針對 H.264 的即時階層式為基礎可調適層級影像通訊系統的設計與實作

Design and Implementation of a Real-Time Layer-based Scalable Video Communication System for H.264

曾嘉影
Chia-Ying Tseng
大同大學資訊工程系
cytseng@ttu.edu.tw

李良德
Liang-Teh Lee
大同大學資訊工程系
ltlee@ttu.edu.tw

劉岡遠
Kang-Yuan Liu
大同大學資訊工程研究所
d9306005@ms2.ttu.edu.tw

杜安蓉
An-Jung Tu
大同大學資訊工程研究所
g9306003@ms2.ttu.edu.tw

摘要

近年來，隨著串流媒體的發展，使得可調適H.264 影像編碼被設計用於即時影像通信系統。此法是建構於精細可調適編碼(FGS)影像編碼方法。這方法提供框架去適應頻道頻寬的變化，最近也在MPEG-4 的串流影像協定中成為標準。FGS 也對編碼器提供選擇性提升明顯可增加影像品質的區域。在本論文中，我們提出並實現了一個簡單和有效的即時的層基礎的可調適影像通訊系統。在此系統中我們使用漸進式精細可調適編碼(PFGS)規劃來編碼進階層和 H.264 的基本層。這個規劃是直接在H.264實做FGS以及不對FGS的使用任何H.264專門的特性。此外，我們使用適應階層式為基礎且以最小鬆散值先執行(LB-LLF)的排程演算法層，來改善在網路上視/音訊的輸出品質以達到同步效果。實驗結果顯示使用我們的方法後，影像串流的品質明顯地改善。
關鍵字 ：串流視訊、串流媒體、視訊處理、即時排程

Abstract

With the recent development in stream media, scalable H.264 video encoding is designed to be used in real-time video communications systems. The method is based on the Fine Granular Scalability (FGS) video coding method. It provides a framework to adapt to variations in the channel bandwidth and was recently standardized in the Streaming Video Profile of MPEG-4. FGS also provides to the encoder the ability to selectively enhance the regions that are visually increasing the subjective video quality. In this paper we proposed and implemented a simple and effective real-time layer-based scalable video communication system. In this system we use Progressive Fine Granularity Scalable (PFGS) scheme to encode the enhancement layer and H.264 to encode the base layer. This scheme is a direct implementation of FGS onto H.264 and does not employ any H.264-specific features for FGS. We also use the adaptive Layer-Based Least-Laxity-First (LB-LLF) scheduling algorithm to improve the output quality of video on network and to achieve synchronized playback effect. The experimental results show that subjective quality of the video stream is significantly improved using our methods.
Keyword: streaming video, streaming media, video processing, real-time scheduling

1. Introduction

Multimedia applications typically require computations for large amount of data. In late 2001, ISO/IEC MPEG and ITU-T VCEG decided on a joint venture towards enhancing standard video coding performance – specifically in the areas where bandwidth and/or storage capacity are limited. This Joint team of both standard organizations is called Joint Video Team (JVT). The standard formed is called H.264/MPEG-4 part 10 and is presently referred to as JVT/H.26L/Advanced Video Coding (AVC) [3]. An emerging video coding standard named H.264 or MPEG-4 Part 10 aims to code video sequences at approximately half the bit rate compared to MPEG-2 at the same quality. It also aims at having significant improvements in coding efficiency, error robustness and network friendliness [4].

H.264 is different from the previous video standards. It introduced several new techniques and had about up to 50% improved performance over the MPEG-4. And it makes H.264 an attractive choice for the streaming video over internet and wireless networks. However, not much research has been done for introducing scalability to the H.264 standard. Although H.264 has features that enable efficient switching of different streams at different rates, but it don’t have the ability to adapt to changing bandwidth conditions in a fine-granular fashion.

There has been some work done combining FGS with H.264. Progressive FGS (PFGS) has been proposed to encode the enhancement layer and H.264 to encode the base layer [1]. This scheme is a direct implementation of FGS onto H.264 and does not employ any H.264-specific features for FGS. Recently, we proposed low-complexity H.264-based FGS structure which increased the error-resilience of the overall system (encoder and decoder) and decreased its complexity by using the features that are already present in the base layer H.264 encoding.

In this system we use Progressive Fine Granularity Scalable (PFGS) scheme to encode the enhancement layer and H.264 to encode the base layer. This scheme is a direct implementation of FGS onto H.264 and does not employ any H.264-specific features for FGS. We also use the adaptive Layer-Based Least-Laxity-First (LB-LLF) scheduling algorithm to improve the output quality of video on network and to achieve synchronized playback effect. The proposed algorithm considered real-time constraint, unequal priorities of scalable media stream in different layers, and a good trade-off between coding efficiency and drifting error. Section 2 gives a brief overview of scalable video coding technique adopted in H.264, and introduced H.264 based FGS bit-plane coding scheme. Section 3 presents the architecture of the layer-based scalable streaming media system for H.264 and our proposed method for transmission through networks. Section 4 presents the real-time Least-Laxity-First (LLF) scheduling algorithm for H.264 layer-based scheduler. Section 5 shows the experimental results that subjective quality of the video stream is significantly improved using our methods. Conclusions and discussions are presented in Section 6.

2. Scalable Video Coding for H.264

The Fine Granularity Scalable (FGS) [2] video coding technique is adopted in MPEG-4 standard. In the proposed scheme, the base layer is predicted from the reconstructed base layer of a reference frame. And all enhancement layers are predicted from an enhancement layer of the reference frame. The Fine Granularity Scalable scheme can easily adapt to channel bandwidth fluctuations, because the bit plane coding technique provides the fine granularity scalability in the enhancement layer. The enhancement bit stream can be truncated according to available channel bandwidth due to the bit plane coding produces an embedded bit-stream with fine granular scalability. However, the coding efficiency of the FGS is not so good as the traditional scalable coding since its motion prediction is always based on the lowest quality base layer. The PSNR of the FGS may drop 3dB or more at the same bit rate while compared with the non-scalable video coding schemes.

The H.264 compression algorithm assumes that a video sequence is composed of similar images, which are related to each other. Typically, there are small spatial temporal changes between the current image and the next one, since most of the image segments are almost identical, and just few might change location or be replaced with new ones. Each incoming image is processed either as I-frame or only as the spatial-temporal difference from the previous image (P-frame) or bi-directional image (B-frame). A conventional video transmission system of H.264 video encoding is based on a sequential encoding of frames. In most existing video coding standards including H.264, within each frame video encoding is typically based on sequential encoding of macro-blocks (MBs). Although slices could he formed, the strategy of one frame in one packet is usually beneficial to exploit the full intra-frame correlations within one frame. Each generated frame is channel encoded and transmitted over the wireless channel. The applied channel coding maps the wireless channel into a perfect packet erasure channel, i.e., frames are either lost or perfectly decoded. In addition to the forward link it possible that a low bit-rate reliable back-channel from the decoder to the encoder is available which allows reporting a frame delayed version the observed channel behavior at the decoder to the encoder. The decoder processes the received sequence of packets. Whereas correctly received packets are decoded as usual far the lost packet an error concealment algorithm has to be invoked. In inter mode, i.e., when motion compensated prediction (MCP) is utilized, the loss of information in one frame has a considerable impact on the quality of the following frames, if the concealed image content is referenced for MCP.

The FGS structure consists of an MPEG-4 non-scalable base layer encoded at R_{base} and an enhancement layer encoded using bit-plane coding at a maximum bit-rate R_{max}. During transmission, the enhancement layer can be truncated at the rate $R_{avaliable}$ to fully utilize the available bandwidth. Because the enhancement portion is encoded using bit-plane coding, the quality of the video at the decoder side increases with more bit-planes received. Figure 1 illustrates the bit-plane coding FGS is standardized in MPEG-4 Streaming Video Profile and it uses MF'EG-4 coding blocks. We have introduced H.264 based FGS and modified the FGS structure that uses H.264's superior features that decreases the complexity and increases the error resilience of the overall system.

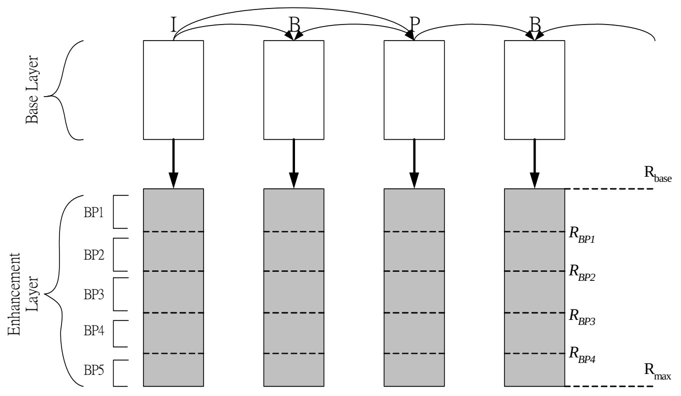

Figure 1: FGS Structure, BP1, BP2...BP5 are the bit-planes for the enhancement layer

3. System Architecture

To increase the traffic quality for streaming media in the internet, a server transmits multimedia stream packets to a receiver that buffers these packets for playback. A typical streaming system consists of clients and servers on a network. Figure 2 shows the architecture of the layer-based scalable streaming media system for H.264. The client requests are sent to the server via network connections, which also serves for transmission of media data. The buffers in each client are used to provide some tolerance on variations in network delay as well as data consumption rates. The scheduler in the server controls the packet size and sequence, manages the server transmitted buffer and packets via the network to the clients' buffers. The scalable video sequence consists of many frames, which are compressed into several layers. The layers are packed as packet and fed into the server's transmission buffer. These are the packets waiting to be scheduled for transmission. The server's scheduler selects one candidate packet at a time from those buffers and sends it to the network channel. [9][10]

The problem of the non-scalable system comes for the "all-or-nothing" transmission strategy. Either the entire frame can be decoded or everything is lost. The probability of the loss obviously depends on the applied channel coding rate. However, with this strategy we have overprotected the source for most channel realizations and therefore limit the average bit-rate for the video transmission. It has been shown for still image transmission that the application of scalable or progressively coded source in combination with unequal error protection can enhance the system significantly. In proposed system, we have presented a channel and complexity scalable transmission system which outperforms previous approaches especially for wireless fading channels.

The client includes a base receiver buffer and several enhance receiver buffers to compensate transmission delay jitter. The dispatcher stores incoming packets in reception order into the base receiver buffer and enhance receiver buffers. Packets are de-capsulated in sequence number order within receiver buffers. If a de-capsulated packet is a single frame unit packet, the unit contained in the packet is passed directly to the decoder from buffers. If a de-capsulated packet is not a frame unit, the units contained in the packet are passed to the decoder in the order they are encapsulated in the packet. If a de-capsulated packet is a fragment, all the fragments of the fragmented frame unit are concatenated and passed to the buffers that they belong to.

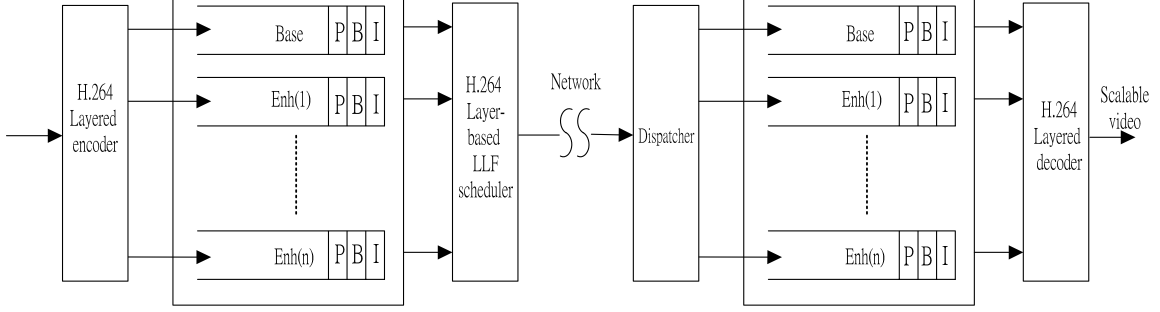

Figure 2: The architecture of PFGS video streaming system

4. Real-Time Scheduling Algorithm for H.264 Layer-Based Scheduler

4.1 Basic Technologies

Least-Laxity-First algorithm (LLF) is optimal dynamic scheduling algorithms [11][12]. A main advantage of these dynamic scheduling algorithms can achieve a theoretically possible 100% processor utilization without deadline misses.

The LLF algorithm is a dynamic scheduling method, i.e., it makes the decision for which task to execute next at scheduling time. For every task ready to run at the given moment the difference ***L*** between the time of deadline ***D*** and the end of processing time ***P*** is computed. This difference is called as laxity or slack which can be seen as an inverted priority value. The task with the smallest ***L***-value is the one to be executed next. Whenever a task other than the currently running one has the smallest laxity, a context switch will occur. LLF algorithm is an optimal scheduling method. That means, if a given set of tasks is schedulable, then it can be scheduled by Least-Laxity-First. Another great advantage of the LLF algorithm is that no further assignment for fixed priorities to the tasks at development time except scheduling test. Furthermore, a task going to miss its deadline is recognized at the same moment when its laxity turns to zero with the task currently not being executed. At that time the deadline is not yet reached and emergency measures can be taken to cover the miss of a deadline. With LLF it is possible for the scheduler to detect an impending deadline miss during the execution of tasks. [13][14]

4.2 Layer-Based LLF Scheduling Algorithm

Each task (packet) *T* is characterized by the following parameters:

$T_x(y)$: the task (packet) of the y^{th} layer in frame x. The tasks are put into the transmission buffers according to the decoding order.

$E_x(y)$: the earliest time at which the task $T_x(y)$ becomes ready for scheduling in transmission buffer.

$P_x(y)$: the processing time of a task $T_x(y)$.

$L_x(y)$: the laxity value of the task $T_x(y)$. $L_x(y) = D(x) - P_x(y)$.

$LT_x(y)$: the laxity value of the task $T_x(y)$ at a given time.

$D(x)$: the latest time at which all packets of frame x should be sent to the client, otherwise it is too late for playback.

It is instinctive to select and schedule packets by Least-Laxity-First scheduling algorithm for delivery of scalable streaming media over a network. We proposed the adaptive Layer-Based Least–Laxity -First (LB-LLF) scheduling algorithm, which combines importance and priority of the layer. As packets at different layers have different effects on the playback quality, we set the higher priority to the lower (more important) layer packets and set the lower priority to the higher (less important) layer packets. Thus, important layer packets should be transmitted earlier, with more chances to be transmitted to client buffer for higher playing back quality. Packets in the same layer are served according to LLF scheduler. The detailed description of the algorithm is given as follows.

Layer-based LLF real-time scheduling algorithm :

```
Step 1: Compare the current time $t_{cur}$ with the
        deadline $D(x)$ of the packets in the server
        transmit buffer. If $t_{cur} > D(x)$, remove the
        packet from the server transmit buffer.
Step 2: Let $T$ = set of ready packets with the lowest
        layer in the server transmission buffers.
        Calculate the least-laxity value $L_x(y)$ for
        each frame set $T$ from frame number $X = 1$
        to $N$ (where $N$ is the maximum buffer size in
        server.)
Step 3: Select the smallest least-laxity value $LT_x(y)$
        packets from $T$ and send it to the client via
        the network.
Step 4: Set the release time as the current time, $t_{cur} =$
        $t_{cur} + P_x(y)$.
        Go to Step 1.
```

The LB-LLF scheduling algorithm has two key points. First, it selects the packets from the lowest layer in ready state, the important packet to be sent by the server will transmit much earlier than its playback time, and this important packet will have more chances to be transmitted to the client buffer for displaying. Second, it calculates the least-laxity value for each frame set from frame number X = 1 to N and services the packet of the smallest laxity set. In real-time constraint, unequal priorities of scalable media stream in different layers and achieving a good trade-off between coding efficiency and drifting error are considered in the proposed algorithm. This guarantees the better usage of available channel bandwidth and the smoother playback in client.

5. Experimental Results

We use the Microsoft H.26L-PFGS and the JVT JM 2.0 encoder/decoder to generate the simulation data. It is an efficient scalable coding scheme with fine granularity scalability, where the base layer is encoded with H.264, and the enhancement layer is encoded with PFGS coding. For comparison performance of the scheduling algorithms in the given channel bandwidth, we set the base layer as the highest priority that can get all the base layer data without any packet losses. In the experiment, the sequence Foreman in QCIF format is used. It is encoded with 25 frames per second and 200 frames are encoded and transmitted by the proposed scheduler. Because the maximum level of bit-plane is 4 in the Foreman sequence, there are 4 layers in the Enhancement layer. Different enhancement layer bit-plane has different frame size. The sizes of the enhancement layers in the first 50 frames are shown in Figure 3. The enhancement layer 1 (Enh1) is the smallest in size, but it is the most significant layer. The enhancement layer 4 (Enh4) is the largest in size, but it is the least significant layer. The average rate of video data with all enhancement layers is 872.5 Kbps. The transmission buffer size is set to 25. The playback frame rate is 25 frames per second. The scheduling time is 0.1ms. The channel bandwidth is various from 100 to 1800 Kbps.

Figure 4 shows comparisons of PSNR among H.263, MPEG-4, H.264, and H.264 Layer-Based system with various bit rates from 100 to 1800 Kbps. With the H.264 Layer-Based system, the experimental results of the average PSNR is 34.71 dB. It improves about 0.95 dB over a H.264 system, 4 dB over a MPEG-4 system, and 6.44 dB over a H.263 system. The gains mainly result from the better average encoding performance which can be observed in Figure 4 from the high probability of relatively good decoded PSNR, e.g. in more than 70% the decoded PSNR is above 2 dB for H.264 over MPEG-4. And the PSNR of H.264 with Layer-Based is still better than the PSNR of H.264. As mentioned above this improvement guarantees the effective usage of available channel bandwidth and the better quality of playback in client.

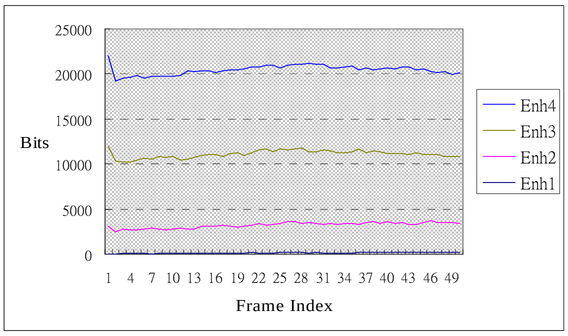

Figure 3: The bit size of the enhancement layers

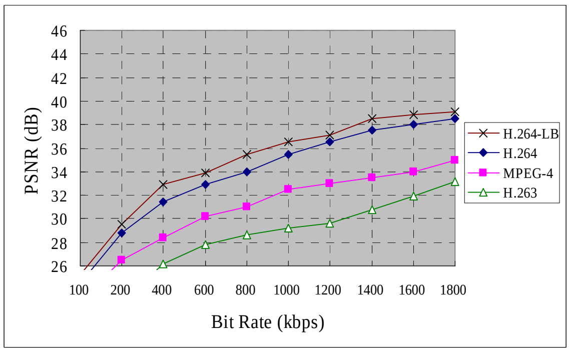

Figure 4: The PSNR with various Bit Rate

6. Conclusions

In this paper we proposed and implemented a simple and effective real-time layer-based scalable video communication system. Our proposed system uses Progressive Fine Granularity Scalable (PFGS) scheme to encode the enhancement layer and H.264 to encode the base layer. The adaptive Layer-Based Least-Laxity-First (LB-LLF) scheduling algorithm is also presented to improve the output quality of video over network and to achieve better playback quality. The proposed algorithm considered real-time constraint, unequal priorities of scalable media stream in different layers, and a good trade-off between coding efficiency and drifting error. The experimental results show that subjective quality of the video stream is significantly improved using our methods.

Acknowledgement

The material reported in this paper is based upon work supported in part by the National Science Council (NSC), Taiwan, under grant NSC 93-2215-E-002 -010／001.

References

[1] Y. He, F, Wu, S. Li, Y. Zhong, and S. Yang, “H.26L-based Fine Granularity Scalable Video Coding,” in Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS), Scottsdale, Arizona, USA, 26-29 May 2002, vol. 4, pp. 548-551.

[2] Wang, Q., Wu, F., Li, S., Zhong, Y., and Zhang, Y., (2001) ‘Fine-granularity spatially scalable video coding’, Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, 2001, (ICASSP '01), vol.3, pp.1801–1804.

[3] Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification (ITU-T Rec. H.264/ISO/IEC 14

496-10 AVC), Mar. 2003.

[4] A. Tamhankar and K. R. Rao, ”An overview of H.264/MPEG-4 Part 10,” The 4th EURASIP Conference focused on Video/Image Processing and Multimedia Communications, Vol. 1, pp.1-51 July 2003.

[5] Gharsalli, F.; Baghdadi, A.; Bonaciu, M.; Majauskas, G.; Cesario, W.; Jerraya, A.A., “An efficient architecture for the implementation of message passing programming model on massive multiprocessor,” Proceedings of the 15th IEEE International Workshop on Rapid System Prototyping, 2004. 28-30 June 2004, pp. 80 – 87.

[6] Chia-Ying Tseng, Liang-Teh Lee, Yu-Lan Shih, and Kang-Yuan Liu, "Adaptive Layer-Based Scheduling for Real-Time Transmission on Scalable Multimedia Stream," Proceedings of the Tenth International Conference on Distributed Multimedia Systems (DMS'2004), Sep. 2004, pp. 389-392.

[7] Chia-Ying Tseng, "The adaptive layer-based scheduling system for embedded real-time transmission on scalable multimedia stream," International Journal of Embedded Systems (IJES), Vol. 2, No. 1, 2006.

[8] K. Ugur and P. Nasiopoulos, “Design Issues and a Proposal for H.264-based FGS,” contribution MPEG03/M9505, ISO/IEC JTC/SC29/WG11, Pattaya, Thailand, March 2003.

[9] Gao, K., Gao, W., He, S., Gao, P., and Zhang, Y., (2003) ‘Real-Time Scheduling on scalable media stream delivery’, Proceedings of the 2003 International Symposium on Circuits and Systems, 2003(ISCAS '03), vol.2, pp. II-824 - II-827.

[10] Gao, K., Zhang, Y., Gao, W., and He, S., (2003) ‘Real-Time Scheduling Supporting VCR Functionality For Scalable Video Streaming’, 14th IEEE Proceedings on Personal, Indoor and Mobile Radio Communications, 2003(PIMRC 2003), vol.3, pp.2711–2715.

[11] Golatowski, F., Hildebrandt, J., Blumenthal, J., and Timmermann, D., (2002) ‘Framework for Validation, Test and Analysis of Real-time Scheduling Algorithms and Scheduler Implementations’, Proceedings of the 13th IEEE International Workshop on Rapid System Prototyping, 2002, pp.146-152.

[12] Golatowski, F., Hildebrandt, J., and Timmermann, D., (1998) ‘Rapid Prototyping with Reconfigurable Hardware for Embedded Hard Real-Time Systems’, 19th IEEE Real-Time Systems Symposium, WIP-Proc., Madrid, Spain ,1998.

[13] Hildebrandt, J., Golatowski, F., and Timmermann, D., (1999) ‘Scheduling Coproessor for Enhanced Least-Laxity-First Scheduling in Hard Real-Time Systems’, Proceedings of the 11th Euromicro Conference on Real-Time Systems, June 1999, pp.208-215.

[14] Oh, S.H. and Yang, S.M., (1998) ‘A Modified Least-Laxity-First Scheduling Algorithm for Real-Time Tasks’, Proceedings of the 5th International Conference on Real-Time Computing Systems and Applications, Hiroshima, Japan, 1998, pp. 31-36.