Towards Multi-Site Collaboration in Tele-Immersive Environments

Wanmin Wu†, Zhenyu Yang†, Klara Nahrstedt†, Gregorij Kurillo‡, Ruzena Bajcsy‡

†Department of Computer Science
University of Illinois at Urbana-Champaign
201 N. Goodwin Ave., Urbana, IL 61801, USA
{wwu23,zyang2,klara}@uiuc.edu

‡Department of EECS
University of California, Berkeley
253 Cory Hall, Berkeley, CA 94720, USA
{gregorij,bajcsy}@eecs.berkeley.edu

ABSTRACT

Tele-immersion is emerging as a new medium that creates 3D photorealistic, immersive, and interactive experience between geographically dispersed users. However, most existing tele-immersive systems can only support two-way collaboration. In this paper we propose a multi-layer framework and a new data dissemination protocol to support *multi-site* collaboration. The problem context is unique as multiple remote sites participate in an interactive tele-immersive session, where each site has multiple correlated 3D video streams to send (later referred as *multi-stream/multi-site* environments). The key challenge is to disseminate such large number of 3D live video streams among these sites subject to the bandwidth and latency constraints while satisfying QoS guarantees in visual quality. Among our findings is that the simple randomized algorithm outperforms many other static algorithms in the unique context. Moreover, the streams generated from one site have high semantic correlation, because often cameras at one site are shooting the same scene, only from different angles. We exploit the stream correlation in the multicast protocol to minimize the *level of loss*.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Distributed applications; H.5.1 [**Multimedia Information Systems**]: Video

General Terms

Design, Experimentation, Measurement

Keywords

Tele-immersion, Multicast, Overlay

1. INTRODUCTION

Tele-immersion is receiving increasing attention today as a new way of collaboration. The strength of tele-immersion

MM'07, September 23–28, 2007, Augsburg, Bavaria, Germany.

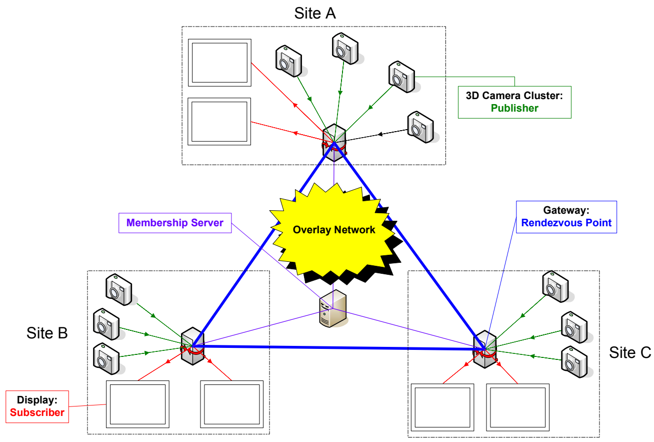

Figure 1: The tele-immersive system architecture from the publish-subscribe perspective.

is that with the combination of real-time 3D reconstruction, high-bandwidth networking, and efficient 3D rendering techniques, it is able to immerse distant users into a shared virtual space such that they have a strong sense of shared presence. Over the last few years, tele-immersion has shown great potentials in a wide range of applications such as video-conferencing, distance learning of physical activities, collaborative art performance, and medical consultation [11, 6, 9, 7]. However, most existing tele-immersive systems can only support two-party communication due to the limits of computing resources and networking bandwidth. We propose a multi-layer framework and a multi-stream/multi-site data dissemination protocol that enables users at multiple sites to participate in a tele-immersive session simultaneously at interactive rate.

Supporting such multi-site participation is challenging in the time-critical and bandwidth-demanding tele-immersive environments. Conventional 2D video-mediated remote collaboration systems (e.g., video-conferencing) adopt an "all-to-all" communication scheme, where the stream from one site is sent to all other sites [3]. We argue that this is infeasible in tele-immersive environments, because (1) the stream sent by each tele-immersive site is in 3D video format, which consumes much more bandwidth than a 2D stream, and (2) worse, each site often has more than one 3D stream to send, with different streams showing different viewpoints. As an example, the experiments described in [11] involved two sites thousands of miles apart, with each sending about ten 3D streams to the other. As the number of streams grows, the "all-to-all" scheme has to be abandoned.

We present a multi-layer framework executed at each site that leverages user preference on streams/views. This way the tele-immersive system is embedded into the publish-subscribe (pub-sub) paradigm. Figure 1 shows the system architecture from this perspective. First, the 3D camera clusters, each capturing the scene and producing a 3D stream using real-time 3D reconstruction, become the *publishers*. Second, several heterogeneous displays that render the 3D video at each sites become the *subscribers*. Users configure the preferred view/streams to receive from other sites for each display. Finally, a service gateway is introduced as the *rendezvous point (RP)* (or core) at each site. The role of RP is very important because it collects 3D video streams from local cameras and other sites, and disseminates them to the displays as requested. All RPs form an overlay network to disseminate the 3D video data among each other. Therefore, *each site collapsed into a RP node which is the aggregate point of the streams on the overlay network*. From thereafter, we use nodes and RPs interchangeably.

In the multi-stream/multi-site environment, every node acts as both sender and receiver of *multiple streams* simultaneously. In other words, the multi-site tele-immersive overlay network has a highly *dense graph*. We formalize the data dissemination problem as a multicast forest construction problem. Several algorithms are evaluated by simulation. We find that randomized algorithms work well in this context. Furthermore, the streams generated from one node have high semantic correlation, because often cameras at one site are shooting the same scene, only from different angles. We exploit the correlation in the multicast protocol to minimize the *level of loss*.

2. FRAMEWORK

2.1 System Model

The graph consisting of all RPs can be modeled as a complete virtual graph of N nodes. Each node i has an *out-degree bound*, B_i, and an *in-degree bound*, I_i, which represent the bandwidth constraints. Each node has a set of *out streams*, S_i, to send. The streams sent from one site have high semantic correlation between each other. The *view of a camera* is given by the normal vector $\overrightarrow{O}$ of its image plane, and the *view of a stream* is the view of the camera that produces it. The *correlation* between streams s_i^p and s_i^q sent from node i is the dot product of the two view vectors.

$$C(s_i^p, s_i^q) = \overrightarrow{O}(s_i^p) \cdot \overrightarrow{O}(s_i^q) \text{ where } s_i^p, s_i^q \in S_i \qquad (1)$$

Note that although the streams from one site have semantic correlation, we assume they are *independently* coded and transmitted.

In the following sections, we describe the multi-layer framework collaterally with the session setup protocol. The multi-layer framework consists of three layers: *front end*, *stream selection*, and *pub-sub substrate*.

2.2 Multi-layer Framework

The *front end* layer provides a user interface for stream or viewpoint selection. Users at each site can *visually* select

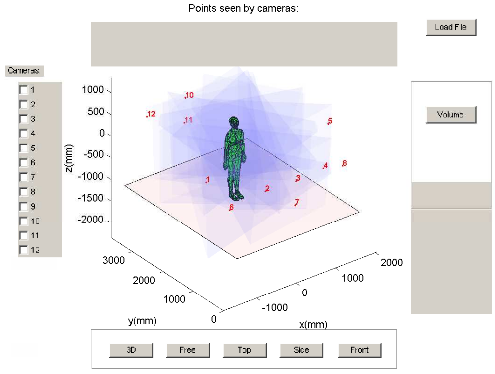

Figure 2: GUI for stream/viewpoint selection.

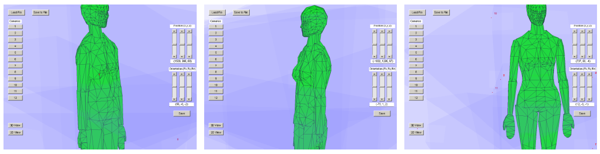

Figure 3: Simulated views

their favorite streams in a GUI as shown in Figure 2. Their subscription requests, each represented by a *streamID* of the stream that they wish to receive, are collected into a list and passed to the underlying *stream selection* layer. If the users feel selecting streams manually is too tedious, an alternative option is to only specify a preferred *view*, $\overrightarrow{O}(k)$, for site k they wish to subscribe and the maximum number of streams (f_i^k) they wish to receive from k (assuming the users are at site i). Figure 3 shows some examples of simulated views of a site. In this case, pairs of $\{O(k), f_i^k\}$ are passed to the layer below.

The *stream selection* layer processes the subscription requests, and generates a final list of requests to send to the membership server. The requests can take the format of either a *streamID* list or pairs of $\{O(k), f_i^k\}$. In the former case, the total number of requests made by the upper layer (R_i) is checked against the in-degree bound (I_i) of this node. If the number of requests satisfy the bound ($R_i \leq I_i$), the list is directly sent to the membership server in a *REQ* message. If the number of requests is larger than the bound by e (i.e., $R_i - B_i = e$), the last e requests are pruned, and the shortened list is sent in a *REQ* message. In the latter case, assuming $t = minimum(\sum_{k=1}^{N} f_i^k, B_i)$, the t most correlated streams to the user specified *view* are selected and sent to the server in a *REQ* message.

The *pub-sub substrate* layer is on the overlay level where each RP sends/receives 3D video data to/from each other. All RPs form an overlay network to disseminate the 3D video data among each other. The structure of the overlay is centrally determined by the membership server. The membership server replies each *REQ* request with a *REP* message containing list of (*Parent, streamID*) pairs where the *Parent* is the node the subscriber should contact to get stream *streamID*.

3. OVERLAY CONSTRUCTION

One key component in the algorithm is the construction of the *multicast forest* overlay. We stress that the tele-immersive environment has a graph with high density, because each node acts as both sender and receiver of multiple 3D streams. In this section, we study several algorithms for overlay construction in the unique context. The challenge is to satisfy bandwidth constraint at each node, ensure end-to-end latency between any pair of nodes, while accommodating as many requests as possible. The forest construction problem in multi-site tele-immersion can be formulated as follows.

Forest Construction Problem. Given a completely connected graph $G = (V, E)$, an in-degree bound $d_{in\_max}(v) \in \mathbb{N}$, and an out-degree bound $d_{out\_max}(v) \in \mathbb{N}$, for each node $v \in V$, a cost $c(e) \in \mathbb{Z}^+$ for each edge $e \in E$, and a set of subgraphs $G'_{subsets} = \{G'_i \mid G'_i \subseteq G\}$ each with a source node s_i, find a *spanning forest*, $F = \{T_i \mid 1 \leq i \leq H\}$ where H is the number of trees, with each tree T_i being a spanning tree that connects the source s_i with a subset of the other nodes ($G''_i \subseteq G'_i - s_i$), such that the total number of excluded nodes, $\sum_i |G'_i - G''_i|$, is minimized, subject to the constraints that for all $v \in T_i$ ($1 \leq i \leq H$), $d_{in}(v) < d_{in\_max}(v)$ and $d_{out}(v) < d_{out\_max}(v)$, and $cost(v, s_i) < Bound_{cost}$, where $cost(v, s_i)$ is the cost from the source s_i to node v and $Bound_{cost}$ is the cost bound.

Wang *et al.* [8] has proven that the problem of finding a solution subject to two or more constraints in any combination is NP-complete. Next we study four heuristic algorithms for the forest construction problem.

Largest Tree First (LTF). The membership server sorts all multicast groups according to the number of subscribers in them. It then computes the spanning trees one by one from the largest tree to the smallest one. The rationale is that even if the last few trees cannot be constructed because of saturation, the number of rejected requests should be small because we are left with the smallest trees.

Smallest Tree First (STF). The reversed algorithm of LTF for comparison study. The hypothesis is that the rejection ratio of LTF should be smaller than that of STF.

Most Difficult Tree First (MDTF). The server evaluates how difficult to construct a tree in terms of the aggregate *forwarding capacity* of a tree. A node i's forwarding capacity, w_i, can be defined as $w_i = B_i - m_i$, where m_i is the number of out streams of i that is subscribed by at least one node. The forwarding capacity of a tree for multicast group G, W_T, can then be defined as $W_T = \sum_{i=1}^{|T|} w_i$. The intuition is that the larger this value is, the easier should it be to construct the tree generally because new nodes are easier to accommodate with a tree with large aggregate forwarding capacity.

Random Join (RJ). Instead of building trees one by one, the server randomly picks a node i and one of its subscription request (to join stream s), then joins i into the existing tree T_s.

All four algorithms construct a tree *incrementally*, i.e., join the requesting node into the tree one by one. To join a node i into a tree T, the algorithm looks for a node j which has free out-degree and maximum left forwarding degree of all nodes in T, subject to the constraint that the cost from i to the source of T if connected to j is smaller than a bound. This `join_node` algorithm always seeks to achieve load balancing, which is essential in such a dense graph as a tele-immersive session. A tree might be *saturated*, i.e., there is no eligible node in the tree to serve the requesting node. In this case, incoming requests for this tree are *rejected*.

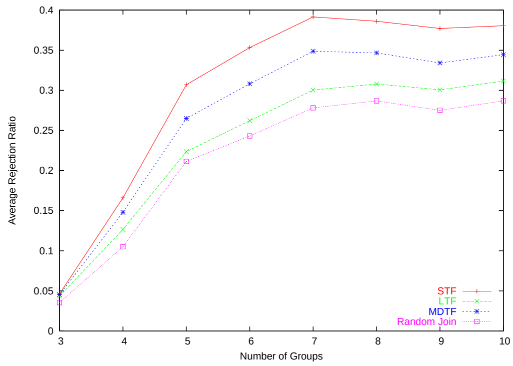

Figure 4: Rejection ratio

4. EXPERIMENTS

We compare the performance of the four algorithms by simulation with 3-10 nodes. Each node has 1-20 streams (randomly distributed) to send, and the out-degree bound is also set to be randomly between 10-30. Two hundred random samples are generated specifying several parameters, e.g., the number of out-streams, out-degree bound, subscription requests, for each site, and the cost (latency) between sites.

Figure 4 shows the average rejection ratios for different algorithms. The data support our hypothesis that *LTF* should perform better than *STF*. However, somewhat surprising is that the simple *Random Join* algorithm actually achieves the lowest rejection ratio. We believe one of the most important reasons is that every tele-immersive node is heavily burdened with serving requests. In tree-wise algorithms, a node is much more likely to be congested in the first few trees if it is the source or a node near the source. This increases the probability of rejection in the construction of latter trees because the node's total bandwidth is shared among different trees. In contrast, the randomized algorithm achieves good balance because it distributes the tasks of request processing among different trees randomly. The likelihood a single node is congested is thus minimized, and the rejection ratio decreased. Assuming the arrival of requests follows a randomized distribution, the advantage of randomized static algorithm actually suggests the potential of dynamic algorithms.

5. CORRELATION

So far, a request is rejected once the tree to join is saturated. However, we note that in tele-immersive environment, the streams generated from one site have high semantic correlation between each other. Often the 3D camera clusters at one site are shooting the same scene, only from different angles. As an example, suppose a site A subscribes to B's four streams, $\mathcal{B}_1, \mathcal{B}_2, \mathcal{B}_3, \mathcal{B}_4$ and C's one stream, $\mathcal{C}_7$. Then losing one of B's stream is less critical than losing the single stream from C.

We describe a modified *RJ* algorithm, called ***Caucasus***, which accounts for stream correlation to minimize the *level of loss*. Suppose v_i^j is the number of streams node i sub-

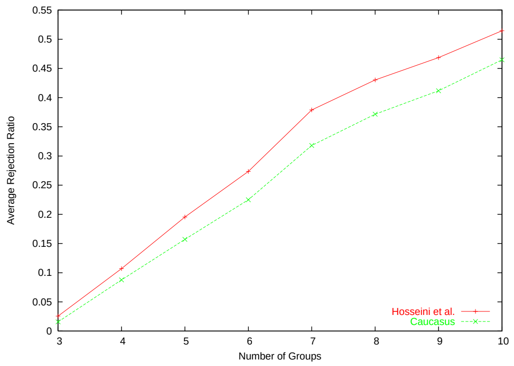

Figure 5: Rejection ratio with correlation taken into account

scribes from node j. We define the *criticality of node i losing a stream $s \in S_j$* as $Q_i^s = 1/v_i^j$. When a request (i, s_j) (node i joining stream s_j) is rejected, the membership server checks the streams that i has received already. If (1) there is a stream s_k with $Q_i^{s_k} \leq Q_i^{s_j}$, and (2) i is a leaf node in tree T_k, and (3) the parent of i in T_k, $p(i)_{T_k}$, has stream s_j, then the edge $p(i)_{T_k} \rightarrow i$ is removed in T_k and a new edge $p(i)_{T_k} \rightarrow i$ in tree T_j is added. In other words, i loses s_k instead of s_j. This operation is done with minimal cost, because i was a leaf node in tree T_k so removing the old link would not cause relocation of any other nodes in T_k.

We compare `Caucasus` with a modified single-stream/multi -site `join_node` algorithm (with three important heuristic optimizations) from [5]. We modify their `join_node` algorithm so that it works in the multi-stream/multi-site context. More specifically, each stream is supposed to come from a unique virtual node, so is each request. The bandwidth constraint (degree bound) is thus evenly shared by the virtual node. Suppose v_i^j is the number of requests node i made to node j, among which r_i^j is rejected, and N is number of nodes. The definition of *rejection ratio* is modified as $U_r' = \sum_{i=1}^{N} r_i^j/(v_i^j \times (N-1))$ to account for stream correlation. As shown in Figure 5, our algorithm further reduces the level of loss compared with one of the best dynamic algorithms by considering stream correlation.

6. RELATED WORK

Raskar *et al.* proposed their vision of “Office of the Future” in 1998 and stimulated the research on tele-immersion. Virtual Auditorium [2] and Digital Amphitheater [4] were among the early efforts that aimed to provide immersive collaboration. However, in these applications each site only transmitted a single 2D video stream and the “all-to-all” communication scheme was adopted. In Coliseum [1], each site sends only 2D frames constructed by visual hull techniques with no depth information associated and also has only one stream to send which simplifies the problem. We were one of the first to build a multi-stream (per site) 3D photorealistic tele-immersive system which features multiple true 3D streams from each site and multiple displays that render the video from different perspectives [10].

7. CONCLUSION

In this paper, we present our study in multi-stream/multi-site data dissemination problem in tele-immersive environments. We propose a multi-layer framework and a new dissemination protocol to support such multi-site collaboration. Among our findings is that simple randomized algorithms work well in the unique context.

8. ACKNOWLEDGEMENT

We thank Dr. Indranil Gupta, Jin Liang and the anonymous reviewers for their helpful comments. This work was supported by the National Science Foundation under Grant CNS 05-20182 and SCI 05-49242. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

9. REFERENCES

[1] H. Baker, N. Bhatti, D. Tanguay, I. Sobel, D. Gelb, M. Goss, W. Culbertson, and T. Malzbender. The coliseum immersive teleconferencing system. *ACM Trans. on Multimedia Computing, Communications, and Applications*, 2005.

[2] M. Chen. Design of a virtual auditorium. In *Proceedings of the ninth ACM international conference on Multimedia*, 2001.

[3] K. E. Finn, A. Sellen, S. Wilbur, K. Finn, A. J. Sellen, and S. B. Wilbur. *Video-mediated Communication.* Lawrence Erlbaum Associates, 1997.

[4] L. Gharai, C. Perkins, R. Riley, and A. Mankin. Large scale video conferencing: A digital amphitheater. In *Proc. 8th International Conference on Distributed Multimedia Systems*, San Francisco, CA, 2002.

[5] M. Hosseini and N. D. Georganas. Design of a multi-sender 3d videoconferencing application over an end system multicast protocol. In *Proc. of ACM Multimedia 2003*, 2003.

[6] S. Jung and R. Bajcsy. Learning physical activities in immersive virtual environments. In *Proceedings of the 4th IEEE Int. Conference on Computer Vision Systems*, New York City, January 5-7th, 2006.

[7] R. Raskar, G. Welch, M. Cutts, A. Lake, L. Stesin, and H. Fuchs. The office of the future: A unified approach to image-based modeling and spatially immersive displays. In *Proc. of ACM SIGGRAPH 98*, Orlando, FL, 1998.

[8] Z. Wang and J. Crowcroft. Qos routing for supporting resource reservation. In *IEEE Journal on Selected areas in Communications*, Sep. 1996.

[9] G. Welch, D. H. Sonnenwald, K. Mayer-Patel, R. Yang, A. State, H. Towles, B. Cairns, and H. Fuchs. Remote 3d medical consultation. In *Proc. of the 2nd International Conference on Broadband Networks*, 2005.

[10] Z. Yang, K. Nahrstedt, Y. Cui, B. Yu, J. Liang, S. hack Jung, and R. Bajcsy. TEEVE: The next generation architecture for tele-immersive environments. In *Proc. of the 7th IEEE International Symposium on Multimedia (ISM'05)*, Irvine, CA, 2005.

[11] Z. Yang, B. Yu, W. Wu, R. Diankov, and R. Bajcsy. A study of collaborative dancing in tele-immersive environment. In *Proc. of the 8th IEEE International Symposium on Multimedia (ISM'06)*, San Diego, CA, 2006.