An Image Sequence Coder Based on Vector Quantization and Progressive Transmission

Paulo Lício de Geus

Departamento de Ciência da Computação - Universidade Estadual de Campinas
DCC - IMECC - UNICAMP — caixa postal: 6065, Campinas SP Brasil
voice: +55 (0)192 39-7470/3115; fax: ... 39-5808
e-mail: paulo@dcc.unicamp.ansp.br (Internet) October 13, 1991

Abstract

Traditional approaches to image sequence coding rely on the Discrete Cosine Transform (DCT) to achieve high compression rates. In this paper, an encoding scheme for image sequences is presented that trades off some image quality for high compression rates and low computational load. The coder is essentially integer-based and works as follows: i) removes temporal redundancy by a simple, block-based movement detection strategy; ii) provides a first image block approximation using tree-searched Vector Quantization (VQ); iii) then progressively transmits complementary information spread along a few frame periods if no further movement is detected. The coder also accepts the DCT in place of VQ.

Introduction

Image sequence coding or compression is useful in a number of applications, such as videoteleconferencing and multimedia systems. Another possible application would be the video viewer, a tool that shows a standard TV signal on a workstation's screen, in a rate as close to real-time as possible, under control of the local window manager. Images are acquired with a camera setup, coded and transmitted through a LAN (Local Area Network) connection, such as the Ethernet. Among its possible uses are as a means of communicating among researchers in ample environments and for teaching purposes. A standard 10 Mbits/s Ethernet running the TCP/IP protocol suite is able to convey useful data at a maximum rate of 1.18 MBytes/s. This is running a simple protocol over the UDP layer. A previous work [Geus90] showed that such rates are attainable with today's hardware. A PAL video signal, digitized so that the resulting image is a 512x512 pixel rectangle, with 8 bits/pixel of brightness (gray levels only, for simplicity) and 25 frames per second, generates a 6.6 MBytes/s stream of data. For two simultaneous video streams on Ethernet, the compression rate needed is at least Ý12:1.

Videoteleconferencing techniques usually employ transforms to obtain very high compression rates, usually as part of hybrid methods. The computational power required is on the order of hundreds of MFLOPS. In contrast, the coder presented in this paper is essentially integer-based, making it much more suitable for cheap implementations, such as a workstation's expansion card. This coder first removes temporal redundancy by a simple movement detection strategy. It subdivides the image in rectangular blocks of sizes 8x8 and 4x4 and applies a distortion criterion, such as MSE (Mean Square Error), to mark the blocks that need updating. Secondly, for those blocks needing updating, an early image description is coded by a tree-searched VQ so that an approximate image is available for reconstruction. Thirdly, the progressive transmission phase codes complementary image information. In these experiments a simple DPCM (Differential Pulse Code Modulation) coder was used, for simplicity. At any time, detection of movement makes the algorithm return to the second step. The results obtained showed good compression rates with low computational load, but image quality was severely compromised in some cases. The advantages and limitations of the scheme are discussed, and suggestions are given to correct the main problem and to further improve performance.

Reasoning behind MDPT

The reasoning behind MDPT (Movement Detected, Progressive Transmission) is centred around lowering the

amount of data to be coded by spreading it along more than one frame period. Experiments by Miyahara [Miyahara75] revealed that for certain kinds of movement on the scene subjects could detect loss of resolution less easily in the areas of movement. Seyler and Budrikis [Seyler65] found that image resolution could be significantly reduced immediately after a scene change, and that restoration to full resolution needed not be particularly fast. If full resolution was restored within approximately 0.75 s subjects would not notice the temporary resolution reduction. In other words, there is evidence that one may lower the image resolution for moving regions in general, and that one may also delay restoration to full resolution after movement in a region has stopped. Moreover, the dynamics of moving regions in typical head-and-shoulders scenes is such that most bursts of movement will last for several frame periods, since a frame period is a relatively short period of time (PAL system: 40 ms).

The strategy adopted provides only an early image description (an approximation) for blocks where movement is detected, generating very little data initially. Further complementary image information is only coded if movement is not detected again. This usually lasts for a few frame periods, till the reconstructed image error gets below a predefined threshold. The data coding savings provided by this scheme come from the fact that a number of moving regions stay in this situation for several contiguous or almost contiguous frame periods in typical scenes. Coding extra image detail would be useless anyway. Only after the burst of movement ends is that the extra image detail is coded.

Movement Detection

Motion compensation techniques try to find how far a certain region of image moves between consecutive frames in a sequence, and to code such displacement. These techniques assume that movement is modeled mainly by linear shifts, which turns out to be a good assumption for most cases [Musmann85]. Interframe conditional replenishment, on the other hand, considers a fixed region of image between consecutive frames to decide for an update, based on significant brightness changes. MDPT goes beyond that by linking the movement that may be happening with the changes in brightness that may be observed. Perhaps this is not true for all situations, for instance, brightness changes may also occur due to camera and digitalization noise, but the assumption is quite rewarding [Geus90].

Simple movement detection algorithms which subdivide the image in 8x8 and 4x4 blocks were tested. 8x8 blocks are more desirable due to potentially greater compression rates. However, either sensitivity is too low and movement is poorly detected (with the reconstruction showing jerkiness), or sensitivity is too high and compression is poor (due to noise). The solution lies in working both with 8x8 and 4x4 blocks. 8x8 blocks are subdivided into four quadrants, and a sum (either absolute or of squares) is taken over the brightness of some or all pixels in each quadrant. Movement is flagged in a quadrant by comparing the error result between two frames with a given threshold. On movement detection, either a single quadrant or the full block is marked for updating. The algorithm will generally code less data and image quality after the first reconstruction step benefits from using smaller blocks. Complexity is very low at $O(N)$. Data is typically reduced to 25–30% of the original with no noticeable degradation.

Early Shape Description

The second phase of the encoding scheme required a block-based approach to image encoding. Among suitable techniques for this phase were BTC (Block Truncation Coding), the DCT and VQ, as well as hybrid solutions involving these techniques. BTC [Delp79] has very low complexity and achieves moderate compression rates, but image rendering is poor for blocks larger than 4x4, therefore limiting the compression rate. The DCT [Ahmed74] is undoubtedly the most indicated solution for an image approximation with high compression rates, but it is expensive. A real-time coder would require on the order of hundreds of MFLOPS (Millions of Floating-Point Operations per Second). VQ [Gray84] allows potentially high compression rates with larger block sizes. Obviously, a full-searched VQ is not feasible for real-time operation, leaving the tree-searched VQ as the basic variant to be examined. The complexity of VQ is $O(2k \cdot N)$, which is slightly higher than that for the DCT at $O(N \cdot log_2(N))$, for typical values of k. The main difference, however, is that VQ requires only integer

operations, making it much easier to VLSI implementation, becoming our choice. For better performance, the input vectors (image blocks) had first their average brightness removed. The average brightness was separately encoded using DPCM.

Progressive Transmission

The third phase codes complementary information to enhance the first image approximation. Many ways can be devised to implement such function. For the experiments reported here, however, we have resorted to a DPCM coder for simplicity. For the Block Eight implementation (8x8 blocks), the block is subdivided into four quadrants and each quadrant coded using 4x4 VQ. The four vectors are coded during two frame periods, after which the DPCM phase proper starts over. For each quadrant, an error image is generated from the original quadrant and the one reconstructed so far (used by the coder). A DPCM coder is then run with 1 or 2 bits/pixel and a scale factor, with each full DPCM cycle taking two frame periods. The cycle is repeated till each quadrant's distortion level gets below a given threshold), with the scale factor being lowered as the error decreases. At any time in the progressive reconstruction process the detection of movement restarts the process from a fresh VQ code.

Details of Codebooks Used

Seven image sequences were chosen for the experiments, some typically head-and-shoulders (named talk*n*), others purposefully containing high movement (named hand*n*). The latter ones insure full reconstruction of most of the scene. Four of these sequences were 64 frames long (256x256 resolution), and the remaining 16 frames long (512x512). The training sequence to build the codebooks contained four intermediate frames from each sequence. The 8x8 codebook had 512 codewords and the 4x4 had 256 codewords. The codebooks were built by splitting, to match the tree-searched VQ strategy. The first pass builds a codebook of rate 0 (1 vector), which is the centroid of the training sequence; the following passes all start by generating a higher rate codebook through adding small amplitude error vectors to each codeword and then running the training sequence again to find the new centroids. The whole tree of intermediate codebooks allows the fast tree-searched VQ operation.

Results

Two main variants of MDPT/VQ were tested: Block Eight and Block Four. Block Eight uses 8x8 vectors in the early shape description stage and starts the progressive transmission stage doing VQ again, this time with 4x4 blocks for each of its quadrants, continued then with conditional DPCM replenishment. Block Four is simpler: the early shape description stage uses 4x4 vectors and the progressive transmission stage uses only conditional DPCM replenishment. Results are summarized in TABLE 1. It should be noted that the purpose of these

TABLE 1. Some Typical Results of the MDPT/VQ Coder

sequence	MDPT variant	average compression rate (n:1)	SNR(dB)
hand3	Block Eight	26.8	27.4
talk2	Block Eight	38.0	29.5
hand3	Block Four	13.0	29.7
talk2	Block Four	23.2	30.6

experiments was to test the strategy devised for the MDPT scheme. As such, we have not spent much efforts in building a full coder, in the sense that some parts in the process were not optimized. The progressive transmission phase certainly deserves more elaborate coding. The resulting stream of data could also be further compressed by statistical methods. FIGURE 1. shows two intermediate frames of sequence hand3 in both original

(top), coded by Block Eight (middle) and coded by Block Four (bottom). As can be seen in the previous table, the Block Eight coder achieved greater compression rates than Block Four. However, the Block Four coder achieved acceptable quality, in contrast with Block Eight's poor image quality. The noise figures do not translate the significant difference in perceptual image quality. Both coders suffered from poor edge rendition, and that accounts for most of the perceived poor quality. The Block Four coder was helped here in edge tracking by its smaller block size, and Block Eight presented also some blockiness effect in low gradient areas.

Conclusions

It is clear from the perceptual results that the codebooks used were not good enough for the task, due to both their sizes and intrinsic limitations of the codebook generation process. Edges occur in such a variety of ways in a typical training sequence the codebook results heavily biased toward low gradient blocks, with few codewords representing the many possible instances of edges in the input images. On the other hand, though, compression rates were quite attractive. This result warrants further research to improve the coding scheme, particularly with the *early shape description* and the *progressive transmission* phases. It is felt that the movement detection phase is already efficient enough, particularly due to its low computational cost. In summary, the MDPT/VQ coder showed good compression rates, but image quality ranged from acceptable to poor. Overall computational load is not high, but most importantly, the computation is virtually integer-based, which contributes for cheaper implementations. The general MDPT compression scheme showed good potential, not only to work with more refined VQ variants, but also with other suitable coding techniques.

Future Work

The MDPT compression scheme may be developed further by improving the current VQ stage. First of all, larger codebooks may improve performance with very little degradation in the compression rates. The progressive transmission stage demands more efficiency. Since edge rendition is the main problem with image quality, the logical way to improve VQ is the concept of Classified VQ, first proposed by Ramamurthi and Gersho [Ramamurthi86]. For real-time operation, however, the image classification stage requires a fast, simplified classification scheme, much in the way movement detection is already performed in MDPT. The recent availability of real-time silicon implementations of the DCT for standard video signals makes the DCT a good alternative for VQ. The more significant coefficients would be coded first, resulting in improved image rendition after each reconstruction phase.

References

[Ahmed74] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete Cosine Transform. *IEEE Trans. Comput.*, C-23:90–94, January 1974.

[Delp79] Edward J. Delp and O. Robert Mitchell. Image Compression Using Block Truncation Coding. *IEEE Trans. Communications*, COM-27(9):1335--1342, September 1979.

[Geus90] de Geus, P. L., Approaches to Live Image Transmission between Workstations over Limited-Bandwidth Networks, *Ph.D. Thesis*, Dept. of Computer Science, University of Manchester, U.K., June 1990.

[Gray84] Robert M. Gray. Vector quantization. *IEEE ASSP Magazine*, pp 4--29, April 1984.

[Miyahara75] M. Miyahara. Analysis of Perception of Motion in Television Signals and its Application to Bandwidth Compression. *IEEE Trans. Communications*, COM-23:761–766, July 1975.

[Musmann85] Hans Georg Musmann, Peter Pirsch, and Hans-Joachim Grallert. Advances in Picture Coding. *Proceedings of the IEEE*, 73(4):523--548, April 1985.

[Ramamurthi86] Bhaskar Ramamurthi and Allen Gersho. Classified Vector Quantization of Images. *IEEE Trans. Communications*, COM-34(11):1105–1115, November 1986.

[Seyler65] A. J. Seyler and Z. L. Budrikis. Detail Perception After Scene Changes in Television Image Presentations. *IEEE Trans. Inform. Theory*, IT-11:31–43, January 1965.

FIGURE 1. MDPT examples: original, Eight and Four