

SVD BASED LINEAR FILTERING IN DCT DOMAIN

Liansheng ZHUANG, Rui ZHAO, Nenghai YU, Bin Liu

MOE-MS Key Laboratory of Multimedia Computing and Communication, USTC, Hefei, China

ABSTRACT

Efficient linear filtering in DCT domain is important in the area of processing and manipulation of image and video streams compressed in DCT-based method. In this paper, we proposed a novel method for linear filtering in DCT domain, regardless of filter type. We decompose any filter by SVD into weighted separable sub-filters which are well studied. Then we do fast linear filtering using these separable sub-filters in DCT domain, and combine their results. To our best knowledge, it is the first method capable to do linear filtering with any type of filters directly in DCT domain. The scheme is demonstrated and discussed by doing Gabor filtering in DCT domain. Experiment results show that convolution result using the proposed solution is the same as that in spatial domain. Furthermore, our scheme is well suitable for distributed computing, which will improve computing speed greatly.

Index Terms— DCT-domain filtering, SVD, Gabor, compressed-domain processing

1. INTRODUCTION

The discrete cosine transforms (DCTs) are popular transforms used in signal compression for their energy compaction property. Using DCT, compressed data with a low data rate offers an attractive possibility of real time processing in many applications such as surveillance. However, as proved in related works, even using available fast DCT algorithms, IDCT takes up at least 40% of decoding time. Thus, many previous works have been done to implement real time processing tasks directly in DCT domain through manipulation of DCT coefficients. Convolution filtering is a basic processing to images or videos, so the operation of convolution in spatial domain is always expected to be realized in transform domain. Though discrete Fourier transform (DFT) has a simple convolution-multiplication property (CMP) in which the convolution of two sequences (for one-dimension) or matrices (for two-dimension) is equivalent to point by point multiplication of their respective counterparts in DFT domain, DCT does not have the property in such a straight forward form. Several researches have proposed some mathematical relations between spatial domain and DCT domain, and similar convolution-multiplication properties are built to calculate filtering result using DCT coefficients directly [1-6].

In [1], DCT of a signal $x(n)$ is proved to have a direct relationship with DFT of symmetrically extended $x(n)$. With this relationship, CMP is applicable considering convolution of extended $x(n)$ and $h(n)$ whose spectral impulse is real and symmetric. Stephen Martucci presented a thorough study of convolution multiplication properties of discrete trigonometric transforms (DTT's) in [2], which derived the properties for all types of DCT's and DST's under the definition of symmetric convolution. It is also shown that it is feasible to use CMP's of symmetric convolution and DTT's to perform linear convolution. The method requires a different preprocessing according to the specific symmetry type of filter and also zero-padding is needed. Based on [2]'s work, V.G. Reju *et al.* [3] proposed circular convolution using DCT and DST in which DCT-II/DST-II is used as forward transform and DCT-I/DST-I is used for inversion. Linear convolution can be implemented by padding zeros. They removed the limits to filter's symmetry type utilizing the fact any one dimensional discrete signal can be split into symmetrical and anti-symmetrical sequences. But it is just for one dimensional convolution filtering. Recently in [4], K. Suresh further extended CMP's of DTT's to MDCT/MDST domain where their implement of filtering is near exact for symmetric filters and approximate for non-symmetric filters. All the techniques mentioned above are based on Martucci's systemic study of CMP's of DTT's. When considering two dimensional convolution filtering, none of these methods can really be perfectly extended into block-based 2-D DCT-II domain since convolution result in DCT-II domain abruptly changes into DCT-I domain, and symmetrical extension and zero-padding are needed in addition in previous study. In [5], R. Kresch and N. Merhav proposed a fast algorithm for 8×8 block based DCT-II domain filtering. An efficient cosine to sine transform (CST) is also proposed for time saving in [5] and the authors showed that linear convolution with any type of separable filter can be realized in the transform domain. But when a two dimensional filter is asymmetric and inseparable, all of these method above are disable. In this paper, we propose a novel method of efficient linear filtering directly in DCT domain. By applying SVD to an original filter, we decompose it into several weighted separable sub-filters. The number of these separable sub-filters is equal to the rank of the original filter, and each sub-filter weight is correlative to the corresponding eigenvalue achieving by SVD. Based on these separable

sub-filters, we do fast linear filtering in DCT domain. At last, we combine these sub-filtering results. Compared with current methods, ours has two advantages: (1) It is applicable to any type of two-dimensional filters, regardless of their separability and symmetry. (2) It is perfectly suitable to parallel or distributed computing, which will improve the computing speed greatly. To demonstrate our scheme, we did Gabor filtering directly in DCT domain.

The rest paper is organized as follows. In Section 2, mathematical derivation of convolution filtering in DCT-II domain is given. Linear filtering with asymmetric and inseparable Gabor filters is described in Section 3. Results of experiment for validation are presented together with discussion on the computational efficiency in the same section. Section 4 is our conclusion.

2. OUR METHODOLOGY

2.1. Mathematical Preliminaries

The 8-point 2-D DCT-II transforms an 8×8 block $x = \{x(m, n)\}_{m, n=0}^7$ in the spatial domain into DCT-II domain $X = \{X(k, l)\}_{k, l=0}^7$, and the transformation can be denoted in matrix form $X^c = Cx C^T$ with the definition of $C = \{c(k, n)\}_{k, n=0}^7$, where

$$c(k, n) = \frac{\sigma(k)}{2} \cos\left(\frac{2n+1}{2} \cdot k\pi\right) \quad (1)$$

$(\sigma(k) = 1, \text{ except } \sigma(0) = 1/\sqrt{2})$

Similarly, 2-D DST-II is denoted as $X^s = Sx S^T$ with the definition of $S = \{s(k, n)\}_{k, n=1}^8$, where

$$s(k, n) = \frac{\mu(k)}{2} \sin\left(\frac{2n-1}{16} \cdot k\pi\right) \quad (2)$$

$(\mu(k) = 1, \text{ except } \mu(8) = 1/\sqrt{2})$

Theorem of SVD[8]: If $A \in R^{h \times w}$ ($h > w$ without loss of generality), and $rank(A) = r$, always has the SVD as

$$A = USV^T \quad (3)$$

where $U = [u_1, u_2, \dots, u_h] \in R^{h \times h}$, $V = [v_1, v_2, \dots, v_w] \in R^{w \times w}$, and $S = (D \ 0)^T \in R^{h \times w}$ with $D = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_r)$, $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_r \geq 0$. Also, we can formulate the SVD in another way as

$$A = \sum_{i=1}^r \lambda_i u_i v_i^T \quad (4)$$

where λ^2 is the eigenvalue of both AA^T and $A^T A$, u_i and v_i are the corresponding eigenvectors of AA^T and $A^T A$, respectively. One property of SVD is that any matrix A can be reconstructed using part of its eigenvalues and eigenvectors in the inverse process.

$$\hat{A} = \sum_{i=1}^n \lambda_i u_i v_i^T \quad n < r \quad (5)$$

where n denotes that we use the first n largest eigenvalues and corresponding eigenvectors to reconstruct A and a explicit mathematical equation exist to show the error as

$$E = \sum_{i=1}^h \sum_{j=1}^w (\hat{A} - A)^2 = \sum_{i=n+1}^r \lambda_i \quad (6)$$

which means that reconstruction error is sum of the rest unused λ_i .

2.2. Proposed Scheme

Filtering or convolution of an input image $\{I(i, j)\}$ and a filter with impulse response $\{f(i, j)\}$ results in an output image $\{J(i, j)\}$ given by:

$$J(i, j) = \sum_{i'} \sum_{j'} f(i', j') I(i - i', j - j') \quad (7)$$

Using SVD to the filter, we get $f = \sum_{k=1}^r \lambda_k u_k v_k^T$, where r is the rank of the filter. u_k and v_k are the eigenvectors of $f \cdot f^T$ and $f^T \cdot f$, respectively and λ_k^2 are their mutual eigenvalues. So for each pixel value, we have

$$f(i', j') = \sum_{k=1}^r \lambda_k u_k(i') v_k^T(j') \quad (8)$$

Incorporating SVD of $f(i', j')$ in (7), the expression can be written as:

$$J(i, j) = \sum_{i'} \sum_{j'} \sum_{k=1}^r \lambda_k u_k(i') v_k^T(j') I(i - i', j - j') \quad (9)$$

Then switching the order of summation yields

$$J(i, j) = \sum_{k=1}^r \lambda_k \sum_{i'} \sum_{j'} u_k(i') v_k^T(j') I(i - i', j - j') \quad (10)$$

In this formula, "subfilter" is defined as $f_k^{sub}(i, j)$ to substitute for $u_k(i') v_k^T(j')$, i.e.

$$f_k^{sub}(i', j') = u_k(i') v_k^T(j') \quad (11)$$

After SVD, the original filter $f(i', j')$ is decomposed into several subfilters, and each subfilter can be factorized as the multiplication of a column vector $\{u_k(i')\}$ and a row vector $\{v_k^T(j')\}$, which means all the subfilters are separable. Incorporating (11) in (10), the convolution filtering process can be denoted as

$$J(i, j) = \sum_{k=1}^r \lambda_k \sum_{i'} \sum_{j'} f_k^{sub}(i', j') I(i - i', j - j')$$

$$= \sum_{k=1}^r \lambda_k J_k^{sub}(i, j) \quad (12)$$

where $J_k^{sub}(i, j)$ represents the k^{th} sub filtering process corresponding to λ_k , i.e.

$$J_k^{sub}(i, j) = \sum_{i'} \sum_{j'} f_k^{sub}(i', j') I(i-i', j-j') \quad (13)$$

For each sub process, it is a convolution filtering with a separable filter whose factorized row vector and column vector are one dimensional sequence with supports $M^- \leq i \leq M^+, N^- \leq j \leq N^+$ respectively. It can be written as

$$J_k^{sub}(i, j) = \sum_{i'=M^-}^{M^+} u_k(i') \sum_{j'=N^-}^{N^+} v_k^T(j') I(i-i', j-j') \quad (14)$$

Namely, one can first perform a one dimensional convolution on each row with horizontal filter component (HFC) $\{v_k^T(j')\}$, and then another convolution on each column with vertical filter component (VFC) $\{u_k(i')\}$. Assuming that $|M^+|, |M^-|, |N^+|$ and $|N^-|$ do not exceed 8, we can write the formula in matrix form

$$y = V \begin{pmatrix} x_{NW} & x_N & x_{NE} \\ x_W & x & x_E \\ x_{SW} & x_S & x_{SE} \end{pmatrix} \cdot H^T \quad (15)$$

where y is a 8×8 block of filtering result in spatial domain, and V is defined as

$$\begin{pmatrix} v_8 & v_7 & \dots & v_1 & v_0 & v_{-1} & \dots & v_{-7} & v_{-8} & 0 & \dots & 0 \\ 0 & v_8 & v_7 & \dots & v_1 & v_0 & v_{-1} & \dots & v_{-7} & v_{-8} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & v_8 & v_7 & \dots & v_1 & v_0 & v_{-1} & \dots & v_{-7} & v_{-8} \end{pmatrix} \quad (16)$$

where $(v_{-8}, v_{-7}, \dots, v_0, v_1, v_2, \dots, v_8)^T = \{u_k(i')\}$, and H is similarly defined as

$$\begin{pmatrix} h_8 & h_7 & \dots & h_1 & h_0 & h_{-1} & \dots & h_{-7} & h_{-8} & 0 & \dots & 0 \\ 0 & h_8 & h_7 & \dots & h_1 & h_0 & h_{-1} & \dots & h_{-7} & h_{-8} & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & h_8 & h_7 & \dots & h_1 & h_0 & h_{-1} & \dots & h_{-7} & h_{-8} \end{pmatrix} \quad (17)$$

where $(h_{-8}, h_{-7}, \dots, h_0, h_1, h_2, \dots, h_8)^T = \{v_k(j')\}$. If we rewrite V and H as

$$V = [V_1 \ V_2 \ V_3] \quad (18)$$

$$H = [H_1 \ H_2 \ H_3] \quad (19)$$

then the filtering process can be presented as a 8×8 block-based operation

$$y = \sum_{i=1}^3 \sum_{j=1}^3 V_i x_{ij} H_j^t \quad (20)$$

where x_{ij} equals to the 8×8 block-based spatial value in the corresponding location (i,j) (e.g. $x_{12} = x_N, x_{23} = x_{SE}$). At last, pre-multiplied by C and post-multiplied by C^t in both sides, the operation (20) is easily transformed into DCT domain

$$Y^c = C \cdot y \cdot C^T = \sum_{i=1}^3 \sum_{j=1}^3 V_i^c X_{ij}^c H_j^{ct} \quad (21)$$

where, X_{ij}^c is 8×8 block-based DCT coefficients of x_{ij} . Fast computing methods for (21) are provided in [5],[7]. As discussed in [5], filtering with separable filters can be efficiently implemented. Therefore, convolution filtering consisting of sub filtering processes with separable filters can be also fast implemented.

3. EXPERIMENTS

Gabor filters are very useful in image processing such as face recognition [9]. In this session, we demonstrate our scheme by doing Gabor filtering in DCT domain. We select an asymmetric and inseparable Gabor filter, which don't satisfy the requirement of current methods, in our experiment. A Gabor filter is a linear filter whose impulse response is defined by a harmonic function multiplied by a Gaussian function

$$g(x, y; \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x^2 + \gamma^2 y^2}{2\sigma^2}\right) \cos\left(2\pi \frac{x'}{\lambda} + \psi\right) \quad (22)$$

where $x' = x \cos \theta + y \sin \theta$ and $y' = -x \sin \theta + y \cos \theta$.

In equation (22), λ represents the wavelength of the cosine factor, θ represents the orientation of the normal to the parallel stripes of a Gabor function, and ψ, σ, γ represent the phase offset, the sigma of the Gaussian envelope, and the spatial aspect ratio respectively.

From the definition of two dimensional Gabor filter, we can see that all of them are inseparable only except when $\theta = k\pi/2, k=0,1,2L$ i.e. only Gabor filters with the horizontal or vertical orientation are separable. Here we choose a Gabor filter with an arbitrary orientation ($\theta = \pi/4$ without loss of generality, as shown in the first block of figure 1 (a)), to implement convolution filtering in DCT domain using the scheme proposed in Section 2.

Experiments for validation give the results presented in figure 1. The input $\{I(i, j)\}$ is a 256×256 Lena's image. As the proposed scheme sets, we first use a 17×17 Gabor filter for SVD, since the rank of it is 17, seventeen subfilters are shown in figure 1 (a). All subfilters can be used to exactly reconstruct the original filter through addition, see figure 2 (a), (b). In figure 1 (b), results of sub convolution filtering with each corresponding subfilter are shown (after IDCT) and it is validated that the summation of sub filtering results is exactly the same as the convolution done in spatial domain, see figure 2 (c), (d). We carried out these experiments using Matlab2009a on a computer with the configuration 1.6GHz CPU, 2048MB RAM. Table 1 gives the comparison of the time consumed using traditional method in spatial domain and that using our scheme in DCT domain. The comparison shows that image processing in DCT domain saves more than 80% of the time. Furthermore, the proposed scheme is not just for Gabor

filter with arbitrary orientation, it can be applied with any other filter as well.

Filtering	Spatial domain	DCT domain
Time	0.750s	0.109s

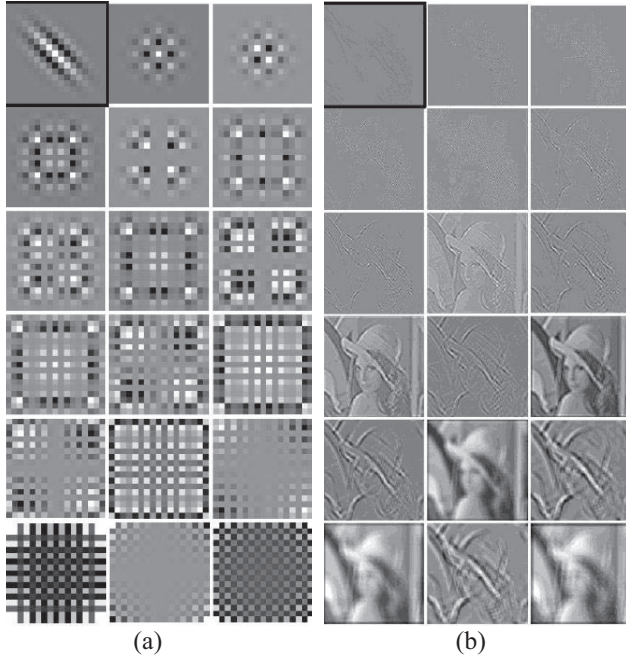


Figure 1. Gabor filter ($\theta = \pi/4$) and its sub-filters decomposed by SVD. (a) is original filter (highlight by block) and its sub-filters. (b) is convolution filtering results with corresponding filters in DCT domain. Note that these results are shown after IDCT.

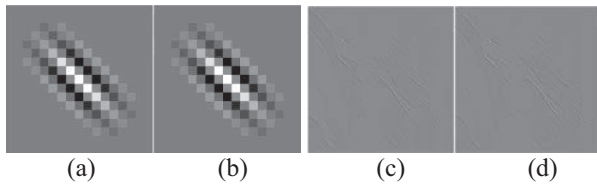


Figure 2. Reconstruction of Gabor filters and combination of its sub-filtering results. (a) is the original Gabor filter. (b) is the reconstruction of sub-filters. (c) is the filtering result in spatial domain. (d) is filtering result in DCT domain.

On the other hand, after SVD, subfilters are independent to each other, and only the input image is mutually used. So we can make the input images shared in the server, then each sub process turns independent as well. Due to the specialness of each sub convolution filtering process, paralleled or distributed computing can be well applied to improve the computational efficiency. Hence, whatever the rank of a filter is, it is able to achieve the efficiency as that of convolution with only one subfilter. (For Matlab, Distributed Computing Toolbox (DCT) exists for improving computational efficiency.)

4. CONCLUSIONS

In this work, we have proposed a scheme for effective linear filtering in DCT domain with filters which are not required to be separable, symmetrical or anti-symmetrical. Singular Value Decomposition is adopted to tackle the previous bottleneck, by decomposing an original filter into several separable sub-filters. To our best knowledge, this is the first work that proposes an integrated solution for any type of filter to realize fast convolution filtering in DCT domain. By taking arbitrarily directional Gabor filtering as an example, we show that our solution gives the same convolution results in DCT domain as those in spatial domain. Furthermore, our proposed method is very suitable to paralleled or distributed computing, which will improve the computational efficiency.

5. ACKNOWLEDGEMENT

This work was supported by the National Natural Science Foundation of China under contract No. 60933013, National High Technology Research and Development Program of China (863) under contract No. 2010ZX03004-003, and the Science Research Fund of University of Science and Technology of China for Young Scholars.

6. REFERENCES

- [1] B. Chitprasert, K. R. Rao, "Discrete cosine transform filtering", *IEEE Transactions on Signal Processing*, Vol.19(3), pp. 233-245, March 1990.
- [2] Stephen A. Martucci, "Symmetric convolution and the discrete sine and cosine transforms", *IEEE Transactions on Signal Processing*, Vol.42(5), pp.1038-1051, May 1994.
- [3] V. G. Reju, Soo Ngee Koh, Ing Yann Soon, "Convolution using discrete sine and cosine transform", *IEEE Signal Processing Letters*, Vol.14(7), pp.445-448, July 2007.
- [4] K. Suresh, T. V. Sreenivas, "Linear filtering in DCT IV/DST IV and MDCT/MDST", *Signal Processing*, Vol.89(6), pp.1081-1089, June 2009.
- [5] Renato Kresch, Neri Merhav, "Fast DCT domain filtering using the DCT and the DST", *IEEE transactions on Image Processing*, Vol.8(6), pp.821-833, June 1999.
- [6] Neri Merhav, Renato Kresch, "Approximate convolution using DCT coefficient multipliers", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.8(4), pp.378-385, Aug. 1998.
- [7] Changhoon Yim, "An Efficient Method for DCT-Domain Separable Symmetric 2-D Linear Filtering", *IEEE Transactions Letters on Circuits and Systems for Video Technology*, Vol.14, No. 4, April 2004.
- [8] From the free encyclopedia, Wikipedia, Singular Value Decomposition, http://en.wikipedia.org/wiki/Singular_value_decomposition, cited on June 30th, 2010.
- [9] Yanwei Pang, Lei Zhang, Mingjing Li, Zhengkai Liu, Weiying Ma, "A Novel Gabor-LDA Based Face Recognition Method", *Advances in Multimedia Information Processing*, Vol.3331/2005, pp. 352-358, 2005.