高品質インターネット　5－15
(2002. 10. 11)

多人数参加可能なネットワーク型仮想空間共有アプリケーションにおける同期機構に関する考察

藤川　和利†　伊藤　敦史††　下條　真司†††　宮原　秀夫††

† 奈良先端科学技術大学院大学情報科学センター　〒630–0101 生駒市高山町 8916–5
† 大阪大学大学院情報科学研究科　〒560–8531 豊中市待兼山町 1–3
† 大阪大学サイバーメディアセンター　〒567–0047 茨木市美穂ヶ丘 5–1
E-mail: †fujikawa@itc.aist-nara.ac.jp, ††{atsusi-i,miyahara}@ist.osaka-u.ac.jp,
†††shimojo@cmc.osaka-u.ac.jp

あらまし　仮想空間を共有するアプリケーションにおいて，サーバを複数用いて同一の仮想空間を管理した場合，同じオブジェクトを複数のユーザがほぼ同時に取得しようとした場合，ネットワークの遅延等により，サーバ間で，異なる判定を行い，仮想空間の不整合が起こることがある．また，不公平な判定となる場合もある．このような問題を防ぐためには，通常待ち時間を設けることが考えられるが，このとき取得要求に対する応答が遅くなり，リアルタイム性が損なわれる．本論文では，ユーザの動きを予測し，その予測に基づくオブジェクトの取得確率を用いることで，応答性を高め，かつ，公平な判定が行える手法を提案する．また，本手法では，サーバ間での仮想空間の不整合が起こらないようにしている．
キーワード　クライアント・サーバアーキテクチャ，ネットワークゲーム

Synchronization Issues on Networked Virtual Environments of Large-Scale Multiparty Applications

Kazutoshi FUJIKAWA†, Atsushi ITOH††, Shinji SHIMOJO†††, and Hideo MIYAHARA††

† Information Technology Center, Nara Institute of Science and Technology　8916–5 Takayama, Ikoma, Nara, 630–0101 Japan
† Graduate School of Information Science and Technology, Osaka University　1–3 Machikaneyama, Toyonaka, Osaka, 560–8531 Japan
† Cybermedia Center, Osaka University　5–1 Mihogaoka, Ibaraki, Osaka, 567–0047 Japan
E-mail: †fujikawa@itc.aist-nara.ac.jp, ††{atsusi-i,miyahara}@ist.osaka-u.ac.jp,
†††shimojo@cmc.osaka-u.ac.jp

Abstract　Recently, since the Internet spreads rapidly and widely with high-speed backbone networks, Networked Virtual Environments (Net-VEs), where user can share a virtual space and interact with each other, become popular. Net-VEs deployed over the Internet, generally adopt a client-server architecture and each client transmit a user's action to a server asynchronously. Moreover, most of Net-VEs provides multiple servers for users from the viewpoint of the scalability of the number of users. As the requirements for Net-VEs, we can consider the consistency of a virtual space among users and the responsiveness and fairness to user's actions, especially competitive events. If each server in Net-VEs processes user's actions in order of reception, unfairness will arise although responsiveness can be provided. Moreover, inconsistency among servers may occur. On the other hand, if a server waits for user's action long time enough, responsiveness will be lost, although fairness can be guaranteed. In this paper, we discuss synchronization issues on Net-VEs and propose a new synchronization method among servers, which can guarantee the consistency of a virtual space and provide the responsiveness and fairness to user's actions.
Key words　Client-server architecture, Online network game

1. Introduction

Recently, since the Internet spreads rapidly and widely with high-speed backbone networks, Networked Virtual Environments (Net-VEs), where user can share a virtual space and interact with each other, become popular. As the architecture of Net-VEs, there are roughly two types: one is client-server type and the other is peer-to-peer (P2P) type. In the client-server type, some Net-VEs use multiple servers.

Most Net-VEs adopt a client-server type. In this architecture, users connect with a server and send "unauthorized updates" of a virtual space. A server select some of unauthorized updates as authorized ones, updates the virtual space according , and send "authorized update to users. Since a communication between users is done via a server, delay becomes large rather than a communication is done directly. In this architecture, although comparatively many users can participate simultaneously, the number of users is limited by a server performance and/or a network bandwidth.

A client-server type with multiple servers can make the number of users increase because it distributes users to multiple servers. Since communication between users might be done via two or more servers, delay may become large further than a client-server type with a single server.

In P2P type, as users communicate directly without a server, delay can become small. However, since the required processing performance in each peer and the required network bandwidth will increase if the number of users increases, only a small number can participate simultaneously.

There also exists the hybrid type which combined the client-server type and the P2P type as the architecture of Net-VEs. In this architecture, important data such as interaction between users are transmitted via a server, and unimportant data such as travel data are done directly between users.

In Net-VEs like online network games which shares virtual space between a lot of users, the virtual space displayed on each user's terminal must be consistent. Therefore, when multiple users try to acquire a object simultaneously in virtual space, Net-VEs must judge who can acquire it and synchronize the virtual space of every user. Such a situation is called "competitive event."

Handling a competitive event, If Net-VE waits for the data from all users in order to carry out a fair judgment, real-time nature will be lost. On the contrary, if Net-VE judges in the order of data received for real-time nature, the user of small delay will become advantageous, and the unfairness occurs among users. In the client-server type with a single server, inconsistency among users does not occur, because collective management of a virtual space can be done in a server. In client-server type with multiple servers and P2P type in which multiple copies of virtual space exist, inconsistency may occur depending on the way of a synchronization among those copies.

In a client-server type with multiple servers, there are two kinds of the method of virtual space management [7]. One is the method of dividing virtual space into several regions and managing each regions by the dedicated server. In this method , although the competition between servers is not generated, a client might change the server which it connect to by travel of the user in a virtual space. Therefore, delay may become large. The other is the method of that all servers manage the same virtual space and the number of users is limited by each server. In this method, if competition occurs between servers, the servers will negotiate with each other. Inconsistency of virtual space may occur depending on the negotiation method. In P2P type, since processing of an event and a judgment are performed by each peer, inconsistency of virtual space may occur between peer. Handling the competitive event, the trade-off between fairness and real-time nature must be considered. The way of synchronization will vary depending on that which is thought as important fairness and real-time nature. The most important thing in Net-VEs is to provide consistency of virtual space among users.

In this paper, we propose the method of guaranteeing fairness and real-time nature as much as possible for the competitive event in Net-VEs of client-server type with multiple servers which shares the same virtual space. The proposed method calculates the acquisition probability which user will acquire the object by using prediction of travel of users. To mitigate the influence of the gap of network delay between users as much as possible, each user connects with a near server on the Internet. Although a server has to wait for the data from all users in order to guarantee completely the fairness to object acquisition between users, it will wait even for data from the user who does not try to acquire the object, and real-time nature will be lost. At first the proposed method determine the waiting time for users and other servers and then shorten the waiting time by the acquisition probability. Thus, the proposed method can provide fairness and real-time nature simultaneously.

2. Related works

2.1 Network games

There are EverQuest [10] and Quake [11] as examples of a client- server type. EverQuest is a 3D-MMORPG (Massively Multiplayer Online Role-Playing Game). In EverQuest, tens of thousands of users are registered and about 200,000 users play all the time. The number of users which can partici-

pate simultaneously is more than 100,000, and the maximum number of users of each server is about thousands. As each server manages its own virtual space, the competitive event will not occur.

Quake is a FPS (First-Person Shooter) game, and use "AOI (Area of Interest)" and "dead reckoning". AOI is the circle region whose center is a user's position and represents the range that the user may have interaction with other users by the next turn. A network bandwidth can be saved by transmitting data only to the user with whom AOI overlaps. If the data of the user is delayed or lost when updating a virtual space, the user's present position will not be clear. The user's present position can be estimated by the previous position and the travel direction and speed. This method is called "dead reckoning". The most Net-VEs widely use the method.

Butterfly.net [9] is a Net-VE of the client-server type with multiple servers and uses the grid technology. In Butterfly.net, more than 1 million people can participate virtual space simultaneously. In Butterfly.net a virtual space is divided into multiple regions and each region is assigned to a server. By using "sentinels system" which monitor the boundary of region which each server manages, the boundary between regions cannot be seen from a user, and travel between regions can be performed seamlessly.

As a P2P type example, there are MiMaze [5], and Age of Empires [8]. MiMaze is the network game where a user explores the 3D maze. MiMaze uses "bucket synchronization" technique, which will be described in 2. 2, to provide consistency of virtual space among users. Age of Empires is an RTS (Real Time Strategy) game, and is changing frame rate for each user according to the network speed and the terminal performance. This game also use bucket synchronization technique.

2. 2 Synchronization method for Net-VEs

For Net-VEs to provide real-time nature, the judgment for a competitive event is done in order of reception of a message from users. In this case, if there is a large gap of network delay among users when a competitive event occurs, unfairness will easily happen because the order of transmitting time and the order of receiving time might change.

In bucket synchronization, all users can see the almost same screen by the technique of processing collectively events including the event in a local terminal, and updating a screen display. In MiMaze, a user's position will be predicted by dead reckoning if updates of the user can be received properly. Although those synchronization techniques are effective in a simple Net-VE like MiMaze, they are not suitable for the complicated Net-VEs. In an RTS game like Age of Empires, even if there is delay for about 500ms, a user can play on the game but feels a motion unnatural [3].

There is a technique which delays a user's screen display so that users will not feel the delay. "Simulation retardation" [2] and "temporal contour" [7] are techniques where the travel speed of an object is changed in accordance with the delay according to the distance between a user and an object. The object which goes away from a user is decelerated and the object which approaches to a user is accelerated. Dead reckoning is performed when the data of an object which approaches are not received. "Simulation retardatin" is a technique for Net-VEs like "Pppong" which limit the number of user and user's operation. "Temporal contour" restricts only user's operation. Although those synchronization techniques are effective since action can predict easily the object which a user does not handle, they are almost ineffective when users approach to each other.

For Net-VEs to provide fairness, Net-VEs have to wait for the message from as many users as possible. "Lockstep synchronization" [1] can prevent the injustice by the user by processing the turn, after determine all users' motion, and it can provide fairness completely. Furthermore, adjustment can be provided also with P2P type. However, since Net-VEs are limited with the latest user of speed in advance of a game, real-time nature can not be provided. "Asynchronous synchronization" [1] performs lockstep synchronization only when a user's SOI (Sphere of Influence) is overlapped with other user's SOI. In this synchronization technique, the number of users which waits for messages from other users becomes fewer, and real-time nature will be improved a little. However, this technique is not suitable for Net-VEs with frame rate yet.

To guarantee fairness and real-time nature, there are "time warp synchronization" [6] and "trailing state synchronization" [4]. In time warp synchronization, the snapshot of the state of all objects is saved for every fixed time. When an event is generated or received, a virtual space is updated and an event list is made simultaneously. If an event of earlier generating time is received, the previous event is undone and the event of earlier generating time is adopted. However, in this synchronization technique, as a checkpoint is needed for every message, a huge memory is required. Therefore, this technique is not practical.

In trailing state synchronization, an event is fundamentally processed in order of reception. This synchronization technique determines a synchronization delay at first. If an event of a earlier time stamp is received within this delay, a rollback is performed. As this synchronization technique is for an experimental system called "Mirrored- Server System," it is difficult to adopt this technique to other Net-VEs.

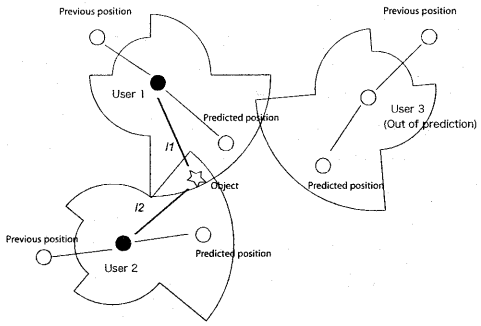

Fig. 1 The region of AOI

3. Synchronization method using prediction

In some Net-VEs of client-server type with multiple servers, to provide real-time nature as well as fairness and consistency of a virtual space, when a competitive event occurs, a server waits a fixed time for events from other users. After the fixed time expires, a server determines an event of the earliest generating time as the representative one and send it to other servers. In this technique, even though there does not exist other users near the object and the probability that the competitive event will occur is very low, a server has to wait fixed time to determine the representative event.

In this paper, in order to provide fairness as well as real-time nature, we propose a new synchronization method that realize fairness of object acquisition by waiting a fixed time for events from users and realize real-time nature of the response to the event of object acquisition by shortening the waiting time using the object acquisition probability calculated from a user's position and travel direction in a virtual space.

3.1 Calculation of object acquisition probability using prediction

A user transmits the current position in a virtual space periodically to a server. Here, "one turn" is defined as the time between each transmission of position data. A server calculates the region of AOI of each user and object by using the current position, the travel speed and the network delay between user and server. AOI is rectified from the prediction position of the next turn. The radius of AOI is not changed in the range between 45 degree of rights and 45 degree of lefts to the travel direction, the one is reduced into 1/2 the range between 45 degree of rights and 90 degree of rights and the range between 45 degree of lefts and 90 degree of lefts, and the one of the rest region is reduced into 1/4. For the static object, the region of AOI is defined as zero. If AOI of a user overlaps with the one of an object, the user seems to be able to acquire an object and a server calculate the probability that the user can acquire the object(Fig. 1).

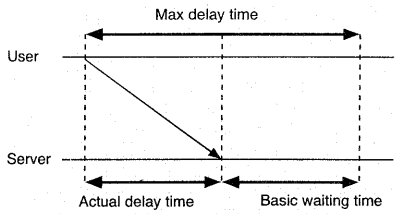

Fig. 2 Calculation of the basic waiting time

Let the distance from the user i's current position to an object be l_i. The probability of object acquisition is calculated based on the ratio of the distance from each user to an object. Supposing the distance from the user i to an object is the half of the one from the user j to the object, the probability of object acquisition for the user i is twice the one of the user j. Therefore, the probability P_i of object acquisition of user i can be represented as Exp. 1. Here, k is the number of users for prediction.

$$P_i = \frac{\frac{1}{l_i}}{\frac{1}{l_1} + \frac{1}{l_2} + ... + \frac{1}{l_k}} \quad (1)$$

3.2 Determine of waiting time

In the proposed method, the waiting time in the server which waits for the event from users is determined using the probability calculated by Exp. 1. To determine the waiting time, at first maximum delay time $Dmax_{C_i}$ is calculated for each user. Here, we suppose that the distribution of the delay time between user and server is the normal distribution. Therefore, the maximum delay time of a user is determined the value of 95% of the distribution of the delay time.

When a server receives the competitive event from a user, the basic waiting time can be calculated by that the actual delay time D_{C_i} between the transmitting time and the receiving time is subtracted from the maximum delay time $Dmax_{C_i}$ (Fig. 2),

When actual delay time is larger than the maximum delay time, the basic waiting time is set to 0. When a server receives the competitive event from the user i, the actual waiting time W_{C_i} in the server which can receives the competitive event from other users connected to the same server is calculated as Exp. 2.

$$W_{C_i} = (max(Dmax_{C_j}) - D_{C_i}) \times (1 - P_i) \quad (2)$$

By Exp. 2, the actual waiting time is shortened according to probability P_i. Since P_i is set to 1, when there is not a user in AOI of an object of the competitive event other than the user who transmits the competitive event, the actual waiting

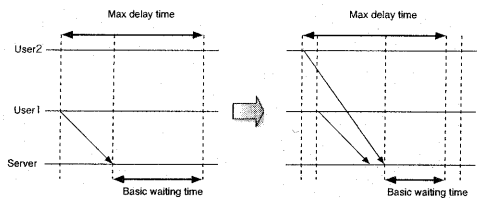

Fig. 3 Recalculation of the actual waiting time

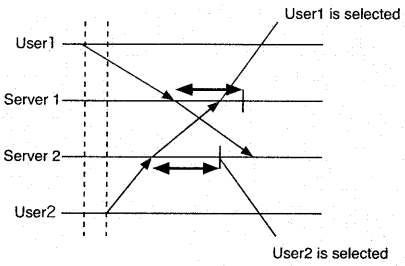

Fig. 4 Example of inconsistency of a virtual space

time is set to 0. When another competitive event with earlier transmitting time is received during the actual waiting time, the basic waiting time will be recalculated (Fig. 3).

3.3 The synchronization between servers

For synchronization between servers, the maximum delay time between severs is used as the waiting time between servers. In this case, the waiting time is not shorten, because inconsistency of a virtual space between servers might occur. However, if there is not a user of other servers in AOI of an object of the competitive event, since it is not necessary to wait, waiting time is set to 0. When the waiting time in a server expires, the server select the candidate of the competitive event whose transmitting time is the earliest in the server and send the event to other servers. The server of the candidate of the earliest transmitting time among candidates can determine the actual update of a virtual space. Then, the server decides the actual update including the competitive event received after the candidate is determined.

However, although the transmitting time of the competitive event from a user is the earliest, the event will be rejected due to the network delay between servers as shown Fig. 4. In such a case, each server has a different result and inconsistency of a virtual space arises.

To avoid such a situation, a server sends the expiration time of the waiting time for other servers as well as the candidate of competitive event. Then, a server which receives the competitive event from a user investigates whether its competitive event can arrive during the waiting time of other servers by using the expiration time of other servers. If the server finds its own candidate of the competitive event can-

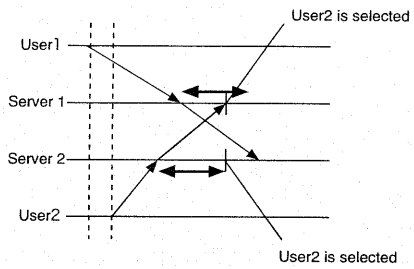

Fig. 5 The way to maintain consistency of a virtual space

not arrive other servers within the expiration time, it rejects the candidate to maintain the consistency of a virtual space (Fig. 5).

4. Evaluation

In this section, simulation experiments were carried out to evaluate the proposed method.

4.1 Simulations

In a simulation, there are two servers each of which has 10 users. The area of a virtual space is 320 × 240 and at the center of the virtual space there is an object. Moreover, users are placed at random in the virtual space. A user moves one unit in the virtual space per movement and can travel in the virtual space five times during one turn. A user sends the current position to a server after one turn expires. Whenever a user reaches the object in the virtual space, he/she transmits the competitive event to the server. The judgment of the competitive event is carried out at the end of a turn.

Movement of a user is determined as follows:

- toward the object: 70%
- right direction from the object: 10–15%
- left direction from the object: 10–15%
- backward direction from the object: 5–10%

For each simulation, the following three types of judgment of the competitive event are used.

(1) the proposed method

(2) using the fixed waiting time both between user-server and between server-server

(3) selecting first receiving event

As network environments, we consider the following two types:

(1) Network over Japan

the average network delay between user-server: 50–80 ms

the average network delay between server-serer: 120ms

(2) Network over the world

the average network delay between user-server: 50–100 ms

the average network delay between server-serer: 250ms

Moreover, ± 10% of fluctuation of average delay time is gen-

erated to both the delay between user-server and the one between server-servers. The maximum delay time is set to 110% of average delay time.

For each network environment, simulations are performed 5,000 times in both cases that the range of AOI is five units and ten units. In simulations, we measured the average time and the maximum time required for the judgment of competitive event. We also measured the probability of unfair result and the probability of inconsistency between servers.

4.2 Results

Table 1 shows the result of simulation of network over Japan in case that the range of AOI is ten units. Table 2 shows the result of simulation of network over Japan in case that the range of AOI is five units. Table 3 shows the result of simulation of network over the world in case that the range of AOI is ten units. Table 4 shows the result of simulation of network over the world in case that the range of AOI is five units.

5. Conclusion

In Net-VEs like an online network game, the competitive event which several users try to acquire the same object simultaneously occurs.

In this paper, we proposed a new synchronization method for Net-VEs of client-server type with multiple servers. The proposed method uses the probability of object acquisition for a user to shorten the waiting time. We consider that the most important thing in Net-VEs is to maintain the consistency of a virtual space among users. Therefore, the proposed method does not provide fairness restrictly. As a future work, we plan to adopt the proposed method to actual Net-VEs that have a lot of servers and users. Moreover, we investigate suitable way of the prediction according to the kind of Net- VEs.

References

[1] N. Baughman and B. Levine. "Cheat-Proof Playout for Centralized and Distributed Online Games," *Proc. of IEEE Infocom 2001*, April 2001.

[2] j. Begole and C.A. Shaffer, "Internet Based Real-Time Multiuser Simulation: Ppong!," *Technical Report*, Virginia Institute of Technology, Department of Computer Science, 1997.

[3] P. Bettner and M. Terrano, "1500 archers on a 28.8: Network programming in Age of Empires and beyond," *Proc. of The 2001 Game Developer Conference*, San Jose, CA, Mar. 2001.

[4] E. Cronin, B. Filstrup, and A. Kurc, "A Distributed Multiplayer Game Server System," Electrical Engineering and Computer Science Department University of Michigan Ann Arbor.

[5] L. Gautier and C. Diot, "Design and Evaluation of MiMaze, a Multiplayer Game on the Internet," *Proc. of International Conference on Multimedia Computing and Systems*, pp.233–236, Austin, TX, USA, 1998.

[6] M. Mauve, "How to Keep a Dead Man from Shooting," *Proc. of the 7th International Workshop on Interactive Distributed Multimedia Systems and Telecommunication Services(IDMS) 2000*, pp.199–204, Enschede, The Netherlands, 2000.

[7] S. Singhal and M. Zyda, *Networked Virtual Environments Design and Implementation*, Addison-Wesley.

[8] "Age of Empires," available at `http://www.microsoft.com/games/empires/`

[9] "Butterfly.net: MMG technology platform for online games," available at `http://www.butterfly.net/`.

[10] "EverQuest.com," available at `http://everquest.station.sony.com/`.

[11] "Quake III: Gold," available at `http://www.idsoftware.com/games/quake/`

Table 1 Japan, AOI = 10

Judgment method		Judgment time(ms)		Unfairness(%)	Inconsistency(%)	# of Competitive event
		Average	Max			
Proposed method	All	119.36	215.22	1.35	1.01	297
	Competitive	190.39	215.22			
	Non-competitive	110.39	215.05			
Fixed waiting time	All	207.53	215.28	0.00	0.00	
	Competitive	201.04	215.28			
	Non-competitive	208.35	215.28			
First receiving	All	195.81	213.28	5.72	2.69	
	Competitive	183.47	210.71			
	Non-competitive	197.37	213.28			

Table 2 Japan, AOI = 5

Judgment method		Judgment time(ms)		Unfairness(%)	Inconsistency(%)	# of Competitive event
		Average	Max			
Proposed method	All	97.45	213.48	4.59	2.53	283
	Competitive	179.07	213.46			
	Non-competitive	87.66	213.48			
Fixed waiting time	All	204.60	197.95	0.00	0.00	
	Competitive	197.95	213.51			
	Non-competitive	194.75	213.67			
First receiving	All	193.29	213.67	6.01	2.83	
	Competitive	181.19	209.09			
	Non-competitive	194.75	213.67			

Table 3 World, AOI = 10

Judgment method		Judgment time(ms)		Unfairness(%)	Inconsistency(%)	# of Competitive event
		Average	Max			
Proposed method	All	178.41	375.35	1.34	0.67	299
	Competitive	329.18	373.35			
	Non-competitive	355.95	373.38			
Fixed waiting time	All	354.92	373.38	0.00	0.00	
	Competitive	346.89	373.38			
	Non-competitive	355.95	373.38			
First receiving	All	339.36	370.75	4.35	3.01	
	Competitive	318.50	373.38			
	Non-competitive	342.01	370.75			

Table 4 World, AOI = 5

Judgment method		Judgment time(ms)		Unfairness(%)	Inconsistency(%)	# of Competitive event
		Average	Max			
Proposed method	All	138.28	379.02	3.97	3.25	277
	Competitive	311.99	376.26			
	Non-competitive	117.9	379.02			
Fixed waiting time	All	316.25	377.73	0.00	0.00	
	Competitive	348.69	376.63			
	Non-competitive	357.14	377.73			
First receiving	All	340.15	376.56	4.69	4.33	
	Competitive	319.65	369.02			
	Non-competitive	342.56	376.56			