VFC-RTS: Vector-Field Consistency para Real-Time-Strategy Multiplayer Games

Manuel Cajada, Paulo Ferreira, and Luís Veiga

INESC-ID/Instituto Superior Técnico
Distributed Systems Group
cajadas@gmail.com, [paulo.ferreira, luis.veiga]@inesc-id.pt

Abstract. Although massive multiplayer online games have established popularity, real-time strategy (RTS) has not been considered a good candidate for this model because of the limited number of players supported, large number of game entities and strong consistency requirements. To deal with this situation, concepts such as continuous consistency and location-awareness have proven useful in order to confine areas with consistency requirements. The Vector-Field Consistency (VFC) model achieves a balance between these notions by defining multiple zones of consistency around the player's location (pivot) with different divergence boundaries. In this work we propose VFC-RTS, an adaptation of the VFC model, characterized for establishing consistency degrees, to the RTS scenario, describing how the concepts of the original VFC model were adapted. We apply our solution to an open source RTS game with full consistency requirements and evaluate the results.

1 Introduction

With the increase of the internet services capabilities over the years there has also been a growth in the new types of real-time applications. One that has been gaining most popularity are massive multiplayer online games (MMOG).

The category most played and with the greatest popularity are the role-playing games (MMORPG), however, there are other real-time network game categories which have been less discussed for using the MMOGs model, among them the real-time strategy games (RTS)[2]. Maintaining game state consistency on a RTS game is a complex task. To reduce the amount of data on a state update process and still provide the user with the relevant state information, MMOGs employ *interest management* (IM) techniques[1]. This concept restricts the amount of data exchanged during the update process by limiting the area of interest (AoI) of a player. Thus, the player only receives the game state data necessary, mostly corresponding to the area where avatars and events relevant to him are located and take place.

This work proposes an adaptation of the Vector-Field Consistency (VFC) model[3,4] to the needs of the RTS games, presenting the architecture for a generic middleware and evaluation of the solution when applied to *WarZone 2100* (http://wz2100.net), an open source, multi-platform RTS game with full consistency requirements.

2 Adapting VFC to Real-Time Strategy games

The proposed adaptation of the VFC model is composed of two parts: adaptation of the player's area-of-interest to the map area captured by the camera and aggregation of entities into units according to the distance among them. By assigning the pivot point to the center of the player's camera view, we guarantee that every object within the player's area of visibility presents an acceptable divergence degree for the correct execution of the game. We believe this pivot point attribution best serves the consistency requirements of real-time strategy since the majority of strategy commands and decisive battle events concerning the player take place inside his camera spectrum. Although the presented adaptation of VFC covers most of the player's area-of-interest, there can also be events affecting the user not followed by the camera. To cover this situation, we propose next an additional feature to the VFC-RTS solution - the concept of entity aggregation into units.

A unit is a group of entities controlled by the same player performing the same set of actions within a limited time period. If the system detects entities within a fixed distance threshold going on the same direction, it establishes a pivot point in the center of the unit formation and consistency zones. Not only this process allows following entities outside the camera span without compromising the consistency of the system, it also reduces the processing and communication load since update messages regarding entities of the group can be aggregated.

Middleware Architecture Following the original VFC model, the VFC-RTS distributed framework is composed of two types of network nodes: **client** nodes and a single **server** node (Fig. 1). Running an instance of the game, clients are accountable for any changes to the game state and submission of local replica modifications to the server node (**client update**). The server node is responsible for game state update reception, applying newly received updates to the global game state, solving possible conflicts between concurrent client updates and propagation of the most recent game state according to each player's consistency requirements (**state update**).

The client-side VFC-RTS Middleware considers a **Consistency Zone Monitor**, responsible for the detection of changes to the consistency zones configurations and forwarding them to the server (**consistency zone updates**). Whenever the player changes his camera view position (above the configured threshold value), or changes the location of a group or its configuration (i.e. number of entities in the group, max consistency radius), a new consistency zone message is dispatched.

The server presents two VFC-RTS enforcement components: the **Consistency Zone Manager**, holding all the information regarding every player's consistency zones; and the **VFC Manager**, responsible by enforcing the VFC algorithm in conformity with each client's consistency requirements.

Considering object o and client c, whenever the VFC Manager detects that o is tagged as dirty and does not respect the consistency boundaries set for c, the server sends c a State Update Message, with o's most consistent state (primary state). Furthermore, the server flags a state update regarding o was sent to c, preventing the server from repeatedly sending clients an object's state before it is again modified.

VFC Enforcement As a formal consistency degree of a zone, VFC uses a 3-dimensional consistency vector: **Time:**, maximum time (in seconds) a local object can stay without being refreshed with its primary replica's latest value; **Sequence:** maximum number of updates to the primary object state not applied to the local object (missing updates); **Value (ν):** maximum divergence between the contents an object's local state and its primary state, i.e., a measure of the error associated with the perceived location of an object (local replica) and its actual location (primary replica).

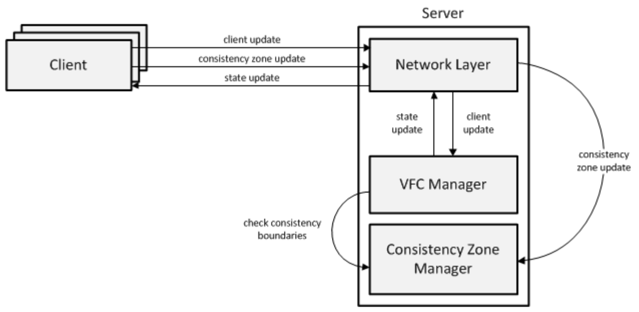

Fig. 1. Client-Server Message Flow

3 Evaluation

Number of Messages. We start by studying how applying the VFC algorithm affected server-client update propagation. Our tests were conducted on two, three, and four players sessions, varying for each case the size of the scenario (small, medium and large maps with a maximum capacity for two, four and eight players, respectively). We verified an overall reduction in the number of exchanged messages between 50% and 60%. Additionally, the tests exposed how the size of the scenario has a direct impact on the update exchange rate. This factor can be explained by the increasing gap between player's headquarters as the scenario gets wider. Hence, the bigger the scenario, the more extensive is the distance between the player's AoI.

Server CPU Performance. In the results (Fig. 2 and 3), we see an increase on the server CPU consumption in between 10 and 20 percent. This can be seen as a consequence of the additional workload required for enforcing the VFC algorithm. Despite this fact, such an increase on the server's computational requirements has little expression since CPU usage rarely surpasses 80%, presenting a consistent consumption rate during the entire game session, even in the stress case.

Frame Rate. Analyzing the experimental results, it is perceptible a slight downward trend on the after VFC, reaching a minimal value of 37 FPS, while before VFC it presents a minimal of 44 frames per second. We believe these may come as a consequence of changes made to the update message structure in order to enforce the VFC algorithm. Although a minimal value of 37 frames per second was reached during a single stress situation, when analyzing remaining tests, we can see that the most common minimal value its about 40 frames per second, very close to the minimal frame rate before applying VFC.

Qualitative Evaluation To evaluate whether the VFC model had a significant impact on user experience, we handed a questionnaire to every player by the end of a *Warzone 2100* session. The tests were conducted on two, three, four and seven players game sessions. The two, three and four players session were performed over

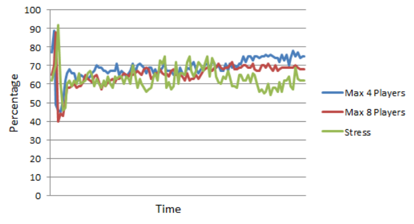

Fig. 2. CPU consumption before VFC

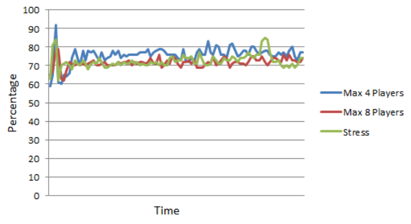

Fig. 3. CPU consumption after VFC

a fixed network while the seven players games over a wireless network, which were the only game sessions where *lag* was experienced. Upon the completion of the qualitative evaluation of VFC-RTS we could verify that: **(a)** Although the latency verified during the seven players sessions may have tampered the result of this study by causing object flickering, these events caused little impact on the players strategy; **(b)** The sessions conducted over a fixed network also present object flickering, however, the frequency of these occurrences is lower than on the game sessions over a wireless network, and presented no impact to the player's strategy or the game's outcome; **(c)** VFC-RTS holds little to none overall impact to the player's game experience, however, this value is subject to the latency of the system.

4 Conclusions

In this paper we presented a continuous consistency model which results from the adaptation of the VFC model to the RTS multiplayer environment. The architecture was implemented on top of *Warzone 2100*, an open-source RTS game with total consistency requirements. Other than showing the correctness of the adaptation and the little impact caused to the player's game experience, in comparison to the total consistency version, we achieve a reduction on network resources usage by the order of fifty percent. This way, a VFC powered RTS game can maintain strict, locality based, consistency levels without compromising the player's perception of the game.

Acknowledgements: This work was partially supported by national funds through FCT Fundação para a Ciência e a Tecnologia, under projects PTDC/EIA-EIA/102250/2008, PTDC/EIA-EIA/108963/2008, and PEst-OE/EEI/LA0021/2011.

References

1. J.-S. Boulanger, J. Kienzle, and C. Verbrugge. Comparing interest management algorithms for massively multiplayer games. In *Proceedings of 5th ACM SIGCOMM workshop on Network and system support for games*, NetGames '06, USA, 2006. ACM.
2. J. Müller and S. GORLATCH. Rokkatan: scaling an rts game design to the massively multiplayer realm. *Comput. Entertain.*, 4, July 2006.
3. N. Santos, L. Veiga, and P. Ferreira. Vector-field consistency for ad-hoc gaming. In R. Cerqueira and R. Campbell, editors, *Middleware 2007*, volume 4834 of *Lecture Notes in Computer Science*, pages 80–100. Springer Berlin / Heidelberg, 2007.
4. L. Veiga, A. Negro, N. Santos, and P. Ferreira. Unifying divergence bounding and locality awareness inreplicated systems with vector-field consistency. *Journal of Internet Services and Applications*, 1:95–115, 2010. 10.1007/s13174-010-0011-x.