# IMPLEMENTATION EXPERIMENT OF VTP BASED ADAPTIVE VIDEO BIT-RATE CONTROL OVER WIRELESS AD-HOC NETWORK

*Hirotaka UJIKAWA and Jiro KATTO*

Dept. of Computer Science, Graduate School of Fundamental Science and Engineering,
Waseda University,
Tokyo, Japan
E-mail: {ujikawa, katto}@katto.comm.waseda.ac.jp

## ABSTRACT

In wireless ad-hoc network, knowing the available bandwidth of the time varying channel is imperative for live video streaming applications. This is because the available bandwidth is varying all the time and strictly limited against the large data size of video streaming. Additionally, adapting the encoding rate to the suitable bit-rate for the network, where an overlarge encoding rate induces congestion loss and playback delay, decreases the loss and delay. While some effective rate controlling methods have been proposed and simulated well like VTP (Video Transport Protocol) [1], implementing to cooperate with the encoder and tuning the parameters are still challenging works. In this paper, we show our result of the implementation experiment of VTP based encoding rate controlling method and then introduce some techniques of our parameter tuning for a video streaming application over wireless environment.

**Keywords:** wireless ad-hoc network, video transmission, adaptive rate control, implementation experiment, TFRC (TCP Friendly Rate Control) [2], VTP.

## 1. Introduction

Recently, the keyword "video" is immediately raising popular interest as seen in spreading video sharing services like YouTube. Furthermore, spreading of smartphones and moderately-priced wireless computers brought a request that people can transmit videos smoothly such as a video chat application even over wireless link. However, the available bandwidth on that environment is severely limited and varies all the time because of the interference, the fading and many other reasons. Therefore, to achieve stable video transmission, a smart adaptive rate controlling method, which can estimate the suitable rate correctly, is necessary. Considering that a packet loss induces serious degradation of video quality, less congestion and less overhead are also required for the method. Moreover, regarding this rate control as a part of video streaming applications, we must harmonize the sending rate to network and the encoding rate because frequent changes of video quality may give a bad impression to users.

To make a smart rate controlling method over wireless environment, some methods have been proposed as the extensions based on the methods like TFRC [2] which are able to achieve their efficiency in wired network. TFRC Wireless [3] and MULTFRC [4] are also the extensions of TFRC. Additionally, a more efficient and stable control has been proposed as VTP in a different approach.

However, there needs an adaptation to use these proposed methods as encoding rate controls, in order to keep stable video streaming. Additionally, these proposals assume that the amount of input data forwarding to the transport layer is always sufficient, while both absence and variance of data amount are often arise in live video streaming. Actually, the results suffering from the variance are observed with our implementation experiment. The main target in this paper is to provide a realistic encoding rate control which is robust against the data absence and variance. Finally, we introduce a simple approach to adapt video encoding rate with more flexible rate estimation based on VTP.

The rest of this paper is organized as follows. In section 2, we describe rate controlling methods for video streaming, TFRC [2] and VTP [1], and the difficulty of suitable rate estimation over wireless environment. Section 3 presents our modifications to provide more stable video streaming as implementation details. In Section 4, we show the environmental settings and the result of our experiment, and introduce a simple solution for the existing frame spacing problem. Section 5 concludes this paper.

## 2 Background

### 2.1 TFRC (TCP Friendly Rate Control) [2]

TFRC is one of the effective rate control methods over wired network which is designed for stable transmission required by applications like video streaming. It does not retransmit the lost packets but send feedbacks to estimate the suitable sending rate. To keep friendliness to TCP, it calculates the suitable rate with the TCP throughput equation (1.1) using RTT and loss probability.

$$X = \frac{s}{RTT\sqrt{\frac{2bp}{3}} + t_{RTO}(3\sqrt{\frac{3bp}{8}}p(1+32p^2))} \quad (1.1)$$

where $X$ is the transmission rate to calculate in bytes per second, $s$ is the packet size in bytes, $RTT$ is the round trip time in seconds, $p$ is the loss event rate, $b$ is the

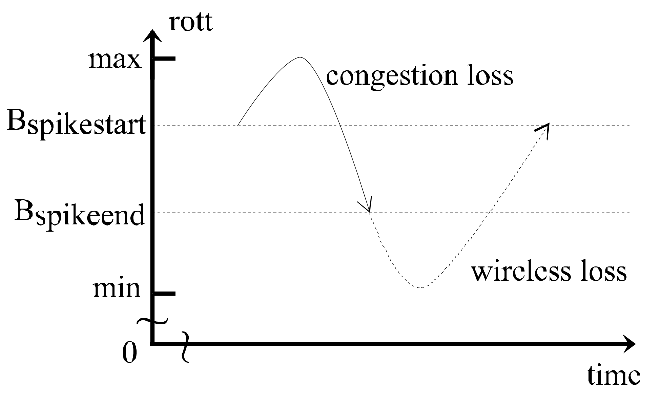

Fig. 2.1: Spike algorithm [3].

number of packets acknowledged by a single TCP acknowledgement which is set to 1 by default and $t_{RTO}$ is the TCP retransmission timeout value in seconds which is set to 4 * $RTT$ by default. With the default value, (1.1) can be transformed into (1.2).

$$X = \frac{s}{RTT * f(p)} \qquad f(p) = \sqrt{\frac{2p}{3}} + 12p(1+32p^2)\sqrt{\frac{3p}{8}} \tag{1.2}$$

Though this equation assumes that the loss happens with congestion, non-congestion loss often happens in wireless environment. Because of this non-congestion loss, TFRC underestimates its rate and therefore cannot provide effective rate control [5].

## 2.2 VTP (Video Transport Protocol) [1]

Different from TFRC, there are proposals trying to be independent from the non-congestion loss and instability of error-prone wireless network. The most of these methods have a commonality that the key is to narrow the trustworthy parameter.

VTP has been proposed for video streaming with its robustness against to the instability over error-prone wireless network. In addition, VTP is also designed to keep friendliness to existing TCP. Firstly, to overcome the effect of the non-congestion loss, VTP has LDA (Loss Differentiation Algorithm) mechanism. Specifically, VTP adopts Spike [3] algorithm as LDA. Spike algorithm of VTP differentiates the loss type using (2.1) with assumption that congestion loss happens when the RTT value is relatively large.

$$\begin{aligned} B_{start} &= RTT_{\min} + \alpha \cdot (RTT_{\max} - RTT_{\min}) \\ B_{end} &= RTT_{\min} + \beta \cdot (RTT_{\max} - RTT_{\min}) \end{aligned} \tag{2.1}$$

where $B_{start}$ and $B_{end}$ are $B_{spikestart}$ and $B_{spikeend}$ in Figure 2.1 respectively, and $RTT_{\max}$ and $RTT_{\min}$ are the RTT values used in place of ROTT (relative one-way trip time) in Figure 2.1.

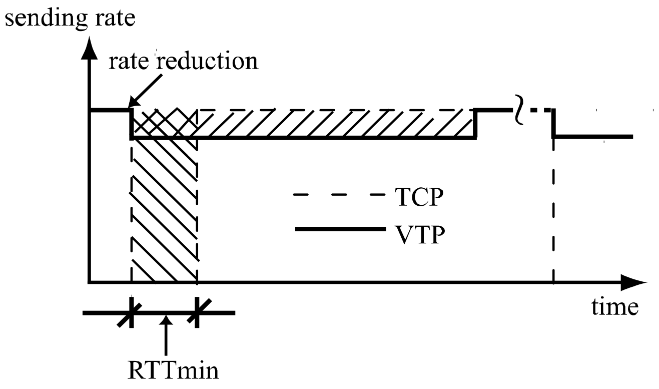

Fig. 2.2: sending rate transition of VTP.

After detecting the congestion loss, VTP keeps the reduced rate based on AR (Achieved Rate) for $\tau$ calculated in (2.2). This is because the transmission rate in the strategy of VTP, which tries to share the equal bandwidth to TCP, is reduced by a small portion but keeps longer. And in order to make VTP to be friendly with TCP, the rate reduction area of Figure 2.2 are to be equal.

$$\tau = RTT_{\max} / (2(1-\gamma)) \tag{2.2}$$

where $\gamma$ between 0 and 1 is the rate reduction ratio selected so that AR * $\gamma$ as the reduced rate is tolerable. AR is measured and smoothed at the receiver as an actual received rate. It is updated as shown in (2.3) and is reported back to the sender.

$$AR_k = \sigma \cdot AR_{k-1} + (1-\sigma) \cdot \frac{1}{2}(S_k + S_{k-1}) \tag{2.3}$$

where $\sigma$ is a fraction close to 1 as the smoothing ratio and $S_k$ is the AR sample, as the number of received bytes during a time period of $T$, divided by $T$.

Then, after keeping the fixed rate, VTP switches to congestion avoidance phase like TCP. To mimic TCP, VTP holds *ewnd* where is the equivalent window size with TCP. As TCP increases its window size on each RTT, VTP calculates the rate by using (2.4) with assumption that RTT grows constantly. Basically, VTP switches these two states triggered by congestion loss.

$$R' = (R + 1/RTT)/(2 - RTT[-1]/RTT) \tag{2.4}$$

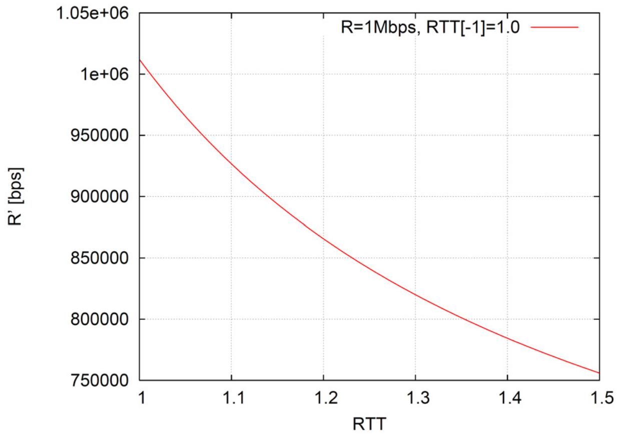

Fig. 2.3: rate update for the next round (2.4)

where $R'$ is the updated transmission rate in packets per second, $R$ is the current transmission rate and $RTT[-1]$ is the RTT value of the previous round in seconds respectively.

With the rate updating formula (2.4), VTP calculates its transmission rate each round, not to queue packets into the buffer. Its concrete curve of (2.4) is diagramed in Figure 3.2 where the previous RTT is 1.0 and the transmission is 1.0 Mbps.

## 2.2 VBR Characteristics of Video Encoder

We have described rate control methods from transporting point of view above. Now, in order to apply the transmission control to live encoding, the VBR (variable bit rate) characteristics of video rate control should be considered. As common technique of video compression, GoP (Group of Picture) structure is used in many encoders. In general, there are different types of frames which have different bit rate allocation for compression efficiency. There are I-Picture (Intra-Picture) which needs larger bit allocation, P-Picture (Predictive-Picture) which needs medium bit allocation and B-Picture (Bi-directional Picture) which needs the smallest bit allocation, respectively. With this GoP structure, the average encoding rate can be assigned per GoP unit. Bit rates of different-type pictures are decided according to rate control strategy of the video encoder. Moreover, the actual bit rate of encoded pictures varies widely according to image complexity, even if the target rate is fixed. Thus, we have to harmonize (or smooth) the VBR characteristics of the video encoder with the averaged CBR (constant bit rate) values estimated by transport protocols.

## 3 Implementation Details

To test video transmission in a real environment, we implemented VTP into x264 [6] which is an open source H.264/AVC [7] encoder as shown in Figure 3.1. Input images at the sender are encoded with encoding rate of the time, and transmitted to the receiver one after another. The encoded frames are split into packets not to overrun the MTU (Maximum Transfer Unit). On transmission, these packets are paced according to the transmission rate of the time. A timestamp is inserted into every packet. The receiver measures the AR, echoes the timestamps and reports the feedback to the sender. To detect a packet loss, the receiver monitors the lack of sequence number which is also inserted into the packet header. On receiving a feedback, the sender calculates its RTT and updates the rate described in Section 2.1. Then, the sender also tries to adapt the encoding rate with the suitable rate.

### 3.1 Restrict the Encoding Rate Transition

Because frequent changes of the encoding rate may cause degradation of video quality, we change the encoding rate only when the rate is varied by over *thresh_video* which is the threshold of video encoding rate. We used 300 kbps as the threshold heuristically.

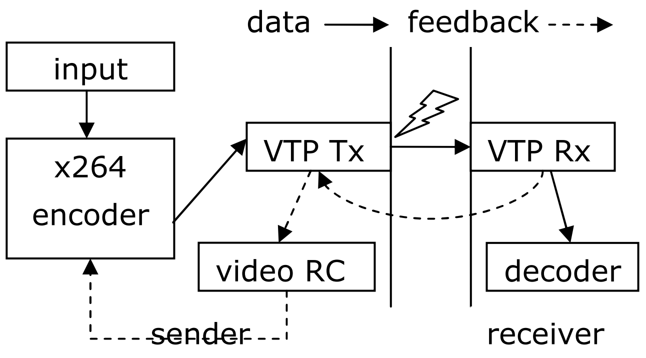

Fig. 3.1: our VTP implementation into x264

## 4 Implementation Experiment

### 4.1 Environment and Settings

The topology consists of two nodes simply which are the sender and the receiver with static routing. To avoid the varying results, the nodes are sufficiently close to each other, where can overcome the effect of interferences. The sender encodes 1000 frames of a specific sequence and transmits them to the receiver. Each node uses IEEE 802.11b with 11Mbps link of BUFFALO WLI-PCM-L11 wireless card with the driver *orinoco_cs* version 0.15 in ad-hoc mode. The sender has two Xeon 1.86GHz processors and the distribution of Linux is Fedora 9 with the kernel version 2.6.26. The parameter settings are represented in Table 4.1.

Table 4.1: parameter settings

| | |
|---|---|
| $\alpha, \beta$ in Spike | 0.5, 1.5 |
| $\gamma$ VTP reduction ratio | 0.9 |
| $\sigma$ EWMA coef. | 0.9 |
| initial encoding rate | 3000 kbps |
| key frame interval | 8 |
| target frame rate | 8 |
| sequence size | 720 x 480 |

### 4.2 Encoding Rate Control with VTP

We are trying to find the suitable encoding rate for live video streaming with the help of VTP rate estimation. However, in the situation that the probing rate in (2.4) is distant from the AR, the encoding rate should not be set simply to the probing rate, because an overlarge encoding rate causes playback delay in a real-time video streaming. The result of this transmission is shown in Figure 4.1. Though AR is keeping stably, the transition of probing rate of VTP and the encoding rate are going far above the AR.

Consequently, this result suggests adapting the encoding rate to the AR. We show the result in Figure 4.2, where a smoother transition of the encoding rate is seen in the result. However, the AR reduction is also seen in the middle, in spite of its vacancy to the maximum achievable rate. Because the incorrectness of AR measuring can also affects transmission control, we have investigated what causes this difference.

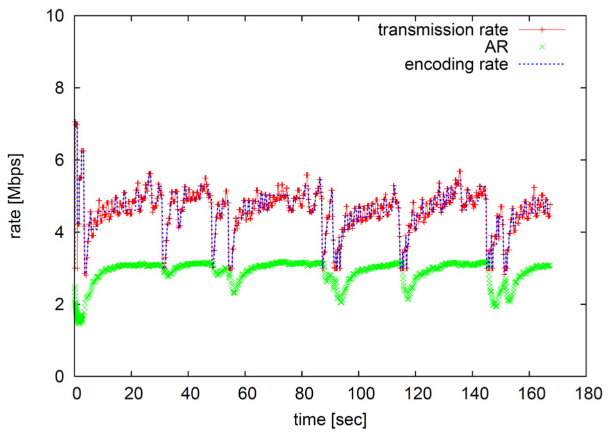

Fig 4.1: applying encoding rate to probing rate

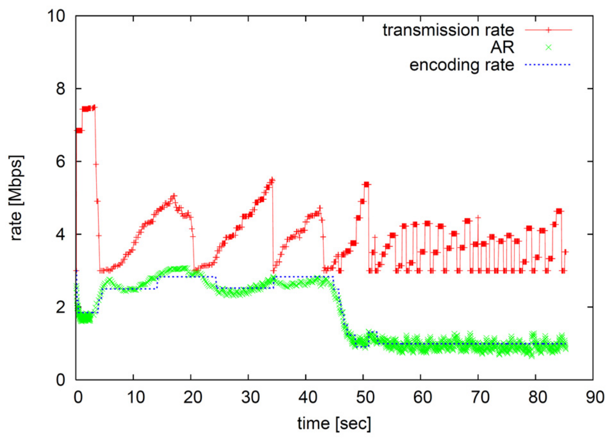

Fig 4.2: applying encoding rate to AR

## 4.3 Underestimation of AR

In the situation that the encoder does not provide data output at the speed of the transmission rate of VTP, the AR may be degraded by sending the insufficient amount of data in contrast with the maximum achievable rate. This is because the AR is measured every time period of *T* in (2.3). Firstly, with large latency of encoding, such as insufficient power of processing, the encoder can not provide the output on time. Secondly, as described in 2.2, the bias of the bit rate per frame according to the frame type may also cause the data lacked span by a large GoP size. Moreover, while the complexity of the input sequence is lower than target rate, the AR is to be underestimated. Finally, all of these cases result in spacing among the frames. As the feedback of degraded AR limits the encoding rate, a different approach of measuring AR is required without using the constant period of time.

Now, we propose frame-driven AR measurement illustrated in Figure 4.3. To be robust against the spacing of frames, which are described above, we modified the timing of AR measurement. The measurement starts when the first packet of a frame is arrived, and calculates the sum of received bytes except the first packet, then let the sum divided by $T'$ as $S_k$. As shown in Figure 4.4, an even smoother transition of encoding rate than Figure 4.2 is seen. Though the recovery of AR looks like positive, the total encoding latency for the larger encoding rate is not ignorable.

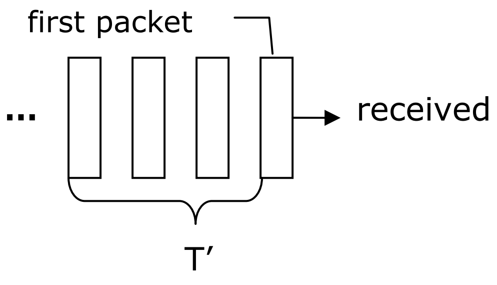

Fig 4.3: AR measurement span instead of *T*

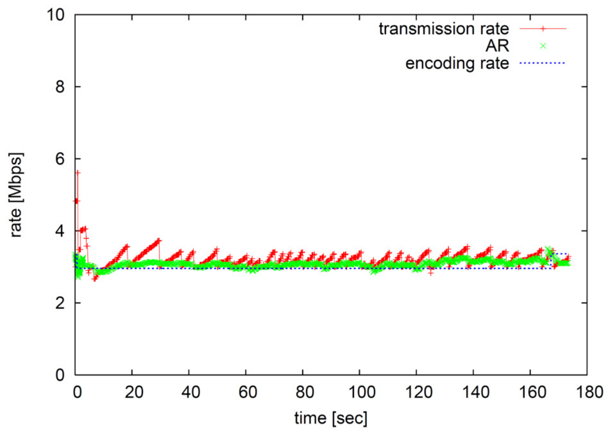

Fig 4.4: applying encoding rate to modified AR

## 5 Conclusion

In this paper, we introduced a simple approach to adapt the encoding rate with VTP based rate estimation for live video streaming over wireless ad-hoc network. And we extended the estimation to be robust against the frame spacing problem, by changing the span of AR measurement. Finally in our implementation experiment, we confirmed the advantage of efficiency and disadvantage that the larger encoding rate may induce longer delay in our proposal.

We are now working on controlling the transmission rate of VTP, increasing unreasonably while the sending throughput is under this rate. And we plan to collaborate with capacity probing method, keeping the encoding rate stably. We are also planning to focus on the latency consideration of this paper in the future.

## 6. REFERENCES

[1] Guang Yang, Mario Gerla and M. Y. Sanadidi, "Adaptive Video Streaming in Presence of Wireless Errors," MMNS 2004, LNCS 3271, pp. 26-38, 2004.

[2] M. Handley, S. Floyd, J. Padhye and J. Widmer, "TCP Friendly Rate Conrol (TFRC)," RFC3448, 2003.

[3] Song Cen, Pamela C. Cosman and Geoffrey M. Voelker, "End-to-End Differentiationi of Congestion and Wireless Losses", IEEE ACM Transactions on Networking, Vol. 11, No. 5, pp. 703-717, 2003.

[4] M. Chen and A. Zakhor, "Rate control for streaming video over wireless," In Proceedings of the IEEE INFOCOM 2004, vol.2, pp. 1181-1190, 2004

[5] Guang Yang, Ling-Jyh Chen, Tony Sun, Mario Gerla and M. Y. Sandidi, "Real-time Streaming over Wireless Links: A Comparative Study," ISCC 2005, pp. 249-254, 2005.

[6] Laurent Aimer, Loren Merrit and Eric Petit et al., "x264 – a free h264/avc encoder", http://www.videolan.org/developers/x264.html

[7] T. Wiegand, GJ. Sullivan, G. Bjontegaard and A. Luthra, "Overview of the H.264/AVC video coding standard," Circuits and Systems for Video Technology, IEEE Transactions on , vol.13, no.7, pp.560-576, 2003