Pakistan Journal of Information and Technology 2 (2): 154-158, 2003
ISSN 1682-6027

Implementation of Matrix Folding in Multimedia Transmission

Abdelfatah Aref Yahya and Ayman Mahmoud Abdalla
Faculty of Science, Al-Zaytoonah University of Jordan,
P.O. Box 130, Amman, Jordan 11733

Abstract: We designed a new algorithm, called matrix folding, to reduce the size of MPEG video files. This technique can also be applied to other video and non-video formats. It usually reduces file size by half and it is easily reversible without any degradation of picture quality. We built a simulator that employs matrix folding to transmit real-time video from source to destination using various bandwidths. Empirical results showed that matrix folding increased efficiency, especially with low-bandwidth transmission.

Key words: Video compression, video transmission, matrix folding

Introduction

In numerous multimedia systems, data of both discrete and continuous media are transmitted in the form of individual units (packets). Packets are sent from one system component (source) to another (destination). A sequence of individual packets transmitted in a time-dependent fashion is called a data stream (Steinmetz and Nahrstedt, 1995). Streaming, in the context of communication, implies that the audio or video file is transmitted while it is being created. The receiving end converts the file into continuous video (Serber *et al.*, 1996).

Synchronous transmission mode requires a maximum end-to-end delay for each packet in a data stream. A compressed-video data-stream is weakly regular, since the amount of data varies periodically with time. Typically, individual images are coded and compressed as individuals and then combined into whole units that represent relatively large packets inside the data stream. These image packets are periodically transmitted, e.g., every two seconds. Additional packets, containing information about the difference between each two consecutive compressed images, are sent after each image packet (Steinmetz and Nahrstedt, 1995).

The basic component of MPEG bit-stream is known as the elementary stream (ES) and it contains a single type of (usually compressed) signal (Chiariglione, 2000). Each ES is input into an MPEG-2 compressor, which accumulates the data into a stream of Packetized Elementary Stream (PES) packets (Fairhurst, 2001). The PESs are combined to form a program or transport stream. A PES packet may be a fixed-size or a variable-size block, with up to 65536 bytes per block and includes a six-byte protocol header. A PES is usually organized to contain an integral number of ES access units (Fairhurst, 2001).

Each video in MPEG format is a one-dimensional array (vector) of Huffman code, which is a compressed version of a given video. We apply the folding algorithm to this compressed vector.

In MPEG-2, each vector value represents the pixel brightness with an integer ranging from 0 to 255, where 0 represents no color and 255 represents full color.

The transmission latency of a network is the delay between the occurrences of two events. Latency is commonly measured with end-to-end trip time, which is the time needed for a packet to travel from source to destination (Dallas, 2001). In communication networks, propagation time is a significant part of end-to-end latency. Delays of 20 milliseconds are not easily detectable, but greater delays are more troublesome. Connectionless networks, such as the Internet, attempt to use throughput capacity more efficiently, than trying to minimize latency. A channel with high throughput can move large quantities of data rapidly, but the first bit of data can never arrive faster than permitted by the latency (Steinke, 2001).

In the study, we will examine the effect of the folding-unfolding algorithm on the performance of networks, where improving performance will require reducing the latency of the individual flows. The folding algorithm can achieve a reducing of 50% of movie size, which is reflected directly on the time needed for transmission. Our
experiments will show that the time required for folding and unfolding movie files (or frames), plus transmission time, is less than the transmission time of the original movie file (or frames).

Algorithm for matrix folding and unfolding

The matrix folding operation merges each pair of elements in the MPEG vector into one element using the equation:

$$m_i = 256\ e_i + e_{(n-i+1)}$$

Where:

m_i is the merged element and e_1, e_2, ... e_n are the brightness values in the Huffman vector.

For each element in the first half of the movie vector, the algorithm multiplies the element e_i by 256 and then adds the result to element $e_{(n-i+1)}$. This multiplication operation may be replaced with an 8-bits shift to the left and the addition operation may be replaced with a bitwise-OR operation. The results are stored in the new vector m, whose length is half the length of the original vector. To obtain the original vector e from vector m, use the unfolding equations:

$$e_i = m_i/256 \text{ and}$$
$$e_{(n-i+1)} = m_i \text{ MOD } 256$$

where the above two equations produce integer quotient remainder, respectively, of dividing an element from vector m by 256. These two operations may be replaced with equivalent bitwise operations. It can be mathematically shown that these two equations produce the original values of vector e, thus showing that this method produces no loss of information.

Since folding and unfolding perform a constant number of arithmetic operations for each entry in the input vector, the time required for folding and unfolding should remain constant for each entry. Therefore, each folding and unfolding operation will be linear in time, i.e., O(n), with

respect to the number of entries, n, in the input vector.

The reduction in size results from using the extra space available in each entry in vector e. Since MPEG and many other video-formats use 16 bits to represent each entry, we use eight bits to store the color information and the extra eight bits are used for folding. This method was successful in several video formats, as demonstrated by our experiments in the next section. Furthermore, this compression method produces no degradation in image quality since there is no loss of information from the original MPEG or other video format file (Yahya *et al.,* 2003).

Results and Discussion

Our experimental results were obtained by implementing the folding-unfolding algorithm to video files of different types and various sizes and transmitting the different files within a local area network. We executed the folding algorithm using MATLAB 6.1 under the Windows-XP operating system on a Pentium-IV PC with a 1600-MHZ CPU and a 256-Mbyte RAM. An MPEG converter was used to read MPEG files, frame-by-frame, into a matrix that can be read by MATLAB, where the folding algorithm was applied and then the result was stored in a binary file. Other video formats were handled similarly. To view the original video, the binary file is unfolded and the resulting matrix is automatically produced in MPEG (or other video format).

The best throughput of a Basic-ISDN channel is 64 Kbit sec^{-1}, so transferring a 100-Kbyte file, for example, requires a delay time of 12.5 seconds (without folding). As expected in the previous section, folding reduced file size by 50%. Folding time (F_T) required an average of 2.23 microseconds per byte. Unfolding time (U_T) required an average of 1.75 microseconds per byte (of original file size). Therefore, transmission delay for a 100-Kbyte file can be computed as follows:

Original Size Before Folding = S = 100 Kbytes
Folded Size = 50 Kbytes
$F_T = (2.23\times10^{-6})\times(100\times2^{10}) = 0.22$ seconds
Delay Time After Folding $= F_D = (50\times8)/64 = 6.25$ seconds
$U_T = (1.75\times10^{-6})\times(100\times2^{10}) = 0.179$ seconds
Total delay (D_{total}) $= F_T+F_D+U_T = 6.645$ seconds

Consequently, the reduction in delay for this example is 0.4684 or 46.84%. This reduction ratio (R) does not depend on the file size, but it depends on the throughput (P) of the channel. The general equation for R can be derived as follows, where a and b denote the times required for folding and unfolding per byte, respectively and the data-size measurement unit is the bit:

Delay time without folding = $D_{orig} = S/P$
$F_T = a\times S$
$F_D = (S/2)/P$
$U_T = b\times S$
$D_{total} = S\times(a+b+(2\times P)^{-1})$
$R = 1-D_{Total}/D_{orig} = 0.5-P\times(a+b)$

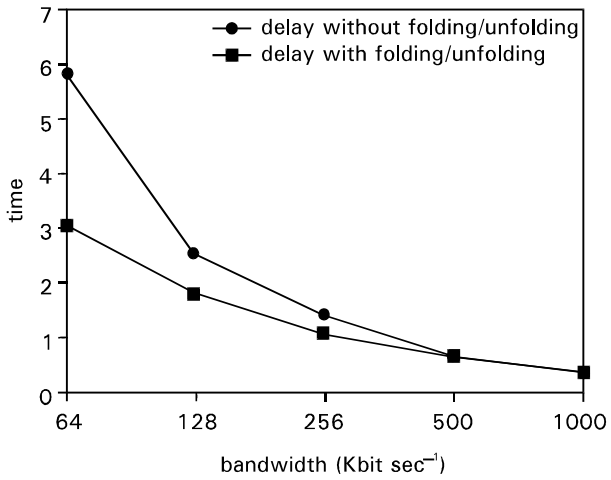

Fig. 1: Time (seconds) needed to transmit 500 movies (24523 bytes) with and without folding-unfolding

This shows that the percentage of reduction in delay time is counter-proportional to the throughout of the channel. As seen in Fig. 1, transmission time decreases as the bandwidth increases. The folding and unfolding times per byte remain constant for the same PC.

Employing the strategy of folding and unfolding, we built a simulator that folds videos transmits them and unfolds the received videos. This simulator shows each movie in two separate windows. The first window shows a real-time movie transmitted without folding-unfolding and the other window shows the same real time movie transmitted with folding by the sender and unfolding by the receiver. The simulator allows the user to set input variables and to display the following output variables:

Folding time (sender)
Unfolding time (receiver)
Transmission time with folding-unfolding
Transmission time without folding-unfolding
Current transmission rate (frames per second) and
Transmission progress in real time

We ran the simulator for bandwidths of 64, 128, 256, 500 and 1000 Kbits sec^{-1} with 500 random-size movies of different types, ranging from one Kbyte to 100 Kbyte. The total size of these files was 24523 bytes. We sat the inter-arrival time (interval between movies) to 0.5 second. We registered the times required for transmitting these files with and without folding-unfolding. Fig. 1 illustrates the relationship between transmission time with and without folding-unfolding. It can be observed that the folding-unfolding strategy increased transmission performance, where this effect becomes greater as the bandwidth becomes lower.

We presented an algorithm that reduces the sizes of files of different types by 50% and we used it to benefit video transmission. Our experiments showed that employing this folding-unfolding algorithm increased transmission efficiency, especially in low bandwidth networks.

References

Dallas, Maxim, 2001. Video Basics. Retrieved from the World Wide Web: http://www.maxim-ic.com/appnotes/.cfm/appnote_number/734.

Fairhurst, Gorry, 2001. MPEG-2 Digital Video. Retrieved from the World Wide Web: http://www.erg.abdn.ac.uk/public_html/research/furture-net/digital-video/.

Chiariglione, Leonardo, 2000. MPEG-2 Generic Coding of Moving Pictures and Associated Audio Information. International organization for Standardisatoin: ISO/IEC JTC1/SC29/WG11.Retrieved from the World Wide Web: http://www.mpeg.telecomitalialab.com/standards/mpeg-2.htm.

Steinnmetz, Ralf and Klara Nahrstedt, 1995. Multimedia: Computing, Communications and Applications. Prentice Hall, NJ, USA.

Serber, R., E. Brosh and S. Chaikin, 1996. Video over the InterNet.Retrieved from the World Wide Web: http://www2.rad.com/networks/1996/video/video.htm.

Steinke, S., 2001. Network Delay and Signal Propagation. NetworkMagzine.com. Retrieved from the World Wide Web: http://www.networkmagzine.com/article/NMG20010416S0006.

Yahya, A., A. Aballa and K. Al-Qawasmi, 2003. Compressing Video Using Matrix Folding, Pak. J. Info. Tech., 2: 61-64.