Advances in Multimedia

Multimedia Networking

Guest Editors: Guobin (Jacky) Shen and Jianfei Cai

Multimedia Networking

Advances in Multimedia

Multimedia Networking

Guest Editors: Guobin (Jacky) Shen and Jianfei Cai

Contents

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2007, Article ID 97262, 1 page
doi:10.1155/2007/97262

Editorial
Multimedia Networking

Guobin (Jacky) Shen[1] and Jianfei Cai[2]

[1] *Microsoft Research Asia, Beijing 100080, China*
[2] *School of Computer Engineering, Nanyang Technological University, Singapore 639798*

Received 29 December 2006; Accepted 29 December 2006

Multimedia contents (animation, audio, and video) are becoming increasingly popular on the Internet and are being accessed by a variety of networked devices through either wired or wireless links. A large number of distributed multimedia applications have been created, including Internet telephony, Internet videoconferencing, on-demand streaming or broadcasting, IPTV, distance learning, entertainment and gaming, multimedia messaging, and so forth. Streaming real-time and on-demand audio and video over the Internet, local and wide area wireless networks have become a reality and will soon become a mainstream means of communication. To accelerate the adoption of these new emerging applications, a number of important issues must be addressed such as the architecture and design of multimedia communication systems, the quality-of-service (QoS) provisioning, the network and content security, and so forth. In this special issue on multimedia networking, we have invited a few papers that address such issues.

The first paper of this special issue addresses the system architecture and mobility management for mobile immersive communications, for both fixed and mobile clients, based on a distributed proxy model. Three possible methods for updating proxy assignments in response to mobility were proposed and their performances are compared. The second paper presents the study on the quality metric that integrates both the geometry resolution and realistic texture resolution, which is an important factor in the design of effective interactive online 3D systems. The third paper is on the efficient delivery of interactive multiview video, leveraging IP multicast, which can support a large number of users while keeping a high degree of interactivity and consuming low bandwidth.

The fourth paper of this special issue presents a fully distributed protocol, scalable island multicast, that effectively integrates IP multicast and application layer multicast (ALM) for media streaming, which brings in lower end-to-end delay lower link stress and lower resource usage than traditional ALM protocols. The fifth paper describes a video surveillance system based on JPEG2000 that allows for transmission of the scene over limited bandwidth networks. The core of the system is a low-complexity transcoding technique that adapts the quality and resolution of the scene based on the available bandwidth with an adaptive rate control algorithm. The two subsequent papers address the bandwidth estimation for wireless streaming using the information from the lower layer of the protocol stack, and the technique to mitigate the impact of the imprecise rate estimation by scheduling with a conservative delay, respectively.

The eighth paper proposes to use signal-to-silence ratio (SSR) as an indication to the channel state information, which in return is used in a cross-layer protocol. An active queue management technique was proposed to differentiate corrupted packets. Such side-information-(SI-) aware processing provides significant performance gain over SI-unaware schemes. The final paper of this special issue is more forward-looking. It presents a new scheme that integrates multiple-description coding (MDC), error-resilient video coding, and unequal error protection with a hybrid space-time coding structure for robust video transmission over the MIMO-OFDM system.

Guobin (Jacky) Shen
Jianfei Cai

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2007, Article ID 53674, 7 pages
doi:10.1155/2007/53674

Research Article

System Architecture and Mobility Management for Mobile Immersive Communications

Mehran Dowlatshahi and Farzad Safaei

Telecommunications and Information Technology Research Institute, University of Wollongong, Level 1, Building 4, Northfields Avenue, Wollongong, NSW 2522, Australia

Received 29 October 2006; Accepted 19 December 2006

Recommended by Guobin (Jacky) Shen

We propose a system design for delivery of immersive communications to mobile wireless devices based on a distributed proxy model. It is demonstrated that this architecture addresses key technical challenges for the delivery of these services, that is, constraints on link capacity and power consumption in mobile devices. However, additional complexity is introduced with respect to application layer mobility management. The paper proposes three possible methods for updating proxy assignments in response to mobility management and compares the performance of these methods.

1. INTRODUCTION

The new generations of wireless technology are poised to relieve the user from the severe bandwidth constraints of the earlier systems, paving the way for true multimedia communications. Most current services over the Internet, such as Voice over IP (VoIP) telephony, are based on a single point-to-point communication channel between the end users. Apart from increasing the wireless access bandwidth, a key technical challenge for these services is to develop fast handoff schemes at layer 2 and layer 3 to enable uninterrupted voice and video communication in mobile scenarios [1].

We are interested in the next generation of communication services that facilitate natural *multiparty* interaction and can be called immersive communications. In immersive communications, the visual and aural scenes of each user create a sense of being in the presence of a group of people. One possible instance where immersive communications can be useful is within the networked (or distributed) virtual environments (NVEs). NVEs are likely to form the basis of a large class of applications for education, entertainment, and collaboration. If complemented with immersive communications, these environments can transform the Internet from a medium primarily used for search and retrieval of information to one that facilitates human interaction, collaboration, and play. There has been a significant increase in popularity of networked virtual environments (NVE) in recent years. For example, reliable estimates indicate that by 2009 more than 230 million people will be playing multiplayer network games and, in particular, mobile games show significant growth [2].

Natural human communication within an NVE requires creating a suitable multimedia scene (voice, video, gestures, and haptics) for each participant to mimic the real world sensory information of being in the presence of a group or a crowd. The audio scene, for example, must include the voices of all avatars in the participant's hearing range, spatially placed at a suitable distance based on the participant's perspective. Unlike the current person-to-person communication services, which are characterized by more or less static point-to-point traffic flows, immersive communication involves a myriad of point-to-multipoint flows with highly dynamic changes in their connectivity arrangements. For example, the voice of each participant has to be included in the audio scene of everyone within the audible range of this voice. Likewise, other multimedia content (visual, gesture, and haptics information) sourced from a given participant should reach everyone who is "interested" in this information. Conceptually, one might view several parallel overlay multicast flows from the source to others within the area of interest (current IP-based multicast mechanisms due to lack of a fast dynamic reconfiguration capability does not seem

to be appropriate for multicasting of media streams to frequently changing groups of clients). This "area of interest" may differ for different types of media. Voice, for example, could propagate through walls while visual information does not. As avatars move within the virtual environment, these overlay multicast trees must undergo change. The immersive communications, therefore, is characterized by a large number of overlay multicast flows that are subject to rapid reconfiguration.

Wireless access to multimedia immersive communications presents additional challenges.

(A) The *access bandwidth* is more likely to be a bottleneck as the wireless link capacity is unlikely to match that of wired access in near future. This would be especially important for the downstream bandwidth where the required number of streams received would depend on how "crowded" the participant's surrounding environment is.
(B) The *energy resources in wireless devices* for the battery operated mobile devices are likely to be scarce. Hence, it would be desirable to minimize transmission by mobile clients.
(C) The *mobility* of wireless devices across networks will create new challenges for mobility management and as will be shown require new functions in addition to layer 2 and layer 3 handovers to control latency.

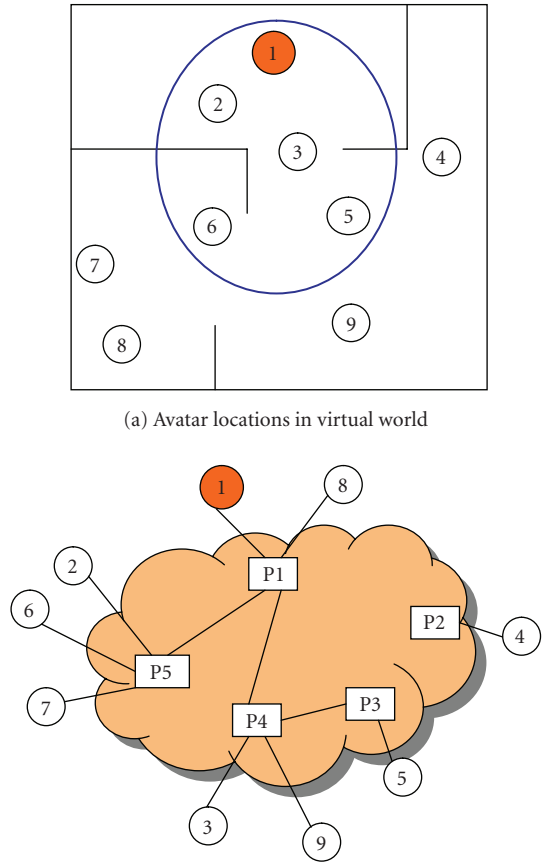
One approach is to tackle these problems head-on by developing higher speed wireless links and more powerful batteries. The second approach, which is the subject of this article, is to perform certain functions in the fixed infrastructure to relieve the wireless devices from excessive transmissions of media streams. Naturally, these two approaches are not mutually exclusive and can work together to offer the best outcome based on a given wireless technology.

In our earlier works [3, 4] a distributed proxy model for media streaming to fixed clients of virtual environments has been proposed. There, in order to minimize end-to-end delay, every fixed client is assigned to its topologically closest proxy. Here the same distributed proxy architecture for media streaming to mobile nodes has been adopted. Network layer mobility management (e.g., [1, 5]) may transparently change a mobile client's network at any instant. In this paper it will be shown that in a distributed proxy model mobility management at the network layer without mobility management at the application layer is likely to deteriorate end-to-end delay performance. Main focus of this paper is therefore *application layer* mobility management techniques for the distributed proxy model.

The rest of this paper is organized as follows. Section 2 describes our proposed system architecture. Section 3 provides a possible solution for managing physical mobility and simulation results on its effectiveness. Section 4 presents the concluding remarks.

2. SYSTEM ARCHITECTURE

A conceptually simple model for wireless immersive communications is to use a peer-to-peer model for transmission of multimedia content between the participants. To use a concrete example, let us consider immersive voice communications for the following discussions. In a peer-to-peer model, each client captures the voice of its user and must identify the subset of participants who would be interested in this voice stream. The voice stream is then overlay multicast to this subset.



(a) Avatar locations in virtual world

(b) Clients and their proxy servers

Figure 1: Distributed proxy architecture for immersive communication.

While simple, this model has some drawbacks. First, the downstream wireless link is a shared media and would limit the number of flows that can be received by each participant. Second, in addition to their locally captured streams the clients may have to participate in forwarding (transmission) received media stream(s) from other clients towards the clients that need them.

To overcome these difficulties, we propose to use a set of distributed servers—referred to as proxies—to aid in the delivery of multimedia streams to clients. Each proxy is responsible for a group of clients and, in essence, performs the necessary functions of these peers on their behalf.

Figure 1 shows this architecture for a small NVE. Every wireless client is connected to a proxy server. To improve latency, it would be best to connect the client to its closest (in terms of network delay) proxy.

On the upstream side, the client will send its voice packets to its proxy. It is the responsibility of the proxy to overlay multicast this voice stream to other proxies who might need this information for the creation of their clients' audio scenes. This is shown in Figure 1. Proxy P1 receives voice packets from one of its clients (avatar 1). P1 will determine the audible range of this signal by analyzing its loudness and the characteristics of the environment (e.g., presence of sound barriers such as walls). The audible range is shown as a closed area in this figure which includes several avatars, namely, avatars 2–6. P1 will then determine the proxies for avatars 2–6 which happen to be P3, P4, and P5. Avatar 1's voice packets are then overlay multicast to P3, P4, and P5 with P1 as the root of the overlay multicast tree. Similarly, the proxies associated with all other (talking) avatars will create overlay multicasts for the purpose of communicating their client's voices. Consequently, at each instant of time, there will be N active overlay multicast trees between the proxy servers, where N is the number of talking avatars. By talking avatars we mean those who have subscribed to the audio service and have active voice signals at this moment. On the downstream side, the proxy should send the audio streams of relevance to its connected clients.

2.1. Coping with limited wireless link capacity

On the downstream side, the proxy should somehow send the audio streams of relevance to its connected clients without exceeding the wireless link capacity. Let us assume that the downstream link capacity allocated to audio is equal to Kr bits per second (bps) for a given wireless device, where r is the required rate for a constant bit rate mono audio stream (in bps) and K is an integer. The proxy is responsible for ensuring that the amount of voice information sent to this client does not exceed this value regardless of how crowded the avatar's surrounding environment is. We have developed two possible mechanisms for this purpose.

In the first case, the proxy merely performs a simple "filter and forward" operation together with silent suppression at the clients. An important characteristic of immersive communications is the fact that usually only a limited number of talking avatars can be heard in each virtual scene (avatars in the same room, vehicle, etc.) and therefore each avatar will need to receive only certain number of audio streams at any instant. For constant bit rate streams, this number may still exceed the wireless access capacity. However, if voices are modeled as independent on-off audio sources (talk spurts and silence periods), by filtering out the silence periods from each stream, it should be possible to pack more voice streams within the downstream flow. The filtering out of the silence periods can be done at the originating client by running a silence detection algorithm. The client will then only send the "active" audio packets to its assigned proxy. The proxy multicasts these voice packets to all other proxies that require the audio signal from this avatar for any of their respective clients. On the receiving side, each proxy receives a number of voice streams from its attached clients and the other proxies. The proxy has information on the access bandwidth limitation of its clients. It uses a priority-based ordered list of all avatars within the hearing range of each client and sends (at most) Kr bps of the active streams to each client. In this case, the first K high-priority voice streams will always be received but there is also a good chance for packets from further away voices to also get through during the silence periods of these. In effect, these voices experience some "packet loss" but only during the talk spurt periods of the closest avatars. In [6] it has been shown that despite its simplicity, this filter and forward operation can lead to good performance in reasonably crowded spaces even when the access bandwidth limit (value of K) is rather low. For example, for $K = 3$, obviously the first three talking avatars closest to the listener are heard with no packet loss. The packet loss of the fourth talking avatar, if any, would be below 10%. However, the losses only happen in bursts during those moments when 4 or more avatars have simultaneous talk spurts. Whether this loss is subjectively significant requires further study. The impact on listener, who is already receiving three active audio streams could be small. According to the same analytical results, more than 80% of the talk spurts of the fifth talking avatar, if any, will also be heard.

In a second implementation, when the number of avatars in one's hearing range is greater than K, the proxy will group these into K separate clusters and perform a partial audio mixing operation for each cluster. The proxy will also calculate the "center of activity" of each cluster. This is the location of an imaginary audio source from which the cluster mix should emanate. The client will be able to render the audio scene by spatially rendering placing each mixed stream at its center of activity. In this case, the whole audio scene is represented without any loss of voice packets. However, there may be some error in the spatial location of far away voices (see [7] for details).

2.2. Coping with mobility in the virtual world

In the distributed proxy architecture, the wireless devices need not participate in formation and reconfiguration of overlay multicast trees. Instead, each proxy will become the root of overlay multicast for all of the streams sourced by its attached clients and will have to join all those overlay multicast trees associated with streams needed by its clients. Apart from the fact that this will make the task of wireless devices much simpler, using proxies as opposed to peers for formation and reconfiguration of overlay multicasts has another important advantage: the reconfiguration of overlay multicast trees due to movements in the virtual world happens less often.

To illustrate this point consider Figure 1 once again. The movement of avatars will change the composition of crowds and the proximity of avatars to each other within a crowd. Consequently, the list of avatars in one's audible range will change due to the movement of both the speaker and the listeners. This, in turn, may lead to a new multicast tree if any of the proxy leaves are different. Given that proxies are participating in multicast trees on behalf of all their attached clients, this change happens less often than a peer-to-peer

model. For example, in Figure 1 if avatar 2 moves out of the audible range of avatar 1, the multicast tree from P1 will not change because there is still another avatar (6) connected to the same proxy that needs avatar 1's voice stream. In recent years many techniques for construction of overlay multicast trees and networks have been proposed [8–10]. In [3, 4] we have proposed an algorithm for construction of overlay multicast trees that is scalable to a large number of highly dynamic trees and will allow rapid reconfiguration on the time scales which are consistent with movements within a virtual environment.

3. MOBILITY MANAGEMENT

In the previous section we demonstrated that using a distributed proxy architecture can significantly improve scalability and robustness of immersive communication services for wireless nodes. The use of distributed proxies, however, creates an additional complexity-mobility management. This is particularly pertinent if the underlying network topology is hierarchical, which is of course very common. To illustrate this point, consider the network of Figure 2 where a portion of a hierarchical infrastructure comprised of two stub domains interconnected using a transit domain is shown. A wireless client is initially connected to stub domain 1 through one of the routers associated with this domain as its care of address (CoA) router [5].

Let us assume that the client is mobile. Figure 2 shows a case when the mobile node has moved outside the range of its stub domain and connected to a new CoA router. Given timely handovers using layer 2 and layer 3, it should be possible for the wireless node to continue its multimedia communication session. However, the location of its proxy may no longer be suitable. For example, the proxy may be connected to the old stub domain and the communication between the new CoA router and this proxy may have to go through one or more transit domains and experience significant increase in delay (although in terms of geographical distance, the change may not be as significant). In this case, it may be important to reduce the latency by assigning the client to a closer (in terms of network delay) proxy such as the proxy in stub domain 2. The key issue is that a change of stub domain could happen as a result of moderate movement but may lead to significant increase in network delay from the wireless device to its proxy.

Using simulation we have investigated the effect of physical movements on round trip time delay of client to proxy nodes. In our simulation experiments we use a transit-stub topology [11] to simulate a two-layer hierarchical topology. The network consists of six transit domains, each with an average of 10 routers. Each transit router is connected to an average of 3 stub domains, and each stub domain consists of 8 routers. Routers at any of the transit or stub domains have an average of 3 physical links to the network. Each stub domain is assumed to have a single proxy attached to one of its routers. The network is assumed to represent a 5000 by 5000 km geographical area. Note that we are not implying that different domains are owned by different network providers. A hierarchical infrastructure is common for large scale carriers that operate across a vast geographical area (such as USA or Australia).

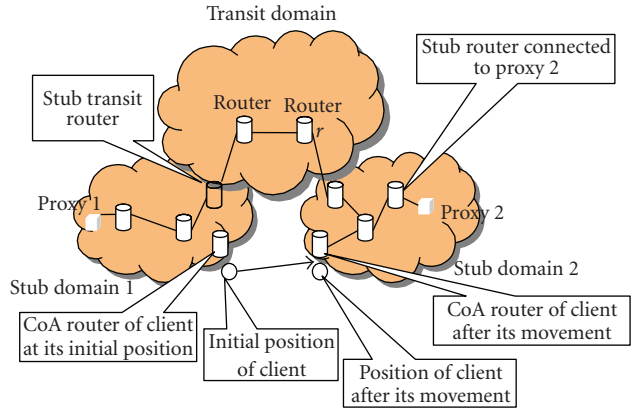


Figure 2: Movement of wireless node from one stub domain to another.

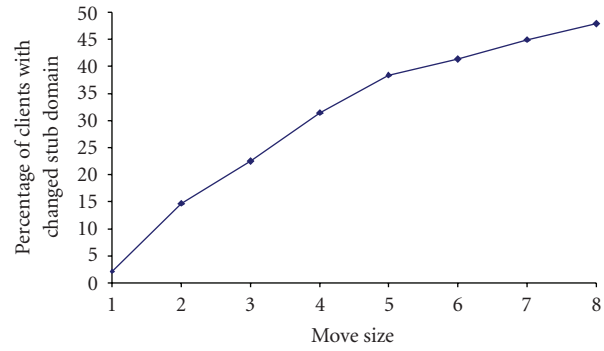


Figure 3: Percentage of stub domain changes versus move size.

In Figure 3 the percentage of moves that lead to a change of stub domain is shown for various ranges of movement from 1 to 8 km. As can be seen, the chances of changing stub domain would increase as we move further from a previous location. However, compared to the overall size of the network, relatively small moves could lead to significant probability of stub domain change.

Figure 4 shows the impact of movement size on the average delay penalty ratio between the wireless node and its assigned proxy. Delay penalty ratio is defined as the ratio of network delay from the wireless node to its previous proxy over this delay to the topologically closest proxy, see (1):

$$\textit{Delay Penalty Ratio} = \textit{Delay}\ (\text{from previous proxy})/\textit{Delay}\ (\text{from closest proxy}). \quad (1)$$

As shown in the Figure 4, without a proxy update mechanism, the latency perceived by the immersive communication service can significantly increase. Since all immersive communication flows to/from the wireless node are sent/received through its assigned proxy, the increased delay penalty also

implies wastage of network resources by using a longer than necessary path. For comparison, the delay penalty after a proxy update using the landmark method (to be described later) is also included.

Summarizing the above observations, there are three possible cases: (A) the range of movement of the wireless node is small and layer 2 handover mechanisms are sufficient to maintain connectivity to the CoA router. In this case, there is no change in the CoA router and no proxy update would be needed; (B) the wireless node moves to such an extent that the CoA router changes and a layer 3 handover is triggered. However, the new CoA router is in the same stub domain as the previous CoA router. If there is only one proxy within this stub domain, a proxy update is unlikely to be required. However, if the stub domain covers a vast geographical area and contains multiple proxies (in other words, the network topology is more or less flat in this region), an update may still improve the service delay; (C) as in case (B) above, but the new CoA router resides in a different stub domain. In this case, (assuming that each stub domain has its own proxy) it is highly probable that a proxy update would be beneficial.

The following observations are relevant to determine the most suitable proxy when an update is required. (1) There is only a very loose relationship between geographical and network proximity. This is particularly true for a hierarchical network topology as shown in Figure 2 where small changes in geographical proximity may translate to large variations in network delay. Nevertheless, it is quite likely that the optimal proxy (in terms of network latency) is not too far away. (2) We do not wish to burden the wireless node to carry out an exhaustive search for finding the optimal proxy (e.g., conducting a statistically reliable set of ping time measurements to all proxies after each move). (3) The wireless link (and in particular its MAC layer) could add significant jitter to delay measurements carried out by the wireless devices themselves.

Both observations (2) and (3) suggest that we need to develop some form of support by the fixed infrastructure for the proxy update mechanism that requires minimal functions from the client.

3.1. Proxy update mechanism

In this article we propose a mechanism to identify the closest proxy by providing a proxy location register (PLR) facility and a set of known landmarks. The steps in identifying the nearest proxy is summarised below.

Client notices a change in its CoA and therefore conducts a measurement of its round trip time (RTT) from the landmarks. The client sends its new care of address and the results of its delay measurements from landmarks (and possibly its derived network coordinates in the coordinate method to be described later) to the PLR. If the PLR already knows the closest proxy to this CoA router, it informs the client of the new proxy and the procedure ends. Otherwise, the PLR determines the closest proxy by the following steps: PLR creates a set of what it considers to be the closest proxies to the CoA router as potential candidates. In this article, we compare three different methods for creating this candidate set

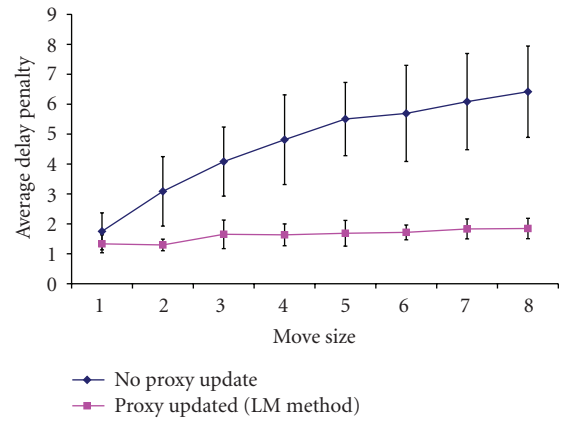


Figure 4: Average delay penalty increase versus move size (in kilometers).

and present comparative results on the effectiveness of these. The PLR sends a request message to each of the proxies in the candidate set to measure their RTT from the new CoA router. These measurements are then returned by the candidates to the PLR and may be cached for future use. The PLR determines the closest proxy for the client by selecting the candidate with minimum delay from the CoA router and informs the new proxy and its newly associated client about the update. Note that the above procedure can also be used at the time when a new client joins the immersive communication service and repeated thereafter in response to mobility.

The key step in the above procedure is producing the candidate set of closest proxies based on the client's CoA and its measured delays from the landmarks. Here, we compare three possible methods for this purpose.

(1) Coordinate method: this method is based on modeling the network by a multidimensional geometric space where measured delay between any two nodes is assumed to be equal to the distance between those two in the network geometric space [12]. The proxy location register in this case has the coordinates of all proxies. Each host (whether proxy or client) based on its RTT measurement from landmarks and coordinates of landmarks finds the optimal coordinates for itself in this space such that the distances in the same space match the measured delays from landmarks as closely as possible. Due to approximation, it is usually not possible to correctly determine topologically closest proxy to a client in the network. Hence, the PLR will choose a fixed number of closest proxies in the geometrical space as the candidate set for further measurement of their delay from the CoA router.

(2) Landmark Closeness Order: The second method is based on comparing the closeness order of proxies and the client to a set of well-known landmarks [13]. The PLR will then choose the candidate proxies by selecting those proxies with minimum distance (from client) in terms of their landmark closeness order. The set of determined proxies is then likely to include the closest proxy to the client. Landmark proximity order is most effective when RTT measurements from landmarks are rather accurate. For wireless hosts, the inaccuracy in RTT measurements may affect the

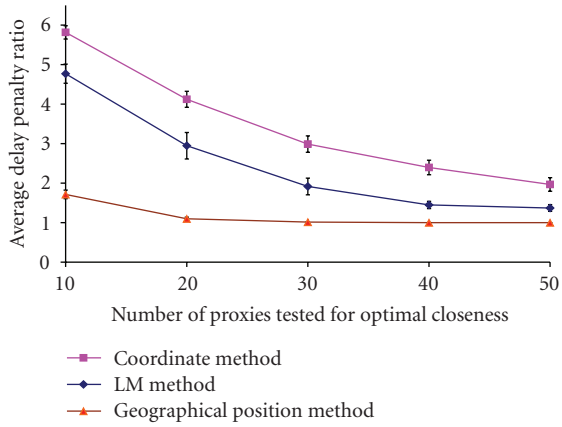


Figure 5: Average delay penalty with respect to the size of the candidate set (20% delay measurement error).

proximity order of landmarks. Distances between any two hosts when proximity orders are not accurate are calculated using weighted landmark closeness order distance as in (2):

$$D(N_1, N_2) = \sum_{i=0}^{LMN-1} (LMN - i) \times d_i(N_1, N_2). \quad (2)$$

Here $d_i(N_1, N_2) = 1$ if ith closest landmarks of nodes N_1 and N_2 are different, and $d_i(N_1, N_2) = 0$ if N_1 and N_2 have the same ith closest landmark. In (2) LMN is the number of landmarks and $(LMN - i)$ is the weight of ith closest landmark. According to (2) closest landmark has the largest weight.

(3) Geographical position: the third method is based on knowing the exact geographical position of the client and proxies. In this method the candidate set is comprised of a number of geographically closest proxies to the client, which is expected to include the topologically closest proxy as well. This method can only be used if the client knows its geographical position, for example, using GPS or by receiving its approximate geographical location from its CoA router. The geographical locations of proxies are also assumed to be known.

Figure 5 shows the delay penalty ratio after the proxy update is completed compared to the optimal proxy (if it could be found). Recall that in all three methods, the candidate proxies are requested to measure their RTT from the CoA router. The proxy having the minimum distance is then assigned to the client. It is possible that the optimal proxy is not within this candidate set. In order to increase the probability of finding the closest proxy, the number of proxies in the candidate set has been increased from 10 to 50 on the horizontal access (i.e., from 5.5% to 27.7% of all proxies in the simulated network model). Clearly, having a large number of proxies in the candidate set increases the accuracy but also raises the computation and bandwidth overhead and delay associated with proxy handover. Delay measurement between a mobile client and any other node is prone to error. A random error is therefore added to measured delay between each mobile client and each landmark node. Delay measurement error reduces mobile nodes' ability to correctly determine their landmark proximity order and network geometric coordinates. In Figure 5 mobile nodes are assumed to have an average of 20 percent delay measurement error.

As shown in Figure 5, by increasing the size of the candidate set, the delay penalty ratio decreases. In other words, the likelihood of finding a better proxy increases. The geographical position method can almost always find closest proxy for a candidate set size of 20 or more (11% of all proxies). The performance of landmark closeness order method and coordinate method are inferior but for sufficiently large set sizes the same average delay penalty can be achieved.

3.2. Updating multicast trees after a proxy update

Ideally, the proxy update in response to mobility should be seamless and without any disruption to the service. The main purpose of update is to improve delay performance and cost of delivery. It is therefore important to reconfigure multicast trees affected by this update as quickly as possible but without disruption. To this end, it is proposed here to have another facility referred to as client proxy association (CPA) server. (This server may be running on the same hardware as the proxy location register if appropriate.) The CPA server maintains a list of every client/avatar and their assigned proxies. After proxy location register completes proxy update of a mobile client, the PLR will update the client's entry in the CPA server with the new proxy. All proxies are required to consult CPA in constructing or reconfiguring their multicast trees after new client allocations as well as on regular time intervals.

After selection of a new proxy, old proxy of a moved client should forward required flows (of the moved client) to the new proxy of the client until a handover from the old proxy to the new proxy is complete. This policy assures uninterrupted forwarding of the media streams to the moving clients.

Handover delay for the proposed architecture comprises of the following components:

(i) delay from client to PLR: this is the delay for the mobile client to send a request for allocation of a new proxy after it detects a change in its CoA node;
(ii) maximum RTT delay from PLR to nominated proxies: this is the delay for the PLR to send a delay measurement request and receive a reply from all nominated proxies;
(iii) maximum RTT delay from nominated proxies to the new CoA of the mobile client (for finding closest proxy to CoA);
(iv) delay from PLR to the newly allocated proxy of the mobile client;
(v) delay from new proxy to the mobile client;
(vi) delay from new proxy to receiver proxies of the mobile client's media stream(s): this is the delay for new client's proxy to update all receiver proxies that need client's media stream.

Figure 6 shows the cumulative distribution of the handover delay after a physical move. For this measurement we

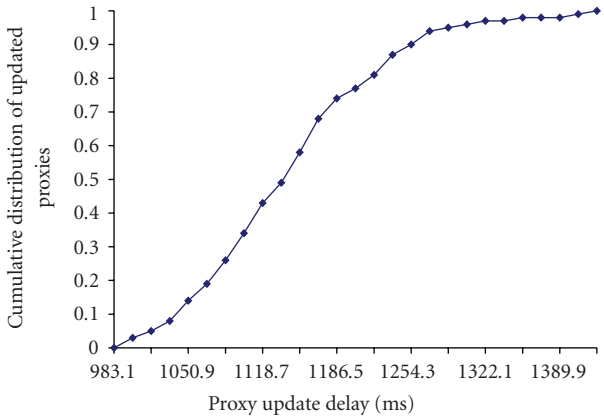


Figure 6: Cumulative distribution of handover time (update time) of proxies.

have scaled up average delay of the simulated network to that of the real networks. As can be seen, in majority of cases the handover is completed in less than 1.3 seconds. This may initially seem large, but it must be noted that the voice communication has not been interrupted during this period as the old proxy is still sending the voice streams to the client. This handover is to connect to a new proxy to improve delay performance of interactive communication and its perceptual impact may not be significant.

4. CONCLUSIONS

Immersive communication is likely to form the basis of many future services for collaborative work, education, and play. Access to these services with mobile devices will be of significant commercial interest. It is important to design a system architecture that can cater for both wired and wireless access. In this article, we have proposed an architecture that can achieve this goal. The distributed proxy model is suitable for both fixed and mobile clients and reduces the required functionality performed by the wireless nodes. We have also proposed possible methods for updating proxies in response to mobility. The role of the wireless device in managing mobility is still minimal and therefore the impact on the wireless link usage and power consumption remains negligible.

ACKNOWLEDGMENT

This work was funded by the Cooperative Research Centre (CRC) for Smart Internet Technology, Australia.

REFERENCES

[1] D. Johnson, C. Perkins, and J. Arrko, "Mobility Support in IPv6," RFC 3775, June 2004.
[2] Online game market, DFC Intelligence, "The Online Game Market 2004," August 2004, California, USA.
[3] M. Dowlatshahi and F. Safaei, "A recursive overlay multicast algorithm for distribution of audio streams in networked games," in *Proceedings of the IEEE International Conference on Networks (ICON '04)*, Singapore, November 16th–19th, 2004.
[4] M. Dowlatshahi and F. Safaei, "Overlay multicasting of real-time streams in virtual environments," in *Proceedings of IEEE Globecom*, San Francisco, 2006.
[5] C. Perkins, "IP Mobility Support," IETF RFC 2002, October 1996.
[6] M. Dowlatshahi and F. Safaei, "Audio distribution and rendering in network games," in *Proceedings of ATNAC*, 2004.
[7] P. Boustead, F. Safaei, and M. Dowlatshahi, "DICE: Internet delivery of immersive voice communication for crowded virtual spaces," in *Proceedings of IEEE Virtual Reality*, 2005.
[8] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proceedings of ACM SIGCOMM*, 2002.
[9] J. Liebeherr, M. Nahas, and W. Si, "Application-layer multicasting with Delaunay triangulation overlays," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, 2002.
[10] E. Brosh and Y. Shavitt, "Approximation and heuristic algorithms for minimum delay application-layer multicast trees," in *Proceedings of IEEE INFOCOM*, 2004.
[11] K. Calvert, E. Zegura, and S. Bhattacharjee, "How to model an internetwork," in *Proceedings of IEEE INFOCOM*, 1996.
[12] T. S. E. Ng and H. Zhang, "Predicting Internet network distance with coordinates-based approaches," in *INFOCOM*, pp. 170–179, 2002.
[13] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker, "Topologically-aware overlay construction and server selection," in *INFOCOM '02*, New York, 2002, IEEE.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2007, Article ID 95218, 10 pages
doi:10.1155/2007/95218

Research Article

Packet-Loss Modeling for Perceptually Optimized 3D Transmission

Irene Cheng,[1] Lihang Ying,[2] and Anup Basu[2]

[1] *Department of Computer and Information Sciences, University of Pennsylvania, Philadelphia, PA 19104, USA*
[2] *Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada T6G 2E8*

Received 31 October 2006; Accepted 21 December 2006

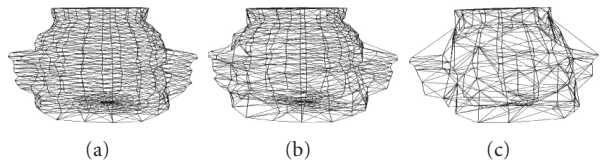
Recommended by Jianfei Cai

Transmissions over unreliable networks, for example, wireless, can lead to packet loss. An area that has received limited research attention is how to tailor multimedia information taking into account the way packets are lost. We provide a brief overview of our research on designing a 3D perceptual quality metric integrating two important factors, resolution of texture and resolution of mesh, which control transmission bandwidth, followed by a suggestion on alternative strategies for packet 3D transmission of both texture and mesh. These strategies are then compared with respect to preserving 3D perceptual quality under packet loss in ad hoc wireless networks. Experiments are conducted to study how buffer size, sending rate, sending intervals, and packet size can affect loss in unreliable channels. A model for estimating the optimal packet size is then proposed. We derive the optimal number of packets based on this model, and relate the theoretical derivations to actual network data.

1. INTRODUCTION

An important consideration in designing effective interactive online 3D systems is to adaptively adjust the model representation, while preserving satisfactory quality as perceived by a viewer. While most research in the literature focus on geometric compression and use only synthetic texture or color, we address both *geometry resolution* and *realistic texture resolution*, and analyze how these factors affect the overall perceptual quality. Our analysis is based on experiments conducted on human observers. The perceptual quality metric derived from experiments allows the appropriate level of detail (LOD) to be selected given the computation and bandwidth constraints. Detailed surveys on simplification algorithms can be found in [1, 2]. In order to easily control the details on a 3D object, we will follow a simple model approximation strategy based on multiresolution representation of texture and mesh. An example of geometric simplification is shown in Figure 1, in which a nutcracker-toy model is simplified to various resolution levels (number of triangles is 1260 left, 950 middle, and 538 right).

One of the major drawbacks with most 3D transmission algorithms is that they do not consider the possibility of packet loss over wireless or unreliable networks. Some wireless protocols proposed in the last decade include transmission control protocol (TCP), user datagram protocol (UDP), indirect-TCP (I-TCP) [3], and so on. For wireless networks, where packet loss occurs as a result of unreliable links and route changes, the TCP strategy leads to further delays and degradation in transmission quality. Even though issues of multimedia transmission over wireless networks have received attention [4], relatively little work has been done addressing wireless 3D transmission. In recent research, approaches for robust transmission of mesh over wireless networks [5, 6] have been outlined. However, these methods do not take joint texture and mesh transmission into account. Also, in [5, 6], it is assumed that some parts of the mesh can be transmitted without loss over a wireless network, allowing progressive mesh transmission to give good results. However, this assumption implies implementing a special standard with a combination of UDP and TCP protocols, which in general cannot be guaranteed in an arbitrary wireless environment. Special models for packet-loss probability have been developed by other researchers [7]. However, these models are usually associated with requirements such as retransmission. To keep our study applicable in an unrestricted ad hoc wireless environment, we simply assume packet-based transmission where a certain percentage of the packets may be lost. In this scenario, we compare how various types of 3D transmission strategies fare, and how to take



(a) (b) (c)

Figure 1: Nutcracker-toy model at various mesh resolution levels.

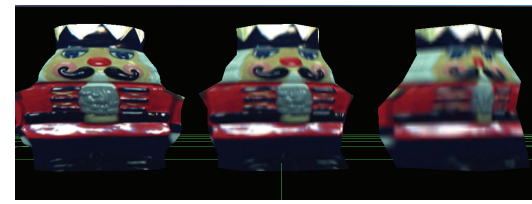


Figure 2: Evaluation example.

perceptual quality into account in designing a better strategy.

We consider an approach based on a perceptual quality metric following our earlier work [8]. Other approaches to joint texture-mesh transmission have been discussed in [9, 10]. The approach in [9] is based on view-dependent rate-distortion optimization, whereas our approach is view-independent. Also, both [9, 10] are progressive, which necessitates greater protection of base layers in case of packet loss; our approach on the other hand does not need to guarantee delivery of certain packets in order to make other packets useful. Joint texture-mesh transmission of terrains was addressed in [11]; however, the author did not consider perceptual quality optimization.

There are two types of methods for compressing 3D meshes over lossy networks. The first approach is to compress 3D meshes in an error-resilient way. Yan et al. [12] propose to partition meshes into pieces with joint boundaries and encode each piece independently. However, if packet loss occurs, there are holes in meshes resulting from missing pieces. Jaromersky et al. [13] introduce multiple description coding for 3D meshes. Each description can be independently decoded. But it assumes the connectivity data is guaranteed to be received correctly, and is conceptually similar to Strategy A. The second set of methods use error protection to restore lost packets [5, 14].

Extensive research has been conducted on error resilience for audio or video communication [15–17]. Forward error correction (FEC) is a common technology to decrease the impact of packet loss. With FEC schemes, redundant data is added into the original data so that the lost original data can be recovered from the redundant data. Sending more redundant data increases the probability of recovering the loss data; however, additional redundancy increases the bandwidth requirements and the loss. The amount of redundant data should be adaptively adjusted by the characteristics of packet loss [17–21].

The remainder of this paper is organized as follows. Section 2 reviews past work on perceptual quality evaluation and discusses how to relate bandwidth with texture and mesh reduction considering perceptual quality. Section 3 examines possible strategies for 3D image transmission and analyzes which one is most suitable for optimizing perceptual quality under packet loss. Experimental results are presented. Different scenarios of packet loss attributed to different factors over a lossy network are presented in Section 4. A strategy for packet-size optimization is proposed in Section 5, before the work is concluded in Section 6.

2. 3D PERCEPTUAL QUALITY OPTIMIZATION

In the area of image compression, mean square error (MSE) is commonly used as a quality predictor. However, past research has shown that MSE does not correlate well to perceived quality based on human evaluation [22]. Since this study, a number of new quality metrics based on the human visual system have been developed [23].

Several 3D objects were used as stimuli in our experiments. These objects were captured with the *zoomage* 3D scanner. The participants (judges) were asked to compare the target stimulus with the two referential stimuli and assign it one of the following ratings: *very poor (1), poor (2), fair (3), good (4), very good (5)*.

Figure 2 illustrates two referential stimuli (left and right) and one target stimulus (center) in the experiment.

Considering perceptual evaluations, we observed that:

(i) perceived quality varies linearly with texture resolution (Figure 3(a));

(ii) perceived quality varies following an exponential curve for geometry (Figure 3(b)). Scaling the texture (t) and geometry (g) between 0 and 1, it can be shown that

$$Q(g,t) = \frac{1}{1/(m+(M-m)t) + (1/m - 1/(m+(M-m)t))(1-g)^c}. \quad (1)$$

Details of the perceptual evaluations and metric derivation can be found in our prior work [8]. Note that the quality value varies in the range of 1 (m) to 5 (M), because of the range of values allowed in the perceptual ratings.

Consider now that b is the estimated total bandwidth for the transmission time interval, T is the texture, and G is the geometry file sizes, possibly compressed, at maximum resolution. We assume that as the texture (or geometry) is scaled by a factor t (or g) in both dimensions, the corresponding file sizes get reduced to t^2T (or g^2G). To utilize the bandwidth completely, we must have

$$b = t^2T + g^2G. \quad (2)$$

Given b we can choose the relative proportion of texture and mesh to create a 3D model in many different ways, as long as (2) is satisfied. The question is "What is the optimal choice maximizing perceptual quality"? Considering $m = 1$, $M = 5$, and $c = 2.7$ (approximately) for many objects based on

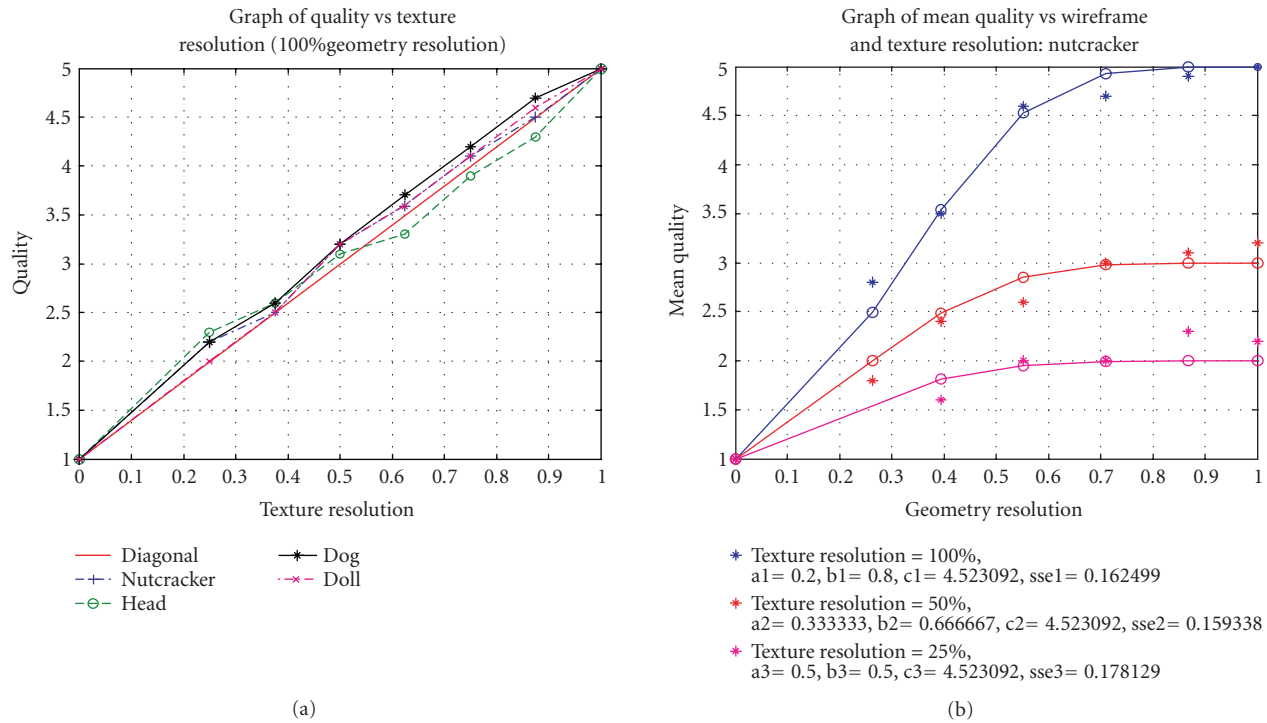


FIGURE 3: (a): Quality versus texture resolution (100% geometry resolution); (b): quality versus geometry for various texture resolutions.

perceptual tests, (1) can be further simplified to

$$Q(g,t) = \frac{1}{1/(1+4t) + (1-1/(1+4t))(1-g)^{2.7}}. \quad (3)$$

Maximizing (3) is equivalent to minimizing the inverse of this equation; considering this and (2), optimizing quality reduces to minimizing

$$Q_{b,G,T}(t) = \frac{1}{1+4t} + \left(1 - \frac{1}{1+4t}\right)\left(1 - \sqrt{\frac{b - t^2 T}{G}}\right)^{2.7}, \quad (4)$$

where b, G, and T are parameters.

Example 1. Let $b = 12$ Mbits, $T = 20$ Mbits, and $G = 10$ Mbits.

In this case, t can only vary in the range $[\sqrt{2/20}, \sqrt{10/20}] = [0.316, .707]$ so that (2) can be satisfied. The graph of (4) for varying t for this case is shown in Figure 4. The optimal value of t is close to 0.6 for this example. In general, given T and G for a 3D object, optimum t can be precomputed for a discrete number of b values in the range $[0, T + G]$.

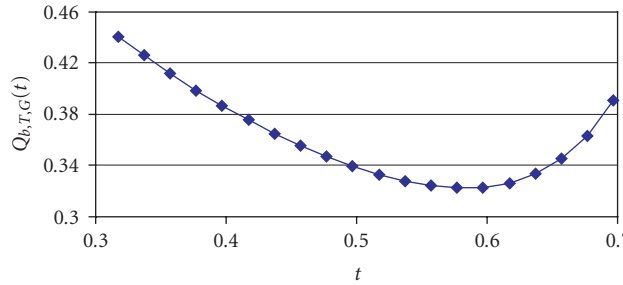


FIGURE 4: Inverse perceptual quality curve for Example 1.

3. PERCEPTUALLY OPTIMIZED TRANSMISSION

To simplify the model of wireless transmission, we assume that data is sent in packets of equal size and there is a possibility that a certain proportion of these packets may be lost. Various protocols [24] suggest retransmission approaches in case of packet loss; however, retransmission is not conducive to time bound real-time applications, such as 3D visualization for online games. We considered several possible strategies for packet construction in wireless 3D transmission, and then analyzed the pros and cons of each. We found that breaking up a 3D image into fragments can cause unacceptable voids; progressive transmission [25] necessitates receiving packets at lower levels before packets at higher levels can become useful; and sending duplicate copies of base layer packets in progressive transmission increases

SERVER SITE

T: *original texture*;
M: *original mesh, in a regular form allowing easy subsampling*;
Construct $T_1, T_2, \ldots, T_n$ *by regular, nonidentical subsampling of T*;
(Comment: *e.g., given a* 100×100 *pixel texture T, we can construct* $T_1, T_2, \ldots, T_{16}$ *by defining* T_1 *as* $T(0+4i, 0+4j)$, $i, j = 0, \ldots, 24$; T_2 *as* $T(0+4i, 1+4j)$, $i, j = 0, \ldots, 24; \ldots, T_{16}$ *as* $T(3+4i, 3+4j)$, $i, j = 0, \ldots, 24$);
Construct $M_1, M_2, \ldots, M_n$ *by regular, nonidentical subsampling of M*;
Form packets $P_1, P_2, \ldots, P_n$ *where* $P_i = T_i + M_i$; $i = 1, \ldots, n$, *with header and subsampling information added to each packet*;
Transmit n packets to a client on request, possibly in a randomized order;

CLIENT SITE

Request server to transmit a 3D object;
Receive packets from server;
Uncompress mesh and texture data stored in this packet;
Set up initial display based on first packet received and interpolation information stored in header;
Update display based on next packet received.

Algorithm 1

bandwidth requirements. We thus focus on the two following strategies, concentrating on regular mesh transmission.

Strategy A

3d partial information transmission

In this approach, we break up the texture and mesh into packets by subsampling into overlapping but nonidentical components. At the client site, the overall texture and mesh are reconstructed based on interpolation from the received packets. One implementation of this approach is given Algorithm 1.

Limitations of Strategy A

One of the shortcomings of this approach is that the texture and mesh data receives equal importance; that is, the same fraction of each is transmitted in a packet. The perceptual quality analysis in the last section shows that for optimizing perceptual quality the relative importance of texture and mesh can vary depending on the available bandwidth; this issue is not taken into account in Strategy A.

Strategy B

3d perceptually optimized partial information transmission

This approach extends 3PIT by taking perceptual quality into account. The algorithm modifies Strategy A by a bandwidth estimation step followed by perceptually optimized packet creation. Details are described in Algorithm 2.

SERVER SITE

T, M: as for Strategy A;
Receive bandwidth estimate (B_e) *and estimated loss proportion* (L) *from requesting client*;
Compute server transmitting bandwidth: $B_s \leftarrow B_e/(1-L)$;
Compute optimum texture and geometry scaling factors t_e *and* g_e *following procedure for minimizing* (4) *in the last section, considering bandwidth to be* B_e;
Compute scaled texture (T_s) *and mesh* (G_s), *assuming transmitting bandwidth* B_s, *based on factors* t_e *and* g_e;
(Comment: *specifically:* $Ts = (t_e^2/(1-L))T$ *and* $Gs = (g_e^2/(1-L))G$; *with texture and mesh possibly being interpolated to higher than the current maximum size in case the scaling factors are greater than 1*);
Construct $T_{s1}, T_{s2}, \ldots, T_{sn}$ *by regular, nonidentical subsampling of* T_s;
Construct $M_{s1}, M_{s2}, \ldots, M_{sn}$ *by regular, nonidentical subsampling of* M_s;
Form packets $P_1, P_2, \ldots, P_n$, *where* $P_i = T_{si} + M_{si}$; $i = 1, \ldots, n$, *with header and subsampling information added to each packet*;
(Comment: *number of packets n is chosen based on prior decision on packet size*);
Transmit n packets to a client, possibly in a randomized order;

CLIENT SITE

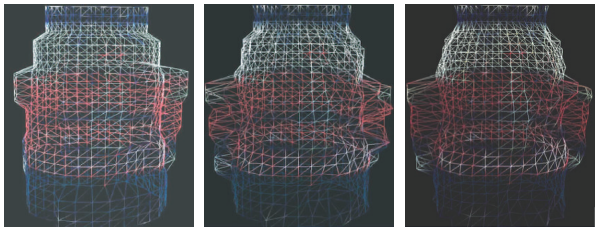
Request server to transmit a 3D object;
Receive packets from server for bandwidth estimation;
Estimate bandwidth (B_e) *based on number of packets received in a certain time interval and estimate loss proportion* (L);
Receive packets from server containing partial data on the 3D object;
Uncompress mesh and texture data stored in this packet;
Set up initial display based on first packet received and interpolation information stored in header;
Update display based on next packet received.

Algorithm 2

Comments on Strategy B

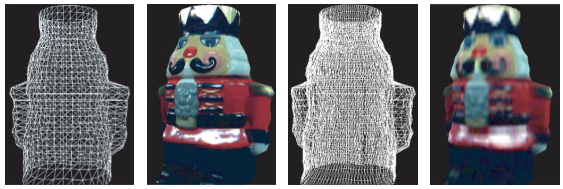
On first observation it may appear that this strategy does not take packet-loss proportion (L) into account in the transmission strategy. However, in reality, this is not the case. Without any packet loss, the transmission bandwidth (B_s) would be used to compute the optimum texture and mesh scaling factors. When packets are lost the remaining packets may not be perceptually optimal for the effective bandwidth after packet loss. We thus form packets that are optimal at a lower bandwidth (B_e).

One of the drawbacks of Strategy B is the need to estimate bandwidth and packet loss ratio. This estimation-based transmission may not be practical where feedback from client to a server is not reliable, or for multicasting over heterogeneous networks with varying packet loss and bandwidths. This issue needs to be addressed in future research.



(a) (b) (c)

Figure 5: Interpolating and reconstructing mesh of nutcracker model when 2 (left), 4 (middle), and 8 of 16 packets are received.



(a) (b) (c) (d)

Figure 6: Two representations of the nutcracker texture + mesh models: left has lower quality mesh, requiring 125 Kb total bandwidth, but higher perceptual quality; right has higher quality mesh, resulting in lower quality texture to keep total bandwidth at 134 Kb, but has lower perceptual quality.

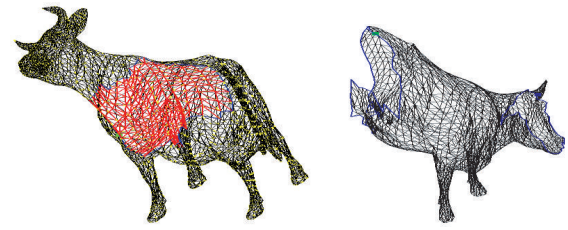
Experimental results of Strategy B

We show some preliminary implementations towards deploying 3POPIT over a lossy wireless network. Figure 5 shows the effect of receiving and combining 2, 4, and 8 of 16 subsamples of the nutcracker mesh. Note that results may vary from one execution to another for a random percentage of packet loss.

Figure 6 shows the effect of optimized *versus* non-optimized transmission on perceptual quality. Two versions of the same model are shown, with the mesh on the left and the texture mapped on the right. Although the texture and mesh together for the left and right models use nearly the same bandwidth, 125 and 134 Kb, respectively, the left one is favored by most viewers based on perceptual experiments.

Although a regular or semiregular mesh is used for illustration in this paper, our strategy can be extended to irregular meshes where connectivity information needs to be transmitted; triangular faces are arranged in continuous long strips following the valence driven algorithm, neighboring vertices and connectivity information are distributed evenly into different packets to minimize the risk of losing data affecting a large neighborhood [26]. Figure 7 shows how our strategy, combined with the valence driven encoding and decoding algorithm [27], can be applied to an irregular mesh.

Multiple-resolution strategy is often used to refine image or mesh data in a progressive manner. However, progressive methods [9, 10] necessitate greater protection of base layers in case of packet loss; our approach on the other hand does not need to guarantee delivery of certain packets in order to make other packets useful.



(a) (b)

Figure 7: Combining our strategy with the valence driven algorithm on an irregular cow mesh, (a) the colored patch shows the neighborhood after 400 vertices are distributed to different packets. (b) Shows the partly reconstructed mesh after 3000 vertices are retrieved from the packets.

Table 1: Receiving rate and packet loss for different buffer sizes.

Buffer size (bytes)	Receiving rate (Bps)	Packet loss (%)
(a) 32 768	126 611	0.98
(b) 16 384	127 450	0.39
(c) 4096	123 138	3.91
(d) 2048	0	100

4. NETWORK EXPERIMENTS ON PACKET LOSS

The long-term objective of our research is to identify the appropriate parameters (packet size, sending rate, sending interval, and buffer size) for different applications to maximize throughput and minimize packet loss of UDP transmission in different Internet environments with wireless LAN access. In the experiments, the server side was a desktop computer in the Department of Computing Science, University of Alberta, Edmonton, Canada. The client was a laptop, which linked to a router following 802.11 b. The router accessed the Internet using a cable network (with a maximum capacity of 640 KBps). The client was located in the same city as the server. Both client and server ran Red Hat Linux Release 9 Shrink (2.4.30 Kernel). The experiments were conducted during the day (8:00–19:00) from November 25 to November 27, 2005.

4.1. Buffer size

We first discuss the effect of socket buffer size on packet loss. With fixed packet size (4096 bytes) and sending rate (128 KBps), Table 1 (a) and (b) show that different buffer sizes larger than the packet size make no significant difference on packet loss. However, if buffer size is less than packet size, all packets are lost as shown in (d). The interesting point is in (c), when buffer size is equal to packet size, there is a significant packet loss as well. This can be attributed to the fluctuating bandwidth; reducing bandwidth capacity can cause an overflow on a congested buffer. In the experiments reported

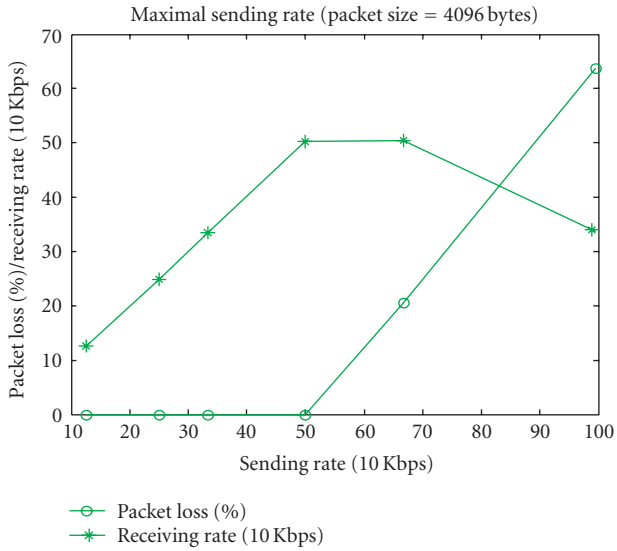


Figure 8: Effect of sending rate on packet loss.

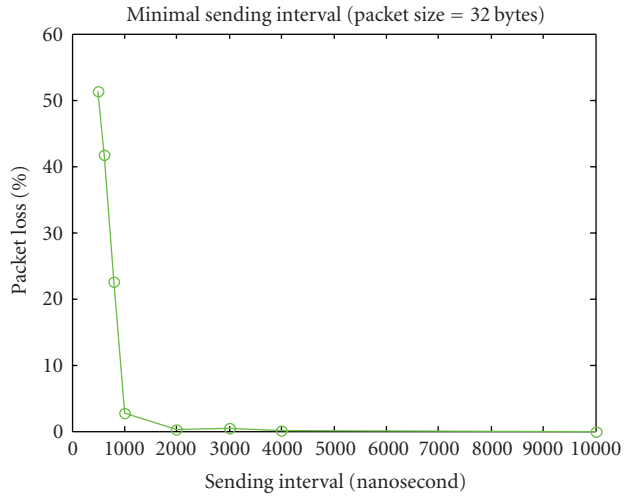


Figure 9: Packet loss versus sending interval.

in Sections 4.2 to 4.4, we wanted to study packet loss independent of buffer size and therefore a large enough buffer size of 65 536 was used.

4.2. Sending rate

Next we look into how sending rate affects packet loss. We consider a large enough sending interval and fixed packet size (4096 bytes), and let sending rate increase from 128 to 1024 Kbps. It can be seen from Figure 8, that as sending rate increases, receiving rate increases and packet loss remains around zero until around 500 Kbps. However, after sending rate overflows the connection (larger than 500 Kbps), packet loss dramatically increases and receiving rate drops owing to packet loss.

4.3. Sending interval

Fixing packet size at 32 bytes and without overflowing the connection, sending intervals varying from 10 000, 4000, 3000, 2000, 1000, 800, 600, 500 nanoseconds were used to test the packet-loss rate. Figure 9 shows the packet loss plotted against the time interval before transmitting the next packet. Clearly, the sending interval should not be too small (<2 ms), otherwise loss rate can be high.

4.4. Packet size

Now we want to see how different packet sizes affect the receiving rate, as well as the packet loss. First, we performed experiments in an environment without other competing connections. With sending rate around 256 Kbps, without overflowing the maximum connection capacity of 640 Kbps, packet size was selected from the set of 65536, 32 768, 16 384, 8192, 4096, 2048, 1024, 512, 256, 128, 64, or 32 bytes. Table 2 shows that when sending interval is large enough (equal to or larger than 2 ms), different packet sizes have not much effect on sending rate, receiving rate, or packet loss. Sending rate remains stable independent of packet size. However, as mentioned above, when the sending interval is too small, packet loss sharply increases and receiving rate drops accordingly.

Table 2: Effect of different packet sizes on sending/receiving rate and packet loss (without competing connections).

Packet size (byte)	Sending interval (nano-second)	Sending rate (Bps)	Receiving rate (Bps)	Packet loss (%)
65 500	256 000	255 510	253 852	0.00
32 768	128 000	255 694	255 694	0.00
16 384	64 000	255 682	252 899	0.78
8192	32 000	255 876	255 637	0.00
4096	16 000	255 690	252 476	0.98
2048	8000	255 686	256 480	0.00
1024	4000	255 688	255 374	0.00
512	2000	255 724	255 668	0.00
256	1000	255 691	210 535	16.58
128	500	255 691	118 407	53.69
64	250	255 682	485	99.81
32	125	255 401	499	99.80

We then performed experiments in an environment with competing connections. In order to setup a competing environment, we configured the bandwidth of 802.11 b to 1 Mb. Four additional FTP concurrent connections were opened between the client and the server. Using a sending rate at 64 Kbps, packet sizes were selected from 128,256,512,1024,2048,4096,8192, or 16 384 bytes. Figure 10 shows how packet loss varies with packet size under such condition. When packet size is very small or very large, packet loss can be large. Loss was at the minimum when packet size was around 2000.

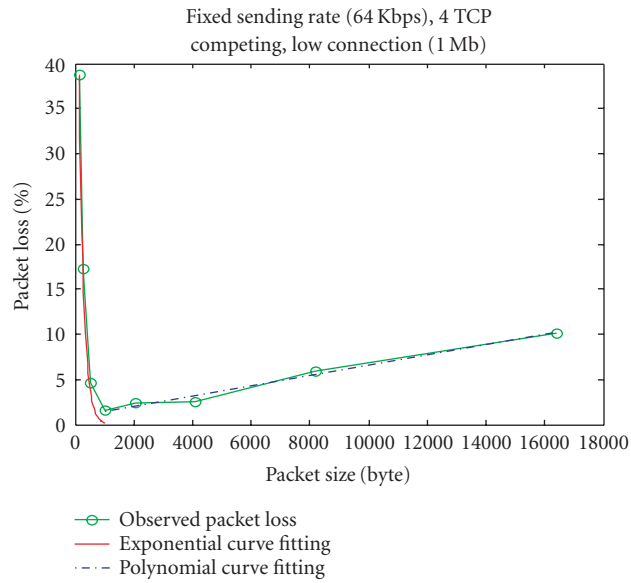


FIGURE 10: Packet loss versus packet size in a congested network.

In related work [28], adjusting the packet sending interval based on feedback from the network was proposed, but the work did not study the effect of different packet sizes. In [29], the authors proposed to determine the sending rate T as a function of the packet size s, round-trip time $\boldsymbol{R}$, steady-state loss event rate $\boldsymbol{p}$, and the TCP retransmit timeout value t_{RTO}. However, they did not study how the theoretical model works in a wireless LAN (WLAN) environment. UDP throughput and CPU utility for bulk data transfer with different sender and receiver buffer sizes and packet sizes were studied in [30] but packet loss was not taken into consideration. The delay of UDP, TCP, and TCP with the option NODELAY for voice applications sending 160 bytes per 20 milliseconds was discussed in [31]. Our work differs from others by carrying out a comprehensive study on packet size, sending rate, sending interval, and buffer size through real world experiments in a competing WLAN environment, where there are packet losses besides congestion.

5. PACKET-SIZE OPTIMIZATION

In Figure 10, it can be observed that in the competing environment when packet size is small, the loss can be quite high because of congestion resulting from the need to route many packets. As the packet size gradually increases beyond an optimum point with low loss, the loss rate increases again. We propose a strategy to model this characteristic and determine the optimal packet size following some simplifying assumptions. We assume exponential and linear models for packet loss depending on packet size; however, the approach can be extended to other models as well. Even though various models for packet loss over wireless networks have been proposed [4, 32], determining the optimal packet size has not received much attention. Larger packet sizes minimize the overhead from packet headers, but have a higher probability of being corrupted. There are several studies on packet size optimization based on different metrics, such as throughput, goodput, transmission range, and energy efficiency, in wired and wireless networks [33–40].

For simplicity, we use the following assumptions and notation:

H: total network header size required for each packet;

S: amount of payload (application) data transmitted, if there is only 1 packet used;

s_n: amount of payload data transmitted in each packet, if n packets used; probability packet of size D is not lost is

$$e^{-\lambda D}, \tag{5}$$

that is, an exponential packet-loss model is used with parameter λ and packet size as variable.

Let the amount of data transmitted *be* $B = S + H$ when only 1 packet is transmitted, and $s_1 = S$. When 2 packets are transmitted, we have $B = 2s_2 + 2H$. Since

$$B = S + H = 2s_2 + 2H, \quad s_2 = \frac{S - H}{2}. \tag{6}$$

In general,

$$s_n = \frac{S - (n-1)H}{n}. \tag{7}$$

For large n, that is, when $(n-1)/n \approx 1$,

$$s_n \approx \frac{S}{n} - H. \tag{8}$$

Lemma 1. *Following the model and assumptions above and independent transmission of packets, for large n, the total expected payload received with n packets given $B = S + H$ is*

$$f(n) \approx e^{\lambda H}\left[e^{-(\lambda S)/n}(S - nH)\right]. \tag{9}$$

Proof. Follows from the definitions, and noting that the total expected payload received equals the sum of individual packet sizes times the probability that it is received

$$f(n) = e^{-\lambda D}\left[n\left(\frac{S}{n} - \frac{(n-1)H}{n}\right)\right] = e^{-\lambda(S/n - H)}(S - nH). \tag{10}$$

Since $(n-1)/n \approx 1$ for large n

$$f(n) \approx e^{\lambda H}\left[e^{-(\lambda S)/n}(S - nH)\right]. \tag{11}$$

□

Theorem 1. *The number of packets n optimizing the expected amount of payload transmitted for the exponential model given $B = S + H$ is an integer equal to either*

$$\left\lfloor \frac{S\sqrt{\lambda^2 H^2 + 4\lambda H}}{2H} - \frac{\lambda S}{2} \right\rfloor \quad \text{or} \quad \left\lceil \frac{S\sqrt{\lambda^2 H^2 + 4\lambda H}}{2H} - \frac{\lambda S}{2} \right\rceil. \tag{12}$$

Proof. Follows from optimizing the function in Lemma 1 and the fact that the number of packets is an integer. □

Now, suppose that the probability a packet of size D is not lost is defined by $a - \lambda D$; that is, a linear packet loss model is used with parameter λ and packet size as variable. This model is more meaningful in case the network characteristics follow the data in Figure 10. For this linear model; can be shown the following lemma.

Lemma 2. *Following the linear model and assumptions above and independent transmission of packets, for large n, the total expected payload received with n packets given $B = S + H$ is*

$$f(n) \approx \left[a - \lambda\left(\frac{S}{n} - H\right)\right](S - nH). \tag{13}$$

Proof. Follows from the definitions, and noting that the total expected payload received equals the sum of individual packet sizes times the probability that it is received,

$$f(n) = (a - \lambda D)[n(S/n - (n-1)H/n)]. \tag{14}$$

Since $(n-1)/n \approx 1$ for large n,

$$f(n) \approx \left[a - \lambda\left(\frac{S}{n} - H\right)\right](S - nH). \tag{15}$$

□

Theorem 2. *The number of packets optimizing the expected amount of payload transmitted for the linear model given $B = S + H$ is an integer equal to either*

$$\left\lfloor S\sqrt{\frac{\lambda}{(aH + \lambda H^2)}}\right\rfloor \quad or \quad \left\lceil S\sqrt{\frac{\lambda}{(aH + \lambda H^2)}}\right\rceil. \tag{16}$$

Proof. Follows from optimizing the function in Lemma 2 and the fact that the number of packets is an integer. □

For the data in Figure 10, we can observe that for the first part the curve fits a decreasing exponential function. If we only consider this part of the curve optimum point is the rightmost point because with increasing packet size (more right on the bottom axis) the overhead from total header sizes of all packets is lower and the packet loss is also lower.

For the second part, after the minimum point of the exponential part, we can fit a linear function $y = 0.8842 + 0.0006x$. Thus the probability of a packet not lost equals $(1 - y/100) = 0.9912 - 0.000005683\,D$, that is, $a = 0.9912$ and $\lambda = 0.000005683$ in Theorem 2. Given S and H we can determine the optimum number of packets following Theorem 2 for the linear section of the graph in Figure 10.

The network packet size in our optimization strategy is independent of the processing performed at the application level; no matter how the application data, for example, texture image and mesh information, are redistributed and segmented, the processed data, likely compress, are passed to the network as a byte stream (compressed or uncompress), which is then packed into the network packets. In our simulation, an IP header (8 bytes) and a UDP (20 bytes) are added to each network packet. 2 Mbytes application data was used in our packet loss experiment, the result of which is plotted in Figure 10. Substituting $S = 2$ Mbytes and $H = 28$ bytes in (16) we obtain the optimal number of packets by Theorem 2 to be either 904 or 905 corresponding to a packet size of about 2212 bytes. This shows that the optimal packet size for maximizing payload (actual multimedia data without packet headers) may not necessarily correspond to the packet size with lowest loss rate. For this experiment we assumed that the header for the multimedia data was not included in the packets. If we consider duplicating multimedia header information in packets for increased reliability under packet loss, H in the formula will increase giving a lower optimal number of packets or higher optimal packet size for this example.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we discussed factors controlling 3D image degradation and outlined an approach for estimating perceptual quality considering variations in mesh and texture resolutions. A theoretical framework for determining the relative importance of texture *versus* mesh was presented. An approach to optimizing perceptual quality under packet loss was then outlined. Experimental results were described to validate our approach. Finally, an approach for estimating the optimal packet size was proposed, following experimental results to collect real data on packet loss in congested wireless networks. In future work, we will extend and verify our packet size estimation method with more realistic models derived from tests over wireless networks, such as taking channel fading and burst error into consideration, to refine our assumptions. Implementations and user evaluations with handheld devices will also be conducted. We will also consider issues relating to MPEG4-3DMC compatibility [41].

ACKNOWLEDGMENTS

The support of Alberta Science and Research Authority (ASRA) and NSERC are gratefully acknowledged. Part of this work was presented in IEEE International Conference on Multimedia and Expo, 2006, in Toronto, Canada.

REFERENCES

[1] P. Heckbert and M. Garland, "Survey of polygonal surface simplification algorithms," Tech. Rep. TR97-045, Computer Science Department, Carnegie Mellon University, Pittsburgh, Pa, USA, 1997.

[2] D. Luebke, M. Reddy, J. Cohen, A. Varshney, B. Watson, and R. Huebner, *Level of Detail for 3D Graphics*, Morgan Kaufmann, San Francisco, Calif, USA, 2002.

[3] A. Bakre and B. R. Badrinath, "I-TCP: indirect TCP for mobile hosts," in *Proceedings of the 15th International Conference on Distributed Computing Systems (ICDCS '95)*, pp. 136–143, Vancouver, BC, Canada, May-June 1995.

[4] D. Wu and R. Negi, "Effective capacity: a wireless link model for support of quality of service," *IEEE Transactions on Wireless Communications*, vol. 2, no. 4, pp. 630–643, 2003.

[5] G. Alregib, Y. Altunbasak, and J. Rossignac, "Error-resilient transmission of 3D models," *ACM Transactions on Graphics*, vol. 24, no. 2, pp. 182–208, 2005, Earlier version in ICASSP'02.

[6] Z. Chen, B. Bodenheimer, and J. Barnes, "Robust transmission of 3D geometry over lossy networks," in *Proceeding of the 8th*

International Conference on 3D Web Technology, pp. 161–172, Saint Malo, France, March 2003.

[7] K. K. Lee and S. T. Chanson, "Packet loss probability for real-time wireless communications," *IEEE Transactions on Vehicular Technology*, vol. 51, no. 6, pp. 1569–1575, 2002.

[8] Y. Pan, I. Cheng, and A. Basu, "Quality metric for approximating subjective evaluation of 3-D objects," *IEEE Transactions on Multimedia*, vol. 7, no. 2, pp. 269–279, 2005, Short version in IEEE International Conference on Image Processing 2003.

[9] S. Yang, C.-H. Lee, and C.-C. J. Kuo, "Optimized mesh and texture multiplexing for progressive textured model transmission," in *Proceedings of the 12th ACM International Conference on Multimedia*, pp. 676–683, New York, NY, USA, October 2004.

[10] D. Tian and G. Al-Regib, "FQM: a fast quality measure for efficient transmission of textured 3D models," in *Proceedings of the 12th ACM International Conference on Multimedia*, pp. 684–691, New York, NY, USA, October 2004.

[11] L. Balmelli, "Rate-distortion optimal mesh simplification and communication," Ph.D. dissertation, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, 2001.

[12] Z. Yan, S. Kumar, and C.-C. J. Kuo, "Error-resilient coding of 3-D graphic models via adaptive mesh segmentation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 11, no. 7, pp. 860–873, 2001.

[13] P. Jaromersky, X. Wu, Y.-J. Chiang, and N. Memon, "Multiple-description geometry compression for networked interactive 3D graphics," in *Proceedings of 3rd International Conference on Image and Graphics (ICIG '04)*, pp. 468–471, Hong Kong, December 2004.

[14] G. Al-Regib and Y. Altunbasak, "An unequal error protection method for packet loss resilient 3D mesh transmission," in *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '02)*, vol. 2, pp. 743–752, New York, NY, USA, June 2002.

[15] Y. Wang and Q.-F. Zhu, "Error control and concealment for video communication: a review," *Proceedings of the IEEE*, vol. 86, no. 5, pp. 974–997, 1998.

[16] Y. Wang, S. Wenger, J. Wen, and A. K. Katsaggelos, "Review of error resilient coding techniques for real-time video communication," *IEEE Signal Processing Magazine*, vol. 17, no. 4, pp. 61–82, 2000.

[17] J.-C. Bolot, S. Fosse-Parisis, and D. Towsley, "Adaptive FEC-based error control for Internet telephony," in *Proceedings of 18th Annual Joint Conference of the IEEE Computer and Communications Societie (INFOCOM '99)*, vol. 3, pp. 1453–1460, New York, NY, USA, March 1999.

[18] L. Baldantoni, H. Lundqvist, and G. Karlsson, "Adaptive end-to-end FEC for improving TCP performance over wireless links," in *Proceedings of IEEE International Conference on Communications (ICC '04)*, vol. 7, pp. 4023–4027, Paris, France, June 2004.

[19] S.-W. Yuk, M.-G. Kang, B.-C. Shin, and D.-H. Cho, "An adaptive redundancy control method for erasure-code-based real-time data transmission over the Internet," *IEEE Transactions on Multimedia*, vol. 3, no. 3, pp. 366–374, 2001.

[20] K. Park and W. Wang, "QoS-sensitive transport of real-time MPEG video using adaptive redundancy control," *Computer Communications*, vol. 24, no. 1, pp. 78–92, 2001.

[21] C. H. Lin, C. H. Ke, C. K. Shieh, and N. K. Chilamkurti, "An enhanced adaptive FEC mechanism for video delivery over wireless networks," in *Proceedings of International Conference on Networking and Services (ICNS '06)*, p. 106, Santa Clara, Calif, USA, July 2006.

[22] J. L. Mannos and D. J. Sakrison, "The effects of a visual fidelity criterion on the encoding of images," *IEEE Transactions on Information Theory*, vol. 20, no. 4, pp. 525–536, 1974.

[23] J. O. Limb, "Distortion criteria of the human viewer," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 12, pp. 778–793, 1979.

[24] R. Caceres and L. Iftode, "Improving the performance of reliable transport protocols in mobile computing environments," *IEEE Journal on Selected Areas in Communications*, vol. 13, no. 5, pp. 850–857, 1995.

[25] H. Hoppe, "Progressive meshes," in *Proceedings of the 23rd Annual Conference on Computer Graphics (SIGGRAPH '96)*, pp. 99–108, New Orleans, La, USA, August 1996.

[26] I. Cheng, L. Ying, and A. Basu, "A perceptually driven model for transmission of arbitrary 3D models over unreliable networks," in *Proceedings of the 3rd International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT '06)*, Chapel Hill, NC, USA, June 2006.

[27] P. Alliez and M. Desbrun, "Valence-driven connectivity encoding for 3D meshes," *Computer Graphics Forum*, vol. 20, no. 3, pp. 480–489, 2001.

[28] S. Floyd, M. Handley, J. Padhye, and J. Widmer, "Equation-based congestion control for unicast applications," in *Proceedings of the ACM Annual Conference of the Special Interest Group on Data Communication (SIGCOMM '00)*, pp. 43–56, Stockholm, Sweden, August-September 2000.

[29] A. C. Feng, A. C. Kapadia, W.-C. Feng, and G. G. Belford, "Packet spacing: an enabling mechanism for delivering multimedia content in computational grids," *Journal of Supercomputing*, vol. 23, no. 1, pp. 51–66, 2002.

[30] Y. Gu and R. L. Grossman, "Optimizing UDP-based protocol implementations," in *Proceedings of the 3rd International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet '05)*, Lyon, France, February 2005.

[31] X. Zhang and H. Schulzrinne, "Voice over TCP and UDP," Tech. Rep. CUCS-033-04, Department of Computer Science, Columbia University, New York, NY, USA, 2004.

[32] R. El-Azouzi and E. Altman, "A queuing analysis of packet dropping over a wireless link with retransmissions," in *Proceedings of the 8th International Conference on Personal Wireless Communications (PWC '03)*, Venice, Italy, September 2003.

[33] V. K. M. Vadakital, M. M. Hannuksela, M. Razaei, and M. Gabbouj, "Optimal IP packet size for efficient data transmission in DVB-H," in *Proceedings of the 7th Nordic Signal Processing Symposium (NORSIG '06)*, pp. 82–85, Reykjavik, Iceland, June 2006.

[34] Y. Sankarasubramaniam, I. F. Akyildiz, and S. W. McLaughlin, "Energy efficiency based packet size optimization in wireless sensor networks," in *Proceedings of the 1st IEEE International Workshop on Sensor Network Protocols and Applications (SNPA '03)*, pp. 1–8, Anchorage, Alaska, USA, May 2003.

[35] I. F. Akyildiz and I. Joe, "A new ARQ protocol for wireless ATM networks," in *Proceedings of the IEEE International Conference on Communications (ICC '98)*, vol. 2, pp. 1109–1113, Atlanta, Ga, USA, June 1998.

[36] P. Lettieri and M. B. Srivastava, "Adaptive frame length control for improving wireless link throughput, range, and energy efficiency," in *Proceedings of the 17th Annual IEEE Conference on Computer Communications (INFOCOM '98)*, vol. 2, pp. 564–571, San Francisco, Calif, USA, March 1998.

[37] E. Modiano, "An adaptive algorithm for optimizing the packet size used in wireless ARQ protocols," *Wireless Networks*, vol. 5, no. 4, pp. 279–286, 1999.

[38] C. K. Siew and D. J. Goodman, "Packet data transmission over mobile radio channels," *IEEE Transactions on Vehicular Technology*, vol. 38, no. 2, pp. 95–101, 1989.

[39] J. D. Spragins, J. L. Hammond, and K. Pawlikowski, *Telecommunications: Protocols and Design*, Addison Wesley, New York, NY, USA, 1991.

[40] D. Minoli, "Optimal packet length for packet voice communication," *IEEE Transactions on Communications*, vol. 27, no. 3, pp. 607–611, 1979.

[41] ISO/IEC 14496-2:2000, Amendment 1, "Coding of Audio-Visual Objects—Part 2: Visual version 2", 2000.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2007, Article ID 97535, 8 pages
doi:10.1155/2007/97535

Research Article

Interactive Multiview Video Delivery Based on IP Multicast

Jian-Guang Lou, Hua Cai, and Jiang Li

Media Communication Group, Microsoft Research Asia, Beijing 100080, China

Received 31 October 2006; Accepted 21 December 2006

Recommended by Jianfei Cai

As a recently emerging service, multiview video provides a new viewing experience with a high degree of freedom. However, due to the huge data amounts transferred, multiview video's delivery remains a daunting challenge. In this paper, we propose a multiview video-streaming system based on IP multicast. It can support a large number of users while still maintaining a high degree of interactivity and low bandwidth consumption. Based on a careful user study, we have developed two schemes: one is for automatic delivery and the other for on-demand delivery. In automatic delivery, a server periodically multicasts special effect snapshots at a certain time interval. In on-demand delivery, the server delivers the snapshots based on distribution of user requests. We conducted extensive experiments and user-experience studies to evaluate the proposed system's performance, and found that it provides satisfying multiview video service for users on a large scale.

1. INTRODUCTION

With the rapid development of electronic and computing technology, multiview video has recently attracted extensive interest due to greatly enhanced viewing experiences. For example, a system called EyeVision [1] was employed to shoot Superbowl 2001. Other systems, such as Digital Air's Movia [2] and Wurmlin's 3D Video Recorder [3], were also proposed to capture multiview video. Later, we proposed an interactive multiview video system (IMV System) for serving real-time interactive multiview video service [4]. Unlike conventional single-view video systems, a multiview video system allows the audience to change view direction and to enjoy some special visual effects such as view switch and frozen-moment. It greatly enhances user experience in interactive and entertainment orientated applications.

As a recently emerging service, multiview video provides a new viewing experience with a high degree of freedom. However, it also brings challenges to data delivery due to the huge data amounts transmitted. Hence, an interactive unicast solution was adopted by the previous IMV system to support a high degree of interactivity. However, unicast cannot meet the requirements of an increasing number of users due to restricted network bandwidth and limited server-processing capability. Different from the conventional unicast streaming, IP multicast is a promising technology that can handle users on a large scale. Many researchers have been investigating this area in the last decade. Among efforts is that work in VoD systems [5]. Cooperating with some delivery policies, such as command batching [6, 7] and video patching [8, 9], VoD multicast systems can provide users near VoD service and keep relatively low bandwidth costs.

When IP multicast technology is used for implementing a multiview video delivery system, interactivity has become an important issue since multiview video has unique features. In this paper, we proposed a multiview video multicast system to support a large number of users and a high degree of interactivity. Based on a detailed user study, we developed two schemes, one for automatic delivery and the other for on-demand delivery. In the automatic delivery, the server periodically multicasts special effect snapshots at a certain time interval. And, in the on-demand delivery, the server delivers the snapshots based on the distribution of user requests. The proposed system was also evaluated by extensive experiments and user-experience studies.

The rest of the paper is organized as follows. In Section 2, we outline the overall structure of multicast IMV system, and we present the video delivery schemes of our conventional and special effect videos in Section 2.2. Some experimental results are presented and discussed in Section 3. In Section 5, we conclude our work.

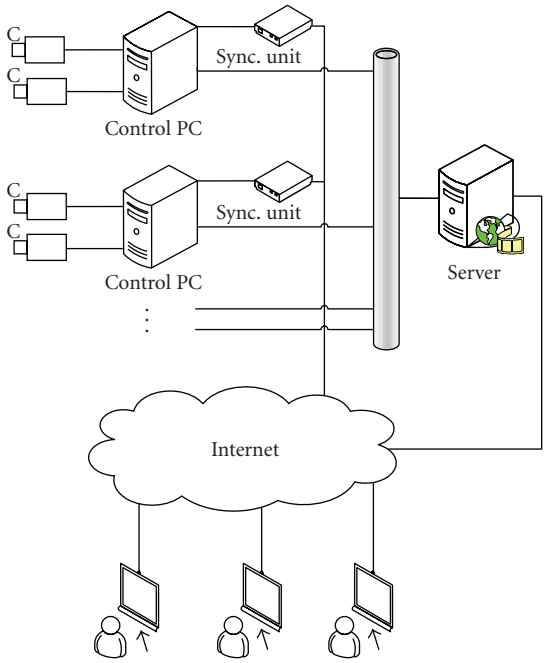


Figure 1: System architecture.

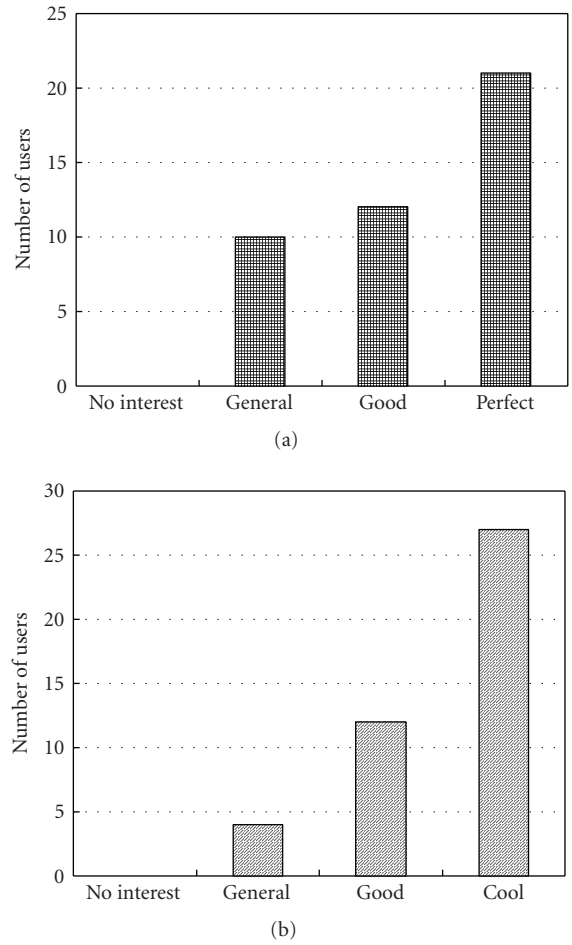


Figure 2: User study results on (a) view switching and (b) frozen moment.

2. VIDEO DELIVERY SCHEME

2.1. System overview

As described in Figure 1, our IMV system mainly consists of K video cameras, K pan-tilt units, a set of control PCs and synchronization units, a server, a network backbone, and many receivers (clients). These components can be classified into three parts [4]: capturing part, server part, and client part. Based on the synchronization signals generated by sync units, the capturing part acquires the same dynamic event simultaneously from multiple cameras with various view directions. The captured video signals are compressed in several control PCs and then sent to the server through a network backbone, for example, a gigabit Ethernet. The server part collects both the K compressed video streams from the control PCs and transcoded special effect snapshots from the transcoding servers. It then provides a multiview video service to end users.

Interactive special effects such as *frozen moment*, *view sweeping*, and *view switching* are three important features of our IMV system. In the frozen moment, time is frozen and the camera view direction rotates about a given point, while view sweeping involves sweeping through adjacent view directions while time is still moving (please refer to Figure 2 of [4] for a detailed description). View switching means that users are able to switch from one camera view direction to another as the video continues along time. Through a usability study, we found that users were highly interested in these new features. In the study, more than 40 people were invited as participants, including people with technical and nontechnical backgrounds. The results are summarized in Figure 2. Figure 2(a) indicates that about 75% of the participants consider view switching is a useful feature in an IMV system. Meanwhile, in Figure 2(b) about more than 90% of them consider that the frozen moment effect is an interesting feature.

Based on these observations, we mainly focused on how to provide multiview video features for users based on IP multicast techniques. The main challenge is to support large scale users with high interactivity using relatively low server bandwidth.

2.2. Multicast video delivery

The multiview video consists of not only conventional live videos from different views, but also the special visual effects mentioned in Section 2.1. In general, the server should broadcast videos of conventional views and special visual effects. Users should be able to enjoy special visual effects at any time, and the effects should be rendered immediately after users' actions. This application poses two requirements: QoS guaranteed transmission and low-delay interactivity. However, lower latency and more flexible action often result in higher bandwidth cost, especially when the number of views

becomes large. To handle this problem, we prepared two kinds of video streams at the server side. The first one is the conventional single-view video stream that is captured and compressed individually at the control PC. The other kind of streams is the frozen moment stream and the view-sweeping stream. These video contents are delivered through $M+N$ video multicast channels. Here, the M channels are assigned to multicast the videos from different views, while the N channels are used for delivering the special effect streams. The values of M and N depend on the available bandwidth of the server. As shown in Figure 3, each client simultaneously joins one conventional video channel and one or more special effects channels. The number of the special effects channels that a user joins depends on available downlink bandwidth. Therefore, users with higher available downlink bandwidth can join more special effects channels, and thus can enjoy the special effects with higher degree of interactivity.

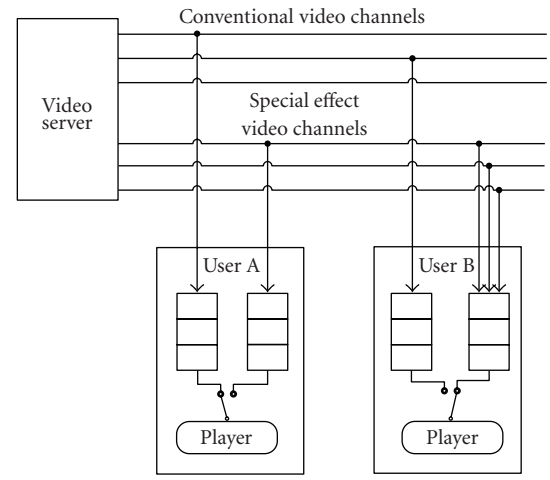


Figure 3: The overview of IP multicast for online user service.

2.2.1. Conventional video channels

One problem of designing the proposed delivery system is how to select views for the M conventional view channels. The simplest solution is that all K views are broadcast through $M = K$ channels to serve users' subscriptions. However, in real world scenarios, we found that the number of conventional view channels M is not necessarily the same as the number of views. Because of the small visual difference between two adjacent views, users are unlikely to do a switch operation between them. In our experiments, we found that $M = 6$ can usually meet user requirements for a total capture angle of 90°. It is only about 1/5 of the original view number (32 in our system). Such a sampling scheme can largely reduce the server bandwidth usage. Given the server available bandwidth, the saved bandwidth can be used to improve the interactivity of special effects.

In our system, view switching is realized by switching from the source to the destination conventional channel. The maximum latency of the view switching is $T_s + T_v$, where T_s is the time of network channel switching and T_v is the latency from the current frame to the next I frame. The value of T_v is determined by the group of picture (GOP) size when compressing the conventional view videos. Figure 4 shows two conventional video channels, view i and view j. Because each conventional video stream is compressed as I and P frames using an MPEG-liked encoder, to guarantee the smoothness of the visual experience, we can only switch one view to another at an I frame. For example, if a user sends a switching command from view i to j at time index $t_0 = t + 1$, it immediately joins channel j and receives video frames following $P_j(t+1)$ from channel j. At the same time, it still receives P frames following $P_i(t+1)$ from channel i until it leaves channel i when the next I frame $I_j(t+g)$ arrives. In our system, the GOP size is set to 30 and the video frame rate is 30 fps. Thus the maximum value of T_v is 1 second and the average value is 0.5 second. In [4], we have found that users can tolerate a relatively long, for example, 1 second, view switching latency, which is similar to the consumers attitude on the program

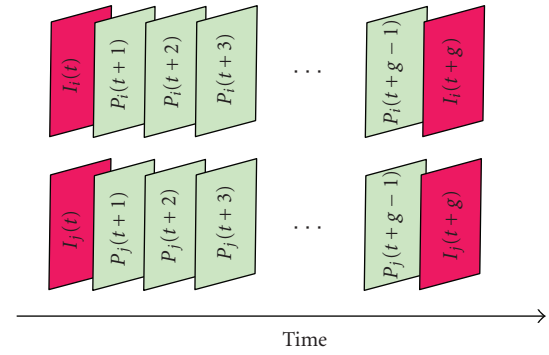


Figure 4: Conventional video channels. Here g is the value of GOP size.

switching latency in digital TV. Therefore, such a delay can meet most users' requirements.

2.2.2. Special effects channels

Due to the limited downlink bandwidth, users cannot get all the special effects snapshots in real time. Fortunately, through the user study in [4], we found that different users often have a similar sense of exciting moments in a multiview video, and they will subscribe to special effects when there is an exciting moment. Furthermore, the visual experiences of neighboring snapshots in the time domain are close to one another. This means that not all snapshots need to be sent to end users. Then the problem is that, for a given available downlink bandwidth, how to select proper special effects snapshots for end users?

For an offline multiview video, suppose that the distribution of all user subscriptions $f(t)$ on special effects snapshots are known beforehand. Then, we can find the optimal

snapshots p_i $(i = 1, \ldots, n)$ by minimizing the total differences from the snapshot that a user wants:

$$\arg\min_{p_i} \left(\sum_{i=1}^{n} \int_{\phi_i}^{\phi_{i-1}} (t - p_i)^2 f(t) dt \right). \quad (1)$$

Note that (1) is very similar to the classic scalar quantization problem. The iterative method proposed by Lloyd [10] can be used to estimate the optimal values of p_i. However, in online multiview streaming, the distribution function $f(t)$ cannot be known in advance. In other words, we are not able to determine the proper snapshots beforehand.

A simple strategy, named as automatic delivery scheme, is that the special effects channels multicast the snapshots with a fixed time interval d_s ($d_s \geq T$, $T = b/B$ is the minimal time interval of sending a snapshot that is determined by the average snapshot size b and the available bandwidth B). In the automatic delivery, all of the sent snapshots are equally distributed in the special effects channels. Obviously, the disadvantage is that the sent snapshots may not be the ones that most users subscribe, because there is no interactivity between users and the server.

To overcome the problem in the automatic delivery, we also design an on-demand delivery scheme that takes user subscriptions into consideration. In the on-demand delivery, the server collects user requests and fetches an appropriate snapshot for most users. The snapshot will be sent when both of the following formulas are satisfied:

$$\begin{aligned} C &\geq \tau \times S, \\ T_\eta &\geq T, \end{aligned} \quad (2)$$

where T_η is the time interval between the sent time of the two snapshots, C is the sum of user requests in the period of T_η, τ is a threshold ($0 \sim 100\%$), S is the total number of logon users, and $T = b/B$.

To better illustrate the process of our on-demand delivery scheme, we give an example in Figure 3. The curve is the distribution of user requests. t_0 is the sent time of the last snapshot, t_s is the snapshot ordered by a user request x, t_1 is the sent time of the current snapshot, and t_c is a proper snapshot for most users. In the on-demand delivery, the special effects video service tries to meet the interaction requirements of most users. It can dynamically adjust the sending frequency based on the number of user requests and the threshold τ. Therefore, more bandwidth cost can be saved when there are fewer requests. However, the disadvantage is that it brings extra interaction latency for the server needs to collect user requests.

To demonstrate the performance and features of the system, we carried out streaming experiments and user-experience studies. The results can help us select a proper video streaming strategy.

3. EXPERIMENTS

In this section, we describe the experiments on the performances of the automatic delivery scheme and the on-demand delivery scheme under various network conditions.

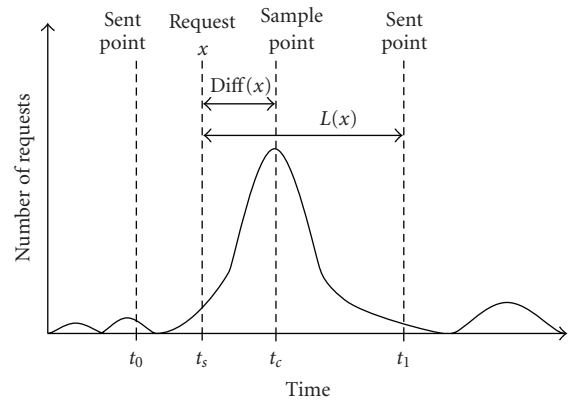


Figure 5: On-demand delivery scheme. Here, t_0 is the sent time of the last snapshot, t_s is the snapshot ordered by a user request x, t_c is the snapshot of most users, t_1 is the sent time of the current snapshot.

3.1. Performance metrics

Before the experiments, we first figured out two metrics that can be used to evaluate the system performance. Here are the definitions.

Special effects latency $D(x)$

Special effects video latency is the time interval from the moment that users send out requests to the moment that the special video starts to be played. In Figure 5, $L(x)$ is the time interval from the request time t_s to the sent time t_1. If t_n is the network RTT, the latency of the command x should be $D(x) = L(x) + t_n$, because a user sends command x at $t_s - t_n/2$, while receives the response at $t_1 + t_n/2$.

Special effects difference $\mathrm{Diff}(x)$

Special effects video difference is the time difference between the effects snapshot the server sends out and the one requested by a user. For example, in Figure 5, $\mathrm{Diff}(x)$ is the video difference of the command x from the time stamp t_s of the snapshot that a user requests to the time stamp t_c of the snapshot that the server sends out.

3.2. User-experience study

Even given the values of latency and difference, we still have no clear knowledge about whether they can meet most user requirements. Therefore, it is necessary to study user experiences on various values of latency and video difference. In this paper, we conducted a user experience study, which is formed from the feedback from 43 users after they observed the videos (including Chinese martial arts and gymnastics) with different latency and difference values. The result is shown in Figure 6, where the height of a bar represents the number of users who consider the interactivity with corresponding latency and difference as acceptable.

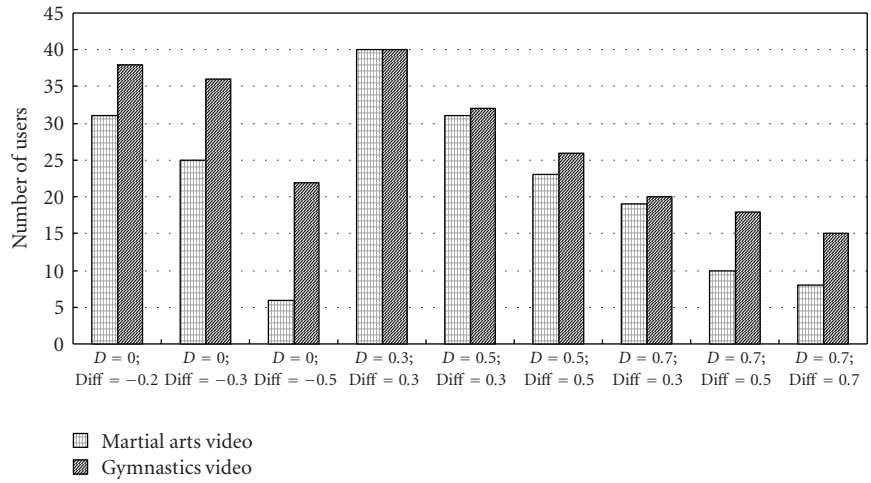


Figure 6: User study on latency and difference.

From Figure 6, we find out that more than 90% of users consider the performance of special effects video as very good when the difference and latency are set to 0.3 seconds. Less than 15% of participants can tolerate the 0.7 seconds latency and video differences. And the configuration of 0.5 seconds latency and 0.3 seconds difference is also acceptable. Most users felt that the latencies and differences in Chinese Martial Art videos are not as comfortable as they are in the gymnastics videos. This means that user responses to different video contents are slightly different. Based on the results, we find out that for a practical system, the latency and difference should be less than 0.5 and 0.3 seconds.

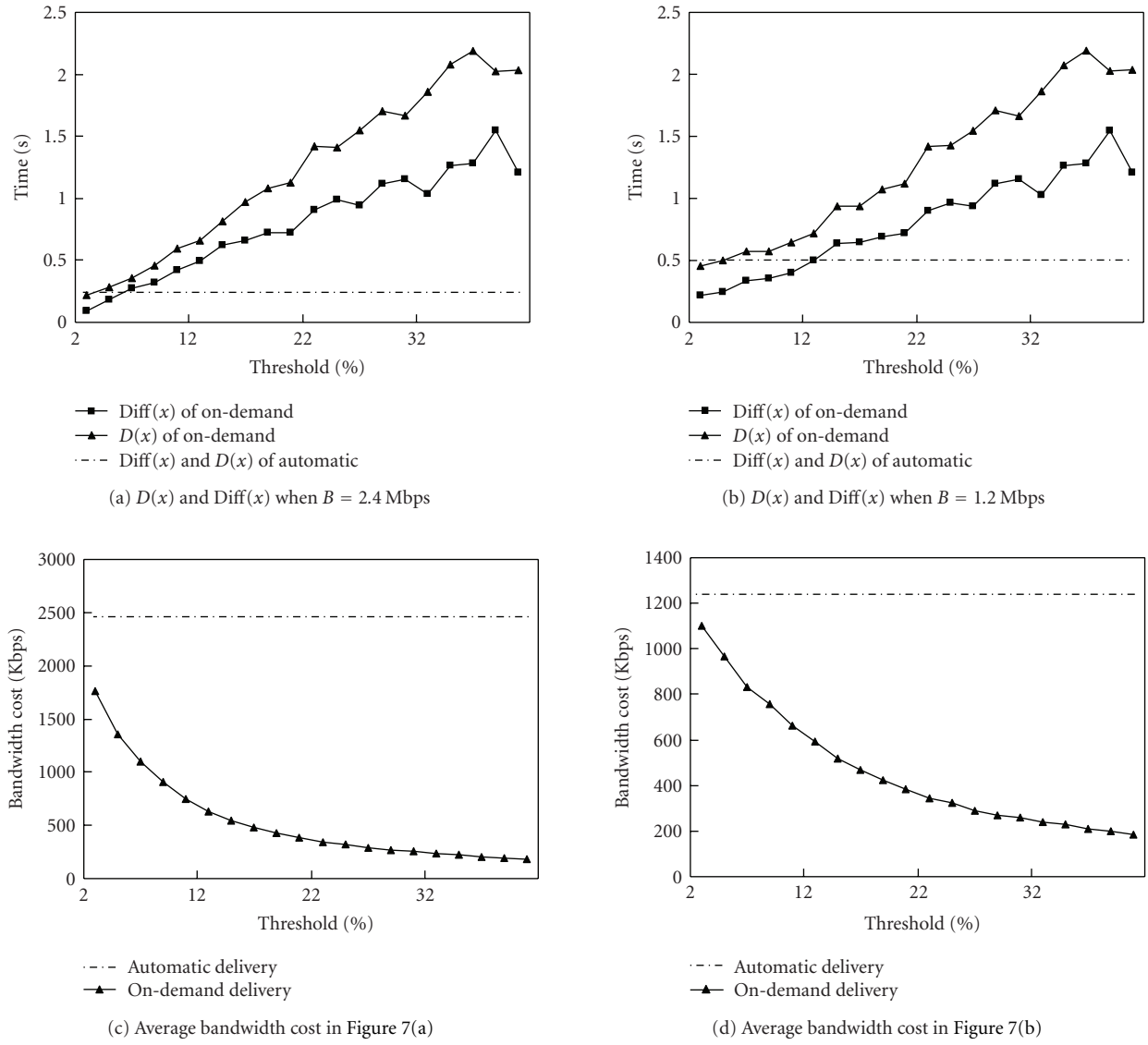
3.3. Experiments on special effects video delivery

Figure 7 shows the results of two delivery schemes with different available bandwidth. The experiments were carried out in a LAN with capacity of 100 Mbps. The run trip time (RTT) is less than 10 milliseconds, and can be neglected in our experiments. Although sometimes the available bandwidth of the LAN is large enough, we use 1.2 or 2.4 Mbps to deliver the special effects video in our experiments. In Figures 7(a) and 7(b), two straight dotted lines are the average values of $D(x)$ and $\mathrm{Diff}(x)$ in the automatic delivery scheme, while the two curves are the average values of $D(x)$ (the curve with triangle points) and $\mathrm{Diff}(x)$ (the curve with quadrate points) as the threshold increases in the on-demand delivery scheme. Figures 7(c) and 7(d) are the corresponding average bandwidth costs of the two schemes. As shown in Figure 7(a), the latency and difference of the automatic delivery and the on-demand delivery are very close when we set a very small threshold (e.g., $\tau = 5\%$). Meanwhile, Figure 7(b) shows that the on-demand delivery scheme ($\tau < 12\%$) will have a smaller difference than the automatic one when the bandwidth is relatively small (e.g., $B < 1.2$ Mbps). The reason is that the sent snapshots are selected to meet the subscriptions of most users in the on-demand delivery. Furthermore, from Figures 7(c) and 7(d), we learn that the on-demand delivery scheme can largely reduce the average downlink bandwidth request. This is because that, in the on-demand delivery, snapshots are only sent when there are user requirements in the system. It seems that if a system has large available downlink bandwidth (e.g., $B > 2.4$ Mbps), both schemes are able to meet user requirements, but the automatic scheme is better because the server does not have to manage any user request. On the other hand, the on-demand scheme will be a better choice when the downlink bandwidth is less than 2.4 Mbps, due to lower latency and differences. Finally, we want to point out that, although the results in Figure 7 come from the videos of Chinese Martial Arts, we can draw a similar conclusion from the results of gymnastics videos which are presented in Figure 8.

4. RELATED WORK

Multiview video has attracted lots of interests from both industry and academy. Most of previous research efforts focused on multiview video compression [11–14]. For example, in [15], the authors proposed a coding structure that can facilitate a free viewpoint switching operation. However, only a few works in previous research literature have discussed the multiview video delivery problem.

In [16], Kimata et al. designed a free viewpoint video system based on RTP and RTSP protocols. In their system, multiple video frames are transmitted as a single elementary stream, and several view frames that have the same time stamp are multiplexed into one RTP packet. Viewpoint prediction is adopted to reduce the average action delay. Recently, Liu et al. [17] proposed an interactive dynamic light field transmission system. In their system, each producer PC sends four compressed streams to a streaming server. Based on the subscription of each client, the server selects some streams and transmits them through a transmission channel. Unfortunately, both of the systems are based



(a) $D(x)$ and Diff(x) when $B = 2.4$ Mbps

(b) $D(x)$ and Diff(x) when $B = 1.2$ Mbps

(c) Average bandwidth cost in Figure 7(a)

(d) Average bandwidth cost in Figure 7(b)

Figure 7: Average latency and difference of a Kongfu video.

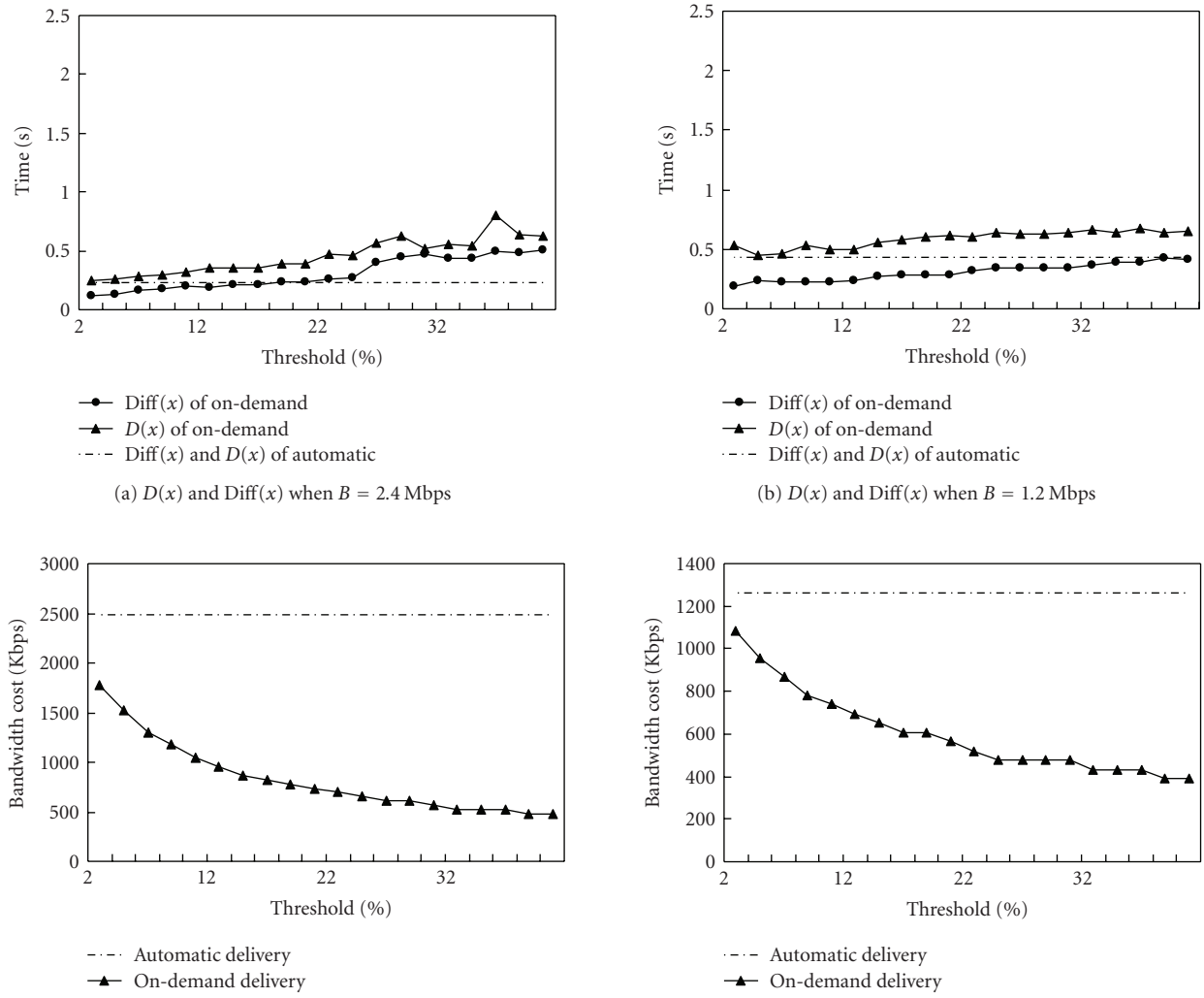
on unicast transmission schemes. Although they can provide a free viewpoint video service with high interactivity, such systems cannot scale to support a large number of users.

Matusik et al. [18] have designed a scalable system for real-time acquisition, transmission, and display of dynamic 3D scenes. In their system, each video stream is compressed individually using an MPEG2 encoder, and then is broadcasted to consumers. Unlike our real-time interactive multiview system, their system does not support users' interactivity. In their system, users cannot enjoy special visual effects.

Video streaming based on multicast has been proved useful for supporting large numbers of users. Kurutepe et al. [19] proposed a multiview video streaming system based on an application level multicast protocol, NICE [20]. Both original views and depth videos are compressed using the H.264 video codec. Clients selectively join different streams based on their available bandwidth and viewing angles. However, their system does not support the frozen moment and view sweeping effects.

5. CONCLUSION

In this paper, we propose a multiview video streaming system based on IP multicast. The multiview videos are transmitted through $M + N$ video channels. This multiple-channel scheme can support various users who have different available bandwidth. Furthermore, two multicast delivery strategies, automatic delivery and on-demand delivery, are presented and evaluated in this paper. Based on the proposed streaming schemes, our system can serve users on a large scale, and provide satisfying interactivity for most of them. Moreover, the analysis can also facilitate selecting proper streaming strategy for different multiview video applications.



(a) $D(x)$ and Diff(x) when $B = 2.4$ Mbps

(b) $D(x)$ and Diff(x) when $B = 1.2$ Mbps

(c) Average bandwidth cost in Figure 8(a)

(d) Average bandwidth cost in Figure 8(b)

Figure 8: Average latency and difference of a gymnastics video.

ACKNOWLEDGMENTS

We thank Li Zuo for his efforts during the implementation and experimental testing of the idea in this paper, and our colleague Dwight Daniels for his editing of the paper.

REFERENCES

[1] "EyeVision project," 2001, http://www.ri.cmu.edu/projects/project_449.html.

[2] "Digital movia camera systems," http://www.digitalair.com/techniques/.

[3] S. Wurmlin, E. Lamboray, O. G. Staadt, and M. H. Gross, "3d video recoder," in *Proceedings of the 10th Pacific Conference on Computer Graphics and Applications (PG '02)*, pp. 325–334, Beijing, China, October 2002.

[4] J.-G. Lou, H. Cai, and J. Li, "A real-time interactive multi-view video system," in *Proceedings of the 13th ACM International Conference on Multimedia (ACM MULTIMEDIA '05)*, pp. 161–170, Singapore, November 2005.

[5] H. Ma and K. G. Shin, "Multicast video-on-demand services," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 1, pp. 31–43, 2002.

[6] W. F. Poon and K. T. Lo,, "New batching policy for providing true video-on-demand (T-VoD) in multicast system," in *Proceedings of IEEE International Conference on Communications (ICC '99)*, vol. 2, pp. 983–987, Vancouver, BC, Canada, June 1999.

[7] A. Dan, D. Sitaram, and P. Shahabuddin, "Scheduling policies for an on-demand video server with batching," in *Proceedings of the 2nd ACM International Conference on Multimedia (ACM MULTIMEDIA '94)*, pp. 15–23, San Francisco, Calif, USA, October 1994.

[8] Y. W. Wong and J. Y. B. Lee, "Recursive patching—an efficient technique for multicast video streaming," in *Proceedings of the 5th International Conference on Enterprise Information Systems (ICEIS '03)*, vol. 1, pp. 306–312, Angers, France, April 2003.

[9] K. A. Hua, Y. Cai, and S. Sheu, "Patching: a multicast technique for true video-on-demand services," in *Proceedings of the 6th ACM International Conference on Multimedia (ACM MULTIMEDIA '98)*, pp. 191–200, Bristol, UK, September 1998.

[10] S. P. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[11] M. Siegel, S. Sethuraman, J. S. McVeigh, and A. G. Jordan, "Compression and interpolation of 3D stereoscopic and multi-view video," in *Stereoscopic Displays and Virtual Reality Systems IV*, vol. 3012 of *Proceedings of SPIE*, pp. 227–238, San Jose, Calif, USA, February 1997.

[12] X. Tong and R. M. Gray, "Coding of multi-view images for immersive viewing," in *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '00)*, vol. 6, pp. 1879–1882, Istanbul, Turkey, June 2000.

[13] G. Chen, K. Ng, and H. Wang, "A multi-view video compression scheme based on direct view synthesis," in *ISO/IEC JTC1/SC29/WG11, MPEG2003/M10025*, Brisbane, Australia, October 2003.

[14] J. Lu, H. Cai, J.-G. Lou, and J. Li, "An effective epipolar geometry assisted motion estimation technique for multi-view image and video coding," in *Proceedings of IEEE International Conference on Image Processing (ICIP '06)*, Atlanta, Ga, USA, October 2006.

[15] X. Guo, Y. Lu, W. Gao, and Q. Huang, "Viewpoint switching in multiview video streaming," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS '05)*, vol. 4, pp. 3471–3474, Kobe, Japan, May 2005.

[16] H. Kimata, M. Kitahara, K. Kamikura, Y. Yashima, T. Fujii, and M. Tanimoto, "System design of free viewpoint video communication," in *Proceedings of the 4th International Conference on Computer and Information Technology (CIT '04)*, pp. 52–59, Wuhan, China, September 2004.

[17] Y. Liu, Q. Dai, and W. Xu, "A real time interactive dynamic light field transmission system," in *Proceedings of IEEE International Conference on Multimedia Expo (ICME '06)*, pp. 2173–2176, Toronto, Ontario, Canada, July 2006.

[18] W. Matusik and H. Pfister, "3D TV: a scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 814–824, 2004.

[19] E. Kurutepe, M. R. Civanlar, and A. M. Tekalp, "Interactive transport of multi-view videos for 3DTV applications," *Journal of Zhejiang University: Science A*, vol. 7, no. 5, pp. 830–836, 2006.

[20] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (ACM SIGCOMM '02)*, vol. 32, pp. 205–217, Pittsburgh, Pa, USA, August 2002.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2007, Article ID 78913, 9 pages
doi:10.1155/2007/78913

Research Article

Scalable Island Multicast for Peer-to-Peer Streaming

Xing Jin,[1] Kan-Leung Cheng,[2] and S.-H. Gary Chan[1]

[1] *Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong*
[2] *Department of Computer Science, University of Maryland, College Park, MD 20742, USA*

Received 9 October 2006; Accepted 19 December 2006

Recommended by Guobin (Jacky) Shen

Despite the fact that global multicast is still not possible in today's Internet, many local networks are already multicast-capable (the so-called multicast "islands"). However, most application-layer multicast (ALM) protocols for streaming have not taken advantage of the underlying IP multicast capability. As IP multicast is more efficient, it would be beneficial if ALM can take advantage of such capability in building overlay trees. In this paper, we propose a fully distributed protocol called *scalable island multicast (SIM)*, which effectively integrates IP multicast and ALM. Hosts in SIM first form an overlay tree using a scalable protocol. They then detect IP multicast islands and employ IP multicast whenever possible. We study the key issues in the design, including overlay tree construction, island management, and system resilience. Through simulations on Internet-like topologies, we show that SIM achieves lower end-to-end delay, lower link stress, and lower resource usage than traditional ALM protocols.

1. INTRODUCTION

With the popularity of broadband Internet access, there has been increasing interest in media streaming services. Recently, peer-to-peer (P2P) streaming has been proposed and developed to overcome limitations in traditional server-based streaming. In a P2P streaming system, cooperative peers self-organize themselves into an overlay network via unicast connections. They cache and relay data for each other, thereby eliminating the need for powerful servers from the system. Currently, there are two types of overlays for P2P streaming: tree structure and gossip mesh. The first one builds one or multiple overlay tree(s) to distribute data among hosts. Examples include ALM protocols (e.g., Narada and NICE) and some P2P video-on-demand systems (e.g., P2Cast and P2VoD) [1–6]. The second one builds a mesh among hosts using gossip algorithms, with hosts exchanging data with their neighbors in the mesh [7–9]. The gossip-based approaches achieve high resilience to network and group dynamics. However, they have high control overhead due to data scheduling and mesh maintenance. They also have high playback delay because in the gossip mesh a host may not always find close peers as their neighbors. On the contrary, trees introduce lower end-to-end delay and are easier to maintain. Therefore, we consider a tree-based approach in this paper.

Most previously proposed ALM protocols (such as Narada, NICE, DT, Overcast, ALMI, etc.) assume that none of the routers are multicast-capable and do not take use of the underlying IP multicast capability [1, 2, 10–14]. Although global IP multicast is not available today, many local networks in today's Internet are already multicast-capable. These local multicast-capable domains, or so-called "islands," are often interconnected by multicast-incapable or multicast-disabled routers. Since IP multicast is more efficient than ALM, it would be beneficial if ALM makes use of the local multicast capabilities in building trees.

This integration is especially important for streaming applications. Let *link stress* be the number of copies of a packet transmitted over a certain physical link [1]. We have done simulations on Internet-like topologies to evaluate some representative ALM protocols. In a group of 1024 hosts, the average link stresses achieved by Narada [1], GNP-based DT [15], TAG [13], and Overcast [11] are 2.9, 2.69, 2.61, and 2.02, respectively. The maximum link stresses achieved by these protocols are 40, 24, 28, and 14, respectively. In other words, if we use Narada for streaming, each underlay link averagely needs to deliver 2.9 streams and the most loaded link has to deliver 40 streams. Considering that a single stream usually requires several hundred Kbps transmission rate, the current Internet often cannot provide enough bandwidth for the streaming. On the other hand, as IP multicast always keeps

the stresses of all the delivery links being 1, it significantly improves the delivery efficiency and reduces the bandwidth consumption. We hence propose a distributed and scalable protocol called *scalable island multicast (SIM)* that combines IP multicast with ALM for media streaming.

In SIM, hosts within an island exchange data with IP multicast. They connect across islands with unicast overlay paths. Each host first distributedly joins an overlay tree. It then detects and joins its multicast island if possible. Each island in SIM has a unique ingress host. The ingress receives streaming data from outside of the island and IP multicasts them within the island. Other hosts within the island receive data from IP multicast instead of from their parents in the overlay tree. We study the following key components in SIM.

(i) Construction of the overlay tree

We design a fully distributed host-joining mechanism for the construction of the overlay tree. A new host can iteratively ping other hosts and select a close peer with enough forwarding bandwidth as the parent.

(ii) Island management

Hosts within the same multicast domain form an island. Each island has two multicast groups: a CONTROL group and a DATA group. The control messages (e.g., for ingress selection) are only multicasted in the CONTROL group, and the streaming data are multicasted in the DATA group. We further study the ingress selection and island detection issues in detail.

(iii) System resilience

A single delivery tree may not provide good streaming quality, especially in a dynamic P2P system. We discuss two possible ways to improve system resilience: (1) build multiple trees for data delivery, and (2) use a quick recovery scheme to address temporary packet loss. In the recovery scheme, each host selects a few recovery neighbors under the constraint of the recovery deadline. Whenever a packet loss is detected, the retransmission request is sent to the recovery neighbors.

We have evaluated SIM through simulations on Internet-like topologies. Our simulation results show that SIM can efficiently combine IP multicast with ALM to achieve low end-to-end delay, low link stress, and low resource usage.

The rest of the paper is organized as follows. In Section 2, we briefly review the related work. In Section 3, we describe the data delivery mechanism in SIM. In Section 4, we discuss how to improve system resilience. In Section 5, we present illustrative simulation results. We finally conclude in Section 6.

2. RELATED WORK

We briefly review previous work on island multicast as follows. Though protocols such as mTunnel, scattercast, YOID, UMTP, AMT, and universal multicast (UM) have been proposed to combine IP multicast with ALM, many of them require special nodes (such as proxies or servers) or manual configuration for interhost connections [16–19]. SIM is fully autonomous and does not require any special or super nodes. Subset multicast (SM) also makes use of local multicast capability [20]. However, it is based on a star topology and is not scalable to large groups. The work in [21] proposes a centralized island multicast algorithm. As opposed to them, SIM is fully distributed and scalable. The work in [22] studies a distributed approach to integrate IP multicast and ALM. Each island has a leader, which identifies some ingress and egress hosts in its island for data delivery. This approach puts heavy control loads to leaders and has complicated mechanism for the management of leaders, ingress hosts, and egress hosts. SIM provides a much simpler data-delivery method and is much more implementable. In SIM, there is no leader, and there is no overhead to select egress hosts.

In HMTP, each island has a unique leader (called *designated member*, or *DM*) [23]. DMs form an overlay tree for data delivery. Each DM then IP multicasts data within its island. While HMTP imposes the responsibilities of data receiving, data forwarding, and island management on a single leader in each island, a leader has high nodal stress and heavy workload. Different from it, SIM distributes these responsibilities to different hosts. Each island in SIM has one ingress and some egress hosts. SIM hence achieves a more balanced load distribution. Furthermore, when islands are large, it is not efficient to represent each island by a single DM, where end-to-end delays depend on the locations of DMs and the selection of appropriate DMs is not easy. In SIM, we can select a close pair of hosts to connect two islands. This method is more efficient and practical. Another important difference between HMTP and SIM is the construction of the overlay tree. In HMTP, a new host starts joining the tree from the tree root in a top-down manner. The top-level hosts in the tree (i.e., those close to the root) are frequently visited by new hosts and are easily overloaded by the ping requests. It is hence not fully scalable. In SIM, each new host obtains a list of randomly selected peers at the beginning and starts joining from these hosts. The communication overhead for joining is hence distributed to all the peers.

A preliminary version of SIM has been discussed in [24]. In this paper, we further propose a loss-recovery scheme to improve streaming quality and conduct more comprehensive simulations to evaluate SIM.

3. DATA DELIVERY IN SIM

In this section, we describe the data delivery mechanism in SIM. Hosts first form a low-delay overlay tree. They then detect multicast islands and use IP multicast if possible. Note that unicast connections in SIM use TCP. But IP multicast within an island uses UDP, for TCP is not available in this case. In the following discussion, a *parent* of a host refers to the host's parent in the overlay tree, and the *source distance* of a host refers to the delay between the source and the host in the overlay tree.

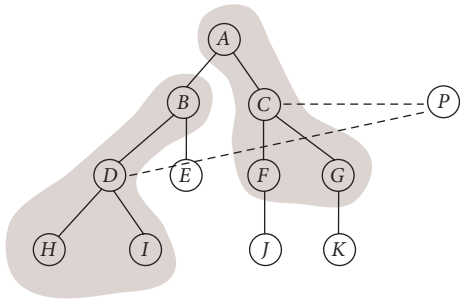


Figure 1: An example of joining the overlay tree in SIM.

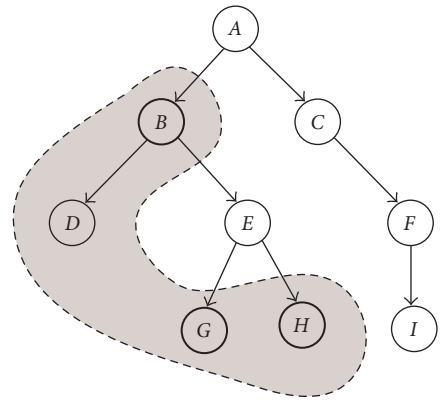


Figure 2: Ingress selection in a SIM tree.

3.1. Construction of the overlay tree

We are interested in building a tree with low end-to-end delay. Clearly, the tree construction mechanism should be distributed and scalable, and the algorithm should be simple with low setup and maintenance overhead.

In SIM, a new host first contacts a public rendezvous point (RP) to obtain a list of current hosts in the system. It pings these hosts and selects k closest ones. Then, it pings the neighbors of these selected hosts, and selects k closest ones from all the hosts (the original k hosts and their neighbors). This process is repeated until the improvement on round-trip time (RTT) is lower than a certain threshold, or the number of iterations exceeds a certain value t. At the end of the process, the new host selects from its current k closest hosts the one with enough forwarding bandwidth as its parent. If there are no qualified hosts, the new host goes back up one level to look for qualified parents.

Figure 1 shows an example of host joining in our scheme. Suppose $k = 2$ and P is a new host. P first obtains a list of hosts from the RP, say, C, D, E, F, and G. P then pings all of these hosts and selects two closest ones, say C and D. P then pings all of C's and D's neighbors. It continues selecting two closest hosts from C, D, C's neighbors (i.e., A, F, and G), and D's neighbors (i.e., B, H, and I). Such iteration stops if any of the above stopping rules is satisfied. Since the list of hosts returned by the RP is randomly generated, the communication overhead for joining is distributed to all the hosts.

If a host leaves, its children need to rejoin the tree and find new parents. A rejoining host starts rejoining from its grandparent and then acts as joining.

3.2. Integrating IP multicast

After a host joins the overlay tree, it detects its island and joins the island if any. Each streaming session has two unique class-D IP addresses for IP multicast. One is used for multicasting control messages, and the other is used for multicasting streaming data. We call the groups corresponding to these two IP addresses a *CONTROL group* and a *DATA group*, respectively.

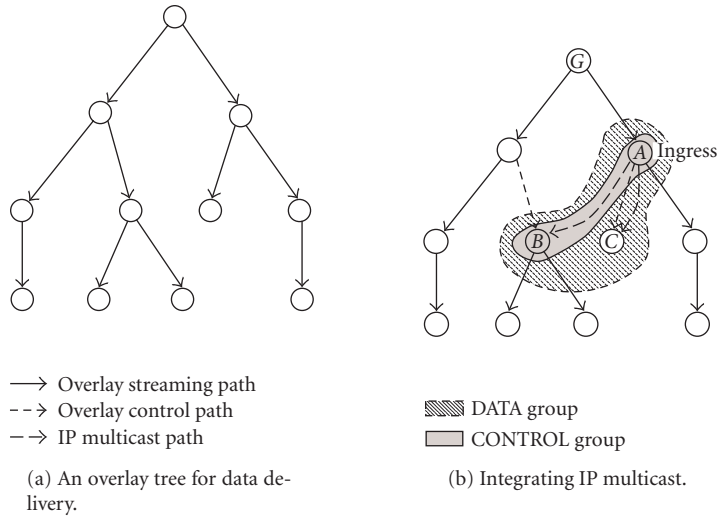
Each island has a unique ingress host, which is responsible for accepting streaming data from outside of the island and multicasting them within the DATA group. Other hosts within the island accept streaming data from IP multicast instead of overlay unicast. We call a host within the island a *border host* if its overlay parent is not within the island. Clearly, the ingress must be a border host. In SIM, border hosts (including the ingress) join both the CONTROL group and the DATA group, and nonborder hosts only join the DATA group.

3.2.1. Selection of the ingress

Proper selection of the ingress is important for efficient data distribution. If an ingress has a high source distance, all the hosts within its island will accordingly suffer high-source distances. Furthermore, not every border host can serve as an ingress. For example, in Figure 2, hosts first form an overlay tree. Later on, it is detected that B, D, G, and H are within the same island, and among them B, G, and H are border hosts. However, among the three border hosts, only B can serve as the ingress. If G becomes the ingress, all the other hosts within the island will stop receiving data from their overlay parents. When B is waiting for the data IP multicasted by G, E cannot receive any data from B. Consequently, G cannot receive any data from E. A deadlock is hence formed. Similarly, H cannot serve as the ingress.

To address these problems, we require each host to record its own source distance. The distance can be computed along the tree in a top-down manner. Namely, the source distance of a host is equal to the sum of its parent's source distance and the delay from its parent to itself. An ingress is hence selected from the border hosts of the island as the one with the minimum source distance.

An ingress host periodically multicasts *KeepAlive* messages in the CONTROL group, which contains its source distance. It also multicasts streaming data within the DATA group. Initially, the ingress of an island is the island's first joining host. The noningress border host with the smallest source distance in the island becomes the new ingress



(a) An overlay tree for data delivery.

(b) Integrating IP multicast.

Figure 3: Combining IP multicast with ALM.

if

(i) the current ingress leaves or fails (detected through the missing of *KeepAlive* messages), or
(ii) the border host has a smaller source distance than the current ingress.

This can be achieved as follows. A border host may multicast its own source distance within the CONTROL group. When a border host finds that its own source distance is smaller than the smallest source distance being multicasted, it keeps multicasting its own source distance within the CONTROL group. On the other hand, if a border host receives a message reporting a smaller source distance than its own one, it stops multicasting its own source distance. In the end, only the border host with the smallest source distance will multicast its source distance. If this source distance is smaller than that of the ingress, the corresponding border host will substitute the current ingress.

3.2.2. Island detection

The two class-D IP addresses are maintained by the RP. When a new host joins the session, it first obtains the class-D addresses and a list of already joined hosts from the RP. The new host then joins the overlay tree as described above. Afterwards, it joins the CONTROL group.

(i) If an island exists, the host will receive *KeepAlive* messages from the ingress. The host then detects whether itself is a border host. If it is, it remains in the CONTROL group and further joins the DATA group. Otherwise, it exits the CONTROL group and only joins the DATA group.

The examination of whether being a border host can be achieved as follows. The host multicasts a *BorderIdentification* message within the CONTROL group. If its parent receives the message, the parent unicasts a response message to the host using TCP. If the host does not receive any response after a certain time, it classifies itself as a border host.

A noningress host in the DATA group stops receiving streaming data from its overlay parent. Instead, it accepts data transmitted by IP multicast. The connection to its overlay parent is then only used for transmitting control messages. If this host becomes an ingress later, it resumes the overlay connection and accepts data from its parent again.

(ii) If the host does not find any island to join, it forms an island only consisting of itself and becomes the island ingress.

We show an example of data delivery with IP multicast in Figure 3. Figure 3(a) shows the overlay tree formed as described above. In Figure 3(b), hosts A, B, and C join the CONTROL group and detect that they are within the same island. A is elected as the island ingress. B is a normal border host and C is a nonborder host. Therefore, A and B stay in both the CONTROL group and the DATA group, while C only stays in the DATA group. A then accepts data from its overlay parent G, and multicasts them within the DATA group. The incoming overlay paths of B and C are then used for delivering control messages instead of media contents. If A leaves the system, B will be elected as the new ingress since it is the only border host within the island. B will then resume data delivery along its overlay path and multicast data within the island.

4. DISCUSSION ON SCHEME EXTENSION

In this section, we discuss two possible ways to improve system resilience, that is, building multiple trees and conducting packet-loss recovery.

4.1. Building multiple trees

Using a single tree may not offer satisfactory service, because, firstly, hosts in the system are heterogeneous with different

incoming and outgoing bandwidth. A host's incoming path may not be able to provide enough bandwidth for streaming. Secondly, quality degradation at a host (e.g., packet loss or host failure) affects all its descendants. In a highly dynamic P2P system, it is difficult for hosts to achieve high streaming quality with a single tree. To address these problems, we can use multiple description coding (MDC) [25] to encode streaming data into multiple descriptions and distribute the descriptions along multiple trees [5].

In MDC, data are encoded into several descriptions. When all the descriptions are received, the original data can be reconstructed without distortion. If only a subset of the descriptions are received, the quality of the reconstruction degrades gracefully. The more descriptions a host receives, the lower distortion the reconstructed data have. Therefore, the source can encode its media content into M descriptions using MDC (where M is a tunable parameter), and transmit the descriptions along M different trees. Note that a host has different descendants in different trees. The descendant of a host in one tree is usually not the host's descendant in other trees. Therefore, packet loss at a host or failure of the host only causes the loss of a single description (out of M descriptions) at each of its descendants. The system resilience is hence improved.

4.2. Packet-loss recovery

Although MDC and multiple-tree transmission can improve resilience, packets may still get lost due to background traffic or path/host failure. An efficient loss recovery mechanism is hence desired to deal with temporary packet loss.

4.2.1. Design principle

Traditional source-recovery and parent-recovery schemes have loss correlation problem and implosion problem [26]. That is, the losses of all downstream hosts are correlated upon an upstream loss. The parent of a failed host is hence likely in error, leading to low retransmission efficiency. Furthermore, such recovery leads to implosion if retransmission requests from downstream hosts are not aggregated (which is often the case for simplicity). To address these problems, lateral error recovery (LER) has been proposed [26, 27]. LER randomly divides hosts into multiple planes and independently builds an overlay tree in each plane. A host needs to identify some hosts from other planes as its recovery neighbors. Whenever a loss occurs, the host performs retransmission from its recovery neighbors. A limitation of LER is that the trees in different planes should be of similar sizes. Otherwise, a host in a small tree needs to serve as the recovery neighbor of multiple hosts in a large tree and has high workload. However, the balancing of multiple trees in a dynamic system requires high control overhead and is not easy.

We consider simplifying the selection of recovery neighbors as follows. We only use a single plane instead of multiple planes. The recovery neighbor of a host should satisfy the following requirements: (1) not in the host's subtree; (2) not the ancestor of the host; and (3) not in the same island as the host. Clearly, the loss correlation between a host and its recovery neighbor in this scheme is higher than that in LER. In LER, the path from the root to a host and the path from the root to the host's recovery neighbor are disjoint on the overlay. But this is not true in our scheme. However, our scheme does not need to balance multiple trees, thereby introducing much lower control overhead.

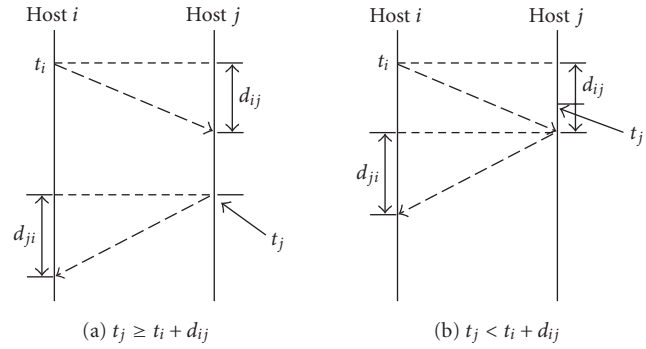


Figure 4: Time diagram for recovery neighbor selection.

Recovery neighbors are useful for improving streaming quality. For example, if the path between a host and its parent does not have enough bandwidth, the host can require missing data from one or multiple of its recovery neighbors. This enables multiple-path delivery as in gossip streaming.

4.2.2. Selection of recovery neighbors

We have listed three basic requirements for selecting recovery neighbors as above. However, to achieve quick loss recovery, more careful selection should be considered. Firstly, a streaming application usually has a certain recovery deadline δ after detecting a loss. Define *recovery latency* as the time interval from the moment a loss is detected to when the repair is received. The recovery latency should be smaller than δ. A host hence cannot select a faraway host as its recovery neighbor, for the retransmission time may exceed the recovery deadline. Secondly, the lost packet at a host is not available at an arbitrary host in the system. If the buffers of hosts have finite sizes, it may happen that the packet needed is no longer in the buffers. Even if the buffers are infinitely large and hosts cache all the data they have received, it may happen that hosts have different end-to-end delays and the requested data have not arrived at all the hosts in the system. In LER, it is assumed that the buffers of hosts are all infinitely large, which is often not true in real systems. We hence study how to select an appropriate recovery neighbor with limited buffer sizes.

Let t_i and t_j be the source distances of host i and j, respectively, where j is a recovery neighbor of i. Let d_{ij} and d_{ji} be the one-way delay from i to j and from j to i, respectively. Let B_i and B_j be the buffer sizes of i and j (in terms of time), respectively. As usual, we assume $B_x \geq \delta$ for any host x in the system. Figure 4 shows the time diagram upon a loss detected at time t_i at host i, given that the packet is transmitted from the source at time 0. When i requests a retransmission

Table 1: Requirements on host j as a recovery neighbor of host i.

Type	Condition	Recovery latency	Requirements
I	$t_j \geq t_i + d_{ij}$	$t_j + d_{ji} - t_i$	$t_j + d_{ji} - t_i \leq \delta$
II	$t_j < t_i + d_{ij}$	$d_{ij} + d_{ji}$	$d_{ij} + d_{ji} \leq \delta$, $t_i + d_{ij} \leq t_j + B_j$

from host j at time t_i, the retransmission request arrives at j at time $t_i + d_{ij}$.

Figure 4(a) shows the case where the arrival of the retransmission request is no later than the arrival of the requested packet, that is, $t_j \geq t_i + d_{ij}$. The recovery latency is hence $t_j + d_{ji} - t_i$. Clearly, the buffer size does not matter in this case, and the only requirement is $t_j + d_{ji} - t_i \leq \delta$. Figure 4(b) shows the case where the arrival of the retransmission request is later than the arrival of the requested packet, that is, $t_j < t_i + d_{ij}$. The recovery latency is hence $d_{ij} + d_{ji}$. Clearly, it is required that $d_{ij} + d_{ji} \leq \delta$. Furthermore, upon the arrival of the retransmission request, the requested packet should still be in host j's buffer, that is, $t_i + d_{ij} \leq t_j + B_j$. Table 1 summarizes these requirements on qualified recovery neighbors. Among all the qualified candidates, it is better to select the ones with high-bandwidth connections. If no qualified recovery neighbor can be found, we select from Type-I hosts the ones with low recovery latency.

The detailed selection process works as follows. As discussed in Section 3.2.1, each host has recorded its source distance (i.e., t_i and t_j). For simplicity, we assume that d_{ij} and d_{ji} are equal to half of the RTT from i to j and from j to i, respectively. A host i first joins the overlay tree for data delivery. It then contacts its ancestors within L_1 hops. In each contact, i checks the ancestor's descendants within L_2 hops. Here L_1 and L_2 are two system parameters. If the host checked satisfies the above requirements (including the three basic principles), i adds the host as a candidate of its recovery neighbor. If the requirements in Table 1 cannot be satisfied but the host is of Type-I, i adds the host to a list of Type-I hosts. After collecting a certain number of recovery neighbors and Type-I hosts, i stops the selection process. Otherwise, i increases L_1 and L_2 and keeps selecting recovery neighbors.

5. ILLUSTRATIVE SIMULATION RESULTS

In this section, we present illustrative simulation results on Internet-like topologies.

5.1. Simulation setup

We generate 10 *transit-stub* topologies with GT-ITM [28]. Each generated topology is a two-layer hierarchy of transit domains and stub domains. The transit domains form a backbone and all the stub domains are connected to the backbone. In our simulations, each topology has 4 transit domains and 280 stub domains. On average, a transit domain contains 10 routers and a stub domain contains 8 routers. Each topology consists of 2280 routers and about 11 000 links. A group of 1024 hosts are randomly put into the network. A host is connected to a unique stub router with 1 millisecond delay, while the delays of core links are given by the topology generator. Link bandwidth is set as follows: a backbone link (at least one end-point is a transit router) can support 8 concurrent media streams, and a nonbackbone link can support 3–6 concurrent media streams. The distribution of islands is set as follows: from the stub domains that consist of at least one host, we randomly select some and set them to be multicast-capable. Define *multicast ratio* θ as the ratio of the number of multicast-capable stub domains to the number of stub domains that consist of at least one host, and define *island size* S as the number of stub domains in an island. Note that in the real Internet, routers in a multicast island are often close to each other. Therefore, in our simulations, only the stub domains connected to the same transit router can be within the same multicast island.

The SIM parameters are set as follows. Each new host obtains a number of (at most 10) randomly selected hosts from the RP when joining. A new host repeats the ping iterations for at most 6 times and in each iteration pings at most 10 hosts (i.e., $t = 6$ and $k = 10$). We further implement two tree-based ALM protocols for comparison, that is, Narada [1] and Overcast [11]. Narada is one of the pioneering ALM protocols and aims at building a low-delay overlay tree. Its performance can serve as the benchmark. In Narada, each host has a degree bound according to its edge bandwidth. Overcast aims at constructing a tree with high bandwidth, which achieves low stresses on links.

We use the following metrics to evaluate the protocols.

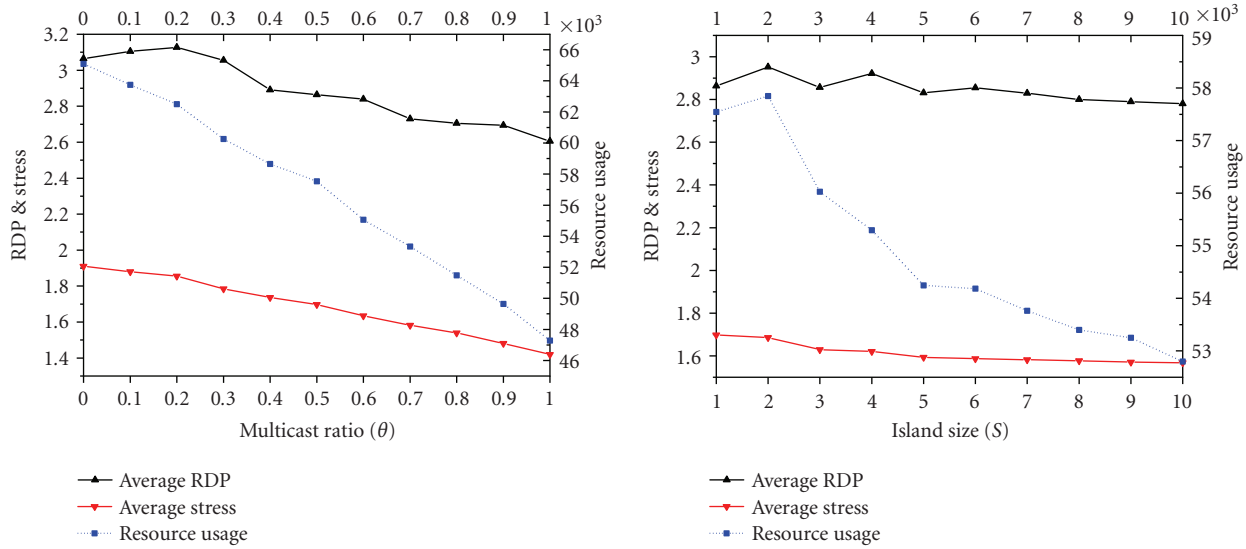
(i) *Relay delay penalty (RDP)*: defined as the ratio of the overlay delay from the source to a given host to the delay between them along the shortest unicast path [1].

(ii) *Link stress*: defined as the number of copies of a packet transmitted over a certain physical link [1].

(iii) *Resource usage*: defined as $\sum_{i=1}^{L} d_i \times s_i$, where L is the number of links active in data transmission, d_i is the delay of link i, and s_i is the stress of link i [1]. Resource usage is a metric of the network resource consumed in data delivery.

5.2. Results

Figure 5 shows the performance of a SIM tree with different parameter settings. Figure 5(a) shows the result of tuning the multicast ratio θ. As θ increases, the average RDP first increases and then decreases. This is because two hosts within the same multicast island are not necessarily close to each other, therefore selecting a host from other domains as the parent may introduce lower delay than simply receiving IP multicast packets from the ingress. However, when most hosts in the system are within multicast islands, the distance between a host and its ingress is often not large. The average RDP is hence low. On the other hand, both the average stress and the resource usage decrease as θ increases. When θ increases from 0 to 1.0, the average stress and the resource usage are reduced by 25.7% and 27.4%, respectively. Therefore,



(a) Performance of SIM with different multicast ratios ($S = 1$).

(b) Performance of SIM with different island sizes ($\theta = 0.5$).

Figure 5: Performance of SIM with different parameter settings.

the utilization of IP multicast in ALM can efficiently reduce the loads on links and the overall bandwidth consumption.

Figure 5(b) shows the results of tuning the island size S. As S increases, the average RDP fluctuates around 2.85. Averagely, a maximum island in our simulations can contain 10 stub domains (i.e., $S = 10$). In fact, from our simulation results, the average number of hosts in an island when $S = 10$ is around 30. Therefore, the islands formed are small as compared to the group size and such small islands cannot significantly reduce the end-to-end delay. We expect lower RDP when the islands are larger. Different from the average RDP, the average stress decreases as S increases. Clearly, IP multicast always achieves an average stress of 1.0. The more IP multicast paths in the delivery tree, the lower average stress. On the other hand, the resource usage first slightly increases and then decreases when S increases. It shows that the penalty in delay exceeds the improvement in stress when S increases from 1 to 2. When S continues increasing, the improvement in stress leads to the reduction in the resource usage.

Figure 6 compares the performance of different ALM protocols. We select a set of (θ, S) combinations for SIM. Figure 6(a) shows the average RDP achieved by different protocols. Overcast has the highest RDP among the protocols. This is because it only aims at improving tree bandwidth and does not optimize the tree in terms of end-to-end delay. Furthermore, to reserve bandwidth for future hosts, each host in Overcast is inserted into the tree as far from the source as the bandwidth constraint allows. It hence performs poorly in terms of RDP. SIM$(0,*)$ shows the result with no multicast islands. It performs slightly worse than Narada. When θ is 0.5, the average RDP of SIM is lower than that of Narada. Furthermore, when $\theta = 1$ and $S = 10$, all the hosts are within islands and the average RDP of SIM is significantly reduced.

Figure 6(b) compares the average stress achieved by different protocols. Narada has the highest average stress. Overcast targets maximizing bandwidth and accordingly minimizes link stress. It hence has lower average stress. SIM shows very low average stress, even in the absence of multicast islands. With the increase of θ and S, the average stress of SIM decreases. When $\theta = 1$ and $S = 10$, the average stress is only 1.14, which is very close to the optimal value of 1.0. The maximum stresses achieved by the protocols show similar trends as the average stresses (as shown in Figure 6(c)). The utilization of IP multicast can efficiently reduce the maximum stress of SIM, from 19.4 (when $\theta = 0$) to 7.5 (when $\theta = 1$ and $S = 10$). Clearly, lower stresses indicate lighter loads on links and hence higher transmission rates. From the figures, SIM can achieve low stresses and is efficient for streaming applications.

Figure 6(d) shows the resource usage achieved by different protocols. Since Narada has the highest stress and Overcast has the highest RDP, their resource usage is high. SIM achieves good tradeoff between RDP and link stress. Its resource usage is significantly lower than Narada and Overcast.

6. CONCLUSION

Traditional ALM protocols only make use of unicast connections to form delivery trees and have not fully taken advantage of the local multicast capabilities. In this paper, we propose a fully distributed multicast protocol (called SIM) for media streaming which combines IP multicast with ALM. Hosts in SIM can distributedly detect multicast domains and use IP multicast if possible. Simulations results show that it can achieve low end-to-end delay, low link stress, and low resource usage.

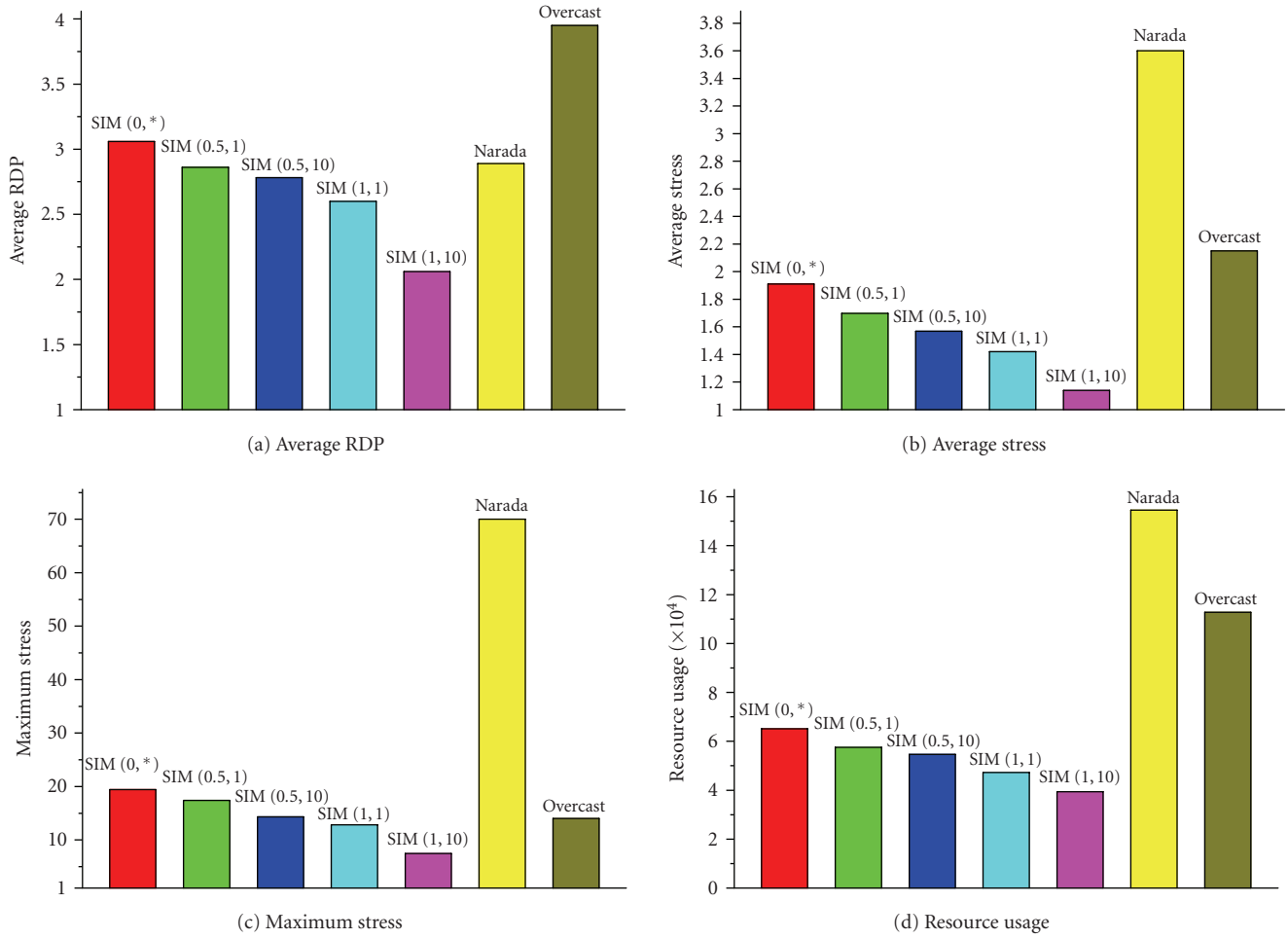


Figure 6: Performance comparison of different ALM protocols.

ACKNOWLEDGMENTS

This work was supported, in part, by Direct Allocation Grant (DAG05/06.EG10) and the Innovation and Technology Commission of the Hong Kong Special Administrative Region, China (GHP/045/05).

REFERENCES

[1] Y.-H. Chu, S. G. Rao, S. Seshan, and H. Zhang, "A case for end system multicast," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1456–1471, 2002.

[2] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, "Scalable application layer multicast," in *Proceedings of the Annual Conference of the Special Interest Group on Data Communication (SIGCOMM '02)*, pp. 205–217, Pittsburgh, Pa, USA, August 2002.

[3] Y. Guo, K. Suh, J. Kurose, and D. Towsley, "P2Cast: peer-to-peer patching scheme for VoD service," in *Proceedings of the 12th International World Wide Web Conference (WWW '03)*, pp. 301–309, Budapest, Hungary, May 2003.

[4] T. T. Do, K. A. Hua, and M. A. Tantaoui, "P2VoD: providing fault tolerant video-on-demand streaming in peer-to-peer environment," in *Proceedings of IEEE International Conference on Communications (ICC '04)*, vol. 3, pp. 1467–1472, Paris, France, June 2004.

[5] M. Castro, P. Druschel, A.-M. Kermarrec, A. Nandi, A. Rowstron, and A. Singh, "SplitStream: high-bandwidth multicast in cooperative environments," in *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03)*, pp. 298–313, Lake George, NY, USA, October 2003.

[6] D. Kostić, A. Rodriguez, J. Albrecht, and A. Vahdat, "Bullet: high bandwidth data dissemination using an overlay mesh," in *Proceedings of the 19th ACM Symposium on Operating Systems Principles (SOSP '03)*, pp. 282–297, Lake George, NY, USA, October 2003.

[7] X. Zhang, J. Liu, B. Li, and T.-S. P. Yum, "CoolStreaming/DONet: a data-driven overlay network for peer-to-peer live media streaming," in *Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '05)*, vol. 3, pp. 2102–2111, Miami, Fla, USA, March 2005.

[8] M. Zhou and J. Liu, "A hybrid overlay network for video-on-demand," in *Proceedings of IEEE International Conference on Communications (ICC '05)*, vol. 2, pp. 1309–1313, Seoul, Korea, May 2005.

[9] GridMedia. http://www.gridmedia.com.cn/.

[10] J. Liebeherr, M. Nahas, and W. Si, "Application-layer multicasting with Delaunay triangulation overlays," *IEEE Journal on Selected Areas in Communications*, vol. 20, no. 8, pp. 1472–1488, 2002.

[11] J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and J. W. O'Toole, "Overcast: reliable multicasting with an overlay network," in *Proceedings of the 4th Symposium on Operating System Design and Implementation (OSDI '00)*, pp. 197–212, San Diego, Calif, USA, October 2000.

[12] D. Pendarakis, S. Shi, D. Verma, and M. Waldvogel, "ALMI: an application level multicast infrastructure," in *Proceedings of the 3rd USENIX Symposium on Internet Technologies and Systems (USITS '01)*, pp. 49–60, San Francisco, Calif, USA, March 2001.

[13] M. Kwon and S. Fahmy, "Topology-aware overlay networks for group communication," in *Proceedings of the 12th International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '02)*, pp. 127–136, Miami, Fla, USA, May 2002.

[14] X. Jin, Y. Wang, and S.-H. Gary Chan, "Fast overlay tree based on efficient end-to-end measurements," in *Proceedings of IEEE International Conference on Communications (ICC '05)*, vol. 2, pp. 1319–1323, Seoul, Korea, May 2005.

[15] W.-C. Wong and S.-H. Gary Chan, "Improving delaunay triangulation for application-level multicast," in *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM '03)*, vol. 5, pp. 2835–2839, San Francisco, Calif, USA, December 2003.

[16] P. Parnes, K. Synnes, and D. Schefström, "Lightweight application level multicast tunnelling using mTunnel," *Computer Communications*, vol. 21, no. 15, pp. 1295–1301, 1998.

[17] Y. Chawathe, *Scattercast: an architecture for internet broadcast distribution as an infrastructure service*, Ph.D. thesis, University of California, Berkeley, Calif, USA, 2000.

[18] P. Francis, P. Radoslavov, R. Lindell, and R. Govindan, "Your Own Internet Distribution YOID," http://www.isi.edu/div7/yoid/, 2002.

[19] R. Finlayson, "The UDP multicast tunneling protocol," draft-finlaysonumtp-09.txt, November 2003.

[20] J. Park, S. J. Koh, S. G. Kang, and D. Y. Kim, "Multicast delivery based on unicast and subnet multicast," *IEEE Communications Letters*, vol. 5, no. 4, pp. 181–183, 2001.

[21] K.-L. Cheng, K.-W. Cheuk, and S.-H. Gary Chan, "Implementation and performance measurement of an island multicast protocol," in *Proceedings of IEEE International Conference on Communications (ICC '05)*, vol. 2, pp. 1299–1303, Seoul, Korea, May 2005.

[22] K.-W. R. Cheuk, S.-H. Gary Chan, and J. Y.-B. Lee, "Island multicast: the combination of IP multicast with application-level multicast," in *Proceedings of IEEE International Conference on Communications (ICC '04)*, vol. 3, pp. 1441–1445, Paris, France, June 2004.

[23] B. Zhang, S. Jamin, and L. Zhang, "Host multicast: a framework for delivering multicast to end users," in *Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '02)*, vol. 3, pp. 1366–1375, New York, NY, USA, June 2002.

[24] X. Jin, K.-L. Cheng, and S.-H. Gary Chan, "SIM: scalable island multicast for peer-to-peer media streaming," in *Proceedings of IEEE International Conference on Multimedia & Expo (ICME '06)*, pp. 913–916, Toronto, Canada, July 2006.

[25] V. K. Goyal, "Multiple description coding: compression meets the network," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 74–93, 2001.

[26] K.-F. S. Wong, S.-H. Gary Chan, W.-C. Wong, Q. Zhang, W.-W. Zhu, and Y.-Q. Zhang, "Lateral error recovery for application-level multicast," in *Proceedings of the 23rd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '04)*, vol. 4, pp. 2708–2718, Hong Kong, March 2004.

[27] W.-P. K. Yiu, K.-F. S. Wong, S.-H. Gary Chan, et al., "Lateral error recovery for media streaming in application-level multicast," *IEEE Transactions on Multimedia*, vol. 8, no. 2, pp. 219–232, 2006.

[28] E. W. Zegura, K. L. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *Proceedings of the 15th Annual Joint Conference of the IEEE Computer and Communications Societies. (INFOCOM '96)*, vol. 2, pp. 594–602, San Francisco, Calif, USA, March 1996.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2007, Article ID 62094, 7 pages
doi:10.1155/2007/62094

Research Article

Scalable Video Streaming Based on JPEG2000 Transcoding with Adaptive Rate Control

Anthony Vetro,[1] Derek Schwenke,[1] Toshihiko Hata,[2] and Naoki Kuwahara[2]

[1] *Mitsubishi Electric Research Labs, Cambridge, MA 02139, USA*
[2] *Mitsubishi Electric Corporation, Amagasaki, Hyogo 661-8661, Japan*

Received 1 November 2006; Accepted 19 December 2006

Recommended by Guobin (Jacky) Shen

This paper describes a scalable video streaming system based on JPEG2000 with various modes of streaming. A core function of the proposed system is a low-complexity transcoding technique that adapts the quality and resolution of the scene based on available bandwidth. One key feature of this technique that is important for surveillance applications is that interesting regions of the scene are assigned higher quality than background regions. To cope with varying network conditions, we also present a rate control algorithm that adaptively transcodes stored JPEG2000 frames. The proposed algorithm is designed to improve overall quality over a uniform rate control method by increasing bandwidth utilization, while satisfying buffer constraints and maintaining consistent quality over time. Simulation results confirm the effectiveness of the proposed system and rate control algorithm in terms of both objective measures and subjective evaluation. Complexity is also evaluated.

1. INTRODUCTION

The JPEG2000 standard [1] is becoming an increasing popular coding format for a variety of applications that require efficient storage and scalable transmission of images and video. One application that is beginning to make use of this coding format is surveillance, where such features are particularly attractive for networked cameras and digital video recorders. Besides enabling access to different quality layers and resolution levels, JPEG2000 is also capable of providing access to regions of interest, which could help to significantly alleviate bandwidth requirements while still providing high quality to important regions of the scene.

Several ROI coding techniques for JPEG2000 images have been proposed in the recent years. These methods can be classified into two categories: static and dynamic ROI coding. In static ROI coding, the ROI is selected and defined during the encoding process. Once the ROI is encoded, it can no longer be changed. Such methods include a general wavelet coefficient scale up scheme [2], the max-shift method [3], bitplane-by-bitplane shift [4], and partial significant bitplane shift [5]. The main problem with such static schemes is that they are not suitable for interactive environments in which the ROI is defined after encoding. On the other hand, in dynamic ROI coding, the ROI is defined during the decoding process or during progressive transmission. One dynamic ROI coding method is described in [6]. This method allows for the definition of ROI in an interactive environment and handles the ROI by dynamically inserting layers. However, the dynamic layer insertion in this scheme reencodes the packet header, which requires rate-distortion recalculation and is an undesirable for real-time applications.

In this paper, we describe a video surveillance system based on JPEG2000 that allows for transmission of the scene over limited bandwidth networks. In our system, an image sequence is encoded and stored as a JPEG2000 bitstream, and then the stored images are efficiently transcoded in the compressed domain using a novel low-complexity adaptation technique that replaces data packets corresponding to higher quality layers with empty packets. This technique supports ROI-based transcoding, where the ROIs are transmitted with higher quality than the background. The proposed technique overcomes the drawbacks of prior dynamic ROI methods in that it does not require the packet headers to be reencoded.

This paper also addresses the problem of rate allocation to each frame. One straightforward method, which will be referred to as uniform rate control and is used as a reference in this work, is to allocate an equal amount of rate to each frame based on available channel bandwidth. The obvious

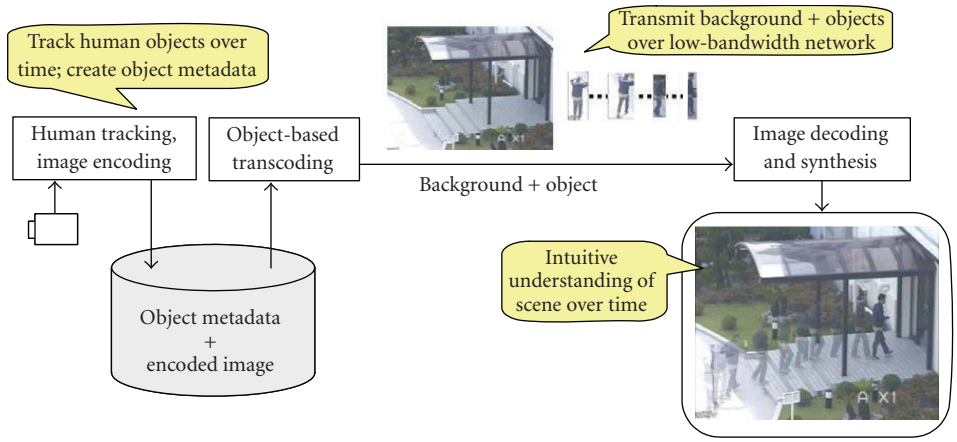


Figure 1: Object-aware streaming system for surveillance.

drawback of this method is that it is not adaptive to the scene contents. Also, since there is a fixed set of rate points that could be achieved by the transcoder, which depends on the rate allocated to each quality layer and other transcoding parameters such as output resolution level and ROI, it is very likely that the available bandwidth is not fully utilized. We propose a rate control technique that is adaptive to scene contents. The proposed rate allocation is designed to improve overall quality over the uniform rate control method by utilizing more of the available bandwidth, while satisfying buffer constraints and maintaining consistent quality over time.

The rest of this paper is organized as follows. In the next section, we provide an overview of our system including a description of the core transcoding techniques. In Section 3, the proposed rate control algorithm is presented. Experimental results are described in Section 4, including both objective and subjective evaluation, as well as complexity evaluation. Concluding remarks are given in Section 5.

2. TRANSCODING SYSTEM

In the following, we provide an overview of the key components of our object-aware surveillance system, including the main functionality of the transcoder.

2.1. Object-aware surveillance system

A diagram of the object-aware surveillance system is shown in Figure 1. The input video is analyzed and encoded. In this system, the analysis performs object detection, for example, for faces or humans. A bounding box with coordinates relative to the image coordinates is stored as the object metadata. The encoding performed using JPEG2000 and the corresponding image files are also stored. In order to enable streaming over low-bandwidth networks, object-based transcoding is employed to reduce the rate required for transmission. The operation of the transcoder will be elaborated on below. At the receiving end, the image data, which may be in the form of background and object data, is decoded and displayed.

There are various methods that the transcoded data could be streaming. In one method, which we refer to as frame-by-frame (FF), the spatial quality of ROIs and background are controlled in a frame-by-frame manner. For each region, a part of the encoded data with higher quality than a specified value is removed. FF transmits a background every frame, so the spatial quality is lower than that of the other methods described below, but changes in the background can be seen dimly. This method does not require synthesis of the decoded images for display and its implementation is simple.

A second method of streaming is to transmit the ROI successively with an occasional background refresh (BR). With this method, the transcoder mainly controls the temporal quality of ROIs considering a small occasional overhead for the background. First, ROI images and a background image with high spatial quality are transmitted. After that, only ROI images with high spatial quality are transmitted. At the receiver, the background image is decoded and held in a working memory. The successive ROI images are decoded and superimposed on the background. The background is renewed with lower frame rate according to its importance and available network bandwidth. BR does not transmit a background every frame, therefore the subjective quality is fine in low-bit rate. The overall bit rate can be significantly lowered with a low-quality background.

A third method that we introduce is mosaic streaming (MS), which is similar to BR mode, but it superimposes successive ROI images on a background in a mosaic style. Human behavior can be understood immediately and intuitively in a mosaic image, and it is very effective for behavior analysis and scene browsing as well as efficient to transmit the surveillance video over a narrow band network.

2.2. JPEG2000 transcoder

A more detailed look into the operations of the transcoder is given in Figure 2. The transcoding is invoked to satisfy

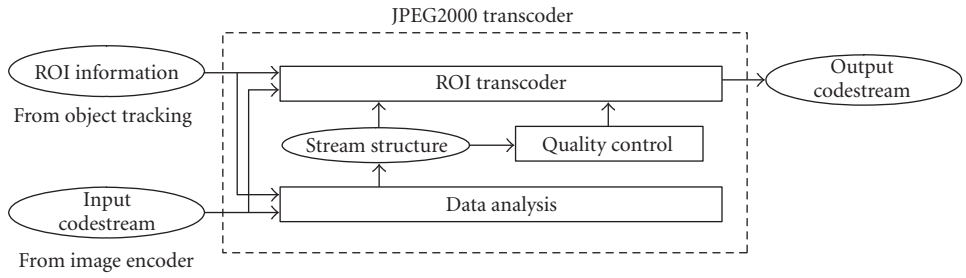


Figure 2: JPEG2000 transcoder.

network and display constraints, yielding an output code stream based on ROI information, and configuration settings. In the following, we describe three main components of our JPEG2000 transcoder including data analysis, ROI transcoding and quality control.

The data analysis module is responsible for extracting indexing information about the structure of the code stream. It is essentially a low-complexity parser that analyzes the packet header for each quality layer, resolution level, and component. A multiple-dimensional array is used to store the packet information, which indicates the byte position, header length and body length for each packet. Since this partial decoding operates on the packet header only without performing entropy arithmetic decoding for code blocks, the computational complexity is very low.

Our transcoder supports reduction of spatial resolution and quality layers. We focus mainly on quality layer reduction considering the ROI information. Given a set of ROI coordinates, we perform ROI transcoding by replacing packets at high-quality layers that are associated with the background of the scene with empty packets as defined by the JPEG2000 standard. This is an effective method for reducing the rate of the overall code stream while retaining the quality of important objects and keeping the complexity low.

The number of quality layers for the background and ROI are determined by the quality control module. In our previous work [7], the quality layers were set manually. In the next section, we describe an adaptive rate control algorithm that determines the quality layers based on target rate, buffer occupancy, and ROI information.

3. RATE CONTROL

The proposed rate control algorithm determines the rate allocation for the current frame based on the target rate, buffer occupancy, and ROI information. Given the bytes allocated to a frame, the transcoder determines quality layers for background and ROI. In the following, we describe the variable rate allocation, a frame skipping technique, as well as a quality stabilization algorithm.

3.1. Variable rate allocation

In the uniform rate control method, a fixed rate, $T_f = R/F$, is allocated to each frame, where R is the target rate and F is the output frame rate. To avoid overshooting the target rate, the quality layers in the transcoded output are chosen so as not to exceed the given budget. In our current system, we choose the quality layers for background and ROI in a systematic manner based on byte counts from the data analysis. We first assign the minimum quality to the background and ROI. The ROI quality is then successively increased. Finally, additional quality layers are added to the background. The main drawback of this uniform rate allocation approach is that it will typically underutilize the available bandwidth for a given stream because the quality layers can only provide a discrete set of rate points.

In the proposed rate control algorithm, we allocate rate nonuniformly to each frame and introduce a buffer to absorb the variations in allocated rate to each frame. The rate allocation to each frame is determined according to the following:

$$T_v = T_{\max} \cdot \max\left[0, \min\left(1, 1 - \alpha^2\right)\right], \tag{1}$$

where $T_{\max}$ sets the upper limit on the variable rate assigned to any frame and is $2T_f$ in our current system, and α is a buffer occupancy parameter that is a function of the buffer occupancy, B, the buffer size, B_s, and a safety margin, γ, with typical values in the range $[0.05, 0.25]$. The buffer occupancy parameter is given by

$$\alpha = \frac{B}{B_s \cdot (1 - \gamma)}. \tag{2}$$

When the buffer occupancy is near the upper margin, α tends towards unity, and a lower rate will be allocated to the current frame. Higher rate is allocated to the current frame when the buffer occupancy is near empty. In the next subsection, we will see how this behavior plays an important role in balancing the spatiotemporal quality tradeoff when frame skipping is employed.

3.2. Frame skipping

When frame skipping is enabled, periodic frames with no ROI defined may be skipped. The rationale behind this strategy is twofold. First, we aim to empty the buffer when there is no ROI to allow greater bandwidth for future frames that contain ROI. Second, we aim to improve the quality of the non-ROI image, which is possible since we could assign more bytes to an image sequence with a reduced frame rate.

With this strategy, frames are skipped to drive the buffer level towards its lower margin. When the buffer reaches this level, frames will no longer be skipped, and since the buffer is nearly empty, these frames are allocated a rate close to $T_{\max}$.

To state the skip condition more precisely, a frame is skipped when the following condition is true:

$$(\alpha > \gamma) \quad \& \quad (\tau < \tau_{\max}), \tag{3}$$

where τ is the interval of successive non-ROI frame skips, and $\tau_{\max}$ is the maximum frame skip interval.

3.3. Quality stabilization

The key objective of the quality stabilization is to establish a period in which the quality layers will be held stable, thereby avoiding unnecessary oscillation or frequent changes in quality. Depending on the available buffer size, the typical window period, ω, will be several frames.

Let Q_p denote the set of determined quality layers for the previous frame, Q_i the set of quality layers for the current frame i with rate allocated according to (1), and ω_c be a window counter that is reset when either the counter reaches the window period or a new set of quality layers for the current frame are determined. With quality stabilization enabled, the set of quality layers would be assigned according to

$$Q = \begin{cases} Q_p, & B_s \cdot \gamma < B < B_s \cdot (1-\gamma) \ \& \ (\omega_c < \omega), \\ Q_i, & \text{otherwise.} \end{cases} \tag{4}$$

With the above, the previous set of quality layers will be used for the current frame when the buffer is not in danger or overflow or underflow and the window counter is less than the window period.

4. EXPERIMENTAL RESULTS

To demonstrate the effectiveness of our transcoding system and the corresponding techniques that have been proposed, we first evaluate the various streaming methods that have been introduced and discuss their pros and cons. We then describe the performance of the proposed rate control algorithm compared to uniform rate allocation. Finally, we provide an analysis of the complexity.

As input data for all experiments, we use an image sequence with 1467 frames and at a frame rate of 7.5 fps. Each frame of the image sequence is a full color image (4 : 4 : 4) that is JPEG2000 encoded with 4 quality layers and 3 resolution levels with LRCP progression. The precinct sizes are set as $\{64\times64, 32\times32, 16\times16\}$, and the rate for each quality layer is set as $\{1.0, 0.5, 0.25, 0.125\}$. Each compressed frame is approximately 38 KB, yielding an overall bit-rate of 2.3 Mbps.

4.1. Evaluation of streaming methods

Various methods of streaming are illustrated in Figure 3. The original image, which is approximately 38 KB, is shown in Figure 3(a). In the following, we evaluate the quality of each setting and the required bandwidth.

Figures 3(c)–3(e) show examples of transcoded images with Qroi = 3 and Qbg = 0, 1, 2, respectively. The image with Qbg = 0 has very noticeable degradation in the background including degradation around the precinct boundaries. The data size is 9.4 KB or 25% of the input image size. The quality with Qbg = 1 is not so good, but better than that of Qbg = 0. The data size of this result is 13.0 KB or 34% of the original size. In the image with Qbg = 2, the background looks a little less sharp than the original and it is hardly noticeable. The data size is 20.9 KB, which corresponds to 55% of the original size. Observing that the transcoded image sequences are moving pictures, visual changes over time in the background with Qbg = 0 and 1 are very noticeable.

Figure 3(b) shows an example of ROI image with Qroi = 3 that is used in BR and MS streaming modes. The data size is only 6.0 KB or 16% of the original. It is very useful for a mobile phone because of its narrowband transmission and low-resolution display. The frame rate of the background depends on its importance and the available bandwidth. For example, when the frame rate of the ROI is 7.5 fps and that of background is 1 fps, the total bit rate is 644 Kbps or 29% of the original image sequence.

Figure 3(f) shows an example of a mosaic image in which ten ROIs with 1.5 fps are superimposed on the background. The walking trajectory is understood intuitively and immediately.

4.2. Rate control simulations

To test the performance of the proposed adaptive rate control algorithm, we perform a number of experiments with varying configurations and buffer size. We evaluate both objective and subjective quality and use the uniform rate control method as a benchmark.

For the purpose of this study, we define the following objective measures.

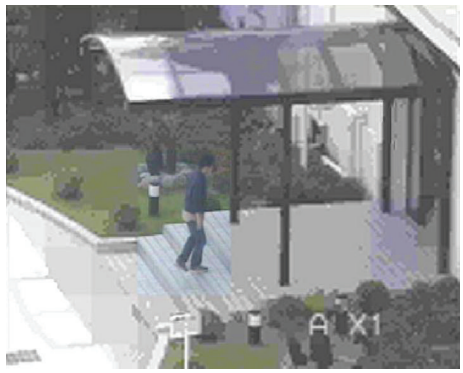
(i) BWU: bandwidth utilization defined as the ratio of transcoded output bits to the target rate.
(ii) ΔP0: number of changes in background quality.
(iii) Avg0: average background quality.
(iv) Avg1: average ROI quality.

As one would expect, achieving higher bandwidth utilization will generally increase overall quality. Also, minimizing the fluctuation in quality layers over time also has a positive impact on perceptual quality. It is noted that the average ROI quality is not as relevant as the average background quality since the ROI typically receives high quality regardless of the rate control method or algorithms used. In addition to the above metrics, we also report the number of frames skipped and MSE. It is noted that the MSE for skipped frames is computed assuming a zero-order hold, that is, based on the previously coded frame.

In our first experiment, we test the effectiveness of the proposed rate control components. The input code stream is transcoded to a target bit-rate of 800 kbps using the following transcoding methods: uniform rate control (URC), adaptive rate control (ARC) with variable rate allocation, ARC with

(a) Original image: 38 256 bytes.

(b) Transcoded image: Qroi = 3, no BG, 6137 bytes.



(c) Transcoded image: Qroi = 3, Qbg = 0, 9611 bytes.

(d) Transcoded image: Qroi = 3, Qbg = 1, 13 302 bytes.

(e) Transcoded image: Qroi = 3, Qbg = 2, 21 432 bytes.

(f) Mosaic image: 1.5 fps, object number = 10.

FIGURE 3: Example images corresponding to various ROI-based streaming methods and parameter settings.

$\omega = 4$ for quality stabilization, ARC with $\tau_{\max} = 7$ for frame skipping, and ARC with both quality stabilization and frame skipping enabled. In all simulations for ARC in this experiment, the buffer size is set to 1 MB.

The results of this first experiment are summarized in Table 1. From the table, we observe that the bandwidth utilization and quality of both background and ROI using URC is relatively low compared with ARC. While the overall quality of ARC is clearly higher than that of the URC method in terms of quality layers and MSE, the quality of the background using ARC fluctuates significantly. Such oscillations in quality have a notable impact on quality for certain segments of the video. With the proposed quality stabilization algorithm, these fluctuations can be controlled with minimal

Table 1: Experimental results comparing URC with various ARC configurations.

Configuration	Skip	BWU	ΔP0	Avg0	Avg1	MSE
URC	N/A	0.75	140	1.7	4.0	83.4
ARC	N/A	1.00	600	2.3	4.0	41.0
ARC + W (**4**)	N/A	1.00	128	2.3	4.0	40.8
ARC + Skip (**7**)	491	0.84	47	2.8	4.0	28.4
ARC + Skip (**7**) + W (**4**)	491	0.84	17	2.8	4.0	28.1

Table 2: Experimental results comparing ARC + W(4) + Skip(7) with various buffer sizes.

Buffer	Skip	BWU	ΔP0	Avg0	Avg1	MSE
1 MB	491	0.84	17	2.8	4.0	28.2
512 KB	472	0.77	39	2.6	4.0	32.7
256 KB	448	0.70	65	2.5	4.0	38.7
128 KB	429	0.64	110	2.3	3.6	50.8
64 KB	417	0.61	170	2.2	3.2	59.9

change to the overall average quality. Finally, with the frame skipping enabled, we see another moderate increase in quality and fewer fluctuations in quality.

From the data, we find that frame skipping accounts for the majority of gains observed for this particular sequence. This is likely due to the relatively high percentage of non-ROI frames in the test sequence, which is a typical surveillance video. Larger differences between the skip-only and skip-with-quality stabilizations could be expected for sequences with a higher percentage of ROI frames.

In our second set of experiments, we investigate the impact of buffer size on the efficiency of the adaptive rate control algorithm. Generally speaking, larger buffers not only require more memory in a device but also increase delay. Depending on the application, limited buffers or strict requirements on the delay may be imposed. Using the same image data and target bit-rates, the ARC algorithm with quality stabilization and frame skipping is simulated with varying buffer sizes from 1 MB to 64 KB.

The results of the second experiment are summarized in Table 2. As expected, we see a slight decline in performance with reduced buffer sizes. With smaller buffer sizes, we observe that the bandwidth utilization is decreased and hence the average quality becomes lower. Reduced buffer sizes also constrain the effect of frame skipping, that is, reducing the average number of bits for a coded frame and lowering overall quality. It is noted that even with reduced buffer sizes, the ARC method still outperforms URC in terms of average overall quality. The most significant gains will be obtained with larger buffer sizes though.

Extensive subjective evaluation has been carried out. The results reveal that the proposed ARC algorithm offers substantial improvement over URC when either quality stabilization and/or frame skipping are enabled. Without at least one of these options, frequent fluctuations in quality occur contributing to an overall decrease in subjective quality. Furthermore, for this particular sequence tested, it has been found that ARC with quality stabilization and a large buffer size is subjectively similar to ARC with frame skipping and small buffer size. Therefore, the skip only option is preferred for low-delay applications.

4.3. Complexity analysis

In order to demonstrate the computational efficiency of the proposed transcoder technique, we provide a run-time analysis of the processes including a breakdown of major components. The simulations are performed using a notebook PC with Mobile Pentium 1.6 GHz, 1 GB memory and Windows XP. The software is written in C and not optimized for performance or to a particular platform.

Table 3: Processing time (Mobile Pentium 1.6 GHz, 1 GB memory, WindowsXP).

Methods	Transcoding (ms)		Decoding (ms)	Display (ms)
	Data anal.	*ROI trans.*		
FF	9.5	0.2	39.5	7.9
BR	9.5	0.1	33.2	7.9

Table 3 shows average time per frame for the data analysis, ROI transcoding, decoding and display, respectively. In our simulations, we used several different ROI settings and different parameter configurations. However, since the vast majority of the total processing is due to the data analysis function, which is independent of the configuration settings, we report a single set of results. From these results, we observe that it takes only 9.7 milliseconds to transcode an image on average. It is clear that the majority of processing is due to the data analysis operation, which includes tag-tree decoding and calculation of the packet body lengths. A significant amount of processing is also due to memory allocation and free operations, which could easily be brought outside the main library routines to improve overall efficiency. Finally, we note that the ROI transcode operation is a very small portion of the total processing time.

It should be clear from these results that the proposed transcoding techniques are suitable for implementation on very low-cost processors and that further optimization of the memory handling and bit I/O is possible.

5. CONCLUDING REMARKS

This paper presented a scalable video streaming system based on JPEG2000 that is oriented towards surveillance applications. Several streaming methods were introduced and an adaptive rate control algorithm for JPEG2000 transcoding was presented. The algorithm allocates rate to each frame in an image sequence based on target rate, buffer occupancy, and ROI information. The key components of the proposed rate control algorithm include variable rate allocation, frame skipping, and quality stabilization. The benefits of these components have been studied and it has been shown that the proposed algorithm significantly outperforms the

reference uniform rate control method. It has also been shown that the complexity of the proposed transcoding technique is very low and suitable for implementation on embedded processors.

In terms of future work, we believe there is still opportunity to improve these results of the rate control further, especially for applications that require a limited buffer size. Another interesting topic to explore is to maximize the perceptual quality considering quality fluctuation and frame skipping in nonbackground frames. Finally, for the BR streaming mode in which a refresh of the background is sent occasionally, an automated method of determining the need for a background update would be desirable, for example, based on scene information such as long term changes in illumination or objects in the scene.

REFERENCES

[1] ISO/IEC 15444-1, "Information technology—JPEG 2000 image coding system—part 1: core coding system," 2000.

[2] E. Atsumi and N. Farvardin, "Lossy/lossless region-of-interest image coding based on set partitioning in hierarchical trees," in *Proceedings of International Conference on Image Processing (ICIP '98)*, vol. 1, pp. 87–91, Chicago, Ill, USA, October 1998.

[3] A. Skodras, C. Christopoulos, and T. Ebrahimi, "The JPEG 2000 still image compression standard," *IEEE Signal Processing Magazine*, vol. 18, no. 5, pp. 36–58, 2001.

[4] Z. Wang and A. C. Bovik, "Bitplane-by-bitplane shift (BbBShift)—a suggestion for JPEG 2000 region of interest coding," *IEEE Signal Processing Letters*, vol. 9, no. 5, pp. 160–162, 2002.

[5] L. Liu and G. Fan, "A new JPEG 2000 region-of-interest image coding method: partial significant bitplanes shift," *IEEE Signal Processing Letters*, vol. 10, no. 2, pp. 35–38, 2003.

[6] R. Rosenbaum and H. Schumann, "Flexible, dynamic and compliant region of interest coding in JPEG 2000," in *Proceedings of International Conference on Image Processing (ICIP '02)*, vol. 3, pp. 101–104, Rochester, NY, USA, September 2002.

[7] T. Hata, N. Kuwahara, T. Nozawa, D. L. Schwenke, and A. Vetro, "Surveillance system with object-aware video transcoder," in *Proceedings of 7th IEEE Workshop on Multimedia Signal Processing (MMSP '05)*, pp. 1–4, Shanghai, China, November 2005.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2007, Article ID 70429, 7 pages
doi:10.1155/2007/70429

Research Article

Bandwidth Estimation in Wireless Lans for Multimedia Streaming Services

Heung Ki Lee,[1] Varrian Hall,[1] Ki Hwan Yum,[2] Kyoung Ill Kim,[3] and Eun Jung Kim[1]

[1] *Department of Computer Science, Texas A&M University, College Station, TX 77843-3112, USA*
[2] *Department of Computer Science, The University of Texas at San Antonio, San Antonio, TX 78249-1644, USA*
[3] *Electronics and Telecommunications Research Institute (ETRI), 161 Gajeong-Dong, Yusung-gu, Daejeon 305-700, South Korea*

Received 2 November 2006; Accepted 21 December 2006

Recommended by Jianfei Cai

The popularity of multimedia streaming services via wireless networks presents major challenges in the management of network bandwidth. One challenge is to quickly and precisely estimate the available bandwidth for the decision of streaming rates of layered and scalable multimedia services. Previous studies based on wired networks are too burdensome to be applied to multimedia applications in wireless networks. In this paper, a new method, *IdleGap*, is suggested to estimate the available bandwidth of a wireless LAN based on the information from a low layer in the protocol stack. We use a network simulation tool, NS-2, to evaluate our new method with various ranges of cross-traffic and observation times. Our simulation results show that *IdleGap* accurately estimates the available bandwidth for all ranges of cross-traffic (100 Kbps ∼ 1 Mbps) with a very short observation time of 10 seconds.

1. INTRODUCTION

Since introduced commercially in 1995, multimedia streaming services have become one of the most promising internet services currently available. In addition, wireless local area networks (WLANs) make multimedia streams commonplace, and terminals are diversifying into hand-held devices such as PDAs, laptops, and audio/video players. These heterogeneous devices have different access patterns and mobility [1]. Most multimedia streams are hungry for stable network bandwidth, but a shared medium WLAN may not support it. To meet their bandwidth requirements, rate scalability can be achieved by layered video representation [2, 3]. However, there are still problems in estimating the point in time to change the bit rate of the transmitted bit stream. Estimating the available network bandwidth in a WLAN is very challenging and crucial for multimedia streaming services.

Although there can be various wireless environments where multimedia services are provided, we mainly focus on the WLAN shown in Figure 1. In this figure, an Internet-based set top box (STB) is the interface between a wired network and a wireless network. Even though wired networks can provide high and stable bandwidths, fragile wireless networks may not support them. Therefore, for layered streaming services, it is very critical for the STB to know the available wireless network bandwidth.

In a wireless network, the IEEE 802.11 protocol in distributed coordination function (DCF) mode, based on CSMA/CA algorithm, is becoming a de facto standard. Previous studies [4–6] based on the bandwidth estimation of wired environments are not applicable to wireless networks that use the DCF protocol. Multimedia streaming is a soft real-time service where each frame is delay-sensitive. Swiftness and availability are critical for real-time system. During bandwidth deviations, the rate of the transmitted multimedia streams should change expeditiously. The accuracy of previous works, *Spruce* [4] and *ProbeGap* [6], is dependent on probing time and the volume of the packets for probing. *ProbeGap* produces good estimates at low cross-traffic rates (2 Mbps cross-traffic regardless of the cross-traffic packet size); however, it significantly overestimates available bandwidth when the cross-traffic is high (4 Mbps cross-traffic generated with 300-byte packets) [6]. Influence by cross-traffic on probe packet sequences causes probe packets in sequences to be split up or even lost.

Our contribution in this paper is twofold. First, we suggest *IdleGap*, which is a bandwidth estimation tool for a real-time system in a wireless network. Second, our system

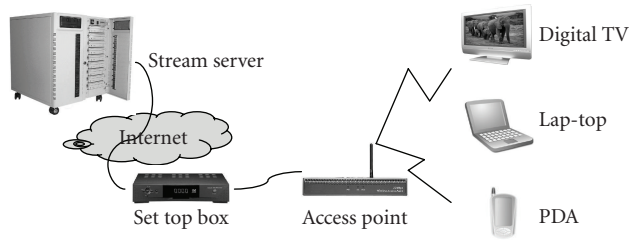


Figure 1: Stream service based on the STB and 802.11.

is independent of cross-traffic. We estimate the available bandwidth via the ratio of free time in the wireless links. To get the ratio of idle time in a wireless network, information from network management at the low layer is used. It provides us with an efficient and fast method for estimating the available bandwidth. The rest of the paper is organized as follows. Section 2 shows the related work in estimating bandwidth and discusses the cross-layer. In Section 3, our new method, *IdleGap*, is proposed and known challenges in bandwidth estimation are addressed. After presenting the results of our method and other tools in Section 4, we conclude this paper in Section 5.

2. RELATED WORK

2.1. Bandwidth estimation in broadband networks

Since the introduction of *Cprobe* [7], many tools have been suggested. *Cprobe* [7] uses Internet Control Message Protocol (ICMP) packet trains to estimate the current congestion along a path. *Cprobe* bounces a short stream of echo packets of a target server and records the time between the receipt of the first packet and the receipt of the last packet. Dividing the number of bytes sent by this time yields a measure of available bandwidth. In order to tolerate packet drops and possible reordering of packets, *Cprobe* uses results of four separate 10-packet streams when calculating the available bandwidth. *Cprobe*'s successors, *Spruce* [4] and *IGI* [8], use the interval of consecutive probe packets, since the interval or gap between probe packets increases in heavy cross-traffic. *Spruce* and *IGI* are both designed based on the ProbeGap model [6] which assumes a single bottleneck. *Spruce* samples the arrival rate at the bottleneck queue before the first packet departs the queue. *Spruce* calculates the number of bytes received at the queue between two probes for the interprobe spacing at the receiver. *Spruce* then computes the available bandwidth as the difference between the path capacity and the arrival rate at the receiver bottleneck. The *IGI* [8] algorithm sends a sequence of packet trains with an increasing initial gap, from the source to the destination host. *IGI* monitors the difference between the average source (initial) and destination (output) gap and terminates when it becomes zero. *Topp* [9] and *Pathload* [5] are also based on the rate of incoming packets. The comparison of the incoming rate from the sender side to the outgoing rate at the receiver side reveals the incoming rate to be less than or equal to the available bandwidth of the probing link. In *ProbeGap* [6], the link's idle time is the milestone for bandwidth estimation of a wireless network; however, *ProbeGap* also must send several probe packets over a specific interval.

All the methods outlined above introduce additional traffic into the link and require a probing sequence time to send and process the probing packets. To account for lost probes, additional probes are sent requiring more processing and filtering out of bad estimates. As a result, most of these methods may not be applicable to certain applications requiring instant bandwidth estimates, and if the link is congested, many probes may not reach the destination. Specifically, strict time bounds required of multimedia applications impose upper limits on delay and jitter in addition to the usual performance metrics of throughput and packet loss.

2.2. Cross-layer feedback

For efficient mobile device communication and interaction, cross-layer feedback is performed by a mobile device accessing its own protocol stack layers that contain information from the transmitted packets. Cross-layer feedback allows interaction between a layer and any other layers in the protocol stack. Packet information retrieval across the protocol stack layers, that is, cross-layering, provides very useful information about mobile devices in a wireless network. Several studies [7, 10–12] which have revealed interaction across-layers aid in improving a system. Shah et al. [10] proposes the use of a centralized *bandwidth manager* (BM), which obtains its channel time proportion (CTP) requirements from each flow at the start of its session. It uses this information to gauge what proportion of unit channel time each flow should be allotted. Its system takes advantage of cross-layer interaction between the application/middleware and the link layer. Davis [11] suggested an 802.11 management method that processes the captured frame to obtain the available bandwidth. The method describes a WLAN traffic probe that operates at the MAC layer and is capable of producing real-time information on resource usage on a per-station basis. For a QoS-sensitive application, a different priority at the MAC layer may be assigned based on the applications [12]. Carter and Crovella [7] used bandwidth probing to measure bandwidth and congestion at the application level. All these methods infer the ability to gather, compute, and share useful information for bandwidth estimation across the OSI layers. Eberle et al. [13] suggested a model for energy-efficient transmission that is based on cross-layer. They insert a *quality of energy manager* (QoEM) into the network protocol stack that manages the transmission.

2.3. Set top box

An STB is a device combining the functionality of analog cable converter boxes such as tuning and descrambling and computers such as navigation, interaction, and display. Today's STBs have four major components: a network interface, an MPEG decoder, graphics overlay, and a presentation engine [14].

2.3.1. STBs on the Internet

Recent successful deployments of IPTV over DSL in Europe and Asia have proven that telecommunication companies can successfully enter the market for television services. Last year Cisco acquired an STB manufacturer Scientific Atlanta (SA). Recently, another STB manufacturer, Motorola, agreed to purchase Kreatel, a Swedish manufacturer of IPTV STBs. For carrier networks and the digital home, this combination makes for a "triple play" solution integrating broadband video, voice, and data access into a single device.

The medium of delivery, the Internet, has also shown itself to be capable of delivering quality video and entertainment. As a result, the digital home consumer market has rapidly grown, and both Motorola and Cisco were aware of how the STB would play a key role in the digital home consumer market. The STB designers are being asked to support an array of new audio, video, and image formats as their products evolve into more open, networked devices. IPTV STBs may be enabled with the functions of personal video recorders (PVRs), digital media adapters (DMAs), voice over IP (VoIP), videophones, and more [15].

Due to the heterogeneous nature of home-based networked devices, each new device with additional functionality layers on different requirements. IPTV and VoD depend on streaming media over a wide area network (WAN) while media applications such as PVR and DMA add a media source in a home LAN environment. For IP video transmitted using the UDP protocol, packet loss can cause significant QoS reduction. A simple video stream can be severely degraded with low levels of packet losses, due to error propagation effects. Video quality is often represented in terms of peak signal-to-noise ratio (PSNR), which is a measure of the root mean square (RMS) error between the original and reconstructed video sequences.

Although all of the issues highlighted require a solution, it is a critical importance for the ability of the STB to adapt to the limited available bandwidth. Telchemy, a leader in VoIP and IPTV performance managements offers a lightweight software agent called VQmon/SA-VM that can be integrated into STBs [16]. VQmon/SA-VM transmits metrics back to service providers during video transmissions. The following are the feedback metrics.

(1) Video service transmission Quality (VSTQ) score, providing data on video transmission quality
(2) Video quality score (VQS), providing an estimate of user perceived quality.

Although this method provides a unique solution for the management of a service provider to STB transmissions, it does not provide a solution for an STB to end-user link management.

2.3.2. Our approach

Typically an STB receives a request from a client, retrieves the requested multimedia data from the server, and forwards it to the multimedia terminal. During this process, the STB can cache portions of the stream and forward the cached stream data to multimedia terminals through a shared resource, the wireless channel. The STB caches and forwards the streaming data between two different networks, wired and wireless networks, in order to reduce negative effects of network traffic such as late packets. The more resources assigned to handle the streams, the less jitter the terminal will experience within the network. The wireless channel is a limited shared resource available for servicing heterogeneous multimedia streams. Therefore, a simple and effective allocation strategy for the wireless channel is critical for improving the quality of the video streams delivered through the STB and the wireless network. In general, the streaming services with high quality may require more resources than the ones with low quality. Unfortunately, the estimation of the available resources required for each case has not been fully understood yet, so currently our research focuses on how to estimate the available resources for heterogeneous streaming services in this environment.

As shown earlier in Figure 1, an STB resides between the server and multimedia terminals, and relays the data flow from the server to the terminals and vice versa. Although the cost of the STB limits its functionality, a simple strategy implemented within the STB can improve the quality of multimedia services dramatically.

3. IDLEGAP USING NETWORK ALLOCATION VECTOR

3.1. Background

Bandwidth estimation is a prerequisite problem for real-time applications in wireless networks. There are two factors making this problem unique. First of all, unlike wired networks, traditional FIFO is not used to schedule bandwidth among connections in wireless networks. To avoid collisions in wireless networks, nodes are arranged in a distributed manner. This arrangement causes bandwidth estimation methods in wired networks using intervals [4, 8] or rates [5, 9] inapplicable for bandwidth estimation in wireless networks. Secondly, probing time for the available bandwidth should be short for time-sensitive multimedia streaming services. References [6, 11] suggested that idle time of a link in a wireless network can be a major milestone for estimating the available bandwidth as follows.

Let C be the capacity of the wireless network.[1] *Idle_rate* indicates the rate at which the link is idle. Then the available bandwidth (AB) can be obtained by the following product:

$$AB = C \times Idle\_rate. \tag{1}$$

However, previous methods [6, 11] using this formula cause too much overhead to be used in a real-time system for the estimation of the available bandwidth. In [6], too

[1] It can be changed by the negotiated data rate between a wireless node and the access point.

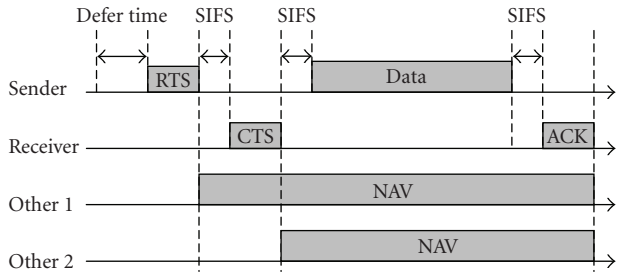


Figure 2: IEEE 802.11 DCF MAC protocol.

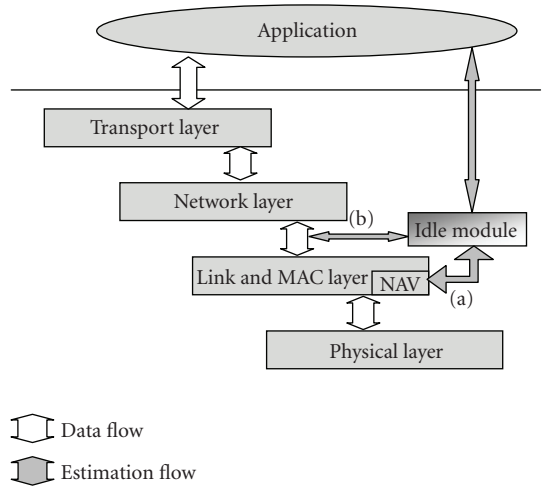


Figure 3: Architecture of idle module.

much time elapsed probing the link and analyzing probing data, and results showed multiple incorrect estimated values in heavy traffic. Reference [11] utilizes too much time in order to capture whole packets in the network and get node information from captured packets. For real-time applications such as multimedia streams, it is difficult to use these methods; therefore, in Section 3.2.2 we introduce an efficient method to calculate the *Idle_rate*.

3.2. IdleGap

3.2.1. Network allocation vector

When two nodes in a wireless network share the same access point (AP) but cannot hear each other, one node will not be able to know whether the other node is already using the shared resource, that is, the wireless channel. For addressing this hidden node problem, each node uses the network allocation vector (NAV) that shows how long other nodes allocate the link in the IEEE 802.11 DCF MAC protocol. Even though a node is located at a place where it cannot reach other active nodes, the node can know whether another node is already using the wireless network by checking its NAV. In Figure 2, when the sender sends RTS (request to send) to the receiver (AP), *Other-1* node that is reachable from sender updates its NAV. However, *Other-2* node does not update NAV, because it is not reachable from sender. When the receiver sends CTS (clear to send), *Other-2* node updates its NAV. The idle time in the wireless network can then be estimated from the NAV information.

3.2.2. Estimation of wireless link idle rate

All nodes in a WLAN share the same resource; that is, a wireless channel. If a node in a WLAN is utilizing the resource, the additional node(s) should await the release of the wireless channel. During a transmission in a WLAN, a node can be one of the following: sender, receiver, or onlooker. If a node transmits data to another node, it is a sender. A node is a receiver if receives data. Finally, when a node does not join the transmission, it is an onlooker.

The busy time of the link can be estimated by adding up all the transactions of nodes in the network as depicted in (2). Here T_l is the busy time of link l and $TT(i, j)$ indicates the transaction time between nodes i and j,

$$T_l = \frac{1}{2} \times \sum_{i=1}^{n} \sum_{j=1}^{n} (TT(i, j)). \tag{2}$$

Unfortunately, we cannot know all the transaction times from the nodes in the network. In addition, obtaining the transaction information can increase network traffic, hence affecting current traffic on the network. Therefore, we propose to obtain all the necessary information from one node in the network as follows.

The transaction time of node i can be obtained via the sum of the sending and receiving times to/from node i ($TT(i, j) = ST_i + RT_i$, where ST_i is the sending time from node i to j and RT_i is the receiving time from node j to i). For the transaction time between other nodes, we can get the onlooking time from the NAV in node i that is updated in other node transactions ($TT(i, j) = OT_i$, where OT_i is the onlooking time at node i). Therefore, we can estimate the busy time T_l in any node i in the network as shown in (3):

$$T_l = ST_i + RT_i + OT_i. \tag{3}$$

We can then obtain *Idle_rate* using the busy time:

$$Idle\_rate = 1 - \frac{\text{busy time}}{\text{total elapsed time}}. \tag{4}$$

3.2.3. System model

We propose to add an *idle module* in the MAC layer of a node in the network. This module obtains the busy time (T_l) from (a) and (b) in Figure 3. The transaction time of a node can be obtained through accessing outgoing and incoming packets ($ST_i + RT_i$) between the network layer and the link and MAC layer (shown in (b)). Idle module also gets the onlooking time (OT_i) from the NAV (shown in (a)). The updating process of the NAV triggers the *idle module* to update its value. An application can access the *idle module* to get the

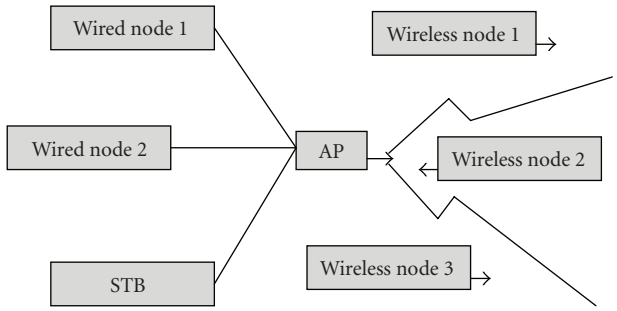


Figure 4: Simulation environment.

idle rate (1 − busy time/total elapsed time). Then applying the idle rate and link capacity C to (1), the estimated bandwidth of the link can be calculated with minimal effort. We call this method *IdleGap*.

4. EXPERIMENTAL RESULTS

To verify the performance of our *IdleGap* method, network simulations were conducted using NS-2. As shown in Figure 4, there are seven nodes including one STB, two other wired nodes, three wireless nodes, and an AP. In the wired network, the capacity of the link was set to 10 Mbps, while the capacity in wireless network was set to 1 Mbps.

In Figure 4, communication in the simulation via the AP involves three connections: wired node 1 to wireless node 2, wired node 2 to wireless node 1, and STB to wireless node 3. wired nodes 1 and 2 generate the cross-traffic, while the algorithm generates timestamps from packets received by STB via packets sent from wired node 3 to estimate the available bandwidth. We compare *IdleGap* with *ProbeGap* [6] and *Spruce* [4], which provides more accurate estimation than other previous works.

4.1. Experiments with increasing cross-traffic

Figure 5 shows the estimated available bandwidth value for each algorithm. The capacity of the wireless network in our simulation is 1 Mbps. Probing time for each algorithm is 1000 seconds and 200 probing packets are allowed. In light cross-traffic, *ProbeGap* produces bandwidth estimates reflective of measured available bandwidth values. However, it shows multiple transition points over 200 Kbps cross-traffic. In the original *Spruce* paper, the intrapair gap is set to the transmission time of the narrow link [4]. This causes the underestimation of the available bandwidth for the link. Therefore, the intrapair gap was calibrated to reflect the available 1.0 Mbps with no cross-traffic. Even after the calibration, *Spruce* overestimates the bandwidth severely with more than 0.5 Mbps cross-traffic. The reason is due to high drop rates with heavy cross-traffic. Thus, the estimated bandwidth value becomes polluted. This could cause the overestimation of the available bandwidth.

The *IdleGap*, which uses NAV to estimate bandwidth, shows the closest match to the real bandwidth. Note that after

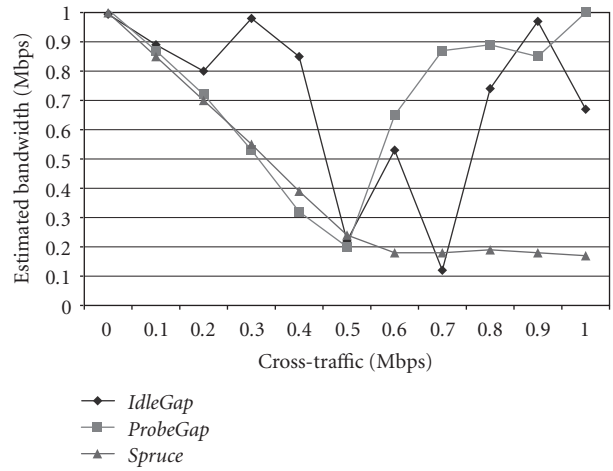


Figure 5: Estimated bandwidth with cross-traffic.

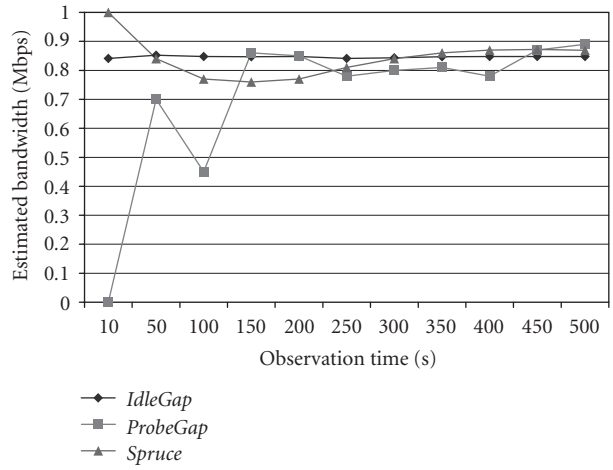


Figure 6: Estimated bandwidth with different observation times.

0.6 Mbps cross-traffic, saturation occurs due to the overhead of the wireless network such as defer time and RTS/CTS.

4.2. Experiments with different observation times

In this experiment, we vary the observation time to estimate the available bandwidth. Since we focus on the effect of observation period, the cross-traffic is set to 10 Kbps, where all three schemes are able to estimate the bandwidth accurately as shown in Figure 5. *ProbeGap* and *Spruce* send the probes at intervals of 5 seconds [6]. Figure 6 shows the estimated values of the available bandwidth for *ProbeGap, Spruce,* and *IdleGap* between observation periods of 10 and 500 seconds. Until 250 seconds, *ProbeGap* and *Spruce* record values not reflective of measured available bandwidth. After 250 seconds, *ProbeGap* and *Spruce* values are near the measured bandwidth values. However, *IdleGap* generates values reflective of measured bandwidth for all periods. Therefore, we can conclude that *IdleGap* provides accurate estimations with short observation times.

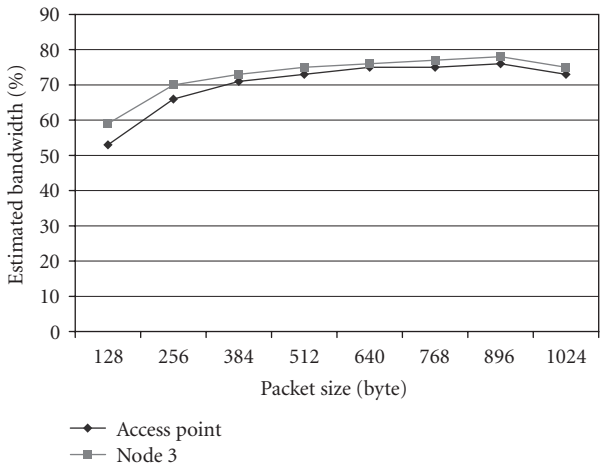


Figure 7: Estimated bandwidth with different packet sizes.

4.3. Experiments with different packet size

In Figure 7, the estimated idle times in the AP and node 3 are depicted with different packet sizes of the same cross-traffic. Cross-traffic in the simulation is 100 Kbits/sec (10%) and the packet size is changed from 128 to 1024 bytes. We observe that packet sizes between 512 and 896 bytes provide more accurate estimation. The estimated idle time with the small size packet is smaller than the one with the large size packet. It is because the overhead of the small packet transmission is larger than the one of the large packet transmission. In order to transmit a packet, the sender should send the RTS, CTS, and ACK to the receiver. The frequent transmission of small packets increases this overhead. That is why the *IdleGap* underestimates the available bandwidth with small size packets. On the other hand, with the largest size packets (1024 bytes), the estimated idle time is also decreased slightly. During the transmission, the large packet is broken into several fragments in the Mac layer to reduce the error rate, which again causes overhead.

Estimated bandwidth in the AP inclines to be smaller than the one in node 3. If node 3 is a hidden node, it receives only the CTS, not the RTS. Then, node 3 cannot detect the busy time gap between the RTS and the CTS.

5. CONCLUDING REMARKS

The most challenging aspect of multidynamic server selection media streaming services is the adaptive bit rate of each multimedia stream according to the network status; therefore, in this paper we focused on a method to estimate the available bandwidth of a wireless link. The method must have the following characteristics: (a) it should be applicable to real-time applications such as multimedia streaming services; (b) be simple and effective in estimating the available bandwidth, and (c) incur low overhead.

We have presented a new bandwidth estimation method, *IdleGap*, which can efficiently calculate the available bandwidth using the information collected from one node in a wireless network. *IdleGap* is simple and does not incur extra network overhead. The simulation result shows that *IdleGap* outperforms the other probing and bandwidth estimation methods, *ProbeGap* and *Spruce*.

REFERENCES

[1] T. Henderson, D. Kotz, and I. Abyzov, "The changing usage of a mature campus-wide wireless network," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking (MOBICOM '04)*, pp. 187–201, Philadelphia, Pa, USA, September-October 2004.

[2] J. Shin, J. W. Kim, and C.-C. J. Kuo, "Quality-of-service mapping mechanism for packet video in differentiated services network," *IEEE Transactions on Multimedia*, vol. 3, no. 2, pp. 219–231, 2001.

[3] D. Quaglia and J. C. De Martin, "Delivery of MPEG video streams with constant perceptual quality of service," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME '02)*, vol. 2, pp. 85–88, Lausanne, Switzerland, August 2002.

[4] J. Strauss, D. Katabi, and F. Kaashoek, "A measurement study of available bandwidth estimation tools," in *Proceedings of the ACM SIGCOMM Internet Measurement Conference (IMC '03)*, pp. 39–44, Miami Beach, Fla, USA, October 2003.

[5] M. Jain and C. Dovrolis, "Pathload: a measurement tool for end-to-end available bandwidth," in *Proceedings of Passive and Active Measurements Workshop (PAM '02)*, pp. 14–25, Fort Collins, Colo, USA, March 2002.

[6] K. Lakshminarayanan, V. N. Padmanabhan, and J. Padhye, "Bandwidth estimation in broadband access networks," in *Proceedings of the ACM SIGCOMM Internet Measurement Conference (IMC '04)*, pp. 314–321, Taormina, Italy, October 2004.

[7] R. L. Carter and M. E. Crovella, "Dynamic server selection using bandwidth probing in wide-area networks," Tech. Rep. TR-96-007, Computer Science Department, Boston University, Boston, Mass, USA, 1996.

[8] N. Hu and P. Steenkiste, "Evaluation and characterization of available bandwidth probing techniques," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 6, pp. 879–894, 2003, special issue in Internet and WWW Measurement, Mapping, and Modeling.

[9] B. Melander, M. Bjorkman, and P. Gunningberg, "A new end-to-end probing and analysis method for estimating bandwidth bottlenecks," in *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM '00)*, vol. 1, pp. 415–420, San Francisco, Calif, USA, November-December 2000.

[10] S. H. Shah, K. Chen, and K. Nahrstedt, "Dynamic bandwidth management in single-hop ad hoc wireless networks," *Mobile Networks and Applications*, vol. 10, no. 1, pp. 199–217, 2005.

[11] M. Davis, "A wireless traffic probe for radio resource management and QoS provisioning in IEEE 802.11 WLANs," in *Proceedings of the 7th ACM Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems (ACM MSWiM '04)*, pp. 234–243, Venezia, Italy, October 2004.

[12] G. Xylomenos and G. C. Polyzos, "Quality of service support over multi-service wireless Internet links," *Computer Networks*, vol. 37, no. 5, pp. 601–615, 2001.

[13] W. Eberle, B. Bougard, S. Pollin, and F. Catthoor, "From myth to methodology: cross-layer design for energy-efficient wireless communication," in *Proceedings of the 42nd Design Automation Conference (DAC '05)*, pp. 303–308, Anaheim, Calif, USA, June 2005.

[14] A. Laursen, J. Olkin, and M. Porter, "Oracle media server: providing consumer based interactive access to multimedia data," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, vol. 23, no. 2, pp. 470–477, Minneapolis, Minn, USA, May 1994.

[15] S. Robertson and R. Rivin, "Analog Devices: Designing IPTV Set-Top Boxes Without Getting Boxed In," http://www.analog.com/.

[16] "Application Note IPTV Performance IPTV Performance Management," January 2005, http://www.telchemy.com/appnotes/Managing_IPTV_Service_Quality.pdf.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2007, Article ID 39524, 8 pages
doi:10.1155/2007/39524

Research Article

Packet Media Streaming with Imprecise Rate Estimation

Dan Jurca and Pascal Frossard

Ecole Polytechnique Fédérale de Lausanne (EPFL), Signal Processing Institute, 1015 Lausanne, Switzerland

Received 2 November 2006; Accepted 19 December 2006

Recommended by Guobin (Jacky) Shen

We address the problem of delay-constrained streaming of multimedia packets over dynamic bandwidth channels. Efficient streaming solutions generally rely on the knowledge of the channel bandwidth, in order to select the media packets to be transmitted, according to their sending time. However, the streaming server usually cannot have a perfect knowledge of the channel bandwidth, and important packets may be lost due to late arrival, if the scheduling is based on an over-estimated bandwidth. Robust media streaming techniques should take into account the mismatch between the values of the actual channel bandwidth and its estimation at the server. We address this rate prediction mismatch by media scheduling with a conservative delay, which provides a safety margin for the packet delivery, even in the presence of unpredicted bandwidth variations. We formulate an optimization problem whose goal is to obtain the optimal value for the conservative delay to be used in the scheduling process, given the network model and the actual playback delay imposed by the client. We eventually propose a simple alternative to the computation of the scheduling delay, which is effective in real-time streaming scenarios. Our streaming method proves to be robust against channel prediction errors, and performs better than other robustness mechanisms based on frame reordering strategies.

1. INTRODUCTION

Media streaming over the internet is experiencing solid growth and success in the past few years. However, the inherent best-effort characteristics of the underlying transport medium, need to be matched by intelligent streaming mechanisms in order to ensure the success of media applications. In particular, flexible rate adaptation mechanisms must be deployed to compensate for the bandwidth variability observed in the internet. When effective media transcoding capabilities are excluded due to application constraints, packet selection and scheduling represent a powerful solution for adapting the media content transmission to the available resources offered by the transport network. Under timing constraints imposed by a fixed playback delay, efficient media scheduling solutions must rapidly adapt the media packets transmission, to the available channel resources, in order to optimize the quality of service at the client.

Among the most popular scheduling schemes, a first family of methods models the underlying network in a stochastic framework. Given a permitted playback delay at the client, they attempt to maximize the expected received media quality by packet dropping or retransmission. They transform the scheduling decision into a stochastic optimization problem whose result achieves the maximum possible steaming quality for the end user [1]. However, even for simple channel models, the optimal solution requires complex algorithms that necessitate large computational resources. Hence, low delay streaming applications cannot rely on this mechanism for successful data transfer over the network. A second set of scheduling solutions rather considers the network topology and parameters as known in advance and realizes a deterministic scheduling of the packets, in order to maximize the received media quality. Such solutions are simpler, and can be employed in real-time applications. Polynomial time algorithms can take fast decisions on the pool of media packets [2, 3], in order to optimize the streaming process to the available channel conditions. However, this method may prove inaccurate and prone to errors in the case where the exact channel parameters are not fully known, or inexactly predicted by available channel estimation protocols.

Even the best rate estimation algorithms are not able to follow the rate variations of the channel, and often work on a coarser timescale [4]. Since channel prediction errors are inevitable and can lead to late arrivals of important media packets, the streaming server has to design robust packet selection and scheduling strategies against estimation mismatches. Our proposed method relies on a simple FIFO

scheduling mechanism. However, we increase the algorithm's robustness by using a conservative virtual playback delay, smaller than the playback delay imposed by the client [5]. The scheduling process considers the conservative playback delay as the hard deadline for packet arrival at the client, hence it is more aggressive in the packet selection process. On the other side, the difference between the conservative scheduling delay and the effective playback delay after which the client starts playing the video, transparently compensates for the eventual late packet arrivals due to the erroneously estimated end-to-end channel rate variations.

Overall, we observe that a very conservative scheduling delay tends to limit the selection of transmitted media data to only a few packets, which penalizes the quality at the receiver. Alternatively, a scheduling delay that is too close to the effective playback delay may result in late arrival of packets, which also penalizes the quality. Hence, the purpose of this work is to analyze the trade-off between robustness against channel prediction errors, and packet selection limitations, observed as a result of tighter scheduling constraints.

The rest of this paper is organized as follows: Section 2 presents an overview of existing work in the domain of robust media streaming. We formulate in Section 3 an optimization problem whose goal is to find the optimal conservative delay used in the scheduling process, which maximizes the quality of the received video for a given channel rate model, and a given playback delay at the client. We discuss the complexity of the exact solution for the optimization problem and we present a fast solution in Section 4. Section 5 presents our simulation results and Section 6 concludes this paper.

2. RELATED WORK

Robustness to network failures is one of the necessary attributes of a successful media application. Application layer tools have been designed for providing the required flexibility in order to cope with network variations. The authors of [6, 7] present an overview of video coding techniques that confer flexibility and robustness to the streaming process. Error resilient video encoding and error-concealment strategies at the client side are detailed. These techniques can be further enhanced with network layer error-robustness strategies like ARQ or FEC [8, 9]. While these mechanisms offer the flexibility needed in order to cope with network channel errors and variations, their design is based on the knowledge of network parameters. Their functionality depends on the accuracy of the channel estimation, hence when these estimations are inexact, they are susceptible to failure. Intelligent scheduling on a packet level can adapt the media streaming decisions in case of network parameter variability, and add an extra layer of flexibility in the wake of adverse network conditions (e.g., bandwidth shortage, or variable transmission delays and jitter).

Informed scheduling decisions optimize the received media quality under network resources constraints. In the Rate-Distortion framework presented in [1], the scheduling algorithm takes an optimal decision (transmission policy) for each media packet/set of packets, based on the parameters of the channel model. The channel is stochastically modelled, and the optimal scheduling solution comes at the expense of complex computations and large delays [10, 11]. On the other hand, deterministic scheduling algorithms [2, 12] are faster, but require exact knowledge of channel parameters, and are prone to errors in case of unpredicted network variability. Previous works [13, 14] enhance the robustness to channel prediction errors, by designing a new scheduling model, in which the packets/frames in a bitstream are rearranged. The most important parts of the bitstream are advanced ahead of the less important ones, so that they are scheduled for transmission with higher priority. Such mechanisms increase the probability of successful transmission of information necessary for correct decoding, even in adverse network scenarios. While these methods increase the robustness to network delay fluctuations, they are also more demanding in terms of codec buffer sizes and computation.

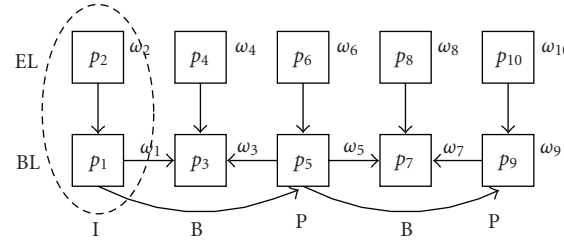


Figure 1: Video packets as a directed acyclic graph.

In this work, we rather enforce a FIFO scheduling mechanism because of its simplicity and efficient use of buffering resources. However, we propose to increase its robustness to channel estimation errors by scheduling packets with a virtual playback delay, smaller than the playback delay imposed by the client. The difference between the scheduling delay and the playback delay after which the client starts playing the video, can transparently absorb the effects of the erroneously predicted end-to-end rate variations on packet arrival times.

3. STREAMING WITH CONSERVATIVE DELAY

3.1. System overview

We consider a single path streaming scenario between a server S and a client C. The media stream can either be prestored at the server (VoD), or can be obtained in real time (real-time streaming). The video content is encoded into one or more layers and fragmented into network packets such that one packet contains information related to one frame and one video layer. Let $P = \{p_1, \ldots, p_n\}$ be the set of available packets at the server, with n representing the total number of packets. Similarly to [3], each packet p_i is completely characterized by its size s_i, its decoding deadline t_i, its importance ω_i, and its list of dependency packets A_i, which are necessary for a correct decoding (see Figure 1).

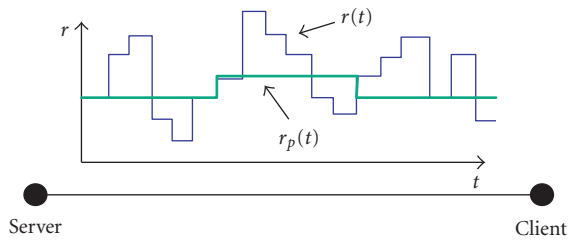


Figure 2: Network end-to-end model with rate variations $r(t)$ and estimated rate $r_p(t)$.

Table 1: Example parameter values for conservative delay scheduling.

Instantaneous rate (kbps)	420
Predicted rate (kbps)	450
Packet size s_i (bits)	8000
Packet weight ω_i	1000
Decoding deadline t_i	0
Playback delay Δ (ms)	200
Conservative playback delay δ (ms)	180
Time t (ms)	0

The intermediate network between S and C is modelled as an end-to-end channel characterized by the variable rate $r(t)$. While we consider no link error in our model, packets can still be lost from a media application perspective, due to late arrivals. The server S estimates on a periodic interval, the available channel rate $r_p(t)$, using any estimation mechanism Γ (see Figure 2). Based on that estimation, the streaming application employs a generic scheduling algorithm Ψ that decides the subset of packets $\pi \subseteq P$ that are sent in a FIFO order to the client, so that the reconstructed video quality is maximized, given the playback delay Δ imposed by the client. The video quality measure Ω can be computed at the client as

$$\Omega = \Omega_S(\pi) - \Omega_L(\pi), \tag{1}$$

where $\Omega_S(\pi) = \sum_i \omega_i$, for all $p_i \in \pi$ represents the quality of the video packets selected for transmission, and $\Omega_L(\pi) = \sum_i(\omega_i \cdot P_i)$ represents the video quality degradation due to packets that cannot be decoded because of late arrivals at the client. P_i represents the probability that packet p_i arrives past its decoding deadline at the client. These late arrivals are caused by channel bandwidth variations, and inaccuracy in the rate estimation used by the server. Indeed, the estimation of the available rate in the future time instants is generally not perfect, and often not able to exactly follow the frequent variations of the bandwidth.

We propose to modify the scheduling strategy, in order to be robust to over estimations of the channel rate. We define a virtual playback delay, or scheduling delay δ, which is used by the server to compute the subset of packets to be sent. As δ is smaller than the actual playback delay Δ, the server will select a reduced number of packets for transmission (Ω_S decreases), but the selected packets have a lower probability to be lost (Ω_L increases). In other words, π now contains only packets that can reach the client before their decoding deadline ($t_i + \delta$) with a streaming rate r_p, and each packet p_i is scheduled and transmitted only once. The choice of the virtual playback delay becomes obviously a trade-off between source quality and robustness to rate variations, and its optimization is proposed in the next sections.

3.2. Illustrative example

We demonstrate the rationale behind our proposed mechanism by a concrete example. Imagine that server S needs to decide at time t whether to send packet p_i to the client C or not. The scheduling decision is based on the predicted network rate at moment t, $r_p(t)$, the size s_i, weight ω_i, dependency list A_i and decoding deadline t_i of packet p_i, and on the conservative playback delay δ. In the same time, C expects packet p_i before time $t_i + \Delta$, so that it can successfully decode it.

For the sake of clarity, assume that the list $A_i = \emptyset$, for example, packet p_i can be independently decoded at C, and that the server's buffer does not contain any other media packets except p_i. The rest of the parameters are set according to Table 1.

Observe that S will take the decision to send the packet on the network after computing the expected arrival time at the client: $T_p = t + s_i/r_p(t) \approx 177\,\text{ms} \leq 180\,\text{ms} = t_i + \delta$. Even if the channel rate is overestimated and packet p_i arrives at the client at $T_a = t + s_i/r(t) \approx 190\,\text{ms} > t_i + \delta$, packet p_i will still arrive on time for successful decoding at the client, as $t_i + \Delta = 200\,\text{ms}$.

On the contrary, imagine the same procedure is applied to packet p_j, under the same conditions, except $s_j = 9.000$ bits and the scheduler does not use the conservative delay δ, rather directly the playback delay Δ. S decides to send the packet as $T_p = t + s_j/r_p(t) = 200\,\text{ms} \leq 200\,\text{ms} = t_i + \Delta$. However, packet p_j is useless for the client as it arrives past its decoding deadline: $T_a = t + s_j/r(t) \approx 220\,\text{ms} > t_i + \Delta$. In such a case packet p_j consumes network resources that could be used more effectively.

Finally, please observe that in the case where S uses the conservative delay δ in scheduling packet p_j, the decision would be to drop the packet, as it would arrive late. This insight lies the ground for the trade-off between robustness against channel prediction errors, and packet selection limitations, observed as a result of tighter scheduling constraints.

3.3. Optimization problem

The virtual playback delay δ used by the scheduler represents a compromise between a conservative selection of packets that minimizes the probability of late arrivals, and the selection of a sufficient number of packets for an effective quality. Given the video sequence, the quality metric Ω, the scheduling strategy Ψ, the rate estimation algorithm Γ, and the playback delay Δ, the optimization problem translates into finding the optimal conservative delay $\delta \leq \Delta$ to be used by

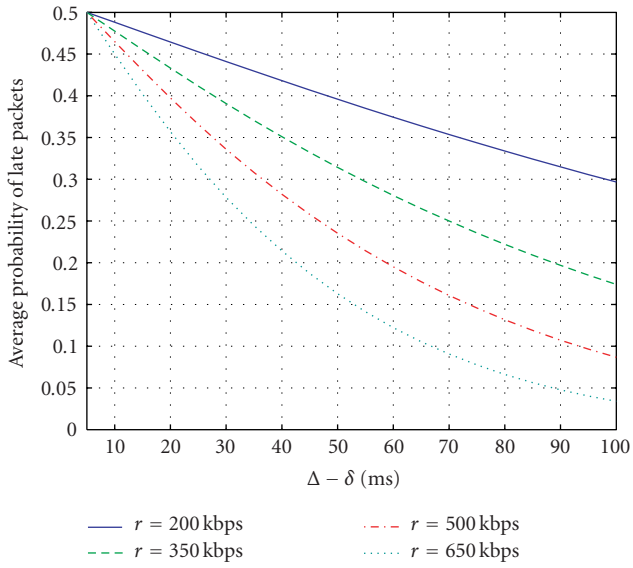


Figure 3: Average probability of late packets ($\Delta = 300$ ms).

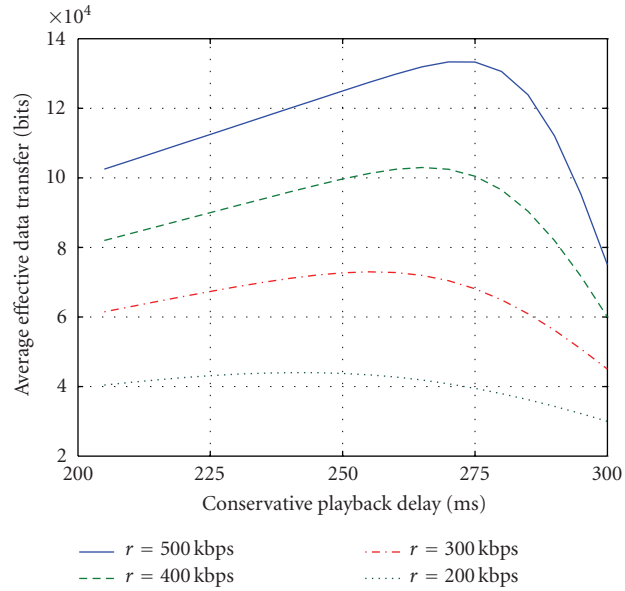


Figure 4: Effective average data transfer ($\Delta = 300$ ms).

the streaming application, in order to maximize the received video quality Ω, for a given channel model

$$\delta^* = \arg\max_{\forall \delta \leq \Delta} \Omega(\delta). \tag{2}$$

In general, this optimization problem does unfortunately not provide any simple solution. Even for fixed Ψ, Γ, and Δ, the scheduling policy π is not constant with the choice of δ, hence finding the optimal solution for the problem has combinatorial complexity. However, for small values of Δ (as in practical real time streaming scenarios), δ^* can be accurately approximated in real time. In the next section, we present our approach towards finding an appropriate solution, based on heuristics from real-time video streaming.

4. FINDING THE CONSERVATIVE DELAY

4.1. General solution

On the one hand, the quality measure $\Omega_L(\pi)$ depends only on the difference $\Delta - \delta$, for a given transmission policy π and the channel model. Very conservative values for δ will ensure a big difference $\Delta - \delta$, hence more margin in dealing with rate prediction errors, and consequently a smaller value for Ω_L (see Figure 3).

On the other hand, the quality measure $\Omega_S(\pi)$ depends only on the scheduled packets according to the predicted rate $r_p(t)$ and δ. Interestingly, our experiments show that, for a given channel model, Ω_S does not vary much with δ, as long as δ is large enough to accommodate the transmission of the largest video packets of the sequence.

Let $R^i(\Delta)$ be the cumulative rate of the channel up to time $t_i + \Delta$: $R^i(\Delta) = \int_0^{t_i+\Delta} r dt$, and $R_p^i(\delta)$ be the cumulative estimated rate up to time $t_i + \delta$: $R_p^i(\delta) = \int_0^{t_i+\delta} r_p dt$. For given δ and Δ, we define the effective data transfer $\mathcal{C}_\Delta^\delta(i)$ on the time interval $[0, t_i+\Delta]$, as the amount of data scheduled according to r_p before $t_i + \delta$, and received before $t_i + \Delta$ according to r:

$$\mathcal{C}_\Delta^\delta(i) = R_p^i(\delta) \cdot Pr\{R_p^i(\delta) \leq R^i(\Delta)\}. \tag{3}$$

An illustration of the effective data rate transfer is given in Figure 4.

Given this measure, we transform the original optimization problem into a new one that chooses δ in order to maximize $\mathcal{C}$, defined as

$$\delta^* = \arg\max_{0 \leq \delta \leq \Delta} \mathcal{C}_\Delta^\delta(i). \tag{4}$$

$\mathcal{C}_\Delta^\delta(i)$ is invariant in time, as long as the channel model does not change, hence it can be computed at any t_i. The previous optimization problem translates into maximizing the chances of every packet p_i, scheduled for transmission at time t, to reach its destination by time $t + \Delta$. Unlike the original optimization problem of (2), (4) depends only on the channel model, hence it is easy to solve, once this model is known. It can be noted that both optimization problems are equivalent in the case of a smooth video model (the video packets have the same size and importance, and there are no dependency among them). We later show in Section 5 that even in realistic video streaming scenarios the solution obtained for this problem is a very good approximation of the optimal solution.

4.2. Example channel model

We now develop all necessary relations for a typical channel modelled as a discrete-time system, with a sampling interval of T_s seconds. The network can communicate a maximum of $r_i T_s$ bits of data in the time interval $[iT_s, (i+1)T_s]$, where r_i is

the available bandwidth of the channel in the ith time interval. The channel rate r_i is given as a Gaussian autoregressive process of the form

$$r_i = \mu + (1-\alpha)\sum_{j=0}^{\infty}\alpha^j n_{i-j}, \quad j \in \mathbf{Z},\ n_k = 0,\ \forall k < 0. \quad (5)$$

Each n_j is an independent zero mean Gaussian random variable with variance σ^2, α is a modelling parameter, and μ denotes the average available bandwidth. The validity of that model for internet traffic traces on time scales of milliseconds up to a few seconds has been verified in [15].

A simple auto-regressive prediction model is used for bandwidth estimation at the server, where the available rate of the network in the next time interval, $k+1$, is given by

$$r_{k+1} = \gamma\frac{\sum_{j=1}^{k-1} r_j}{k-1} + (1-\gamma)r_k, \quad (6)$$

where γ is the prediction coefficient. The estimation is run periodically, on time windows of size T_p. While instantaneous rate variations of the channel can happen on very small time scales (of tens to hundreds of milliseconds), the fastest estimation mechanisms provide accurate results on time intervals of the size of a few round-trip times (e.g., one second or more), and prediction inaccuracies cannot be avoided.

Assuming that $t_i + \Delta = k \cdot T_s \le T_p$, with k an integer,[1] we can compute

$$R^i(\Delta) = k\cdot\mu + \sum_{j=0}^{k}(1-\gamma)\cdot\gamma^{j-1}\cdot\sum_{l=1}^{k-j} n_l. \quad (7)$$

Finally, $\mathcal{S}_i$ denotes the cumulative size of the transmitted packets up to packet p_i : $\mathcal{S}_i = \sum_{j=1}^{i} s_j$, for all $p_j \in \pi$. The probability that a packet arrives too late at the receiver, P_i, can be computed as

$$P_i = Pr\left\{\mathcal{S}_i > R^i/\mathcal{S}_i \le R_p^i\right\}. \quad (8)$$

Since R^i is a normal random variable and R_p^i is a known constant, given any δ and Δ, the error probabilities P_i can be easily computed with the help of the $erfc$ function.

4.3. Scheduling algorithm

While the presented robustness mechanism is generic, and can be applied to any packet scheduling algorithm, in this section we describe the specific algorithm employed in the experimental phase of this paper.

The algorithm is an adaptation of the LBA scheduling algorithm introduced in our prior work [12], to the single-path network scenario presented above. In short, the algorithm performs a greedy scheduling of the most valuable packets first. Less valuable packets are scheduled only if the network capacity permits, and only if they do not lead to the loss of a more valuable packet already scheduled (due to subsequent late arrivals at the client).

First, the n network packets are arranged in descending order of their weight, obtaining a new representation of the encoded bitstream, $P' = \{p'_1, p'_2, \ldots, p'_n\}$. Then, the algorithm attempts a greedy scheduling of the packets on the network link, starting with the most important one. To decide which action to take on each packet p'_i, the algorithm first attempts to schedule all ancestors that have not been scheduled yet. If one of them cannot be scheduled, then the algorithm automatically drops the packet p'_i. This ensures that our algorithm does not waste network resources on transmitting network packets that cannot be correctly decoded at the receiver.

Finally, all packets marked to be transmitted, are reordered according to their decoding deadlines before transmission. When a new packet is inserted for transmission, it triggers a new packet ordering. If packet p'_i can be inserted, without compromising the arrival time of any other already scheduled packet, then it is marked for transmission. Otherwise, packet p'_i is dropped. Please observe that the scheduling algorithm can be run on the total video sequence to be streamed, in the case of VoD streaming, or on a limited window of video packets in the case of real-time streaming.

The total complexity of the scheduling algorithm is driven mainly by the sorting and insertion operations. While the sorting can be performed by any algorithm in time $O(n\log n)$, the insertion of each packet p'_i requires a complete parse through all previously scheduled packets. Hence the total complexity of the algorithm is $O(n^2)$.

5. SIMULATIONS

We discuss the performance of the streaming application with conservative delay and we compare the results obtained by our heuristic solution for δ with the optimal one, and with other frame reordering techniques. We scalably encode the *foreman_cif* sequence (130 frames) using MPEG4-FGS, at 30 frames per second, with a GOP structure of 31 frames (IPBPBPB...). By splitting the bitplanes, we encode one BL and 2 ELs of average rates of 260 kbps. In all our experiments we use the simple packet scheduling algorithm as presented above. We set the weights ω_i of the packets as a function of their relative importance to the encoded bitstream (depending on the type of encoded frame, I, P, or B, and on the encoded layer they represent, BL, EL1, or EL2), as illustrated in Figure 1. In a first approximation, we choose the following packets weights: 5 for I frame BL packets, 4 for the P frame BL packets, 3 for the B frame BL packets, 2 for the EL1 packets, and 1 for the EL2 packets [3].

For the channel model and estimation mechanism, we set the required parameters to $\alpha = \gamma = 0.8$, $T_s = 20$ ms, $T_p = 1$ s, and we vary $\sigma^2 \in [100, 250]$, according to the channel average rate. These values insure realistic channel variations on small time scales around the average bandwidth value. Finally, we set $\Delta = 200$ ms.

[1] The extension of the computation for the general case, on multiple prediction intervals, and when k is not an integer, is straightforward, and omitted in this manuscript.

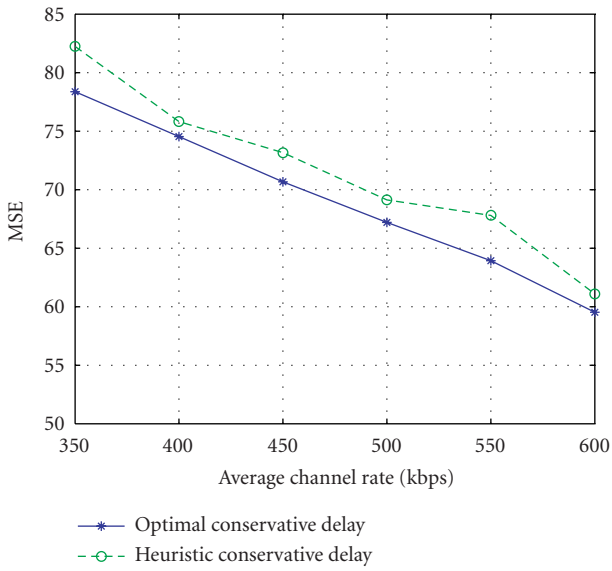


Figure 5: Quality evaluation for scheduling with heuristic and optimal δ.

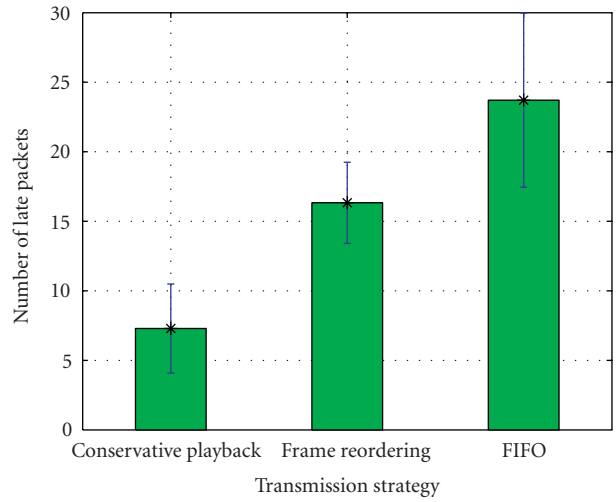


Figure 6: Late packets: conservative δ; frame reordering; FIFO scheduling.

Table 2: δ^* and δ for various average channel rates.

Rate (kbps)	350	400	450	500	550	600
Optimal δ^* (ms)	163	156	172.5	161	154	155.5
Heuristic δ (ms)	172	170	168	167	166	165
$\frac{\Omega(\delta^*)-\Omega(\delta)}{\Omega(\delta^*)}$ (%)	4.94	1.71	3.53	2.86	6.04	2.63

First we compare the results obtained by streaming with the heuristic δ, computed according to (4), and the optimal δ^*, obtained after a full search through all possible values for $\delta \in [0, \Delta]$. We use different channel average rates and we average over 10 simulations for each case. The results are presented in Figure 5. We observe that for all simulated rates, our results in terms of MSE are very close to the optimal ones. This validates our simplification to the original optimization problem, presented in Section 4. In the same time, Table 2 presents the obtained values for the heuristic and optimal δ for the same channel conditions as above, along with the relative error between the streaming performances. We observe that the values are very close and that δ^* is in general more conservative than δ. An explanation to this phenomenon resides in the fact that the sequence under consideration does not present any scene changes and the packet sizes remain constant in time.

Next, we compare the proposed conservative δ streaming with other frame reordering streaming techniques. We use a simple technique similar to the one presented in [13], which brings forward all I and P frames by two positions in the original bitstream before scheduling. Both techniques are compared in terms of number of late packet arrivals with a simple FIFO scheduling scheme that is unaware of channel rate variations. Simulation results are averaged over 100 channel realizations for an average rate of 500 kbps. Figure 6 presents the number of late packets for each of the 3 schemes with the 95% confidence intervals. We observe that the conservative δ scheme performs the best in terms of average number of late arrivals, due to the fact that the application can transparently use the difference $\Delta - \delta$ to compensate for unpredicted channel rate variations. Figure 7 presents one scheduling example for the conservative δ and frame reordering techniques. We observe that in the case of frame reordering, the strategy trades off a higher confidence in receiving I and P frames on time, at the expense of less important B frames. Hence, some B frames are lost due to late arrivals. On the contrary, the conservative δ strategy manages to schedule a similar amount of packets, and uses the extra time $\Delta - \delta$ to minimize the impact of rate variations on late arrivals. Hence, less packets are late at the receiving end of the application.

Finally, we test the proposed conservative delay scheduling method on network rate traces generated with the help of the ns-2 simulator in the presence of background traffic. We simulate 10 background flows that use the same bottleneck link as our media stream. These flows are generated according to the on/off exponential distribution, with average rates between 100 and 300 kbps. The available instantaneous rate for our streaming application is considered to be the difference between the total link bandwidth and the aggregated instantaneous rate of the background traffic. Even if the average available rate stays constant, instantaneous rate variations can be larger than 100%. We compare the performance of the scheduling obtained by using the heuristic and the optimal conservative delays, respectively, by averaging the obtained results over 10 randomly generated network rate traces. Results are presented in Figure 8 for average network rates of 300 and 450 kbps. We observe that the results are very close, even if the exact channel model is not known when the conservative delay is computed, and the channel estimation

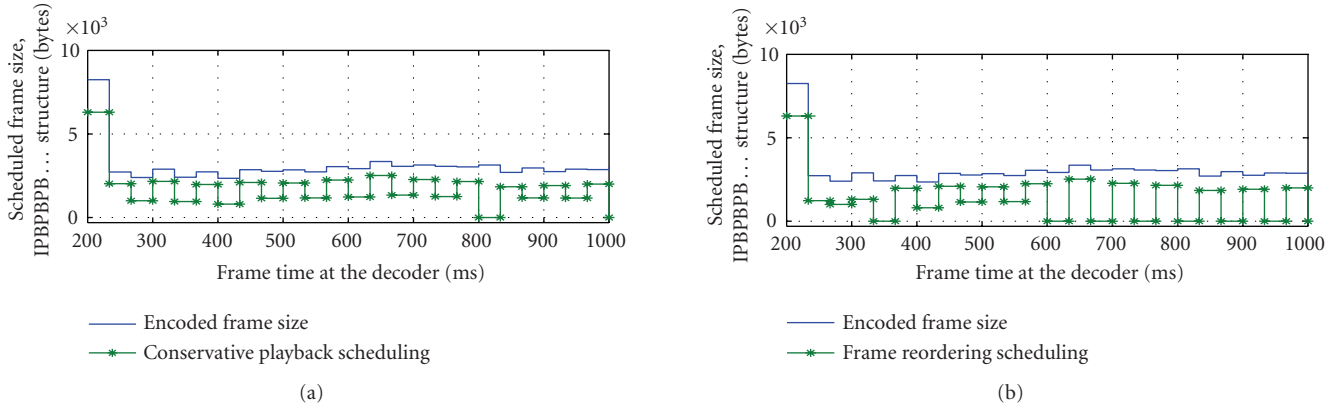


Figure 7: Example of conservative δ and frame reordering scheduling.

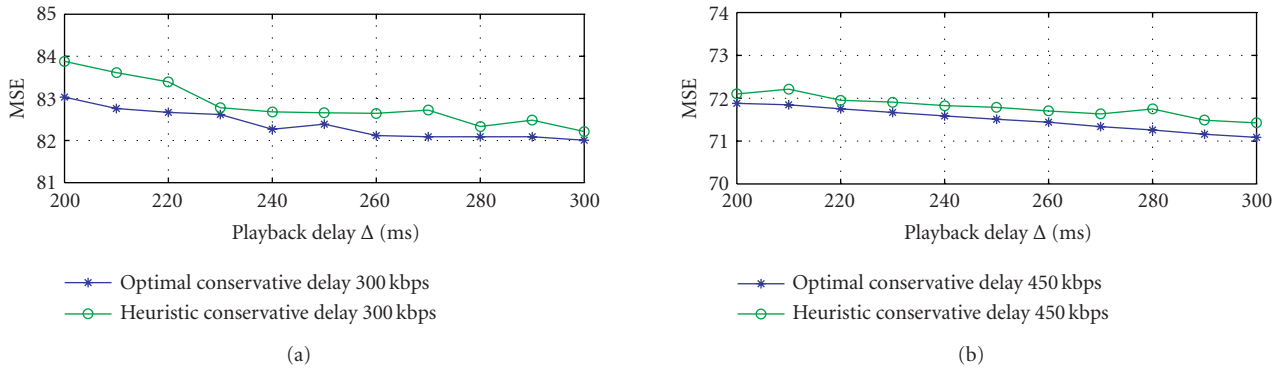


Figure 8: Quality evaluation for scheduling with heuristic and optimal δ for ns-2 network rate traces.

method is imperfect.[2] Results show that being conservative in terms of scheduling delay and initial channel rate estimate, increases the robustness of the streaming application, without significantly penalizing the received video quality. It indicates that our method is robust even in extreme cases when exact information related to the channel model is not available.

6. CONCLUSIONS

We present a new mechanism to improve the robustness of adaptive media stream scheduling algorithms against network channel variability and estimation inaccuracies. By using a conservative virtual playback delay in the scheduling process, we compensate for possible prediction errors. The difference between the conservative and actual playback delay imposed by the client transparently absorbs the negative effects of inexact rate estimation (e.g., increased packet delay at the client due to channel variations). We propose a method to determine the value of the conservative delay, as a trade-off between source quality, and robustness to bandwidth variations. The proposed solution is generic and can be employed with any given streaming mechanism. Results show that being conservative in choosing the scheduling delay pays off, even if the exact channel model is unknown (e.g., on simulated network rate traces with competing background traffic) and the rate estimation mechanism only approximates the channel rate variations over time. The simplicity of our solution and its effectiveness make it appropriate for any real-time streaming mechanism over best-effort networks.

[2] For more details on efficient bandwidth estimation mechanisms, we refer the reader to [4].

ACKNOWLEDGMENT

This work has been supported by the Swiss National Science Foundation under Grant no. PP-002-68737.

REFERENCES

[1] P. A. Chou and Z. Miao, "Rate-distortion optimized streaming of packetized media," *IEEE Transactions on Multimedia*, vol. 8, no. 2, pp. 390–404, 2006.

[2] K. Chebrolu and R. R. Rao, "Bandwidth aggregation for real-time applications in heterogeneous wireless networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 4, pp. 388–403, 2006.

[3] D. Jurca and P. Frossard, "Distortion optimized multipath video streaming," in *Proceedings of the International Packet Video Workshop*, Irvine, Calif, USA, December 2004.

[4] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, and L. Cottrell, "pathChirp: efficient available bandwidth estimation for network paths," in *Proceedings of Passive and Active Measurement Workshop (PAM '03)*, La Jolla, Calif, USA, April 2003.

[5] D. Jurca and P. Frossard, "Media streaming with conservative delay on variable rate channels," in *Proceedings of the IEEE International Conference on Multimedia and Expo (ICME '06)*, pp. 1841–1844, Toronto, Ontario, Canada, July 2006.

[6] B. Girod and N. Färber, "Feedback-based error control for mobile video transmission," *Proceedings of the IEEE*, vol. 87, no. 10, pp. 1707–1723, 1999.

[7] Y. Wang, S. Wenger, J. Wen, and A. K. Katsaggelos, "Error resilient video coding techniques," *IEEE Signal Processing Magazine*, vol. 17, no. 4, pp. 61–82, 2000.

[8] F. Wu, H. Sun, G. Shen, et al., "SMART: an efficient, scalable, and robust streaming video system," *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 2, pp. 192–206, 2004.

[9] D. G. Sachs, I. Kozinetsev, M. Yeung, and D. L. Jones, "Hybrid ARQ for robust video streaming over wireless LANs," in *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC '01)*, pp. 317–321, Las Vegas, Nev, USA, April 2001.

[10] R. Zhang, S. L. Regunathan, and K. Rose, "End-to-end distortion estimation for RD-based robust delivery of pre-compressed video," in *Proceedings of the 35th IEEE Annual Asilomar Conference on Signals, Systems and Computers*, vol. 1, pp. 210–214, Pacific Grove, Calif, USA, November 2001.

[11] C.-Y. Hsu, A. Ortega, and M. Khansari, "Rate control for robust video transmission over burst-error wireless channels," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 5, pp. 756–773, 1999.

[12] D. Jurca and P. Frossard, "Video packet selection and scheduling for multipath streaming," to appear in *IEEE Transactions on Multimedia*.

[13] S. Wee, W.-T. Tan, J. Apostolopoulos, and M. Etoh, "Optimized video streaming for networks with varying delay," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME '02)*, vol. 2, pp. 89–92, Lausanne, Switzerland, August 2002.

[14] J.-W. Ding, Y.-M. Huang, and C.-C. Chu, "An end-to-end delivery scheme for robust video streaming," in *Proceedings of 2nd IEEE Pacific Rim Conference on Multimedia (PCM '01)*, vol. 2195 of *Lecture Notes In Computer Science*, pp. 375–382, Bejing, China, October 2001.

[15] A. Sang and S.-Q. Li, "A predictability analysis of network traffic," in *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFCOM '00)*, vol. 1, pp. 342–351, Tel Aviv, Israel, March 2000.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2007, Article ID 64695, 10 pages
doi:10.1155/2007/64695

Research Article

Survival of the Fittest: An Active Queue Management Technique for Noisy Packet Flows

Shirish S. Karande, Kiran Misra, and Hayder Radha

Department of Electrical and Computer Engineering (ECE), Michigan State University, East Lansing, MI 48824, USA

Received 7 November 2006; Accepted 21 December 2006

Recommended by Jianfei Cai

We present a novel active queue management (AQM) technique to demonstrate the efficacy of practically harnessing the predictive utility of SSR indications for improved video communication. We consider a network within which corrupted packets are relayed over multiple hops, but a certain percentage of packets needs to be dropped at an intermediate node due to congestion. We propose an AQM technique, *survival of the fittest* (SOTF), to be employed at the relay node, within which we use packet state information, available from SSR indications and checksums, to drop packets with the highest corruption levels. On the basis of actual 802.11b measurements we show that such a side information (SI) aware processing within the network can provide significant performance benefits over an SI-unaware scheme, *random queue management* (RQM), which is forced to randomly discard packets. With trace-based simulations, we show the utility of the proposed AQM technique in improving the error recovery performance of cross-layer FEC schemes. Finally, with the help of H.264-based video simulations these improvements are shown to translate into a significant improvement in video quality.

1. INTRODUCTION

The traditional version of the 802.11b MAC [1, 2] drops all the packets that contain any bit corruptions. While such protocols definitely help in modularizing the system architecture, under certain channel conditions, the packet drops due to bit corruptions can be prohibitively high, especially for bandwidth hungry multimedia applications. Consequently, many recent studies (e.g., [3–17]) have recommended the development of protocols that do not drop corrupted packets and recover information even from partially damaged packets. In this paper, we collectively refer to such (and only such) protocols as cross-layer protocols. Prior investigations into cross-layer protocols have reported significant utility for delivery of video over wireless channels.

In a traditional protocol, a packet is either erased (i.e., dropped) or received without any distortions (i.e., errors). Thus, a receiver has complete channel state information (CSI). As against this, in cross-layer protocols, where the channel conditions are defined by the bit error rates (BERs) in the corrupted packets, a receiver does not inherently have CSI about the corruption levels in individual packets. Consequently, even though it is well known that receiver side CSI leads to a capacity gain (see [18, 19]). Most of the prior work on the discussed cross-layer protocols completely ignores the need and utility of such CSI. In [14] to motivate the utility of monitoring the corruption levels in packets we consider the use of checksums to differentiate between corrupted and un-corrupted packets. Even, such a simplistic binary CSI proved to have significant utility in improving error recovery. However to gain further performance gains it is necessary to identify practical mechanism that could facilitate further differentiation between corrupted packets. Motivated by the above discussion, in [16, 17, 20] we observed that typical 802.11b radio devices can associate signal-to-silence ratio (SSR) indications with individual packets. Experimental studies in [20] suggested that these SSR indications could be used to infer the BER in each individual packet with sufficient accuracy. In [16] it was shown that such improved CSI obtained from SSR indications can be used to realize performance gains in error recovery and video quality.

The predictive utility of SSR indications facilitates the design of novel cross-layer protocol optimizations, which would otherwise be infeasible. In this work, to exhibit the practical utility of this prediction tool, we consider a network scenario which has been largely ignored by studies that recommend relay of corrupted packets. We consider the relay of corrupted packets over multiple hops in presence of congestion. Congestion control [21] is a component essential to maintain the operability of any network. Hence, to scale cross-layer protocols to larger networks it is essential to develop resource allocation strategies that are optimized to work in presence of corrupted packets. We believe that prior works have not investigated such strategies due to the lack of a practical predictive tool. In this work, we exhibit the utility of SSR indications in optimizing the resource allocations within the network. We propose an active queue management scheme, *survival of the fittest* (SOTF), under which the relay node ranks the received packets in accordance with the BERs predicted by the side information. Thus under SOTF, rather than randomly discarding the packets, we propose to discard the packets that are suspected to have the highest corruption levels. If the BER estimates are robust, then such a mechanism naturally reduces the total bit corruptions seen by the eventual receiver leading to an improvement in the capacity. To the best of our knowledge, this is a first attempt to realize such a queue management mechanism.

To demonstrate the utility of SOTF we compare its performance with a random queue management (RQM) scheme that randomly discards packets at the relay node. Our comparisons are based on experimentation with actual 802.11b error and SSR traces. We limit our analysis to a simple yet representative two-hop topology. We firstly compare the performance of the above schemes in terms of the statistical analysis of the channel conditions that are presented to the eventual receiver. Subsequently, at the receiver, for both queue management schemes, we employ an error recovery mechanism that utilizes SSR and checksums as side information. Such an error recovery mechanism currently provides the state-of-the-art performance. We employ low density parity check (LDPC) code-based FEC simulations to show that side information at the relay node can improve the performance of cross-layer protocols beyond which was the previously best reported performance [16]. Finally we utilize the proposed FEC scheme to deliver H.264 [22] compressed video. Our simulations show that SOTF can improve the video quality by peak signal-to-noise ratio (PSNR) gain of several dB.

The organization of the paper is as follows: in Section 2 we review the related works. In Section 3 we briefly review our trace collection methodology and some essential observations from [16, 20]. In Section 4, we describe the network model and the proposed SOTF scheme. Section 5 focuses on statistical analysis of the BER observed by the eventual receiver under both queue management schemes. In Section 6 we present the analysis based on FEC and video simulations. Finally, in Section 7 we summarize the key conclusions of this work.

2. RELATED WORK

2.1. Cross-layer protocols that do not drop corrupted packets

Utility of relaying corrupted packets was first explored by Larzon et al. in their UDP-lite protocol [3]. This work was subsequently extended by Singh et al. for cellular video [4]. The complete UDP protocol [5], which used the frame error rate from the radio link physical layer for improved performance, was presented by Zheng and Boyce These cross-layer protocols were further extended, especially to 802.11b WLANs. The following studies have investigated various aspects of relaying corrupted packets: Servetti and De Martin have investigated design of unequal error protection [6], Masala et al. investigated performance of standard compatible multipacket combining [7], Riemann and Winstein investigated the utility of cross-layer protocols in extending the range of 802.11b [8], Pauchet et al. proposed techniques for robust source coding in presence of bit corruptions in [9], Jiang proposed a technique to recover packets from header corruptions without any additional redundancy [10]. The above studies have largely concentrated on single-hop networks, however [11, 12] exhibit the utility of cross-layer protocols in multihop networks.

The authors have been involved in the design of cross-layer protocols and the prior contributions that we build upon are the following.

(i) In [13] we investigated the utility of relaying corrupted packets in 802.11b WLANs on the basis of actual error traces. This study was the first attempt at exploring the utility of corrupted packets in 802.11b WLANs and also the first study to base their analysis on actual 802.11b error traces. In the works mentioned in the above paragraph, with the exception of [8, 9], all studies on 802.11b networks are based on some model-based simulations and not actual error traces. Additionally the work presented in [9] bases their analysis on the error traces used in [13].

(ii) In [14] we considered an abstraction of the cross-layer protocols and conducted theoretical analysis to identify important design guidelines for cross-layer protocols. Prior to this work the importance of side information in cross-layer protocols was completely over looked. In [14] we showed that realizing cross-layer protocols by merely turning off the checksums could have adverse consequences. In particular, it was shown that in the absence of side information the performance of cross-layer protocols can actually be worse than conventional protocols. Nevertheless, in [14], we exhibited that, with side information (SI) cross-layer protocols always perform better than conventional protocols, and also better than cross-layer protocols without SI.

(iii) In [14] it was shown that the improvement of cross-layer protocols could be severely diminished if a large number of packets are dropped due to corruptions in the header. Introduction of parity bits in the header,

for error correction, is not always architecturally feasible. Hence, in [15] we presented a scheme, which can detect the packet header and identity in presence of bit corruptions. Based on experiments with actual networks, it was shown that under realistic traffic load, the header detection scheme could drastically reduce the number of packet drops. A similar scheme has since been presented in [10].

(iv) In [16, 17, 20] we extend the work presented in [14]. In particular, we explore the feasibility of improving the side information mechanism. In [20] we observed that the BER in a corrupted packet has a largely environment invariant relationship with SSR indications. Hence, in [16] we used SSR indications as side information for the error recovery algorithms. We observed that utilization of SSR indications leads to significant performance gains on top of the performance achievable by just employing the recommendations in [14, 15].

The work presented in this paper improves upon the recommendations in [14–17, 20] by utilizing side information at the relay node. Our work differs significantly from prior multihop investigations, since in [11, 12, 17] no side information is used for protocol optimizations, while in [14, 16] the side information is used only at the eventual receiver.

2.2. Other related works

Performance evaluation of protocols on the basis of actual 802.11b measurements is essential to verify their practical efficacy. A number of prior works (e.g., [23, 24]) have presented performance evaluations of actual deployment of 802.11b networks, but these studies do not include analysis of corrupted frames. The authors' prior work in [13–17, 20, 25–27] is among the few studies that actually utilize bit level measurements. Additionally [16, 17, 20] are the only studies, we are aware of, that relate the bit corruptions and thus the capacity under cross-layer protocols to SSR indications. As explained previously, the presented work advances the analysis presented in [16]. It is also important to note that link quality-based real-time protocol optimizations have been investigated in many prior efforts (e.g., [28–30]). However, studies such as these mostly concentrate on packet drops and/or base their analysis on theoretical channel models. The work in [16] and the presented work are the only studies we are aware of that utilize actual SSR measurements for real-time protocol optimizations.

3. TRACE COLLECTION METHODOLOGY

The error traces used in this work are drawn primarily from the channel modeling study presented in [20] and a few similar additional measurements. The traces we use are categorized into "Home" and "Office" traces to represent a low interference and relatively high interference environment. (However, note that in both cases the amount of traffic on an identical channel is low compared to the amount of data we collected.) We base our analysis on a total of 3 million packets, where each packet is of size 1024 bytes. For all the traces used in this paper, the 802.11b PHY data rate was always maintained at 11 Mbps. The packet payload size was maintained at 1024 bytes. The trace collection methodology we employ is such that we are able to associate an SSR indication with each packet. In [20] we empirically evaluated the relationship between SSR and (a) $1-\delta$, the probability of getting a corruption-free packet (b) ε, the probability of bit error are conditioned on the fact that the packet is corrupted. To facilitate an independent reading of this document we have replicated the results presented in [20], in Figure 1. Figure 1(a) shows the relationship of SSR to $1-\delta$ and Figure 1(b) shows the relationship of SSR to ε. In [20] it was shown that SSR indications have an environment invariant relationship with $1-\delta$ and ε. In Figure 1(a) it can also be observed that the probability of packet corruption is negligible for SSR > 14 dB and the bit error rate for SSR < 8 dB is very high. Thus the

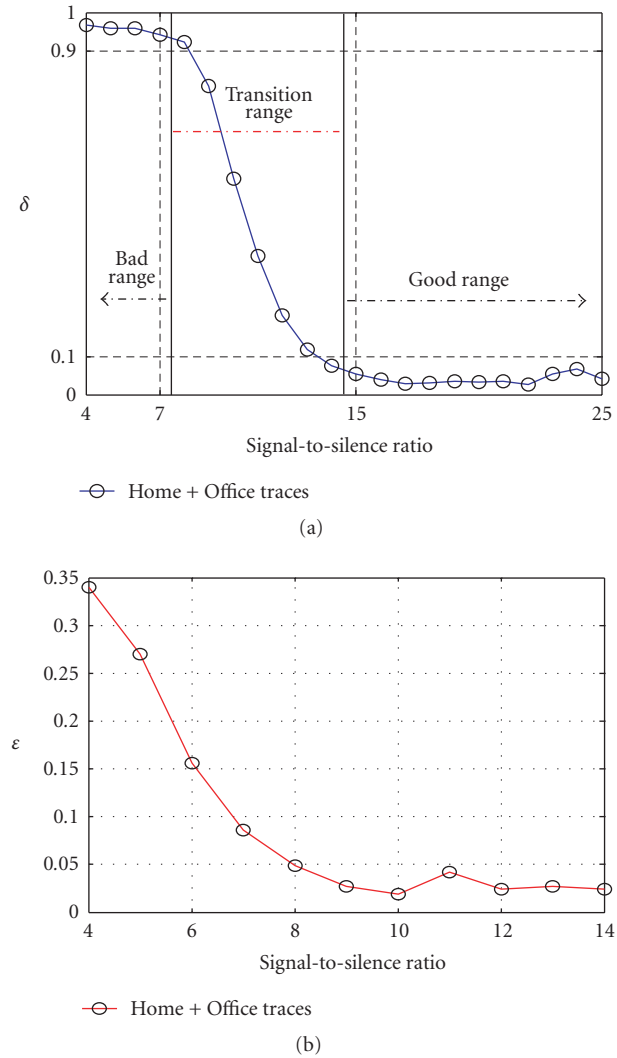


Figure 1: (a) δ(SSR), probability of receiving an error-free packet (b) ε(SSR), probability of bit error in a corrupted packet as a function of SSR [16].

utility of cross-layer protocol is primarily exhibited when a significant proportion of the packets is received with SSR values in the "transition range" (8–14 dB) [16, 17, 20]. Consequently, we focus our analysis on the error traces primarily in this range.

4. NETWORK MODEL

The network topology we consider in this paper is motivated by the illustration in Figure 2(a). Figure 2(a) shows a media server connected to a wireless access point (AP), A. Through A it serves video to clients C, D, and E. The content meant for client C has to be routed through an intermediate wireless router B. However, router B is also involved in cross-traffic from X to Y. Here X and Y could represent a single node or a group of nodes. Since at router B the access to the medium has to be shared, the bandwidth available to the video flow on link B-C is less than that available on link A-B. Additionally as the transmissions at A are meant for multiple receivers, it is not always feasible to optimize the rate allocations, at the media server, purely for client C. In such a scenario router B has to drop a certain number of packets due to congestion. The packets at router B can be dropped actively to avoid congestion or get dropped due to buffer overflow. The network topology illustrated by Figure 2(a) could be considered to represent a multihop 802.11b WLAN or could be considered to be a part of larger ad hoc/mesh network. Consequently, even though we focus on a simple network topology, the work presented in this paper can form an important component that can be used to improve communication in larger networks.

In this work we will concentrate on the flow to client C. Additionally, to avoid unnecessarily obscuring the presented discussion, we will assume that the link B-C is corruption-free. Thus the topology illustrated in Figure 2(a) can be represented by Figure 2(b). The packet drops at node B can be represented by a pseudolink B-$\underline{B}$. We assume that an AQM methodology is employed at the node B and packets are discarded actively to avoid overflow. The processing at router B can be described by the flow chart provided in Figure 3. The AQM techniques we consider employ (a) a collection buffer (CB) where all received packets, meant for client C, are input (b) a transmission buffer (TB) from which we opportunistically transmit packets (c) a *packet discarding mechanism* (PDM). The PDM removes blocks of u packets from the CB to always maintain the length of the CB to less than u packets. The PDM then discards v packets from the block of u packets before inputting these packets into the TB for transmission. The parameter v is controlled to ensure that no overflows occur in the TB. Under a steady-state assumption, the length of the TB can be maintained to a value slightly greater than u packets. The buffers CB and TB always operate in a first-in first-out (FIFO) mode. In all the simulations in this work, we assume constant flow loads and thus value of v is maintained constant. The controllability of v under time-varying loads is not a focus of this work and hence detailed investigation from a control perspective is part of future work. The network and queue management model we consider here is

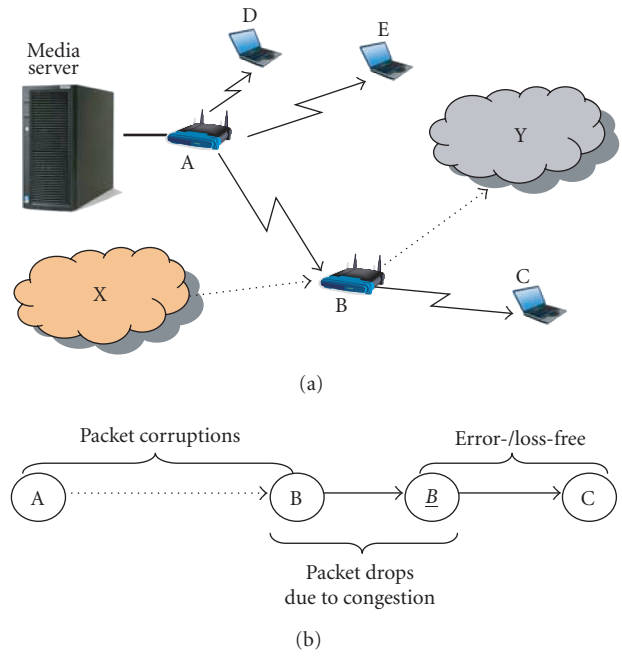


Figure 2: (a) Network topology used for the current work on AQM. (b) The representative equivalent topology actually used for simulations.

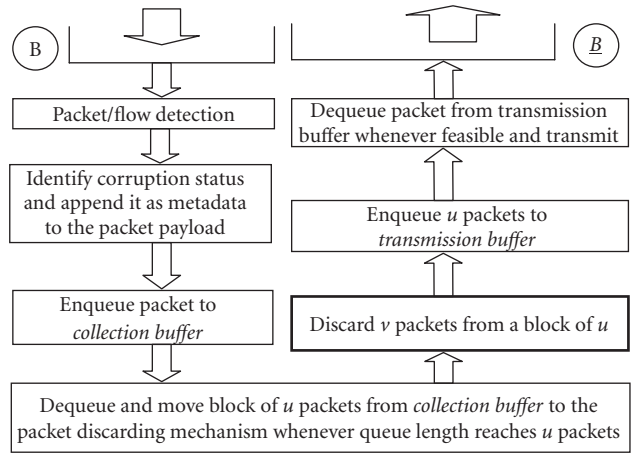


Figure 3: Processing and active queue management at router B.

simplistic, but nevertheless representative. Additionally such a simplistic model allows us to avoid unnecessarily obscuring the presentation of our contributions.

The processing at node B consists of two additional components.

(a) Packet/flow detection

The buffering mechanism we utilize here is specific to a particular flow and thus it is essential to establish the identity of a packet in presence of bit corruptions. A possible algorithm for flow detection is the header detection scheme

presented in our prior work in [15]. The header detection algorithm utilizes flow statistics and baye's rule [18] to detect the critical header fields (CHFs) in presence of bit corruptions. These CHFs are sufficient to establish the flow ID of the packet. In this work, we assume that flow from the media server to the client C forms the major portion of the traffic flowing through router B. In such circumstances, the header detection mechanism works particularly well for the corrupted packets being received from A. For additional details of header detection, we refer the reader to [15]. For the purposes of our simulation, we assume that the header detection works perfectly and thus the cross-layer protocols do not drop any packets due to corruptions.

(b) Identification of the corruption status

For efficient operation of the PDM and to facilitate efficient error recovery at the eventual receiver, it is essential to estimate the BER in each packet. We assume that a checksum has been employed on the entire packet body and thus we are able to differentiate between corrupted and uncorrupted packets. Additionally, we assume that the relay node employs an 802.11b radio device that can provide SSR indications. More specifically, in this paper, we assume that the device is identical to the one used in [16, 20] and hence the corruption status can be determined by utilizing the relationships provided in Figure 1(b). The corruption status of a packet i, with slight abuse of notation, is denoted by ε_i. If the checksums are successful we set $\varepsilon_i := 0$, else we set $\varepsilon_i := \varepsilon(\mathrm{SSR}_i)$ where SSR_i is the SSR indication associated with the packet i. To permit side information-based processing at the eventual receiver, the corruption status of the packet has to be recorded and appended to the packet body as metadata. To record the corruption status we can either record the quantized version of ε_i or record the checksum result along with the SSR indication. In either case we assume that the cost of accurately relaying such metadata over link B-C is negligible.

4.1. Efficient packet discarding

We consider two packet discarding mechanisms in this work.

(a) Random queue management

In this scheme the PDM completely disregards the corruption status of individual packets and discards packets randomly.

(b) Survival of the fittest

In this scheme the PDM sorts the packets in a block of u packets in ascending order of corruption status. In such a sorted list, v packets with the highest corruption status are discarded.

In this paper, as described before, we focus on exhibiting the performance improvement that SOTF can provide over RQM, especially in terms of video quality. To the best of our knowledge this paper is the only study that investigates AQM in presence of bit corruptions. Hence we have to limit our performance comparisons to RQM and SOTF.

5. STATISTICAL ANALYSIS

In this section, we compare the performance of SOTF and RQM in terms of the BER that is observed by the eventual receiver in the relayed packets. SOTF seeks to discard the most corrupted packets and hence in principle should be able to facilitate lower BER. However, performance improvements due to SOTF are critically dependent on the predictive utility of SSR indications. Hence the first statistic we measure is $p(\mathrm{BER}_{\mathrm{RQM}} > \mathrm{BER}_{\mathrm{SOTF}})$, which represents the proportion of packet blocks for which the BER observed by the eventual receiver under SOTF is less than that observed under RQM. Prior to presenting the results of such measurements, it is important to note that existence of corrupted packets is essential to differentiate between RQM and SOTF. Hence, we base our analysis only on those blocks of packets that have at least one corrupted packet. The above restriction is roughly equivalent to continuously operate under 20 dB.

In accordance with the above discussion, we evaluated $p(\mathrm{BER}_{\mathrm{RQM}} > \mathrm{BER}_{\mathrm{SOTF}})$ on our entire data set. The results of our empirical measurements, for various block lengths u and congestion-based packet drops v, are presented in Table 1. It can be observed in Table 1 that 90% of the times, even for short block lengths, SOTF performs better than RQM. Such an observation remains true irrespective of the number of packets being dropped due to congestion. Additionally as the block length is increased, the guarantee with which SOTF provides improved performance also increases.

The results in Table 1 are sufficient to indicate that with a high likelihood, SOTF should perform better than RQM. However to quantify the magnitude of average improvement we evaluated $E[\Delta\mathrm{BER}] = E[\mathrm{BER}_{\mathrm{RQM}} - \mathrm{BER}_{\mathrm{SOTF}}]$, where $\mathrm{BER}_{\mathrm{RQM}}$ and $\mathrm{BER}_{\mathrm{SOTF}}$ are random variables that represent the BERs observed in each u-v packet block received by the eventual receiver. Table 2 presents the results of our evaluations. It can be seen that SOTF can reduce the average BER by 0.3 to 1%. The average BER we observed in RQM was 4.5%, hence the reduction in BER due to SOTF can be considered significant.

Finally, it is important to note that the link quality experienced by each block of packets can vary significantly. Hence, the BER in each block and the performance improvement due to SOTF can also vary. To illustrate the performance improvement provided by SOTF in presence of such variations, in Figure 4 we present scatter plots for Home and Office setup. We arbitrarily chose $u = 64$ and $v = 6$ for the measurements presented in Figure 4, but similar results can be obtained for other parameter values. From Figure 4 it can be observed that even for identical values of average SSR the performance improvements due to SOTF can vary significantly. There indeed exist a few cases when SOTF performs worse than RQM. However despite the variation in performance improvement, for most of the blocks, irrespective of the average SSR, SOTF provides reductions in BER. Thus results in this section provided substantial statistical evidence of the

Table 1: Empirical evaluation of $p(\text{BER}_{\text{RQM}} > \text{BER}_{\text{SOTF}})$.

v/u	Packet block length u			
	32	64	128	256
0.03	0.88	0.92	0.95	0.97
0.06	0.89	0.93	0.95	0.97
0.09	0.85	0.91	0.95	0.97
0.12	0.90	0.93	0.95	0.97
0.15	0.87	0.92	0.95	0.97

Table 2: Evaluation of $E[\Delta\text{BER}] = E[\text{BER}_{\text{RQM}} - \text{BER}_{\text{SOTF}}]$ in 10^{-2}.

v/u	Packet block length u			
	32	64	128	256
0.03	0.3030	0.3417	0.3649	0.3747
0.06	0.5716	0.6029	0.5950	0.5779
0.09	0.7165	0.7480	0.7269	0.6842
0.12	0.9639	0.9223	0.8514	0.7892
0.15	1.0211	0.9896	0.9006	0.8361

capacity gain that can be facilitated to the eventual receiver by employing SOTF at the relay node.

6. SIMULATIONS

6.1. Error recovery performance of LDPC -based FEC

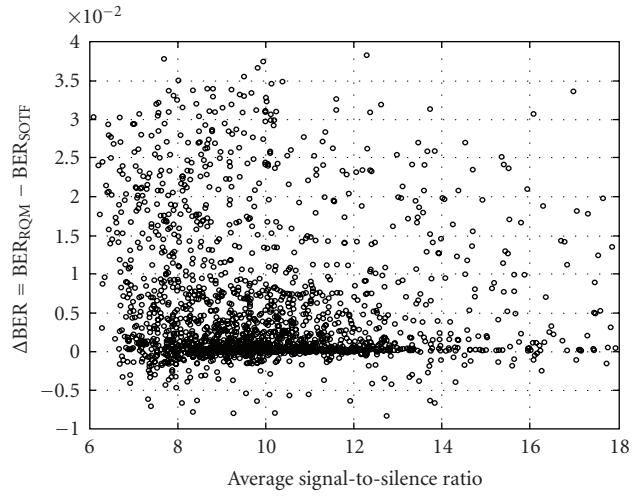
In this section we exhibit the utility of SOTF in improving the performance of a cross-layer FEC and hence in improving the video quality. Without loss of generality for the remainder of the document we will focus our analysis on parameter values $u = 64$ and $v = 6$. For our simulations we assume that the media server employs an LDPC-based FEC scheme. The FEC scheme is block-based and each block consists of 64 packets of size 1024 bytes. Each FEC block can be broken down into multiple code words. The length of the individual code words and the total code words in each FEC block vary in accordance with the chosen channel coding rate. However, we maintain the number of message bits in each code word to 8192 bits. We realize FEC schemes with varying degree of robustness by increasing the length and hence the redundancy in each code word. We consider code length varying from 9216 bits to 16384-bits. Thus, we consider channel-coding rates varying from 0.89 to 0.5. The code words are mapped to the FEC block by interleaving them equally across all the packets in the block. For example, a 16384- bit code word is interleaved across 64 packets in the FEC block by mapping 256 bits from the code word to each packet. Thus in this case the FEC block consists of 32 code words. Additionally note that the LDPC codes we utilize are regular codes and are girth optimized using progressive edge growth (PEG) techniques [31].

At the eventual receiver, the LDPC-based FEC scheme is decoded using log-likelihood ratio (LLR) domain sum-product algorithm. For background information about this algorithm and LDPC codes please refer to [31]. The LLR initialization $L(c_i)$ associated with code bit c_i is dependent on the received bit y_i and an prior assumption of the bit error probability. As suggested in [16], we employ the following initialization.

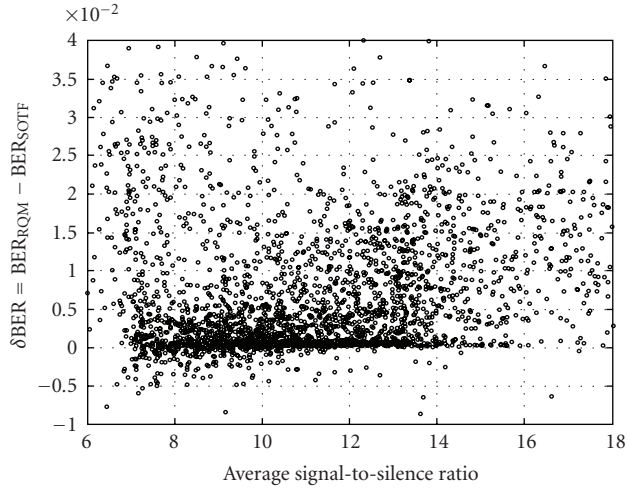
(i) If bit belongs to a dropped/discarded packet, then set $L(c_i) = 0$.

(ii) If bit belongs to an uncorrupted packet, then set the LLR to ∞ if the received bit is 0 or to $-\infty$ if the received bit is 1.

(iii) If the bit belongs to a corrupted packet with SSR indication SSR, then set the LLR as $L(c_i) = (-1)^{y_i} \log((1 - \varepsilon(\text{SSR}))/\varepsilon(\text{SSR}))$ where the relationship $\varepsilon(\text{SSR})$ is obtained by utilizing the corruption status provided by the metadata and the relationship provided in Figure 1.



(a) Home traces



(b) Office traces

Figure 4: The above figures present an XY-plot (Average SSR, ΔBER) for various packet blocks. The block length is fixed at $u = 64$ and the number of packets discarded per block due to congestion is fixed at $v = 6$.

Note that as per the above discussion we assume that at the receiver, the error recovery mechanism can utilize SSR indications as side information irrespective of the AQM technique employed at the relay node. Finally, it should be mentioned that the LDPC decoding algorithm we employ stops upon converging to a code word or when the number of iterations exceeds 25.

Prior to discussing the results, it is essential to take note of the fact that the capacity of wireless channel can vary significantly with variation in SSR. Even in the transition region the capacity of the wireless channel can vary from almost 1 to well below 0.5. The traces we utilize for our simulations have been collected under a variety of channel conditions and hence it is impossible to guarantee a successful decoding for all code words without choosing an impractically low channel coding rate (<0.5). Hence in our analysis presented here we do not make an explicit effort to identify a channel coding rate that recovers almost all the code words. Instead we compare the performance of RQM and SOTF in terms of the chosen set of rates and interpret the results from an outage perspective [18]. In practice we expect our investigations to be combined with other error recovery optimizations (e.g., [19, 32]) however to maintain the clarity of our presentation we avoid these optimizations for now.

Figure 5 shows the result of our LDPC simulations. In Figure 5(a) the performance improvement due to SOTF can be clearly observed. Even at low channel coding rates (0.5 to 0.7) SOTF can improve error recovery performance by over 10% for the Office data and by over 5% for Home data. The performance improvement due to SOTF can be appreciated further by observing the results presented in Figure 5(b). Figure 5(b) shows the difference in the average number of iterations required to process a received word. It can be seen that even at low coding rates SOTF can reduce the number of iterations by over 10%. Such reduction in complexity can be important for power-constrained wireless receivers. Thus the results in this section are consistent with our observation in Figure 4 and provide further evidence of the utility of SOTF. In the subsequent section we will demonstrate the impact of improved error recovery on the quality of video.

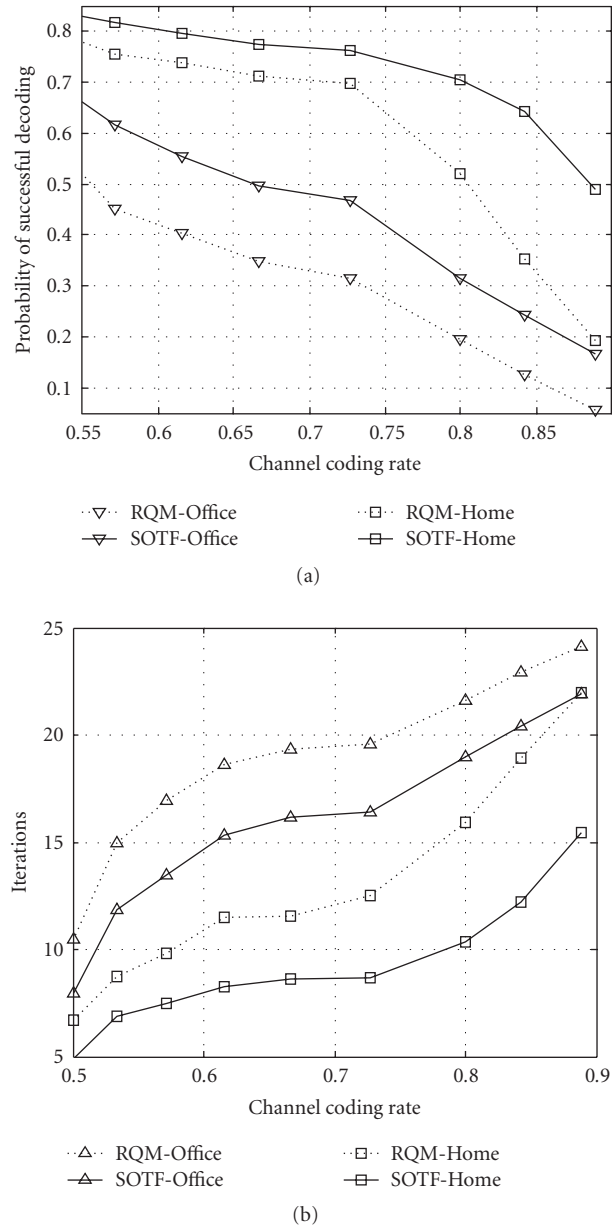
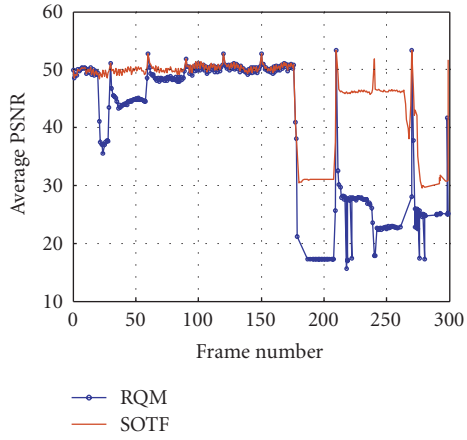


Figure 5: Comparison of SOTF and RQM in terms of the error recovery performance of LDPC based FEC. (a) Probability of successful decoding. (b) Average number of iterations per code word. The block-length is fixed at $u = 64$ and the number of packets discarded per block due to congestion is fixed at $v = 6$.

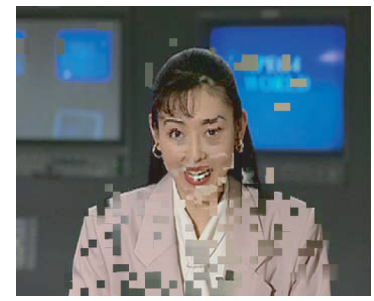
6.2. Comparison based on video quality

In this section we compare the performance of SOTF and RQM on the basis of video quality. We assume that the media server employs an H.264-based source encoder. The compressed video bit stream can be broken into independently decodable video packets of size 8192 bits. The server maps each of these video packets to a single code word in the LDPC-based FEC scheme. At the receiver the compressed video data is recovered from the output of the FEC decoder. If a code word is decoded successfully, then the video packet is sent to the H.264 video decoder, however if the channel decoding is unsuccessful, then the video packet is dropped. For our simulations we employ version 10.1 of the reference software for H.264. The source decoder employs the error concealment feature of "*frame copying*." Our analysis in this section is based on the standard video test sequences Akiyo, Foreman, and Stefan. We use the frame resolution "CIF" and frame frequency of 30 frames/sec. The GOP structure we employ is of the form IBPBP with a GOP size of 30 frames. The source coding employs constant quantization, however the quantization parameter is chosen so as to achieve a bit rate of 1.2 Mbps for the compressed video stream. Additionally, it should be highlighted that the 802.11b error traces we employ for our video simulations are chosen arbitrarily from the larger set of traces. The actual performance metrics can vary on the basis of the chosen traces, nevertheless, the results we

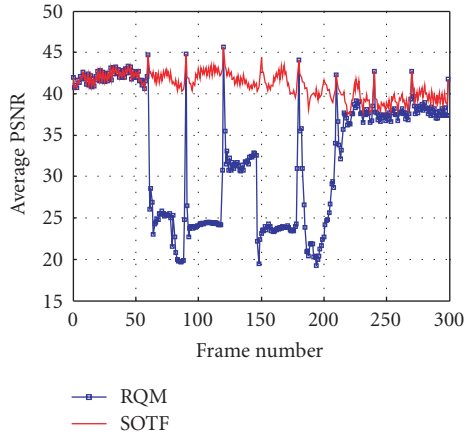


(a)

(b) SOTF, frame 250



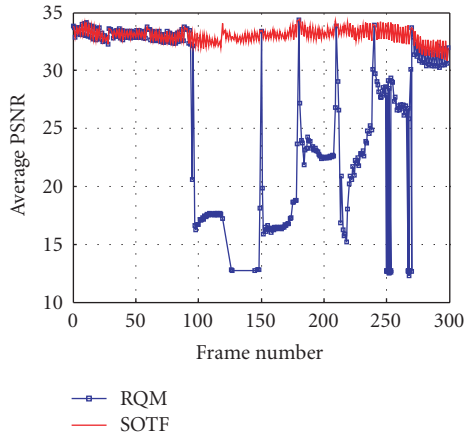
(c) RQM, frame 250



(d)

(e) SOTF, frame 100

(f) RQM, frame 100



(g)

(h) SOTF, frame 140

(i) RQM, frame 140

FIGURE 6: Video quality comparison of RQM and SOTF. Above results are based on trace samples obtained from the Home setup and channel coding rate of 0.66. (a), (d), (g) provide a temporal snapshot of PSNR performance. Picture frame-based subjective comparison is presented in (b)–(h) for Akiyo, (e)–(f) for Foreman, (h)–(i) for Stefan. Also note that (b), (e), (h) represent the performance of SOTF and (c), (f), (i) represent the performance of RQM.

Table 3: Video quality: average gain in PSNR (in dB).

Sequence	Akiyo		Foreman		Stefan	
environment	Home	Office	Home	Office	Home	Office
Coding rate						
0.57	0.2	7.19	0.65	3.99	1.48	2.09
0.61	0.2	9.11	2.55	3.99	1.48	2.03
0.66	9.74	11.18	6.16	8.27	7.23	8.43

present in this section are representative of the comparative performance trends observed on all the error traces.

Table 3 presents the results of the video simulations. The performance gain in PSNR due to SOTF has been recorded in Table 3. It can be observed that even at a coding rate of 0.57 the performance gain due to SOTF in the chosen Office traces can be in excess of 2 dB. A similar gain is not seen for all sequences for the chosen Home traces. Nevertheless as the coding rate is reduced the performance gains due to SOTF are evident under all the channel conditions. In particular it was observed that the capacity under RQM drops below 0.66 frequently, while remaining above 0.66 more often for SOTF. Hence for a coding rate of 0.66 it can be observed that video quality degradation under RQM is significantly more drastic than the degradation under SOTF. From Table 1 it can be observed that SOTF can provide PSNR gain in excess of 8 dB for all the considered traces and video sequences. The exact amount of performance improvement is dependent on the choice of error trace, coding rate, and source bit rate. Hence the exact amount of performance gains due to SOTF in actual deployments can vary, but, nevertheless the presented simulations provided sufficient evidence of its significant utility under a variety of channel conditions.

To further illustrate the performance improvement in video quality we present the results for the Home traces for the coding rate 0.66 in additional detail. Figure 6 compares the performance of SOTF and PSNR in terms of temporal snapshots in Figures 6(a), 6(d), and 6(g) and visual samples in Figures 6(b), 6(c), 6(e), 6(f), 6(h), and 6(i). From the temporal snapshots it is clearly evident that SOTF can provide drastic performance gains over muliple GOPs. Especially for the traces used for Foreman and Stefan it can be seen that, while the video quality under RQM deteriorates frequently, SOTF is able to provide quality equivalent to the lossless stream. In Figures 6(b), 6(c), 6(e), 6(f), 6(h), and 6(i) we have chosen a specific picture frame from the corresponding temporal snapshot to facilitate a visual comparison between SOTF and RQM. In the presented examples it can be seen that while the video quality under SOTF is excellent, the video under RQM is incomprehensible. Thus from the presented simulations it is clearly evident that SOTF can provide substantial improvement in video quality compared to RQM.

7. CONCLUSION

In this work we explored the feasibility of harnessing the predictive utility of SSR indications at a relay node. We considered the use of SSR indications in improving the efficiency of AQM at a relay node. We proposed a novel AQM scheme, SOTF, which utilizes side information at the relay node to identify the most corrupted packets and discard them. SOTF was compared with a scheme, RQM, which randomly discards packets without any side information. Experiments with actual 802.11b traces reveal that SOTF can significantly reduce the BER observed at the eventual receiver. Such reductions in BER are responsible for improving the capacity offered to the eventual receiver. The improvement in capacity is exhibited on the basis of LDPC-based FEC simulations and H.264-based video simulations. Our experiments clearly exhibit that SOTF can lead to significant improvement in the error recovery performance and a PSNR gain of several dB in the video quality. Thus in this work we have clearly exhibited the utility and feasibility of improving the efficiency of video communication by utilizing side information at the relay node. In future we hope to extend the work in this paper to multiple relays and to time-varying traffic patterns.

ACKNOWLEDGMENTS

The authors gratefully appreciate the support for this work under NSF Awards CCF-0515253, CNS-0430436, and MEDC Grant GR-296. The authors also appreciate Utpal Parrikar's contributions to an initial version of this work.

REFERENCES

[1] ISO/IEC 8802-11:1999(E), Part11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specications, August 1999.

[2] IEEE Std 802.11b-1999, Part11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specications: Higher-Speed Physical Layer Extension in the 2.4GHz band, September 1999.

[3] L. Larzon, M. Degermark, and S. Pink, "The UDP lite protocol," IETF Internet Draft, February 2001.

[4] A. Singh, A. Konrad, and A. D. Joseph, "Performance evaluation of UDP lite for cellular video," in *Proceedings of the 11th IEEE International Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '01)*, pp. 117–124, Port Jefferson, NY, USA, June 2001.

[5] H. Zheng and J. Boyce, "An improved UDP protocol for video transmission over Internet-to-wireless networks," *IEEE Transactions on Multimedia*, vol. 3, no. 3, pp. 356–365, 2001.

[6] A. Servetti and J. C. De Martin, "Link-level unequal error detection for speech transmission over 802.11b networks," in *Proceedings of Special Workshop in Maui (SWIM '04)*, Maui, Hawaii, USA, January 2004.

[7] E. Masala, A. Servetti, and J. C. De Martin, "Standard compatible error correction for multimedia transmissions over 802.11 WLAN," in *Proceedings of IEEE International Conference on*

Multimedia and Expo (ICME '05), pp. 880–883, Amsterdam, The Netherlands, July 2005.

[8] R. Riemann and K. Winstein, "Improving 802.11 Range with Forward Error Correction," MIT-CSAIL-TR-2005-011.

[9] F. Pauchet, C. Guillemot, M. Kerdranvat, S. Pateux, and P. Siohan, "Conditions for SVC bit error resilience testing," in *The 17th Meeting of Joint Video Team (JVT '05)*, Nice, France, October 2005.

[10] W. Jiang, "A bit error correction without redundant data: a Mac layer technique for 802.11 networks," in *Proceedings of the 2nd International Workshop on Wireless Network Measurement (WiNMee '06)*, Boston, Mass, USA, April 2006.

[11] E. Masala, M. Bottero, and J. C. De Martin, "MAC-level partial checksum for H.264 video transmission over 802.11 ad hoc wireless networks," in *Proceedings of the 61st IEEE Vehicular Technology Conference (VTC '05)*, vol. 5, pp. 2864–2868, Stockholm, Sweden, May-June 2005.

[12] S. Yi, Y. Shan, S. Kalyanaraman, and B. Azimi-Sadjadi, "Header error protection for multimedia data transmission in wireless ad hoc networks," in *Proceedings of the 13th IEEE International Conference on Image Processing (ICIP '06)*, Atlanta, Ga, USA, October 2006.

[13] S. A. Khayam, S. S. Karande, H. Radha, and D. Loguinov, "Performance analysis and modeling of errors and losses over 802.11b LANs for high-bit-rate real-time multimedia," *Signal Processing: Image Communication*, vol. 18, no. 7, pp. 575–595, 2003.

[14] S. S. Karande and H. Radha, "Hybrid erasure-error protocols for wireless video," *IEEE Transactions on Multimedia*, vol. 9, no. 2, pp. 307–319, 2007.

[15] S. A. Khayam, S. S. Karande, M. U. Ilyas, and H. Radha, "Header detection to improve multimedia quality over wireless networks," *IEEE Transactions on Multimedia*, vol. 9, no. 2, pp. 377–385, 2007.

[16] S. S. Karande, U. Parrikar, K. Misra, and H. Radha, "Utilizing SSR indications for improved video communication in presence of 802.11b residue errors," in *Proceedings of IEEE International Conference on Multimedia and Expo (ICME '06)*, pp. 1973–1976, Toronto, ON, Canada, July 2006.

[17] S. S. Karande, K. Misra, and H. Radha, "CLIX: network coding and cross-layer information exchange of wireless video," in *Proceedings of the 13th IEEE International Conference on Image Processing (ICIP '06)*, Atlanta, Ga, USA, October 2006.

[18] T. M. Cover and J. A. Thomas, *Elements of Information Theory*, Wiley Series in Telecommunications and Signal Processing, John Wiley & Sons, New York, NY, USA, 2006.

[19] S. C. Draper, B. Frey, and F. R. Kschischang, "Rateless coding for non-ergodic channels with decoder channel state information," http://www.eecs.berkeley.edu/~sdraper/researchDir/pdf/ratelessTimeVary.pdf.

[20] S. S. Karande, U. Parrikar, K. Misra, and H. Radha, "On modeling of 802.11b residue errors," in *Proceedings of the 40th Annual Conference on Information Sciences and Systems (CISS '06)*, pp. 283–288, Princeton, NJ, USA, March 2006.

[21] R. Srikant, *The Mathematics of Internet Congestion Control*, Birkhäuser, Boston, Mass, USA, 2003.

[22] ISO/IEC JTC 1/SC29/WG11 and ITU-T SG16 Q.6, "Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264 — ISO/IEC 14496-10 AVC)," Doc. JVT.

[23] D. B. Faria and D. R. Cheriton, "No long-term secrets: location-based security in overprovisioned wireless LAN," in *Proceedings of the 3rd ACM Workshop on Hot Topics in Networks (HotNets-III '04)*, San Diego, Calif, USA, November 2004.

[24] M. Rodrig, C. Reis, R. Mahajan, D. Wetherall, and J. Zahorjan, "Measurement-based characterization of 802.11 in a hotspot setting," in *Proceedings of the ACM Workshop on Experimental Approaches to Wireless Network Design and Analysis (SIGCOMM '05)*, pp. 5–10, Philadelphia, Pa, USA, August 2005.

[25] S. A. Khayam and H. Radha, "Linear-complexity models for wireless MAC-to-MAC channels," *Wireless Networks*, vol. 11, no. 5, pp. 543–555, 2005.

[26] S. A. Khayam and H. Radha, "Constant-complexity models for wireless channels," in *Proceedings of the 25th Annual Joint Conference on the IEEE Computer and Communications Societies (INFOCOM '06)*, Barcelona, Catalonia, Spain, April 2006.

[27] S. A. Khayam, H. Radha, S. Aviyente, and J. R. Deller Jr., "Markov and multifractal wavelet models for wireless MAC-to-MAC channels," *Elsevier Performance Evaluation Journal*, vol. 64, no. 4, pp. 298–314, May 2007.

[28] S. Shakkottai, T. S. Rappaport, and P. C. Karlsson, "Cross-layer design for wireless networks," *IEEE Communications Magazine*, vol. 41, no. 10, pp. 74–80, 2003.

[29] M. van der Schaar, S. Krishnamachari, S. Choi, and X. Xu, "Adaptive cross-layer protection strategies for robust scalable video transmission over 802.11 WLANs," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 10, pp. 1752–1763, 2003.

[30] M. van der Schaar and S. Shankar, "Cross-layer wireless multimedia transmission: challenges, principles, and new paradigms," *IEEE Wireless Communications*, vol. 12, no. 4, pp. 50–58, 2005.

[31] X.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Transactions on Information Theory*, vol. 51, no. 1, pp. 386–398, 2005.

[32] http://www.icsi.berkeley.edu/PET/.

Hindawi Publishing Corporation
Advances in Multimedia
Volume 2007, Article ID 61491, 8 pages
doi:10.1155/2007/61491

Research Article

Video Transmission over MIMO-OFDM System: MDC and Space-Time Coding-Based Approaches

Haifeng Zheng,[1] Congchong Ru,[2] Chang Wen Chen,[3] and Lun Yu[1]

[1] *Department of Information and Communication Engineering, Fuzhou University, Fuzhou, Fujian 350002, China*
[2] *Department of Electronic Engineering, Tsinghua University, Beijing 100084, China*
[3] *Department of Electrical and Computer Engineering, Florida Institute of Technology, Melbourne, FL 32940, USA*

Received 16 November 2006; Accepted 21 December 2006

Recommended by Jianfei Cai

MIMO-OFDM is a promising technique for the broadband wireless communication system. In this paper, we propose a novel scheme that integrates multiple-description coding (MDC), error-resilient video coding, and unequal error protection strategy with hybrid space-time coding structure for robust video transmission over MIMO-OFDM system. The proposed MDC coder generates multiple bitstreams of equal importance which are very suitable for multiple-antennas system. Furthermore, according to the contribution to the reconstructed video quality, we apply unequal error protection strategy using BLAST and STBC space-time codes for each video bitstream. Experimental results have demonstrated that the proposed scheme can be an excellent alternative to achieve desired tradeoff between the reconstructed video quality and the transmission efficiency.

1. INTRODUCTION

Video communication over wireless network has been a significant challenge for current multimedia technology. As it is well known, wireless channels often suffer from multipath fading, shadowing, intersymbol interference, and so forth. Meanwhile, compressed video is very sensitive to error-prone environment. Any transmission error may lead to the loss of decoding synchronization and severe degradation to the received video quality.

Fortunately, great progresses have been made in the recent development of wireless communication and video transmission. Orthogonal frequency-division multiplexing (OFDM) has become a promising technique for transmission of signals in the broadband wireless communication systems. Moreover, multiple antennas system with multiple transmitters and multiple receivers, called a multiple-input and multiple-output (MIMO) system, has been shown to be an effective way to transmit high data rate over wireless channels. Therefore, OFDM in conjunction with multiple-input and multiple-output is not only able to enhance the capacity of system but also able to combat the channel fading and interference effectively.

It is widely believed that multiple-transmit and-receive antennas can improve the performance of wireless systems. Actually, this benefit can be exploited in two ways: spatial diversity and spatial multiplexing. Spatial diversity can improve the reliability of reception to combat channel fading by sending signals that carry the same information through independent paths. Such diversity schemes can be implemented using trellis-based space-time codes and orthogonal designs. For example, space-time block codes (STBCs) have been proposed to achieve diversity gains [1]. On the other hand, spatial multiplexing can provide higher data rate by transmitting different data streams in parallel through independent channels. Several schemes have been proposed to exploit the benefit of spatial multiplexing phenomenon. For example, Bell Labs space-time architecture (BLAST) [2] is such a scheme. In another words, multiple-transmit and-receive antennas can be used to provide higher reliability of reception using spatial diversity or give higher throughout using spatial multiplexing. In this paper, we will exploit the benefits from both types of these two schemes for robust video transmission over wireless channels.

Meanwhile, in image and video transmission applications, several approaches have been proposed to improve the robustness of image and video transmission over error-prone network such as Internet and wireless network. Multiple-description coding (MDC) is such an effective source coding method. MDC generates multiple encoded bitstreams

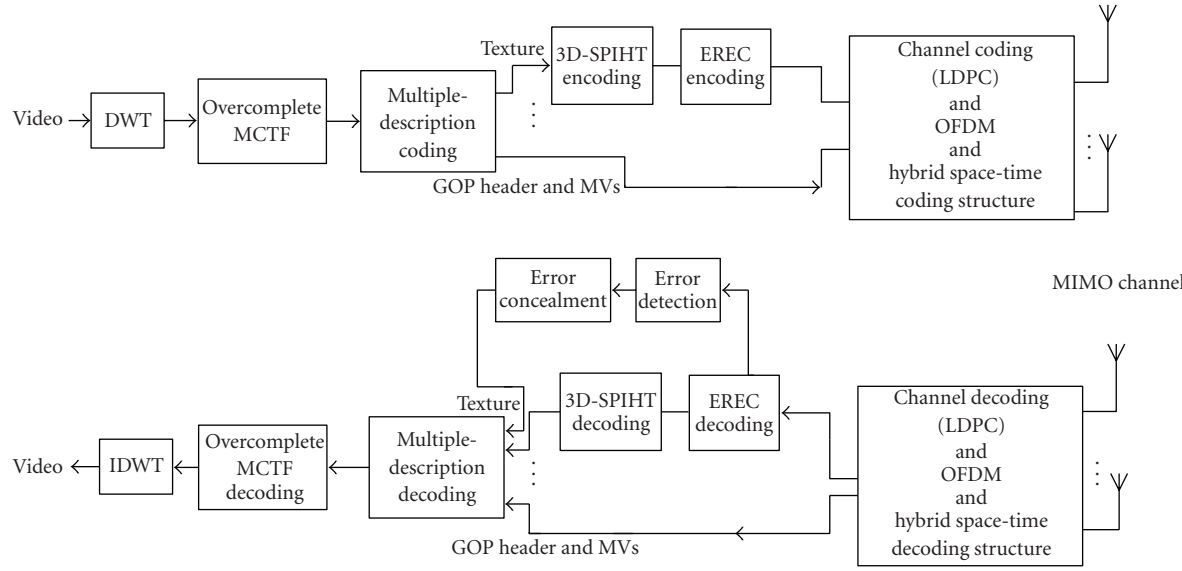


Figure 1: The overall block diagram of the proposed system.

that are equally important and independent. The objective of MDC is that if all bitstreams have been received correctly, a high signal quality can be reconstructed, whereas, if some bitstreams have been lost, a low-quality, but acceptable signal quality can still be reconstructed from the received description. MDC uses the idea of diversity to transmit bitstreams along different channels with the premise that each channel experiences independent failure events, such that the probability of receiving the information from each channel is equal. This assumption is also true in MIMO channel case. In a system with transmit and receive antennas, the channel between individual antenna pairs is assumed to be independent and identically distributed (i.i.d.). Moreover, the probability of all channels between antenna pairs falling into deep fading simultaneously is small. Therefore, MDC is also very suitable for robust video transmission over multiple antennas system

There have been several works to report video transmission over multiple antennas system [3–7]. However, none of these existing schemes have explored the integration of data bitstreams transmission using MDC and MIMO-OFDM. In this research, we construct a new system that integrates multiple-description coding, error-resilient video coding, unequal error protection scheme using hybrid space-time coding structure for robust video transmission over MIMO-OFDM system. In this paper, we propose a multiple-description coding scheme based on wavelet video coding, where each generated description still retains the main quality of the original image. In the present of lost descriptions, we can perform error concealment method to compensate the lost descriptions without introducing some amount of redundancy between the descriptions. Unlike traditional multiple-description schemes, the proposed multiple-description algorithm enables us to generate more than two bitstreams that may be more appropriate for multiple-antenna transmission of compressed video.

In addition, to improve the error resilience of each description, each description of the compressed video will be further decomposed into multiple bitstreams. We also introduced error-resilient entropy coding (EREC) [8] to solve the problems of packetization and synchronization for multiple bitstreams transmission. Furthermore, as mentioned before, in MIMO system, spatial diversity performs more robustly, while spatial multiplexing provides higher data rate increment when given a fixed reliability level. This work is inspired by this idea. In this work, diversity scheme can be used to achieve a better error protection for important data such as motion vectors and GOP header information, whereas multiplexing scheme can be applied to obtain high transmission data rate. Therefore, we use such hybrid space-time coding structure to achieve unequal error protection for video transmission.

The rest of the paper is organized as follows. The proposed transmission scheme is described in Section 2. Then, simulation results are presented in Section 3 to demonstrate the effectiveness of the proposed scheme. Finally, we conclude in Section 4 with some discussion.

2. THE PROPOSED SCHEME

Figure 1 shows the overall block diagram of the proposed system. At the encoder side, the input video sequence is first divided into groups of pictures (GOPs). And then, within a GOP, each frame is hierarchically decomposed by the critically sampled discrete wavelet transform (DWT). Furthermore, we adopt the motion-compensation-based approach to decorrelate the video signal along the temporal dimension, which jointly implements the temporal WT and MC. To avoid the shift-variant problem existing inherently in critically sampled discrete wavelet transform and to achieve high coding performance, motion estimation and

motion compensation are performed in the overcomplete wavelet domain, which is so called the overcomplete motion-compensated temporal filtering (OMCTF). After the overcomplete motion-compensated temporal filtering operation, the generated texture information and motion vector information are distributed into each description by a multiple-description algorithm. In this work, four descriptions are generated from the encoded bitstreams. For the coding of residual wavelet coefficients in each description, we develop a modified 3D-SPIHT algorithm [9], which is not a block-based coding algorithm and is suitable for our MDC approach. As discussed before, in order to make each description still resilient to channel errors, we partition wavelet coefficients into a number of independent spatial orientation trees, which is borrowed from Creusere's work [10], and each spatial orientation tree can be encoded and decoded independently. Thus, an error in the bitstream belonging to one tree does not affect the others. However, such an algorithm is still sensitive to bit errors, because each spatial orientation tree produces variable-length bitstreams. A single bit error may lead to loss of synchronization for each spatial orientation tree. In order to generate entropy-coded video bitstreams that are self-synchronized for robust decoding at the receiving end, we also introduce EREC to generate fiexed-length bitstreams, and then each description is encoded by channel coding. In this scheme, we use low-density parity-check codes (LDPCs). After channel coding, each antenna employs an OFDM modulator with N subcarriers. As mentioned above, hybrid space-time coding structure using STBC combined with BLAST is then applied according to different priority levels of the bitstreams. In this scheme, we allocate different numbers of the subcarriers to different space-time coders according to the proportion between important bits and less important bits. In the decoding stage, the signal at the receiver can be separated and recovered from different subcarriers. In the following sections, we will discuss more technical details for this scheme.

2.1. Multiple-description and error-resilient video coding

2.1.1. Multiple-description coding based on OMCTF

Recently, motion-compensated temporal filtering (MCTF) has been proposed to replaced motion-compensated prediction (MCP) for developing scalable video coder. Compared to MCP, MCTF is an open-loop structure, which does not employ recursive structure for ME/MC. Motion-compensated temporal filtering is firstly proposed by Ohm and improved by Choi and Woods. Motion-compensated temporal filtering is able to not only remove the redundancy between interframes using motion estimation and motion compensation, but it also removes the temporal redundancy by a pyramidal temporal decomposition structure. MCTF has two forms, which are denoted as spatial-domain MCTF and overcomplete MCTF (OMCTF) or in-band MCTF [11]. The difference between them is that MCTF is performed in the original frames in spatial-domain MCTF, otherwise, MCTF is performed in the wavelet domain in overcomplete MCTF. However, in the architecture of overcomplete MCTF, due to the problem caused by shift-variant property, important information of motion accuracy would be lost if motion estimation and motion compensation (ME/MC) are performed only in critically sampled wavelet domain. Therefore, to achieve high coding efficiency in the overcomplete MCTF, ME/MC is performed in overcomplete wavelet domain. Moreover, overcomplete motion-compensated temporal filtering provides flexible scalable feature for scalable video coding. Our MDC scheme is based on overcomplete MCTF wavelet video coding.

Many MD approaches introduce the correlation between descriptions by inserting some amount of redundancy .However, a class of MD approaches assumes that there is a strong correlation between adjacent coefficients so that the value of given coefficients can be reasonably predicted by the value of their neighbors. Several such MD schemes have been proposed for image and video applications [12–14]. Tanabe and Farvardin have shown that wavelet coefficients in the lowest frequency subband have similar spatial correlation with that of the original image [15]. This means that this characteristic can be utilized to construct an MD wavelet-based video coder. By exploiting the nature correlation of the adjacent coefficients in the wavelet domain, it is possible to develop a multiple-description algorithm under the framework of overcomplete motion-compensated temporal filtering. As mentioned before, overcomplete motion-compensated temporal filtering jointly implements the spatial wavelet transform (WT) and ME/MC in the temporal direction to decorrelate the video signal. Thus, the frames within a GOP after performing overcomplete motion-compensated temporal filtering, the correlation of the signal will be further decorrelated except the lowest frequency subband in the temporal lowest frequency frame. Here we denote the temporal lowest frequency frame as LLL frame. Therefore, what should be carefully considered is how to make use of the correlation of the coefficients in the lowest subband frequency in the LLL frame. In order to exploit this correlation, adjacent wavelet coefficients should be dispersively distributed into different descriptions. An example for this method is shown in Figure 2. As depicted in Figure 2, different circles represent wavelet coefficients in different descriptions. In the example for four descriptions, the wavelet coefficients in the lowest frequency subband in the LLL frame are equally divided into different descriptions. The other wavelet coefficients in the high subband in the LLL frame will be partitioned into four descriptions as shown in Figure 2. The partitioning of the other wavelet coefficients in the other frames within a GOP will be the same as that of LLL frame.

Another key problem in such an MD scheme is how to partition the wavelet coefficients into different descriptions. As it is known, there is a direct relationship between wavelet coefficients in the different subbands, which is called parent-child orientation tree structure. Several wavelet-based image coding algorithms based on this idea have been developed, which have demonstrated excellent coding efficiency [16, 17]. In this scheme, we use the 3D-EZW tree structure

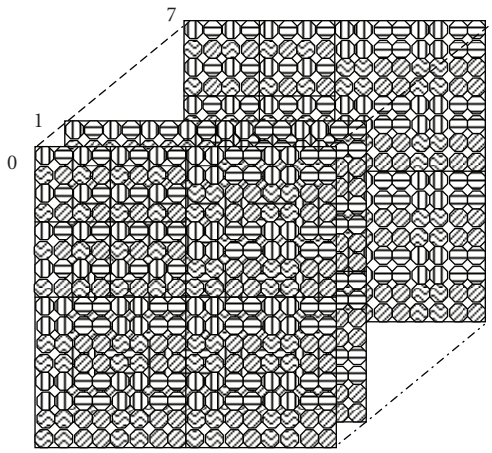


Figure 2: An example of partitioning strategy.

[18]. Figure 2 also illustrates the tree structure. Meanwhile, in order to improve the performance of error resilience, we should further divide each description into several packets. In this scheme, we packetize each orientation tree into one packet. For robustness, we also expect that each packet could be encoded and decoded independently. Thus, the decoding failure in one packet will not affect the others. This is a welcome feature for robust image and video transmission over error-prone environment.

2.1.2. EREC and error concealment

When compressed video is transmitted over wireless channel, a single bit error may cause the error to propagate not only to frames along the temporal direction, but also within an individual frame. The problem that causes such error propagation is that VLC coding scheme is usually adopted for coding the texture information after ME/MC. As mentioned before, each orientation tree is encoded and decoded independently. To balance the quality for each orientation tree, we encode each orientation tree to the same bitplane. A single bit error may cause the loss of synchronization among these orientation trees. To overcome this drawback, we need to design a scheme that is able to limit the bit error within each orientation tree.

Error-resilient entropy coding (EREC) is an effective means to recover lost or erroneous information to enhance error resilience for coding variable-length blocks of data. The scheme is originally proposed by Redmill and Kingsbury to handle the sequential transmission of DCT coded data blocks over noisy channels. The key of the EREC is to reorganize the variable-length data blocks into fixed-length slots with negligibly increased data size. Figure 3 shows EREC bit reorganization algorithm. It has been shown that the EREC can significantly reduce the channel error propagation effects and that remaining channel error propagation is most likely to affect data from the end of longer blocks [8]. The characteristic of EREC decoding is very suitable for some embedded coding methods, such as 3D-SPIHT algorithm. As we

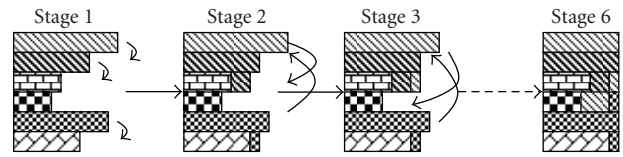


Figure 3: EREC bit reorganization algorithm.

know, 3D-SPIHT encodes the wavelet coefficients from high bitplane to low bit-plane by threshold. This suggests that the encoded bitstream contains more important information in the beginning of the bitstream. Therefore, to further enhance the error resilience of the proposed scheme, we apply EREC to the symbols coded by 3D-SPIHT algorithm to avoid synchronization loss among the orientation trees.

As we know, EREC decoding needs a priori known information about the position where it should stop. There are usually three methods to attain this purpose: adding special bitstream end symbols, coding a length before each bittream, or coding bitstream to a known bitplane. The first two methods will reduce the coding efficiency due to adding a certain overhead. In this work, we adopt the third approach, that is, encoding each orientation tree to the same bitplane. Thus, each slot will have an explicit ending position. This method also keeps the balanced quality for each orientation tree because each orientation tree stops decoding at the same threshold value. Another key problem in the EREC algorithm is how to detect error during EREC decoding. In [19], Cao and Chen proposed a method for image wavelet-based coding. They added one bit for the first several bits in each slot data for the purpose of error detection in the EREC encoding stage. This is because the first several bits coded by SPIHT have higher energy than other bits. However, this scheme does not consider that each wavelet tree may not have the same energy distribution. Therefore, if one bit is added at the fixed position in the first several bits for each wavelet tree, some slots with errors that still contain high energy may miss the error detection. In this work, we add one bit for parity check at the tail of each bitplane for the first three bitplanes, respectively. Thus, if we detect any error in these parity-check bits in the slot, we will regard the slot corrupted. If some slots are detected with errors, they will not take part in the decoding stage.

Meanwhile, additional advantage of the proposed scheme using wavelet tree coding combined with EREC in MD scheme is that simple error concealment methods can be employed to estimate the lost information. Since MDC strategy assumes that not all bitstreams in the different descriptions experience failures simultaneously, thus, if we detect the errors occurring in one slot in some description, most likely we can apply error concealment method to estimate the lost information from the neighboring uncorrupted slots in the other descriptions. As mentioned before, the advantage of the MDC algorithm proposed in this scheme is that there exists strong correlation between the adjacent wavelet coefficients in the lowest frequency subband in the LLL frame. Therefore, the lost wavelet coefficients can be estimated by

calculating the average of their neighboring coefficients by decoding the corresponding slots. In this scheme, error concealment is only carried out in the lowest subband in the LLL frame.

2.2. Unequal error protection using BLAST and STBC

In this paper, we consider a MIMO-OFDM system with M_t ($M_t = 4$) transmit and M_r ($M_r = 4$) receiver antennas for robust video transmission. Each antenna employs an OFDM modulator with N subcarriers. Transmitting signals of different subcarriers will be transmitted simultaneously over all transmit antennas. As mentioned before, the coded source bits transmitted over each antenna also contain unequally important information according to their contribution to the decoding process. Thus, data partitioning can be used in this scheme. As we know, data partition is a very effective tool for the transmission of video over an error environment. Data partitioning divides a coded signal bitstream into two components. In this scheme, each description is further divided into two partitions: motion vectors and GOP header information, and texture information. Due to their different priority levels, we can carry out unequal error protection strategy. Motivated by this, we propose a hybrid space-time coding structure to transmit multiple bitstreams generated by multiple-description coding scheme as discussed in the previous section. In this research, we combine BLAST and STBC in this multiple antennas system to obtain unequal error protection for robust video transmission. By transmitting the most critical information using MIMO diversity scheme, the average reconstructed quality at the receiver will not degrade largely. Meanwhile, by sending the texture information at each antenna simultaneously using MIMO multiplexing scheme, the transmission data rate will be enhanced. The illustration of the proposed UEP scheme is shown in Figure 4.

As we have discussed, the coded source bits might be divided into two parts according to their contribution to the decoding process, namely TINF, which stands for the texture information, and MIB, which presents the more important bits including motion vectors and GOP header information. Considering the independent channels generated by BLAST, it is intuitively suitable for the transmission of TINF bits. Additionally, the transmission efficiency will also be increased greatly. However, MIB is a crucial component in a video codec. For example, even one bit error in the MIB bits may cause a decoding collapse or cause serious error propagation within a GOP and greatly degraded video quality. Compared with BLAST, STBC achieves full diversity gain. It performs much better than BLAST in terms of BER, however, with much lower transmission efficiency. Consequently, to guarantee tradeoff between reconstructed video quality and transmission efficiency, we apply different space-time coding methods in the scheme, as shown in Figure 4. In this scheme, we employ four transmit antennas to match the four MDC bitstreams from the video encoder. Assuming the space-time code rate is R, MIB has B_{MIB} bits and TINF has B_{TINF} bits, we will have $P = \lfloor B_{\mathrm{MIB}}N/(B_{\mathrm{TINF}}R + B_{\mathrm{MIB}}) \rfloor$ subcarriers to transmit MIB. These subcarriers are denoted as $\{f_1 \;\; f_2 \;\; \cdots \;\; f_P\}$.

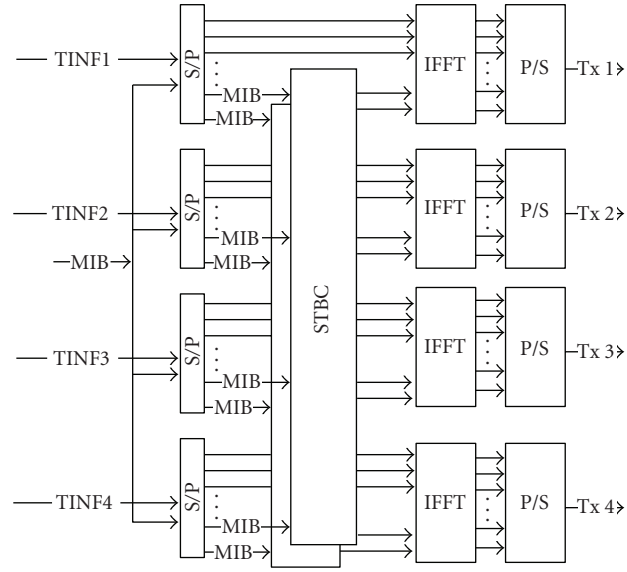


Figure 4: The block diagram of the proposed UEP scheme.

2.2.1. BLAST

V-BLAST structure is employed in this scheme. At the transmitter, each MDC stream is assigned to one antenna which holds an independent transmit channel as mentioned before. At the receiver, the frequency-domain signals at subcarrier $\{f_{P+1} \;\; f_{P+2} \;\; \cdots \;\; f_N\}$ will be sent to a simple ZF detector [10] to recover the transmit symbols. Note that the optimal detect algorithms might bring better performance, but the process complexity and delay would be increased together. An efficient way to solve this problem is to import channel codes. In this scheme, the LDPC code is employed [20]. As a result, the transmission performance can be greatly increased.

2.2.2. STBC

For four-transmit-antenna system, assuming that the channel is stable during four OFDM symbol periods, we construct a 3/4 rate STBC code:

$$C = \begin{pmatrix} c_1 & -c_2^* & \frac{1}{\sqrt{2}}c_3^* & \frac{1}{\sqrt{2}}c_3^* \\ c_2 & c_1^* & \frac{1}{\sqrt{2}}c_3^* & -\frac{1}{\sqrt{2}}c_3^* \\ \frac{1}{\sqrt{2}}c_3 & \frac{1}{\sqrt{2}}c_3 & -\frac{1}{2}(c_1+c_1^*-c_2+c_2^*) & \frac{1}{2}(c_1-c_1^*+c_2+c_2^*) \\ \frac{1}{\sqrt{2}}c_3 & \frac{1}{\sqrt{2}}c_3 & \frac{1}{2}(c_1-c_1^*-c_2+c_2^*) & -\frac{1}{2}(c_1+c_1^*+c_2-c_2^*) \end{pmatrix}. \tag{1}$$

It is obvious that three complex symbols per subcarrier are transmitted during four OFDM symbol periods. At the receiver, in order to obtain an orthogonal channel matrix, we define the received signal at the jth receiving antenna, nth

subcarrier as

$$Y_j(n) = \left[y_j^1(n) \;\; y_j^2(n) \;\; y_j^3(n) \;\; y_j^4(n) \;\; (y_j^1(n))^* \;\; (y_j^2(n))^* \;\; (y_j^3(n))^* \;\; (y_j^4(n))^* \right]. \quad (2)$$

Here $(\bullet)^*$ denotes conjugate operation and $y_j^t(n)$ presents the received signal at the tth OFDM symbol period.

Then the transmit symbol can be recovered as

$$\hat{c}(n) = \frac{1}{M_r * \sum_{i=1}^{M_t} ||h_{i,j}||^2} \sum_{j=1}^{M_r} H_j^H(n) * Y_j(n), \quad (3)$$

where $\hat{c}(n) = [\hat{c}_1^*(n) \;\; \hat{c}_2^*(n) \;\; \hat{c}_3^*(n) \;\; \hat{c}_1(n) \;\; \hat{c}_2(n) \;\; \hat{c}_3(n)]^T$, and $H_j(n)$ is the orthogonal channel matrix at the subcarrier n in the receive antenna j, and $(\bullet)^H$ denotes the conjugate transpose operation. Finally, the video-coded bit sequence will be ready for decoding after constellation demapping and bits rearrangement.

2.3. Channel model

We consider a frequency-selective Rayleigh fading channel which has L independent delay paths with arbitrary delay power profiles. The baseband equivalent channel can be modeled as

$$h_{i,j}^k = \sum_{l=0}^{L-1} \alpha_{i,j}^k(l)\sigma(t - \tau_l), \quad (4)$$

where $\alpha_{i,j}^k(l)$ is the path gain coefficient of the lth path between transmit antenna i and receiver antenna j at the kth OFDM symbol period, and τ_l presents the lth path delay. The $\alpha_{i,j}^k(l)$ is modeled as zero-mean complex Gaussian random variable with variance $E|\alpha_{i,j}^k(l)|^2 = \sigma_l^2$. The channel coefficients are assumed spatially uncorrelated.

The received signal at nth subcarrier at jth receive antenna during kth OFDM block may be denoted as

$$y_j^k(n) = \sqrt{\rho} \sum_{i=1}^{M_t} x_i^k(n) H_{i,j}^k(n) + \varpi_j^k(n), \quad (5)$$

where ρ is the average signal-to-noise ratio per receiver and $H_{i,j}^k(n) = \sum_{l=0}^{L-1} \alpha_{i,j}^k(l) e^{-j2\pi n \Delta f \tau_l}$ is the subchannel gain. Here $\Delta f = 1/T_s$ is the inter-subchannel space and T_s is the OFDM symbol period. The additive noise $\varpi_j^k(n)$ is modeled as independent complex Gaussian random variable with zero mean and unit variance.

3. SIMULATION RESULTS

In this section, we conduct several simulation experiments to show the performance of the proposed scheme over MIMO channels. The simulations are tested on the standard video sequence "Foreman," whose frames are in QCIF format with only luminance component. In this scheme, four descriptions are generated from the complete bitstream. For spatial decomposition, we apply the CDF 9/7 biorthogonal filter and three levels of decomposition. For overcomplete MCTF, we apply Haar filter and three levels of temporal decomposition. We adopt 3D-SPIHT core algorithm without arithmetic coding. Each tree can be encoded and decoded independently. The source coding rate is at 0.93 bpp.

We build a MIMO-OFDM simulation system as introduced in Section 2. The channel code used is a 1/2 rate irregular LDPC code with a length of 6096 bits. The OFDM symbols are designed following the IEEE 802.11a standard. Specifically, each OFDM symbol occupies 64 subcarriers, in which 52 subcarriers may be used for transmitting QPSK data symbols. The frequency-selective multipath channel model is referred to as COST207 indoor model [21]. Finally, known timing and zero carrier frequency offset (CFO) are given throughout simulations. Detail simulation parameters are listed in Table 1.

3.1. PSNR performance comparison

In this simulation, we compare the performance of three schemes: the proposed UEP scheme, EEP scheme using only V-BLAST algorithm, and STBC scheme. PSNR performance is evaluated through several experiments. Figure 5 shows the PSNR performance of the proposed scheme compared with other schemes. In the MIMO-OFDM system without channel coding, we can achieve a great improvement. Moreover, by importing LDPC code, the SNR requirement is greatly reduced and the UEP scheme still obtains a few dB PSNR gains. It can be seen that the proposed scheme outperforms the EEP scheme. From the figure, we also observe that STBC scheme has the best performance among the three schemes. However, the transmission efficiency will drop with noticeable loss. As a result, the proposed UEP scheme can be an alternative to achieve the tradeoff between reconstructed quality and transmission efficiency for video transmission. The main advantage of the proposed scheme is that it is simpler than existing schemes using power allocation or adaptive modulation algorithm to combat with multipath fading in MIMO transmission system. Moreover, the proposed UEP scheme is more suitable for multiple antennas system than traditional UEP schemes based on channel coding.

3.2. The performance comparison for error concealment

We also conduct experiments to show the performance of the proposed scheme with and without error concealment over MIMO channels. Simulations are performed with channel coding and without channel coding, respectively. As discussed in the previous section, we apply the error detection strategy so as to facilitate the corresponding error concealment scheme. Error concealment scheme is only applied to the LLL frame. If we find some wavelet trees corrupted, we can compensate these "bad" wavelet coefficients by calculating the average of their surrounding wavelets coefficients in the lowest subband in the LLL frames. The other wavelet coefficients in the corrupted wavelet tree will be set to zero. Figure 6 shows the comparison of average PSNRs of

Table 1: Simulation parameters.

Antenna (Tx × Rx)	4 × 4
Number of subcarriers (data/all)	52/64
Guard interval (chips)	16
Modulation	QPSK
Bandwidth (MHz)	20
Multipath latency (ns)	0, 100, 200, 300, 500, 700
Multipath decay (dB)	0, −3.6, −7.2, −10.8, −18, −25.2

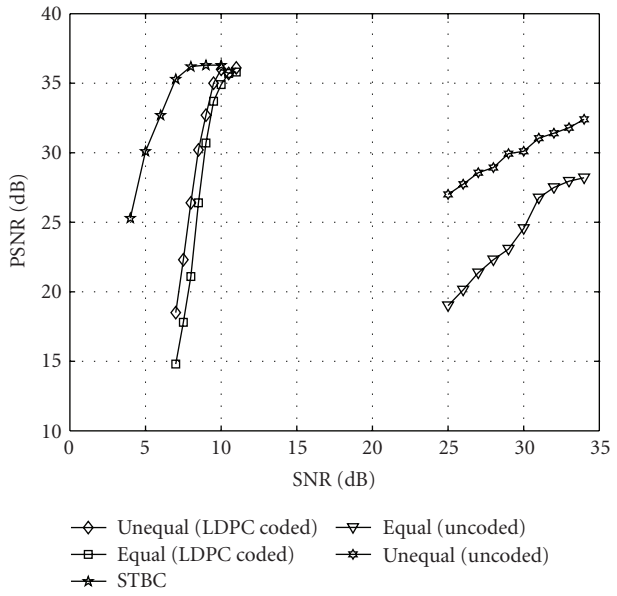


Figure 5: PSNR versus SNR comparison for EEP and UEP schemes.

the reconstructed frames at different SNRs with and without error concealment. Figure 7 shows both visual quality and PSNR results of some sample frames before and after error concealment. From these figures, we can demonstrate that visual quality of the reconstructed frames have been gradually improved as SNR increases, and the average PSNRs of the reconstructed frames with error concealment are higher than those without error concealment. These results show the importance of error concealment.

4. CONCLUSION

In this paper, we have developed a new scheme for robust video transmission over MIMO-OFDM system using MDC scheme to improve the robustness of signal source and adopting hybrid space-time coding structure to obtain unequal error protection during transmission. Experimental results have demonstrated that the proposed scheme can be an excellent alternative to achieve the tradeoff between received video quality and transmission efficiency. We believe that such an integrated approach will provide additional dimensions for optimal design of MIMO-OFDM wireless video communication systems.

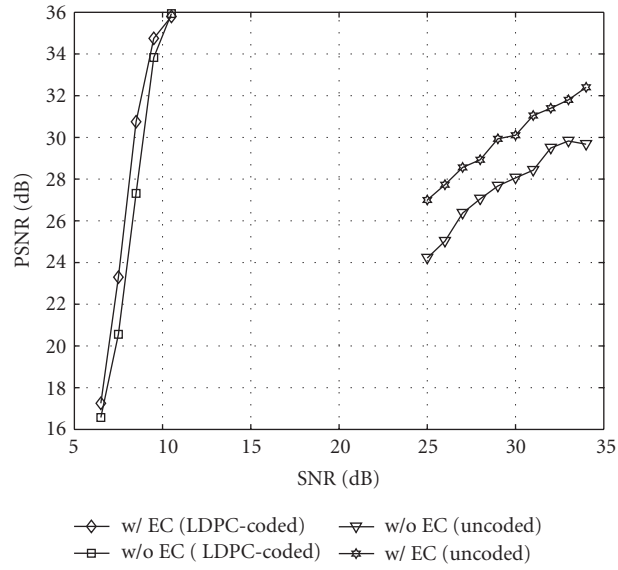


Figure 6: PSNR versus SNR comparison for with and without error concealment schemes.

PSNR = 28.132 (before EC) PSNR = 30.927 (after EC)

(a) SNR = 33 dB (uncoded)

PSNR = 27.433 (before EC) PSNR = 30.855 (after EC)

(b) SNR = 8.5 dB (LDPC-coded)

Figure 7: Visual quality and PSNR results of some sample frames before and after error concealment.

ACKNOWLEDGMENTS

This work is supported by China National Natural Science Foundation under Research Grant no. 60328103 and the Fujian Education Department Foundation Grant no. K04004.

REFERENCES

[1] V. Tarokh, H. Jafarkhani, and A. R. Calderbank, "Space-time block codes from orthogonal designs," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1456–1467, 1999.

[2] G. J. Foschini, "Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas," *Bell Labs Technical Journal*, vol. 1, no. 2, pp. 41–59, 1996.

[3] H. Zheng and K. J. R. Liu, "Space-time diversity for multimedia delivery over wireless channels," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS '00)*, vol. 4, pp. 285–288, Geneva, Switzerland, May 2000.

[4] H. Zheng and D. Samardzija, "Performance evaluation of indoor wireless systems using BLAST testbed," in *Proceedings of IEEE Vehicular Technology Conference (VTC '01)*, vol. 2, pp. 905–909, Atlantic City, NJ, USA, May 2001.

[5] Z. Ji, Q. Zhang, W. Zhu, Z. Guo, and J. Lu, "Power-efficient MPEG-4 FGS video transmission over MIMO-OFDM systems," in *Proceedings of IEEE International Conference on Communications (ICC '03)*, vol. 5, pp. 3398–3402, Anchorage, Alaska, USA, May 2003.

[6] Z. Ji, Q. Zhang, W. Zhu, J. Lu, and Y.-Q. Zhang, "Video broadcasting over MIMO-OFDM systems," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS '03)*, vol. 2, pp. 844–847, Bangkok, Thailand, May 2003.

[7] C.-H. Kuo, C.-S. Kim, and C.-C. J. Kuo, "Robust video transmission over wideband wireless channel using space-time coded OFDM systems," in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC '02)*, vol. 2, pp. 931–936, Orlando, Fla, USA, March 2002.

[8] D. W. Redmill and N. G. Kingsbury, "The EREC: an error-resilient technique for coding variable-length blocks of data," *IEEE Transactions on Image Processing*, vol. 5, no. 4, pp. 565–574, 1996.

[9] B.-J. Kim and W. A. Pearlman, "Embedded wavelet video coder using three-dimensional set partitioning in hierarchical trees (SPIHT)," in *Proceedings of Data Compression Conference (DCC '97)*, pp. 251–260, Snowbird, Utah, USA, March 1997.

[10] J. Liu and J. Li, "A simple soft-detector for the BLAST system," in *Proceedings of IEEE Sensor Array and Multichannel Signal Processing Workshop*, pp. 159–163, Rosslyn, Va, USA, August 2002.

[11] Y. Andreopoulos, A. Munteanu, J. Barbarien, M. Van Der Schaar, J. Cornelis, and P. Schelkens, "In-band motion compensated temporal filtering," *Processing: Image Communication*, vol. 19, no. 7, pp. 653–673, 2004, special issue on Subband/Wavelet Interframe Video Coding.

[12] N. Franchi, M. Fumagalli, R. Lancini, and S. Tubaro, "Multiple description video coding for scalable and robust transmission over IP," in *Proceedings of the Packet Video Workshop (PV '03)*, Nantes, France, April 2003.

[13] I. V. Bajić and J. W. Woods, "Domain-based multiple description coding of images and video," in *Visual Communications and Image Processing*, vol. 4671 of *Proceedings of SPIE*, pp. 124–135, San Jose, Calif, USA, January 2002.

[14] A. C. Ashwin, K. R. Ramakrishnan, and S. H. Srinivasan, "Wavelet domain residual redundancy-based descriptions," *Signal Processing: Image Communication*, vol. 18, no. 7, pp. 549–560, 2003.

[15] N. Tanabe and N. Farvardin, "Subband image coding using entropy-coded quantization over noisy channels," *IEEE Journal of Selected Areas in Communications*, vol. 10, no. 5, pp. 926–943, 1992.

[16] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3445–3462, 1993.

[17] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 6, no. 3, pp. 243–250, 1996.

[18] Y. Chen and W. A. Pearlman, "Three-dimensional subband coding of video using the zero-tree method," in *Visual Communications and Image Processing*, vol. 2727 of *Proceedings of SPIE*, pp. 1302–1312, Orlando, Fla, USA, March 1996.

[19] L. Cao and C. W. Chen, "Robust image transmission based on wavelet tree coding, error resilient entropy coding, and error concealment," *Journal of Electronic Imaging*, vol. 13, no. 3, pp. 646–653, 2004.

[20] C. Ru, L. Yin, and J. Lu, "An LDPC coded MIMO-OFDM system with simple detection and channel estimation scheme," in *Proceedings of the 6th IEEE Circuits and Systems Symposium on Emerging Technologies: Frontiers of Mobile and Wireless Communication*, vol. 2, pp. 693–696, Shanghai, China, May-June 2004.

[21] "Digital Land Mobile Radio Communications - COST 207," Commission of the European Communities, Final Report, Marzo 1984–Settembre 1988, Office for Official Publications of the European Communities, Luxembourg, 1989.