# MPEG-TFRCP: Video Transfer with TCP-friendly Rate Control Protocol

Masaki Miyabayashi, Naoki Wakamiya, Masayuki Murata, Hideo Miyahara
Department of Informatics and Mathematical Science,
Graduate School of Engineering Science, Osaka University,
1-3 Machikaneyama, Toyonaka, Osaka 560-8531, Japan
Tel: +81-6-6850-6588, Fax: +81-6-6850-6589
E-mail: {miyabays, wakamiya, murata, miyahara}@ics.es.osaka-u.ac.jp

*Abstract*— **As the use of real-time multimedia applications increases, bandwidth available to TCP connections is oppressed by "greedy" UDP traffic and their performance extremely deteriorates. In order that both TCP and UDP sessions fairly co-exist in the Internet, UDP sessions should properly react against congestion as TCP. In this work, we implement a "TCP-friendly" rate control mechanism suitable to video applications and investigate its applicability to a real system through observation of the video quality at the receiver. It is shown through our experimental system that we can achieve high-quality and stable video transfer while fairly sharing the network bandwidth with TCP by applying our rate control at a control interval of 16 or 32 times as long as RTT.**

## I. Introduction

Since the current Internet does not provide QoS (Quality of Service) guarantee mechanisms, each application chooses the preferable transport protocol to achieve the required performance. As the use of real-time multimedia applications increases, a considerable amount of "greedy" UDP traffic would dominate network bandwidth. As a result, the available bandwidth to TCP connections is oppressed and their performance extremely deteriorates.

In order that both TCP and UDP sessions fairly co-exist in the Internet, it is meaningful to consider the fairness among protocols. In recent years, several researches have been focused on the investigation of the "TCP-friendly" rate control [1-10]. "TCP-friendly" is defined as "a non-TCP connection should receive the same share of bandwidth as a TCP connection if they traverse the same path" [7]. The TCP-friendly system regulates its data sending rate according to the network condition, typically expressed in terms of the round-trip-time (RTT) and the packet loss probability (denoted as $p$), to achieve the same throughput that a TCP connection would acquire on the same path.

Control mechanisms proposed in [1-8] achieve the TCP-friendliness according to its definition, but those require carefully chosen parameters to achieve the fairness among TCP and UDP connections. Especially when those mechanisms are to be applied to the real-time video transfer, characteristics of video applications should be taken into account to provide the high quality video transfer while satisfying the TCP-friendliness in the real-time multimedia applications. In addition, as a result of rate control, the video sending rate inherently fluctuates and the resultant video quality frequently changes in the case of pseudo-TCP mechanisms which employ the AIMD control to imitate the TCP's behavior, [1-3]. On the other hand, although the stable video presentation can be accomplished with equation-based ones [4-8], they cannot adequately adapt to changes of network condition.

We have been devoted into investigation of the applicability of TCP-friendly rate control to real-time MPEG-2 video communications, and proposed an effective control mechanism, called MPEG-TFRCP in [9, 10]. Our mechanism employs the equation proposed in [6] and thus can be categorized into the equation-based approach. However, our approach dynamically adjusts the video sending rate according to the network condition. The effectiveness was evaluated through simulation experiments and it was shown that the high quality and TCP-friendly real-time video transfer can be accomplished. However, in the experiments, the ideal network system environment was assumed. That is, we did not take into account several factors which may affect the effectiveness of rate control. Those include the fluctuation of control intervals or the quality degradation during playback of a video sequence.

In this paper, we implement several TCP-friendly rate control protocols on an actual video transfer system. Our implemented mechanisms are based on the MPEG-TFRCP. First we demonstrate the applicability of MPEG-TFRCP to a real system through evaluation of the perceived video quality and observation of the traffic on the link. Then we consider the improved versions of MPEG-TFRCP by carefully examining the rate and control interval determination methods.

The paper is organized as follows. In Section II, we introduce our MPEG-TFRCP and explain how it is implemented. Then we compare several TCP-friendly rate control mechanisms on our implemented video applications in Section III. Comparisons are performed in terms of the rate variation, the packet loss variation and the perceived video quality. Finally, we summarize our paper and outline our future work in Section IV.

## II. Implementation of MPEG-TFRCP

In this section, we introduce our MPEG-TFRCP (TCP-friendly rate control protocol for MPEG-2 video transfer) [9, 10]. In order to transfer a video sequence with high and stable quality while fairly sharing the network bandwidth with TCP, our MPEG-TFRCP behaves as illustrated in Fig. 1; at the end of the control interval $i-1$ whose duration is $I_{i-1}$, the sender estimates the network condition and the throughput of a TCP session, from information gathered within the interval, and finally determines and regulates its sending rate $r_i$ for the next interval $i$.

### A. MPEG-TFRCP mechanism

In MPEG-TFRCP, a sender divides the MPEG-2 video data into packets whose header contains the timestamp and the sequence number as in RTP (Real-time Transport Protocol) and

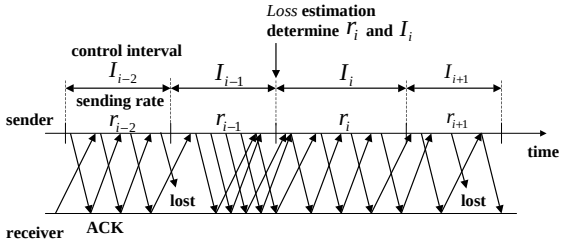


Fig. 1. TFRCP video transfer

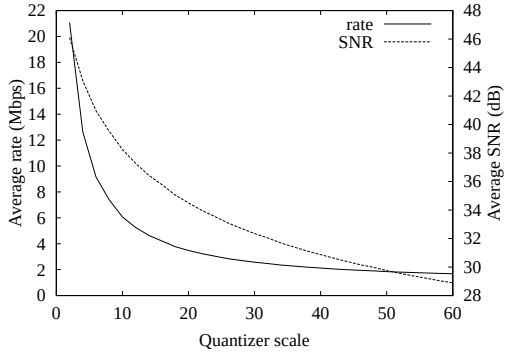


Fig. 2. Relationship among quantizer scale, video rate and video quality

sends them to a receiver. On receiving the packet, the receiver sends back an acknowledgement to inform the sender of a successful reception. Then, the sender obtains RTT, RTO (Retransmission Time Out) and the packet loss probability $p$.

At the end of the interval $i-1$, the sender determines the sending rate $r_i$ for the next interval $i$. If the packet loss probability $p$ is non-zero, the sender estimates the throughput of a TCP connection which traverses the same path from the equation proposed in [6]. Then $r_i$ is set at the estimated TCP throughput expecting the fair share of network bandwidth. When the network is under-utilized and there is no packet loss, the sender only doubles the rate because the estimator is not applicable. The algorithm is summarized as

$$r_i = \begin{cases} \dfrac{MTU}{RTT\sqrt{\frac{2p}{3}} + T_0 \min(1, 3\sqrt{\frac{3p}{8}})p(1+32p^2)}, & \text{if } p > 0 \quad (1) \\ 2 \times r_{i-1}, & \text{if } p = 0 \quad (2) \end{cases}$$

where *MTU* stands for the maximum transfer unit size, $p$ is the packet loss probability, *RTT* and $T_0$ are for the round trip time and the retransmission timeout, respectively.

The duration of each interval is 32 times as long as RTT in the MPEG-TFRCP. Considering the MPEG-2 video coding algorithm, the duration is further rounded to the multiple of GoP (Group of Picture) time, which is given as the number of pictures in a GoP ($N$) divided by the frame rate. We call this strategy as 32-RTT. That is, the control interval of 32-RTT is given by

$$I_i = \lceil \frac{32 \times RTT}{\text{GoPtime}} \rceil \times \text{GoPtime}.$$

The sender adjusts the video coding rate to the determined target rate $r_i$. In this paper, we consider the MPEG-2 VBR coding algorithm. In the MPEG-2 algorithm, each captured picture is first Discrete-Cosine-Transformed and then each

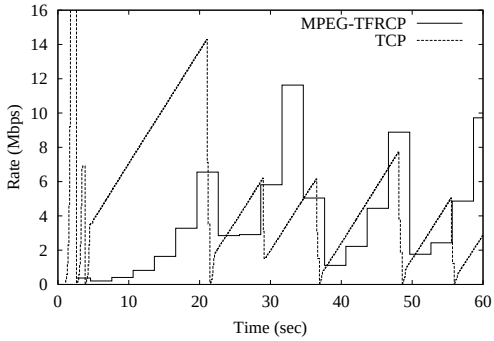


Fig. 3. Simulation result of MPEG-TFRCP

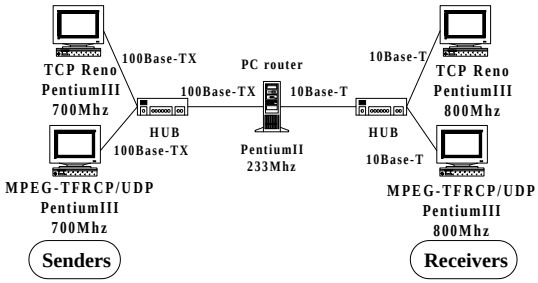


Fig. 4. System configuration

DCT coefficient is quantized with specified quantizer scale. Thus, the coded video quality and the amount of data can be regulated by choosing the appropriate quantizer scale. An example of the relationship among the quantizer scale, the resultant average video rate and the video quality in terms of SNR (Signal to Noise Ratio) [11] is depicted in Fig. 2. The video is 640×486 pixels large and its GoP structure is $N = 30$ and $M = 1$ (IPPP$\cdots$). By applying the method proposed in [11], we can easily determine the appropriate quantizer scale to adjust the video rate to the target rate.

In Fig. 3, we show an example of simulation results with our MPEG-TFRCP. The rate variation of a TCP session and that of the determined target rate of a MPEG-TFRCP session are depicted. As shown in the figure, the MPEG-TFRCP behaves similarly to TCP and they fairly share the bandwidth.

### *B. Implemented system*

To investigate the applicability of the MPEG-TFRCP to the actual system, we built the small-scale 10Base-T network as illustrated in Fig. 4. The system consists of four personal computers (RedHat Linux kernel-2.2.14), two shared HUBs and one PC router. Packet size is 1000 Bytes and identical among connections.

The implemented MPEG-TFRCP system is depicted in Fig. 5. The MPEG-2 encoder applies the VBR coding algorithm to original video data where the quantization scale is specified by the QoS manager. Then, the transmitter divides the video data into RTP packets and sends them to the receiver over the UDP session. The MPEG-2 video data are reconstructed from received packets at the receiver, then decoded and displayed on the monitor.

While sending video data, the sender also transmits RTCP control packets to obtain the feedback information from the receiver, which is indispensable for the TCP throughput es-

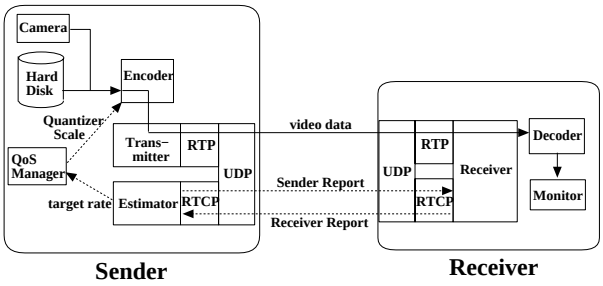


Fig. 5. MPEG-TFRCP sender & receiver

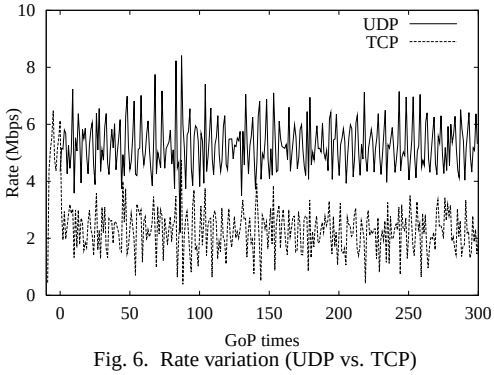


Fig. 6. Rate variation (UDP vs. TCP)

timation. The estimator transmits RTCP `SenderReport` packets to the receiver at the regular interval of, in our experiments, every five frames (5/29.97=0.167 sec). On reception of the control packet, the receiver sends back a RTCP `ReceiverReport` packet which contains the information necessary for the sender to estimate the network condition. Those are the expected number of packets sent from the server between two RTCP packets and the number of packets received. With which the sender can easily derive the packet loss probability. The sender obtains the observed RTT value by subtracting the timestamp of the `SenderReport` from the reception time of the `ReceiverReport`. Then, the estimated RTT is derived by applying the smoothing algorithm of Jacobson's to the observed RTTs.

## III. Evaluation of MPEG-TFRCP

In this section, we evaluate the practicality of our MPEG-TFRCP by comparing some variants of MPEG-TFRCP for the MPEG video transfer system implemented on the small scaled network. The comparison is performed in terms of the rate variation of TCP and MPEG-TFRCP sessions observed on the link using `tcpdump`, the packet loss probability and the perceived video quality quantified by MOS (Mean Opinion Score) evaluation.

In the case where both TCP and UDP sessions transfer the same video stream in our system, the resultant rate variations measured at the receiver sides are shown in Fig. 6. Those two sessions transfer the identical video stream of the average rate 5.25 Mbps (See Fig. 2. Quantizer scale=12, 37.28 dB). The experiment time on an x-axis is expressed in a unit of GoP time, i.e. GoP size/frame rate=30/29.97=1.001 sec, and zero corresponds to the start of the UDP session. In this experimental environment, we observe that average RTT is about 30 msec. It is obvious that the throughput of TCP drastically deteriorates as the UDP session starts sending the video data. On the other hand, the UDP session occupies much portion of the link bandwidth and freely transfers the video data at the desirable rate. The averaged throughput of TCP is 2.27 Mbps and that of UDP is 5.25 Mbps. Then the perceived video quality in terms of MOS is high, 4.25.

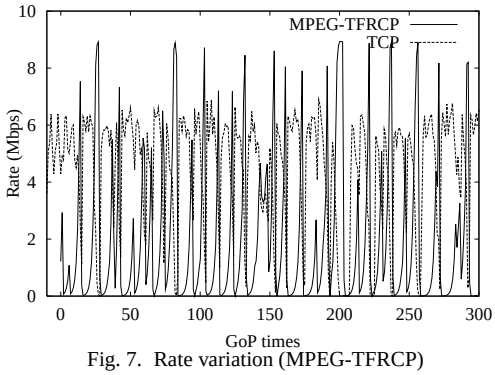


Fig. 7. Rate variation (MPEG-TFRCP)

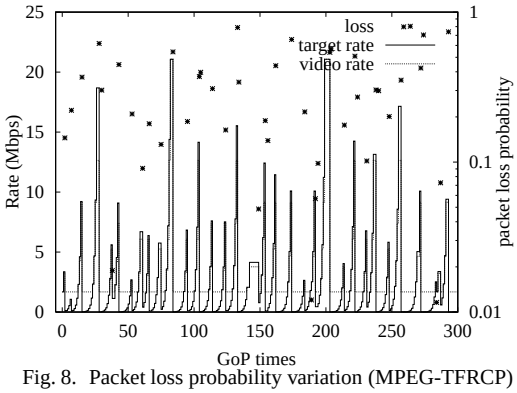


Fig. 8. Packet loss probability variation (MPEG-TFRCP)

### A. Original MPEG-TFRCP

First, we investigate the practicality of our MPEG-TFRCP originally proposed in [9, 10] in the actual system environment. Figure 7 shows the experimental result where TCP and MPEG-TFRCP sessions compete for the bandwidth. It can be observed that the MPEG-TFRCP connection regulates its data sending rate during the session and the degradation of TCP performance becomes smaller than when we use UDP. However, the obtained result seems quite different from the simulation result in Fig. 3. Frequent and considerable fluctuation of MPEG-TFRCP rate is due to the high packet loss probability caused by the aggressive rate increase. Fig. 8 depicts the packet loss probability $p$ ("loss"), the target rate $r_i$ ("target rate") determined by (1) and (2), and the average video rate selected for transmission ("video rate"). The MPEG-TFRCP sender doubles its data sending rate during a loss-free period. As it encounters packet losses, it suddenly shrinks the sending rate because the loss probability is high due to the network congestion caused by itself. Although not shown in figures, the perceived video quality also varies as the video rate changes. Especially when congestion occurs, the high packet loss probability leads to decreased target rate far below the minimum video rate, and no picture can be displayed at the receiver. Here we should note that the MPEG-TFRCP sender first sends as much data as possible, then stops video transmission when

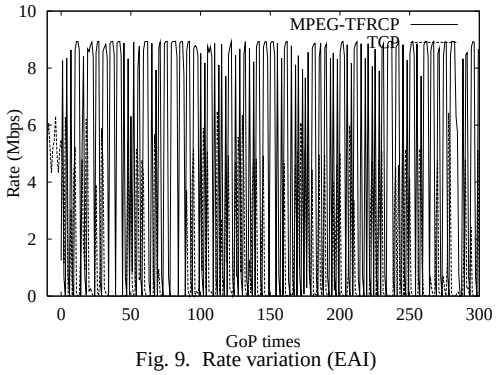

Fig. 9. Rate variation (EAI)

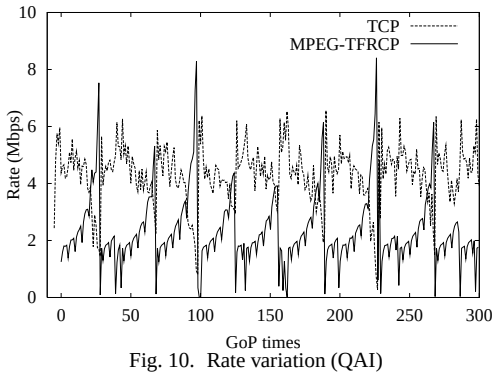

Fig. 10. Rate variation (QAI)

the "video rate" is higher than "target rate". For those reasons, the subjective video quality of the original MPEG-TFRCP in terms of MOS is only 1.25.

### *B. Variants for rate determination*

In this subsection, considering sudden and aggressive rate increase causes network congestion as shown in Subsection III-A, we investigate into variants in rate determination algorithms which are non-aggressive.

The previous work [10] proposes an alternative for (2), which imitates TCP's window-based flow control.

$$r_i = r_{i-1} + \frac{MTU \times I_{i-1}}{RTT^2}, \qquad \text{if } p = 0 \tag{3}$$

With this new algorithm, the additive rate increase as in TCP's congestion avoidance phase can be performed in our MPEG-TFRCP. We call this algorithm as EAI (Equation-based Additive Increase).

The experimental result of EAI MPEG-TFRCP algorithm is depicted in Fig. 9. As shown in this figure, the sending rate considerably oscillates, because the rate increase is unexpectedly aggressive. It is sometimes more aggressive than the original algorithm due to small RTT values. As in (3), RTT dominates the degree of rate increase. If the observed RTT is stable and large enough as in the simulation environment in [10], the EAI algorithm contributes to smoother rate variation. However, in the actual system where RTT is relatively small and often changes, the EAI algorithm provides unpreferable results.

An alternative algorithm for an additive rate increase is Quantizer-scale-based Additive Increase (QAI). The QAI is based on the MPEG-2 coding algorithm. It increases the sending rate with regard to, not $r_i$, but the quantizer scale. It is initially set at 60, with which the perceived video quality is worst and the video rate is lowest (Fig. 2). Then, the quantizer scale is decreased by two as long as no loss is observed during each control interval.

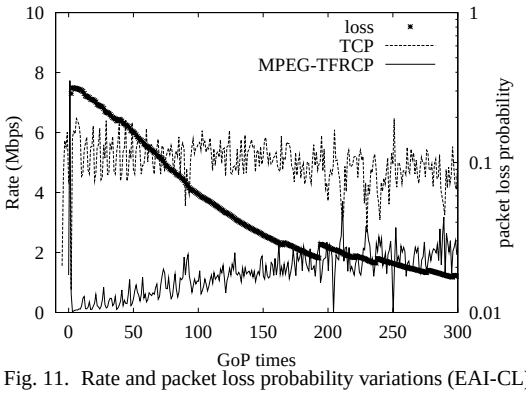

Fig. 11. Rate and packet loss probability variations (EAI-CL)

We show the experimental result using QAI MPEG-TFRCP algorithm in Fig. 10. It can be observed that both the degradation of TCP performance and the rate variation become smaller than that of the original MPEG-TFRCP. However, the average throughput of TCP becomes 4.29 Mbps, whereas that of MPEG-TFRCP is 2.34 Mbps. That is, this method is far from TCP-friendly. The subjective video quality of QAI is 2.50 and much smaller than that of UDP, 4.25. We found that QAI MPEG-TFRCP does not attain high-quality and TCP-friendly video transfer.

### *C. Variants in packet loss probability derivation*

As evaluated in the previous section, changing a rate-increase mechanism alone is not helpful to accomplish the desirable control. In this section, we investigate an alternative way for derivation of the packet loss probability $p$. Originally, it is derived by dividing the number of lost packets by that of transmitted packets within each control interval. As a result, MPEG-TFRCP reacts so quickly against the short-term congestion that it leads to the extreme rate fluctuation. In addition, according to its assumption, the network condition should be stable for (1) to accurately estimate the TCP throughput. One candidate suitable to be applied to (1) is to calculate the packet loss probability over longer interval; an extreme case, from the beginning of the session.

$$p_i = \frac{the\ total\ number\ of\ lost\ packets}{the\ total\ number\ of\ transmitted\ packets} \tag{4}$$

Figure 11 indicates the experimental result of EAI MPEG-TFRCP with cumulative packet loss probability ("EAI-CL"). As described in Subsection III-B, the EAI algorithm leads to the aggressive rate increase in our system, the considerable amount of packets are lost at the beginning and the packet loss probability stays high persistently. In consequence, users are forced to wait long time until they receive the satisfactory video presentation.

To avoid the unpreferable affect of the initial condition, the sending rate of MPEG-TFRCP should be initially small enough and then gradually increased, leading to QAI MPEG-TFRCP with cumulative packet loss probability ("QAI-CL"). The experimental result of QAI-CL is shown in Fig. 12. As

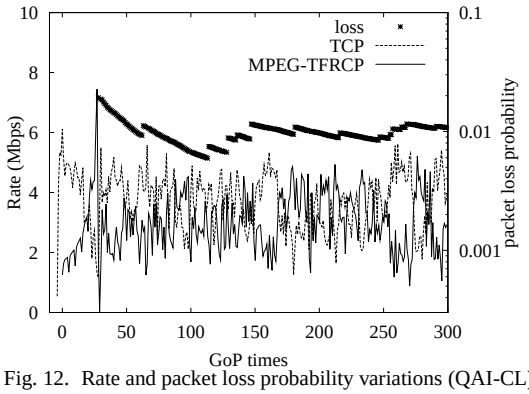

Fig. 12. Rate and packet loss probability variations (QAI-CL)

shown in this figure, the rate variation becomes relatively small and the video quality is improved, MOS value of 3.00. Further, the average throughputs of TCP and our QAI-CL are 3.70 Mbps and 2.97 Mbps, respectively, and the TCP-friendly control is reasonably accomplished. However, rather long time has passed before the sending rate and the estimated packet loss probability become stable.

### *D. Variants in control interval*

In order to attain the effective rate control, the duration of control interval $I_i$ (Fig. 1) must be carefully determined. For example, when the control interval is too short, the sending rate fluctuates enormously by the frequent rate control, and the perceived video quality also becomes unstable. Conversely, in the case of the longer interval, the fairness with TCP connection cannot be accomplished because video applications cannot react changes of the network condition. In addition, since the perceived video quality gradually increases with our QAI-CL MPEG-TFRCP mechanism, a smaller interval is preferred.

The results are summarized in Table I for several settings of control interval such as 8-RTT, 16-RTT, 64-RTT and 96-RTT, in MPEG-TFRCP with QAI-CL. Table I demonstrates that frequent control achieves comparatively high friendliness. However, the frequent rate control introduces a great variation of the video quality, and the subjective video quality by MOS evaluation becomes worse. On the other hand, by the longer control interval, not only the friendliness but the subjective video quality becomes worse. This is because the video application cannot follow the network condition and transmit the video data of the undesirable quality. Further, as the interval becomes longer, users wait longer time for the satisfactory video presentation.

From those results, we conclude that either 16-RTT or 32-RTT control interval is preferable in our system in order that our MPEG-TFRCP connection receives the almost same throughput as a TCP connection and the perceived video quality becomes high and stable.

## IV. Conclusion

In this paper, we have implemented MPEG-TFRCP suitable for video transfer, and then evaluated its practicality. Through several considerations and experiments, we have shown that it is appropriate for real-time video applications to apply our rate control algorithm, MPEG-TFRCP with QAI-CL, where sending rate is gradually increased by quantizer scale control

TABLE I
Results of several settings of control interval

| | Throughput (Mbps) | | Friendliness | MOS value |
|---|---|---|---|---|
| | TFRCP | TCP | | |
| 8-RTT | 3.10 | 3.53 | 0.878 | 2.25 |
| 16-RTT | 2.87 | 3.71 | 0.774 | 3.25 |
| 32-RTT | 2.97 | 3.70 | 0.802 | 3.00 |
| 64-RTT | 2.51 | 4.06 | 0.618 | 3.33 |
| 96-RTT | 2.33 | 4.29 | 0.543 | 2.50 |

and the packet loss probability is cumulatively estimated. We can achieve high-quality and stable TCP-friendly video transfer when video rate is regulated at a control interval of 16 or 32 times as long as RTT.

Although results are not shown in the paper due to space limitation, we applied our QAI-CL MPEG-TFRCP to the other sequences, including basket game, animation and news, and verify our conclusion. Our results might be different from ones on any other operating systems such as Windows and Solaris with different characteristics of the UDP/TCP transmission software. Nevertheless, we will be able to obtain similar results because it is network conditions (RTT, packet losses, and so on) that has considerable influence on the control of MPEG-TFRCP, which we have investigated in this work.

However, we leave several issues for future research. First, we must investigate its practicality to the larger network where great number of connections co-exists and they leave and join during sessions. Second, we should consider the variation of RTT. In the experimentations, observed RTT considerably changes. Then, the perceived video quality becomes unstable and deteriorates.

## References

[1] J.-C. Bolot and T. Turletti, "Experience with control mechanisms for packet video in the Internet," *ACM SIGCOMM Computer Communication Review*, vol. 28, pp. 4–15, January 1998.

[2] R. Rejaie, M. Handley, and D. Estrin, "RAP: An end-to-end rate-based congestion control mechanism for realtime streams in the Internet," in *Proceedings of IEEE INFOCOM'99*, March 1999.

[3] D. Bansal and H. Balakrishnan, "TCP-friendly congestion control for real-time streaming applications," *MIT Technical Report MIT-LCS-TR-806*, May 2000.

[4] M. Mathis, J. Semke, J. Mahdavi, and T. Ott, "The macroscopic behavior of the TCP congestion avoidance algorithm," *ACM SIGCOMM Computer Communication Review*, vol. 27, pp. 67–82, July 1997.

[5] T. Turletti, S. F. Parisis, and J.-C. Bolot, "Experiments with a layered transmission scheme over the Internet," *INRIA Research Report 3296*, November 1997.

[6] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose, "Modeling TCP throughput: A simple model and its empirical validation," in *Proceedings of ACM SIGCOMM'98*, vol. 28, pp. 303–314, September 1998.

[7] J. Padhye, J. Kurose, D. Towsley, and R. Koodli, "A model based TCP-friendly rate control protocol," *UMASS-CMPSCI Technical Report*, October 1998.

[8] W. tian Tan and A. Zakhor, "Real-time internet video using error resilient scalable compression and TCP-friendly transport protocol," *IEEE Transactions on Multimedia*, vol. 1, pp. 172–186, June 1999.

[9] N. Wakamiya, M. Murata, and H. Miyahara, "On TCP-friendly video transfer with consideration on application-level QoS," in *Proceedings of IEEE International Conference of Multimedia & EXPO 2000*, July 2000.

[10] N. Wakamiya, M. Murata, and H. Miyahara, "On TCP-friendly video transfer," in *Proceedings of SPIE International Symposium on Information Technologies 2000*, November 2000.

[11] K. Fukuda, N. Wakamiya, M. Murata, and H. Miyahara, "QoS mapping between user's preference and bandwidth control for video transport," in *Proceedings of IFIP IWQoS'97*, pp. 291–302, May 1997.