# A P2P-based Three-Dimensional Virtual Environment Management and Collaborative Streaming System

Chuan-Feng Chiu, Steen J. Hsu, Sen-Ren Jan
Department of Information Management, Minghsin University of Science and Technology, Hsinchu, Taiwan
Email: cfchiu@must.edu.tw

***Abstract*—Because of the growth of network and device capability, to realize interactive 3D virtual environment over Internet have become a popular and challenging research area. In recent development on practical 3D virtual environment, most of them focus on client-server architecture that deploys a central server or proxy to process 3D content rendering, storage, delivery and management. Therefore, the heavy computation and processing will be the bottleneck of the central server so that the overall performance might be worse. On the other hand, 3D scenes always have large volume so that 3D scenes which are transmitted from server may gain worse transmission performance and user's device will not have enough storage space as well as server machine. Therefore, in this paper we propose a collaborative 3D scene management and streaming system to reduce the load of server-based architecture, and we use peer-to-peer technology to realize distributed virtual environment.**

***Index Terms*—3D Streaming, Peer-to-Peer network, Collaborative Streaming, LOD, Virtual Environment**

## I. INTRODUCTION

Because of the growth of network and device capability, to realize interactive 3D virtual environment over Internet have become a popular and challenging research area. But in order to establish interactive 3D virtual environment over Internet, there still exist some issues that need to be solved. First, in recent development on 3D virtual environment, most of them focus on client-server architecture that deploys a central server to process 3D content rendering, storage, delivery and management. The heavy computation and processing will be the bottleneck on the central server so that the overall performance might be worse. Second, 3D scenes always have large volume so that transmitting large volume of 3D scenes data from the server may gain worse transmission performance. Therefore, the present work in this paper is to propose a real-time collaborative 3D scenes management and transmission mechanism over peer-to-peer overlay network. The proposed management mechanism is based on CAN-based peer-to-peer network and the collaborative 3D scene transmission mechanism is based on multi layer segmentation, multi-flow streaming and mixing receiving data. On the other hand, in order to achieving the goal of multi-flows streaming, we will use the LOD(Level of Detail)[24] to encode each object in 3D scene.

The rest of this paper is organized as followings. Section 2 will reveal literatures related with our work. P2P-based virtual environment management will be described in Section 3. A collaborative 3D streaming mechanism over peer-to-peer overlay network is proposed in Section 4. The design of distributed 3D scene management and streaming system based on the proposed mechanism is described in Section 5. Finally, we give the conclusion of our work in Section 6.

## II. RELATED WORK

Before presenting our work, we describe the related literatures regarding peer-to-peer network architecture, streaming over peer-to-peer overlay network and 3D content streaming in the following.

### *A.. Peer-to-Peer Overlay Network*

Peer-to-peer network is popular application-level network architecture in recent year. It can provide resource sharing in a distributed way. It resides on underlying network architecture and is a distributed architecture without central server involving. The first peer-to-peer network is Napster[3]. It deploy a central server as an index server, each peer can find the location of interest data items from the index server and contact the corresponding peer directly. With the development of peer-to-peer network, two peer-to-peer network architectures are appeared. One is Unstructured peer-to-peer network and the other is Structured peer-to-peer network. Gnutella[2] is such kind of Unstructured peer-to-peer network. Peers are connected in a distributed way and do not need to know overall topology. It uses flooding as the mechanism to send query to find the related information. In Structured peer-to-peer network, it assign key to the data and compute a value for the key. The *(key, value)* pair is used for retrieving and locating the data item. CAN[4], Chord[5], Pastry[6] and Tapestry[7] are the famous Structured peer-to-peer network system. Unlike flooding in Unstructured peer-to-peer network, Structured peer-to-peer network routing can be bounded in *O(logN)* hops, therefore, Structured


doi:10.4304/jsw.5.6.622-629

peer-to-peer network is efficient comparing with Unstructured peer-to-peer network.

### *B. Peer-to-Peer Streaming*

In the past streaming over Internet is popular research topic. Recently, because of the appearance of peer-to-peer network, many research projects start to pay attention on streaming over peer-to-peer network which is an application level service without considering underlying networks. In this section, we review some literatures regarding the peer-to-peer streaming system.

CoopNet[8] proposed by V. Padmanabhan *et al.* is a central server based solution for streaming service over peer-to-peer network. CoopNet use a central server to maintain the content delivery tree, the status of peer joining and leaving and employs multiple trees for managing and streaming data. However, the server will have heavy control loading to maintain overall information about the media streaming over peer-to-peer network. Another central server like solution is to deploy a directory server which holds peer address, peer network status etc.. All clients contact the directory server to obtain the streaming peer node and then download data from the peer directly. [11] and [12] are such kind of peer-to-peer streaming application. ZIGZAG[9], SplitStream[10] SpreadIt[14] and Narada[13] are another solution which use tree-based architecture. ZIGZAG[9] is a hierarchy based solution for peer-to-peer streaming service. ZIGZAG focus on one sender and multiple receiver fashion. ZIGZAG organizes the receiver in a multicast tree and content are delivered alone the multicast tree. SplitStream[10] divide the data into multiple data pieces and build multiple multicast tree for streaming the data pieces individually. Therefore, peer might join multiple trees to have the complete data. PROMISE[15] is a multi-sender solution which a receiving peer will collect streaming data from multiple senders and combine all data together. Because of PROMISE use multiple senders to provide data, when the network status is poor, it could switch the active downloading sender to better ones. CoolStreaming[16] is a mesh-based topology for peer-to-peer media streaming service. CoolStreaming is based on data-driven approach that the data are forwarded to peers which expects. CoolStreaming also use gossip-based approach to manage the membership of the peer-to-peer network without building predefined topology for peers.

### *C. 3D Streaming*

Because of the growth of network bandwidth and device capability, to streaming large 3D data is gained more attention in recently. In order to provide efficient 3D data transmission over Internet, many researchers consider the solution for streaming 3D data as well as video streaming. Progressive mesh[19],LOD(Level of Detail)[24] and QSplat[17] are the mechanism to provide progressive 3D streaming. Progressive mesh is proposed by Hoppe to render the 3D content from coarse to clear fashion in continuous transition way with edge split and collapse operations for 3D mesh model. Cohen *et al.*[24] proposes LOD(Level of Detail) approach to generate refinement data pieces with continuous detail of 3D polygon. QSplat is another solution for progressive streaming 3D model, but QSplat use point-based 3D model representation. However, the above approaches are view-independent without considering view of human. [22], [23], [18] provide view-dependent progressive streaming for 3D content which are dynamic based on the field of view of human eyes. The above researches focus on individual 3D model transmission. [20] and [25] propose the approach to stream not only 3D model but also 3D scene including many 3D models. Shun-Yun Hu[20] propose a 3D scene streaming framework based on Voronoi-based Overlay Network (VON) which is a kind of peer-to-peer network and limit user visibility within a area of interest. On the hand, Hu also use progressive mesh to preprocess the 3D content for steeaming. Dihong Tian *et al.*[25] propose the multistreaming approach to download 3D scene. It decompose the 3D scene into several 3D model within the scene as single streaming channel and propose a rate-optimization mechanism to have optimized performance and rendering quality for 3D scene.

### *D. P2P-Based Virtual Environment*

In the past, with the advent of peer-to-peer computing architecture, several applications replace the centralized architecture by distributed peer-to-peer computing diagram. The serverless architecture can not only reduce the server load but also balance the network load. In the virtual environment research area, some researchers also start to consider the development of peer-to-peer based virtual environment. In order to develop distributed virtual environment, the most important issue is the management of virtual environment status. Jehn-Ruey Jiang *et al.*[35] resolve neighbors consistency issue in virtual environment by using Voronoi-based Overlay Network(VON) as the basis of distributed virtual environment. Knutsson, B. *et al.*[37] apply peer-to-peer technology to develop massively multiplayer games. They divide the virtual environment into several disjoint regions and use the Pastry[6] as the peer-to-peer overlay network architecture to maintain virtual environment consistency. Y. Kawahara *et al.*[36] establish the connection between the neighbors directly and exchange the virtual environment status to keep the state consistency.

## III. DISTRIBUTED DYNAMIC 3D SCENE MANAGEMENT

In this section, we consider the distributed dynamic 3D scene management scheme first. 3D scene is a complex environment that includes many 3D materials. In order to manage different type of 3D objects, we consider the spatial relationship of 3D objects and propose multi-layer approach to manage 3D objects. First, we divide 3D objects into two categories include static and dynamic objects. Static objects in the 3D scene will not change. On the other hand, dynamic object will change its representation or position in the 3D scene. These objects in 3D scene have spatial relationship,. The objects are adjacent in 3D scene, the location of objects in CAN will

be adjacent also. Therefore, we use Content-Addressable Network(CAN)[4] as peer-to-peer network infrastructure to realize 3D object management which use Cartesian coordinate space to organize the location of peer and is suitable for spatial sensitive 3D scene. Because 3D scene includes two types of 3D objects, in order to manage different kinds of objects, we manage dynamic and static objects in different layers which are dynamic objects layer and static objects CAN layer. The complete 3D scene is represented by overlapping the dynamic and static objects' layer. Figure 1 shows the multi-layer segmentation of 3D scene. Each dynamic object belong to a zone in CAN-based P2P network and the corresponding static objects in the area of the zone are managed by the dynamic objects.

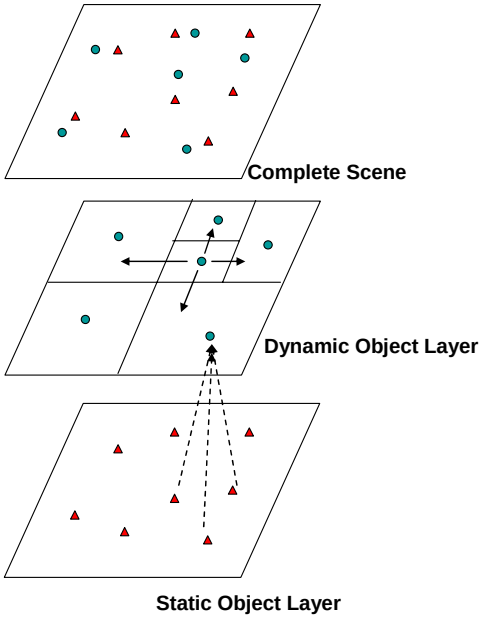


Figure 1. Multi-Layer Segmentation

### *A. State Management*

In order to keep the state consistence between 3D objects in 3D scene, we use push-based approach to update object state and pull-based approach to detect the status of neighbors. Because we divide the dynamic and static into different layer, static objects will be managed by the dynamic objects. Therefore, each dynamic object has two neighbor lists which are dynamic neighbor list and static list. Dynamic neighbor list maintain the adjacent objects which are needed to receive update message. Static list is to maintain the static object information in the same zone. In the following we describe dynamic and static objects management separately.

***Dynamic 3D Object Management*** : Dynamic 3D object will change its position, change its representation, enter 3D scene or leave 3D scene. When an object is changing, the changed object will send UPDATE message to the adjacent objects in the CAN network showed in Figure 2. Each UPDATE message receiving peer will determine whether the message will be forwarding or not with respect to the location of changing and receiving peer. If the receiving peer is far away from changing peer, the remaining adjacent peers of receiving will be far away from changing peer also. Otherwise, the receiving peer will forward the UPDATE message to remaining peers. On the other hand, when a dynamic object is moving out the zone and entering the other zone, the object will send REORGANIZE message to neighbor objects of entering and leaving zone to reorganize the CAN-based peer-to-peer network.

***Static 3D Object Management*** : Static 3D object is persistent, so static 3D object will not need UPDATE message. But in order to share the content between dynamic objects, each zone of dynamic object will be responsible for manage the static within the area of the zone. And the static 3D object will be distributed by the peer of the zone. As showed in Figure 1, triangle objects are static objects and theses object will be managed by corresponding dynamic object within same zone area.

Dynamic objects might leave the zone suddenly, so the 3D objects might lose the consistency. Therefore, besides the push-based UPDATE message, the object will send pull-based KEEPALIVE message to detect the status of neighbors periodically. When the object detect the neighbor is not active, the object will send the TKAEOVER message to other alive neighbors to decide which peer will take the un-available object's responsibility. At this moment, the static objects' data will be downloaded from bootstrap server.

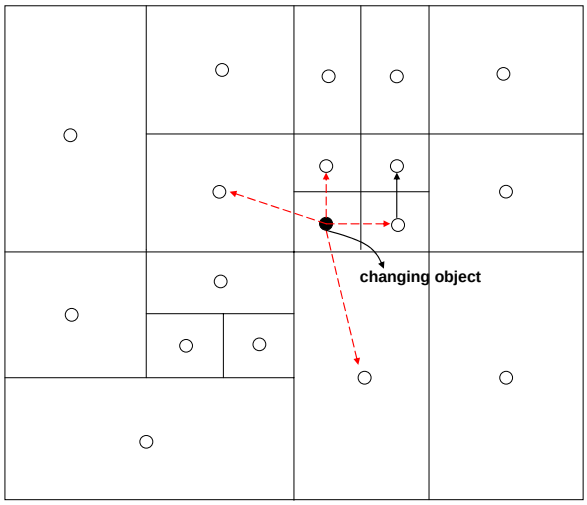


Figure 2. CAN-based Object Locating

## IV. COLLABORATIVE 3D STREAMING

In order to realize the real-time streaming for 3D content over P2P network, traditional streaming technology is not suitable for 3D content delivery. In this section, we propose our design to achieve 3D content streaming efficiently over P2P networks.

### *A. Concept*

In order to streaming massive 3D content, we describe the overall architecture of 3D scene streaming over Peer-to-Peer network in Figure 3. Peer A is the node that requiring the 3D scene. Peer A would acquire the 3D

scene information from server. After acquiring the information from the server, peer A would determine the object streaming priority with respect to the object importance by multi-layer segmentation for 3D scene and monitor network status in the peer-to-peer network. Finally, peer A would render the 3D data and display the 3D scene by using multi-flow approach which is responsible for partial data of 3D scene.

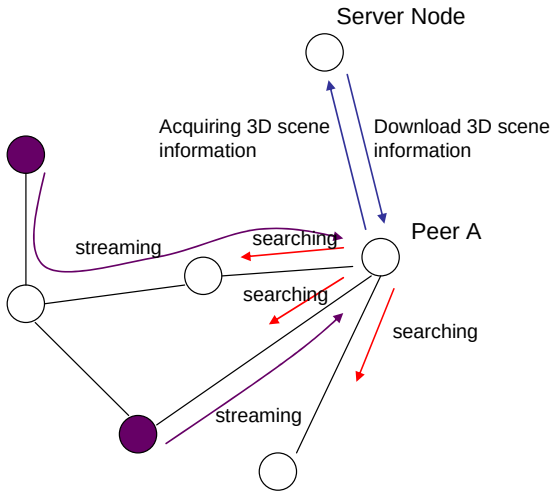


Figure 3. 3D Scene Streaming over Peer-to-Peer Network

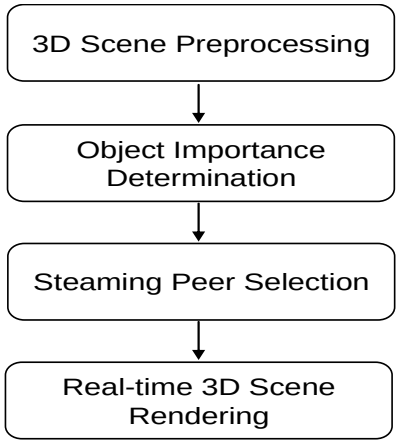


Figure 4. Steps for 3D Scene Streaming Processing

In order to achieve the 3D scene streaming, we divide the overall processes in the following steps and shows in Figure 4.

***3D Scene Preprocessing*** : 3D scene contain several individual objects and there exists temporal and spatial relationships between these objects in the scene. In order to represent the streamed 3D scene, 3D scene should be encoded in a flexible way. Therefore, we use XML-based description format which is a text-based representation and easy to process to describe 3D scene.

***Object Importance Determination*** : Because of the objects in the 3D scene have spatial and temporal relationships, objects have different priority and visualization with respect to the objects' relationship and users' viewpoint. Therefore, we need to decide the streaming priority for each object in this process.

***Steaming Source Selection*** : After assigning the importance for each objects, we lookup the object data and search for the data in the peer-to-peer network according the object's importance and priority.

***Real-time 3D Scene Rendering*** : When the object streaming priority is determined, we could start to stream each object and render the 3D scene real-time.

### *B. 3D Scene Graph*

The 3D scene graph is used for represent the objects' relationship. Based on the 3D scene graph, we could determine which object should be displayed first. The proposed 3D scene graph is similar with other scene graph, for example, x3D, Java 3D etc., but our proposed 3D scene graph is simpler than others. The main purpose of our 3D scene graph is to present the relative position of objects that is the implicit annotation for object importance. Therefore, a preprocessing server will recognize the 3D scene into 3D scene graph first. In order to encode and decode the 3D scene graph in a general approach, we use the XML markup language to define the 3D scene graph structure. Figure 5 shows the XML-based 3D scene graph structure. Each 3D scene graph representation is starting at 3DGraph tag. Because each 3D scene conclude server objects, each object will be starting at Object tag and include Name tag, Identifier tag, LOD_Sequence tag, and position tag to represent object's name, object's identifier, object LOD(Level of Detail) sequence, object's x-axis and y-axis.. On the other hand, some objects might be occluded with each other, the occluded objects will be starting at GroupObject tag with its x-axis and y-axis. Each GroupObject also conclude several objects as well as non-occluded object in the 3D scene.

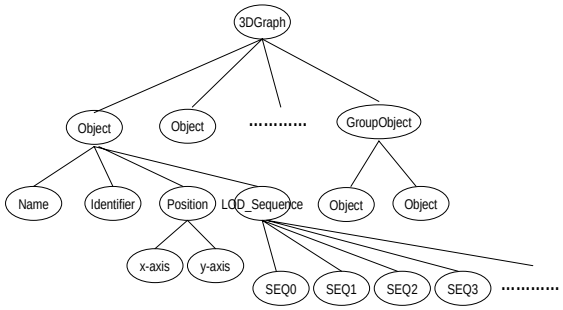


Figure 5 : XML-based 3D scene graph

### *C. Multiple Streaming Source Selection*

The 3D scene includes several objects and the objects will reside on some of peers of the peer-to-peer network. When a peer requires to stream 3D scene from peer-to-peer network, the requesting peer would lookup the peers which have the objects. In our propose system, the 3D scene is divided into several objects and the objects are encoded into flexible LOD(Level of Detail) format. Therefore, the requesting peer searches the LOD piece of objects instead of searching for individual objects. LOD is a technology for reducing the complexity of 3D object which is to be rendering. The overall LOD information of a 3D object includes a base mesh and several refinement

meshes. Based on LOD technology, a 3D rendering system can only download the necessary LOD pieces which are closed to human eyes. Therefore, in order to have better streaming quality, we need to set priority for objects in the 3D scene. First, we introduce the object importance which is the download priority for 3D scene streaming. We define two types of object importance including horizontal importance and vertical importance. We use multilayer approach to divide 3D scene into different layer with respect to distance from human eyes to object. We use constant distance interval to separate different layer. The closer layers from human eye to object have high priority, therefore, the object in the layer would have high importance which is called object vertical importance and illustrated in Figure 6. On the other hand a layer might have several objects, we calculate the importance according to the visual angle of human eyes, this kind of importance is called object horizontal importance which is showed in Figure 7. In Figure 8 we reveal the algorithm for determining the object vertical and horizontal importance with the 3D scene graph information.

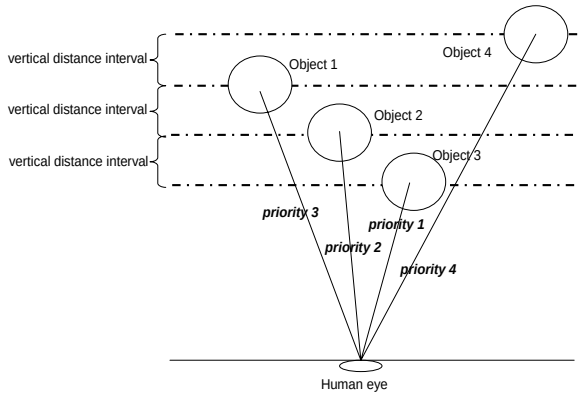


Figure 6 : The Illustration for Object Vertical Importance

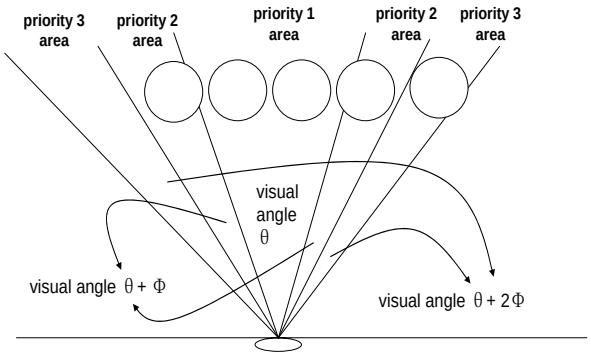


Figure 7 : The Illustration for Object Horizontal Importance

After obtaining the object vertical importance and object horizontal importance form the 3D scene graph, we could determine the object streaming priority with the following rules :

- If vertical importance of object *i* is higher than object *j*, object *i* have higher priority.
- If vertical and horizontal importance of object *i* and object *j*, object *i* and object *j* have same priority.
- If vertical importance of object *i* and object *j* are same, and horizontal importance of object *i* is higher than object *j*, object *i* have higher priority.

In our proposed system, we use LOD(Level of Detail) to encode each object of 3D scene, each object has a base mesh and different number of refinement mesh with corresponding texture data. And the streaming data unit is LOD pieces, that is, base mesh or refinement mesh with its corresponding texture data. Therefore, we use greedy algorithm to calculate the better streaming quality with respect to the maximum of requesting peer's downlink capacity, the source peer's uplink capacity, the delay between source peer and requesting peer, and object importance. In our designed greedy algorithm, we maximize the requesting peer's downlink capacity with minimizing the cost which is defining in Equation (1). And Figure 9 shows the proposed greedy algorithm for selecting source peer.

$$Cost_{min} = min \{ \sum delay_k \times Size_{ij}/rate_k : i \in objects, j \in mesh\ of\ object\ i, k \in peers\ that\ have\ mesh_{ij} \} \quad (1)$$

where $Size_{ij}$ means the size of sum of $mesh_{ij}$ and $texture_{ij}$

TABLE 1 : NOTATION FOR ALGORITHM OF OBJECT IMPORTANCE DETERMINATION

| Notation | Description |
|---|---|
| *Oi* | the *i*th object |
| *Oi.vi* | the vertical importance of object *i* |
| *Oi.hi* | the horizontal importance of object *i* |
| *NP* | the nearest distance from user view to the closest object |
| *vDist(view-Oi)* | the vertical distance from user view to object *i* |
| *hDist(view-Oi)* | the horizontal distance from user view to object *i* |
| *Li* | *i*th layer of 3D scene |
| $\varepsilon$ | the horizontal distance offset |
| *vDistOnterval* | the vertical distance offset |

*D. Real-time 3D Scene Streaming*

In order to streaming overall 3D scene, we use multiple streaming flows to download each object and then combine the downloaded object data into scene which will be explained in next section. The requesting peer broadcasts a search message that including all object identifier to lookup the peers that holds the object data. But in our proposed system, the unit of searching object data is LOD data pieces identifier which is composited by object identifier and LOD sequence number instead of searching object identifier. When object LOD pieces are found, the peer will response to requesting peer with its network uplink capacity which denote the peer's network status. And then requesting will probe a message to estimate network delay between requesting peer and other peers that contain the LOD data pieces.

In real-time 3D streaming process, streaming data and rendering data are the two important tasks. In previous section, we already have the object downloading priority. In order to guarantee the streaming quality and real-time rendering, we need to monitor the network status.

Because our proposed method use the LOD data pieces instead overall object, therefore, the object is streaming from multiple source peers and rendering. We use OCPN[31] to model the collaborative streaming system. When we obtain the streaming source set with respect to object importance and network status described previous section, we can model the streaming system as streaming machine showed in Figure 10. In the figure, there exist two places indicating two object streams, one rendering place to render 3D scene and one transition state indicating the longest time that the rendering state can wait. If the actual time to reach transition state is exceeded a defined threshold, we need to execute the object lookup again to obtain the better ones. Alone with the streaming machine, we have propose multi queues including multiple source queues and one rendering queue to maintain the streaming data. As the example of Figure 10, we have two source queues to maintain the LOD data piece for two objects and each queue would be filled from multiple peers collaboratively. Then the data in the multiple queues will be move to rendering queue for 3D scene rendering. The process is showed in Figure 11.

```
Input : 3D Scene Graph
Output : Object Importance
for each Obejct Oi
        if NP>vDist(view-Oi)
                NP ← vDist(view-Oi)
for each Object Oi
        Δ ←1
        while vDist(view-Oi) < NP+ Δ *vDistInterval
                Δ ← Δ +1
        Oi.vi ← Δ -1
for each Layer Li
        for each Object j in Li
                Δ ←1
        while hDist(view-Oj) >Δ*ε
                Δ ←Δ+1
        Oj.hi ←Δ-1
```

Figure 8 : Algorithm for Object Importance Determination

```
Input : Object Download Priority
Output : downloadSet
for each priority group g
        find the peer k with Cost_min
        if (downlink – rate_k)>0
                downlink ← downlink – ratek
                downloadSet ← downloadSet ∪ k
        else return
```

Figure 9 : Algorithm for Source Peer Selection

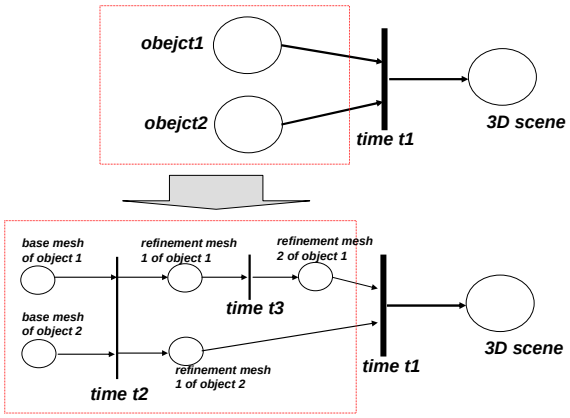


Figure 10 : An Example of Streaming Machine

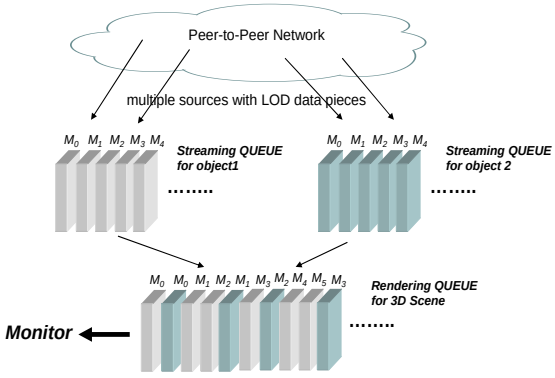


Figure 11 : Queuing Machine for Collaborative Streaming

## V. System Implementation

In order to make our proposed method is practical, we implement a system prototype based JXTA technology[26] proposed by Sun Microsystem Inc.. JXTA is a common platform for developing peer-to-peer applications and defines a set of protocols to realize the peer-to-peer application's behavior. According to the JXTA core protocols, we describe our design of proposed collaborative 3D streaming system. In our design, each JXTA peer is a device that obtains the partial or fully LOD data pieces and the peers that have LOD data pieces belong to same object will formed a JXTA PeerGroup. We focus on two JXTA protocols including Peer Resolve Protocol(PRP) and Peer Information Protocol(PIP) mainly. When we lookup the LOD data pieces for the object, we will send query message via Peer Resolve Protocol(PRP). And the message includes the hash value from the object identifier and LOD sequence number that indicating the LOD data index. On the other hand, we need to monitor the status of the streaming peer, so we will query the status via Peer Information Protocol(PIP). After obtaining the LOD data pieces from the JXTA-based peer-to-peer network, we will stream the LOD data pieces from multiple peers by RTP protocol[27], mixing the received data and rendering to 3D scene. Figure 12 shows the design software block of proposed collaborative 3D virtual environment software architecture. The system includes three major sub-blocks including JXTA block, collaborative 3D scene streaming

software block and 3D scene management. The JXTA block composes all basic processing module of peer-to-peer network including searching, and communicating etc.. And the collaborative 3D scene steaming block includes the processing procedure regarding 3D scene graph processing, object importance determination, multi-flow steaming management etc.. 3D scene management will be responsible for maintaining the neighbors' lists, replicated data pieces and management messages processing. When the replicated data pieces size exceed the buffer size, the less referenced data pieces will be move out the buffer first.

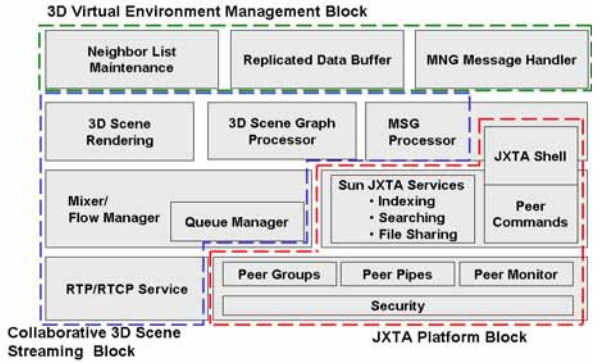


Figure 12 : Collaborative 3D Virtual Environment Software Architecture

## VI. CONCLUSION

In this paper, we propose the design of collaborative 3D scene management and streaming system for distributed management and real-time transmission and rendering 3D scene. The proposed system includes the following contributions. First, object importance determination is proposed based on multi layer segmentation on 3D scenes. Second we use collaborative approach to steaming 3D data from multiple peers to have better performance and quality. Third, we propose distributed 3D scene management mechanism based on CAN-based peer-to-peer network. Finally we use standard implementation platform JXTA to realize peer-to-peer network architecture and standard RTP/RTCP real-time transmission protocol to implement real-time streaming service. However, we do not consider robust issue to make fault tolerance system that is an importance topic regarding real-time transmission application. We will address this issue in the future. On the other hand, in order to make RTP can carry the 3D content, a standard 3D content encoding scheme also need to define for future 3D virtual environment and real-time applications.

## ACKNOWLEDGMENT

We would like to thank lab members in Minghsin University of Science and Technology for their help. This research is also supported by National Science Council, Taiwan, with grant no. NSC-97-2221-E-159-016-.

## REFERENCES

[1] E Lua,J Crowcroft,M Pias,et al., "A survey and comparison of peer-to-peer overlay network schemes", IEEE Communications Survey and Tutorial, Vol.7, No.2, 2005, pp. 72–93.
[2] Gnutella development forum, the gnutella v0.6, 2001, Avaliable: http://groups.yahoo.com/group/the_pdf/files/
[3] Napster.[Online]. available: http://www.napster.com/
[4] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content addressable network", in Proceedings of the ACM SIGCOMM, 2001, pp. 161–172.
[5] Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup protocol for internet applications," IEEE/ACM Transitions on Networking, vol.11, no.1, 2003, pp. 17–32.
[6] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems", IFIP/ACM International Conference on Distributed Systems Platforms (Middleware), Heidelberg, Germany, November, 2001, pp. 329–350.
[7] Y. Zhoa, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowica, "Tapestry: A resilient global-scale overlay for service deployment," IEEE Journal on Selected Areas in Communications, vol. 22, no. 1, January 2004, pp. 41–53.
[8] V. Padmanabhan, H. Wang, P. Chou, and K. Sripanidkulchai, "Distributing Streaming Media Content Using Cooperative Networking", In Proc. of NOSSDAV'02, Miami Beach, FL, USA, May 2002.
[9] D. Tran, K.Hua, and T. Do., "ZIGZAG: An Efficient Peer-to-Peer Scheme for Media Streaming", In Proc. of IEEE INFOCOM'03, San Francisco, CA, USA, April 2003.
[10] M. Castro, P. Druschel, A-M. Kermarrec, A. Nandi, A. Rowstron and A. Singh, "SplitStream: High-bandwidth content distribution in a cooperative environment", In Proc. the International Workshop on Peer-to-Peer Systems, Berkeley, CA, February, 2003.
[11] http://www.ppstream.com
[12] http://www.pplive.com
[13] Yang-Hua Chu, Sanjay G. Rao, and Hui Zhang, "A Case for End Systems Multicast", In ACM SIGMETRICS, 2000, pp. 1-12.
[14] H. Deshpande, M. Bawa, and H. Garcia-Molina, "Streaming Live Media over a Peer-to-peer Network", Technical report, Stanford University, 2001.
[15] M. Hefeeda, A. Habib, B. Botev, D. Xu, and B. Bhargava, "PROMISE: Peer-to-Peer Media Streaming Using CollectCast", In Proc. of the 11th ACM international conference on Multimedia, Berkeley, CA, USA, November, 2003.
[16] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "Coolstreaming/DONet: A data-driven overlay network for live media streaming", In Proc. IEEE INFOCOM'05, 2005.
[17] S. Rusinkiewicz and M. Levoy, "Streaming qsplat: a viewer for networked visualization of large, dense models," in ACM Interactive 3D 2001 Conference Proceedings, 2001, pp. 63–69.
[18] S. Yang, C.-S. Kim, and C.-C. J. Kuo, "A progressive view-dependent technique for interactive 3-d mesh transmission," IEEE Transactions on Circuit and System for Video Technology, vol. 14, no. 11, 2004, pp. 1249–1264.
[19] H. Hoppe, "Progressive meshes," in ACM SIGGRAPH 1996 Conference Proceedings, 1996, pp. 99–108.

[20] S.-Y. Hu, “A case for peer-to-peer 3d streaming,” in ACM Web 3D 2006 Conference Proceedings, 2006, pp. 57–63.
[21] N.-H. Lin, T.-H. Huang, and B.-Y. Chen, "3D model streaming based on a jpeg 2000 image," in Proceedings of IEEE 2007 International Conference on Consumer Electronics, 2007.
[22] J. C. Xia and A. Varshney, “Dynamic view-dependent simplification for polygonal models,” in Proc. Visualization, 1996, pp. 327–334.
[23] R. Southern, S. Perkins, B. Steyn, A. Muller, and P. M. Blake, “A stateless client for progressive view-dependent transmission,” in Proc. Web3-D Symp., ACM, 2001, pp. 43–50.
[24] J. Cohen, A. Varshney, D. Manocha, G. Turk, H. Weber, P. Agarwat, F. Brooks, and W. Wright, “Simplification envelope,” in Proc. Computer Graphics, Annu. Conf. Series, ACM SIGGRAPH, Aug. 1995, pp. 119–128.
[25] Dihong Tian, Ghassan AlRegib, “"Multistreaming of 3-D Scenes With Optimized Transmission and Rendering Scalability”, IEEE TRANSACTIONS ON MULTIMEDIA, VOL. 9, NO. 4, JUNE 2007, pp. 736–745
[26] JXTA v2.0 Protocols Specification, Project JXTA http://www.jxta.org
[27] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, “RTP: A Transport Protocol for Real-Time Applications”, IETF RFC 3550, July 2003.
[28] M. Schlosser, M. Sintek, S. Decker, W. Nejdl, "Hypercup shaping up peer-to-peer networks", Technical Report, Stanford University, 2001.
[29] F. Furtek, “A New approach to Petri Nets”, MIT Project MAC, Apr. 1975.
[30] K. Jensen, “Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use”, Volume 2, Analysis Methods. Monographs in Theoretical Computer Science, Springer-Verlag, 2nd corrected printing 1997.
[31] T. C. D. Little and A. Ghafoor, “Synchronization and storage models for multimedia objects”, IEEE Journal on Selected Areas in Communications, Vol. 4, pp413 – 427, Apr. 1990.
[32] M. Woo, N. U. Qazi, and A. Ghafoor, “A synchronization framework for communication of pre-orchestrated multimedia information”, IEEE Network, pp 52-61, Feb. 1994.
[33] S.-U. Guan and S.-S. Lim, “EP-net: A Synchronization Model for Authoring Interactive Multimedia Applications”, 1999.
[34] S.-U. Guan, H.-Y. Yu, J.-S. Yang, "A Prioritized Petri Net Model and Its Application in Distributed Multimedia Systems," IEEE Transactions on Computers, vol. 47, no. 4, pp. 477-481, Apr. 1998
[35] Jehn-Ruey Jiang, Jiun-Shiang Chiou and Shun-Yun Hu, "Enhancing Neighborship Consistency for Peer-to-PeerDistributed Virtual Environments," the First IEEE International Workshop on Cooperative Distributed Systems (CDS), 2007.
[36] Y. Kawahara, T. Aoyama, and H. Morikawa, “A peer-to-peer message exchange scheme for large-scale networked virtual environments”, Telecomm. Sys., Vol. 25, No. 3-4, pp.353-370, 2004.
[37] Knutsson, B., Honghui Lu, Wei Xu, and Hopkins, B., "Peer-to-peer support for massively multiplayer games", In Proceeding of INFOCOM 2004, pp. 96-107, 2004

**Chuan-Feng Chiu** received his Ph. D. degree in Computer Engineering from Tamkang University, Taiwan in 2002. He also received his MS and BS degrees in the same major in 1999 and 1995, respectively. From 2002 to 2006 he was a senior engineer in Panasonic Taiwan Laboratory in Taiwan. Since 2006 he has been working as a Assistant Professor in Department of Information Management, Minghsin University of Science and Technology, Xinfeng Hsinchu, Taiwan. His research interests include multimedia system, peer-to-peer computing, home network and network security. Since 1999, he published many refereed papers in international conferences and journals. He is an honor member of Phi-Tau-Phi Scholastic Honor Society of the Republic of China.

**Steen J. Hsu** was born in Taipei County, Taiwan, Republic of China, on September 10, 1963. He received the M.S. and Ph.D degrees in computer science and information engineering in 1992 and 1999, respectively, from the National Chiao Tung University, Hsinchu,Taiwan.

Form August 1999 to July 2005, he was an Assistant Professor of the Department of Information Engineering at I-Shou University, Kaohsiung County, Taiwan. Since August 2005, he has been an Assistant Professor of the Department of Information Management at Minghsin University of Science and Technology, Hsinchu County, Taiwan. His research interests include wireless LAN, multimedia communication, information security, and P2P networks.

Dr. Hsu is a member of the IEEE Communication Society and the Phi Tau Phi honor society of the Republic of China.

**Sen-Ren Jan** received the BS degree in computer science from Tamkang University, Taipei, Taiwan, in 1983 and the MS, PhD degree in computer and information science from National Chiao Tung University, Hsinchu, Taiwan, in 1991, 2000, respectively. He is currently an assistance professor in the Department of Information Management at Minghsin University of Science and Technology, Hsinchu, Taiwan. His current research interests include mathematical morphology, image processing, data hiding and pattern recognition.