RESIZING OF DCT-BASED COMPRESSED IMAGES

Mukhopadhyay J.

Department of Computer Science & Engineering,
Indian Institute of Technology, Kharagpur, 721302 , INDIA
EMail: jay@cse.iitkgp.ernet.in

Mitra S.K.

Department of Electrical and Computer Engineering,
University of California, Santa Barbara, CA, 93106, USA.
Email: mitra@ece.ucsb.edu

Abstract. In this work we propose an algorithm for resizing the digital images in its DCT-based compressed form. This is a modification of a recently proposed elegant image-resizing algorithm by Dugad and Ahuja [1]. We have also applied the techniques to color images and observed their performances at different levels of compressions for an image.

Keywords: Discrete Cosine Transform (DCT), Subband DCT Image Resizing, JPEG image compression standard.

1. Introduction

Resizing of images is required in various applications. For example, in different communication channels with different channel capacity, the same image may be transmitted at different spatial (or spectral) resolutions. Interpolation techniques are frequently used for directly resizing the image in the spatial domian. But for efficient storage, images are usually represented in the transform domain as compressed data. It is thus of interest to develop resizing algorithms directly in the compressed stream. As DCT based JPEG standard is widely used for image compression, a number of approaches have been advanced to resize the images in the DCT space [2], [3],[4],[5],[6]. Very recently, Dugad and Ahuja [1] have proposed an elegant scheme for changing the image sizes in the DCT space. They have suggested a simple fast computation technique for halving and doubling of images using their low frequeny components. The principle behind the algorithms developed by Dugad and Ahuja is similar to the subband DCT computation [7]. In this work a modification to their algorithm is suggested. In our approach we have used subband DCTs for image resizing [7].

2. Discrete Cosine Transform from Subband Computation

Let us present here briefly the computation technique for DCT's of an image from its subbands from [7]. The definition of DCT for a 2D images x(m,n) of size NxN is as follows:

$$C(k,l)=\sum_{m=0}^{N-1}\sum_{n=0}^{N-1}\left(4x(m,n)\cos\left(\tfrac{(2m+1)\pi k}{2K}\right)\cos\left(\tfrac{(2n+1)\pi l}{2N}\right)\right) \quad (1)$$

$$0\le k,l\le N-1$$

The *low-low* subband x_{LL} (m,n) of the image be obtained as:

$$x_{LL}(m,n)=\tfrac{1}{4}\{x(2m,2n)+x(2m+1,2n)$$
$$+(2m,2n+1)+x(2m+1,2n+1)\},\ (2)$$
$$0\le m,n\le \tfrac{N}{2}-1.$$

Let C_{LL} (k,l), $0 \le k,l \le N/2-1$ be the 2D DCT of x_{LL}(m,n). Then the *subband approximation* of DCT of x(m,n) is given by:

$$C(k,l)=\begin{cases}4\cos(\frac{\pi k}{2N})\cos(\frac{\pi l}{2N})C_{LL}(k,l), & k,l=0,1,\ldots\ldots,\frac{N}{2}-1\\ 0, & otherwise\end{cases} \quad (3)$$

It may be noted that depending upon the definition of DCT, subband DCT's are multiplied by a factor (in this case $4\cos\left(\frac{\pi k}{2N}\right)\cos\left(\frac{\pi l}{2N}\right)$) The definition of inverse DCT (IDCT) should also be modified accordingly. We refer this as *subband approximation* of DCT. On the otherhand, the approximation as carried out in the approach proposed by Dugad and Ahuja [1], is as follows:

$$C(k,l)=\begin{cases}4C_{LL}(k,l), & k,l=0,1,\ldots\ldots,\frac{N}{2}-1\\ 0, & otherwise\end{cases} \quad (4)$$

We refer this approximation as the low-pass truncated approximation of DCT. Interestingly, the multiplication factor 4 in Eq. (4) appears due to the definition of DCT used in this work (refer to Eq. (1)). However, this factor does not have any effect in the final results obtained by them (PSNR values of downsized (halved) and then upsized (doubled) images in [1]). While halving an image, DCT coefficients for N/2-point DCT are obtained by dividing the N-point DCT coefficients with the factor (e.g 4 in our case). Subsequently, during image doubling, the N-point DCT coefficients are obtained by multiplying the N/2-point DCT coefficinets with the same factor (see the next section for the description of their algorithm). For the details of the subband computation of DCT, the readers are requested to refer the work in [7].

2.1 The Resizing Algorithms

In the algorithm proposed by Dugad and Ahuja[1] while halving an image, from a 8x8 DCT block, 4x4 block in the spatial domain is obtained. This is carried out by applying a 4-point inverse DCT (IDCT) on the 4x4 lower frequency-terms. In the next stage, this image (in the spatial domain) is once again compressed by 8x8 block DCT encoding (JPEG standard).

For doubling the images, first the DCT encoded image is transformed to its spatial domain. Then for each 4x4 block, the DCT coefficients are computed applying a 4-point DCT. These 4x4 DCT coefficients are directly used as the low frequency components of 8x8 blocks, which are subsequently converted to a 8X8 block in spatial domain by applying a 8-point inverse DCT (IDCT).

2.1.1 Proposed Modifications

In our modified versions of the above algorithms, first we use the approximations as described in Eq.(3) (or Eq.(4)) while converting DCT coefficients from N/2-point to N-point or vice versa. Secondly, during image-doubling we directly use the DCT coefficients of the compressed image for converting it to a 16x16 block in the spatial domain by applying a 16-point IDCT.

In Table 1, the PSNR values obtained after halving and doubling a grey level image are presented. In the table the proposed modifications are denoted by SB for subband and TR for lowpass trucated approximations, respectively. The technique proposed by Dugad and Abuja is denoted by DA. In this table we have included the results on the same set of images [1] used in [1]. It could be seen that in almost all the cases, the proposed modifications have resulted into improved performances. Interestingly, the *low pass truncated approximations* (technique denoted by TR) yields the best result in each case presented here.

3. Extension to Color Images

The above resizing algorithms are directly extended for color images. As DCT-based JPEG encoding scheme represents color images in YUV color space, we have applied resizing algorithms separately for each color component. It may be noted that in YUV color space, Y represents the luminance part, whereas U and V represent the chrominance components. Moreover they are subsampled so that corresponding to four luminance values there is only one pair of UV values. The PSNR values obtained after halving and doubling of color images are shown in the Table 2. For color images also SB and TR algorithms perform better than the DA algorithm in most cases. Again, out of the last two, TR has the best performance. However, for the image Baboon the DA algorithm has slightly higher

PSNR than the other two.

Images	PSNR (dB)		
	DA	SB	TR
Lena	33.82	34.00	34.09
Peppers	26.39	26.54	26.59
Baboon	22.90	22.87	22.88

Table 1: PSNR values after halving and doubling a grey level image.

Images	PSNR (dB)		
	DA	SB	TR
Lena	34.64	34.83	34.95
Watch	29.26	29.57	29.72
Cap	34.33	34.33	34.37
F-16	32.43	32.70	32.82

Table 2: PSNR values after halving and doubling a color image.

[1] Images obtained from http://vision.ai.uiuc.edu/ simdugad/draft/dct.html

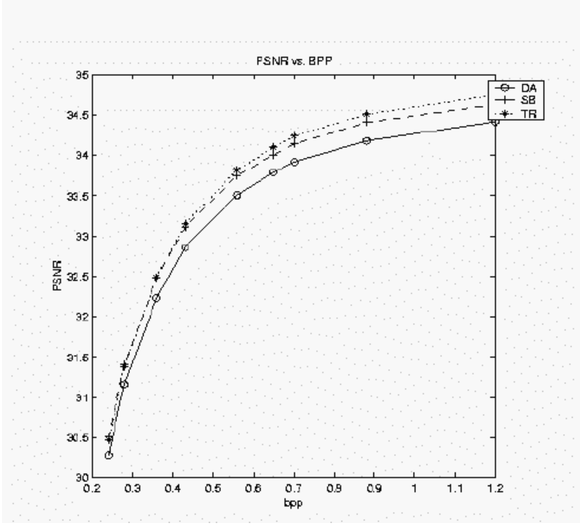

Figure 1. PSNR plots for different techniques at varying compression ratio for color image Lena.

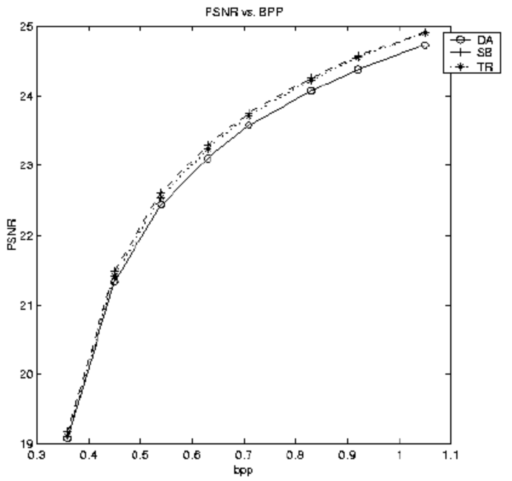

Figure 2: PSNR plots for different techniques at varying compression ratio for color image Peppers.

4. Experimentations at Different Compression Levels

Experimentations are carried out for studying the performances of the three algorithms forimages compressed at different levels. In the context of the JPEG compression, the effect of quantizations on the approximated coefficients during image-halving or image-doubling should be observed here. The PSNR values for different compression levels for the image Lena are plotted in Figure 1 for all the three techniques. For the color image Peppers similar plots are shown in Figure 2. It can be observed that with the subband and low pass truncated approximations, resizing algorithms exhibit improved performances over the DA algorithm. Interestingly, at low compression ratio, low-pass truncated approximation performs best in the most of the cases. For some images, we found the DA algorithm gives the best result at low compression (e.g. Baboon in Table 2).

5. Efficient Computations

Dugad and Ahuja have developed an elegant computational model for converting the DCT blocks of an image to the DCT blocks of its reduced or enlarged version. These conversions could be performed by multiplying the blocks with a given set of matrices and finally adding the intermediate results (please refer [1]) to the final DCT representations. The given set of matrices are derived from 8x8 and 4x4 DCT matrices. Using these efficient computations, they could restrict the image-halving and image-doubling tasks to 1.25 multiplications and 1.25 additions per pixel of the original image. In this section, we will discuss about the efficient implementation of our algorithms following similar analytical approach. It may be noted here that we present our analysis with respect to a gray level image.

It could be trivially seen that the image-halving method is almost same as the method proposed by them, except the fact that we have modified our DCT coefficients of the original image using Eq.(3) or Eq.(4). For these we require 16 more extra multiplications (for subband approximation) for 256 pixels in the original image. These increase marginally the average number of multiplications to 1.31. The number of additions remains same.

Interestingly, our image-doubling method substantially differs from the approaches proposed by Dugad and Ahuja. In this case, we have used a 16x16 inverse DCT matrix for converting 8x8 block to its spatial domain (of 16x16 pixels). As we want our output should be again in the compressed domain using 8x8 block DCT encoding scheme, we require to convert this 16X16 pixels to four 8x8 DCT blocks. These two tasks could be efficiently performed by single matrix multiplications as described below.

Let **B** be a block of DCT coefficients in the original image. Let **B** be the approximated 16x16 DCT coefficients obtained from **B** as follows:

$$\hat{B} = \begin{bmatrix} \tilde{B} & 0 \\ 0 & 0 \end{bmatrix} \qquad (5)$$

In the above equation $\tilde{B}$ is obtained from **B** according to Eq.(3) or Eq.(4). Let T_{16} and T_8 be the DCT matrices for 16-point and 8-poin DCT respectively. Let us also represent T_{16} matrix by its four sub-matrices as follows:

$$T_{16LL} = \begin{bmatrix} T_{16LL} & T_{16LH} \\ T_{16HL} & T_{16HH} \end{bmatrix} \qquad (6)$$

Let **b** be 16x16 block of pixels in the spatial domain obtained from $\hat{B}$ and **b** be represented by the following submatrices.

$$b = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} \qquad (7)$$

It could be proved that, the four 8x8 DCT blocks could be computed as follows:

$$\begin{aligned} B_{11} &= DCT(b_{11}) = (T_{16LL}T_8^t)\tilde{B}(T_{16LL}T_8^t) \\ B_{12} &= DCT(b_{12}) = (T_{16LL}T_8^t)\tilde{B}(T_{16LH}T_8^t) \\ B_{21} &= DCT(b_{21}) = (T_{16HL}T_8^t)\tilde{B}(T_{16LL}T_8^t) \\ B_{22} &= DCT(b_{22}) = (T_{16HL}T_8^t)\tilde{B}(T_{16LH}T_8^t) \end{aligned} \qquad (8)$$

It is interesting note that in the earlier method [1] all the high frequency components are forced to zero values for a 8x8 block DCT coefficients in the enlarged image. But, in our case, the high frequency terms also have some contributions towards the formation of the resulting image. This is the reason why our techniques yield better PSNR than the earlier one.

Let us call the matrices ($T_{16LL}T_8^t$), ($T_{LH}T_8^t$) and ($T_{16HL}T_8^t$) as P, Q and R. They are all 8x8 matrices.. It could be observed that 50% of the terms of those matrices are either zero or an intergral power of 2. This reduces the computation requiring eight (8) multiplications and seven (7) additions per pixel for direct matrix multiplication with them. However one may further enhance the computation by considering P=E+F and Q=E-F. One can see that the matrices E and F are more sparse. To reduce the computation, we can first compute $P^t\tilde{B}$ and $R^t\tilde{B}$. The number of multiplications and additions for each computation will be 8(32 M + 14 A). Following the same notations used in [1], for **a** multiplications and **b** additions, we denote it by **a** M +**b** A. Next we compute $P^t\tilde{B}E$ and $P^t\tilde{B}F$. Adding and subtracting these two will give respectively B_{11} and B_{12}. Similarly, we compute B_{21} (B_{22}), by computing first $R^t\tilde{B}E$ and $R^t\tilde{B}F$ and then adding (subtracting) them. This brings down the number of multiplications per pixel to 4. At the same time the number of additions per pixel is also decreased to 3.375. Hence though there is an improvement in the quality of the reconstructed picture, the computation overhead is more in our proposed scheme.

6. Conclusion

In this work we propose two modifications to the image resizing algorithms developed by Dugad and Ahuja[1]. In our modified approaches, we have used subband DCT computation [7] and also we have used 16-point IDCT for doubling the images. These have resulted in improved performances in many cases. With a simple extension of the proposed algorithms, color images are also reduced or enlarged in sizes. A comparative study of the performances of these algorithms were also carried out at different level of compression. We have observed that our proposed modification perform better than the Dugad-Ahuja method in many cases. But we should also point out that the proposed modifications considerably increase the computational overhead during image doubling.

Acknowledgement:

This work was supported by a University of California MICRO grant with matching support from Conexant Corporation and Lucent Technologies.

References

[1] R.Dugad and N.Ahuja, “A fast scheme for image size change in the compressed domain,” IEEE Transactions on Circuits and systems for Video Technology, vol. 11, no. 4, pp. 461--474, 2001.

[2] N.Merhav and V.Bhaskaran, “Fast algorithms for DCT-domain image down-sampling and for inverse motion compensation,” IEEE Transactions on Circuits and systems for Video Technology, vol. 7, no. 6, pp. 468--476, 1997.

[3] S.-F.Chang and D.G. Messerschmitt, “Manipulation and composition of MC-DCT compressed video,” IEEE J. Selected Areas Commun., vol. 13, no. 1, pp. 1--11, 1995.

[4] B.C.Smith and L.Rowe, “Algorithms for manipulating compressed images,” IEEE Comput. Graph. Applicat. Mag., vol. 13, no. Sept., pp. 34--42, 1993.

[5] Q.Hu and S.Panchanathan, “Image / video spatial scalability in compressed domain,” IEEE Transactions on Ind. Electron., vol. 45, no. Feb., pp. 23--31, 1998.

[6] A.Neri, G.Russo, and P.Talone, “Inter-block filtering and downsampling in dct domain,” Signal Processing: Image Commun., vol. 6, no. Aug., pp. 303--317, 1994.

[7] S.-H. Jung, S.K. Mitra, and D.Mukherjee, “Subband DCT: Definition, analysis and applications,” IEEE Transactions on Circuits and systems for Video Technology, vol. 6, no. 3, pp. 273--286, June 1996.