Live Video Streaming on Embedded Devices through Wireless Channel

R.Susanto, D.Wu, X. Lin, K.P. Lim, R. Yu, F. Pan, Z. Li, S. Yao, G. Feng, S. Wu

Signal Processing Program
Laboratories for Information Technology
21 Heng Mui Keng Terrace, Singapore 119613, email: susantorahardja@ieee.org; djwu@lit.a-star.edu.sg

Abstract

We introduce a PDA-based wireless live video streaming system based on MPEG-4 video compression standard. The system is implemented in software. Compared to a dedicated hardware implementation, software implementation is programmable thereby providing flexibility to incorporate new algorithms. Due to limited computational resources of PDA, all the key modules of MPEG-4 codec are efficiently implemented and optimized such as multithreading, buffer design, wireless communication, encoder and decoder. Several novel techniques are developed in the coding and streaming portions as well as the post-processing stages of the system. The results indicate that the proposed system has successfully reached the aim of achieving real-time video coding through the wireless channel.

Keywords: Video Compression, MPEG-4, Embedded System, Communication System, Multithreading

1. Introduction

In recent years, Personal Digital Assistants (PDA) have attracted tremendous attention. Contrast to the bulky notebook computers, these electronic devices enable users to carry them much more easily and run application programs including taking notes, scheduling plans, surfing the Net, accessing email-boxes and many more. Generally, these PDAs such as Palm, Jornada, iPac, etc can be considered as embedded systems. The reason is partly because the underlying software platforms inside are the embedded operating systems like Windows CE and Palm OS.

With the advent of high speed and low power embedded processors for PDAs, there is a growing trend towards a strong demand for better multimedia applications "on the move". The added dimension of broadband networks such as 2.5G or 3G communication network will undoubtedly proliferate embedded multimedia applications on PDA devices in a very near future. Of particular interest, live video streaming is becoming a viable application on PDA devices. This is because of the improvement in embedded processor's speed and the availability of various wireless broadband networks such as wireless LAN, GPRS (General Packet Radio Service), and emerging 3G communication services.

However, the video stream content involved in a video streaming system contains so much data that compression techniques should be utilized to transmit video data through the narrow bandwidth transmission channels. In another word, codec which stands for compression /decompression techniques is indispensable in this kind of video streaming system.

In a live video streaming system, data exchange must be performed in real-time. Thus having efficient algorithm and implementation to achieve the state of real-time in the streaming is a necessity in the codec here. Since MPEG-4 [1,2,3] is one of the best compression standards suitable for low bit-rate compression, it is adopted in the system.

The paper consists of seven parts. Part 1 gives an introduction to this paper. Part 2 gives a systematic view of the developed video streaming system. Part 3 explains the main idea of video compression techniques. Part 4 presents some implementation and optimization details. Part 5 introduces several novel techniques implemented in the system. Part 6 shows the result of the system. Part 7 gives future plans and concludes the paper.

2. System description

The PDA-based wireless video streaming system developed is mainly related to software. The encoder side of the system consists of a HP Jornada 568 PocketPC with Intel StrongARM SA-1110 processor inside, Socket Low Power CompactFlash (Type 1) IEEE802.11b Wi-Fi wireless LAN Card and the encoder application software. In addition, a video camera and a video capture card is plugged into the slot of encoder PDA for live video capture. In this case, the FlyCAM-CF digital video camera made by Animation Technologies is used. The decoder side consists of the similar configuration with the exception of the decoder application software. The two sides are connected through Wireless LAN and Internet.

SDK is provided with the video capture hardware so that the encoder can read the raw YUV format live video. After compression, the MPEG-4 compressed video bit stream is sent out to the wireless channel before it is

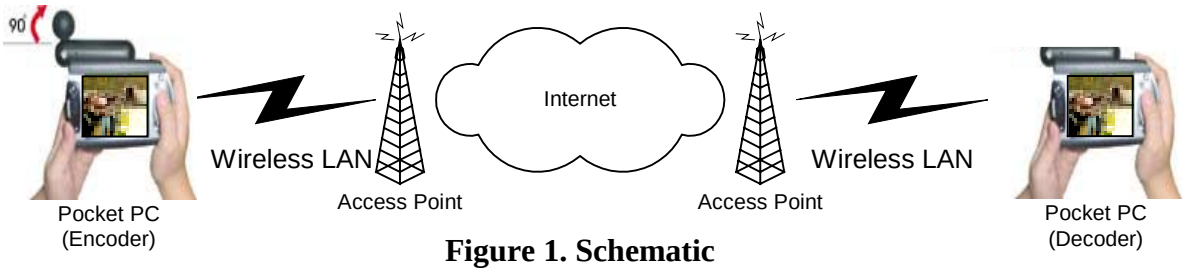

Figure 1. Schematic diagram of the system

decoded on the remote side in the end. A schematic diagram of the system is shown below in Figure 1.

3. Video compression background

Uncompressed video takes so much storage space and transmission bandwidth that video compression provides an effective approach to save the space and bandwidth. Video can be viewed as continuous image slices along the time axis. There exist different degrees of data redundancy between the neighboring image slices. Compression of image data without magnificent loss of visual information is mainly due to three reasons. Firstly, images contain a high degree of spatial redundancy due to correlation between neighboring pixels. Secondly, they contain some spectral redundancy because of correlation among the color components. Thirdly, they show some degree of psychovisual redundancy due to human being's visual system. From theoretical viewpoint, we should obtain as high compression as possible based on the redundancy information in the video data.

Statistical (spatial) redundancy exists because the pixel values of images in this world usually are not totally random, but representing a degree of gradual changing. Psychovisual redundancy is due to the reason that the human visual system is insensitive to some spatial frequencies. Since the neighboring image slices inside the video sequence generally shows strong data redundancy, this feature is also being utilized to compress the video data more. A typical image compression system [4,5,6,7] is generally comprised of the following parts: motion estimation, motion compensation, transformation, quantization, and coding.

The video codec adopted in the system is MPEG-4, which is a source codec that facilitates efficient storage, transmission and manipulation of video data in multimedia environments. One of its targeted application is just coding and transmitting video stream through the wireless channel.

4. Implementation and optimization

Our implementation is based on the MPEG-4 codec developed previously by Laboratories for Information Technology. However this codec is written under Win32 environment [8,9], rather than Windows CE. To make things much worse, the initial performance of the video codec on the StrongARM platform (after porting to Windows CE) is unsatisfactory. Thus a lot of optimization has to be done to improve its performance. Major optimizations and implementation issues are discussed below.

(a) MPEG-4 video encoder optimization

1) Encode_MacrocBlock module optimization

It was found that the Encode_MacroBlock() function executed subroutines that are the most computationally intensive. All the modules in the function have been successfully re-coded and optimized to boost the performance. There are mainly two tasks that Encode_MacroBlock() does: to encode the INTRA-blocks and the INTER-blocks. To compute the INTRA-blocks, the following steps have been optimized and implemented: i) transform the input image blocks using our new fast fixed-point DCT transform; ii) quantize the DCT-coefficients in the transformed blocks; iii) dequantize the quantized-DCT-coefficients; iv) reconstruct the blocks by performing inverse-DCT transform on the dequantized-DCT-coefficients. To compute the INTER blocks, the following steps have been optimized and implemented: i) generate the error-signal-blocks by subtracting the current-blocks-to-be-coded with the motion-compensated-blocks; ii) transform the error-signal-blocks using DCT transform; iii) quantize the DCT-coefficients in the transformed blocks; iv) dequantize the quantized-DCT-coefficients; v) perform the inverse-DCT transform on the DCT-coefficients and adding the results to the motion-compensated-blocks to reconstruct the image.

2) Advance-detection of Zero-DCT-Coefficients

It is realized that in INTER coding, a large percentage of the DCT-blocks after quantization are all

zero. By predicting these all zero DCT-coefficient-blocks, we can skip the DCT transform, quantization, dequantization and Inverse DCT transform steps thus increasing the speed of encoding. [10]

3) Combining quantization with Fast DCT

We have adopted the fast DCT proposed by Arai, Agui and Nakajima [11]. This version of DCT requires 13 multiplications and 29 additions/subtractions for each 8-point DCT and is one of the fastest algorithms at present. The 2-D DCT is performed in two stages. The first stage is to perform DCT on the 8-point columns and the second stage on the 8-point rows. In the second stage, by combining the quantization with the DCT, instead of using 13 multiplications, only 5 multiplications are required. In addition, the integration reduces the unnecessary data write and read that is required to store the temporary DCT coefficients before doing the quantization.

(b) MPEG-4 video decoder optimization

1) Fixed-point inverse DCT

The StrongARM processor does not have a dedicated floating point unit. As such, doing floating point operation has to rely on software emulation which is a very computationally intensive process. By converting the operations to fixed-point IDCT computation, the performance is substantially increased without hurting the precision and the visual quality output.

2) Optimizing the motion compensation module

This is done by looking into how data can be efficiently moved from one segment of the memory area to another. The data movement is important because the StrongARM processor has a very small cache memory: 48KB for data and instruction cache. Comparing this with the 256KB or 512KB cache found in the x86 platform, the cache in StrongARM processor is very small and we expect much more cache misses when running memory intensive application such as our MPEG-4 program. In case of cache misses, at least 4 CPU cycles are used which is a relatively heavy penalty. To prevent this shortcoming, unnecessary memory read and write are avoided. Another optimization strategy employed is to read one long word (four bytes) in one memory access rather than reading a single byte four times. This will speed up the processing by at least two times.

3) Optimizing the video display module

The decoded image is written directly into the video memory buffer. This is only possible if we know the virtual start address of the video memory buffer. We have successfully implemented the direct access technique on Jornada 568 PocketPC.

4) Combining motion compensation with IDCT

When these two processes are separated, the result of IDCT has to be written into a temporary buffer and is later retrieved (read) for motion compensation. This write and read process can be eliminated by directly doing motion compensation immediately after the IDCT coefficients have been computed.

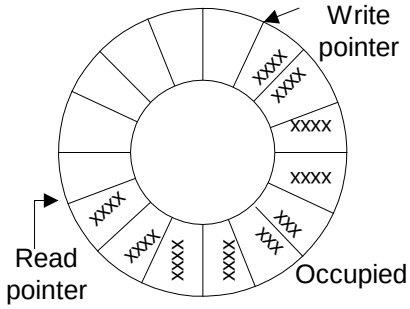

Figure 2. Circular buffer

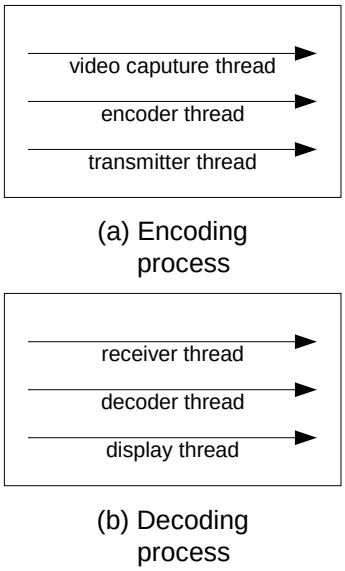

Figure 3. Concurrency in the codec

(c) Buffer issue

Memory space is used to buffer the encoded video bit streams. The encoder fills up the buffer while the network thread reads the data and sends it out to the network. Once the data has been fetched, the memory space can be used to store new coded frame from the encoder. Once the encoder's pointer reached the end of the buffer, it is brought to the beginning of the buffer. The similar buffer operation is also applied to the decoder side. In this way,

the same memory block could be used over and over again in a circular manner. This process can be illustrated in Figure 2.

(d) Multithreading

At the encoder side, the video capture, encoder and transmitter have to be run concurrently. At the decoder side, the receiver, decoder and display also need to run at the same time. This type of concurrency is obtained by using multi-thread programming techniques. In Windows programming, this is achieved by forking the codecs as threads, as illustrated in Figure 3.

(e) Communication through wireless LAN

Wireless LAN standard (IEEE 802.11b) provides the way to connect PDA with wireless access point, which is further connected to the Internet. Since wireless LAN implements TCP/IP stack, Windows CE socket programming is adopted in order to implement communication functionalities such as opening a connection, listening, sending data, receiving data and so on.

5. Novel techniques utilized

Besides implementing and optimizing MPEG-4 coding on PDA, we also develop several novel techniques in coding, streaming and post-processing stages of the whole system. These techniques can make the encoding process faster, streaming better and decoded video quality more pleasing. Figure 4 shows the whole data flow of the system. On the encoding side, raw live video data captured and stored in the video buffer of the video camera is input to the encoder (Enc). The encoder encodes and stores the encoded bits in the encoder circular buffer (Enc buffer) and do self-adjustment by receiving feedback from Enc Buffer. The transmitter (Tran) gets data from Enc Buffer if it is not empty and undertakes the task of putting these bits to the network. On the decoding side, receiver reads data from net and puts it in the decoder circular buffer (Dec buffer) so long as it is not full. The decoder (Dec) gets data from Dec Buffer if available and does the decoding. Then decoded video is post-processed before being displayed on the screen. The following introduces the three novel techniques used.

(a) 2-stage partial distortion search

A novel 2-stage partial distortion search (2S-PDS) algorithm [12] is proposed and utilized in the system to reduce the computational complexity in block motion estimation algorithms. In the algorithm, an early-rejection stage is introduced where the partial distortion of a decimated pixel block is calculated and compared with its local minimum. A block is rejected without calculating the full distortion of the entire block if the partial distortion is larger. The local minimum is amplified by a pre-defined threshold value before the comparison in order that the probability of false rejection can be reduced. The algorithm reduces the complexity of block motion estimation significantly with only marginal performance penalty. In addition, the algorithm can be used in combination with full-search or other fast search algorithms.

(b) Robust rate control

During the process of developing the system, we found that since both the available channel bandwidth for the live video coding process and the statistics of video are usually time varying, the quality of each picture vary vastly if the encoding frame rate is always fixed at a predefined rate. Meanwhile, since the live video is encoded on the embedded devices which can not always allocate necessary computational resources to the coding process, the actual frame rate is sometimes less than the specified frame rate. Some portion of the channel

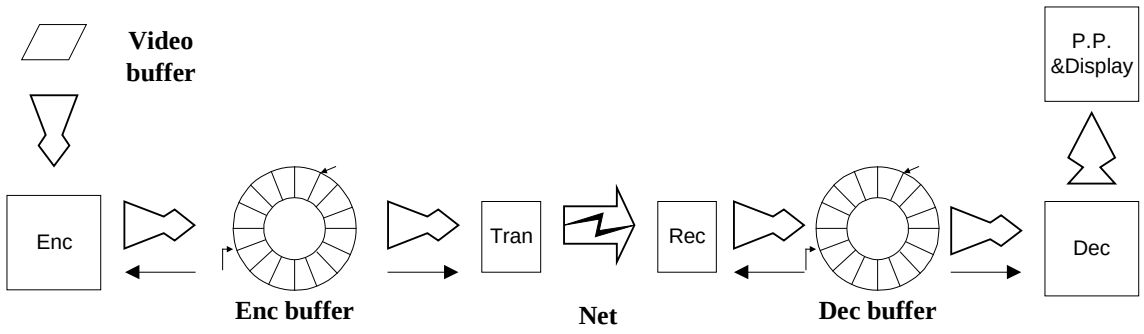

Figure 4. Data flow graph of the system

bandwidth will be wasted if this phenomenon is not considered in the rate control algorithm of encoding process. Accordingly, a new rate control scheme is designed and implemented to address the problems. With our rate control algorithm, the encoder can adaptively adjust itself to both the computational resource allocated and the time varying channel bandwidth, resulting to better utilization of channel bandwidth and better PSNR performance. The average PSNR has been improved by 1.56dB.

(c) Post-processing after decoding

Due to coarse quantization at low bit rate coding, the perceived quality of decoded video frames are severely degraded by various artifacts, such as blocking, blotchiness and ringing effects. In order to remove annoying coding artifacts and substantially improve the video quality, we developed an efficient post-processing algorithm [13] that is based on image degradation model and use half-quadratic regularization for preserving image edges and removing ringing effects. The proposed algorithm has been tested using several images and compared with the current proposals proposed by University of Wisconsin and University of Southern California, which have been included in the international standard JPEG 2000. The experiments have shown that the objective quality of decompressed images in terms of peak signal-noise-ratio (PSNR) is largely improved and the visual quality of our results is much better than others. With this post-processing technique for the video coding, the PSNR value of decoded video can be increased 0.25dB in comparison with the results of MPEG-4 post-processing algorithm.

6. Results

The encoder software developed is able to compress and transmit video data with bit rate of around 120 kbps for the size of QCIF (176x144). The decoder can receive the video signal remotely via wireless LAN. After the optimizations, the final deliverables of the project can achieve 21 frames per second for the encoding rate and 29 frames per second for the decoding rate.

7. Conclusion

We have considered two approaches to enhance video streaming system. One is to transform our wireless video communication system into a duplex one. Another one is to use Bluetooth or GPRS as the underlying wireless communication channel so that we could investigate the codec performance under these kinds of wireless environment.

By and large, the objective of the Project of developing a real-time ISO/IEC MPEG-4 Simple Video Profile conformant software encoder/decoder module on the Intel StrongARM RISC processor (SA-1110) platform has been attained. The key constituent software encompasses the Base Layer MPEG-4 encoder/decoder, Error Resilient Tools, and the system is designed on Microsoft Windows CE operating system. Besides implementing and optimizing the source codes, a few novel techniques are also utilized. Results have shown that the video codec developed can perform real-time compression. The live video streaming system on embedded devices through wireless channel has satisfied the requirements set by our clients and we are certain that it can be deployed in the wireless video communication world of the present day.

8. References

[1] ISO/IEC JTC1/SC29/WG11, "Information technology, coding of audio-visual objects: Visual, ISO/IEC 14496-2, Committee Draft", ISO/IEC N2202, March 1998.

[2] ISO/IEC JTC1/SC29/WG11, "MPEG-4 video verification model version 11.0", MPEG98/N2172, March 1998.

[3] ISO/IEC JTC1/SC29/WG11, "MPEG-4 overview - (Fribourg version)," MPEG97/N1909, October 1997.

[4] W. Kou, Digital mage compression : algorithms and standards, Boston, Kluwer Academic, 1995.

[5] A.M. Tekalp, Digital Video Processing, Prentice -Hall International, INC., 1995.

[6] Veldhuis & Breeuwer, An Introduction to Source Coding, Prentice-Hall International, INC., 1993.

[7] R.J. Carke, Digital Compression of Still Images and Video, Academic Press, 1995.

[8] W.A. Redmond, Microsoft Visual C++: Class Library Reference for the Microsoft Foundation Class Library, Microsoft Press, 1994.

[9] E. Olafsen, K. Scribner, K. David White, et al, MFC Programming with Visual C++ 6 Unleashed, Sams Publishing, 1999.

[10] D.C. Soong, A.R. Ramli and M.R. Mukerjee, "All-zero-AC block detection using energy preservation theorem for H.263 video coding", IEEE TENCON 2000 Proceedings, Vol. 2, 2000, pp.425-430.

[11] Y. Arai, T. Agui and M. Nakajima, "A Fast DCT-SQ Scheme for Images", IEEE Transactions of the IEICE, Vol. E71, No.11, 1988, pp. 1095-1097.

[12] R. Yu, K. P. Lim, D. Wu, F. Pan, Z. G. Li, G. Feng, S. Wu, "A 2-stage partial distortion search algorithm for block motion estimation", The 3rd IEEE Pacific-Rim Conference on Multimedia (PCM 2002), Tsing-Hua University, Hsinchu, Taiwan, December 16 - December 18, 2002

[13] S. Yao and X. Lin, "Post-processing for removing coding artifacts using edge-preserving regularization", Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing, 2001, pp. 121 -124