REAL – TIME IMAGE ENCODER WITH DCT COMPRESSION ALGORITHM

Sriram. R[*] and Venkatesh. S[#]

**Department of Electronics and Communication Engineering, Meenakshi College of Engineering, Chennai -78.*
Email: sriram@ccf.iisc.ernet.in
#School of Communication and Information, Rutgers University – New Brunswick, New Jersey.

ABSTRACT

The main focus of the paper is to compress an image through Discrete Cosine Transform (DCT) technique; this can be accomplished by reducing the chrominance value of the image. To discard such things, the compressor divides each DCT output value by Co efficient of quantization and rounds to integer. Thus the JPEG lossy is more frugal to grey scale and more frivolous to color. The experiment shows that this new scheme of image compression tremendously reduces the image size upto 18 – 22 times of the original size, which is a quite important feature required by real time system.

Key words - Discrete Cosine Transform (DCT), Chrominance, Quantization, Compression.

I. INTRODUCTION

Currently, the most popular image compression scheme is JPEG Standard for still images and MPEG standard for video sequence [1]. The system aims at converting an image file to its compressed format by using some encoding techniques. This can be accomplished by using a set of primitives. These primitives are meant to process an image through various steps and result an image file in compressed format [9]. For typical images, the adjacent pixels usually possess a high degree spatial correlation that is a great deal of information about a pixel value can be obtained by investigating its neighboring pixel values.

Predictive techniques exploit this spatial correlation and encode only the new information between successive pixels. Intra frame coding is one such predictive technique.

Intra frame compression is the technique which can be applied to still images, such as photographs and diagrams, and exploits the redundancy within the images, known as spatial redundancy. Intra frame compression techniques can also applied to individual frames of a video sequence.

JPEG Encoder was designed specifically to discard information that the human eye cannot easily see. Slight changes in color are not perceived well by the human eye, while slight changes in intensity (light and dark) are. Therefore JPEG's lossy encoding tends to be more frugal with the gray-scale part of an image and to be more frivolous with the color [1].

The JPEG specification defines a minimal subset of the standard called baseline JPEG, which all JPEG-aware applications are required to support. This baseline uses an encoding scheme based on the Discrete Cosine Transform (DCT) to achieve compression. DCT is a generic name for a class of operations identified and published some years ago. DCT-based algorithms have since made their way into various compression methods.

DCT algorithms are capable of achieving a high degree of compression with only minimal loss of data. This scheme is effective only for compressing continuous-tone images in which the differences between adjacent pixels are usually small.

II. DOWN SAMPLING THE CHROMINANCE COMPONENTS

The simplest way of exploiting the eye's lesser sensitivity to chrominance information is simply to use fewer pixels for the chrominance channels. For example, in an image nominally 1000x1000 pixels, we might use a full 1000x1000 luminance pixels but only 500x500 pixels for each chrominance component.

When the uncompressed data is supplied in a conventional format (equal resolution for all channels), a JPEG compressor must reduce the resolution of the chrominance channels by down sampling, or averaging together groups of pixels.

A DCT is applied to each 8x8 block. DCT converts the spatial image representation into a frequency map: the low-order or "DC" term represents the average value in the block, while successive higher-order ("AC") terms represent the strength of more and more rapid changes across the width or height of the block.

The highest AC term represents the strength of a cosine wave alternating from maximum to minimum at adjacent pixels. The DCT calculation is fairly complex; in fact, this is the most costly step in JPEG compression. The point of doing it is that we have now separated out the high- and low-frequency information present in the image.

The equations for the Discrete Cosine Transforms are [3]:

$$F(u,v) = \frac{1}{4} C(u)C(v)\left[\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y) * \cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right]$$

$$f(x,y) = \frac{1}{4}\left[\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)C(v)F(u,v) * \cos\frac{(2x+1)u\pi}{16}\cos\frac{(2y+1)v\pi}{16}\right]$$

$$\text{where: } C(u),\ C(v) = 1/\sqrt{2} \text{ for } u,v=0;$$

$$C(u),\ C(v) = 1 \text{ otherwise.}$$

We can discard high-frequency data easily without losing low-frequency information. The DCT step itself is lossless except for round off errors. If you start with a 8x8 block of 64 values, f (x,y), they can be transformed to a new set of 64 values, F(x,y), by the Forward Discrete Cosine Transform and then back to the original 64 values, f(x,y), by the Inverse Discrete Cosine Transform.

III. ENCODER IMPLEMENTATION

To discard an appropriate amount of information, the compressor divides each DCT output value by a quantization coefficient and rounds the result to an integer. The larger the quantization coefficient, the more data is lost, because the actual DCT value is represented less accurately. Each of the 64 positions of the DCT output block has its own quantization coefficient, with the higher-order terms being quantized more heavily than the low-order terms (i.e, the higher-order terms have larger quantization coefficients).

Furthermore, separate quantization tables are employed for luminance and chrominance data, with the chrominance data being quantized more heavily than the luminance data. This allows JPEG to exploit further the eye's differing sensitivity to Luminance and chrominance. It is this step that is controlled by the "quality" setting of most JPEG compressors. The compressor starts from a built-in table that is appropriate for a medium-quality setting and increases or decreases the value of each table entry in inverse proportion to the requested quality.

The complete quantization tables actually used are recorded in the compressed file so that the decompressor will know how to (approximately) reconstruct the DCT coefficients. The resulting coefficients contain a significant amount of redundant data. Huffman compression will losslessly remove the redundancies, resulting in smaller JPEG data. The encoder connection establishment is shown in below figure (1).

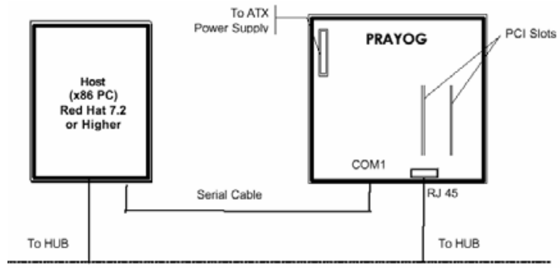

Figure 1. Connection Establishment of the x86 Processor and the development board.

The system board that we have used is technically called Prayog Development Board. Prayog is based on Intel Strong Arm Microprocessor (SA-1110) a highly integrated microcontroller that incorporates a 32-bit Strong arm processor core, system control module, multiple communication channels and an LCD controller [7]. We can use PRAYOG as the stand-alone system as it has all the required hardware interfaces to be a stand-alone computer.

RS232 is a standard for serial binary single-ended data and control signals connecting between a DTE (Data Terminal Equipment) and a DCE (Data Circuit-terminating Equipment). It is commonly used in computer serial port. So the connection between the x86 processor and development board is established by using RS-232.

IV. RESULTS AND DISCUSSIONS

To compile the developed code you need to have the Makefile in the directory where the source code is available. This Makefile is meant to compile JPEG source code on x86 processor in platform is Linux.

The connection should be made from the x86processor to prayog board through Local Area Network [6]. IP address of the both systems is to be known in order to have a uninterrupted communication and execution of the program. Here the x86 processor acts as a server. The first window displayed below is to find the IP address of the NFS server.

```
[root@faculty ~] # ifconfig
eth0        Link encap : Ethernet   HWaddr  00:13:D4:56:C0:44
            inet addr : 192 . 168 . 0 . 1   Bcast : 192 . 168 . 0 . 255   Mask:
255 . 255 . 255 . 0
```

The implementation of the connection is shown below in the second window.

```
Enter the IP address of PRAYOG :
192 . 168 . 0 . 2
ITE8152   debug
Do you want to use a NFS root  ?    (y/N)
y
Please specify the IP of NFS server .
192 . 168 . 0 .1

sh - 2 . 04#
sh - 2 . 04#
sh - 2 . 04#
```

The output of executed code is displayed below, which intimates the size of the input image and output image (Compressed image). Figure 2 represents the input image of the encoder system and the resultant output image is displayed in the figure 3.

```
sh - 2 . 04#   . /jpegencoder   cooks.bmp   sample.jpg
        Image File Name :   cooks . bmp

        Image Size Before Compression   =   921600  bytes

        Image Size After Compression  =  50587  bytes

sh - 2 . 04#
sh - 2 . 04#
sh - 2 . 04#
```

Figure 2. Input image of the Encoder system.

Figure 3. Output image of the Encoder system.

V. CONCLUSION

The quality of the image obtained using JPEG compression Techniques depends on the Quantization values chosen. Low Quantization values produce a slight loss of resolution, but no significant loss of picture quality. As the Quantization values increase, the compression process starts to become visible as blocking of the image. Use of IPP to reduce the development time greatly, and also the code is more optimized.

VI. REFERENCES

[1] Bhaskaran V, and Konstantinides K, Image and Video Compression Standards Algorithms and Architectures, 4th ed., Dordrecht: Kluwer Academic, 2007.

[2] Q. Lu, J. Luo, and R. L. Joshi, “Image Processing Method for Reducing Noise and Blocking Artifact in a Digital Image,” U.S. Patent 6 636 645, Oct. 21, 2003.

[3] Chen W H, Smith C, and Fralick S, "A Fast Computational Algorithm forthe Discrete Cosine Transform," IEEE Transactions on Communications, vo1.25, no. 9, pp. 1004-1009, Sept. 1977.

[4] Y. Kawasaka, “Block Distortion Corrector and Image Signal Expander,” U.S. Patent 5 949 917, Sep. 7, 1999.

[5] Cho N I, and Lee S U, "Fast Algorithm and Implementation of 2D Discrete Cosine Transform," IEEE Transactions on Circuits and System, vol. 38, no. 3, pp. 297-305, Mar. 1991.

[6] Agostini L V, Silva I S, and Bampi S "Pipelined fast 2-D DCT architecture for JPEG image compression," in Proc. 14th Symposium on integrated Circuits and Systems Design , Pirenopolis, Brazil, Sept. 2001,pp. 226-231.

[7] Steve Furber. "ARM System-on-Chip Architecture". Addison-Wesley, 2000.

[8] Adam Wiggins et el. "Implementations of Fast Address-Space Switching and TLB Sharing on the Strong ARM Processor", in the Proceedings of the 8th Australia-Pacific Computer Systems Architecture Conference, Aizu – Wakmatsu City, Japan, September 2003.

[9] R. Gonzales, R. Woods *Digital Image Processing*, Addison-Wesley Publishing Company, 1992, pp 81 – 125.

[10] G. K. Wallace, “The JPEG still picture compression standard,” *IEEE Trans. Consum. Electron.*, vol. 38, no. 1, pp. 18–19, Jan. 1992.

[11] Lee B, "FCT-A Fast Cosine Transform," in Proc. IEEE International Conference on Acoustics, Speech and Signal Processing, San Diego, USA, Mar. 1984, vol. 9, pp. 477-480.

[12] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages and Applications*. San Diego, CA: Academic, 1990.

[13] J. Luo, C. W. Chen, K. J. Parker, and T. S. Huang, “Artifact reduction in low bit rate DCT-based image compression,” *IEEE Trans. Image Process.*, vol. 5, no. 9, pp. 1363–1368, Sep. 1996.