Collaborative Virtual City Design in Heterogeneous Computer Environment

Stefanus S. Rahadi (210103)

School of Information Science,
Japan Advanced Institute of Science and Technology

February 13, 2004

Keywords: virtual reality, city design, heterogeneous computer environment, networked virtual environment, collaboration.

1 Introduction

City design is a task of arranging an area with artistic and functional features as its primary consideration. This task is one of the design tasks that requires realistic visualization for evaluation. Realistic visualization allows accurate design evaluation in city design task. Therefore, city design task may exploit virtual reality in order to generate realistic visualization.

Practitioners of city design have had fast and realistic visualization for accurate design evaluation. Unfortunately, they are demanding more than just realistic visualization. As the nature of city design task, which is a multi-person task, city designers demand collaboration in virtual reality. Collaboration will allow city designers to work together on the same virtual environment at the same time and collaborate with each other.

Supporting collaboration in virtual reality becomes a challenge, where the system is expected to deliver real-time visualization and real-time virtual environment interaction at the same time. Providing such real-time components is the hardest part of creating a system that supports collaboration. This problem becomes severer if the collaboration is done on heterogeneous computer environment.

Heterogeneous computer environment is an environment consists of several different computers with different specifications, platforms, and even different devices. These machines have their own architecture and interaction paradigm. Supporting heterogeneous computer environment will give more freedom in collaborating with virtual environment, where participants can select their own preferred machine for collaborating with others. Although supporting heterogeneous computer environment has such advantages, heterogeneity of the system itself makes it very difficult to perform collaboration in such environment.

Recognizing these demands for collaborative virtual city design in heterogeneous computer environment, this research proposed a system to support collaborative city design

task by providing real-time realistic 3D virtual environment and interaction for heterogeneous computer environment. The proposed system will allow not only system with different platform to collaborate, but also system with different devices to collaborate together.

2 Proposed Method

The proposed system adopts networked virtual environment for collaboration. Such virtual environment allows users in geographically different places to work together in collaboration. The system is configured as client-server to enable collaboration in networked virtual environment. The virtual environment will be stored on server side while the clients manage their own replicated scene-graph. The server is responsible for managing the collaboration and updating the replicated scene-graph resides on each client.

To enable various computer to work together in collaboration, a controller that can handle various interaction from various different computer is required. Therefore, client-server is the best for the proposed system. Using such configuration allows the user to handle all interaction from the client and control the virtual environment. In this case, if a client wants to modify the virtual environment, the client must not directly modifies it, instead the client must send the request to the server. The server will process this request and update the virtual environment along with all replicated virtual environment on each client. Thus the server will have full control of virtual environment interaction.

To realize such collaboration, we proposed remote event for updating virtual environment over the network. Remote event can be seen as a message sent from one machine to other machine to activate some operations on the designated machine. The server uses this remote event for updating all the replicated virtual environments on all clients. Upon receiving a remote event, the client will act and or update its replicated virtual environment according to the remote event.

Remote event is generated from event that actually generated in client side. Not all events created on client side will be sent to the server and become candidate for remote event. In order to lower server's processing load, client must send only events that will directly modify the virtual environment. The other events must be processed locally. Such behavior will allow interaction from various different type of clients, even for client with different devices.

Client-server configuration solves problems on real-time collaboration in heterogeneous computer environment. As a matter of fact, as the increasing number of collaboration participants, server load will also increase. This will produce huge latency in collaboration, preventing the system from achieving real-time virtual environment interaction. To solve this problem, a method was introduced to cut system latency by filtering unnecessary events in server side. We call this method as event filtering.

Event filtering can be realized by giving time-stamps and priorities to all events. Time-stamp will prevent the server from processing event which is too old to be processed, while priorities will classify which event can be filtered and which cannot be filtered. Error level that an event will produce if the event is lost in processing or transmission become the category in event classification. In this research, event can be divided into 3 types, critical, normal, and ignorable. Critical event will produce chaos if the event is

lost. Normal event will produce a temporary error. When other event with the same type received, the system will be recovered. Ignorable event will not produce any error if the event is lost. Losing an ignorable event will only produce update skips that barely visible to users and can be ignored.

Event filtering were done by using rules based on this time-stamp and classification. In this research, the rules are defined as follows.

1. Filter only ignorable event.
2. Event older than time limit will be discarded. In this research, the limit is 1 second.
3. If necessary, filter the event using probability. This research uses 80% for event acceptance.

3 Experiment Results

From the experiment, system response time was measured. The result shows that remote event method gave significant performance boost compared to collaboration using scene dump method. Remote event method is proven to have up to 45 times faster response time than scene dump method. This speed up comes from the remote event that have very fast processing time and network transmission time compared to the scene dump method.

In this research, the system response time is divided into 3 parts; network time, queue time, and processing time. Network time is the amount of time for transmitting event via network from client to server and server to client. Queue time is the overall time an event spend in queues before it being processed (in parent process' queue) and before it being transmitted (in client side and child process' queue). Processing time is the time for server to process the event.

The results show that the remote event method had stable response time over the increasing number of objects. While the system had slower response time over the increasing number of participating clients. This is obviously true, because more clients will result in higher load to the network and also for the server. In this case, the response time will have longer network time, longer queue time, and longer processing time.

From all the results obtained, processing time and queue time is quite small compared to the network time. From these result, it can be seen that the network still become the biggest bottle neck in networked virtual environment. The network will be saturated first before the server load become the bottle neck in processing interaction for collaboration.

For observing scalability of the proposed system, an experiment that generates a very huge load for the server has been done. In this experiment, a very huge load that is almost impossible in normal usage is used. Thus the server behavior in handling huge load can be examined. From this experiment, system latency or delay was obtained.

Giving huge load to the server will increase the system latency. Huge system latency will prevent the server from achieving real-time collaboration. From the results, giving more than 4000 events per second had create over $100ms$ of latency, which is above the threshold. It resulted in very poor virtual environment update rate that lead to very poor collaboration.

To deal with such load, event filtering were used. Using event filtering significantly reduces the latency. Server with event filtering only produce 50% of latency compared to server without event filtering. Thus, server with event filtering is more scalable than the one without event filtering.

4 Summary

This research proposed the system for supporting city design using collaborative virtual environment. Along with providing visualization using virtual environment, heterogeneous computer environment was also supported. The system allows users to choose their own preferred collaboration client and provides the user with various degree of reality by adopting heterogeneous devices in collaboration.

For collaboration, the remote event method has been introduced. This method has acchieved very good system response in delivering real-time visualization and real-time collaboration. However, due to the client-server configuration of the system, the proposed system produced huge latency when a huge load came to the server.

To deal with such problem, the event filtering was introduced in order to cut the system latency. The event filtering has been proven to have very good performance in lowering system latency. This has resulted the proposed system can still deliver real-time visualization and collaboration even if the server has a huge load in it.