ELSEVIER

Information Processing Letters 79 (2001) 1–5

Information Processing Letters

www.elsevier.com/locate/ipl

An efficient algorithm for the Fourier transform on a compressed image in restricted quadtree and shading format

Kuo-Liang Chung [a,*], Wen-Ming Yan [b,1]

[a] Department of Information Management, Institute of Computer Science and Information Engineering, National Taiwan University of Science and Technology, No. 43, Section 4, Keelung Road, Taipei, Taiwan 10672

[b] Department of Computer Science and Information Engineering, National Taiwan University, No. 1, Section 4, Roosevelt Road, Taipei, Taiwan 106

Received 9 October 2000; received in revised form 23 October 2000
Communicated by L. Boasson

Abstract

Given a compressed image in restricted quadtree and shading format, this paper presents an efficient algorithm for the Fourier transform on the compressed image directly. The proposed algorithm takes $\mathrm{O}(K^2 \log K + N^2)$ time, where the decompressed grey image is of size $N \times N$ and K denotes the number of nodes in the restricted quadtree. The proposed algorithm is more general than the previous results of Anguh [IEEE Trans. Signal Processing 45 (1997) 2896] and Philips [IEEE Trans. Signal Processing 47 (1999) 2059] since in their restricted quadtree format, each quadrant is of constant grey value. © 2001 Elsevier Science B.V. All rights reserved.

Keywords: Algorithms; Fourier transform; Gouraud shading; Quadtree; Compressed image

1. Introduction

Suppose we have one grey image with size $2^n \times 2^n$ $(= N \times N)$. In [2,6], the Fourier transform on the image can be performed in $\mathrm{O}(N^2 \log N)$ time. In [1, 5], the $N \times N$ image is first partitioned into $2^s \times 2^s$ $(= S \times S)$ squares, $s \leqslant n$, where each square is of size $2^{n-s} \times 2^{n-s}$ and each pixel within the square has the same grey level. Using the subsampling technique, the subsampled image with size $2^s \times 2^s$ is called the image in the restricted quadtree format [1,5]. Although both authors present two different algorithms for Fourier transform, their efficient algorithms have the same time complexity, say $\mathrm{O}(S^2 \log S + N^2)$. In their compressed image domain, the constructed quadtree is restricted and is a complete quadtree. The experimental results [3] reveal that combining the quadtree structure and Gouraud shading technique [4] has a better compression effect. That is, given the same grey image of size $N \times N$, we have $K \leqslant S$ when comparing the compressed image with size $K \times K$ in the restricted quadtree and shading format and the compressed image with size $S \times S$ in the restricted quadtree format. The motivation of this research is to design an efficient algorithm for performing the Fourier transform on the compressed image in the restricted quadtree and shading format. Since the compressed domain considered in this research is different from that in the previous

[*] Corresponding author, supported by NSC89-2213-E011-062.
E-mail addresses: klchung@cs.ntust.edu.tw (K.-L. Chung), ganboon@csie.ntu.edu.tw (W.-M. Yan).
[1] Supported by NSC87-2119-M002-006.

algorithms [1,5], our proposed algorithm is different from the previous algorithms.

Under the compressed image in restricted quadtree and shading format, this paper presents an efficient algorithm for the Fourier transform and the proposed algorithm takes $O(K^2 \log K + N^2)$ time. The proposed algorithm has the same time complexity as that in the previous algorithms [1,5], but the compressed image domain is more general than in the previous algorithms. However, the compressed image in the restricted quadtree and shading format considered in this paper is a special case of the one in quadtree and shading format [3]. In [3], the constructed quadtree may be incomplete and it is still an open problem to design an efficient algorithm for Fourier transform on such a compressed image.

2. The compressed image in restricted quadtree and shading format

Following the shading concept used in [3], the given image is first augmented by duplicating the original last column and the original last row to become of size $(N+1) \times (N+1)$. Then the augmented image is partitioned into $2^k \times 2^k = K \times K$ overlapping homogeneous squares, i.e. subimages, as few as possible, where each square is of size $(2^{n-k}+1) \times (2^{n-k}+1) = (M+1) \times (M+1)$. Here, $N = K \times M$. For each square, we only save the grey levels of its four corners. Each corner's grey level of one square is shared by the neighboring three corners in the neighboring squares. Thus, any homogeneous square shares its top edge with the northern homogeneous square's bottom edge, and so on. The formal definition of a homogeneous square [3] will be given in next paragraph. Using Gouraud shading method [4], the grey levels of the pixels within one square can be interpolated using the the four corners' grey levels of that square. When compared to the compressed image in quadtree format [1,5], the purpose of making the blocks overlap can alleviate the blockiness effect [3] when employing the shading technique and leads to better image quality. Throughout this paper, the subsampled image in the restricted quadtree and shading format is called the subsampled image without confusion.

In one subimage, we know that there are four corners, namely, the left-bottom corner, the right-bottom corner, the left-upper corner, and the right-upper corner. The grey levels of the pixels within one square can be interpolated using the the four corners' grey values at positions (x_1, y_1), (x_2, y_1), (x_1, y_2), and (x_2, y_2) associated with four grey levels are g_1, g_2, g_3, and g_4, respectively. Here, $x_2 = x_1 + M$ and $y_2 = y_1 + M$. The estimated grey level of the pixel at (x, y) within the subimage is calculated as

$$f(x,y) = g_5 + \frac{g_6 - g_5}{y_2 - y_1}(y - y_1), \tag{1}$$

where

$$g_5 = g_1 + \frac{g_2 - g_1}{x_2 - x_1}(x - x_1) \quad \text{and}$$

$$g_6 = g_3 + \frac{g_4 - g_3}{x_2 - x_1}(x - x_1).$$

Given a specified error tolerance ε, if the image quality condition

$$\left|g(x,y) - f(x,y)\right| \leqslant \varepsilon$$

holds for all the estimated pixels at position (x, y) in the subimage, $x_1 \leqslant x < x_2$ and $y_1 \leqslant y < y_2$, then the subimage, i.e., square, is homogeneous. Here, $g(x, y)$ denotes the real grey level of the pixel at position (x, y). The resulting subsampled image consists of all these homogeneous squares. From Eq. (1), we have

$$\begin{aligned} f(x,y) &= c_0 + c_1(x - x_1) + c_2(y - y_1) \\ &\quad + c_3(x - x_1)(y - y_1), \end{aligned} \tag{2}$$

where

$$\begin{aligned} c_0 &= g_1, \\ c_1 &= \frac{g_2 - g_1}{x_2 - x_1}, \\ c_2 &= \frac{g_3 - g_1}{y_2 - y_1}, \quad \text{and} \\ c_3 &= \frac{g_4 - g_3 - g_2 + g_1}{(x_2 - x_1)(y_2 - y_1)}. \end{aligned}$$

Using the above restricted quadtree and shading approach to compress the given image with size $N \times N$, we obtain the compressed subsampled image with size $K \times K$. Among these $K \times K$ squares, for each square, only four corners' grey values are required to be stored. Totally there are $4K^2$ grey values to be stored and these $4K^2$ grey values will be used as the input of our proposed algorithm for the Fourier transform. Following the same assumption in [1,5], we omit the preprocessing time for preparing the subsampled image.

3. The proposed algorithm

For convenience, we let $0 \leqslant k_x, k_y < 2^k$, $x_1 = k_x M$, $x_2 = k_x M + M$, $y_1 = k_y M$, and $y_2 = k_y M + M$. According to the Gouraud shading method mentioned in Eq. (1), the grey level $f(x_1 + l_x, y_1 + l_y)$, $0 \leqslant l_x, l_y < M$, at position $(x_1 + l_x, y_1 + l_y)$ is interpolated using the four corners' grey levels, $g_1 = g(x_1, y_1)$, $g_2 = g(x_2, y_1)$, $g_3 = g(x_1, y_2)$, and $g_4 = g(x_2, y_2)$. Since $x_2 - x_1 = y_2 - y_1 = M$, by (2) if we let

$$\begin{aligned} c_0(k_x,k_y) &= g_1 = g(x_1,y_1),\\ c_1(k_x,k_y) &= \frac{g_2-g_1}{x_2-x_1} = \frac{g(x_2,y_1)-g(x_1,y_1)}{M},\\ c_2(k_x,k_y) &= \frac{g_3-g_1}{y_2-y_1} = \frac{g(x_1,y_2)-g(x_1,y_1)}{M},\\ c_3(k_x,k_y) &= \frac{g_4-g_3-g_2+g_1}{(y_2-y_1)(x_2-x_1)}\\ &= \frac{g_4-g_3-g_2+g_1}{M^2}, \end{aligned} \tag{3}$$

then we have

$$\begin{aligned} &f(k_xM+l_x, k_yM+l_y)\\ &\quad = f(x_1+l_x, y_1+l_y)\\ &\quad = c_0(k_x,k_y) + c_1(k_x,k_y)l_x\\ &\qquad + c_2(k_x,k_y)l_y + c_3(k_x,k_y)l_xl_y \end{aligned}$$

for $0 \leqslant l_x, l_y < M$. Instead of using the original given image with grey levels $g(x, y)$, the approximate image with grey levels $f(x, y)$ is used to help the computation of the $N \times N$ 2D Fourier transform and we have

$$F(u,v) = \sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\exp\left(\frac{-2\pi \mathrm{j}(ux+vy)}{N}\right).$$

Associated with the resulting subsampled image, we have

$$\begin{aligned} F(u,v) &= \sum_{k_x=0}^{K-1}\sum_{l_x=0}^{M-1}\sum_{k_y=0}^{K-1}\sum_{l_y=0}^{M-1} f(k_xM+l_x, k_yM+l_y)\\ &\quad \times \exp\left(\frac{-2\pi \mathrm{j}[u(k_xM+l_x)+v(k_yM+l_y)]}{N}\right)\\ &= \sum_{k_x=0}^{K-1}\sum_{k_y=0}^{K-1}\sum_{l_x=0}^{M-1}\sum_{l_y=0}^{M-1} f(k_xM+l_x, k_yM+l_y)\\ &\quad \times \exp\left(\frac{-2\pi \mathrm{j}[u(k_xM+l_x)+v(k_yM+l_y)]}{N}\right). \end{aligned}$$

Let

$$\begin{aligned} h(k_x,k_y) &= \sum_{l_x=0}^{M-1}\sum_{l_y=0}^{M-1} f(k_xM+l_x, k_yM+l_y)\\ &\quad \times \exp\left(\frac{-2\pi \mathrm{j}[ul_x+vl_y]}{N}\right), \end{aligned}$$

then we have

$$\begin{aligned} F(u,v) &= \sum_{k_x=0}^{K-1}\sum_{k_y=0}^{K-1} h(k_x,k_y)\\ &\quad \times \exp\left(\frac{-2\pi \mathrm{j}[uk_xM+vk_yM]}{N}\right)\\ &= \sum_{k_x=0}^{K-1}\sum_{k_y=0}^{K-1} h(k_x,k_y)\\ &\quad \times \exp\left(\frac{-2\pi \mathrm{j}[uk_x+vk_y]}{K}\right). \end{aligned} \tag{4}$$

Previously, we know that

$$\begin{aligned} &f(k_xM+l_x, k_yM+l_y)\\ &\quad = c_0(k_x,k_y) + c_1(k_x,k_y)l_x\\ &\qquad + c_2(k_x,k_y)l_y + c_3(k_x,k_y)l_xl_y. \end{aligned}$$

We thus have

$$\begin{aligned} h(k_x,k_y) &= \sum_{l_x=0}^{M-1}\sum_{l_y=0}^{M-1} \Big[c_0(k_x,k_y) + c_1(k_x,k_y)l_x\\ &\qquad + c_2(k_x,k_y)l_y + c_3(k_x,k_y)l_xl_y\Big]\\ &\quad \times \exp\left(\frac{-2\pi \mathrm{j}[ul_x+vl_y]}{N}\right)\\ &= c_0(k_x,k_y)S_0(u,v) + c_1(k_x,k_y)S_1(u,v)\\ &\quad + c_2(k_x,k_y)S_2(u,v) + c_3(k_x,k_y)S_3(u,v), \end{aligned} \tag{5}$$

where

$$\begin{aligned} S_0(u,v) &= \sum_{l_x=0}^{M-1}\sum_{l_y=0}^{M-1}\exp\left(\frac{-2\pi \mathrm{j}[ul_x+vl_y]}{N}\right)\\ &= \sum_{l_x=0}^{M-1}\exp\left(\frac{-2\pi \mathrm{j}ul_x}{N}\right)\sum_{l_y=0}^{M-1}\exp\left(\frac{-2\pi \mathrm{j}vl_y}{N}\right), \end{aligned}$$

$$\begin{aligned}
S_1(u,v) &= \sum_{l_x=0}^{M-1}\sum_{l_y=0}^{M-1} l_x \exp\left(\frac{-2\pi \mathrm{j}[ul_x+vl_y]}{N}\right)\\
&= \sum_{l_x=0}^{M-1} l_x \exp\left(\frac{-2\pi \mathrm{j}ul_x}{N}\right)\sum_{l_y=0}^{M-1}\exp\left(\frac{-2\pi \mathrm{j}vl_y}{N}\right),\\
S_2(u,v) &= \sum_{l_x=0}^{M-1}\sum_{l_y=0}^{M-1} l_y \exp\left(\frac{-2\pi \mathrm{j}[ul_x+vl_y]}{N}\right)\\
&= \sum_{l_x=0}^{M-1} \exp(\frac{-2\pi \mathrm{j}ul_x}{N})\sum_{l_y=0}^{M-1} l_y\exp\left(\frac{-2\pi \mathrm{j}vl_y}{N}\right),\\
S_3(u,v) &= \sum_{l_x=0}^{M-1}\sum_{l_y=0}^{M-1} l_x l_y \exp\left(\frac{-2\pi \mathrm{j}[ul_x+vl_y]}{N}\right)\\
&= \sum_{l_x=0}^{M-1} l_x \exp\left(\frac{-2\pi \mathrm{j}ul_x}{N}\right)\\
&\quad\times\sum_{l_y=0}^{M-1} l_y\exp\left(\frac{-2\pi \mathrm{j}vl_y}{N}\right).
\end{aligned}$$

Further, let

$$\begin{aligned}
A(t) &= \sum_{x=0}^{M-1}\exp\left(\frac{-2\pi \mathrm{j}tx}{N}\right) \quad \text{and}\\
B(t) &= \sum_{x=0}^{M-1} x\exp\left(\frac{-2\pi \mathrm{j}tx}{N}\right),
\end{aligned}$$

then we have

$$\begin{aligned}
S_0(u,v) &= A(u)A(v),\\
S_1(u,v) &= B(u)A(v),\\
S_2(u,v) &= A(u)B(v),\\
S_3(u,v) &= B(u)B(v).
\end{aligned}\tag{6}$$

We now need the following two straightforward and well-known identities to derive the proposed algorithm.

Lemma 1.

$$S(r)=\sum_{x=0}^{M-1} r^x = \begin{cases}\frac{1-r^M}{1-r} & \textit{when } r\neq 1;\\ M & \textit{when } r=1.\end{cases}$$

Lemma 2.

$$T(r)=\sum_{x=0}^{M-1} xr^x = \begin{cases}\frac{r(1-r^M)}{(1-r)^2}-\frac{Mr^M}{1-r} & \textit{when } r\neq 1;\\ \frac{M(M-1)}{2} & \textit{when } r=1.\end{cases}$$

From Lemmas 1 and 2, let $r=\exp(\frac{-2\pi \mathrm{j}t}{N})$, then we have

$$A(t)=S(r)=S\left(\exp\left(\frac{-2\pi \mathrm{j}t}{N}\right)\right) \quad \text{and}$$

$$B(t)=T(r)=T\left(\exp\left(\frac{-2\pi \mathrm{j}t}{N}\right)\right).$$

From Eqs. (5) and (6), we have

$$\begin{aligned}
h(k_x,k_y) &= c_0(k_x,k_y)S_0(u,v)+c_1(k_x,k_y)S_1(u,v)\\
&\quad + c_2(k_x,k_y)S_2(u,v)+c_3(k_x,k_y)S_3(u,v)\\
&= c_0(k_x,k_y)A(u)A(v)+c_1(k_x,k_y)B(u)A(v)\\
&\quad + c_2(k_x,k_y)A(u)B(v)+c_3(k_x,k_y)B(u)B(v).
\end{aligned}$$

From Eq. (4), it yields to

$$\begin{aligned}
&F(u,v)\\
&= \sum_{k_x=0}^{K-1}\sum_{k_y=0}^{K-1} h(k_x,k_y)\exp\left(\frac{-2\pi \mathrm{j}[uk_x+vk_y]}{K}\right)\\
&= A(u)A(v)\sum_{k_x=0}^{K-1}\sum_{k_y=0}^{K-1} c_0(k_x,k_y)\exp\left(\frac{-2\pi \mathrm{j}[uk_x+vk_y]}{K}\right)\\
&\quad + B(u)A(v)\sum_{k_x=0}^{K-1}\sum_{k_y=0}^{K-1} c_1(k_x,k_y)\exp\left(\frac{-2\pi \mathrm{j}[uk_x+vk_y]}{K}\right)\\
&\quad + A(u)B(v)\sum_{k_x=0}^{K-1}\sum_{k_y=0}^{K-1} c_2(k_x,k_y)\exp\left(\frac{-2\pi \mathrm{j}[uk_x+vk_y]}{K}\right)\\
&\quad + B(u)B(v)\sum_{k_x=0}^{K-1}\sum_{k_y=0}^{K-1} c_3(k_x,k_y)\exp\left(\frac{-2\pi \mathrm{j}[uk_x+vk_y]}{K}\right).
\end{aligned}\tag{7}$$

Let

$$\begin{aligned}
&C_0(u,v)\\
&= \sum_{k_x=0}^{K-1}\sum_{k_y=0}^{K-1} c_0(k_x,k_y)\exp\left(\frac{-2\pi \mathrm{j}[uk_x+vk_y]}{K}\right),\\
&C_1(u,v)\\
&= \sum_{k_x=0}^{K-1}\sum_{k_y=0}^{K-1} c_1(k_x,k_y)\exp\left(\frac{-2\pi \mathrm{j}[uk_x+vk_y]}{K}\right),
\end{aligned}$$

$$C_2(u,v) = \sum_{k_x=0}^{K-1}\sum_{k_y=0}^{K-1} c_2(k_x,k_y)\exp\left(\frac{-2\pi \mathrm{j}[uk_x+vk_y]}{K}\right),$$

$$C_3(u,v) = \sum_{k_x=0}^{K-1}\sum_{k_y=0}^{K-1} c_3(k_x,k_y)\exp\left(\frac{-2\pi \mathrm{j}[uk_x+vk_y]}{K}\right),$$

then from (7), it yields to

$$F(u,v) = A(v)\big[A(u)C_0(u,v)+B(u)C_1(u,v)\big] + B(v)\big[A(u)C_2(u,v)+B(u)C_3(u,v)\big]. \tag{8}$$

Our proposed algorithm for the Fourier transform associated with the compressed image consists of the following four steps:

Step 1. By Eq. (3), we compute $c_1(k_x,k_y)$, $c_2(k_x,k_y)$, $c_3(k_x,k_y)$, and $c_4(k_x,k_y)$ for $0 \leqslant k_x,k_y < K$, this step takes $\mathrm{O}(K^2)$ time.

Step 2. We compute the four $K \times K$ FFTs, namely, $C_0(u,v)$, $C_1(u,v)$, $C_2(u,v)$, and $C_3(u,v)$ for $0 \leqslant u,v < K$. Applying the conventional $K \times K$ FFT algorithm [2,6] four times, this step takes $\mathrm{O}(K^2 \log K)$ time.

Step 3. Compute $A(u)$ and $B(u)$ for $0 \leqslant u,v < N$. From Lemmas 1 and 2, this step takes $\mathrm{O}(N)$ time. In fact, $A(v)$ ($B(v)$) can be obtained from the value of $A(u)$ ($B(u)$) directly.

Step 4. From Eq. (8), the Fourier transform

$$F(u,v) = A(v)\big[A(u)C_0(u,v)+B(u)C_1(u,v)\big] + B(v)\big[A(u)C_2(u,v)+B(u)C_3(u,v)\big]$$

for $0 \leqslant u,v < N$ can be obtained in $\mathrm{O}(N^2)$ time.

From Step 1 to Step 4, the total time complexity required in the proposed algorithm is $\mathrm{O}(K^2\log K + N^2)$ $(= \mathrm{O}(K^2 + K^2\log K + N + N^2))$. We thus have the main result.

Theorem 1. *On the compressed image in the restricted quadtree and shading format, the proposed algorithm can perform the Fourier transform in* $\mathrm{O}(K^2\log K + N^2)$ *time, where the decompressed grey image is of size* $N \times N$ *and* K *denotes the number of nodes in the restricted quadtree.*

The proposed algorithm is more general than the previous results [1,5] since in their restricted quadtree format with size $S \times S$, each quadrant is of constant grey value. The experimental results in [3] reveal $K \leqslant S$. This better compression effect implies that the proposed algorithm may have a computation-saving effect. However, in [3], the constructed quadtree may be incomplete, so it is still an open problem to design an efficient algorithm for the Fourier transform on such a compressed image.

Acknowledgement

We appreciate the referees and the Editor Professor Luc Boasson for their valuable comments that lead to the improved representation of the paper.

References

[1] M. Anguh, Quadtree and symmetry in FFT computation of digital images, IEEE Trans. Signal Processing 45 (12) (1997) 2896–2899.

[2] E.O. Brigham, The Fast Fourier Transform, Prentice-Hall, Englewood Cliffs, NJ, 1974.

[3] K.L. Chung, J.G. Wu, Improved image compression using S-tree and shading approach, IEEE Trans. Commun. 48 (5) (2000) 748–751.

[4] J.D. Foley, A.V. Dam, S.K. Feiner, J.F. Hughes, Computer Graphics, Principle, and Practice, 2nd edn., Addison-Wesley, Reading, MA, 1990.

[5] W. Philips, On computing the FFT of digital images in quadtree format, IEEE Trans. Signal Processing 47 (7) (1999) 2059–2060.

[6] C. van Loan, Computational Frameworks for the Fast Fourier Transform, SIAM Press, Philadelphia, PA, 1992.