Vector-Field Consistency for .NET Multiplayer Games
Extended Abstract

Dinis Lage
dinis.lage@ist.utl.pt

Instituto Superior Técnico de Lisboa/INESC-ID
Av. Rovisco Pais, 1049-001 Lisboa

Abstract: Multiplayer games are currently one of the major sources of entertainment worldwide. With the widespread use of the Internet it becomes essential to use the available communication resources effectively allowing the best interaction possible for the players. There already are many techniques and models dedicated to improve the gameplay experience and increase the maximum number of participants supported in each session.

The problems of consistency and scalability in terms of computational and communication cost are addressed in this work. There are several solutions that essentially try to balance between flexibility, consistency and performance of distributed systems and simulations.

The vector-field consistency model and its applicability to multiplayer games are thoroughly examined in this work.

Keywords: multiplayer games, locality-awareness, scalability, playability, consistency management

1 Introduction

Multiplayer games currently represent one of the most popular uses of group interaction on the Internet. The real-time pace of many of these games has special interest because these require short response and communication periods. By controlling the quantity of information transmitted and adapting the consistency, depending on the interest level of each player, it is possible to increase the number of supported participants and improve the way in which they interact in the virtual world.

Multiplayer games require some consistency to be enforced in order to allow a smooth interaction of the players. The requirements vary on each type of game and the game in itself.

Another important aspect of multiplayer gaming that motivates this work is the urge for efficient methods of consistency enforcement due to its usual real-time pace.

Understandably, to accommodate multiplayer games consistency and performance requirements, an effective optimistic approach to the consistency enforcement should be used.

1.1 Objectives and contributions

The work presented in this thesis has the following objectives:

1. Make an assessment of the work related to networking in multiplayer games, consistency enforcement mechanisms and the use of locality-awareness in those mechanisms;
2. Adapt and apply the VFC (*Vector-Field Consistency* [1]) model to a multiplayer shooter game and analyse its impact on the gameplay and performance in a distributed environment of personal computers;
3. Implement a prototype XNA game that uses VFC and evaluate its performance;
4. Evaluate the complexity required in order to use the VFC in the game communication.

1.2 Document Roadmap

In section 2 we discuss some work related to consistency enforcement and communication mechanisms used in multiplayer online games. Section 3 describes the architecture for consistency enforcement in multiplayer games, used on the adaptation of Vector-Field Consistency to a multiplayer action game. In section 4 we describe the implementation details of the prototype game. Section 5 exposes the evaluation methodology and the results that were obtained while evaluating the prototype. Finally, in section 6 we summarize the achievements and make some observations for the forthcoming work.

2 Related Work

In this section, some relevant consistency mechanisms, with possible application in multiplayer games, are discussed. This section begins with an introduction to some general concepts that can be applied to these games and then expose some models that use locality-awareness in multiplayer games. Section 2.3 presents some general requirements depending on the game genre. In 2.4 an analysis is made of multiplayer games' architecture and some examples are given of actual implementations in prototype and commercial games. In section 2.5 an assessment of the work related with networking in multiplayer games is made.

2.1 Consistency Mechanisms

In any distributed system there is a need for mechanism to enforce some type of consistency of the shared data. These mechanisms are typically separated in two groups: pessimistic and optimistic. Following we present an analysis of mechanisms used to enforce consistency among replicas.

2.1.1 Pessimistic Replication

Traditional replication techniques try to maintain an approximation to a single-copy of data, giving users the illusion that there is only one highly available copy. There are many ways of accomplishing this objective but, in general, the mechanism blocks access to data when it is not possible to prove it is up to date. This is the reason why these are so-called pessimistic techniques.

2.1.2 Optimistic Replication

The algorithms of optimistic replication offer many advantages over the pessimistic ones: improved availability of data; flexibility with respect to networking, even with variable and/or unknown communication channels; should be able to scale to a large number of replicas because they require little synchronization among them; replicas and users are highly autonomous allowing replicas to be added with no change to the existing ones and also asynchronous collaboration between users like in version control systems as CVS [2] and SVN [3]; finally, optimistic algorithms provide quick feedback since they can apply updates tentatively as soon as they are submitted.

However, there is a big challenge in the optimistic techniques which is the possibility of diverging replicas and concurrent operations. Thus, it is only applicable to applications that can tolerate occasional conflicts and inconsistent data.

Availability vs. Consistency

In applications that can tolerate relaxed consistency there were different optimistic consistency models proposed [4-7]. Unfortunately, typically, optimistic models do not impose limits on the consistency of data that is made available to users. To solve this and limit the inconsistency of the objects some models emerged that tried to balance between availability and consistency.

Real-time guarantees [8] allow obsolete object replicas to be used for a specified time without confirming its freshness. During those periods the system allows the data to be used even if it is stale, reducing considerably the cost of managing that replicated data and improving the availability.

Order bounding [9] may be used to limit the number of uncommitted changes that may be applied to a replica. This allows transactions to proceed faster because a bounded number of preceding transactions can be ignored.

Numeric bounding is based on the definition of acceptable limits for the number of possible updates for each replica. When the quota is exhausted in a given replica, it must be made consistent with the others. This concept was first introduced with TACT [10],[11].

2.2 Locality Awareness in multiplayer games

The notions found in locality awareness are fundamentally related with the concept of interest management, used to filter a large number of data in large-scale distributed simulations [12]. By communicating only relevant data to other nodes it is possible to optimize the use of the communication channel.

2.3 Main Genres of Multiplayer Games

In order to improve the network performance in interactive multiplayer games it is necessary to understand the types of traffic generated in each game genre [13]. The communication network requirements of most of the games can be inserted in one of the following categories or, in some cases, might have a group of elements from each game genre:

First Person Shooters (FPS) – this game genre involves a big part of combat that requires rapid response times. It is necessary to have a great number of messages, such as position updates or players' actions. In [14] a study was made that shows that the latency starts being noticed at about 100ms and that at approximately 200ms the game playability becomes impossible. This game genre has the highest latency requirements.

Role Playing Games (RPG) – visually, this game genre is similar to FPS but the speed of the interactions of the players is less intense. Online RPGs usually aim at supporting a large number of players, and so, require good scalability. These games require the most in terms of concurrent game flows.

Real-Time Strategy (RTS) – in this game genre the effect of latency, even if easily detectable by the player, does not interfere in the playability nor in the performance of the player due to its nature that clearly promotes the strategy over real-time aspects as studied in [15] and [16].

2.4 Multiplayer Games Architecture

Nowadays, most of the commercial games are implemented with client-server architecture, using only one or a cluster of servers. Each client has an independent connection to the server and all the communication is made through it. This way it is easier to reduce the possibilities of cheating, assure the anonymity of the players and the administration is simplified. In this architecture the way we can deal with the generated traffic differs between server to client and client to server, for the following aspects:

Server to client: servers identify groups of clients to which it sends the same information. Servers can use multicast as a way to improve the efficiency in the network use or, even if that is not possible, it is possible to prioritize or group the messages.

Client to server: clients may generate events periodically or because of user interaction. Usually clients only communicate with a server and not directly between them.

2.5 Assessment

Optimistic consistency mechanisms favour significant improvements in the availability and performance of a system by relaxing the requirement for an exact match between replicas.

Client-server architecture provides a single location for the coordination of the state of shared data, including coordination of the consistency level that must be applied to each replica. When choosing peer-to-peer for the system architecture many other concerns must be accounted for (e.g. synchronization of the permitted actions).

Optimist consistency mechanisms typically apply divergence limits on the data by limiting: the interval in which a replica must be updated; the number of updates before a refresh must be made; the difference in value allowed for a replica. These limits are usually enforced to the data in general disregarding the importance of the locality and importance of the data to each user. By employing some awareness of the locality of interest of each participant, it is possible to prioritize and reserve resources where they are most needed.

Multiplayer games represent systems which allow for an optimistic approach to be made since these can tolerate some inconsistencies. Particularly, for FPS games, we propose a mechanism for consistency enforcement that is: **optimistic; centralized;** and **location-aware** (inside the virtual world).

3 Architecture

This section describes the architecture for consistency enforcement in multiplayer games, used on the adaptation of Vector-Field Consistency to a multiplayer action game.

Our solution was based on a XNA starter kit game called *Net Rumble*, modified to use client-server architecture, which would use a VFC model.

To analyse the VFC model, originally proposed in [1], it is now proposed to implement it on a multiplayer game in an environment of distributed personal computers. In this section we describe the design of our solution.

In this section we detail some important considerations of the VFC model. In 3.2 we describe possible approaches and considerations of the system architecture. Finally, in 3.3 the main aspects of the software architecture are explained, describing the platform used in this work and denoting some considerations about the integration of the specific layers.

3.1 VFC Model

In VFC, each object is positioned in an N-dimensional space. Each node of the network has a local replica of the virtual world, its view. VFC delineates how the bounded inconsistencies between views are managed.

Within each view, object consistency depends on their distance to a *pivot*. Pivots are characterized by a position in the virtual world and can move over time. Figure 1 illustrates a virtual world populated with o_1, o_2, o_3, o_4 and o_5, with the consistency zones z_1, z_2, z_3 and z_4. Depending on the object's corresponding consistency zone, different requirements may apply.

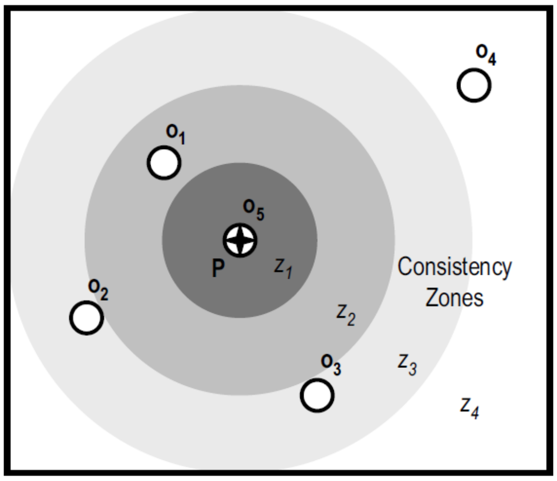

Figure 1 - Consistency zones centred on a pivot within the virtual world

Each zone maps to a consistency degree of a consistency scale. This scale is an ordered set of consistency degrees, which specify the consistency to be enforced for each zone. Determining the consistency degree of an object depends on its relative position with respect to the pivots.

VFC describes the consistency degrees as a 3-dimensional vector. Each dimension is a numerical scalar that defines the maximum divergence of the constraints of *time*, *sequence* and *value*.

The *time* dimension specifies the maximum period allowed for a replica not to be refreshed with its latest value, regardless of the number of updates performed during that interval.

Sequence represents the maximum number of replica updates that can be lost, that is, updates not applied to a replica.

Value represents the maximum relative difference between the replica contents against: the actual value (e.g. the difference between a player’s position and the replica’s value for it cannot exceed a certain value); or a constant (e.g. a player’s score is approaching the top value).

To enforce this consistency model a coordinator is required. Its role is to control whether or not the consistency requirements for each pair of objects and pivots are being complied and act accordingly.

3.2 System Architecture

To make a comprehensive study of the effects of the use of VFC in the communication it is important to use multiple methods in the communication and compare the impact it has on all of them. Following we compare the approaches to the system architectures of client-server and peer-to-peer. We further describe the interaction in a multiplayer game in each of these architectures and the differences of behaviour in regards of consistency enforcement.

Using client-server communication, the determination of the interest areas around each pivot is simplified because the server has the correct data for all players. Given this, the server can decide where to send the information needed and ensure that the consistency requirements are enforced.

Besides security issues, that are out of the scope of this work, a peer-to-peer implementation also aggravates the work required to comply with the game rules. In this type of architecture, when allowing some divergence between the replicas, situations may happen where a certain player performs an action in a moment when it would not be allowed by some of the participants and disallowed by other. This would require an agreement mechanism which would increase the complexity of the coordination process.

Considering that, it is better to implement the game in a client-server structure in which all clients send the desired actions to the server and only consider updates to the game state sent by the server. To minimize the delay between the player’s actions and its application to the controlled entity in the game, it can suppose that his action will be accepted by the server and correct possible differences in the execution (e.g. a packet sent to the server is lost).

3.3 Software Architecture

When designing an implementation of VFC in a multiplayer game it is advantageous to approach it as an independent module that abstracts from the game logic and grants the possibility to use it in different games with minimal effort.

Besides the advantages exposed in the previous section, in client-server architecture, clients do not need to be aware of the use of VFC. For clients, the implementation of VFC is then transparent. In a peer-to-peer architecture all nodes would need to have the VFC logic and, in order to enforce a certain consistency, would need to settle the parameters among themselves.

3.3.1 XNA

XNA™ is a set of tools with a managed runtime environment provided by Microsoft® that facilitates video game development and management.

The XNA Framework is the .NET-based framework for development of video games and simulations for deployment on Windows

PC, Xbox 360 or Zune. It includes an extensive set of class libraries, specific to game development, to promote maximum code reuse across target platforms.

The XNA Framework was developed with two primary goals: to enable cross-platform game development; and to simplify game development. To describe the XNA Framework it is appropriate to think of it as a series of layers. In a bottom-up approach these are:

Platform – the lowest layer which consists of the low-level native and managed APIs that the framework is built on top of;

Core Framework – provides the core functionality extended by the other layers;

Extended Framework – group of components focused on easing game development;

Games – the highest layer which groups the game specific content

3.3.2 VFC Integration

Typically, in XNA and multiplayer games in general, a common method to update players' actions and game state is to use packets containing relevant information. The selection of dispensable packets can be placed in an alternate method that, instead of updating all clients, enforces VFC and only sends the update to the relevant participants.

3.3.3 Net Rumble Considerations

Net Rumble[1] is a complete XNA Game Studio game available under the Microsoft Permissive License. It consists of a two-dimensional space shooter where the players compete inside an arena filled with asteroids and power-ups. This game can be included in the FPS genre by its network communication requirements as well as type of gameplay. So, it makes it very suitable for the study of the effects of the implementation of VFC since it belongs to the kind of game with the most strict requirements, as seen in 2.3.

4 Implementation

In this section we present the relevant difficulties and issues that arose during the development, explaining how these were overcome.

Besides the necessary modifications to use VFC as the communication model, several features were added to help or improve the study of the impact of the model.

Section 4.1 describes the details related with the changes that were made in the communication used in the game. Next, in section 4.2, the implementation of the VFC model applied to the Net Rumble prototype are explained. Section 4.3 we explain the automation and testing process of the prototype achieved.

4.1 Communication

The communication in Net Rumble is done by sending update messages between participants in the game session. For VFC to be enforced some changes in the game communication system were required.

By centralizing the responsibility for the correct state of the game it'd be possible to later make accurate decisions related to locality interest.

4.2 VFC Model

The implementation of the VFC model is made as a module that can be easily integrated and detached of a similar game. It abstracts from the specific game logic in which it was implemented.

4.2.1 Pivot

Pivots are the centre of the consistency zones for each player. Since each player only controls one ship the pivot is given by the ship's position. In XNA's Gamer Services system, each gamer has a *tag* object to which the programmer can add required information (e.g. ship colour). The class that represents that information (*PlayerData*) then implements the *IVFCPivot* interface for the middleware to be able to extract its position.

In addition to the position the interface also requires an ID to be defined so it can later identify each pivot in the lists of maintained consistencies.

4.2.2 Objects

Each object, for which updates are to be managed using VFC, implements *IVFCObject* interface. The object has a variable that specifies the consistency limits, made of an array of VFC zones. Furthermore, it contains a dictionary with the status vector for each player which is managed by the VFC middleware. This was done to encapsulate the necessary information, avoiding unique IDs for all objects. The logic that gives the current *value* VFC dimension is defined in the implementation of this interface as well.

In this game the VFC objects are ships and asteroids.

[1] http://creators.xna.com/en-US/starterkit/netrumble

4.2.3 Dimensions

An essential procedure to use the VFC model is to define the logical dimensions to be used by the algorithm.

To calculate the distance of objects and pivots, its positions in the virtual world were used. The distance then translates into a specified consistency zone.

Time and sequence dimensions of the consistency vector correspond to the number of updates of each object and the time it was last updated. For the value dimension, which is a qualitative dimension, the length of the velocity of each object was used.

4.2.4 Consistency Enforcement

In this implementation the consistency is enforced in the method used to send packets to all players. This is encapsulated in the class *VFCManager*. Since a two-dimensional space is being considered here, the distance between pivots and other objects is given by $\sqrt{(X_1 - X_2)^2 + (Y_1 - Y_2)^2}$.

4.3 Automation and Testing

To conduct the tests and be able to automate players' and gather data about the gameplay in order to further study the behaviour of the VFC model along the game.

4.3.1 Automated Player Simulation

The need for automation of the ships controller was evident when trying to test the game. This led to the development of some basic artificial intelligence used to control the ships. To simulate the input the terminal would just follow some basic logic to keep wandering inside the arena and shooting randomly until an enemy ship approached. Then, it enters a pursuit mode in which it tries to eliminate the opponent.

By ensuring the ships would constantly move and change their actions this certainly stresses the communication as well, or better, than a human player.

4.3.2 Record game data

During the gameplay, the necessary data is recorded in CSV format in order to later process the meaningful information to elaborate on the impact of VFC on the game communication mechanism.

The data recorded include elapsed time, players' positions, and average amount of data being sent and received. This enables a further study of the divergence that occurred. It also allows a deeper analysis of the effect of the VFC model on the performance and consistency.

4.3.3 Deployment

There is a XNA limitation to run only one game at a time on each terminal. This diminishes considerably the number of participants for testing purposes. To circumvent this, the *VMware Workstation* is used to increase the number of participants in each session. With this virtualization and simulating adverse network conditions it is then possible to examine, or at least speculate, the behaviour of the game in actual conditions (e.g. over the Internet).

5 Evaluation

This section presents the methods we used in the evaluation process and the results obtained in the experiments. The evaluation is done both qualitatively and quantitatively.

5.1 Evaluation Method

There are several methods for evaluating the effect of the implementation of VFC on a game. In this section it is discussed what the relevant criteria are, and how they should or should not be affected. These range from accordance to the game rules to the bandwidth consumed in the course of the game:

Bandwidth – Given that the comparison made to a model that does no filtering of messages and sends everything to all players, the use of VFC should greatly reduce the bandwidth used in the course of the game.

Latency – The latency has an important role on the gameplay experience of players. For a given pivot, the latency grows with the virtual distance to it. It is important to measure the latency in each zone and compare it with the latency on without using VFC. It would also be interesting to possibly reduce the average latency.

Frame-rate – the calculations needed to use the VFC model should be as light as possible and should not have a negative impact on the frame-rate possible. Preferably, by reducing the number of messages that need to be processed by each terminal, it should be possible to increase the frame-rate to improve user experience.

User experience (playability) – an important factor on the evaluation of the work since all attempts to improve game communication have an essential goal: to improve user experience. The impact that the use of VFC has on the gameplay is an important study object. An experiment related to this is to simulate network conditions and compare the latency and packet loss that can be sustained by VFC.

Compliance with game rules – it is crucial for the game logic not to be modified and the rules must still be obeyed, even if some divergence is *temporarily* allowed. A major factor is that each player has, at least, enough information that does not affect negatively his decisions. It should as well have no effect on the score in the game, not favouring even the host player.

5.2 Results

The tests were run on two machines: a desktop computer with a quad-core AMD Phenom™ II X4 945 Processor 3.00 GHz, with 8 GB of RAM, running Windows 7 64-bit OS; a laptop computer with a Intel® Core™2 Duo T6400 at 2.00 GHz with 4 GB of RAM. Each virtual machine had 512MB of RAM allocated and access to one processor and had Windows XP 32-bit OS. All virtual machines had to have Microsoft® Visual Studio® 2008 and XNA Game Studio 3.1 installed, both needed for the use of networking provided by the XNA framework. The host player was run on the desktop machine and clients on virtual machines both on the desktop and on the laptop. The machines, even the virtual machines (which was bridged to the physical computers), are connected through a 100 Mbps network.

Multiple tests were run with each configuration to verify the integrity of the data acquired during gameplay.

The data regarding network and players' positions was recorded at a rate of about 5 samples per second (0.2s intervals).

The tests were run without the use of VFC, transmitting all packets, and with a VFC configuration that uses the definitions on Table 1 for the consistency vectors.

	Range	Time (s)	Sequence	ValueDiff
Zone1	500	0	0	0
Zone2	1000	0.3	5	2
Zone3	2000	0.5	10	5
Zone4	5000	1	20	10
Zone5	∞	3	40	20

Table 1 – VFC Zones with strict constraints

5.2.1 Bandwidth

The relevant data for the study of bandwidth usage are: data received by each client; data sent by server.

The data presented here refers to specific cases that can be used as examples for typical game progress. Surely there are variations in the traffic observed depending on the actual game session but these negligible in the comparisons made.

Since there is no filtering on the messages that the client sends to the server there is no interest in analysing neither the data sent by clients nor data received by the server. These, as expected, are not affected by the usage of VFC. The maximum data sent by each client is around 4,5kBps.

There was a substantial reduction of the rate at which data was transferred throughout the game with reduction of the average rate ranging from 25% for two players to 64% with ten players on the data sent by the server. On the client-side reductions on the rate of the amount of data received ranged from 25% with two players to 65% with ten players.

5.2.2 Divergence

Measuring the divergence on the clients turned out to be very complex. The game time has an intrinsic error among the participants in the game. This introduces a divergence in the positions between the data recorded in the host and other participants. The divergence due to this lack of synchronization of the data recorded during the game was too significant, so, it did not allow us to measure the real difference of the positions at a certain time frame during the game.

5.2.3 Latency

Due to the difficulties previously mentioned about the measurement of the divergence of the views of each client there was no precise method to measure the differences when using VFC and not with simulated conditions.

In relation to packet loss simulation it revealed a fine equilibrium. The fact that there are fewer packets being transmitted means that there are also less packets dropped. On the other hand, there is a higher probability of a packet of greater importance (in a tight consistency zone) being dropped since these represent most of the traffic.

5.2.4 User Experience

The playability of the game was maintained with the decrease in network usage provided by the use of VFC. There were occasional controlled inconsistencies observed when viewing a large portion of the game arena (using the zoom feature) but these happened outside the radius of action of the player so had no effect on the players' decisions. The information each player had was accurate enough to make a good decision because the deviation to the real position of other players does not have an impact on, for example, the direction the player had to choose to move towards an opponent or shoot.

5.2.5 Frame-rate

Using a component to draw the frame-rate in-game, it was possible to observe the frame-rate during the game. It suffered no effect from the use of VFC. This is a rather desirable situation since, if a decrease on the frame-rate was observed, it would possibly mean that the VFC model was unsuitable to be used in multiplayer games.

5.2.6 Summary of results

The results obtained with the testing of our solution are very encouraging. We were able to extend the XNA framework in order to implement new communication patterns and a new consistency model. This allows users to play games with larger scenarios and/or with an enlarged view of the playing field, which improves playability. We are able to provide this by reducing the network usage (both in number of messages as well as in total bandwidth). With the obtained results, many games can be played with good frame rates inside LANs and even on wide area networks.

The tests have also shown us good signs of scalability of VFC since the reduction of the amount of data transferred was increasingly higher. This represents a decrease on the processing load both to transfer and to process the packets. Employing scenarios big enough for the playing field not to be saturated, VFC is expected to scale smoothly.

6 Conclusions

The network communication in multiplayer games is an object of extensive study and wide interest. Despite all the work done in the area there is yet to be a solution that can be generalized and applied intuitively. Some previous work incorporates the notions of locality of interest or consistency radius but, an all-or-nothing approximation is usually adopted.

VFC is an intuitive and flexible model that is easily translated to most games' semantics.

Reiterating over the objectives of this thesis:

1. An assessment of the related work was made that lead us to some specific decisions about the architecture of the system that was implemented;
2. The VFC model was successfully applied to a shooter game developed in XNA for personal computers with minimal impact in the gameplay and very encouraging performance results;
3. The XNA game implemented, using the developed VFC module had very good performance and resulted in high bandwidth savings and, consequently, avoided the need to process much of the data previously transferred between server and client;
4. There is a considerable ease in the use of VFC for the game communication due to the modularity of the solution implemented which allowed us to easily switch between methods for prioritizing data used in the communication. It was possible to use VFC with a very small number of changes to the client-server approach of the game.

We were able to implement a prototype of a game provided a case study for the test of the VFC model. VFC proved to be an efficient model that escalates well.

The problems discussed in this work apply to a wide range of practical applications, other than distributed games.

Consistency mechanisms are used in distributed systems in order to manage the divergence of replicas. By allowing controlled divergence among replicas it is possible to considerably increase the availability of systems. When applying these techniques to multiplayer games it is valuable to introduce locality awareness to sophisticated consistency models such as TACT. VFC provides a unified model with simple and flexible abstractions that allow it to be intuitively expressed according to the application semantics.

As multiplayer games vary in genre, its requirements for consistency also change. VFC appears to be fit for any game genre since in all games there is some locality in the players' interest.

There are numerous combinations that can be used in the design of a system for multiplayer games. In terms of network organization of the nodes there are two main architectures: client-server and peer-to-peer. Frequently, in peer-to-peer systems there are portions of the applications' logic that are centralized or grouped and distributed among the nodes (e.g. portions of a game playing field).

The design of a consistency enforcement system for a multiplayer game must take into account numerous factors that may influence the user experience. The number of controlled entities, the importance of the information that may be disregarded, are some of the factors that must be considered.

Implementing this consistency system in a modular manner allows seamless integration with further games and comparison with other alternatives.

It is crucial to have the means to test and study the behaviour of a system when new models are applied and it is useful to automate the testing process.

By evaluating the VFC behaviour we were able to verify its applicability to game semantics and empirically observe the effects it has on the game.

6.1 Future Work

In the future, we intend to pursue the following lines of investigation.

It would be interesting to study the performance of VFC compared to other consistency enforcement models, even if these have an all-or-nothing approach.

The operation in a peer-to-peer architecture requires further investigation in order to comprehend the implications and requirements to enforce VFC.

An interesting exercise would be to extend the VFC model implemented with dynamic consistency requirements. This needs to have a mechanism to either limit the network or enough resources to reach a saturation point of the network used.

The implementation analysed in this work only accounts for one pivot. A study can be made of the behaviour of VFC using a multi-pivot system, e.g. applied to a real-time strategy game. One factor that should be taken into account is that there might be a significant increase in the necessary effort to determine the consistency that needs to be enforced.

7 References

[1] N. Santos, L. Veiga, and P. Ferreira, “Vector-Field Consistency for Ad-Hoc Gaming,” *Middleware 2007*, 2007, pp. 80-100.

[2] P. Cederqvist and R. Pesch, *Version Management with CVS*, Network Theory Ltd., 2002.

[3] B. Collins-Sussman, B.W. Fitzpatrick, and C.M. Pilato, *Version Control With Subversion*, O'Reilly Media, Inc., 2004.

[4] D.B. Terry, M.M. Theimer, K. Petersen, A.J. Demers, M.J. Spreitzer, and C.H. Hauser, “Managing update conflicts in Bayou, a weakly connected replicated storage system,” *Proceedings of the fifteenth ACM symposium on Operating systems principles*, 1995, pp. 172-182.

[5] R.G. Guy, J.S. Heidemann, W. Mak, T.W. Page Jr, G.J. Popek, and D. Rothmeier, “Implementation of the Ficus replicated file system,” *USENIX Conference Proceedings*, vol. 74, 1990, pp. 63-71.

[6] J.J. Kistler and M. Satyanarayanan, “Disconnected operation in the Coda file system,” *ACM Transactions on Computer Systems*, vol. 10, 1992, pp. 3-25.

[7] R. Guy, P. Reiher, D. Ratner, M. Gunter, W. Ma, and G. Popek, “Rumor: Mobile Data Access Through Optimistic Peer-to-Peer Replication,” *Advances in Database Technologies: ER'98 Workshops on Data Warehousing and Data Mining, Mobile Data Access, and Collaborative Work Support and Spatio-Temporal Data Management, Singapore, November 19-20, 1998: Proceedings*, 1999.

[8] Y. Saito and M. Shapiro, “Optimistic replication,” *ACM Computing Surveys (CSUR)*, vol. 37, 2005, pp. 42-81.

[9] N. Krishnakumar and A.J. Bernstein, “Bounded ignorance: a technique for increasing concurrency in a replicated system,” *ACM Transactions on Database Systems (TODS)*, vol. 19, 1994, pp. 586-625.

[10] H. Yu and A. Vahdat, “Design and evaluation of a conit-based continuous consistency model for replicated services,” *ACM Transactions on Computer Systems (TOCS)*, 2002, pp. 239-282.

[11] H. Yu, H. Yu, and A. Vahdat, “Building replicated Internet services using TACT: a toolkit for tunable availability and consistency tradeoffs,” *Advanced Issues of E-Commerce and Web-Based Information Systems, 2000. WECWIS 2000. Second International Workshop on*, 2000, pp. 75-84.

[12] K.L. Morse, *Interest management in large-scale distributed simulations*, Citeseer, 1996.

[13] C. Majewski, C. Griwodz, and P. Halvorsen, “Translating latency requirements into resource requirements for game traffic,” *to appear Proceedings of the International Network Conference (INC’06), Samos*, Citeseer, 2006.

[14] L. Pantel and L.C. Wolf, “On the impact of delay on real-time multiplayer games,” *Proceedings of the 12th international workshop on Network and operating systems support for digital audio and video*, Miami, Florida, USA: ACM, 2002, pp. 23-29.

[15] P. Bettner and M. Terrano, “1500 Archers on a 28.8: Network Programming in Age of Empires and

Beyond," *Presented at GDC2001*, vol. 2, 2001, p. 30p.

[16] N. Sheldon, E. Girard, S. Borg, M. Claypool, and E. Agu, "The effect of latency on user performance in Warcraft III," *Proceedings of the 2nd workshop on Network and system support for games*, Redwood City, California: ACM, 2003, pp. 3-14.