A DCT-Domain H.263 Based Video Combiner for Multipoint Continuous Presence Video Conferencing

Da-Jin Shiu, Chia-Chiang Ho, and Ja-Lin Wu, *senior member IEEE*
Communication & Multimedia Lab.,
Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan
wjl@cmlab.csie.ntu.edu.tw

Abstract

This paper proposes an H.263 based DCT-domain video combiner, which is suitable for a multipoint continuous presence videoconference system and supports up to six conferees. Main issues of the H.263 video combiner are discussed. A software-based combiner is implemented and tested for various test sequences. The combined videos have promising quality and the combiner is considered very efficient for practical usage.

1. Introduction

Compared with traditional two-point videoconference, multipoint videoconference is of more practical usage. In a multipoint videoconference system, it usually includes a multipoint control unit (MCU) which receives video signals from each participant and provides a suitable video to be distributed to all participants.

Multipoint videoconference can be in one of the two types: "switched presence" or "continuous presence" [1][2]. In switched presence scheme, MCU transmits only one video signal from a particular conferee, typically the man who is speaking, to every other conferee. The selection of such person can be accomplished through the conference chairman's arbitration, or through monitoring the audio channel activity. ITU-T Recommendation H.231 [3] and H.243 [4] provides more detail of switched presence videoconference.

Switched presence is not an ideal scheme for videoconference since only one conferee can be seen at one time. A better choice is based on the "continuous presence" scheme. In such scheme, a video combiner is included in the MCU, responsible for combining input video signals from the conference participants to generate a combined output video signal. The MCU can then deliver the combined video to each user, who can simultaneously see one or more of the others. Therefore, as stated in [2], a continuous presence conference is more closely emulating an actual in-person conference than its switched presence counterpart. However, since the combined video need higher bandwidth than the original input videos, such a scheme requires an asymmetric connection between each terminal and the MCU. That is, from user's viewpoint, the downloading capacity should be larger than the uploading capacity. Fortunately, the up-to-date communication technologies, such as V.90 modem, cable modem, and ASDL, fit well with the required asymmetric connection property.

Video combining can be achieved by two approaches: the pixel-domain combining and the coded-domain combining. In the pel-domain combing, the compressed video is decoded to pixel domain for combining, and the combined video is encoded again for transporting over network. On the other hand, in the coded-domain combining, the compressed video is partially decoded instead of completely decoded down to the pel-domain. The advantage of the pel-domain approach is its flexibility in allowing different coding methods for different participants, while that of the latter approach is shorter end-to-end delay and lower MCU cost.

Depending on different coding schemes, coded-domain combining can be done by two different approaches: VLC-domain approach and DCT-domain approach [1][2]. In some cases, video combiner only has to process data headers of the compressed data and concatenates the remaining data stream without modification. Since we are mainly dealing with data coded by variable length codes, this approach is referred to VLC-domain approach. For other coding schemes, video combining can be done only after decoding variable-length codes. Such a scheme is referred to DCT-domain approach. Further discussions of the pros and cons of pel-domain, VLC-domain, and DCT-domain approaches can be found in [1][2].

Featuring multipoint, continuous presence, and VLC-domain combining, Sun et al. proposed an GOB-based H.261 video combining system [2]. Three important

0-7695-0253-9/99 $10.00

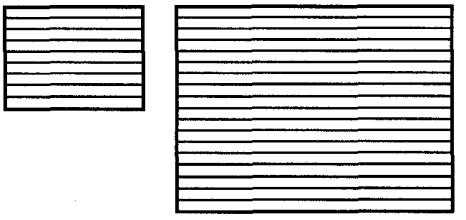

Fig. 1 The GOB structures in H.263 (a) QCIF (b)CIF

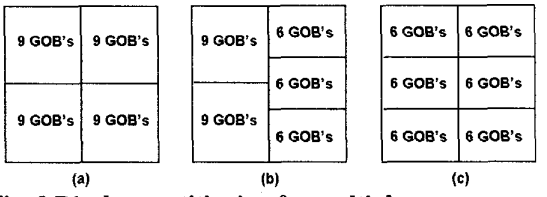

Fig. 2 Display partitioning for multiple users

technical issues: frame rate synchronization, combiner delay accumulation, and potential quality degradation, have been addressed in detail. The viability of a software-based video combiner was also been demonstrated.

This paper proposes an H.263 [6] based DCT-domain video combiner, which is also suitable for a multipoint continuous presence videoconference system. The major challenge of the proposed work comes mainly from the specific GOB structure of the H.263 standard, which is quite difficult to be located in a H.263 bitstream, as compared with its H.261 [5] counterpart.

The remainder of this paper is organized as follows. Section 2 describes the general concept of the proposed H.263 video combiner, and explains why H.263 video combining needs to be implemented in the DCT-domain. Section 3 addresses two main issues of the proposed video combiner. Section 4 deals with the frame synchronization and accumulation delay problems. Section 5 provides experimental results of a software implementation of the proposed combiner. Finally, a summary and future work are presented in section 6.

2. H.263 GOB Structure based DCT-domain Video Combining

The combiner proposed in this paper takes four (up to six) input videos of QCIF (176x144) frame size, and outputs a single combined video of CIF (352x288) frame size. The input videos are encoded in H.263 basic mode, that is, the four optional modes (unrestricted motion vector mode, syntax-based arithmetic coding mode, advanced prediction mode, and PB-frames mode) of H.263 are not used. The combined video is still an H.263

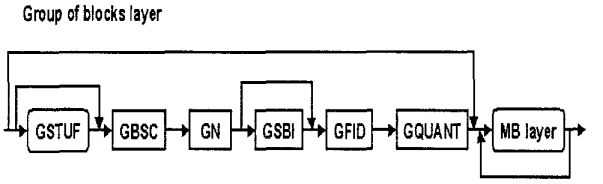

Fig. 3 The GOB Syntax in H.263

compliant bitstream, decodable by any H.263 compliant decoder.

In [2], an H.261 GOB (group of block) is chosen as a basic combining unit. The reason is that each GOB has a clear delimiter (that is, a byte-aligned GOB header) that can be detected by the combiner without having to decode variable length codes. Furthermore, H.261 specifies the same GOB size for both CIF and QCIF frame formats. These characteristics make H.261 based video combiner quite easy to be implemented in the VLC-domain.

H.263 also possesses the GOB structure, but the GOB size is different for CIF and QCIF frame formats. As shown in Fig. 1, the number of GOB's is 9 per QCIF frame, and 18 per CIF frame.

A CIF frame is double-sized in both width and height of a QCIF frame, so it is intuitive to combine four QCIF videos into one CIF video. But sometimes there is a need of a conference with more than four participants. The video combiner proposed in [2] supports at most six users, and we adopt the similar scheme to develop our H.263 based video combiner. Fig. 2 shows the screen partitioning of four, five, and six users. Note that when five or six users are involved, three GOB's of some (three or six, respectively) input QCIF frames must be cut off so that all conferees can be fit in a frame of CIF size. To avoid excessive computation, we can just cut off top three GOB's of an input QCIF frame, when necessary. The reasons are:

(1) Generally, the user will adapt himself or the camera to fit his head in the remained six GOB's.

(2) Motion vectors of macroblocks in the six GOB's have very little probability to point to the cut-off part, so the degradation of output video quality will not be noticeable.

The video combiner we proposed is implemented in the DCT-domain, due to the specific H.263 GOB structure. In an H.263 bitstream, a GOB may have a header or not (see Fig. 3), except for the topmost GOB, which always have no header. This means that not all GOB's have a clear delimiter, as the H.261 case does. To separate neighboring GOB's, variable length decoding is necessary. Even if a GOB has a header, detection of GOB boundary may still need variable length decoding! The reason is that the header of a GOB needs not to be byte-aligned in H.263. According to the above discussion, variable length decoding is unavoidable in an H.263 based video combining system. That is, the proposed

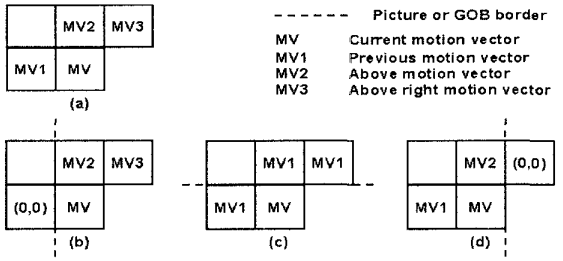

Fig. 4 Motion vector prediction in H.263

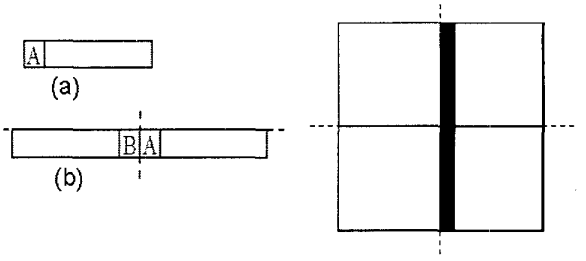

Fig. 5 Inconsistent motion vector prediction

Fig. 6 The minimum set of motion vectors to be re-encoded

combiner must be realized in the DCT-domain, as compared to VLC-domain approach adopted in [2]. Fortunately, it is convincible that VLC decoding and encoding consumes very little computation power, as compared with other modules of a video coding system (such as DCT/IDCT and motion estimation/compensation). In other words, the proposed DCT-domain video combiner still retains the low-cost advantage.

3. Main issues of H.263 Video Combiner

The description of section 2 makes the main action of the proposed combiner very clear: merging two QCIF GOB's from two input videos into a CIF GOB, pair by pair. Due to the specific GOB structure in H.263, there are two main issues must be considered so that GOB merging can be done successfully:

1. Avoiding inconsistent motion vector prediction.
2. Avoiding inconsistent quantization scale adjusting.

The following two sub-sections discuss these two issues and provide reasonable solutions.

3.1. Avoiding Incorrect Motion Vector Prediction

In H.263, the motion vector (abbreviated as MV in the following discussions) is differential coded. That is, when encoding an MV, a prediction value is first generated, and only the difference between current MV and the prediction value is coded by using variable length codes. As shown in Fig. 4(a), the median value of three candidate predictors is treated as the prediction value. Moreover, H.263 standard says:

In the special cases at the borders of the current GOB or picture, the following decision rules are applied in increasing order:

1): When the corresponding macroblock was coded in INTRA mode (if not in PB-pictures mode) or was not coded (COD = 1), the candidate predictor is set to zero.

2): The candidate predictor MV1 is set to zero if the corresponding macroblock is outside the picture (at the left side). See Fig. 4(b).

3): The candidate predictors MV2 and MV3 are set to MV1 if the corresponding macroblocks are outside the picture (at the top) or outside the GOB (at the top), if the GOB header of the current GOB is non-empty. See Fig. 4(c).

4): The candidate predictor MV3 is set to zero if the corresponding macroblock is outside the picture (at the right side). See Fig. 4(d).

Note that rule 3 implies that when a GOB has a header, or it's the topmost GOB in a picture, each MV (except the first one) in this GOB only takes the left neighboring MV as its predictor. This property is very useful for solving the inconsistent MV prediction problem discussed below.

The proposed combiner has to merge GOB's pair by pair. By examining the above rules, it can be found that MV's of some macroblocks in input video sequences must be re-encoded properly, otherwise, incorrect motion compensation will occur when decoding the combined video sequence. For example, the MV of macroblock *A* in Fig. 5(a) has a (0,0) predictor (here we assume each GOB's has a header), but after merging GOB's, it will have a new predictor comes from macroblock *B*, as showed in Fig. 5(b). If the MV of macroblock *B* is not (0,0), such inconsistent MV predictor will result in motion compensation error when decoding macroblock *A* in the combined video sequence. Furthermore, such motion compensation error may propagate to following macroblocks in the same GOB.

To solve this problem efficiently, we can simply restrict every GOB's in the input videos to have a header. Under such restriction, only the leftmost macroblocks of right-side QCIF GOB's has to do MV re-encoding (Fig. 6). However, this method requires that all involved H.263 terminals can be negotiated to add headers for all GOB's. But this may not always be achievable. In general cases (without the above restriction), the GOB's in an input video sequence may all have a header, all have no header, or mixture of both. As for the output video sequence, we choose to equip all GOB's with a header. Though adding headers need a slightly augmented bandwidth, such a choice makes bitstream packeting (when transmitting over networks) and error detection/concealment (when

decoding by terminals) more easily and efficiently.

We now describe a general method to solve the inconsistent MV prediction problem. An MV buffer with 22 entries is used to store MV's of the previous coded CIF GOB, for the sake of motion re-encoding. When composing a CIF frame, the following procedures are applied to accomplish proper MV re-encoding:

Step 1: For the first two QCIF GOB to be merged (into the topmost CIF GOB), only the MV of the leftmost macroblock of the right-side QCIF GOB is re-encoded. Store all 22 MV's into the MV buffer. Continue step (2) with the next two QCIF GOB's.

Step 2: If the left-side QCIF GOB is coded with a GOB header, no MV needs to be re-encoded; else all MV's are re-encoded. Store all 11 MV's.

Step 3: If the right-side QCIF GOB is coded with a GOB header, only the MV of the leftmost macroblock is re-encoded; else all MV's are re-encoded. Store all 11 MV's.

Step 4: If all QCIF GOB's are combined, go to step 1 for encoding the next frame; else go to step 2 for merging the next two QCIF GOB's.

3.2. Avoiding Inconsistent Quantization Scale Adjusting

In H.263, there are three ways to change quantization scales: PQUANT field in the picture layer, GQUANT field in the GOB layer, and DQUANT filed in the macroblock layer. When encoding a macroblock, the last quantization scale set by any of these three ways will be applied to quantize the DCT coefficients of current macroblock. Note that PQUANT and GQUANT specify the quantization scales directly, while DQUANT can only decrease or increase the quantization scale by at most 2.

When combining two QCIF GOB's, the quantization scale of the last macroblock of the left-side GOB may be different from that of the first macroblock of the right-side GOB. If we just concatenate the two QCIF GOB's, such inconsistency of quantization scale will result in incorrect inverse quantization when the combined video is being decoded.

To fix this problem, it's obvious that only the DQUANT field can be used. If the difference of quantization scale between each of the key macroblock pair is always not larger than two, we can just modify the DQUANT field of the latter macroblock to fix the problem. Unfortunately, this is not always the case.

The simplest way to solve this problem thoroughly is forcing all input video sequences compromise with a common quantization scale for all macroblocks. That is, we make the difference between each of the key macroblock pair always be zero. This can be done during the handshaking step in the initialization of a conference. However, this method may not be reasonable, especially when we prefer a constant bit-rate environment. One possible solution is to do quantization scale adjusting through DQUANT field more than once, until the quantization scale becomes consistent with the right-side QCIF GOB. This may sacrifice video quality. But if the "difference is larger than 2" situation is rare, the quality degeneration will be tolerable. This assumption is correct if (1) the inputs are typical head-and-shoulder video sequences, (2) they compromise with a common constant bit-rate constraint and (3) the backgrounds are clear.

4. Frame Synchronization

Different terminals equipped with H.263 video coding capabilities may support different frame-rates. A terminal cannot encode or decode a video sequence that exceeds its frame-rate capability. So in a video conference environment, all the terminals have to operate at a common frame rate capability which is supported by all of them. However, this does not means all terminals will produce video sequences with the same frame rate all the time, in other words, the time interval between successive video frames may be variable. So frame synchronization problem occurs, and it means we need a procedure to synchronize all input video frames so that no frame is skipped in the output video sequence. To solve this problem, Sun et al. [2] have proposed a TR (temporal reference) re-mapping method. Since H.263 syntax possesses similar TR field, our video combiner can use the same scheme to achieve frame synchronization.

TR is a necessary field defined in the H.263 picture layer header. It is an 8-bit number, which can have 256 possible values. The following procedures perform the re-mapping of TR:

(1) Convert each incoming TR sequence by first resetting the first available TR to "0" and then the subsequent TR's are offset by the original TR using modulo 256 arithmetic. That is,

$$TR_0' = 0$$

$$TR_i' = (TR_i - TR_0) \bmod 256, \quad i > 0$$

where TR_i' is the shifted TR from the original TR_i

(2) The shifted TR of each input frame is then mapped according to the following equation:

$$TR_i'' = \lfloor TR_i' / f_{inc} \rfloor \times f_{inc},$$

where f_{inc} is the minimum allowable frame increment for the common frame rate capability

By re-mapping each TR in each frame of the input videos, the output frames are formed by combining GOB's from various inputs that have the same mapped TR number. For those GOB's which are not available

Fig. 7

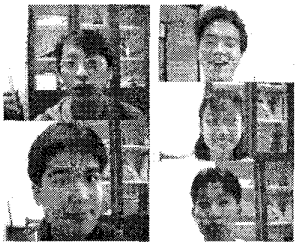

Fig. 8

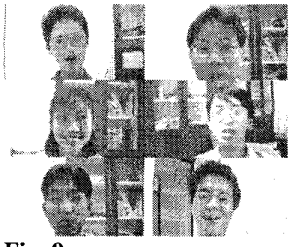

Fig. 9

(from input videos) for a particular TR of the combined video, we can just set the COD field of related macroblocks to 1, which will instruct the decoder to copy macroblock data from the previous decoded picture. Further information about TR re-mapping can be found in [2].

We conclude this section by presenting the high-level algorithm of the proposed video combiner:

Step 1: Do initialization work.

Step 2: For each QCIF input frame, perform TR mapping.

Step 3: Modify the picture layer header to represent a CIF picture header.

Step 4: Get two GOB's from related QCIF bitstream, combine them with GOB header modification and necessary re-encoding of motion vectors and quantization scale adjusting.

Step 5: If all 18 pairs of GOB's have been combined, continue; else go to step 4.

Step 6: If users choose to break current session, stop; else go to step 2.

5. Experimental Results

The proposed video combiner is implemented and tested on a Pentium-166 PC platform, with Microsoft Windows 95 operation system. The test sequences come from (1) typical test sequences, such as "Miss America", "Grandma", "Salesman", etc., and (2) live sequences captured and encoded at our laboratory. Fig. 7, Fig. 8, and Fig. 9 show decoded pictures of the conference snapshots with four, five, and six conferees, respectively. Note that when with five or six conferees, some sequences are previously shifted and re-encoded so that the head-and-shoulder portions are located at the bottom six QCIF GOB's.

6. Summary

This paper proposes an H.263 based DCT-domain video combiner, which is suitable for a multipoint continuous presence videoconference system and supports up to six conferees. Two main issues of the proposed video combiner are discussed and reasonable solutions are provided. We implement a software-based combiner and test it for various test sequences. This implementation is considered very efficient. If invoked H.263 terminals can be restricted to have some preferred properties, the combined video can be of no quality loss. However, if no restriction is put on the invoked H.263 terminals, the proposed video combiner can still provide combined video of compromising quality.

Our future work is to implement the combiner under a real networking environment.

7. References

[1] S. M. Lei, T. C. Chen, and M. T. Sun, "Video bridging based on H.261 Standard," *IEEE Trans. Circuits and System for Video Tech.*, vol. 4, Aug. 1994, pp. 425-437.

[2] M. T. Sun, A. C. Loui, T. C. Chen, "Coded-Domain Video Combiner for Multipoint Continuous Presence Video Conferencing," *IEEE Trans. Circuits and System for Video Tech.*, vol. 7, No. 6, Dec. 1997, pp. 855-863.

[3] ITU-T Study Group XV—Recommendation H.231, "Multipoint control units for audiovisual systems using digital channels up to 1920 kbit/s," Mar. 1993.

[4] ITU-T Study Group XV—Recommendation H.243, "Procedures for establishing communication between three or more audiovisual terminals using digital channels up to 2 Mbit/s," Mar. 1993.

[5] ITU-T Study Group XV—Recommendation H.261, "Video codecs for audiovisual services at p * 64 kb/s," Mar. 1993.

[6] ITU-T Study Group XV—Recommendation H.263, "Video coding for low bit rate communication," Mar. 1996.