Proxy-based Mechanism in Mobile Distributed Virtual Environment

Lin Xu
State key lab. of virtual reality technology and systems, Beihang University, Beijing, 100191, PR China
xulin97@gmail.com

Xiaohui Liang
State key lab. of virtual reality technology and systems, Beihang University, Beijing, 100191, PR China
lxh@vrlab.buaa.edu.cn

Ke Xie
State key lab. of virtual reality technology and systems, Beihang University, Beijing, 100191, PR China
xieke@vrlab.buaa.edu.cn

Abstract

With the recent increases in wireless network bandwidth and mobile device performance, researchers pay more attention to construct mobile distributed virtual environment (MDVE) that can support users to participate in shared virtual environment (VE) by mobile device. However, MDVE faces many challenges due to the limitations of mobile devices. The short life of batteries may cause unexpected power interruption that may give rise to irreparable loss; the limitation of processing capacity and bandwidth will impact on interaction among avatars in real time. In this paper, we present a proxy-based mechanism that adds a proxy module in server, including proxy-based state dissemination and avatar migration. Our experiments which show the efficiency of proposed mechanism successfully are given at last.

1. Introduction

Distributed virtual environment (DVE) allows multiple users (locating around the world and working on different computers that are interconnected through different networks) to interact in a shared virtual environment (VE) [1]. Each user is represented in the shared VE by an entity called avatar, whose state is controlled by the user input, and propagated to other avatars in the shared VE by networks. With the recent increases in wireless network bandwidth and mobile device performance, researchers pay more attention to construct mobile distributed virtual environment (MDVE) that can support users participate in the shared VE by mobile device. MDVE has been applied to various fields, such as wireless multiplayer online games, tour navigation applications, and augmented reality systems.

With the limitations of mobile devices, MDVE faces many challenges [2]. For example, the short life of batteries may cause unexpected power interruption that may give rise to irreparable loss; the limitation of processing capacity and bandwidth will impact on interaction among avatars in real time. Especially, while avatar is migrating from one server to another, users suffer from perceiving abrupt scene change [3, 4], which influences the normal processing of application. In order to make mobile devices better to participate in MDVE, we must extend the life of batteries and reduce the processing of data packets, particularly in avatar migration.

For the issues above, we present a proxy-based mechanism that adds a proxy module in server, including proxy-based state dissemination and avatar migration. Proxy-based state dissemination uses proxy to reduce the sending and filter the receiving of state packets. Proxy-based avatar migration uses proxy to assist the completion of avatar migration. As avatar migrates directly between the servers through high-speed network, mobile devices only process a few data, so the seamless avatar migration can be ensured.

The rest of the paper is organized as follows. Section 2 presents related work about power-aware state dissemination and avatar migration in fixed networks. Section 3 describes the system model of our MDVE system. Section 4 details proxy-based mechanism for mobile devices in MDVE, including proxy-based state dissemination and avatar migration. Section 5 presents and discusses the results of the experiments. Finally, Section 6 concludes our paper with a discussion on possible future work.

2. Related work

In this section, we introduce the recent research of power consumption in mobile devices, the power-aware protocol in MDVE system and existing migration techniques for DVE.

Due to the limited life of mobile devices batteries and the irreparable loss caused by the sudden power interruption, we must minimize the power consumption of simulation application for mobile devices. Recent researches [5, 6, 7, 8] show that wireless network interface cards (WNIC) consume a significant portion of the total power needed by a communicating mobile device, even as high as 50% [2].

A typical WNIC supports multiple modes of operation, with varying levels of power consumption. The common modes include the following power consumption in descending order: Transmit, Receive, Idle, Suspended and Off modes [2, 5].

According to the results of above researches, Shi et al [2] proposed a new power-aware dead reckoning framework for power-efficient state dissemination. By switching power mode between active and suspended mode, the power of mobile devices can be saved. However, suspending the WNIC can result in increased message latency and a higher probability of lost packets, and the suspension time is decided according to the global minimum among all the mobile devices' suspension periods. When the number of mobile devices increases, how to obtain the minimal suspension period becomes relatively difficult.

When avatars simulate in multi-server architecture (see Section 3) DVE systems, the systems must provide an avatar migration mechanism to achieve moving from one region managed by a server to another region managed by a different server. Recent studies [3, 4, 9] have proposed some avatar migration mechanisms for non-mobile devices. They set buffer zone around the boundary of region. When avatar moves into the buffer zone, it will connect to more than one server, and receive data from those servers. In this case, the avatar will receive more data than usual, while it is in buffer zone, however, mobile devices cannot afford this task. Furthermore, because the migration processing needs the participation of avatars and mobile devices are restricted by the limited bandwidth and processing capacity of mobile devices, the migration becomes relatively tardiness and users experience decline.

3. System model

Our MDVE system adopts multi-server architecture. Multi-server architecture is becoming a defacto standard for DVE systems. In this architecture, the shared VE shall be partitioned into regions, and each of which is assigned to a separate server called region server, distributing the workload among them. Systems of this approach include RING [10], NetEffect [11], ATLAS [12, 13], and Citation [14]. The architecture of our MDVE system is illustrated in Figure 1.

In this architecture, each avatar issues messages to the region server for further propagation to other avatars and region servers in the same VE. The advantage of this architecture is that the region server may perform message filtering to minimize the amount of messages needed to be handled by each client and to be propagated through the network. In this paper, we take advantage of this architecture to implement data filter to reduce the amount of data processing (see Section 4.1.2). This is the reason why we adopt multi-server architecture.

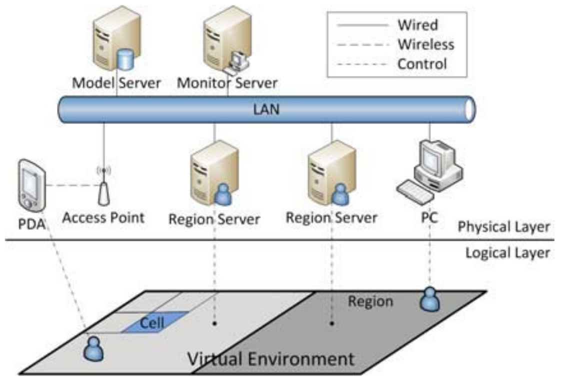

Figure 1: The architecture of our MDVE system

3.1. Logical layer side

The whole VE is partitioned into a set of N_R regions denoted as R_i as shown in Figure 1.

$$VE = \{R_1, R_2, \cdots, R_{N_R}\} \tag{1}$$

The region is regularly subdivided into a set of N_C rectangular cells denoted as $C_{i,j}$ as shown in Figure 1.

$$R_i = \{C_{i,1}, C_{i,2}, \cdots, C_{i,N_C}\} \tag{2}$$

We assume that all regions are with the same size, and they have the same number of cells. We represent a partition that each server manages as a single square as shown in Figure 1.

3.2. Physical layer side

Monitor Server: this server is responsible for region server and avatar login and logout. While the system is running, it monitors the state of the region server and load balancing between the region servers. Monitor server also provides wireless network signal testing to help mobile devices with better interaction.

Region Server: this server manages a region in VE. When an avatar in the region, the information of the avatar will be filtered and forwarded by the server which manages this region. When the region server receives messages from avatars, it determines if avatars reach the boundary of the region, and implements avatar migration mechanism to achieve seamless migration. In order to support mobile devices, we add MDProxy (see Section 4) in the region server to manage mobile devices in system.

Model Server: this server provides service for mobile device. It will deploy a few models in mobile devices before the system runs, because of the limited memory capacity in mobile devices. While the system is running, mobile devices request model server to obtain model according to the requirement.

4. Proxy-based mechanism

Proxy-based mechanism for mobile devices is composed of two parts: state dissemination and avatar migration. The tasks of the proxy are assigned to the region server, and come into being an independent module called MDProxy (see Figure 2).

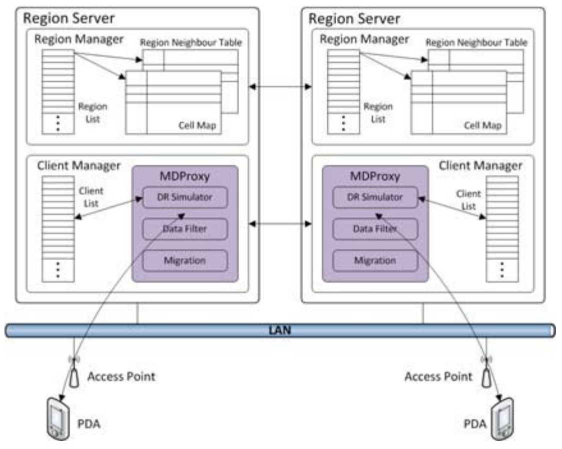

Figure 2: The architecture of proxy-based mechanism

The MDProxy has three components: DR (Dead Reckoning) simulator, data filter and migration. DR simulator and data filter serve state dissemination. The former is used for reducing the amount of state protocol data units (PDUs) those mobile devices send; the latter is used for reducing the amount of state PDUs those mobile devices receive. Migration serves avatar migration, and it will obtain help from data filter to reduce the amount of state PDUs.

4.1. Proxy-based state dissemination

The process of traditional state dissemination is introduced below: an avatar continuously generates state PDUs by users' input and issues them to the region server which the avatar belongs to, then the region server propagates these state PDUs to other region servers and avatars, while the avatar receives state PDUs of others from the region server. However, mobile devices are not suitable for continuing sending and receiving state PDUs because of the limitations of themselves. Based on the short life of batteries and the limitation of processing capacity of mobile devices, we hope to assign some of the tasks to other devices, such as the region server. The DR simulator and data filter of the MDProxy is in the region server and complete some works instead of mobile devices. DR simulator proposes a new state PDUs dissemination approach for avatar simulations that can timely determine to issue a state PDUs under given consistency constraint. Data Filter provides a new filtering approach for mobile devices that can dynamically determine a set of avatars' state PDUs to be sent under given filtering criteria.

A. DR simulator

A method of position and orientation estimation called dead reckoning shall be employed to limit the rate at which avatar state PDUs are issued. By estimating the position and orientation of other avatars, it is not necessary to receive a report about every change in position and orientation that occurs in the avatar's trajectory over time. Only when a change in position and orientation differs by a prespecified amount (threshold) from the dead reckoned position and orientation is a new update required [15].

As is mentioned above, dead reckoning can reduce avatar state PDUs, so we design DR simulator to undertake this work. DR simulator shall maintain a dead reckoned model of every avatar of mobile device. The designated dead reckoning formula [15] is used to estimate the position and orientation of avatar which belongs to mobile device in the region server. The dead reckoning formula is chosen as follow. You can choose other dead reckoning formula instead of following.

The position in world coordinates after time delta is:

$$P = P_0 + V_0\Delta t + \frac{1}{2}A\Delta t^2 \tag{3}$$

Where P_0 is position vector in world coordinates at initial time, V_0 is velocity vector in world coordinates at initial time, A is acceleration vector, and Δt is time increment for dead reckoning step.

The orientation matrix that takes world coordinates into body coordinates at time $t_0 + \Delta t$ is:

$$[R]_{w\to b} = [DR][R_0]_{w\to b} \tag{4}$$

Where $[DR]$ is dead reckoning matrix, and $[R_0]_{w\to b}$ is avatar's initial world to body orientation matrix.

With the estimation of DR simulator, the region server will generate state PDUs and issues them to other region servers and avatars. This processing does not require the participation of mobile devices. When the region server receives a new update from one of mobile devices it is dead reckoning, it shall correct its dead reckoning model and base its estimation on the most recent position, velocity, and acceleration information.

Each simulation application in mobile device shall maintain an internal model of itself (representing its actual position and orientation) and a dead reckoned model of itself. Thresholds for position and orientation shall be established as criteria for determining if the avatar's actual position and orientation has varied from the dead reckoned model. When the avatar's actual position and orientation have varied from the dead reckoned position and orientation by more than the threshold value denoted as δ, the avatar shall issue an avatar state PDU to communicate to DR simulator the actual position and orientation to correct the dead reckoned model of itself (see Figure 3).

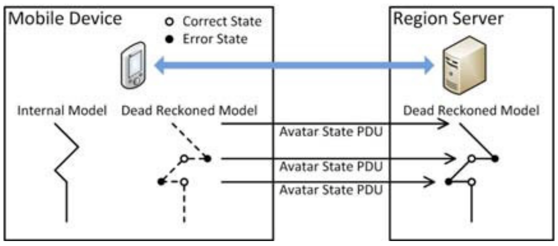

Figure 3: Example of DR simulator

By the collaboration of mobile devices and region servers, the whole tracking in figure 3 only needs to issue three avatar state PDUs, greatly reducing the time of using WNIC in mobile devices. Of course, our approach also exists some limitations. From Figure 3, we can clearly see that two tracks are different between each other. This is the inherent problems of dead reckoning, but we can reduce such differences by decreasing the threshold δ. Compared to the disastrous consequences due to unexpected power interruption, this deficiency of dead reckoning in real-time simulation applications supported mobile devices can still be accepted. Furthermore, the error can be decreased by replacing with better dead reckoning approaches or reducing the dead reckoning threshold δ. We implement a fundamental tradeoff between state consistency and power consumption.

B. Data filter

With the limited processing capacity, mobile devices are unable to deal with large amounts of data per second. In DVE systems, every avatar will receive large amounts of state PDUs form other avatars. For example, there are 100 avatars in VE, and as we know ensuring real time ability must provide 25 frames per second, which means needing to receive 25 state PDUs from every avatar in VE, well then, mobile device will receive 2500 state PDUs per second. With the expanding the scale of DVE system, the data received per second in mobile device will also increase. In this case, the view of mobile device will be significantly different from the others, called the consistency problem.

Due to reasons mentioned above, we need to filter data for mobile device. Because mobile device only receive state PDUs from region server, the task of filtering data can assign to the region server, and is the data filter module in MDProxy.

To minimize the amount of data, most existing methods consider only the area of interest (AOI) of an avatar [16, 17, 18]. We adopt this method to filter data. If an avatar falls inside the AOI of another, the avatar is considered visible to that one. Otherwise, the avatar is considered too far to be visible. In the system, we define an AOI of avatar as a circular region. Therefore, each AOI is characterized by a radius.

In order to reduce the number of the interaction between the WNIC and the simulation application, we divide one second into 25 times, each time the region server will send a combination state PDU to mobile device, and the combination state PDU includes all avatars' state PDUs which mobile device needs in that time. Whose state PDU can be put into the combination state PDU is determined by data filter in MDProxy.

4.2. Proxy-based avatar migration

In our system, an avatar in mobile device uses proxy-based state dissemination, so when the avatar in mobile device locomotes to an adjacent region that is managed by another region server, it will make connections and disconnections, furthermore, the DR simulator and data filter in region server relating to an avatar in mobile device will also migrate from one to another region server. However, some avatar migration mechanisms for non-mobile devices are unsuitable for mobile devices due to their limitations. In this case, our system provides an effective avatar migration mechanism for mobile devices. We refer it as proxy-based avatar migration.

As it is the same with avatar migration mechanisms for non-mobile devices, we define two types of cells called general cell and boundary cell (see Figure 4). While an avatar is in boundary cell, this situation signifies the possibility that it will migrate into an adjacent region. In order to provide a seamless migration, the adjacent region server needs to know the avatar's state information when it is in boundary cell. Hence, when the region server is aware of the event that an avatar enters the boundary cell, it will notify its adjacent region server, a vicarious dead reckoned model of that avatar is then created by the adjacent region server.

We classify the process of avatar migration in three kinds of cases (see Figure 4), and avatar A_1, A_2, A_3 all run on mobile devices. Case one just like avatar A_1 shown in Figure 4. When avatar A_1 is at position P_0, it just has one dead reckoned model in RS_1, as mentioned in section 4.1.1. When avatar A_1 gradually moves into the boundary cell which is also managed by RS_1, for instance, position P_1, a vicarious dead reckoned model will be created in RS_2, but only the dead reckoned model in RS_1 instead of avatar A_1 issue the state PDUs to other avatars in VE, and the vicarious dead reckoned model has to preserve a state for avatar A_1 only. The vicarious dead reckoned model in RS_2 becomes a dead reckoned model to simulate the track for avatar A_1 when avatar A_1 gets across the boundary line, for example, position P_2, and the dead reckoned model in RS_1 will become the vicarious dead reckoned model to preserve a state for avatar A_1. In other words, DR simulator for avatar A_1 in RS_1 and RS_2 will exchange the roles when avatar A_1 gets cross the boundary line. When avatar A_1 moves from position P_2 to position P_3, it will delete the vicarious dead reckoned model in RS_1. For case two, the track of avatar A_2 is different from avatar A_1. Since it does not cross the boundary line, and get away from the boundary cell and back to the general cell managed by RS_1, such as moving from position P_2 to position P_3 in the track of avatar A_2. When avatar A_2 goes back to region R_1, the vicarious dead reckoned model in RS_2 will be deleted. In case three, the role of the dead reckoned model and the vicarious dead reckoned model of avatar A_3 will be exchanged twice.

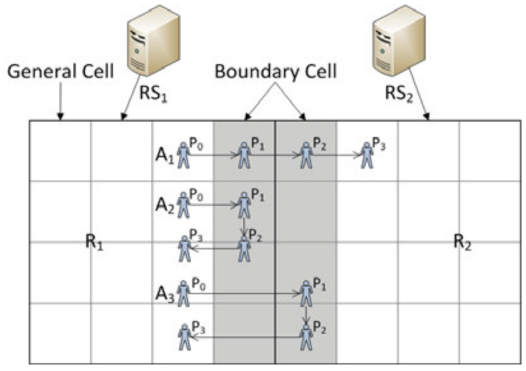

Figure 4: The avatar migration mechanism

Because of the track of internal model in mobile and the dead reckoned model are not consistent, when the dead reckoned model moves from the general cell to the boundary cell, it will ask mobile device for state synchronization to confirm whether the avatar is in the boundary cell, and vice versa.

Taking avatar A_1 in Figure 4 as an example, we will introduce the interaction between the region server and the mobile device (see Figure 5).

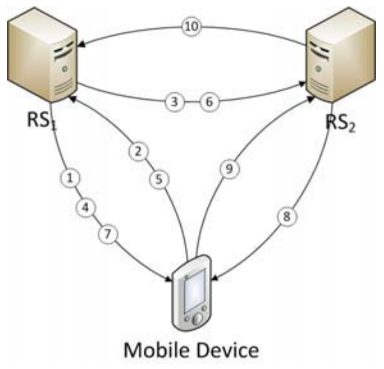

Figure 5: Step involved when avatar migrates

Procedure:

Step 1. RS_1 asks mobile device for state synchronization when the dead reckoned model moves into the boundary cell.

Step 2. Mobile device returns a state PDUs to RS_1.

Step 3. If the state is correct, then RS_1 will inform RS_2 to create a vicarious dead reckoned model.

Step 4. When the dead reckoned model gets across the boundary line, RS_1 will ask mobile device for state synchronization.

Step 5. Mobile device returns a state PDUs to RS_1.

Step 6. If the state is correct, then RS_1 will notify RS_2 to dead reckoning for mobile device and stop itself.

Step 7. RS_1 will notify mobile device that it shall interact with RS_2 at next time.

Step 8. When the dead reckoned model moves from the boundary cell to the general cell managed by RS_2, RS_2 will ask mobile device for state synchronization.

Step 9. Mobile device returns a state PDUs to RS_2.

Step 10. RS_2 inform RS_1 to delete the vicarious dead reckoned model.

When avatar migrates between the region servers, we will also filter data for mobile device. With the help of the dead reckoned model and the vicarious dead reckoned model, we can model the interest management that supports the avatar migration as given in [3]. Let A_i represent the dead reckoned model of an avatar with index i on the region server M, which is controlled by client C_i, and $A_i^{'}$ is its vicarious dead reckoned model on the adjacent server N. Then, the interest management for A_i can be defined as follow:

$$AOI(A_i) = \{C_k \mid \forall A_k \propto AOI(A_i) \vee \forall A_k^{'} \propto AOI(A_i)\} \quad (5)$$

$$A_k \propto M \text{ , } A_k^{'} \propto M$$

In addition, the interest management for the vicarious dead reckoned model $A_i^{'}$ can be defined as follow:

$$AOI(A_i^{'}) = \{C_k \mid \forall A_k \propto AOI(A_i^{'})\} \qquad (6)$$

$A_k \propto N$, A_k is not located in the boundary cell

These two formulas together provide a mechanism to perform interest management with spatial AOI technique for avatar migration. The detail mathematical proof of the correctness and completeness of two formulas is given in [19].

5. Results and discussions

We have developed a simulation model and conducted a series of experimental studies to quantify and evaluate the performance of the proxy-based mechanism. In our simulation model, the complete VE is partitioned into N_R regions, and the region is regularly subdivided into 10×10 cells. Since each region is managed by one region server, N_R region servers will be required. Both the region server and the monitor server programs run on a PC with a 2.2 GHz CPU and 2 GB RAM. The client program runs on a HP iPAQ 212 PDA. Both PDA and access point (AP) are based on the IEEE 802.11g standard.

5.1. Experiments for state dissemination

In this section, we study the validity of proxy-based state dissemination mechanism. Proxy-based state dissemination mechanism includes DR simulator and data filter, applied to reduce the amount of sending and receiving data respectively.

In the first experiment, we make the avatar move along a circular track, and set the dead reckoning threshold δ (see Section 4) to 2 pixels. The radius of an AOI of avatar is set to 20 pixels, and there are 1000 even distribution of avatars in every region server. Table 1 compares the amount of interactive data between the use and the non-use of proxy-based state dissemination mechanism.

Table 1: Compare the amount of interactive data between the use and the non-use of proxy-based state dissemination mechanism

Use/Non-use	Send (PDU/100s)	Receive (PDU/s)
Use	27	25
Non-use	2589	526

From table 1, we could detect that proxy-based state dissemination mechanism can decrease the amount of interactive data, and then can save the power of mobile devices. The emerging problem of consistency still exists with the reducing of the amount, however, you can replace with better dead reckoning approaches or reducing the dead reckoning threshold δ to reduce the influence of the problem.

In the second experiment, we set the dead reckoning threshold δ to 2, 4, 6 and 8 pixels. Figure 6 shows the result of the experiment.

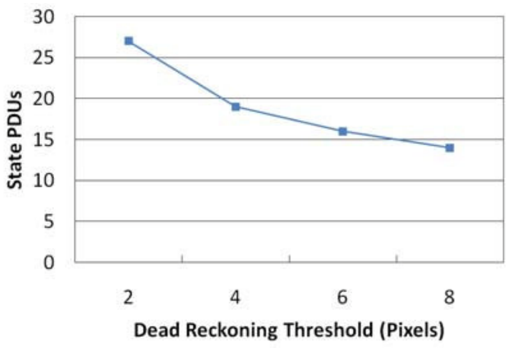

Figure 6: Relationship between the threshold and the amount of sending data

As we know, the increasing of the threshold will cause the decreasing of sending data. From figure 6, the rule also exists in our system. We implement a fundamental tradeoff between state consistency and power consumption.

5.2. Experiments for avatar migration

In this section, we study the performance of proxy-based avatar migration mechanism. In this experiment, the whole VE is divided into two regions corresponding two region servers, and the region is regularly subdivided into 10×10 cells. We join a certain number of avatars to the VE, 10 avatars in every cell. Then we survey the processing time when an avatar in a mobile device migrates from region server 1 to region server 2 (see Figure 7).

When the avatar of a mobile device moves into the boundary cell managed by region server 1 at time t_1, the region server 1 asks the mobile device for state synchronization and informs RS_2 to create a vicarious dead reckoned model (see Figure 5). These tasks will take some processing time, so we can see that the processing time rises. Region server 2 will maintain the new status of the vicarious dead reckoned model and assist the avatar to compute AOI, therefore they will also take processing time as the red line we can see from time t_1 in figure 7. At time t_2, the avatar gets across the boundary line, and migrates from region server 1 to region server 2 synchronously, so the processing time will exchange between two region servers. When the avatar moves from the boundary cell to the general cell managed by region server 2, figure 7 shows the exchange at time t_3. From figure 7, we can see the processing time of proxy-based avatar migration mechanism is acceptable and well-balanced.

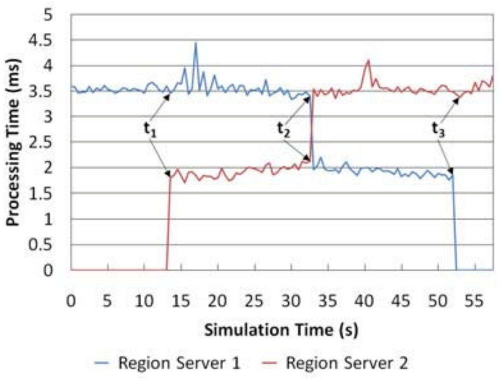

Figure 7: The processing time of avatar migration

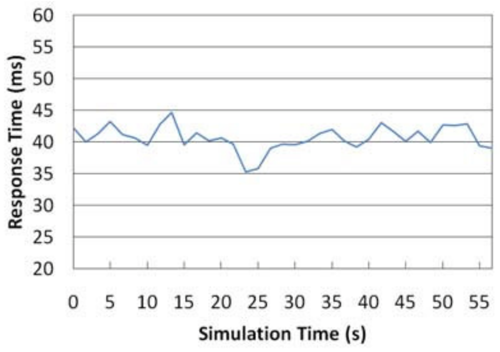

Figure 8: The response time of region server in our avatar migration mechanism

When an avatar migrates from one to another region server, the related region server will generate extra processing mission. So we want to survey the response time of region server in our avatar migration mechanism is satisfactory or not.

While avatars migrate from one to another region server, we can see from figure 8 the response time keeps between 35 ms and 45 ms. Compared to the response time of normal situation, it does not increase significantly in the process of avatar migration, so the interaction of avatar will not be affected generally.

6. Conclusions

In this paper, we present a proxy-based mechanism that adds a proxy module in region servers, including proxy-based state dissemination and avatar migration. The former proposes a new state PDUs dissemination approach and a new filtering approach for mobile devices that can dynamically determine a set of avatars' state PDUs to be sent under given filtering criteria. Proxy-based avatar migration uses proxy to assist the completion of avatar migration. Avatars can migrate directly between the region servers through high-speed network, so the seamless avatar migration can be ensured.

We have developed a simulation model and conducted a series of experimental studies to quantify and evaluate the performance of the proxy-based mechanism. Through experiments, proxy-based mechanism could tradeoff between state consistency and power consumption, and could ensure the seamless avatar migration for mobile devices.

7. Acknowledgments

This research is supported by National High Technology Project (863 project), No. 2006AA01Z333, Graduate Innovative Practice Fund Project of Beihang University.

References

[1] Sandeep Singhal, Michael Zyda. Networked Virtual Environments: Design and Implementation. New York: ACM Press Siggraph Series and Addison-Wesley, 1999.

[2] Weidong Shi, Kalyan Perumalla, Richard Fujimoto. Power-aware state dissemination in mobile distributed virtual environments. In Proceedings of the Seventeenth Workshop on Parallel and Distributed Simulation (PADS' 03), pages 181-188, 2003.

[3] Jiung-yao Huang, Yi-chang Du, Chien-Min Wang. Design of the Server Cluster to Support Avatar Migration. In Proceedings of IEEE Virtual Reality, pages 7-14, March 2003.

[4] Qingshu Yuan, Dongming Lu. A Latency-adaptive Communication Architecture for Inter-networked Virtual Environments. In IEEE International Conference on Systems, Man and Cybernetics, pages 6296-6301, 2004.

[5] L.Feeney and M.Nilsson. Investigating the Energy Consumption of a Wireless Network Interface in an Ad Hoc Networking Environment. In Proc. IEEE INFOCOM, anchorage, AK, April 2001.

[6] T.simunic, l.Benini, P.W. Glynn, and G.D. Micheli. Dynamic Power Management for Portable Systems. In Proc. ACM MOBICOM, pages 11-19, Boston, MA, 2000.

[7] J.-P. Ebert, B, Burns, and A. Wolisz. A Trace-based Approach for Determining the Egergy Consumption of a

WLAN Network Interface. In Proceedings of European Wireless, pages 230-236, Feb, 2002.

[8] R. Kravets and P. Krishnan. Power Management Techniques for Mobile Communication. in Porc. of 4th Annual international Conference on Mobile Computing and Networking (MOBICOM 98), 1998.

[9] Beob Kyun Kim, Dong Ki Won, Dong Un An, Seung Jong Chung. Reducing Migration Latency for Distributed Virtual Environments by Mediation. In International Symposium on Information Technology Convergence, pages 396-400, 2007.

[10] T. Funkhouser. RING: a Client-server System for Multi-user Virtual Environments. In Proc. Symp. Interactive 3D Graphics, Monterey, CA, Apr. 1995.

[11] T. Das and G. Singh et al. NetEffect: a Network Architecture for Large-scale Multi-user Virtual World. In Proc. ACMVRST, pages 157-163, 1997.

[12] D. Lee, M. Lim, S. Han. ATLAS: A Scalable Network Framework for Distributed Virtual Environments. Proc. ACM Collaborative Virtual Environments Conf (CVE 2002), pages 47-54, 2002.

[13] S. Han, M. Lim. ATLAS-II: Scalable and Self-tunable Network Framework for Networked Virtual Environments. In Proc. The Second Young Investigator's Forum on Virtual Reality (YVR'03), 2003.

[14] M. Hori, T. Iseri, and K. Fujikawa et al. Scalability Issues of Dynamic Space Management for Multiple-server Networked Virtual Environments. In Proc. IEEE Pacific Rim Conf. Communications, Computers and Signal Processing, pages 200-203, 2001.

[15] IEEE Standard for Distributed Interactive Simulation-Application Protocols. IEEE Std 1278.1-1995, 1995.

[16] J. Falby, M. Zyda, D. Pratt, R. Mackey. NPSNET: Hierarchical Data Structures for Real-time Three-dimensional Visual Simulation. Comput. Graph., vol. 17, no. 1, pages 65-69, 1993.

[17] M. Macedonia, M. Zyda, D. Pratt, P. Brutzman, P. Barham. Exploiting Reality with Multicast Groups: a Network Architecture for Large-scale Virtual Environments. In Proc. IEEE VRAIS, pages 2-10, March 1995.

[18] D. Schmalstieg and M. Gervautz. Demand-driven Geometry Transmission for Distributed Virtual Environments. In Proc. Eurographics '96, pages 421-432, 1996.

[19] J.Y. Huang, Y.C. Du, L. Hui. The Avatar Migration Mechanism for the Large Scale of Networked Virtual Environment. Submitted to ACM TOMACS, July 2002.