NEW GENERATION OF REAL-TIME SOFTWARE-BASED VIDEO CODEC: POPULAR VIDEO CODER II (PVC-II)

Ho-Chao Huang[†] Ja-Ling Wu[‡]

[†]Institute of Information Science, Academia Sinica, Taipei, Taiwan, ROC

[‡]Communications & Multimedia Lab.

Department of Computer Science and Information Engineering
National Taiwan University, Taipei, Taiwan, ROC

Abstract

A new generation of real-time software-based video coder, the Popular Video Coder II (PVC-II), is presented in the paper. The transform and the motion estimation parts are removed and the quantizer and the entropy coder are modified in the PVC-II as compared with the traditional video coder. Moreover, the PVC-II improves the coding performance of its previous version, the Popular Video Coder[1], by introducing several newly developed efficient coding techniques, such as the adaptive quantizer, the adaptive resolution reduction and the fixed-model intraframe DPCM, into the codec. The coding speed, compression ratio and picture quality of the PVC-II are good enough for applying it to various real-time multimedia applications. Since no compression hardware is needed for the PVC-II to encode and decode video data, the cost and complexity of developing multimedia applications, such as video phone and multimedia e-mail systems, can be largely reduced.

Keywords:

Communications, multimedia system, software-based video compression, video data compression, video on demand, video phone/conference.

1 Introduction

How to efficiently compress the video data is one of the most important functions in multimedia applications. Most existing video coding techniques, such as H.261[2] and MPEG[4], have been designed to compress video data by using the discrete cosine transform (DCT)[5], the DPCM[6] and the motion compensation (MC)[7]. One of the major problems in MC/DCT-based technique is the corresponding high computation complexity, which inhibits the real-time applications of it by using software only. As a result, the cost of such video applications is high and the implementation and maintenance are difficult.

Recently, to realize the video coder by software to reduce both the cost and the complexity of multimedia systems has been widely discussed. The real-time software-based moving picture coding (SBMPC) system[8] and the Popular Video Coder (PVC)[1] have been proposed to achieve this goal. In [1], the PVC was applied to implement a video phone system, and was shown good performance. However, the coding speed and the compression ratio of the PVC are not high enough for applying it to higher resolution and/or higher speed video applications, such as video conferencing and video recording systems. Moreover, due to its low compression ratio for intraframe data, the PVC is unsuitable for random accessed video playback applications (such as video on demand), either.

In this paper, a new generation of Popular Video Coder, the Popular Video Coder II (PVC-II), is proposed. The PVC-II improves the performance of the Huffman coder of the PVC, combines the fixed-model DPCM technique for intraframe coding, and applies adaptive resolution reduction technique for interframe coding. With the aids of these new techniques, the PVC-II compresses the video data more efficiently than the PVC does. More interestingly, the coding speed of PVC-II is much more faster than other well-known software video coders, such as the Video-for-Windows¹ and the MPEG coder². In our experiments, the average compression ratio, the peak SNR, and the average frame rates for both encoding and decoding are about 40, 27 dB, 31 frames/second and 51 frames/second, respectively. The resolution of each video frame in these experiments is 128 × 128 with color. In our laboratory³, the PVC-II

¹The Video-for-Windows used in this paper is VFW V1.0.

²The MPEG coders are CMPEG V1.0 for MPEG encoder and DMPEG V1.1 for MPEG decoder, copyright by Stefan Eckart, June 1993, and was posted on newsgroup: alt.binaries.pictures.utilities. Email: stefan@lis.e-technik.tu-muenchen.de

³The Communications and Multimedia Laboratory of National Taiwan University

has been successfully applied to implement a prototyping video on demand system for real-time recording and playback the video with a resolution of 352×288 .

This paper is organized as follows. Section 2 reviews the problems of the MC/DCT-based video coder and the previous version of Popular Video Coder. Section 3 presents the proposed Popular Video Coder II. Experimental results are shown in section 4. The video on demand prototyping system using the PVC-II as video coding kernel is introduced in section 5 and conclusions are stated in the last section.

2 The problems of the MPEG and the Popular Video Coders

MC/DCT-based video coders have the capability of encoding moving pictures with relatively high compression ratio and good picture quality. However, the high computation complexity involved (especially in the encoding side) makes this coding scheme hard to implement multimedia applications using software coders only, even though lots of fast algorithms of MC and DCT have been developed. Software-based MPEG-1 decoders, such as three well known commercial products: Xing, Comcore and VMPEG, have real-time performance, but the encoding still be a serious problem.

The PVC has the capability to encode video data efficiently and is suitable for some low bit rate communication applications such as video phone. The block diagram of the PVC encoder is shown in figure 1. Some drawbacks of the PVC make it hard to be used more widely. This is the reason why the PVC-II is proposed. The drawbacks of the PVC are stated as follows.

- Low compression ratio for intraframe coding.

Because the interactive video playback systems, such as video on demand and video games, are required to provide the random access services when retrieving the compressed video data, many intraframe data should be periodically inserted in the video sequence. Due to the low compression ratio of the PVC for intraframe coding, the overall compression ratio would be reduced significantly when applying the PVC to those kind of applications.

- Unable to achieve very high compression ratio and speed for interframe coding.

Although the coding speed and compression ratio of the PVC can meet the real-time compression and transmission requirements for video phone system, they are still not high enough for applying to more high-speed and/or high-definition applications, such as video conferencing and video on demand systems.

3 The Popular Video Coder II (PVC-II)

The block diagram of PVC-II encoder is shown in Fig. 2. The input video sequence of the PVC-II is organized as groups of frames (GOFs). There are four frame types within a GOF, which will be described in section 3.2. These frames are controlled by the *Frame Type Controller*. The type of frame affects the functions of the *Adaptive Quantizer*, the *Frame Buffer*, the *Frame Coder Selector* and the *Windowed Length-Limited Huffman-Type Coder*.

In order to meet the real-time constraint of video applications, the PVC-II discards both the DCT coder and the motion compensator. In addition, the adaptive resolution reduction and the fixed-model intraframe DPCM techniques (selected by the *Frame Coder Selector*) are adopted to speed up the coding process and increase the coding efficiency of the PVC further. A high performance version of the dynamic Huffman coder, called the *Windowed Length-Limited Huffman-Type Coder*, is proposed in this paper as part of the entropy coder of the PVC-II. The *Adaptive Quantizer* of the PVC-II controls the bit rate of the output data and the picture quality of the reconstructed image. Finally, the behavior of the *Adaptive Quantizer* is controlled by the *Frame Type Controller*. The flow of the PVC-II coding process is described as follows.

For color image input, the pixel components are first converted from the RGB domain to the YUV domain, and then the Y, U and V components (in 4:2:0 format) of the image are encoded independently. When encoding an intraframe, the Y, U and V components of the input pixels are subtracted from the reconstructed previous pixels. When encoding an interframe the Y, U and V components of the input image are subtracted from the reconstructed previous frame stored in the *Frame Buffer*, and then subsampled by the *Adaptive Resolution Reducer*. The residual pixels are then quantized by the *Adaptive Quantizer* and encoded by the *Run-Length Encoder* and the *Windowed Length-Limited Huffman-Type Coder*. Meanwhile, the quantized Y, U and V components are dequantized (reconstructed) and stored in the *Frame Buffer*, to be used as the reconstructed previous frame for encoding the next frame. The decoding process of the PVC-II is similar to the reverse process of the encoder and its block diagram is shown in Fig. 3. The following subsections will depict the function blocks of the PVC-II in details.

3.1 The Format Converter

The *Format Converter* converts the RGB components of pixels to YUV components. After the format conversion, the U and V components are then 4 to 1 subsampled.

Figure 1: The block diagram of the PVC encoder.

Figure 2: The block diagram of the PVC-II encoder.

Figure 3: The block diagram of the PVC-II decoder.

3.2 The Frame Type Controller

The *Frame Type Controller* defines the organization of the GOF, i.e. sets the type of each frame, and controls the parameters of the *Frame Coder Selector*, the *Adaptive Quantizer*, the *Frame Buffer* and the *Windowed Length-Limited Huffman-Type Coder*. There are four types of frames within a GOF: Refresh Frame (R-Frame), the Fine Frame (F-Frame), the Medium Frame (M-Frame) and the Coarse Frame (C-Frame). The R-Frame is also called as the intraframe, and the other frames are classified as the interframes. The use of the R-Frame is for the error control of the communicated video and for the random access of the interactive playback systems, and it is used as the beginning of a GOF. Excluding the R-Frame, the F-Frame has the smallest compression ratio but possesses the best picture quality. Most of the gain of the compression ratio comes from the C-Frame, but the picture quality of the C-Frame is the worst among all types. The picture quality and compression ratio of the M-Frame are in between of those of the F-Frame and the C-Frame. The use of the M-Frame is to tradeoff between the picture quality and the compression ratio. Their descriptions are given below.

Refresh Frame:

In the R-Frame, the *Frame Buffer* is cleared, the Huffman table of the *Windowed Length-Limited Huffman-Type Coder* is reset to a predefined table, the *Frame Coder Selector* is set to the mode of using the intraframe coding, and the quantization steps of the *Adaptive Quantizer* are set to be similar to those in the F-Frame. The type of the first frame in a GOF is always set to be the R-Frame.

Fine Frame:

In the F-Frame, the *Frame Coder Selector* is set to be the mode of using the interframe coding. The quantization steps are set to be the smallest one among all four frame types. Therefore, the picture quality of the F-Frame is the best of all.

Medium Frame:

In the M-Frame, the *Frame Coder Selector* is set to be the mode of using the interframe coding. The quantization steps are set to be larger than those in the F-Frame.

Coarse Frame:

In the C-Frame, the *Frame Coder Selector* is set to be the mode of using the interframe coding. The quantization steps used in the C-Frame are the largest of all, and hence the picture quality of the C-Frame is the worst.

The parameters that control the *Frame Type Controller* can be written as a vector $FT: (GL, FL, ML)$,

where GL indicates the length of the GOF, FL sets the frame distance between two F-Better-Frames (where an F-Better-Frame is defined as an F-Frame or an R-Frame), and ML sets the frame distance between two M-Better-Frames (where an M-Better-Frame is defined as an M-Frame or an F-Frame or an R-Frame). The number of C-Frames between any two other types is equal to $ML - 1$.

3.3 The Intraframe DPCM

When encoding intraframes, each pixel component, Y, U or V, is subtracted from the reconstructed previous pixel component in the same frame. The differences between pixels are coded instead of the pixels themselves. It can simply remove the spatial redundancies among intraframes. In our experiments, the intraframe DPCM can reduce the bitrate of the intraframe data to about 30% of that of the PVC coder.

3.4 The Adaptive Resolution Reducer

When encoding interframes, the adaptive resolution reduction technique is applied to speed up the coding process and to obtain higher compression ratio. The *Adaptive Resolution Reducer* compares the pixel components in the current frame and those in the previous reconstructed frame. If the difference of a pixel component between two frames is less than a threshold T , which will be described in section 3.5, then the next pixel component will be decimated, otherwise the components are coded pixel by pixel. In the other words, the subsample factor in the still background region is two, and that in the moving objects is one. The decimated pixel components between still background region and moving objects will be interpolated back at the decoding side.

Fig. 4 illustrates the process of adaptive resolution reduction. The pixel type S stands for the sampled pixel, D means the decimated pixel, and I indicates the interpolated pixel. Only the sampled pixel component will be coded, and the other pixel components are all eliminated. In this example, the pixel values of the Error Frame are obtained by subtracting the pixel values of the Current Frame from the corresponding pixel values of the Previous Frame. The threshold T is 1 in this example. In pixel 0, the error value (0) is less than the threshold T , which indicates the pixel 0 is within the background area. The encoder encodes the error value of pixel 0 but decimates that of pixel 1. Again, the error value of pixel 2 is still less than T , so the coder codes error value of pixel 2 and omits that of pixel 3. The error value of pixel 4 (28) is larger than T , which makes the encoder switch to moving object mode. In moving object mode, the encoder encodes all the error values in succession until an error value less than T is concurred, so the error values of pixels 4, 5, 6 and 7 are coded.

When coding pixel 8, the coder switches back to background mode again. In this example, the pixel 3 is at the boundary of the moving object and the background area, and is decimated by the encoder. The decoder has no idea about whether the pixel 3 is within the moving object or within the background area in the original video, so the decoder interpolates pixel 2 and pixel 4 to reconstruct pixel 3.

By using the adaptive resolution reduction technique, the picture quality of the PVC-II is slightly worse than that of the PVC, but the coding speed and compression ratio of the PVC-II are much higher than those of the PVC.

PVC-II decoder can automated detect which sampled pixels are in the background region because the sampled pixels are always less than the threshold T . By the same technique, PVC-II decoder can automated detect which sampled pixels are in the moving objects too. Therefore, the decimated and interpolated pixels could be automated reconstructed. No tokens or extra information need to be transmitted.

3.5 The Adaptive Quantizer

The *Adaptive Quantizer* quantizes the Y, U and V components output by the DPCM. The quantizer is controlled by two parameters: the step size S and the dead zone size Z . The function of the *Adaptive Quantizer* is:

$$Q(x) = \begin{cases} \left\lfloor \frac{x+S-Z-1}{S} \right\rfloor & , \text{ if } x \geq Z \\ 0 & , \text{ if } |x| < Z \\ \left\lfloor \frac{x-S+Z+1}{S} \right\rfloor & , \text{ if } x \leq -Z \end{cases} \quad (1)$$

The determination of parameters of the *Adaptive Quantizer* depends on which component to be quantized. Therefore, the parameters to control the *Adaptive Quantizer* can be written as the vector AQ : (YS, US, VS, YZ, UZ, VZ) , where YS, US and VS respectively are the quantization step sizes for quantizing Y, U, and V components and YZ, UZ and VZ are the dead zone sizes for quantizing Y, U and V components, respectively.

3.6 The Run-Length Encoder

The *Run-Length Encoder* represents the quantized values as the run-length tuples $RL(L, V)$, where L indicates the length of zeros in succession, and this zero run is followed by a quantized value V . A special run-length tuple $RL(0, 0)$ is used to indicate the situation of the end of frame.

3.7 The Windowed Length-Limited Huffman-Type Coder

After the quantized signal is encoded as the run-length tuples, these tuples are then represented in more com-

pact form by using Huffman code[9]. In order to speed up the coding process to meet the real-time constraint, the *Windowed Length-Limited Huffman-Type Coder* modifies the *Construct_Codetree*, the *Initial* and the *Update* procedures of [10], which are used to construct and adjust the Huffman codetree, respectively.

The maximum run-length used in the *Run-Length Encoder* is 15 so that 4 bits are required for representing each run-length. The range of a quantized value in the *Adaptive Quantizer* is from -7 to 8 so that 4 bits are needed to represent it, too. In the PVC-II, the two fields of a run-length tuple $RL(L, V)$ are combined as an 8-bit symbol. That is, the alphabet size of the *Modified Windowed Huffman-Kind Encoder* is 256. The concept of the length-limited Huffman code[11][12] is used to simplify the encoding procedure. In order to speed up the process of code tree reconstruction, a Huffman-type version of the *Construction* procedure similar to that used in the MPEG coder[4] is adopted in the PVC-II. The maximum code length of encoded symbols is limited to 12 bits.

The encoding process of the *Windowed Length-Limited Huffman-Type Encoder* is briefly described as follows. Before encoding each symbol, the encoder first checks whether it is time to reconstruct the Huffman codetree. If it is, the encoder calls the *Construct_CodeTree* procedure to reconstruct the Huffman codetree. In the PVC-II, the Huffman codetree is reconstructed for every seven frames and a predefined Huffman codetree is used when encoding intraframes. After the reconstruction process, the next input symbol is encoded by the *Encode* procedure and the code table is adjusted by *Update* procedure. In view of computation speed, the *Update* procedure in *Windowed Length-Limited Huffman-Type Encoder* increases the weight of the encoded symbol only.

By combining the intraframe DPCM and the predefined Huffman codetree, called the fixed-model DPCM technique, the compression ratio of intraframes in the PVC-II is about three times higher than that of the PVC. Therefore, by using the adaptive resolution reduction technique, the compression ratio and the interframe encoding speed in the PVC-II is about 1.4 times higher than those in the PVC.

4 Experimental Results

This section shows several experiment results. In these experiments, the resolution of each image is 128×128 , the pixel format is 15 bits/pixel where the R, G, B fields occupy 5 bits each. The AQ vectors are fixed as $(3, 1, 1, 0, 0, 0)$, $(2, 1, 1, 0, 0, 0)$, $(2, 2, 2, 2, 0, 0)$ and $(2, 1, 1, 2, 1, 1)$ in the R-Frame, the F-Frame, the M-Frame and the C-Frame, respectively. PVC-II is run on a PC/486-33 under MS-DOS 6.0 real-mode operation system with 640×480 , high color (15 bits/pixel)

Pixel Index:	0	1	2	3	4	5	6	7	8	9	10	11
Current Frame:	3	4	3	3	30	34	34	37	3	3	4	4
Previous Frame:	3	3	3	3	2	3	3	3	3	3	4	4
Error Frame:	0	1	0	0	28	31	31	34	0	0	0	0
Pixel Type:	S	D	S	I	S	S	S	S	S	D	S	D
Reconstructed Error Frame:	0	0	0	14	28	31	31	34	0	0	0	0
Reconstructed Frame:	3	3	3	17	30	34	34	37	3	3	4	4

Figure 4: An example of adaptive resolution reduction with $T = 1$.

Functions	Measurements
Net Encoding Time of the PVC-II	31.64 ms/frame
Net Encoding Speed of the PVC-II	31.61 frame/sec
Gross Encoding time of the PVC-II	47.35 ms/frame
Gross Encoding Speed of the PVC-II	21.12 frame/sec
Net Decoding Time of the PVC-II	19.4 ms/frame
Net Decoding Speed of the PVC-II	51.54 frame/sec
Gross Decoding Time of the PVC-II	32.2 ms/frame
Gross Decoding Speed of the PVC-II	31 frame/sec
Net Encoding Speed of the PVC	29 frame/sec
Gross Encoding Speed of the PVC	17 frame/sec
Net Decoding Speed of the PVC	43 frame/sec
Gross Decoding Speed of the PVC	30 frame/sec
Encoding Speed of the MPEG coder	0.45 frame/sec
Decoding Speed of the MPEG coder	6.6 frame/sec
Video-for-Windows (Enc 160×120)	2.2 frame/sec
Video-for-Windows (Dec 160×120)	14 frame/sec

Table 1: The measurements of the PVC-II, the PVC, the MPEG coder and the Video-for-Windows.

SVGA screen mode. The distortion measurement function used in the following experiments is the weighted sum of peak-to-peak signal-to-noise ratio (PSNR) of the RGB components. The distortion function is defined as:

$$PSNR = 10 \times \log_{10} \frac{\sum_{i=1}^{31^2} (\hat{R} - R)^2 + 0.587 \times (\hat{G} - G)^2 + 0.114 \times (\hat{B} - B)^2}{\sum_{i=1}^{31^2} (R^2 + G^2 + B^2)} \quad (2)$$

In order to evaluate the encoding speed of the PVC-II, two time intervals are measured. One is the *gross encoding time* including the frame grabbing time, encoding time, and the time interval for playing back to the display. The other is the *net encoding time* including the encoding time only. Two time intervals are also evaluated for the decoding process of the PVC-II. The performance measurements of the PVC-II and comparison with the PVC, the Video-for-Windows 1.0 and the MPEG coders are shown in Table 1. These results show that the coding speed of the PVC-II is well suited for real-time video applications.

The first set of experiments evaluates the PSNRs and average compression ratios of the PVC-II by changing the value of FL , where the values of GL and ML are set to be 256 and $FL/3$, respectively. Fig. 5 shows the

Figure 5: The (a) average compression ratios, and (b) PSNRs of encoding the salesman and the Chinese girl video sequences.

PSNRs, average compression ratios of encoding the salesman and the Chinese girl sequences. These figures show that the larger the FL is, the lower the average PSNR and the higher the compression ratio will be.

Since for video playback systems, the coding performance of intraframe (R-Frame) is an important issue of a video coder. The second experiment was designed to compare the PVC-II with the PVC by changing the value of GL , where the vector $FT = (GL, 6, 2)$. As shown in Fig. 6, the PVC-II is more suitable than PVC for those video applications.

The third experiment compares the PSNR of the PVC-II to that of the MPEG coder at almost the same compression ratio. As the experimental results shown in Fig. 7, the PSNRs of the PVC-II are as good as those of the MPEG coder for the salesman and the Chinese girl

Figure 6: The average compression ratios of the PVC-II and PVC for encoding the salesman sequence with changing the size of the GOF.

sequences.

In the third experiment, the MPEG coder uses two B-frames between each P-frame pair and adopts logarithmic block-matching algorithm in which the sum of magnitude differences are used as the cost function. The search range for the motion estimation algorithm is $(-7.5)-(7.5)$ pixels for adjacent frame-pairs and $(-15.5)-(15.5)$ pixels for other frame-pairs. The quantizer's factor for P and B frames is twice that of I frames. The coding speed is 0.45 frames/second for MPEG encoding and 6.6 frames/second for decoding.

The fourth experiment compares the PVC-II with the PVC under the same parameters. The results are shown in Fig. 7 and Table 1. Generally speaking, the inter-frame quality of the PVC-II is slightly worse than that of the PVC, but the quality and the compression ratio of intraframes of the PVC-II is much better than those of the PVC.

The final experiment compares the PVC-II with the Video for Windows V1.0 package. The coding speed of the Video for Windows for 160×120 DIB frames is 2.2 and 14 frames/second for encoding and decoding respectively. In this experiment, no audio data and key frames are included in the output file or the AVI file of Video for Windows, and the Microsoft Video 1 coding algorithm is used. The experimental results are also shown in Fig. 7 and Table 1. Although it is unfair to compare PVC-II to Video for Windows because Video for Windows is run under windows system, this experiment is to show that the PVC has the real-time video data encoding capability but other coders have not.

The original salesman video sequence and the sequences encoded by the PVC-II, the PVC, the MPEG coder, and Video for Windows are shown in Fig. 8 (a), and the sequences of the Chinese girl are shown in Fig. 8 (b). The average compression ratios in Fig. 8 (a) and (b) are 30 and 20 respectively.

Figure 7: The PSNRs to compression ratios of salesman, Chinese girl (upper part) and tennis (lower part) sequences.

5 PVC-II for a Video on Demand System: the Popular VOD

In our laboratory the PVC-II has been successfully incorporated into a video on demand prototyping system which is called the Popular VOD. The system overview of the Popular VOD is shown in figure 9. Some interactive playback and communication issues are stated in the following subsections.

5.1 Interactive Playback Issues of Popular VOD

In order to achieve the interactive playback functions such as random access, fast forward playback and reverse playback, of Popular VOD, an index file to indicate the offsets of each R-Frame of the compressed video is created. The use of the index file depends on the interactive playback functions, and is described as follows.

1. Normal playback

It is not necessary to use the index file when the normal playback function is performed. The index file is used only for resynchronization when transmission error occurred, and the resynchronization technique will be described in the next subsection.

2. Fast forward playback

Figure 8: The comparisons of the PVC-II, PVC and MPEG coder for subsequences of salesman (upper) and Chinese girl (lower).

Figure 9: The system overview of the Popular VOD.

There are two methods to perform the fast forward playback function in Popular VOD.

(a) Continuous fast forward playback

Since the decoding speed of the PVC-II is much faster than the encoding speed of the PVC-II. It's capable of decoding and playing back the video two to three times faster in Popular VOD without discarding any frame. In this case, the index file is also used for error control only.

(b) Discontinuous fast forward playback

Another way of performing the fast forward playback function is to decode and display R-Frames only. Since the index file for the offsets of R-Frames has been already created, the video server uses the index file to find the R-Frames of the video and only the R-Frames are sent to the client end. The advantages of this technique are that it can provide high speed fast forward playback and save communication/storage access bandwidth, but the disadvantage is the unsatisfied discontinuous playback.

3. Reverse playback

In reverse playback function the index file is referred in reverse order, and the R-Frames are sent to the client end in reverse sequence too. Only the R-Frames will be played back in this function.

5.2 Communication Issues of Popular VOD

1. Control Flow and Protocols of Popular VOD

When starting up the Popular VOD, the system initializes the network interfaces and sets up the connection to video server first. After the initial process, Popular VOD uses the double buffering technique to play video/audio back continuously. One is the decoding buffer which contains the video/audio ready to be decoded, the other is the receiving buffer which receives the video/audio data sent from video server through the local area network. If there are compressed video data in the decoding buffer, PVC-II decoder decodes the video and the decoded video frame is then displayed on the monitor. When the video frame is decoded, the error control procedure is performed for error detection, and the procedure will be further described later.

When the decoding buffer is empty, Popular VOD moves the data in the receiving buffer to the decoding buffer and a request is sent to the video server to send more data. At this period, the video decoding of PVC-II and data receiving of Popular VOD will progress simultaneously.

2. Error Control of Popular VOD

The error control of Popular VOD is performed both by PVC-II and Popular VOD system and the packet

error and packet lost situations can be detected by the following ways:

(a) Detecting packet loss

In Popular VOD, each packet has a packet sequence number. The VOD client end can detect packet loss by examining the sequence number of each packet.

(b) Detecting the error within the error frame

Because the variable length code is used in PVC-II, a packet lost situation is very likely to make the decoded run-length mismatch with the original one sent by the sender. Such errors can be detected when matching the start field of the next frame.

(c) Detecting the error when rebuilding the Huffman tree.

If the errors are not detected in the first case, they will make the accumulated frequencies of symbols on the PVC-II decoder mismatch with those on the PVC-II encoder. Most of such errors will make the reconstructed Huffman code trees of the encoder and the decoder inconsistent. Most of such inconsistency will lead to a mismatch of run-length between sent data and received data which can be detected in the first case when coding further video frames.

To resynchronize the PVC-II sender and receiver is necessary after the errors are detected. The receiver discards the video frame with error and sends a request to VOD server to retransmit. When the VOD server receives the request, the index file is used to find the nearest previous R-Frame, and retransmit the video started from the R-Frame. When the transmission error occurred, the Popular VOD will not skip any frame of the video.

6 Conclusions

This paper presents a new generation of software-based moving picture coding system: the Popular Video Coder II (PVC-II). The PVC-II is based on the structure of its previous version, the PVC, in which the MC/DCT-based video coder is simplified, and the quantizer and the Huffman coder are modified. Furthermore, the PVC-II improves the performance of the Huffman coder of the PVC. The fixed-model intraframe DPCM and the adaptive resolution reduction techniques are proposed in the PVC-II to obtain higher coding speed and higher compression ratio so as to meet the requirements of real-time multimedia applications. As shown by the experimental results, when the input data is a low resolution and slow motion video, the video quality of the PVC-II is as good

as that of the MPEG video coder and no block, blurring and ripple effects are induced. The coding speed of the PVC-II is much faster than that of the MPEG coder and the video quality of the PVC-II is good enough for commercial video applications, such as video phone, video conferencing and video on demand. By using the PVC-II as the video coder, both the cost and the complexity of developing, maintaining, and installing of video applications can be greatly reduced. In our laboratory, the PVC-II has been successfully incorporated into a video on demand prototyping system with an acceptable video quality.

Acknowledgment

This work is supported by National Science Council, Taiwan, R.O.C., under grants no. NSC83-0408-E-002-003 and no. NSC83-0408-E-002-004.

References

- [1] H.-C. Huang, J.-H. Huang, and J.-L. Wu. Real-time software-based video coder for multimedia communication systems. *ACM Multimedia Systems*, 1:110-119, 1993.
- [2] 1989 CCITT Study Group XV, TD 35. Draft revision of recommendation h.261: Video codec for audiovisual services at p x 64 kbits/s. *Image Communication*, 2:221-239, August 1990.
- [3] CCITT Study Group XV. *Description of Reference Model 8 (RM8)*. Working Party XV/4, Specialists Group on Coding for Visual Telephony, Doc. 525, June 1989.
- [4] *MPEG standard draft ISO-IEC/JTC1 SC29 on 22*, November 1991.
- [5] N. Ahmed, T. Natarajan, and K. Rao. Discrete cosine transform. *IEEE Trans. on Computers*, pages 90-93, January 1987.
- [6] P. Elias. Predictive coding. *IRE Trans. on Information Theory*, pages 16-33, March 1955.
- [7] R. Srinivasan and K. R. Rao. Motion-compensated coder for video-conferencing. *IEEE Trans. Commun.*, COM-35:397-404, March 1987.
- [8] H.-C. Huang and J.-L. Wu. Novel real-time software-based video coding algorithms. *IEEE Trans. on Consumer Electronics*, 39(3):570-580, August 1993.
- [9] D. A. Huffman. A method for the construction of minimum-redundancy codes. *Proc. IRE* 40, 9:1098-1101, September 1952.

- [10] H.-C. Huang and J.-L. Wu. Windowed Huffman coding algorithm with size adaption. *IEE Proc.I*, 140(2):109–113, April 1993.
- [11] L. L. Larmore. Height-restricted optimal binary trees. *SIAM J. Comput.* 16, pages 1115–1123, 1987.
- [12] L. L. Larmore and D. S. Hirschberg. A fast algorithm for optimal length-limited Huffman codes. *Journal of ACM*, 37:464–473, July 1990.
- [13] A. A. Rodriguez and K. Morse. Evaluating video codecs. *IEEE Multimedia Magazine*, 1(3):25–33, Fall 1994.

of Technology, Taipei, Taiwan. Since 1987 he has been with the Department of Computer Science and Information Engineering, National Taiwan University, where he is presently a Professor.

Dr. Wu was the recipient of the 1989 Outstanding Youth Medal of the Republic of China and the Outstanding Research Award sponsored by the National Science Council, from 1987 to 1992.

Dr. Wu has published more than 100 technique and conference papers. His research interests include Neural Networks, VLSI Signal Processing, Parallel Processing, Image Coding, Algorithm Design for DSP, Data Compression, and Multimedia Systems.

Ho-Chao Huang received the B.S. and Ph.D. degrees in the Department of Computer Science and Information Engineering from the National Taiwan University, Taipei, Taiwan, ROC, in 1990 and 1995, respectively.

He is Postdoctoral Fellow of Institute of Information Science at Academia Sinica, Taipei, Taiwan, ROC. His research interests include Video Data Compression, Model-Based Image Coding, Virtual Reality, Computer Vision, Computer Networks, Computer Graphics, Graphics User Interface and Multimedia Systems.

Ja-Ling Wu received the B.S. degree in Electronics Engineering from the Tamkang University, Tamshoei, Taiwan, in 1979, the M.S. and Ph.D. degree in Electrical Engineering from Tatung Institute of Technology in 1981 and 1986, respectively.

From 1986 to 1987 he was an Associate professor of the Electrical Engineering Department at Tatung Institute