Progressive Cache Replacement for Massive Peer-to-Peer WebVR Worlds

Wei Wang
School of Electronics and Information
Tongji University, China

Jin-Yuan Jia, Yang Yu
School of Software Engineering
Tongji University, China
jyjia@tongji.edu.cn

Shun-Yun Hu
Institute of Information Science
Academia Sinica, Taiwan, R.O.C.
syhu@iis.sinica.edu.tw

***Abstract*—As the popularity of WebVR applications rises in recent years, a great disparity exists between the huge 3D content on servers and the limited cache capacity on clients. In order to alleviate server loading, peer-to-peer (P2P)-based 3D content distribution has been proposed recently. However, given the limited cache sizes at clients, how to maintain and replace the cache content effectively becomes an important issue. We present a *progressive scene replacement mechanism* (PSRM) in this paper to support interactive walkthrough in P2P-based large-scale WebVR worlds. First, we define a new metric called *preservation degree*, based on both the visual attention paid by the user and the content's potential relevance for sharing. Second, cached objects are replaced progressively with ascending order in terms of their preservation degrees. Experimental results have shown that PSRM enables users with limited cache to walkthrough a large-scale WebVR world with high visual quality; while download requests handled by the servers are reduced significantly.**

***Keywords*-WebVR; P2P; preservation degree; progressive scene replacement mechanism**

I. Introduction

With the fast advancement of Internet and virtual worlds, *distributed virtual environments* (DVE) [1] and Web based Virtual Reality (WebVR) [2] have become popular in recent years (e.g., *Massively Multiuser Online Games* (MMOG) [3] and military simulations [4]). By sharing a *virtual environment* (VE), geographically dispersed users could roam or interact with others on the Internet. Originally, the 3D content in a VE system (e.g., models and textures that allow the rendering of rooms, scenes, and virtual representations of users called *avatars*) is all pre-installed and stored at the local client before the user enters the DVE. However, as the full content of a VE grows into the range of terabytes (e.g. over 34 TB data exists for the virtual world *Second Life* [5]), it becomes increasingly inconvenient or even impossible for users to install all the content at once. For thin clients such as browsers, personal digital assistants (PDAs), or mobile devices, it is also neither necessary nor practical for client machines to store all content. Various 3D content distribution techniques (also known as *3D streaming*) have thus been proposed, and fall into two main categories: client/server (C/S) (e.g., Cyberwalk [6], GameOD [3]) and *peer-to-peer* (P2P) (e.g., Solipsis [7], HyperVerse [8], LODDT [9] and FLoD [10]). C/S DVEs are easier to construct and maintain, but the servers can become a bottleneck as their resources deplete with the sharp growth in user size. In contrast, P2P-DVEs [7] utilize bandwidth and CPU resources from clients, so server workload may reduce significantly as more users join the DVE. Hence, if we were to support a WebVR world shared by a massive number of users, for example, a walkable, multi-user Google Earth joined by millions of users concurrently, P2P provides a potentially more efficient solution.

The basic idea for P2P-based 3D scene distribution is that as users may have overlapped visibility within the DVE, nearby users within a user's visibility can thus exchange 3D content mutually to save some requests to the server. However, given the limited cache size at clients, certain content must be removed from cache as the user browses the VE. Although some DVE scene replacement polices have been proposed for client-server architectures [11], [6], to the best of our knowledge, there is not yet a specially designed policy for P2P-DVE. In this paper, in addition to the classic visual factor widely used for scene replacement in C/S-DVE, we propose the new concept of *potential relevance degree*; then, based on these two factors, we design a progressive scene replacement mechanism for real-time walkthrough in large scale P2P-DVEs.

The rest of this paper is organized as follows. Related work of scene replacement are summarized in Section II. Section III introduces the network architecture of the P2P-DVE. The procedures for computing preservation degree of an object are described in Section IV. Section V describes progressive WebVR scene replacement mechanism in details. Experimental results and performance analysis are discussed in Section VI. Finally, Section VII concludes this paper and describes future work.

II. Related Work

A. Progressive Transmission of Scenes

As a viewer often can only observe a portion of the whole WebVR world given its limited visibility and the occlusion between objects, to save disk space and download time, viewers are allowed to just download visible scenes at the current viewpoint for immediate rendering. Then as the viewpoint moves, newly visible scenes are downloaded progressively.

Some papers have employed such *area of interest* (AOI) [6], [12] criterion for the progressive downloading of scenes.

As scenes located in AOI may still be huge in a dense WebVR world, researchers have developed *Level of Detail* (LOD) for the progressive transmission (or *streaming*) of objects. Since such multi-resolution model simplification is beyond our scope, we utilize a typical continuous LOD for the progressive transmission of scenes (e.g., *progressive meshes* (PM) [13]). We assume that after processing by PM techniques, a 3D object is divided into a *base mesh* representing the lowest resolution of the object, and a series of *PM increment* pieces that can restore the model to its original resolution.

B. Replacement of Scenes

When the limited local cache is crammed by scenes, some existing scenes have to be removed for the new scenes. Classic page-based data replacement policies such as *Least Recently Used* (LRU), *Most Recently Used* (MRU) [14], [15] have been utilized in database applications widely; the good performance of these replacement policies depends on the *principle of locality* to a great extent (i.e., the higher degree of locality, the better performance). However, researchers have shown that these replacement policies are unsuitable for replacing scenes as objects accessed by a client might change over time [16].

In [11], [10], objects are removed first if they are far from the viewpoint. However, both methods did not consider the impact of objects' deviating angle from the viewpoint. The *Most Required Movement* (MRM) in [6] streams each object with a PM technique and assigns an *access score* to objects in terms of its distance and deviation angle from the viewpoint. First, the object with the lowest access score is selected for replacement, then extra PM increments of this object are removed until its *optimal resolution* is reached. If there is still no enough room, the object with the next lowest access score would be similarly processed, and the process iterates. If there is still no room when all cached objects have been reduced to their optimal resolutions, all the remaining PM increments of the object with the lowest access score would be removed, leaving only its base mesh, and this process will be iterated again. Finally, the base mesh of the object with the lowest access scores would be removed. A hybrid scene replacement policy has been proposed in [17], its replacement order of a 3D object is determined by both temporal and spatial coherence.

The above work on scene replacement mainly focuses on the spatial relation between objects and viewpoint, and are designed for C/S-DVE. However, when determining object removal in a P2P-DVE, besides the spatial relation between objects and the viewpoint, specific characteristics of the P2P-DVE (e.g., the potential influence of a cached object on the viewer's neighbors) should also be taken into account.

III. P2P-DVE Scenario

We first describe the scenario of our P2P-DVE in Fig. 1, so that discussions in the rest of this paper will be more concrete. The network architecture consists of a physical network layer and a P2P overlay layer. The physical network is composed of a server (or server clusters) and geographically dispersed client nodes (i.e., viewers).

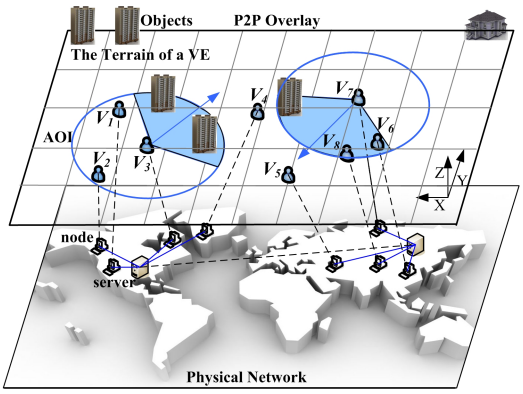

Fig. 1: Network architecture of the P2P-DVE

According to the spatial relation of viewers in the VE, if a viewer is located in the AOI of another viewer, the former is treated as the latter's AOI neighbor. As shown in Fig. 1, viewers V_1 and V_2 are both AOI neighbors of viewer V_3. Three types of participator exist in the P2P-DVE as follows.

- **Server**: the server stores a copy of the whole WebVR world; it also manages node login.
- **Requester**: a node that requires scenes pieces.
- **Downloading Source**: a node that holds a required piece for another requester.

Similar to the procedures used by FLoD [10], each viewer client goes through the following steps in the DVE: 1) join the DVE; 2) determine visible scenes; 3) discover AOI neighbors; 4) discover downloading sources; 5) select downloading sources; 6) transmit scenes; 7) replace scenes. Please refer to [10] for details.

IV. Preservation Degree of Objects

When a viewer has to replace some scenes from its local cache, assume that the set of its cached objects is O, $O = \{O_1, O_2, ..., O_m\}$. We define the *preservation degree* of an object O_i to show how important is O_i to the viewer, by computing: the *visual attention degree* and the *potential relevance degree* on the AOI neighbors. A viewer will integrate these factors to evaluate a cached object's final preservation degree; then, the preservation degrees will determine which objects shall be replaced.

A. Visual Attention Degree

Assume that D_i is the distance of O_i to the viewpoint, R is the radius of AOI, θ_i is the angle O_i deviates from the line of sight ($0 \leq \theta_i \leq \pi$). We define *visual attention degree* as follows.

Definition 1: Assume that $A(O_i)$ is the *visual attention degree* of O_i for current viewpoint:

$$A(O_i) = \lambda(1 - \frac{D_i}{R}) + (1 - \lambda)cos\frac{\theta_i}{2} \quad (1)$$

Among all possible formulae which can be employed to calculate the visual attention degree, here we select a simple and intuitive one to model the distance and angle. An illustration of each parameter of Formula (1) is shown in Fig. 2. The λ in Formula (1) is the weight, $0 \leq \lambda \leq 1$, usually, λ is set as 0.5. From Formula (1) we can deduce that: the farther an object is from the viewer and the larger angle an object is from the viewer's line of sight, the smaller the visual attention degree.

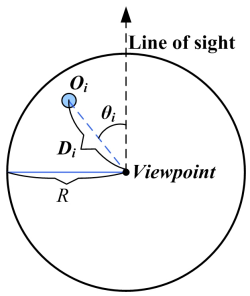

Fig. 2: Illustration of visual attention degree

B. Potential Relevance Degree

As shown in Section II, the order of scene replacement in most C/S-DVEs is only determined by the visual attention degrees. However, in P2P-DVE, the removal of a 3D object from a node's local cache will have potential influences on its AOI neighbors, so such influence has to be considered for better replacement performance. In this section, we consider such influence when removing certain objects from cache.

1) **Area of Objects that Affect Neighbors (AOA):** During a walkthrough, a viewer and its AOI neighbors are interested in the same 3D objects within their shared viewing areas; considering the areas a viewer and its AOI neighbors might have visited or will visit, removing objects within these areas may adversely affect a viewer's AOI neighbors' to obtain scenes. Here, the *Area of Objects that Affect Neighbors* (AOA) is defined and shown in Fig. 3.

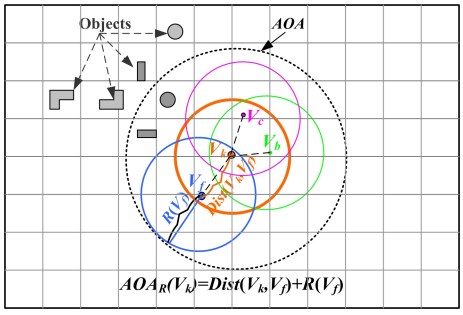

Fig. 3: Illustration of AOA

Assume that V_k denotes the viewer *k*, the radius of its AOI is $R(V_k)$, AOI_k is the set of AOI neighbors of V_k, AOI_k = $\{V_1, V_2, ..., V_n\}$. the distance from V_k to its AOI neighbor V_i is $Dist(V_k, V_i)$ (1$\leq$ i $\leq$ n), then, the AOA of V_k is the radius:

$$AOA(V_k) = \max_{V_i \in AOI_k} Dist(V_k, V_i) + R(V_i) \qquad (2)$$

For example, in Fig. 3, as the farthest AOI neighbor of V_k is V_f, so $AOA(V_k)$ can be computed as: $Dist(V_k, V_f)$ + $R(V_f)$.

2) **Potential Influence on AOI Neighbors:** From the perspective of a given viewer V_k, the more times an object O_i is downloaded by its AOI neighbors, the more AOI neighbors are in possession of this object. Assume that V_k removes O_i with a higher priority, then normally, it is unlikely that all nodes in AOI_k will have removed O_i. When V_k requires O_i again, V_k can still get it from some nodes in AOI_k.

On the contrary, if V_k removes (with higher priority) an object O_i that is requested less by AOI neighbors, it is possible that no other sources in AOI_k will have O_i. When O_i is needed again, V_k cannot download it directly from some AOI neighbors instantly, and will have to search for new downloading sources. In a bad case, the requests would be served until several attempts, which increases both latency and bandwidth consumption. This analysis shows that a viewer and its AOI neighbors' removal behaviors are related and could affect scene retrieval.

Definition 2: Assume that there are *n* nodes in AOI_k, given an object O_i in the AOA of V_k, $DI_{V_j}(O_i)$ is the number of times PM increments of O_i are downloaded by V_j, and $DB_{V_j}(O_i)$ is the number of times the base mesh of O_i is downloaded by V_j. The *potential relevance degree* of O_i on one AOI neighbor V_j(0 $\leq$ j $\leq$ n) in AOI_k is $R(O_i, V_j)$:

$$R(O_i, V_j) = \frac{1}{\sqrt{1 + DI_{V_j}(O_i) + DB_{V_j}(O_i)}} \qquad (3)$$

From Formula (3), we can deduce that the fewer times an object is downloaded by AOI neighbors, the higher the $R(O_i, V_j)$. Based on Formula (3), given a viewer V_k, the potential relevance degree of O_i for all nodes in AOI_k is:

$$R^{AOI_k}(O_i) = \sum_{i=1}^{n} \frac{R(O_j, V_i)}{n} \qquad (4)$$

C. Preservation Degree of Objects

Given the visual attention degree of O_i and the potential relevance degree on AOI neighbors of O_i, the *preservation degree* (PD) of O_i can be computed by (5), where the α is a weight, $0 \leq \alpha \leq 1$. Usually, the α is set as 0.5.

$$PD(O_i) = \alpha A(O_i) + (1 - \alpha) R^{AOI_k}(O_i) \qquad (5)$$

V. Progressive Replacement of Scenes

Assume that the cache capacity of V_k is C_{whole}, current cached data volume is $C_{occupied}$, and the data volume of the next downloading request is as $D_{required}$.

1) If C_{whole}-$D_{required} \leq C_{occupied}$, then invoke the object replacement procedure;
2) If C_{whole}-$D_{required} > C_{occupied}$, then stop the object replacement procedure.

Our *progressive scene replacement mechanism* (PSRM) can be divided into three major stages: 1) remove PM increments until the optimal resolution (we use the method in Cyberwalk

[6] for computing optimal resolution); 2) remove remaining PM increments; 3) remove the base mesh.

First, PSRM reduces objects to their optimal resolutions, starting from the cached object with the lowest PD, so the first stage of PSRM is similar to that of MRM. When there is still not enough cache after the first stage, PM increments will be removed incrementally, starting with the highest resolution of the object with the lowest PD; then, the same operation will repeat on the object with the next lowest PD, until no PM increments can be removed (see Fig. 4). Finally, in the worst case, if there is still not enough room, the base meshes of objects will be removed, starting with the object with the lowest PD, and this process will iterate.

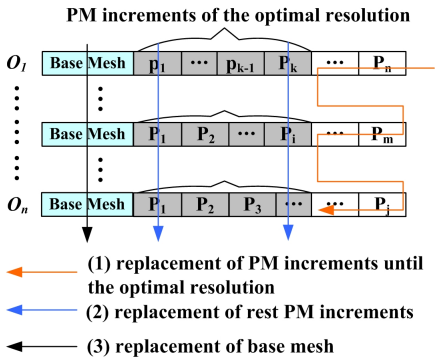

Fig. 4: Illustration of PSRM, the arrows show the replacement order of 3D contents

VI. Experimental and Performance Analysis

A. Parameters Configuration

We will compare our scheme with MRM (in Cyberwalk [6]) and FLoD [10] due to the following two reasons: 1) among current scene replacement policies [11], [10], [6], MRM is a progressive replacement mechanism, not the complete removal of whole objects, so it is more similar to PSRM; 2) among existing P2P-DVE designs (e.g., HyperVerse, LODDT, FLoD), our assumed scenario is more similar to FLoD, and we do not know the specifics of LODDT and HyperVerse. We thus compare the three mechanisms, denoted as P2P-MRM, FLoD, and PSRM, respectively.

We build a large scale P2P-DVE on a simulated experimental platform from an open source software [18]. The experimental WebVR world is cached in a server initially, all 3D objects are set randomly in terms of the model data volume and positions, assuming that each object has been processed by PM. In our simulation, we follow the two viewer movement behaviors utilized in FLoD: random way-point (RW) and clustering walk (CW). To perform the simulations more realistically, some new configurations are given in addition to the experimental conditions set by FLoD:

1) FLoD did not consider network latency among different nodes. However, viewers can be far apart from each other geographically. Here, network latency among different viewers are randomly assigned from 100ms to 1500ms.
2) Viewers in FLoD are allowed to start the walkthrough only when 99% of the scene data in their initial AOI has been downloaded. Since users often have no patience to wait, we cancel this strong restriction.

TABLE I: Simulation Parameters

VE dimension (units)	5000*5000
Cell width (units)	50*50
Object size (KB)	20-50
Object Number	10000
Base mesh portion in whole model (%)	10
PM increment size (Bytes)	50
AOI radius	75
AOI Neighbor connection limit	20
Number of nodes	500, 1000
Movement units / Timestep	1
Timestep / second	10
Timesteps simulated	3000
Server Upload Bandwidth	10MB/s
Client Download Bandwidth	64KB/s
Client Upload Bandwidth	32KB/s

Detailed simulation parameters are shown in Table I. Assume that the total data volume of the WebVR scene is M, the radius of viewer AOI is R, the length and width of whole WebVR world are L and W respectively. The average data volume within the AOI (AOI_{avg}) can be computed as: $AOI_{avg} = M\pi R^2/LW$. We define the *cache ratio* as the ratio of users' local cache capacity to AOI_{avg}. It is clear that the larger the cache, the fewer replacements, so the effectiveness of replacement mechanism cannot be shown thoroughly. We thus perform experiments under small caches.

B. Performance Metrics

The following metrics are used for the evaluation:

Fill Ratio, FR: When the viewer moves to a new position, the ratio between the available scene on clients (downloaded) and the visible scene (should be downloaded). Note that we use the *optimal resolution* as defined in [6] as the divisor (the total amount of scene data needed for this view) instead of all the scene content in AOI.

Base Latency, BL: Time between the initial query of a base mesh and when the base mesh is downloaded. Once the base mesh is available, a viewer could start a meaningful walk.

Requests to Server, RS: The total number of piece downloading requests that are sent to the server per step. Since client nodes may not fulfill all scene downloading services, some requests are sent to the server for processing.

C. Detailed Results

1) **Fill Ratio:** We take several measurements (when cache ratios are 3, 2, 1, 0.5 and 0.25, respectively) to show how FR changes with different cache capacities. Fig. 5 shows a general tendency that when the cache ratio drops from 3 to 2, there are no linear decline of FR, which suggests that a higher cache

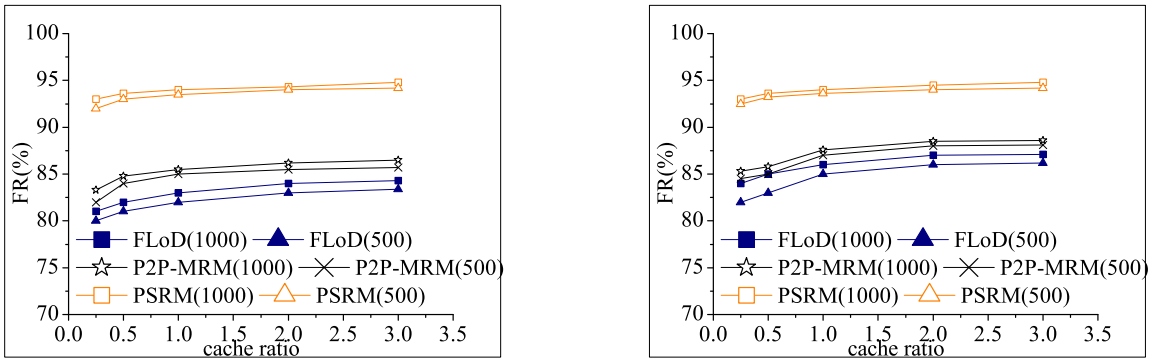

Fig. 5: Effect of cache ratio on fill ratio (the left is RW and the right is CW)

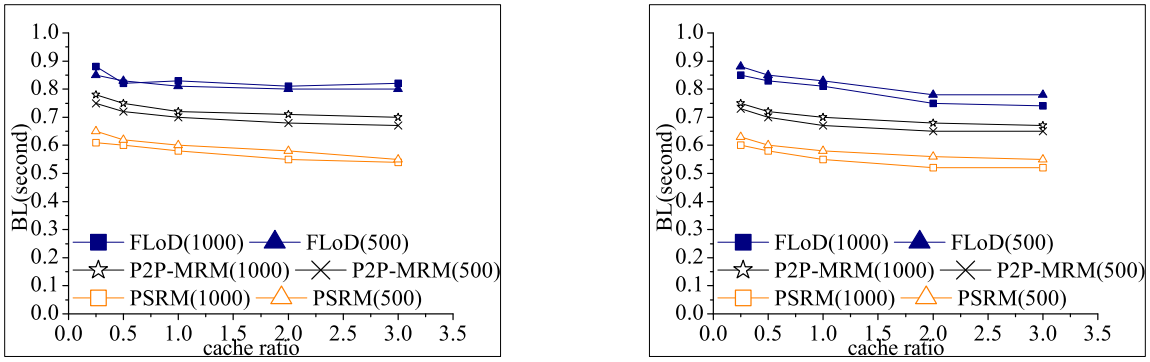

Fig. 6: Effect of cache ratio on base latency (the left is RW and the right is CW)

ratio does not lead to higher FR. This might be because the bandwidth of clients is limited so that a user cannot obtain more scenes in a short time.

As shown in Fig. 5, with 1000 nodes and the cache ratio dropping from 2 to 1, the FR in FLoD is reduced from 84% to 82% with random way-point (RW) and the FR in FLoD is reduced from 86% to 85% with cluster walk (CW). If the cache ratio is only 0.25, the FR in FLoD with 1000 nodes are reduced to 81% with RW and 83% with CW, respectively. The FR of FLoD for 500 nodes is similar to the case of 1000 nodes. There is an obvious decline of FR in P2P-MRM too, when the cache ratio drops from 2 to 1, regardless of node size and movement models. Note that because our definition of fill ratio is based on the content needed for *optimal resolution* [6] as the divisor, it is still possible to retain a relatively high fill ratio (which is more meaningful from a visual perspective) even when cache ratio is low.

However, with 1000 nodes and the cache ratio dropping from 2 to 1, the FR in PSRM is always higher than 92% regardless of the movement models. When the cache ratio is only 0.25, the FR in PSRM drops slightly, but still maintains at a high level. Similarly, the FR of PSRM for 500 nodes are like that in the case of 1000 nodes.

The experimental results have shown the advantage of PSRM, which enables a viewer to observe more visible scenes instantly in a small-cache condition. Such an advantage of PSRM might be meaningful for thin clients like mobile phones and PDAs. Different with FLoD and MRM, which just consider the visual factors when removing objects, PSRM is more suitable for P2P-DVE by also considering another important factor (i.e., the potential relevance degree on AOI neighbors). With help of the PSRM, those frequently visited scenes could be cached as many as possible. In some sense, PSRM avoids the repetitive download of existing scenes to save bandwidth for the download of new scenes.

2) **Base Latency:** As shown in Fig. 6, no matter how the node size and cache ratio changes, the BL of FLoD is basically more than 0.8 seconds with both RW and CW; and the BL of P2P-MRM is slightly less than that of FLoD.

However, regardless the node size, movement models, or cache ratios, the BL of PSRM is always less than 0.7s, which is lower than both FLoD and P2P-MRM. We can understand the results since P2P-MRM and PSRM always postpone to remove the based meshes as much as possible, each viewer thus preserves as many base meshes as possible in its local cache, therefore, download requests of base meshes from AOI neighbors could be answered more successfully. P2P-MRM and PSRM are different with FLoD, which needs to transfer some base mesh requests to other sources besides the AOI neighbors, which causes longer base latency.

3) **Requests to Server:** Table II and Table III show that under both RW and CW movements, the RS in FLoD is more than 500 requests regardless of the cache ratio or node size. Under both RW and CW, the RS in P2P-MRM is slightly less than 500 requests regardless of cache ratio and node size. However, we see that regardless of the cache ratio and node

TABLE II: Requests to server per step in RW

cache ratio	FLoD-500	P2P-MRM500	PSRM-500	FLoD-1000	P2P-MRM1000	PSRM-1000
0.25	549	496	114	543	497	112
0.5	544	499	106	541	495	109
1	542	497	103	546	496	108
2	545	495	104	545	492	105
3	544	494	105	545	493	104

TABLE III: Requests to server per step in CW

cache ratio	FLoD-500	P2P-MRM500	PSRM-500	FLoD-1000	P2P-MRM1000	PSRM-1000
0.25	540	492	112	538	491	113
0.5	543	490	104	536	486	105
1	539	496	101	535	485	104
2	534	491	102	532	486	101
3	535	493	101	530	487	102

size, the RS of PSRM is always at about 100 requests, which is far less than that of FLoD and P2P-MRM.

Reasons for the above can be explained as: by considering the potential relevance degree on AOI neighbors of objects, PSRM enables nodes to preserve more of the scenes which are visited frequently by their AOI neighbors, most of the piece downloading requests could thus be answered immediately with higher hit rates than that of FLoD and P2P-MRM.

VII. Conclusions

In large-scale P2P-DVEs that demand scene streaming, to resolve the contradiction between the massive scene content at servers and the small cache capacities at clients, a new progressive scene replacement mechanism called PSRM is proposed in this paper. Unlike conventional object replacement mechanisms, which only consider visual factors and replace objects in whole, we consider the potential influence brought to a viewer's AOI neighbors when objects are removed, and policies that replace each 3D object progressively instead of wholly. Experiments have shown that PSRM outperforms current scene replacement mechanisms for P2P-DVEs, in visual quality (shown by fill ratio), responsiveness (shown by base latency), and resource usage (shown by requests to server).

Our research has just found a way to the real-time walkthrough of huge WebVR world on clients with small cache, leaving some questions still left unsolved. For example, how to determine the optimal weights of each factor of the preservation degree is an open problem. Another interesting question is how clients should interact with content servers collaboratively, to achieve a balanced performance for heavy content demand. Finally, the simulations can be improved with more realistic user traces and bandwidth distributions.

Acknowledgements

The authors would like to appreciate all anonymous reviewers for their insightful comments and constructive suggestions to polish this paper. The research was supported by the Key Breakthrough of Research & Technology Grant of Shanghai Science and Technology Committee (Grant No: 08511501000).

References

[1] S. Singhal and M. Zyda, *Networked Virtual Environments: Design and Implementation*. Addison-Wesley Professional, 1999.

[2] K. P. Beier, "Web-based virtual reality in design and manufacturing applications," in *Proceedings of COMPIT'00*. IEEE, 2000, pp. 191–194.

[3] F. Li, R. W. H. Lau, D. Kills, and L. Li, "Game-on-demand: An online game engine based on geometry streaming," *ACM Transactions on Multimedia Computing, Communications and Applications(to appear)*.

[4] M. R. Macedonia, M. J. Zyda, D. R. Pratt, D. P. Brutzman, and P. T. Barham, "Exploiting reality with multicast groups: a network architecture for large-scale virtual environments," in *Proceedings of VRAIS'95*. IEEE, 1995, pp. 2–10.

[5] http://secondlife.com.

[6] J. Chim, R. W. H. Lau, H. V. Leong, and A. Si, "Cyberwalk: A web-based distributed virtual walkthrough environments," *IEEE Transactions on Multimedia*, vol. 5, no. 4, pp. 503–515, 2003.

[7] J. Keller and G. Simon, "Solipsis: a massively multi-participant virtual world," in *Proceedings of PDPTA'03*, 2003, pp. 262–268.

[8] J. Botev, A. Hohfeld, H. Schloss, I. Scholtes, P. Sturm, and M. Esch, "The hyperverse- concepts for a federated and torrent-based 3d web," *International Journal of Advanced Media and Communication*, vol. 2, no. 4, pp. 122–128, 2008.

[9] J. Royan, P. Gioia, R. Cavagna, and C. Bouville, "Network-based visualization of 3d landscapes and city models," *IEEE Computer Graphics and Application*, vol. 27, no. 6, pp. 70–79, 2007.

[10] S. Y. Hu, T. H. Huang, S. C. Chang, W. L. Sung, J. R. Jiang, and B. Y. Chen, "Flod: A framework for peer-to-peer 3d streaming," in *Proceedings of INFOCOM'08*. IEEE, 2008, pp. 2047–2055.

[11] G. V. Popescu and C. F. Codella, "An architecture for qos data replication in network virtual environments," in *Proceedings of Virtual Reality'02*. IEEE, 2002, pp. 41–48.

[12] J. Y. Jia, P. Wang, S. Wang, and Y. Wang, "An integer incremental aoi algorithm for progressive downloading large scale vrml/x3d environments," in *LNCS*, 2007, pp. 711–722.

[13] H. Hoppe, "Progressive meshes," in *Proceedings of SIGGRAPH'96*. ACM, 1996, pp. 99–108.

[14] A. Silberschatz, H. Korth, and S. Sudarshan, *Database System Concepts*. McGraw-Hill, 1996.

[15] M. Franklin, M. Carey, and M. Livny, "Global memory management in client-server dbms architectures," in *Proceedings of VLDB'92*, 1992, pp. 596–609.

[16] A. Si and H. V. Leong, "Adaptive caching and refreshing in mobile databases," *Personal Technologies*, vol. 1, no. 3, pp. 156–170, 1997.

[17] T. Y. Li and W. H. Hsu, "A data management scheme for effective walkthrough in large-scale virtual environments," *Visual Computer*, vol. 20, no. 11, pp. 624–634, 2004.

[18] http://ascend.sourceforge.net.