A Physics-Aware Dead Reckoning Technique for Entity State Updates in Distributed Interactive Applications

Patrick J. Walsh, Tomás E. Ward and Séamus C. McLoone

Department of Electronic Engineering,

National University of Ireland Maynooth,

Maynooth, Co. Kildare, Rep. of Ireland.

email : pwalsh@eeng.nuim.ie; tomas.ward@eeng.nuim.ie; seamus.mcloone@eeng.nuim.ie

***Abstract*— This paper proposes a novel entity state update technique for physics-rich environments in peer-to-peer Distributed Interactive Applications. The proposed technique consists of a dynamic authority scheme for shared objects and a physics-aware dead reckoning model with an adaptive error threshold. The former is employed to place a bound on the overall inconsistency present in shared objects, while the latter is implemented to minimise the instantaneous inconsistency during users' interactions with shared objects. The performance of the proposed entity state update mechanism is validated using a simulated application.**

***Keywords* – Distributed Interactive Applications, Physics-aware Networked Games, Consistency**

I INTRODUCTION

Distributed Interactive Applications (DIAs) allow geographically remote users to interact within a shared virtual environment [1, 2]. Many DIAs, such as networked computer games, strive to present a rich and realistic virtual world to users, both on a visual and a behavioural level. They aim to present to users an environment whose behaviour is consistent with the users' experience and perception of the real world [3]. A relatively recent addition to DIAs to achieve this aim has been the simulation of realistic physical phenomena between objects in the environment.

The application of physics simulation to virtual environments in DIAs, however, presents new challenges in the context of maintaining consistency among multiple users. This is particularly evident in applications built on a peer-to-peer architecture, where a lack of a single authority presents additional challenges in synchronising the state of shared objects while presenting a responsive simulation. There is currently a dearth of suitable entity state update mechanisms which can maintain consistency in such physics-rich environments.

This paper proposes a novel physics-aware technique for the synchronisation of entity states within a peer-to-peer, physics-aware DIA that employs dead reckoning algorithms [4] for the purposes of traffic reduction. In doing so, the *physics-consistency-cost* is introduced. It is this cost that is subsequently used in the generation of entity state updates. This new technique is implemented for a physics-aware simulated application and demonstrates a marked improvement in consistency for this application.

The rest of this paper is structured as follows. The next section describes the concept of consistency in DIAs and introduces the *physics-consistency-cost*. The standard dead reckoning model, used for traffic reduction, is also presented. Section III outlines the proposed physics-aware dead reckoning technique for updating entity states in DIAs. Section IV presents the simulation used to evaluate the proposed technique, with the results given in section V. The paper ends with some conclusions in section VI.

II PHYSICS-CONSISTENCY-COST

a) Consistency

Consistency in a DIA refers to the ability of the DIA to ensure that each user's view of the world is identical, or as close to identical as can be achieved for given conditions. Traditional consistency metrics examine entities on an individual basis, often giving a measure of the application's ability to represent a host's controlled entity's states at remote hosts. A basic aspiration for a distributed simulation is to present entities as being *in the right place at the right time* and a simple metric to capture this notion is a spatially-derived measure of consistency that can be calculated using distance measures between the entity positional state at its local peer and its representations at remote peers. In real-time applications, the consistency-throughput trade-off notes that true consistency is unachievable [3]. Hence, most real-time DIAs accept a controlled level of inconsistency. They employ approximated models of controlled entity motion to reduce the traffic on the network. The recognised standard model used within the Distributed Interactive Simulation (DIS) is Dead Reckoning [4].

b) Dead Reckoning

Dead reckoning (DR) is a short-term linear extrapolation algorithm which utilises information relating to the dynamics of an entity's state and motion, such as position and velocity, to model and predict future behaviour. All users model all entities, including those local to the user. Each user knows both the local and modelled behaviour of local entities, and can compare them at all times. Local users send updates to remote users when they determine that the error between modelled and actual behaviour has exceeded a predefined error threshold. A first order example of DR is shown in Figure 1.

Depending on the error threshold employed, it is possible, and indeed likely, that the actual and modelled entity positions will differ somewhat. As a result, when updates are received, modelled remote entities may appear to jump, or 'snap' to the updated position. Convergence mechanisms are usually employed to smooth this jump [3], with a good convergence algorithm being capable of correcting the modelled behaviour quickly, without presenting too distorted a world view to the user.

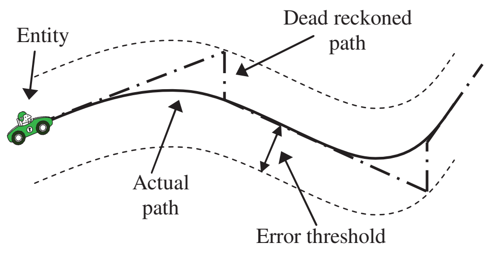

Figure 1: Dead reckoning (DR).

c) Physics-consistency-cost

The existing inconsistency metrics are less than ideal for use in physics-aware DIAs, as they do not present a clear picture of all the inconsistency in the environment. These metrics only capture the discrepancies between the local controlled entities (the users' avatars) and their remote representations. They do not allow for discrepancies that may occur in surrounding environmental entities. A better inconsistency metric would ideally capture all spatial inconsistencies present in the state of both the controlled avatar and the physics-aware entities with which the avatar has collided. The latter is what we refer to as the *physics-consistency-cost*.

Consider the *ball example* given in Figure 2 where the entity represented by ball A (at its local peer) approaches ball B but stops just short of it, as shown. If the distance between A and B is less than or equal to the error threshold of the dead reckoning model then a remote peer could observe ball A disturbing ball B, and ball B subsequently rolling away down the hill, resulting in a spatial inconsistency of δ_1 in the state of ball B. It is this state divergence, and the resulting visual disturbance to the application required to correct it, that is currently missing from existing consistency maintenance techniques. In this example, δ_1 is referred to as the *physics-consistency-cost* associated with entity A's path.

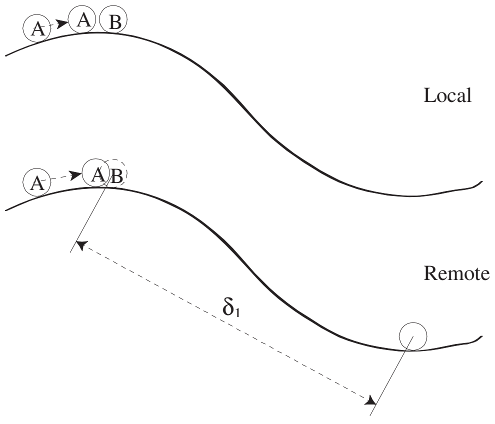

Figure 2: The ball example.

Now consider the *domino example* presented in Figure 3. At the local host the ball approaches a row of dominos and stops close to them. However the dead reckoning model is observed to collide with the first domino, which in turn topples the entire row. In this instance, the magnitude of the physics-consistency-cost of the first domino, δ_2, is relatively small but due to the nature of dominos, a single incorrectly disturbed domino can in turn disturb its neighbours, resulting in a physics-consistency-cost of approximately N x δ_2, for N dominos. Arguably, the magnitude of the inconsistency imparted to all the dominos could still be significantly smaller than that of the ball in the previous example. However, the number of entities involved is significantly more and, therefore, returning all the dominos to their original state has a greater visual impact. Hence, even small spatial inconsistency in physics-aware

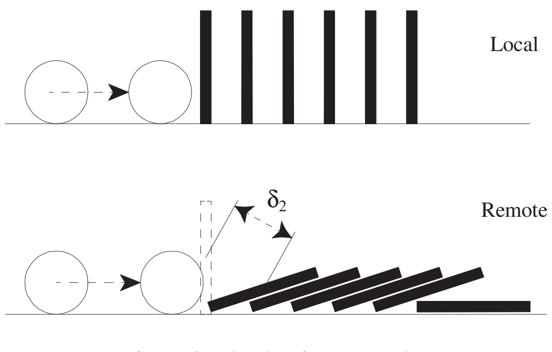

Figure 3: The domino example.

entities can result in a considerable physics-consistency-cost, if complexity (in this case, the number of entities affected) is considered.

It is worth noting that not all physics-enabled interactions lead to such cascades in inconsistency. As a counter example, consider a scenario where the dead reckoning model of a remote entity is incorrectly observed to collide with a much larger physics-aware entity, with large inertia, such as a ball avatar colliding with a van. Under realistic physics simulation the modelled avatar may impart a spatial disturbance to the van. However, this would be significantly smaller than the disturbance experienced by the ball in Figure 2 (especially in terms of relative size) – it may even be imperceptible to the user. In this case, the error in state is easier to correct than the error in the domino example. The incorrect disturbance is still a physics-consistency-cost, but is significantly smaller than the previous two examples.

Currently entity state update protocols for peer-to-peer distributed simulations, such as dead reckoning, do not explicitly take into account the inconsistency costs associated with physics-enabled environments. Thus, the ability of these techniques to regulate consistency in such situations is not optimal and improvements may be possible by incorporating information about the local physics environment. The next section proposes a consistency maintenance technique that exploits such information.

III PHYSICS-AWARE DR TECHNIQUE

a) Dynamic Authority in Physics-aware DIAs

In traditional peer-to-peer DIAs, each peer is authoritative on the state of its local entities. This authority is static in nature as it is assigned when the peer joins the DIA and is not changed or reassigned thereafter. Theoretically, authority over physics-aware entities could be granted to a single peer, with all other peers having to validate or seek approval of interactions between their local entities and the physics-aware entities. However, this would place a significant burden on that one peer, effectively turning it into a server for physics simulation. In addition, the application becomes less fault-tolerant as the entire physics simulation is dependent on that peer.

Dynamic authority refers to being able to change which peer provides authoritative state for an entity while the application is running. Our proposed algorithm makes use of a dynamic authority scheme, whereby each peer assumes authority over entities with which it interacts. This removes the need for a peer to remotely validate interactions of its entities with physics-aware entities who might otherwise have had remote authoritative peers. Under such an authority scheme, each peer would be responsible for notifying other remote peers of the resulting state from interactions between their local entities, and physics-aware entities. In addition, controlled entities are simulated using dead reckoning models as outlined in section II.

In the event of a controlled entity interacting or colliding with a physics-aware entity, the resulting state of both the controlled entity and the physics-aware entity is supplied by the local peer to all remote peers. Deterministic physics simulation at each peer means that once supplied with the state immediately after the collision, the simulation of the play out at each peer will result in the same final state. Finally, collisions between modelled remote entities and physics-aware entities are ignored unless a collision notification, or state update, for the physics-aware entity is received.

b) Updating state of physics-aware entities

Initial tests showed that providing a single update at the instant of contact between an avatar and the colliding body proved to be unreliable and insufficient, with inconsistency manifesting after collisions. Many collisions persisted beyond a single simulation frame. Consequently, the potential for user input over the course of the collision meant that the play-out was not guaranteed to be deterministic after the initial contact. If a user were to change their speed or direction slightly during the collision, they could potentially remain within the threshold of the dead-reckoned model, and thus not generate an update. Traditionally this would not be an issue in peer-to-peer simulation, but with the sensitivity of physics engines to variation in initial conditions such slight changes were still a source of significant inconsistency.

In order to resolve this issue, we require that all peers also update the state of their avatar and the colliding body once the collision had ceased, i.e. the bodies are no longer in contact.

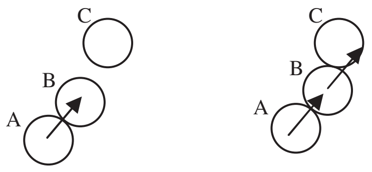

Figure 4: In a collision by proxy, user-controlled entity A collides with entity B, and during this collision, entity B contacts entity C.

c) Collisions by proxy

A shortcoming of the proposed approach for updating entities is illustrated in Figure 4. A collision by proxy is where in the course of the user's (circle A) collision with an environmental body (circle B), the body in turn touches a second environmental body (circle C). In this instance, the ultimate path of body C will be inaccurately modelled, as during the collision, bodies A and B's paths are approximated. The motion of the latter two bodies will be corrected once they separate, but at this point the remote state of body C will be inaccurate, and will remain inconsistent. This problem can easily be solved by carrying out a recursive check, for collisions, of all bodies that the user's entity is in contact with. In other words, if the controlled entity is in contact with another body, that body is then checked for contacts, and if any are found, those other bodies in the collision are subsequently checked, and so on. A record is kept of these contacts, and if a contact ends, then any bodies no longer in contact are updated. If the controlled entity loses its contact with the first object, then all objects can be updated, as the play out will be deterministic with the possibility of user input having been removed from the system.

d) Minimising the physics-consistency-cost

As already outlined, the physics-consistency-cost occurs as a result of the inconsistency in controlled entities being transferred to physics-aware entities in their surrounding environment. The aforementioned authority scheme now ensures that the inconsistency in the physics-aware entities is controlled in the same manner that dead reckoning permits a controlled level of inconsistency in the controlled entities.

A further enhancement can be achieved by actually minimising the physics-consistency cost in the first instance. Since such costs arise as a result of the error in the dead reckoning model of entities, it is proposed that peers should attempt to minimise the error present in these models at the time of collisions. This can be achieved by using a DR model with a small error threshold. However, this has the negative impact of unnecessarily utilising bandwidth when collisions are not occurring. Thus, we propose using a variable, or adaptive, threshold.

Although adaptive threshold dead reckoning models already exist [5, 6], these only serve to control the level of inconsistency between an avatar's entity and its remote representations, while reducing the number of updates generated by an application. In this paper, the DR threshold can also be adapted in response to a predicted measure of the physic-consistency-cost, i.e. a physics-aware adaptive threshold DR algorithm.

One means of predicting collisions between a locally-controlled entity and a physics-aware entity is to predict the future states of the environment based on current and/or past states, and then checking for collisions between the locally controlled entity and any physics-aware entities in each future state. This could be accomplished by copying the environment within a certain radius of the controlled entity, and advancing the copied simulation a set number of steps, checking for collisions between steps. If a collision is imminent, the physics-consistency cost increases and the DR error threshold can be reduced to ensure that more updates are communicated between peers in order to minimise the overall inconsistency. Similarly, if no collisions are imminent then the DR error threshold can remain relatively large so that less updates are required and the bandwidth between peers can be better utilised.

An alternative and heuristic means of predicting an increase in the physics-consistency-cost is to relate this cost to the density of entities in the immediate vicinity of a controlled entity, or along its projected path. Clearly, the likelihood of physics-consistency-costs arising (i.e. collisions occurring) is dependent on the number of physics-aware entities in the surrounding environment. It should be intuitive that for two environments, identical in every way except for the number of entities present, there is a greater likelihood of colliding with a physics-aware entity in the environment with more entities.

e) The proposed technique

Figure 5 shows the entity state management process followed by each peer using the proposed state management technique. Taking each loop to start at the *Sample User Input* block, the proposed physics-aware dead reckoning entity state update technique operates as follows.

The user's input is sampled from their input device(s) (e.g. keyboard/mouse), and these inputs are applied to the user's controlled entity. The network connection is checked for state updates from other peers that have yet to be applied to the local world database. The local world simulation is advanced by one frame based on existing state information and the newly applied information (inputs and updates).

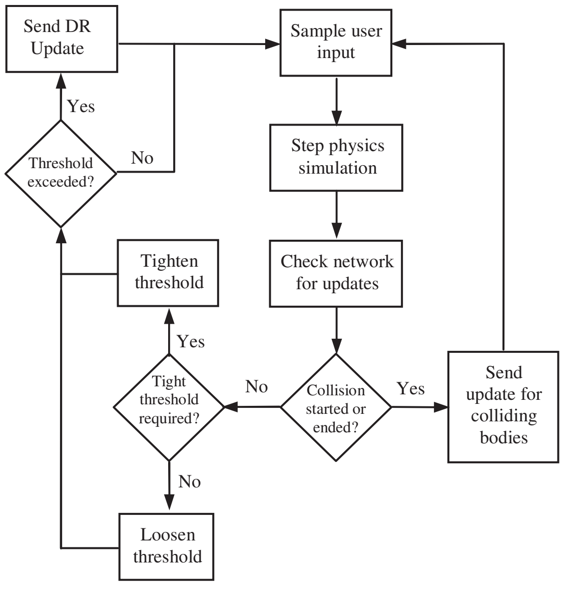

Figure 5: Flowchart of proposed authority scheme with adaptive threshold dead reckoning model.

The controlled entity's list of contacts is then checked to see if it has begun a new collision or ended an existing collision. If it has, a state update is sent for both the controlled entity and the entity that it collided with and the loop repeats once more. If no collision has occurred, then the technique checks for possible future increases in the physic-consistency-cost. The DR error threshold is adapted accordingly, i.e. either increased or decreased (one of which will be the current position). If the error threshold in use for dead reckoning in this simulation tick has been exceeded, a state update for the controlled entity is sent to remote peers. Either way, the process returns to the start of the loop and the process repeats once more.

IV SIMULATION

The previous section outlined a state management technique for physics-aware peer-to-peer DIAs. Here, we briefly outline the development of a simple simulated application, used to illustrate the performance of this technique.

The application contains two separate copies or instances of a single world, with entity state being shared from one (local environment) to the other (remote environment). A facility to simulate latency and jitter in the connection between these two environments is required, as these are real world scenarios and problems affecting DIAs. With this in mind, a *packetPipe* class was implemented, which could be extended and modified, depending on the nature of the updates being transmitted. This class would serve as a link between the two simulation instances, and be capable of simulating typical network conditions such as latency and jitter.

The test environment consists of an arrangement of physics-aware entities and a single controlled entity, which are randomly positioned within an enclosed space. The controlled entity moves along a pre-defined path. The physics state resulting from the controlled entity's motion is then simulated by the Box2D engine, an open-source, two-dimensional physics simulation engine that can be integrated into applications as middleware [7]. For simulation of the controlled entity's motion, a path, collected from real user motion, was read from comma separated values (CSV) files, with a force being applied to the controlled entity in order to move it to the next point along its path. A sample path is shown in Figure 6 below.

In the following section, several simulations were carried out for several different paths and environments. However, due to lack of space, only the results for the sample path in Figure 6 are presented. These results are a representative sample of the results obtained.

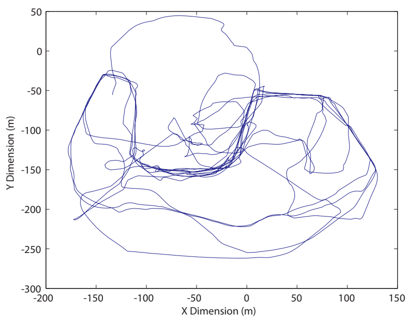

Figure 6: A sample user path.

V RESULTS AND DISCUSSION

a) The need for an authority scheme

The first set of simulations was run for a range of fixed DR error thresholds. However, only the DR models of the controlled entity were supplied to the simulated remote peer. Latency is set to zero for the ease of illustration, as it does not impact on the key observations and conclusions.

Figure 7 shows the combined inconsistency of all physics-aware entities in the test environment over a period of time. It can be easily seen that uncontrollable inconsistency occurs in the absence of an authority scheme. Note that, here, inconsistency refers to both the spatial error in the controlled entity and the spatial error in all the physics-aware environment entities. This inconsistency clearly increases over time, reflecting the fact that physics-aware entities have been disturbed and are not corrected. Hence, it is obvious that there is a need for an authority scheme to update these entities and to ensure a bound on inconsistency. The dynamic authority scheme, as outlined in section III (a), is employed in all test simulations for the remainder of this paper.

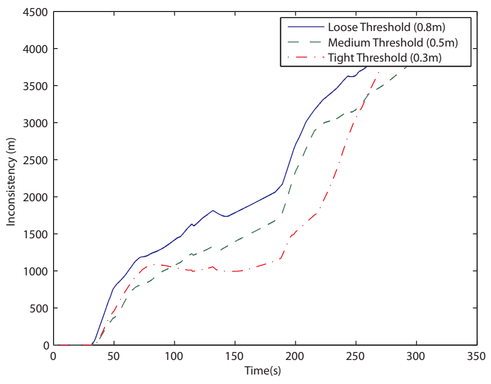

Figure 7: Combined inconsistency of all physics-aware entities

b) Using DR with fixed thresholds

Figure 8 shows the variation of inconsistency for all entities, as a result of the controlled entity following the sample path, for a fixed threshold dead reckoning model and the dynamic authority scheme. Results are presented for three different thresholds, namely 0.8m, 0.5m and 0.3m respectively.

Considering the graph for the 0.8m threshold, note how most of the inconsistency is bounded at this value. In most of these instances no collisions have taken place and the inconsistency shown is solely that of the avatar, i.e. the error between the local and remote representation of the controlling entity. As latency is set to zero, this error is limited to the value of the error threshold employed by the avatar's DR model. When the inconsistency exceeds the threshold value, it is clear that a collision has taken place and the resulting measure includes the error in both the avatar's position and that of the environmental entity with which the avatar collided. Similar observations can be made for the other two graphs.

Furthermore, it can be observed that the use of a tighter threshold produces a reduction in the magnitude of the peaks of the physics-aware inconsistency. In general, utilising a tighter threshold for a DR model produces less inconsistency in physics-aware entities at the time of collisions. However, employing a tight threshold for the entire simulation means that more unnecessary updates are sent at times when no collisions are taking place and, as such, the bandwidth is being flooded with unnecessary packets of information. Hence, a DR with adaptive thresholds offers the optimal solution, whereby extra update packets are only sent when collisions are imminent.

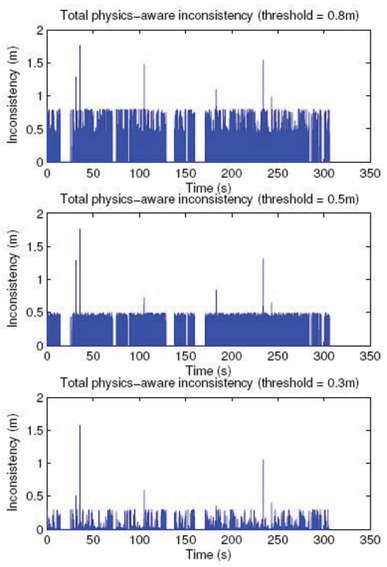

Figure 8: Physics-aware inconsistency for entity following sample path, with three fixed threshold DR models, 0.8m, 0.5m and 0.3m respectively.

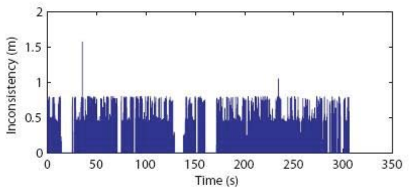

Figure 9: Physics-aware inconsistency for entity following sample path, with adaptive threshold DR model, using thresholds of 0.8 and 0.3m respectively.

c) Using DR with adaptive thresholds

Here, an adaptive DR threshold is employed, in which the threshold is switched between two different values, 0.8m and 0.3m respectively, in response to a prediction of possible imminent collisions. The results are given in Figure 9. With careful examination, we can see that, in general, the DR model uses the higher threshold while no collisions are predicted. Hence the inconsistency, for the most part, is bounded by the upper threshold of 0.8m. When collisions are predicted, and subsequently occur, the DR model employs the lower threshold bound. This can be deduced from the peaks of the inconsistency in all physics-aware entities, which have similar magnitudes as those obtained for the 0.3m fixed threshold in Figure 8.

The results clearly show that the adaptive threshold DR model is performing as expected, using the lower threshold when collisions are predicted and using the higher threshold otherwise. However, it should be noted that in some instances inconsistency seems to be bounded to the lower threshold of 0.3m. This occurs when collisions are predicted but do not actually occur. Hence the avatar's entity is unnecessarily updated when its error exceeds the lower bound. A better prediction model would alleviate this issue.

VI CONCLUDING DISCUSSION

A novel physics-aware, adaptive threshold dead reckoning technique for entity state management has been proposed for peer-to-peer DIAs. Inconsistency, in this case, is taken as the overall physics-aware inconsistency present in the application, or the sum of the spatial inconsistencies in both the local peer's controlled entity and all physics-aware entities. Simulation results illustrate the usefulness of the proposed technique. They show that adapting the dead reckoning model to utilise a tighter threshold for the motion of controlled entities in advance of collisions can lead to more accurate simulation of those collisions. In addition, the threshold remains looser (or larger) in cases where collisions are not predicted and, as such, the bandwidth is freed up to be utilised in a more optimal manner.

Future work will examine the performance of the proposed technique for more realistic virtual environments.

VII REFERENCES

[1] M. R. Macedonia and M. J. Zyda, "A Taxonomy for Networked Virtual Environments", *IEEE Multimedia Systems' 98*, 4(1), 1997, pp. 48-56.

[2] E. Churchill, D. Snowdon and A. Munro, "*Collaborative Virtual Environments: Digital Places and Spaces for Interaction*", UK: Springer- Verlag, 2002.

[3] S. Singhal and M. Zyda, "*Networked Virtual Environments: Design and Implementation*", New York ACM Press/Addison-Wesley Publishing Co., 1999.

[4] "IEEE Standard for Distributed Interactive Simulation - Application Protocols", 1998, *IEEE Std 1278.1a-1998.*

[5] W. Cai, F.B.S. Lee and L. Chen, "An auto-adaptive dead reckoning algorithm for distributed interactive simulation", *Proceedings of the thirteenth workshop on Parallel and distributed simulation,* Atlanta, Georgia, US: IEEE Computer Society, 1999.

[6] F.B.S. Lee, W. Cai, S.J. Turner and L. Chen, "Adaptive Dead Reckoning Algorithms for Distributed Interactive Simulation", *Int. Journal of Simulation: Systems, Science & Technology,* 1**,** 2000, pp.21-34.

[7] E. Catto, "*Box2D User Manual",* 2007, www.box2d.org/manual.html (accessed 23/09/11).