Extraction of Desired Partial Block of a Frame from MPEG Video Stream

Sandeep Kumar
Division of Computer Engineering
Netaji Subhas Institute of Technology
Sector-3, Dwarka, New Delhi – 110 078 India

Satish Chand
Division of Computer Engineering
Netaji Subhas Institute of Technology
Sector-3, Dwarka, New Delhi – 110 078 India

ABSTRACT
In MPEG video encoding, the next frame is encoded based on the previous frame. For forward replay of the video, extraction of data of the next frame from the current frame is a trivial process. However, for backward replay, it may not always be possible to extract data of the next (previous) frame completely from the currently displayed frame. In this paper, we propose a method for extracting unpredictable portion of a frame from MPEG video. This helps in extracting the desired portion rather than decoding the entire frame.

Categories and Subject Descriptors
I.4.0 [**Image Processing and Computer Vision**]: General – *Image displays, Image processing software.*

General Terms
Algorithms, Design.

Keywords
Video Streaming, MPEG Video

1. INTRODUCTION
The MPEG [1]-[4] is one of the most widely acceptable standards for video encoding in which a video has three types of frames, namely, I-, P-, and B-frames. The I-frame is independently encoded. The P-frame is encoded based on previous I- or P-frame and the B-frame is encoded using previous and/or next (I or P) frame. The I-frames occur periodically. Between two I-frames, there is a well defined combination of P- and B-frames. The frames starting from an I-frame up to the next I-frame constitute a group of pictures (GOP). A particular sequence of I-, P-, and B-frames is IBBPBBPBBPBBI… comprising 12 frame in a GOP. For encoding a frame, it is divided into macroblocks of 16x16 pixels. Each macroblock is further divided into blocks of 8x8 pixels. To carry out I-frame coding, each block is transformed using Discrete Cosine Transform (DCT). First element of the block is called DC coefficient and the remaining are called AC coefficients. These DCT coefficients are then quantized. The DC coefficients are differentially coded with respect to the DC coefficient of the previous block. AC coefficients are run-length encoded and then Huffman coded. Coding of P-frame is done with reference to the previous I- or P-frame and hence called inter-frame coded. Inter-frame coding exploits the temporal redundancy which occurs in consecutive frames of a scene. Each macroblock of the frame is compared to the respective macroblock and its neighbors in the reference frame (previous I- or P-frame). The best matched macroblock is taken into consideration. This process is called Motion Estimation [5]. The offset of the best matched macroblock from the current macroblock is recorded as its motion vector. This is the horizontal and vertical length of the offset. The difference between both macroblocks is found and called as prediction errors. These prediction errors are coded in the same way as the I-frame coding. B-frame uses both past and future frames as reference frame. If it uses past frame or future frame as reference frame, then its coding is same as that of the P-frame. If it refers to both past and future frames, then average of both prediction errors is taken and both of the motion vectors are recorded and coded. Coding of prediction errors is same as in the P-frames coding.

While decoding a P-frame, we need previous frame which in turn depends on the previous to previous frame and so on. This process continues till the last I-frame. In order to find a small portion of a frame, we need to decode the desired frame completely and for that we need to decode the previous frame completely up to the last I-frame. Many applications may require partial data of a frame, e.g., when the transmitting channel is error prone and data gets corrupted or reverse play of MPEG video. To extract even very small portion of a frame, we need to decode the entire frame. This process is not only inefficient but it wastes important resources too. Therefore, a new mechanism is required so that the required portion of frame is decoded rather than decoding the entire frame. In this paper, we propose a new algorithm for extracting the desired partial block of the frame from MPEG video stream by only decoding the required portion of the frame.

The rest of the paper is organized into four sections. In Section 2, we propose a method for extracting partial block from the MPEG stream. In section 3, we present the results. Finally, in Section 4 we conclude the paper and discuss the further scope of research.

2. BLOCK FINDING METHOD
The client requests a block from the server using the coordinates of the upper left corner of the block (a, b), its width (w), height (h) and the corresponding frame number (n). On receiving this request the server invokes FindBlock algorithm to find the requested block out of the MPEG stream. The first step is to find out the macroblock that contains this requested block. The requested block may fall in many macroblocks. We first spilt the requested block into separate parts such that each part falls in a single macroblock. Falling of a block in different macroblocks can be of anyone of the four scenarios as shown in Figure 1.

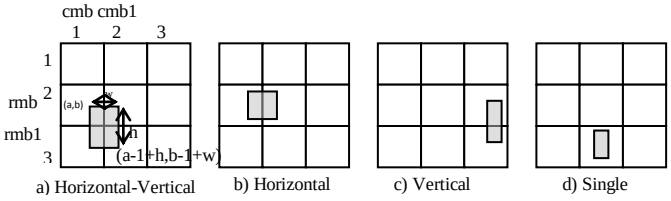


Figure 1. Four different Scenario of a block falling in multiple macroblocks

First we need to find the scenario out of above four scenarios for the requested block. This is done by finding the row & column numbers of the macroblocks that contain the upper left and lower right corners of the requested block. Let rmb & cmb denote row & column number of the macroblock that contains the upper left corner (a, b) and rmb1 & cmb1 denote row & column number of the macroblock that contains the lower right corner (a-1+h, b-1+w). If rmb and rmb1 are different and cmb and cmb1 are also different, then it is Horizontal-Vertical scenario. In this scenario, the block is divided into four separate parts and for each part the algorithm FindBlock is called recursively. If rmb and rmb1 are same but cmb and cmb1 are different, then it is Horizontal scenario. In this scenario, the block is divided into two separate parts horizontally. If rmb and rmb1 are different but cmb and cmb1 are equal, then it is vertical scenario and block is divided into two parts vertically. For each part obtained the same algorithm FindBlock is called recursively. Lastly, if rmb & rmb1 are equal and cmb & cmb1 are also equal, then it falls in a single macroblock and no division is required. In this case, we check the type of the macroblock and decode it. For a P-frame, the decoded information would be the prediction error of the macroblock that is to be added to the corresponding motion compensated macroblock of the previous frame as per the following equation:

$$MB^{n}_{(rmb,cmb)} = MCMB^{n-1}_{(rmb,cmb)}(mv^{n}_{(rmb,cmb)}) + e^{n}_{(rmb,cmb)}$$

where

$MB^{n}_{(rmb,cmb)}$ signifies macroblock at (rmb, cmb) position in the nth frame,

$MCMB^{n-1}_{(rmb,cmb)}$ is motion compensated macroblock at (rmb, cmb) in (n-1)th frame,

$mv^{n}_{(rmb,cmb)}$ is motion vector of the macroblock at (rmb, cmb) position in nth frame,

$e^{n}_{(rmb,cmb)}$ is prediction errors of the macroblock at (rmb, cmb) position in nth frame.

The algorithm FindBlock is called recursively for the previous frame with the coordinates obtained by adding the motion vectors to the coordinates of the requested block. The decoded prediction error is added to the block data returned by the FindBlock algorithm. For an I-frame, the decoded data is the frame data and it is returned by the FindBlock algorithm. The above process is pictorially shown in Figure 2. Pseudo code of the algorithm is given bellow.

[Block] Algorithm FindBlock(a, b, w, h, n)

Begin

Calculate rmb as ceil(a/16).

Calculate cmb as ceil(b/16).

Calculate rmb1 as ceil((a-1+h)/16).

Calculate cmb1 as ceil((b-1+w)/16).

If (rmb=rmb1) and (cmb=cmb1)

Decode the (rmb, cmb) macroblock of nth frame and save it in mb.

Calculate x as (a-((rmb-1)*16)).

Calculate y as (b-((cmb-1)*16)).

If nth frame is an I-frame

Set block as the mb(x to x-1+h, y to y-1+w).

Else

Set error as the mb(x to x-1+h, y to y-1+w).

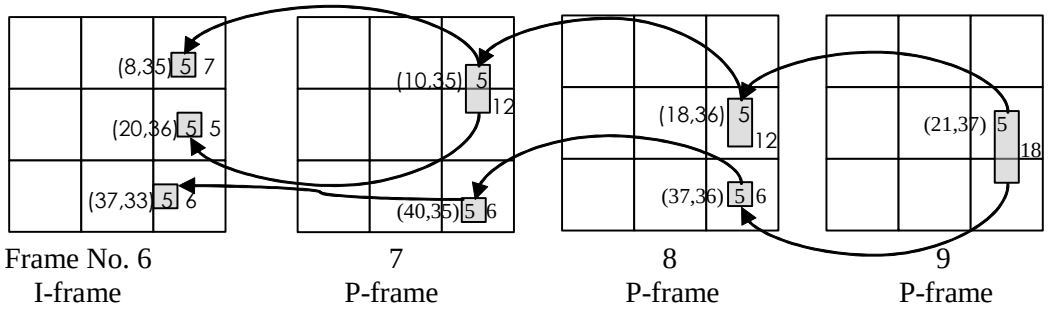


Figure 2. Block Finding Process

Set mvx as the x motion vector of (rmb, cmb) macroblock of nth frame

Set mvy as the y motion vector of (rmb, cmb) macroblock of nth frame

Update a with a + mvx.

Update b with b + mvy.

Set block as the (error + FindBlock(a, b, w, h, n-1)).

End if

Else if (rmb=rmb1) and (cmb!=cmb1)

Set w1 as (cmb*16)-(b-1)

Set w2 as (w-w1).

Get the block1 by calling FindBlock(a, b, w1, h, n).

Get the block2 by calling FindBlock(a, (cmb*16)+1, w2, h, n).

Set block(1to h,1 to w1) as block1.

Set block(1 to h,w1+1 to w) as block2.

Else if (rmb!=rmb1) and (cmb=cmb1)

Set h1 as ((rmb*16)-(a-1))

Set h2 as (h-h1).

Get the block1 by calling FindBlock(a, b, w, h1, n).

Get the block2 by calling FindBlock((rmb*16)+1, b, w, h2, n).

Set block(1 to h1,1 to w) as block1.

Set block(h1+1 to h,1 to w) as block2.

Else

Set w1 as ((cmb*16)-(b-1))

Set w2 as (w-w1).

Set h1 as ((rmb*16)-(a-1)).

Set h2 as (h-h1).

Get the block1 by calling FindBlock(a, b, w1, h1, n).

Get the block2 by calling FindBlock(a, (cmb*16)+1, w2, h1, n).

Get the block3 by calling FindBlock((rmb*16)+1, b, w1, h2, n).

Get the block4 by calling FindBlock((rmb*16)+1, (cmb*16)+1, w2, h2, n).

Set block(1 to h1,1 to w1) as block1.

Set block(1 to h1,w1+1 to w) as block2.

Set block(h1+1 to h,1 to w1) as block3.

Set block(h1+1 to h,w1+1 to w) as block4.

End if

End Algorithm FindBlock

The limitation of algorithm is that we have to decode the complete macroblock in spatial domain to get a small portion out of it. Still this process is better because we do not need to decode the complete frame; only the related macroblock needs to be decoded.

3. RESULTS

To evaluate the practical aspect of the proposed algorithm, we have performed simulations on three video sequences Clarie, Carphone, and Football. These video sequences have been taken from [6]. These video sequences are standard sequences and have been used in literature for video processing research. These video sequences have been encoded and decoded as given in [7]. Figures 3(a)-5(a) show the partially predicted frames which we obtained during our reverse play experiments [8]. In these figures, the black region corresponds to predicted pixels and white region signifies unpredicted pixels. These small portions of unpredicted pixels are to be found from MPEG stream. Figures 3(b)-5(b) show the completely constructed frames in which unpredicted pixels have been obtained using our algorithm

4. CONCLUSION AND FUTURE SCOPE

In order to find a small portion of a frame, we need to decode the desired frame completely and for that we need to decode the previous frame completely up to the last I-frame. To extract even very small portion of frame, we need to decode the entire frame. In this paper we have discussed an algorithm to find the desired portion of a frame from the MPEG video stream without decoding the entire frame. Extraction of the partial frame data is required in many applications such as reverse play, transmission of data using error prone channels. Currently this algorithm decodes a macroblock in spatial domain. Its performance can be further improved by doing this in compressed domain.

5. REFERENCES

[1] Mohammed Ghanbari, Video Coding – An introduction to standard codecs, IEE, 1999

[2] Fred Halsall, "Multimedia Communications: Applications, Networks, Protocols and Standards", Addison-Wesley, 2000.

[3] D.L. Gall, "MPEG: A Video Compression Standard for Multimedia Applications", Communications of the ACM, Vol.34 No. 4, pp. 46-58, April 1991.

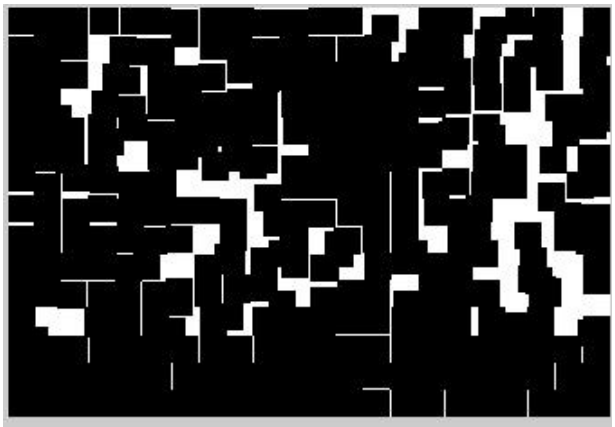
[4] "MPEG-1 Video Codec", http://www.cmlab.csie.ntu.edu.tw/cml/dsp/training/coding/mpeg1/index.html

[5] "Motion Estimation in MPEG I", http://www.cmlab.csie.ntu.edu.tw/cml/dsp/training/coding/motion/me3-1.html

[6] http://media.xiph.org/video/derf/

[7] Steve Hoelzer, "MPEG-2 overview and MATLAB codec project", April 18, 2005, http://www.cs.cf.ac.uk/Dave/Multimedia/Lecture_Examples/Compression/mpegproj/

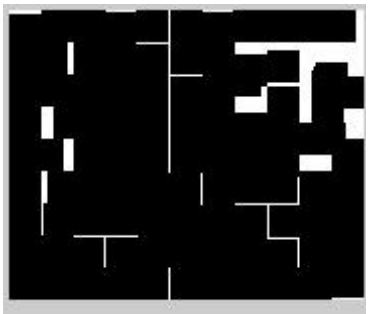
[8] Sandeep Kumar, Satish Chand, "A New Method for Reverse Prediction of MPEG frames in Video Streaming," IEEE International Advance Computing Conference, 2009. pp. 1119-1123, 6-7 March 2009



a) Reverse predicted frame

Figure 3. P124 frame of the football sequence encoded at the quality scale=31

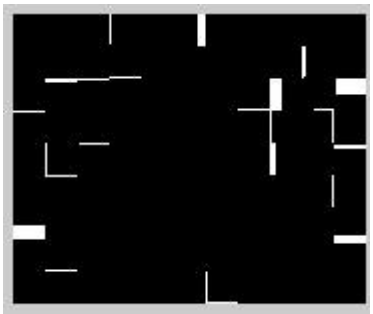
b) Completely constructed frame



a) Reverse predicted frame

b) Completely constructed frame

Figure 4. P189 frame of the Carphone sequence encoded at quality scale=16



a) Reverse predicted frame

b) Completely constructed frame

Figure 5. P244 frame of the Claire sequence encoded at quality scale=16