Exploring the Spatial Density of Strategy Models in a Realistic Distributed Interactive Application

Damien Marshall, Declan Delaney
Department of Computer Science,
National University of Ireland, Maynooth,
Co. Kildare, Republic of Ireland.
E-mail:{damienm, decland}@cs.may.ie

Seamus McLoone, Tomas Ward
Department of Electronic Engineering,
National University of Ireland, Maynooth,
Co. Kildare, Republic of Ireland.
E-mail: {seamus.mcloone, tomas.ward}@eeng.may.ie

Abstract

As Distributed Interactive Applications (DIAs) become increasingly more prominent in the video game industry they must scale to accommodate progressively more users and maintain a globally consistent worldview. However, network constraints, such as bandwidth, limit the amount of communication allowed between users. Several methods of reducing network communication packets, while maintaining consistency, exist. These include dead reckoning and the hybrid strategy-based modelling approach. This latter method combines a short-term model such as dead reckoning with a long-term strategy model of user behaviour. By employing the strategy that most closely represents user behaviour, a reduction in the number of network packets that must be transmitted to maintain consistency has been shown. In this paper a novel method for constructing multiple long-term strategies using dead reckoning and polygons is described. Furthermore the algorithms are implemented in an industry-proven game engine known as Torque. A series of experiments are executed to investigate the effects of varying the spatial density of strategy models on the number of packets that need to be transmitted to maintain the global consistency of the DIA. The results show that increasing the spatial density of strategy models allows a higher consistency to be achieved with fewer packets using the hybrid strategy-based model than with pure dead reckoning. In some cases, the hybrid strategy-based model completely replaces dead reckoning as a means of communicating updates.

1. Introduction

Distributed computer games are becoming one of the more important facets of an already burgeoning games industry. Such games pertain to a class of computer application known as Distributed Interactive Applications (DIAs), and can involve tens of thousands of simultaneous players interacting in a simulated environment. Given the scale of DIAs, the volume of data transmitted can be considerable, for example, Everquest [1]. As the complexity of the DIAs increases and the number of concurrent users multiplies, the application and connecting network must be able to scale without losing the real-time experience of participating in a shared environment [2]. This ability to scale is inherently linked to the number of network packets that need to be transmitted between participants to maintain a consistent global worldview of the DIA. Several techniques exist to reduce the quantity of network traffic, including relevance filtering, compression and dead reckoning [3, 4].

Dead reckoning uses a short-term model to predict entity dynamics. However, it does not take a priori information regarding entity behaviour in relation to an objective or goal into account. In contrast the recently proposed hybrid strategy-based modelling approach combines a short-term dead reckoning model with a long-term strategy model [5]. Using this approach a reduction in the number of packets that are required to maintain global consistency has been demonstrated compared to using dead reckoning alone. These results were based on a restricted two-dimensional test environment that served to prove the concept but did not allow the experiments to scale to large numbers of strategy models. The long-term strategy models themselves were constructed using visual heuristics. Strategy models can be of two types: an exploratory or *transient* strategy that users employ when they are unfamiliar with the environment and a *steady-state* strategy, which corresponds to a preferred way to satisfy the user goal in the environment.

Here, a novel implementation of the strategy model technique in an industry-proven game engine called Torque is presented [6]. This method builds on previous work by facilitating the construction of an arbitrary number of strategy models. Furthermore, simulation results are presented for experiments featuring a varying spatial density of strategy models using this new test environment. The results show that as the number of models increases, there is a decrease in the number of packets required in order to maintain consistency in the DIA. An optimum number of these models exist for steady-state trajectories, above which no further packet

reduction is obtained. The actual value itself will vary from application to application.

2. Implementing the Hybrid Strategy-Based Model

One of the key difficulties with the hybrid strategy-based model approach is the determination of the strategy models. To date, actual user data has been recorded in a limited game-like environment and a representative steady state strategy was chosen based on a visual analysis of actual user steady-state trajectories[5].

Here, a novel approach based on dead reckoning is proposed. When dead reckoning is employed, the error threshold is always exceeded at certain key areas. By recording the points where the dead reckoning threshold is exceeded by this user, a set of points is obtained that provides an average piecewise linear model of a user's trajectory – see Figure 1.

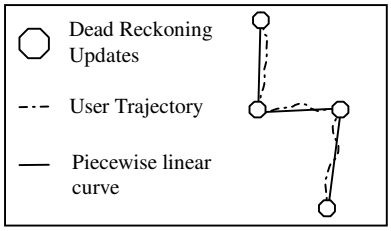

Figure 1 A sample piecewise linear model.

The piecewise linear model is then combined with the error threshold values which defines bands on both sides of the model to describe a polygon shape known as a "strategy polygon". An entity located within the strategy polygon is deemed to be following that particular strategy.

The first step in the creation of the strategy polygon is the definition of the polygon edges. Each edge consists of a line parallel to a segment of the piecewise linear curve. The distance between an edge and the appropriate segment of the curve is the required error threshold – see Figure 2a. Next, the end points of the segments need to be joined so as to provide a closed polygon. If this is implemented incorrectly, the resulting polygon could be distorted, leading to erroneous assumptions about the entity's behaviour. Again, see Figure 2a.

The problem is avoided by considering two contiguous segments of the piecewise linear curve in order to define the strategy polygon vertices. The parallel segments are created as detailed above. However, the vertices of the strategy polygon are given by the locations where the newly generated parallel segments intersect. Examples of such intersections are detailed in Figure 2b. Repeated application of this technique to all the segments of the piecewise linear curve results in a group of vertices that describes the strategy polygon.

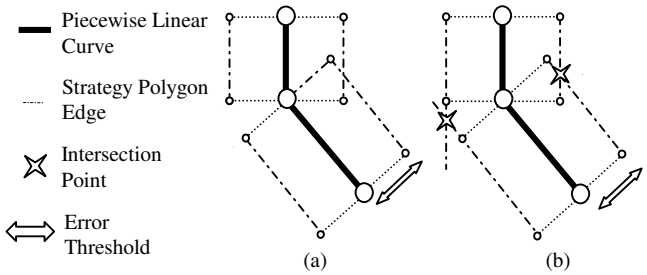

Figure 2(a) Distortion can occur when creating the strategy polygon (b) The polygon distortion is resolved using the intersection points of parallel segments

This process can be used in order to define multiple strategy models. Initially, the required number of strategies is created. For the purposes of this paper, the newly created strategies are always parallel to that of the original piecewise linear curve. Therefore, the process by which the polygon edges are created can be used to define each individual strategy. The parallel distance between each curve is the required distance between strategies as detailed in Figure 3.

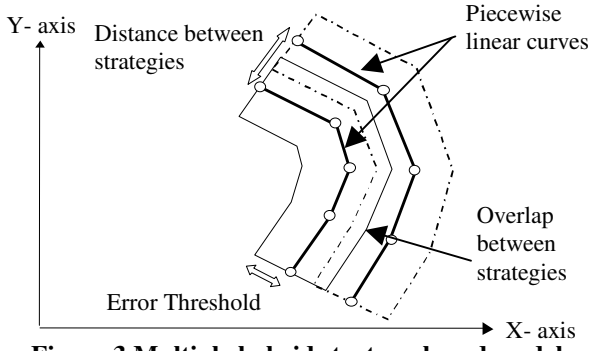

Figure 3 Multiple hybrid strategy-based models

Next, each curve is converted into a full strategy polygon, using the required error threshold. This threshold is always set to a value greater than half the distance between strategies in order to provide an overlap between models, and therefore avoid unnecessary updates due to excessive switching between models.

With multiple models, it becomes important to determine which model, if any, the entity's current position resides in. The 'point containment test' is used to do this [7]. This test is used extensively in computer games in a process known as polygon filling [8]. It works by extending a horizontal line from the current entity position and counting the number of intersections between this line and the edges of the polygon. An odd number of intersections indicates that the position is inside the polygon. Otherwise the position is outside it. The

decision to make the switch between models is taken when an update from the currently employed model is required.

The strategy model approach was implemented in Torque using the algorithms described in this section, along with a first order dead-reckoning algorithm. The engine was also modified so that user trajectories could be recorded and steady state strategies identified. The following section details the experiments performed using these models.

3. Experimental Simulations

The experimental test platform was implemented using the Torque game engine and a full client-server architecture was employed. Each simulation was carried out in a game environment that consisted of a unique route from a start to a target position. These positions remained invariant throughout. The user's goal was to navigate from the start to the target position in the shortest time possible [6]. The application simulated the number of packets transmitted in order to maintain global consistency within a certain threshold error tolerance. The error thresholds and separation distance between the strategies is defined in Torque game units. Each game unit corresponds to a meter in distance within the virtual environment.

The first set of experiments measured the number of packets transmitted as the number of strategies was increased and a constant error threshold was maintained. The second group of experiments focused on increasing the density of strategies within the same area of the environment. This was achieved by beginning with a single strategy with a large error threshold. This defined the area, and encompassed all recorded user trajectories. The error threshold was then decreased and the number of strategies increased so that the strategy models always covered the same area.

A total of six subjects participated in the experiments. All user trajectories were recorded. The application then employed this data to simulate the generation of update packets under various experimental conditions. The simulated entity navigated the test level using the positional data recorded from each participant. When the target position was reached, the following data was recorded to disk: type of update, the position where the update occurred, the total number of updates transmitted for that attempt, the actual entity position, the modelled position and the strategy polygon vertices.

4. Results

Here, results are presented for the two sets of experiments outlined in the previous section. In all experiments, both the Hybrid Strategy-Based Model (H), and Dead Reckoning (D), are considered. In every case, U_n denotes a test user.

4.1 Constant Error Threshold

As discussed in section 3, the first set of experiments concentrated on the comparison of pure dead reckoning with the hybrid strategy-based model approach as the number of strategies increased and the error threshold was maintained constant. Table 1 details results from two test users who are representative of all users. In this case an error threshold of 6 game units for both dead reckoning and the strategy model, and a distance of 9 game units between strategies, is employed.

Table 1 Error Threshold 6 Strategy Width 9

Trial	DR		1 Model		3 Models		5 Models	
	U1	U2	U1	U2	U1	U2	U1	U2
	D	D	H	H	H	H	H	H
1	20	30	16	28	16	29	16	31
2	17	30	11	28	12	29	12	31
3	12	19	9	17	5	16	5	15
4	12	11	6	8	5	9	5	10
5	12	12	5	1	5	1	5	1

From this table we can see that as the user converges to a steady state trajectory, there is a significant reduction in packets when compared to pure dead reckoning. We can also note a reduction in packets sent as the number of available models increases. When a user first enters the environment, as exemplified in trial attempts 1 and 2 of both users, they are following an exploratory or transient trajectory. In these cases, the user is exploring the environment, and dead reckoning is used extensively. With the transient trajectories, as the number of models increases, the number of strategy packets also increases. This occurs because there are more models available and there is increased likelihood that the entity will cross back and forth across them in getting to know the environment and accomplish the goal.

Once the user has become more familiar with the test level, as seen in trial 3, trajectories close to that of the steady state strategy are adopted. This leads to a notable reduction in updates, and overall better performance from the hybrid model in comparison to dead reckoning. When a steady state trajectory is adopted, the addition of models above a certain quantity has little or no effect on packet update rates. In these cases, the optimum number of models for that trajectory has been found. This can be seen in trials 4 and 5 in both tables, where minimal packet reduction is noted after the use of 3 strategy models. In the final attempts of both users dead reckoning is made completely redundant.

A series of experiments were also conducted featuring an error threshold of 9 game units, and a distance of 12

game units between strategies. Results similar to that of table 1 were noted. Due to space restrictions, however, they are not presented here.

4.2 Variable Error Threshold

The next set of results focuses on an increasing spatial density of strategy models – see Table 2. In this case, we initially start with a single large model with an error threshold of 36, and progressively increase the number of models found within the area occupied by that model.

Table 2 reinforces the fact that the hybrid strategy-based model performs better than pure dead reckoning. As would be expected, the number of pure dead reckoning packets increases as the error threshold decreases. At least one packet will be required regardless of the model used; trials 4 and 5 indicate this for the hybrid model. The packet transmitted in these trials refers to the long-term strategy model.

Table 2 Variable Error Threshold Results for 1 User

Trial	1 Model		3 Models		5 Models	
	D	**H**	**D**	**H**	**D**	**H**
1	10	6	20	10	24	16
2	8	8	16	14	18	18
3	5	1	11	5	15	7
4	5	1	8	3	11	5
5	4	1	6	1	9	1

In trial 2 of Table 2, we can see that, as the threshold decreases, an increase in the spatial density of models occurs, and switching between strategies becomes more likely. This accounts for the increasing number of packets as the density of models increases.

The results indicate that when a single model is used it covers all steady-state trajectories as shown by the single packet transmitted in trials 3 to 5 in Table 2. However, the threshold is very high and consequently consistency is low. By increasing the number of models within the same space, the value of the threshold decreases without incurring a significant increase in packets sent. Trials 4 and 5 clearly illustrate this. A tighter threshold provides greater consistency. This is a clear illustration of the consistency-throughput tradeoff [4].

6. Conclusions and Future Work

In this paper we have described a novel method for constructing multiple user strategies using dead reckoning and polygons in a realistic DIA known as Torque. This was carried out in order to provide a real world implementation of the hybrid strategy-based model, and to test the effects of varying spatial density on the amount of packets needed to maintain consistency of a DIA.

Using this approach, we have shown that the hybrid strategy-based model performs better than pure dead reckoning. As the number of models increases, there is a reduction in the number of packets needed for steady state trajectories. In these cases, an increase in the number of models above a certain quantity has a negligible effect on packet reduction. The point at which this occurs is the optimum number of models. The actual optimum value will vary with the application and the route represented by the steady state strategy.

An increase in the density of strategies in one area results in the minimisation of dead reckoning packets. When the trajectories have converged to a steady state, the hybrid strategy-based model can provide increased consistency using fewer packets than dead reckoning alone.

Future work will involve the examination of the psycho-perceptual effects of latency on users of Distributed Interactive Applications to establish criteria for choosing error threshold values.

REFERENCES

[1] Humble, R., *Inside EVERQUEST*, in *Game Developer*. May 2004. p. 18-23.

[2] Bondi, A.B. (2000). "Characteristics of Scalability and Their Impact on Performance".In *Second international workshop on Software and Performance*, (Ottawa, Ontario, Canada, 2000), ACM Press, New York, NY, USA, 195 - 203.

[3] Bassiouni, M.A., M.H. Chiu, M. Loper, M. Garnsey, and J. Williams, *Performance and reliability analysis of relevance filtering for scalable distributed interactive simulation.* ACM Transactions on modeling and computer simulation, 1997. **7**(3): p. 293-331.

[4] Singhal, S.K. and M. Zyda, *Networked Virtual Environments*, ed. S. Spencer. 1999, New York: ACM Press, New York, New York.

[5] Delaney, D., T. Ward, and S. Mc Loone (2003). "Reducing Update Packets in Distributed Interactive Applications using a Hybrid Model".In *16th International Conference on Parallel and Distributed Computing Systems, August 13-15*, (Reno, USA, August 13-15), 417-422.

[6] Marshall, D., A. McCoy, D. Delaney, S. McLoone, and T. Ward (2004). "A Realistic Distributed Interactive Application Testbed for Static and Dynamic Entity State Data Acquisition".In *Irish Systems and Signals Conference*, (Belfast, Ireland, 30 June - 2 July), IEE, 83-88.

[7] Mortenson, M.E., *An Introduction to the Mathematics and Geometry of Computer Graphics*. 1989: Heinemann Newnes, Jordan Hill, Oxford.

[8] Jenq, J.J. (1999). "Parallel Polygon Scan Conversion on Hypercube Multiprocessors".In *ACM symposium on Applied computing*, (San Antonio, Texas, United States, February 28 - March 02), 110 - 114.