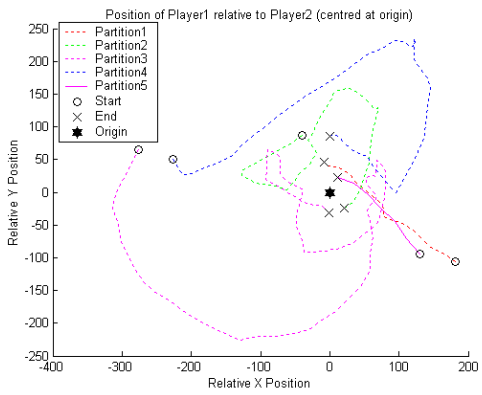


**Figure 13: Trajectory for advanced user relative to bot.**

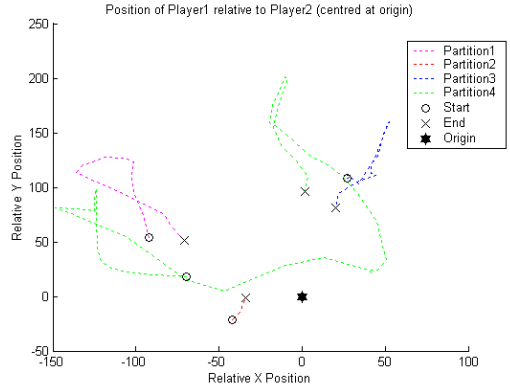


**Figure 14: Trajectory for expert user relative to bot.**

**Test Scenario 2**

Figures 6-8 below present state data over time for users of varying level of ability as in the previous section. However, here we have chosen to omit the z positional data subplot in favour of a subplot showing the 3-dimensional Euclidean distance between the user and the bot at each time step. As before, the first two subplots of each figure represent the x and y positional data for both the user and the bot, overlaid with disabled events for *both* (unlike the last section, which only showed disabled events for the bot). Finally, the last subplot shows shooting events for *both* user and bot (where the shooting events for the user are above the center line and the shooting events for the bot are below the center line). Figures 12-14 present plots of the user's position relative to the bot for each timestep, as detailed in the previous section.

From inspection of the plots, we can see that in general the distance between the user and the bot tends to converge before a disabled event occurs, implying the user moving closer to the bot in an attempt to increase their chances of disabling it. This is again evident from the plots of user trajectories relative to the bot (Figures 12-14), where we can see the convergence of the trajectories (although in the case of the expert user, it is less pronounced). These disabled events provide natural partitions of the data due to the random respawning of a player within the environment after they are disabled, leading to jumps in the positional state (random respawning is a very common system used in FPS game in general). In the case of the novice user (Figure 6),

we tend to see less of the pursuit strategy that was so prominent for test scenario 1 (Figure 3). This is quite evident from the subplot showing shooting events, where we see bursts of firing spread apart, indicating more rapid engagements between user and bot rather than prolonged pursuing. Part of the reason for this is that because the bot now has the ability to fire back, users are much more cautious about how they approach the bot, and often like to hide behind cover and attempt to 'ambush' the bot as it navigates along it's path network. It is interesting to note the variation in distance between user and bot for the case of the expert user, where we can see an extended period of time (from 50 seconds onwards) during which distance converged but subsequently diverged, coupled with a reasonably high degree of shooting events. This would indicate that the user engaged the bot and subsequently broke off their attack, but did not engage the bot in direct pursuit. Rather, having learned the bot's movement pattern, they most likely waited for the bot to return on its course and then re-engage it, at which point they finally disabled the bot. This is consistent with the results shown in Figure 14, where we can observe the divergence of the user's trajectories coupled with the subsequent convergence, indicating a section of time that most likely involved multiple periods of interaction between user and bot. Also evident is the fact that the end points of the expert user's trajectories (marked by dark 'x' points) have a higher spread as opposed to both the novice and advanced users (Figures 12 and 13 respectively) – this would indicate greater accuracy on the part of the expert user at disabling the bot with his weapon.

## CONCLUSION

From the results already shown for a number of users it is clear that certain patterns of behaviour do emerge. In both test scenarios it is that correlated dynamical behaviour can occur indicating that the respective entity states cannot be statistically independent. In intuitive terms this means for example that if the target player changes velocity dramatically it is often the case that the pursuing player will do likewise. This rather obvious observation could be exploited in entity state extrapolation through predicting such changes where the conventional paradigm of dead-reckoning would have had to transmit a packet indicating a velocity change. It may even be possible to preempt shooting events which conventional techniques have no possibility of predicting. In another step towards modeling the human-human interaction that is so important in DIM another intermediate step will be taken in which the bot can exhibit more human-like behaviour. This can be as simple as a hunting and pursuit behaviour where the bot having spotted Player 1 will attempt to close the distance before attempting to disable the opponent. Such behaviour should elicit more interesting defensive or evasive behaviour in the human user that will be important to analyze. Further analysis along these lines should yield insight, tools and results that will allow better and more comprehensive analysis of human users interacting in the same environments. Consequently it will be possible to make better guesses about what such users may do next and hence achieve quality of interaction and scalability benefits for the distributed case.


## ACKNOWLEDGEMENT

This material is based upon works supported by Enterprise Ireland under grant no. SC/2002/129/.


## REFERENCES


Delaney, D., Ward, T., and S. McLoone. 2003. "Reducing Update Packets in Distributed Interactive Applications using a Hybrid Approach". 16th International Conference on *Parallel and Distributed Computing Systems (PDCS 2003)*. August 13-15, Reno, USA, pp.417-422.

IEEE. 1993. IEEE Standard for information technology – protocols for distributed simulation applications: Entity information and interaction. IEEE Standard 1278-1993. New York: *IEEE Computer Society*, 1993.

Internet2. 2004. WebSite: *http://www.internet2.edu*

Marshall, D., McCoy, A., Delaney, D., Ward, T., and S. McLoone. 2004. "A Realistic Distributed Interactive Testbed for Static and Dynamic Entity State Data Acquisition". In Proceedings of *The Irish Signals and Systems Conference 2004*. June 30 - July 2, Belfast, Northern Ireland, pp. 83 - 88.

McCoy, A., Delaney, D., McLoone, S., and T. Ward. 2004. "Investigating Behavioural State Data-Partitioning for User-Modelling in Distributed Interactive Applications". To appear in proceedings of *The 8-th IEEE International Symposium on Distributed Simulation and Real Time Applications.* October 21-23, Budapest, Hungary.

Singhal, S. and M. Zyda. 1999. *Networked Virtual Environments: Design and Implementation.* First Ed, Addison-Wesley, pp. 195 - 213.

Van Hook, D.J., J.O. Calvin, and D.C. Miller. 1994. "A protocol independent compression algorithm (PICA)". *Advanced Distributed Simulation Memorandum* 20PM-ADS-005, MIT Lincoln Laboratories, Lexington, MA.


## BIOGRAPHY

Aaron McCoy received his B.Sc. degree from the National University of Ireland, Maynooth in 2002, specializing in computer science and theoretical physics. His main areas of interest are distributed systems, networked virtual environments and the use of artificial intelligence in interactive games. He is currently studying for his PhD at NUI Maynooth in the area of user modeling in Distributed Interactive Applications.