離散分數訊號轉換之研究(III)

Discrete Fractional Signal Transforms (III)

計畫編號： NSC90-2213-E-002-040

執行期限： 90 年 8 月 1 日至 91 年 7 月 31 日

主持人： 貝蘇章　　台灣大學電機系教授

中文摘要

DFT 有良好的效能以及快速運算法。但是當我們裝置 DFT 時，我們需用到浮點乘法。在這裡，我們將介紹整數離散傅立葉轉換 (ITFT) 。它與 DFT 相近，不過由於它的轉換矩陣中所有數字皆為整數，因此在裝置時，只需用到整數乘法。它大大地簡化了裝置的複雜度，尤其適合於用 VLSI 來裝置。它的許多性質與原來的 DFT 相似，例如位移不變性質。在許多應用上，我們可用 ITFT 來取代 DFT。

關鍵詞： 整數離散傅立葉轉換，離散傅立葉轉換，離散正交轉換

ABSTRACT

DFT has the good quality of performance and fast algorithm. But when we implement the DFT, we will require the floating-points multiplication. In this project, we will introduce the *integer Fourier transform* (ITFT). ITFT is approximated to the DFT, but all the entries in the transform matrix are integer numbers. So it only requires the fixed-points multiplication, and the implementation can be much simplified, especially for VLSI. This new transform will work very similar as the original DFT, for example, the transform results are similar and the shifting-invariant property is also preserved for ITFT. In many applications, we can use ITFT to replace DFT.

Keywords : integer discrete Fourier transform, discrete Fourier transform, discrete orthogonal transform

一、 介紹

Because of the direct relations with frequency spectrum and the fast algorithm, DFT is a very popular tool for signal processing. But when we implement the DFT, some floating-points multiplication operations are required. The implementation of floating-points multiplication is usually trouble and time-consuming. Besides, for the computer, it requires the floating-points processor, and will be more complex and expensive.

In this project, we will derive the *integer Fourier transform* (ITFT). It approximates to the DFT, but all the entries in the transform matrix are integer numbers. We can implement the ITFT without any the floating-points multiplication operations because there are no non-integer entries. In section 2, we will introduce the general method to derive the integer transform from some real discrete transform. This method is the generalization and the modification of the method introduced by Cham [1][3] (He used his method to derive the integer cosine transform). In section 3, we will derive the 8-points ITFT. Then, in section 4, we will discuss the performance and the property of ITFT. We will show ITFT remains almost all the performance quality of DFT. In section 5, we make a conclusion.

二、 推導整數轉換的一般方式

Integer transform is the discrete transform that all the entries in the transform matrix are integers. When we try to find the integer transform analogous to some real transform, we can follow the algorithm described as below. Here we use **A** to denote the transform matrix of the original real transform, use **B** to denote the forward transform matrix of the integer transform, and use **D** to denote the inverse integer transform matrix.

(1) Find the ***symmetry relation and the equality relation*** for each row of the original real transform, and find the *sign* of the entries of the transform matrix.

(2) ***Forming the prototype matrix of the forward integer transform*** from the relations obtained in the step 1. The prototype must satisfy the following rules:

(a) If for the original transform matrix,

$A(m, a_1) = C \cdot A(m, a_2)$ where $C = 1, -1, j, -j$

then for the forward and inverse integer transform, the equality relations as below must be satisfied:

$B(m, a_1) = C \cdot B(m, a_2)$ $D(m, a_1) = C \cdot D(m, a_2)$

(b) If for the original transform matrix,

$A(m_1, n_1) = 0$ $A(m_2, n_2) > 0$ $A(m_3, n_3) < 0$

then for the forward and inverse integer transform,

$B(m_1, n_1) = 0$ $B(m_2, n_2) > 0$ $B(m_3, n_3) < 0$

$D(m_1, n_1) = 0$ $D(m_2, n_2) > 0$ $D(m_3, n_3) < 0$

From these rules, we assign the unknowns in each entry of the prototype matrix. To be convenient, all the unknowns are constrained to be positive and real integers. If the prototype matrix has too many unknowns, we can try to make some unknowns be the same, or make some unknowns to be 1.

(3) ***Forming the prototype matrix of the inverse integer transform***. The prototype of transform matrix of the inverse transform is of the same form as the forward transform, but we use another set of unknowns.

(4) ***Constraints for orthogonality***. In this step, we search for the requirements to make the transform matrices of the forward and inverse transform to be orthogonal, and make the inner product of the same rows to be the values of 2^k (k is integer):

$$\sum_{k=0}^{N-1} B(m,k)\overline{D(n,k)} = R_m \delta_{mn} \quad \text{where } R_m = 2^k \qquad (1)$$

From this, we obtain the *equality constraints* for the unknowns of the prototypes matrices.

(5) ***Constraints for inequality***. In this step, we find the inequality relations among the unknowns of the prototypes matrices from the inequality relations for each row of the original non-integer transform matrix. That is, if for the original transform,

$A(m, n_1) > A(m, n_2)$

then for the forward and inverse integer transforms,

$B(m, n_1) > B(m, n_2)$ $\quad D(m, n_1) > D(m, n_2)$

These will be the *inequality constraints* for the unknowns.

(6) ***Assign the values for all the unknowns***. At last, we assign the values of unknowns. They must be real, positive integer numbers, and satisfy the constraints obtained in steps (4), (5).

We note, the transform matrix **B** of the forward transform and the transform matrix of the inverse transform **D** would be different. Thus, if $X(m)$ is the forward integer transform of $x(n)$,

$$X(m) = \sum_{n=0}^{N-1} B(m,n) \cdot x(n) \tag{2}$$

Then we can recover $x(n)$ from $X(m)$ by

$$x(n) = \sum_{m=0}^{N-1} D^*(m,n) \cdot R_m^{-1} \cdot X(m) \tag{3}$$

where * represents the conjugation, and R_m is defined in Eq. (1). Eqs. (2), (3) can also be written as

$$\mathbf{X} = \mathbf{B} \cdot \mathbf{x} \qquad \mathbf{x} = \mathbf{D}^{\mathrm{H}} \mathbf{C}^{-1} \mathbf{X} \tag{4}$$

where $^{\mathrm{H}}$ is the Hermitian operation. The method introduced in [1] makes the transform matrix for the forward and the inverse integer transform to be the same, and can't avoid the floating-points multiplication for the inverse transform (since the values of R_m in Eq. (1) would not be the form of 2^k). Here we allow the forward and the inverse integer transform to be different. This enables us to fully avoid the floating-points multiplication operations, no matter for the forward or inverse transform. But since **B**, **D** are of the same forms, so the structures of the implementation of the forward and inverse transform are basically the same, except for the direction would be reversed and the parameters are different.

From the process introduced above, we can assure the integer transform we derived are very similar to the original non-integer transform, because (1) the symmetry, equality relations for each row, (2) the sign of each entry, (3) the orthogonality property, (4) the inequality relations for each row have been preserved. Thus, many properties of the original transform (such as the shift-invariant property of DFT) will be almost preserved.

三、 八點整數離散傅立葉轉換

The original 8-points DFT is:

$$F(m,n) = \exp(-j\,mn\pi/4) \quad m, n \in [0, 1, \ldots 7] \tag{5}$$

To derive the 8-points integer Fourier transform (ITFT), we first form its prototype from the equality relation and the sign of each entry of the original 8-points DFT (steps 1, 2 in section 2). We list the equality relations of the original 8-points DFT as below:

row	row 0	row 1	row 2	row 3	row 4	row 5	row 6	row 7
G=1	C=1		C=-j		C=-1		C=j	
G=2	C=1	C=-j	C=-1	C=j	C=1	C=-j	C=-1	C=j
G=4	C=1	C=-1	C=1	C=-1	C=1	C=-1	C=1	C=-1

Table 1 the equality relation of the original 8-points DFT

The values of G and C mean the m^{th} row of the original 8-points DFT will have the following relations

$$F(m, n \oplus G) = C \cdot F(m,n) \qquad \text{if } n \oplus G > n \tag{6}$$

$$F(m, n \oplus G) = C^{-1} \cdot F(m,n) \qquad \text{if } n \oplus G < n \tag{7}$$

where $\oplus$ is the exclusive-OR addition:

$$A \oplus B = \sum_{i=0}^{2} a_i \cdot 2^i \oplus \sum_{i=0}^{2} b_i \cdot 2^i = \sum_{i=0}^{2} (a_i \, XOR b_i) \cdot 2^i$$

where a_i, b_i = 0, 1. We note, from the step 2 described in section 2, the 8-points ITFT must also have the same equality and symmetry relations for each row as the original 8-points DFT, so Eqs. (6), (7) must also hold for the 8-points ITFT. Then together with the sign of the entries in the 8-points DFT matrix, we can construct the prototype matrix of the 8-points forward ITFT as:

$$\mathbf{FI_P} = \begin{bmatrix}
g_1 & g_1 & g_1 & g_1 & g_1 & g_1 & g_1 & g_1 \\
a_1 & a_2-ja_2 & -ja_1 & -a_2-ja_2 & -a_1 & -a_2+ja_2 & ja_1 & a_2+ja_2 \\
g_2 & -jg_2 & -g_2 & jg_2 & g_2 & -jg_2 & -g_2 & jg_2 \\
b_1 & -b_2-jb_2 & jb_1 & b_2-jb_2 & -b_1 & b_2+jb_2 & -jb_1 & -b_2+jb_2 \\
g_3 & -g_3 & g_3 & -g_3 & g_3 & -g_3 & g_3 & -g_3 \\
c_1 & -c_2+jc_2 & -jc_1 & c_2+jc_2 & -c_1 & c_2-jc_2 & jc_1 & -c_2-jc_2 \\
g_4 & jg_4 & -g_4 & -jg_4 & g_4 & jg_4 & -g_4 & -jg_4 \\
d_1 & d_2+jd_2 & jd_1 & -d_2+jd_2 & -d_1 & -d_2-jd_2 & -jd_1 & d_2-jd_2
\end{bmatrix} \tag{8}$$

We assign the unknowns for the real part and image part of the entries separately to assure all the unknowns will be real numbers. In this prototype matrix, there are totally 12 unknowns. The amount of unknowns seems to be too much, so we will set some unknowns to be 1 and some unknowns to be equal. We set

$$g_1 = g_2 = g_3 = g_4 = 1, \tag{9}$$

$$b_1 = c_1, \quad b_2 = c_2, \qquad d_1 = a_1, \quad d_2 = a_2 \tag{10}$$

Then the prototype is simplified as:

$$\mathbf{FI_P} = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
a_1 & a_2-ja_2 & -ja_1 & -a_2-ja_2 & -a_1 & -a_2+ja_2 & ja_1 & a_2+ja_2 \\
1 & -j & -1 & j & 1 & -j & -1 & j \\
c_1 & -c_2-jc_2 & jc_1 & c_2-jc_2 & -c_1 & c_2+jc_2 & -jc_1 & -c_2+jc_2 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
c_1 & -c_2+jc_2 & -jc_1 & c_2+jc_2 & -c_1 & c_2-jc_2 & jc_1 & -c_2-jc_2 \\
1 & j & -1 & -j & 1 & j & -1 & -j \\
a_1 & a_2+ja_2 & ja_1 & -a_2+ja_2 & -a_1 & -a_2-ja_2 & -ja_1 & a_2-ja_2
\end{bmatrix} \tag{11}$$

and there are only 4 unknowns. But we must assure the ITFT can still be derived after the simplification, otherwise we must remove some of the equality relations Eqs. (9), (10). We note, in Eq. (11), the m^{th} row of the prototype matrix will be the conjugation of the $(7-m)^{\text{th}}$ row, as the original DFT.

Then we form the prototype for the inverse ITFT. It is similar to Eq. (11), but $\{a_1, a_2, c_1, c_2\}$ are changed as $\{a_3, a_4, c_3, c_4\}$:

$$\mathbf{IFI_P} = \begin{bmatrix}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
a_3 & a_4-ja_4 & -ja_3 & -a_4-ja_4 & -a_3 & -a_4+ja_4 & ja_3 & a_4+ja_4 \\
1 & -j & -1 & j & 1 & -j & -1 & j \\
c_3 & -c_4-jc_4 & jc_3 & c_4-jc_4 & -c_3 & c_4+jc_4 & -jc_3 & -c_4+jc_4 \\
1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\
c_3 & -c_4+jc_4 & -jc_3 & c_4+jc_4 & -c_3 & c_4-jc_4 & jc_3 & -c_4-jc_4 \\
1 & j & -1 & -j & 1 & j & -1 & -j \\
a_3 & a_4+ja_4 & ja_3 & -a_4+ja_4 & -a_3 & -a_4-ja_4 & -ja_3 & a_4-ja_4
\end{bmatrix} \tag{12}$$

There are 4 unknowns for the inverse ITFT, so there are totally 8 unknowns for ITFT.

Then we search the constraints for the unknowns. First, we try to find the orthogonality constraints to make each row of the forward, inverse ITFT to satisfy the Eq. (1).

From the prototype of the forward, inverse ITFT, we find except for {row 1, 5}, {row 5, 1}, {row 3, 7}, and {row 7, 3}, all pairs of different rows will be orthogonal to each other. And calculating the inner product of {row 1, 5}, {row 5, 1}, {row 3, 7}, and {row 7, 3} directly, we find if we want these pairs to be orthogonal, then the following constraints must be satisfied:

(i) $a_1 c_3 = 2 a_2 c_4$, (ii) $a_3 c_1 = 2 a_4 c_2$

Then, from the requirement that the inner product of the same row must be the power of 2, we obtain the constraint as:

(iii) $a_1 a_3 + 2 a_2 a_4 = 2^k$, (iv) $c_1 c_3 + 2 c_2 c_4 = 2^h$

where k, h are integer numbers. The constraints of (i), (ii), (iii), (iv) will be the equality constraints of the unknowns.

Then, from the inequality relations for each row of original DFT matrix, we obtain the inequality constraints for the 8-points integer DFT as:

(v) $a_1 \geq a_2$ (vi) $c_1 \geq c_2$ (vii) $a_3 \geq a_4$ (viii) $c_3 \geq c_4$

These are the inequality constraints. Thus we have obtained all the constraints for unknowns.

Then we assign the values of unknowns. Since there are 8 unknowns and only 4 equality constraints, so there are infinite choices for the values of unknowns. But to search the values of unknowns to satisfy all the constraints, especially to satisfy the constraints (iii), (iv), is a difficult task. We introduce a process as below. This process will make the work of searching for the values of unknowns to be more efficient.

(a) Choose the values of a_1, a_2 such that a_1, a_2 are integer numbers and

$$2 \cdot a_2 \geq a_1 \geq a_2 \qquad (13)$$

(b) Find the integer values of c_1, c_2 such that $c_1 \geq c_2$, and

$$a_1 \cdot c_2 + a_2 \cdot c_1 = 2^n \qquad n \text{ is integer} \qquad (14)$$

(c) Set the value of a_3, a_4, c_3, c_4 as

$$c_3 = 2a_2, \quad c_4 = a_1, \quad a_3 = 2c_2, \quad a_4 = c_1 \qquad (15)$$

Then the values of unknowns are all obtained. We list some possible choices of the unknowns as below:

(1) a_1=2, a_2=1, c_1=2, c_2=1, a_3=2, a_4=2, c_3=2, c_4=2

This is the smallest, simplest integer solution for the unknowns. And in this case, the value of R_m in Eq. (1) is:

$$R_0 = R_2 = R_4 = R_6 = 2^3, \quad R_1 = R_3 = R_5 = R_7 = 2^4.$$

(2) a_1=7, a_2=5, c_1=13, c_2=9, a_3=18, a_4=13, c_3=10, c_4=7

We note, when we choose the parameters as the values listed above, the ratios of $a_1 : a_2$, $c_1 : c_2$, $a_3 : a_4$, $c_3 : c_4$, are all near to 1.414:1, which is ratio of the original discrete Fourier transform. If R_m is defined as Eq. (1), then in this case

$$R_0 = R_2 = R_4 = R_6 = 2^3, \quad R_1 = R_3 = R_5 = R_7 = 2^{10}.$$

We can implement the forward/inverse 8–points integer Fourier transform (8-points ITFT) in the following ways:

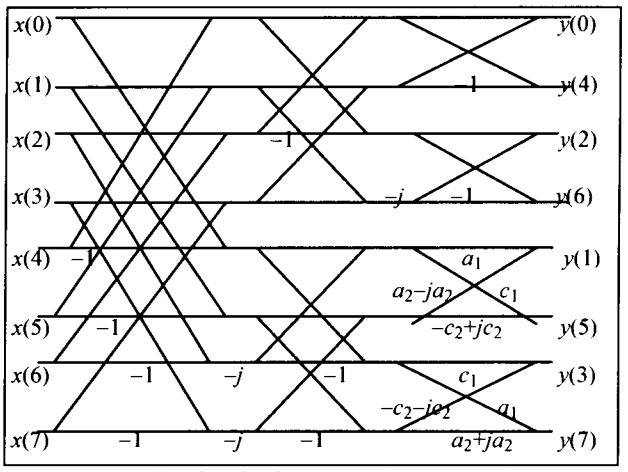


Fig. 1 Decimation–in–frequency implementation for the forward 8–point integer Fourier transform (ITFT)

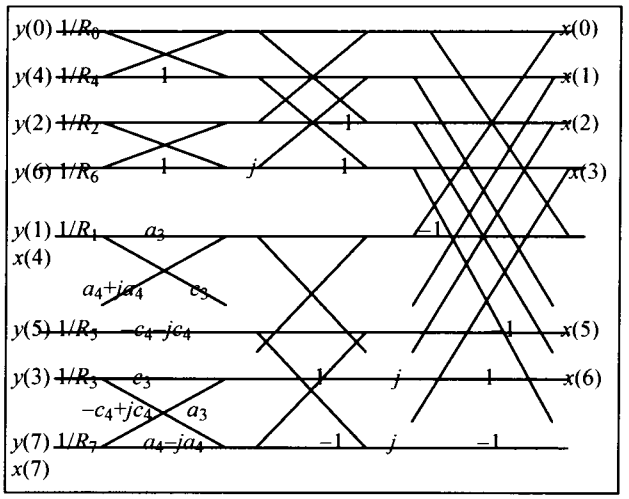


Fig. 2 Decimation–in–frequency implementation for the inverse 8–point integer Fourier transform

We find, for the 8-points ITFT, the number of the multiplication operations required is:

- forward/inverse transform: 8 fixed-points multi. operations
- normalization: 8 fixed-points multi. operations
- total: 24 fixed-points multiplication operations

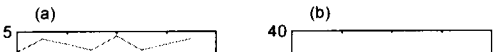
And for the original 8-points DFT,

- forward/inverse transform: 2 floating-points multi. ops.
- normalization: 8 fixed-points multi. operations
- total: 4 floating-points, 8 fixed-points multi. operations

That is, we replace 4 floating-points multiplication operations with 16 fixed-points multiplication operations. The cost for doing 16 fixed-points multiplication operations is much less than the cost for doing 4 floating-points multiplication operation. Thus, 8-points ITFT will be much faster than the original 8-points DFT. Besides, we can further reduce the number of multiplication operations by setting a_1=c_1, or a_2=c_2, or a_3=c_3, or a_4=c_4, and each of them will save 2 fixed-points multiplication operations.

The advantages of the ITFT are the simpler implementation and saving the computation time, and its performance is much like the original DFT (see section 4). But some properties, such as the convolution theorem, can just be approximated, not be

preserved. We can choose the ratios of $a_1 : a_2$, $c_1 : c_2$, $a_3 : a_4$, c_3 :



preserved. We can choose the ratios of $a_1 : a_2$, $c_1 : c_2$, $a_3 : a_4$, $c_3 : c_4$ near to 1.414:1, then the performance of the ITFT will be more approximated to the original DFT.

四、 整數離散傅立葉轉換的效能

We will see the performance of the 8-points integer Fourier transform (8-points ITFT). Here the parameters we choose are a_1=7, a_2=5, c_1=13, c_2=9, a_3=18, a_4=13, c_3=10, c_4=7.

We first see the transform result. We choose 2 inputs:

$\mathbf{x}_1$ = [2, 3, 4, 5, 4, 5, 2, 3]. (16)

$\mathbf{x}_2$ = [2.8, 4.3-0.6i, 3.7+0.9i, 3.1-0.6i, 4.6, 3.1+0.6i, 3.7-0.9i, 4.3+0.6i] (17)

Then we do the original 8- points DFT, and plot the results in Fig. 3(c), 3(d). And then, we do the 8-points ITFT, and plot the results in Fig. 3(e), 3(f). We will normalize the transform results for comparison. That is, for the transform result of the ITFT

$$X(m) = \sum_{n=0}^{7} FI(m,n) \cdot x(n) \tag{18}$$

We will normalize $X(m)$ as below in Fig. 3(e), 3(f):

$$\tilde{X}(m) = X(m) / FI(m,0). \tag{19}$$

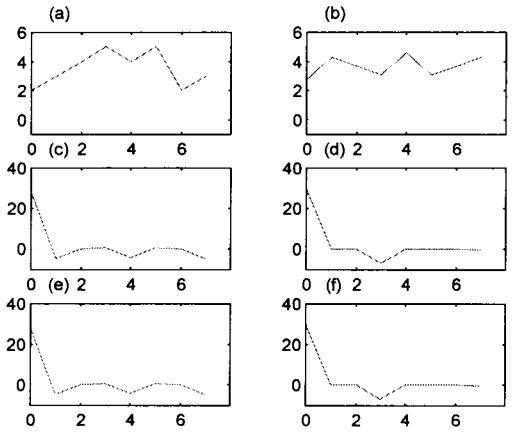


Fig. 3 The transform results of the original DFT and ITFT. (a) $x_1[n]$, (b) $x_2[n]$, (c) original DFT for $x_1[n]$, (d) original DFT for $x_2[n]$, (e) ITFT for $x_1[n]$, (f) ITFT for $x_2[n]$. (heavy lines): real part, (light lines): imaginary part

We find, the normalized transform results for the 8-points ITFT are very similar as the transform results for the original 8-points DFT. Also, we note that input $\mathbf{x}_2$ is the combination of a low frequency component and a high frequency component. As these 2 components can be separated by both the original DFT and the IDFT. So the IDFT can also be used for the filter design.

Then, we see the displacement property of the IDFT. Here we use the displacement of input $x_2[n]$ (see Eq. (19)) as the input:

$x_3[n] = x2[((n+1))_8]$ $x_4[n] = x_2[((n+2))_8]$ (20)

And we plot $x_2[n]$, $x_3[n]$, $x_4[n]$ in Fig. 4(a), 4(c), 4(e). Then we do the 8-points ITFT for $x_2[n]$, $x_3[n]$, $x_4[n]$, and plot the amplitudes of the normalized results in Fig. 4(b), 4(d), 4(f).

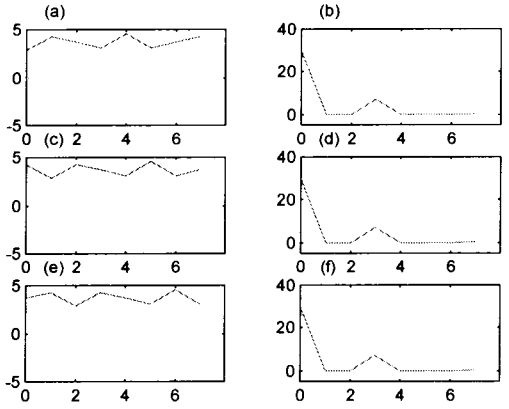


Fig. 4 The space-invariant property for the ITFT. (a) $x_2[n]$, (b) $|X_2[m]|$, (c) $x_3[n]$, (d) $|X_3[m]|$, (e) $x_4[n]$, (f) $|X_4[m]|$, where $X_2(m)$, $X_3(m)$, $X_4(m)$ are the ITFT of $x_2(n)$, $x_3(n)$, $x4(n)$.

We find, for the ITFT, the displacement property also keeps. The signal after displacement will have the same transform amplitude as the original signal. When we compare the phase, there is an interesting phenomenon. We find, if $X_2(m)$, $X_3(m)$, $X_4(m)$ are the transform results of $x_2(n)$, $x_3(n)$, $x4(n)$, then

angle($\mathbf{X}_3$)−angle($\mathbf{X}_2$) = [0, -45, -90, -135, 0, 135, -270, -315]

angle($\mathbf{X}_4$)−angle($\mathbf{X}_2$) = [0, -90, -180, -270, 0, -90, -180, -270]

This is exactly the same as the original DFT.

五、 結論

In Sec. 3, we have used the method introduced in Sec. 2 to derive the 8 points integer Fourier transform (8 points ITFT). In fact, we can also derive the ITFT for other number of points (such as the 6-points ITFT). We can also derive other types of integer transforms, such as the integer cosine, Hartley transforms.

For the integer transform, we can replace all the floating-points multiplication operations with the fixed-points multiplication operations, so the integer transform is very efficient. If the datum we want to process are all integer number, using the integer transforms is especially efficient. The concept of the integer transform is very new, and there are only a few researches about it. Because they are convenient for implementation, and almost remain the quality of original transform, so they can compete with the original discrete transform in many applications. We believe the integer transform will be very popular in the future.

參考資料

[1] W. K. Cham, *IEE Proceeding*, v. 136, n. 4, p 276-282, 1989

[2] A. V. Oppenheim, and R. W. Schafer, "*Discrete-Time Signal Processing*", Prentice-Hall, Inc., 1989

[3] T. C. J. Pang, C. S. O. Choy, C. F. Chan, and W. K. Cham, *IEEE Trans. Circuits Syst. for Viedo Tech.*, v. 9, n. 6, p. 856-860, 1999

行政院國家科學委員會九十一年度計劃執行進度報告

離散分數訊號轉換之研究 (3/3)

Discrete Fractional Signal Transforms (3/3)

計劃編號 : NSC 90-2213-E-002-040

執行期限 : 88 年 8 月 1 日至 91 年 7 月 31 日

計劃主持人 : 貝蘇章

處理方式 : 立即對外提供參考

一年後可對外提供參考

二年後可對外提供參考
(必要時，本會得展延發表時限)

執行單位 : 國立台灣大學電機工程學研究所

中 華 民 國 91 年 8 月 10 日

摘 要

近幾年來，連續傅立葉分數轉換已被廣泛的應用在各種領域，這些領域包括：時間頻率訊號分析、傅氏光學、圖形識別、訊號壓縮等等。但是數位式的離散分數傅立葉轉換仍未完全開發，由於其在訊號處理及分析方面的應用潛力極廣，因此我們提出未來三年的研究計劃針對「離散分數訊號轉換」做一整合性深入探討，包括離散分數傅立葉轉換、分散傅立葉級數、離散分數希爾伯轉換及在影像處理邊緣偵測之應用以及分散時間頻率分佈函數等，並可分別用來做雷達與時變訊號偵測等應用。

第一年：離散傅立葉分數轉換的延伸，改良及應用

1.二度空間離散傅立葉分數轉換

2.Discrete fractional Fourier transforms based on orthogonal projections.

3.快速轉換演算法的研發

4.訊號處理之應用

第二年：分數傅立葉級數及離散分數希爾伯特轉換

1.Fractional Fouier series for periodic signals

2.Discrete fractional Hilbert transform

3.Edge detection in Image Processing applications

第三年：訊號解析、處理及圖形識別應用

1.Analytic discrete fractional Fourier transform and affine Fourier transforms

2.Fractional correlation for pattern recognition

3.Fractional time-frequency distributions for radar chirp rate detection.

第一年的研究成果：

1)“ Two-dimensional discrete fractional Fourier Transform, ” by S. C. Pei and M.H.Yen, Signal Processing, vol.67, p.99-108, Feb. 1998.

2) “Discrete fractional Fourier transform based on orthogonal projections, ” by S. C. Pei and M. H. Yen and C. C. Tesng, IEEE Trans. on Signal Processing, vol.47, No.5, pp.1335 -1348, May 1999.

第二年的研究成果：

1) "Fractional Fourier series expansion for finite signals and dual extension to discrete-time fractional Fourier transform,"

by S. C. Pei, M. H. Yeh and T. L. Luo, IEEE Trans. on Signal Processing, Vol.47, No.10, pp.2883-2888, Oct. 1999.

2) "Discrete fractional Hilbert transform," by S. C. Pei, and M. H. Yeh, IEEE Trans. on Circuits and Systems, Part II : Analog and Digital Signal Processing, Vol.47,No.11,pp.1307-1311, Nov. 2000.

3) "Analytical design of maximally flat FIR fractional Hilbert transformers," by S. C. Pei and P. H. Wang, Signal Processing, vol.81, pp. 643-661, March 2001.

第三年的研究成果：

1) "Closed form discrete fractional and affine Fourier transform," by S. C. Pei, and J. J. Ding, IEEE Trans. on Signal Processing,Vol.48, No.5, pp.1338-1353, May 2000.

2) "Simplified fractional Fourier transform," by S. C. Pei and J. J. Ding, J. Opt. Soc. Am. A, Vol.17, No.12, pp.2355-2367, Dec. 2000.

3) "The integer transforms analogous to discrete trigometric transforms," by S. C. Pei and J. J. Ding, IEEE Trans. on Signal Processing, Vol.48, No.12, pp.3345-3364, Dec. 2000.

關鍵詞：連續傅立葉分散轉換、離散分數傅立葉轉換、時變訊號分析。

ABSTRACT

In recent years, continuous fractional Fourier transforms have been developed and widely used in several practical applications, such as time-frequency signal analysis, Fourier optics, pattern recognition, and signal compression etc. However discrete fractional Fourier transforms have not been developed very completely yet; Since it has very high potentiality in digital signal processing applications, we have proposed a 3-year term research project, it will concentrate on the study of several discrete fractional signal transforms, such as discrete fractional Fourier transform, discrete fractional Fourier series, discrete fractional Hilbert transform and its applications on image edge detection, and fractional time-frequency distribution function for radar and time-varying signals detection.

Keywords : Continuous Fractional Fourier Transform, Discrete Fractional Fourier Transform, Time-Varying Signal Analysis.

Final Version: SP–30277

Two-Dimensional Affine Generalized Fractional Fourier Transform

Soo-Chang Pei, Jian-Jiun Ding

Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, R. O. C.

E-mail: pei@cc.ee.ntu.edu.tw Fax: 886-2-23671909

Abstract

As 1-D Fourier transform can be extended into the 1-D fractional Fourier transform (FRFT), we can also generalize the 2-D Fourier transform. In [1], they have generalized the 2-D Fourier transform into 2-D separable FRFT (replaces each variable 1-D Fourier transform by 1-D FRFT, respectively) and 2-D separable canonical transform (further replaces FRFT by canonical transform). It has also been generalized into the 2-D unseparable Fourier transform with 4 parameters in [2]. In this paper, we will introduce the 2-D affine generalized fractional Fourier transform (AGFFT). It has even further extended the 2-D transforms described above. It is unseparable, and has totally 10 degree of freedom. We will show the 2-D AGFFT has many wonderful properties, such as the relations with the Wigner distribution, shifting-modulation operation, and the differentiation-multiplication operation. Although 2-D AGFFT form seems very complex, but in fact the complexity for the implementation will not be more than the implementation of the 2-D separable FRFT. Besides, we will also show the 2-D AGFFT extends many of the applications for the 1-D FRFT, such as the filter design, optical system analysis, image processing, and pattern recognition, and will be a very useful tool for 2-D signal processing.

I. Introduction

The **fractional Fourier transform (FRFT)** [3][4], which is the generalization of the 1-D Fourier transform, is defined as:

$$F_\alpha(u) = O_F^\alpha(f(t)) = \sqrt{\frac{1-j\cot\alpha}{2\pi}} \cdot e^{\frac{j}{2}\cdot\cot\alpha\cdot u^2} \cdot \int_{-\infty}^{\infty} e^{-j\cdot\csc\alpha\cdot u\cdot t} \cdot e^{\frac{j}{2}\cdot\cot\alpha\cdot t^2} \cdot f(t)\cdot dt. \quad (1)$$

It has the additivity property as following:

$$O_F^\beta O_F^\alpha = O_F^{\alpha+\beta}. \quad (2)$$

It has been used in many applications such as optical system analysis, filter design, solving differential equations, phase retrieval, and pattern recognition, etc.

In fact, the FRFT is the special case of the **canonical transform** [5] (also called the **special affine Fourier transform (SAFT)** [6]). The canonical transform is defined as below:

$$F_{(a,b,c,d)}(u) = O_F^{(a,b,c,d)}(f(t)) = \sqrt{\frac{1}{j2\pi b}} \cdot e^{\frac{j}{2}\cdot\frac{d}{b}\cdot u^2} \cdot \int_{-\infty}^{\infty} e^{-j\cdot\frac{u}{b}\cdot t} e^{\frac{j}{2}\cdot\frac{a}{b}\cdot t^2} \cdot f(t)\cdot dt \quad (3)$$

when $b \neq 0$,

$$F_{(a,0,c,d)}(u) = O_F^{(a,0,c,d)}(f(t)) = \sqrt{d} \cdot e^{\frac{j}{2}\cdot cd\cdot u^2} \cdot f(d\cdot u) \quad \text{when } b = 0. \quad (4)$$

And the constraint that $ad - bc = 1$ must be satisfied. The FRFT is just the special case of SAFT with $\{a, b, c, d\} = \{\cos\alpha, \sin\alpha, -\sin\alpha, \cos\alpha\}$:

$$F_\alpha(u) = \sqrt{e^{j\alpha}} \cdot F_{(\cos\alpha,\sin\alpha,-\sin\alpha,\cos\alpha)}(u). \quad (5)$$

The canonical transform has also the additivity property as:

$$O_F^{(a_2,b_2,c_2,d_2)} O_F^{(a_1,b_1,c_1,d_1)} = O_F^{(e,f,g,h)}, \quad (6)$$

where
$$\begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix} \cdot \begin{bmatrix} a_1 & b_1 \\ c_1 & d_1 \end{bmatrix}. \quad (7)$$

The canonical transform has extended the utilities of FRFT in some applications, and is a useful tool for the optical system analysis.

The FRFT and SAFT (canonical transform) defined above in Eqs. (1) and (3) are all 1–D transforms. In [1], they have generalized them from 1–D into the 2–D cases. The 2–D canonical transform they introduce is equivalent to the following equation:

$$O_F^{(a,b,c,d,a_1,b_1,c_1,d_1)}(g(x,y)) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} K_{(a,b,c,d)}(f,x)\cdot K_{(a_1,b_1,c_1,d_1)}(h,y)\cdot g(x,y)dxdy \quad (8)$$

where
$$K_{(a,b,c,d)}(f,x) = \sqrt{1/j2\pi b} \cdot e^{\frac{j}{2}\cdot\frac{d}{b}\cdot f^2} \cdot e^{-j\cdot\frac{f}{b}\cdot x} \cdot e^{\frac{j}{2}\cdot\frac{a}{b}\cdot x^2} \quad \text{when } b \neq 0, \quad (9)$$

$$K_{(a,b,c,d)}(f,x) = \sqrt{d} \cdot e^{\frac{j}{2}\cdot cd\cdot f^2} \cdot \delta(x - d\cdot f), \quad (10)$$

and $K_{(a_1,b_1,c_1,d_1)}(h,y)$ is of the same form as $K_{(a,b,c,d)}(f,x)$, and $ad-bc = 1$, $a_1d_1-b_1c_1 = 1$. That is, the 2-D canonical transform defined as Eq. (8) can be viewed as the combination of 2 independent 1-D canonical transforms. The 2–D fractional Fourier transform they introduce is the special case of the 2–D canonical transform defined above with $\{a, b, c, d\} = \{\cos\alpha, \sin\alpha, -\sin\alpha, \cos\alpha\}$ and $\{a_1, b_1, c_1, d_1\} = \{\cos\beta, \sin\beta, -\sin\beta, \cos\beta\}$. Although the 2–D canonical transform introduced by [1] has generalized the 2–D Fourier transform, but it is not general enough because it treats 2 variables independently. In this paper, we will call the 2–D fractional Fourier/canonical transforms introduced by [1] as the **2–D separable fractional Fourier/canonical transform**.

Recently, in [2], they introduce the 2–D unseparable fractional Fourier transform. This transform is the same as the 2–D separable fractional Fourier transform for

$$f\left(\frac{\cos\theta_1\, x+\sin\theta_1\, y}{\cos(\theta_1-\theta_2)}, \frac{-\sin\theta_2\, x+\cos\theta_2\, y}{\cos(\theta_1-\theta_2)}\right), \tag{11}$$

and there are totally 4 parameters (θ_1, θ_2, and the order of the FRFT for each dimension). It treats the 2 variables unseparably, and generalizes the 2–D separable fractional Fourier transform. In fact, the 2–D unseparable fractional Fourier transform [2] can be further generalized. In this paper, we will introduce a new type of generalized 2–D fractional Fourier transform, it will be much more general than the transforms introduced in [1] and [2].

In [7], they have introduced an N–D operator (the operation for the N dimensional functions) defined as below:

$$O_{\mu}^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}\left(f(\vec{\mathbf{x}})\right)$$
$$= \left(-\det(\mathbf{B})\right)^{-\frac{1}{2}} \int \exp\left(j\pi\left(\vec{\mathbf{w}}\cdot\mathbf{D}\mathbf{B}^{-1}\cdot\vec{\mathbf{w}}^{\mathbf{T}} - 2\,\vec{\mathbf{x}}\cdot\mathbf{B}^{-1}\cdot\vec{\mathbf{w}}^{\mathbf{T}} + \vec{\mathbf{x}}\cdot\mathbf{B}^{-1}\mathbf{A}\cdot\vec{\mathbf{x}}^{\mathbf{T}}\right)\right)\cdot f(\vec{\mathbf{x}})\cdot d\vec{\mathbf{x}}, \tag{12}$$

where

$$\vec{\mathbf{x}} = (x_1, x_2, \cdots\cdots, x_N) \qquad \vec{\mathbf{w}} = (w_1, w_2, \cdots\cdots, w_N).$$

In Eq. (12), **A**, **B**, **C**, **D** are all $N\times N$ matrices, and satisfy the following constraints:

$$\mathbf{A}^{\mathbf{H}}\mathbf{C} = \mathbf{C}^{\mathbf{H}}\mathbf{A}, \qquad \mathbf{B}^{\mathbf{H}}\mathbf{D} = \mathbf{D}^{\mathbf{H}}\mathbf{B}, \qquad \mathbf{A}^{\mathbf{H}}\mathbf{D} - \mathbf{C}^{\mathbf{H}}\mathbf{B} = \mathbf{I}, \tag{13}$$

or equivalently,

$$\mathbf{A}\mathbf{B}^{\mathbf{H}} = \mathbf{B}\mathbf{A}^{\mathbf{H}}, \qquad \mathbf{C}\mathbf{D}^{\mathbf{H}} = \mathbf{D}\mathbf{C}^{\mathbf{H}}, \qquad \mathbf{A}\mathbf{D}^{\mathbf{H}} - \mathbf{B}\mathbf{C}^{\mathbf{H}} = \mathbf{I}. \tag{14}$$

We use $^{\mathbf{H}}$ to denote the Hermitian operation (conjugation and transpose). The operators defined in Eq. (12) can be represented by the following $2N\times 2N$ matrix (In [7] they call it as the *metaplectic representation*):

$$\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{C} & \mathbf{D}\end{bmatrix}. \tag{15}$$

And the operation defined in Eq. (12) has the additivity as below:

$$O_{\mu}^{(\mathbf{A}',\mathbf{B}',\mathbf{C}',\mathbf{D}')} O_{\mu}^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})} = O_{\mu}^{(\mathbf{P},\mathbf{Q},\mathbf{R},\mathbf{S})}, \tag{16}$$

where

$$\begin{bmatrix} \mathbf{P} & \mathbf{Q} \\ \mathbf{R} & \mathbf{S} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{A}' & \mathbf{B}' \\ \mathbf{C}' & \mathbf{D}' \end{bmatrix}. \tag{17}$$

In this paper, we will discuss a special case of Eq. (12) with $N = 2$ (2 dimensions). We call it as the **2–D affine generalized fractional Fourier transform (2–D AGFFT)**. It is unseparable, totally has 10 degree of freedom, and is more general than the transforms defined in [1] and [2]. We will discuss it in detail, especially for its properties and implementation. We will also show AGFFT can do many things that can't be done for the 2-D separable fractional Fourier/canonical transform introduced by [1] and the 2-D unseparable FRFT introduced by [2]. In section 2, we will give the definition of the 2-D affine generalized fractional Fourier transform (AGFFT), some special cases of it, and the 2-D affine generalized fractional convolution and correlation. Then, we will discuss the properties of AGFFT in section 3. In section 4, we will discuss some efficient ways to calculate and implement this 2-D transform, and some simplified form of 2-D AGFFT. In section 5, we will discuss some applications of AGFFT, such as the 2-D filter design, and optical system analysis. Finally in section 6, we make some conclusions.

II. The 2-D Affine Generalized Fractional Fourier transform

2-1 The Definition of 2–D AGFFT

The **2-D affine generalized fractional Fourier transform (2–D AGFFT)** we define here is the special case of Eq. (12) with dimension 2:

$$\textbf{AGFFT: } O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(g(x,y)) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} K_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,x,y)\cdot g(x,y)\cdot dxdy, \tag{18}$$

where

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{bmatrix}, \quad \mathbf{D} = \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix} \tag{19}$$

represents the 16 parameters of 2-D AGFFT (Here **we restrict all the parameters to be real**), and the kernel is:

$$K_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,x,y) = \frac{1}{2\pi\sqrt{-\det(\mathbf{B})}} \cdot e^{\frac{j}{2\det(\mathbf{B})}\left(k_1\cdot f^2 + k_2\cdot f\cdot h + k_3\cdot h^2\right)} \cdot e^{\frac{j}{\det(\mathbf{B})}\left((-b_{22}f + b_{12}h)x + (b_{21}f - b_{11}h)y\right)}$$
$$\cdot e^{\frac{j}{2\det(\mathbf{B})}\left(p_1\cdot x^2 + p_2\cdot x\cdot y + p_3\cdot y^2\right)}, \tag{20}$$

where

$$k_1 = d_{11}b_{22} - d_{12}b_{21}, \qquad k_2 = 2(-d_{11}b_{12} + d_{12}b_{11}) = 2(d_{21}b_{22} - d_{22}b_{21}),$$
$$k_3 = -d_{21}b_{12} + d_{22}b_{11}, \qquad p_1 = a_{11}b_{22} - a_{21}b_{12},$$
$$p_2 = 2(a_{12}b_{22} - a_{22}b_{12}) = 2(-a_{11}b_{21} + a_{21}b_{11}), \qquad p_3 = -a_{12}b_{21} + a_{22}b_{11}. \tag{21}$$

The constraints of Eq. (13) will become the following 6 constraints:

$$a_{11}c_{12} + a_{21}c_{22} = a_{12}c_{11} + a_{22}c_{21}, \qquad b_{11}d_{12} + b_{21}d_{22} = b_{12}d_{11} + b_{22}d_{21},$$
$$a_{11}d_{11} + a_{21}d_{21} - (c_{11}b_{11} + c_{21}b_{21}) = 1, \qquad a_{12}d_{12} + a_{22}d_{22} - (c_{12}b_{12} + c_{22}d_{22}) = 1,$$
$$a_{11}d_{12} + a_{21}d_{22} = c_{11}b_{12} + c_{21}b_{22}, \qquad a_{12}d_{11} + a_{22}d_{21} = c_{12}b_{11} + c_{22}b_{21}, \tag{22}$$

or equivalently, from Eq. (14),

$$a_{11}b_{21} + a_{12}b_{22} = a_{21}b_{11} + a_{22}b_{12}, \qquad c_{11}d_{21} + c_{12}d_{22} = c_{21}d_{11} + c_{22}d_{12},$$
$$a_{11}d_{11} + a_{12}d_{12} - (b_{11}c_{11} + b_{12}c_{12}) = 1, \qquad a_{21}d_{21} + a_{22}d_{22} - (b_{21}c_{21} + b_{22}c_{22}) = 1,$$
$$a_{11}d_{21} + a_{12}d_{22} = b_{11}c_{21} + b_{12}c_{22}, \qquad a_{21}d_{11} + a_{22}d_{12} = b_{21}c_{11} + b_{22}c_{12}. \tag{23}$$

Specially, we find when **A** or **D** is an identical matrix **I**, then $b_{12}=b_{21}$, $c_{12}=c_{21}$. And when **B** or **C** is **I**, then $a_{12}=a_{21}$, $d_{12}=d_{21}$. Because there are 16 parameters and 6 constraints, so the free dimension of the 2-D AGFFT is 10. In contrast, the free dimension for 1-D canonical transform is 3, and for the 2-D separable canonical transform defined as Eq. (8) is 6 (8 parameters with 2 constraints).

Since the 2-D AGFFT has too many parameters, so in this paper we will usually use the matrix or vector notations instead of the explicit notations. We will usually use

$$\vec{\mathbf{x}} = (x, y), \qquad \vec{\mathbf{w}} = (f, h) \tag{24}$$

to denote the variables in time and frequency domains, use **A**, **B**, **C**, **D** defined as Eq. (19) to denote the 16 parameters. Besides, in this paper we will use $G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f, h)$ to denote the result of 2-D AGFFT of $g(x, y)$:

$$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(\vec{\mathbf{w}}) = G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f, h) = O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(g(x, y)) = O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(g(\vec{\mathbf{x}})). \tag{25}$$

The additivity property for the 2-D AGFFT is also the same as Eqs. (16) and (17). And the reversible property of the 2-D AGFFT is:

$$\left(O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}\right)^{-1} = O_F^{\left(\mathbf{D}^{\mathrm{T}},-\mathbf{B}^{\mathrm{T}},-\mathbf{C}^{\mathrm{T}},\mathbf{A}^{\mathrm{T}}\right)}. \tag{26}$$

We note, when $a_{12} = a_{21} = b_{12} = b_{21} = c_{12} = c_{21} = d_{12} = d_{21} = 0$, the 2-D AGFFT becomes the 2-D separable canonical transform defined as Eq. (8). More specially, when $\mathbf{A} = \mathbf{D} = \mathbf{0}$ and $\mathbf{B} = -\mathbf{C} = \mathbf{I}$, the 2-D AGFFT becomes the 2-D forward Fourier transform. And when

$$\mathbf{A} = \begin{bmatrix} \cos\alpha \cdot \cos\theta_2 & -\cos\alpha \cdot \sin\theta_1 \\ \cos\beta \cdot \sin\theta_2 & \cos\beta \cdot \cos\theta_1 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} \dfrac{\sin\alpha \cdot \cos\theta_1}{\cos(\theta_1 - \theta_2)} & \dfrac{-\sin\alpha \cdot \sin\theta_2}{\cos(\theta_1 - \theta_2)} \\ \dfrac{\sin\beta \cdot \sin\theta_1}{\cos(\theta_1 - \theta_2)} & \dfrac{\sin\beta \cdot \cos\theta_2}{\cos(\theta_1 - \theta_2)} \end{bmatrix},$$

$$\mathbf{C}=\begin{bmatrix} -\sin\alpha\cdot\cos\theta_2 & \sin\alpha\cdot\sin\theta_1 \\ -\sin\beta\cdot\sin\theta_2 & -\sin\beta\cdot\cos\theta_1 \end{bmatrix},\qquad \mathbf{D}=\begin{bmatrix} \dfrac{\cos\alpha\cdot\cos\theta_1}{\cos(\theta_1-\theta_2)} & \dfrac{-\cos\alpha\cdot\sin\theta_2}{\cos(\theta_1-\theta_2)} \\ \dfrac{\cos\beta\cdot\sin\theta_1}{\cos(\theta_1-\theta_2)} & \dfrac{\cos\beta\cdot\cos\theta_2}{\cos(\theta_1-\theta_2)} \end{bmatrix}, \quad (27)$$

the 2-D AGFFT becomes the 2-D unseparable fractional Fourier transform introduced by Sahin, Kutay, and Ozaktas [2].

We show the relations between the AGFFT and its special cases in Fig. 1. The number in the ( ) shows the degree of freedom of the corresponding transform.

2-2 The definition of 2–D AGFFT when det(B) = 0

We note, in Eq. (18), if det(**B**) = 0, then we can't apply this equation directly. In these cases, we must convert the 2-D AGFFT defined as Eqs. (18)~(21) into another form. We discuss each of these cases as below.

(1) **B** = **0**.

We note, because $\mathbf{A}^{\mathrm{T}}\mathbf{D}-\mathbf{C}^{\mathrm{T}}\mathbf{B}=\mathbf{I}$, so when **B** = **0**, $\mathbf{A}^{\mathrm{T}}\mathbf{D}=\mathbf{I}$. That is, the equality relation $\mathbf{A}=(\mathbf{D}^{\mathrm{T}})^{-1}$ must be satisfied in the case that **B** = **0**. And since

$$\begin{bmatrix} \mathbf{D}^{\mathrm{T}^{-1}} & \mathbf{0} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}=\begin{bmatrix} \mathbf{0} & -\mathbf{D}^{\mathrm{T}^{-1}} \\ \mathbf{D} & -\mathbf{C} \end{bmatrix}\cdot\begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{I} & \mathbf{0} \end{bmatrix}, \quad (28)$$

so in the case that **B** = **0**, the 2-D AGFFT can be defined as

$$G_{(\mathbf{D}^{\mathrm{T}^{-1}},\mathbf{0},\mathbf{C},\mathbf{D})}(\vec{\mathbf{w}})=\frac{\sqrt{\det(\mathbf{D})}}{4\pi^2}\cdot\exp(j\cdot\vec{\mathbf{w}}\mathbf{C}\mathbf{D}^{\mathrm{T}}\vec{\mathbf{w}}^{\mathrm{T}}/2)\cdot$$
$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\exp(j\cdot\vec{\mathbf{p}}\mathbf{D}^{\mathrm{T}}\vec{\mathbf{w}}^{\mathrm{T}}-j\cdot\vec{\mathbf{x}}\vec{\mathbf{p}}^{\mathrm{T}})\cdot g(\vec{\mathbf{x}})\cdot dxdydpdq, \quad (29)$$

where $\vec{\mathbf{p}}=(p,q)$, and $\vec{\mathbf{x}}$, $\vec{\mathbf{w}}$ are defined as Eq. (24). After some calculating, we obtain

$$G_{(\mathbf{D}^{\mathrm{T}^{-1}},\mathbf{0},\mathbf{C},\mathbf{D})}(f,h)=\sqrt{\det(\mathbf{D})}\cdot e^{\frac{j}{2}\left((c_{11}d_{11}+c_{12}d_{12})f^2+2(c_{11}d_{21}+c_{12}d_{22})fh+(c_{21}d_{21}+c_{22}d_{22})h^2\right)}$$
$$\cdot g(d_{11}f+d_{21}h,d_{12}f+d_{22}h) \quad (30)$$

This is the formula for 2-D AGFFT when **B** = **0**. We find when **C** = **0**, it is just the geometric twisting operation. And when **D** = **I**, it is just the chirp multiplication operation.

(2′) {**A**, **B**, **C**, **D**} = {**I**, **B**, **0**, **I**}, det(**B**) = **0**, **B** ≠ **0**.

Before discussing the other cases, we first discuss the formula of the 2-D AGFFT with the parameters {**A**, **B**, **C**, **D**} = {**I**, **B**, **0**, **I**}. In the case that det(**B**) ≠ **0**, we can apply Eqs. (18)~(21) directly. But in the case that det(**B**) = **0**, we must use other ways to find the formula. Because

$$\begin{bmatrix} \mathbf{I} & \mathbf{B} \\ \mathbf{0} & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & -\mathbf{I} \\ \mathbf{I} & \mathbf{0} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ -\mathbf{B} & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ -\mathbf{I} & \mathbf{0} \end{bmatrix}, \tag{31}$$

so

$$\begin{aligned} O_F^{(\mathbf{I},\mathbf{B},\mathbf{0},\mathbf{I})}(g(x,y)) &= IFT\left(O_F^{(\mathbf{I},\mathbf{0},-\mathbf{B},\mathbf{I})}(FT(g(x,y)))\right) \\ &= IFT\left(e^{-\frac{j}{2}(b_{11}f^2+2b_{12}fh+b_{22}h^2)}\right) * g(x,y). \end{aligned} \tag{32}$$

We have used the Eq. (30) and the multiplication theory of the original Fourier transform. In this paper, the definition of Fourier transform, inverse Fourier transform, and the convolution we used are as below:

$$FT(g(x,y)) = O_F^{(\mathbf{0},\mathbf{I},-\mathbf{I},\mathbf{0})}(g(x,y)) = \frac{1}{\sqrt{-1}\cdot 2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-j(fx+hy)} g(x,y)\cdot dxdy, \tag{33}$$

$$IFT(g(x,y)) = O_F^{(\mathbf{0},-\mathbf{I},\mathbf{I},\mathbf{0})}(g(x,y)) = \frac{1}{\sqrt{-1}\cdot 2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{j(fx+hy)} g(x,y)\cdot dxdy, \tag{34}$$

$$\begin{aligned} g(x,y) * q(x,y) &= IFT\left(FT(g(x,y))\cdot FT(q(x,y))\right) \\ &= \frac{1}{\sqrt{-1}\cdot 2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x-\tau, y-\eta)\cdot q(\tau,\eta)\cdot d\tau d\eta \end{aligned} \tag{35}$$

From Eqs. (32), (34), (35), we find the formula of the 2-D AGFFT with parameters {**I**, **B**, **0**, **I**} (det(**B**) = **0**, **B** ≠ **0**) can be expressed as:

① $b_{11} = 0$, $b_{12} = 0$, $b_{22} \neq 0$:

$$G_{(\mathbf{I},\mathbf{B},\mathbf{0},\mathbf{I})}(f,h) = \frac{1}{\sqrt{j2\pi\, b_{22}}}\int_{-\infty}^{\infty} e^{\frac{j}{2b_{22}}(h-y)^2}\cdot g(f,y)\cdot dy \tag{36}$$

because $$IFT\left(e^{-\frac{j}{2}(b_{11}\cdot f^2+2b_{12}\cdot f\cdot h+b_{22}\cdot h^2)}\right) = \sqrt{\frac{j2\pi}{b_{22}}}\delta(x)\, e^{\frac{j}{2b_{22}}y^2}. \tag{37}$$

② $b_{11} \neq 0$, $b_{22} \neq 0$, $b_{12}^2 = b_{11}b_{22}$:

$$G_{(\mathbf{I},\mathbf{B},\mathbf{0},\mathbf{I})}(f,h) = \sqrt{\frac{j2\pi}{b_{11}}}\int_{-\infty}^{\infty} e^{\frac{j}{2b_{11}}(f-x)^2}\cdot g(x, \frac{b_{12}}{b_{11}}(x-f)+h)\cdot dx \tag{38}$$

because $$IFT\left(e^{-\frac{j}{2}(b_{11}\cdot f^2+2b_{12}\cdot f\cdot h+b_{22}\cdot h^2)}\right) = \sqrt{\frac{j2\pi}{b_{11}}}\, e^{\frac{j}{2b_{11}}x^2}\,\delta(\frac{b_{22}}{b_{11}}x - y). \tag{39}$$

③ $b_{11} \neq 0$, $b_{22} = 0$, $b_{12} = 0$:

$$G_{(\mathbf{I},\mathbf{B},\mathbf{0},\mathbf{I})}(f,h) = \frac{1}{\sqrt{j2\pi\, b_{11}}}\int_{-\infty}^{\infty} e^{\frac{j}{2b_{11}}(f-x)^2}\cdot g(x,h)\cdot dx \tag{40}$$

because $$IFT\left(e^{-\frac{j}{2}(b_{11}\cdot f^2+2b_{12}\cdot f\cdot h+b_{22}\cdot h^2)}\right) = \sqrt{\frac{j2\pi}{b_{11}}}\, e^{\frac{j}{2b_{11}}x^2}\,\delta(y). \tag{41}$$

(2″) $\det(\mathbf{B}) = 0$, $\mathbf{B} \neq 0$, $\det(\mathbf{A}) \neq 0$.

We will generalize the case (2′), and discuss all the cases that $\det(\mathbf{B}) = 0$, $\mathbf{B} \neq 0$, $\det(\mathbf{A}) \neq 0$. In this case, because

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{CA}^{-1} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ 0 & \mathbf{D}-\mathbf{CA}^{-1}\mathbf{B} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{CA}^{-1} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{BA}^{\mathrm{T}} \\ 0 & \mathbf{DA}^{\mathrm{T}} - \mathbf{CA}^{-1}\mathbf{BA}^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{A}^{\mathrm{T}-1} \end{bmatrix}, \tag{42}$$

and from Eq. (14) (the Hermitian operations can be replaced with the transpose operations in this equation since all the parameters are real),

$$\mathbf{BA}^{\mathrm{T}} = \mathbf{AB}^{\mathrm{T}},$$

$$\mathbf{DA}^{\mathrm{T}} - \mathbf{CA}^{-1}\mathbf{BA}^{\mathrm{T}} = \mathbf{DA}^{\mathrm{T}} - \mathbf{CA}^{-1}\mathbf{AB}^{\mathrm{T}} = \mathbf{DA}^{\mathrm{T}} - \mathbf{CB}^{\mathrm{T}} = \left(\mathbf{AD}^{\mathrm{T}} - \mathbf{BC}^{\mathrm{T}}\right)^{\mathrm{T}} = \mathbf{I}, \tag{43}$$

so we obtain

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{CA}^{-1} & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{I} & \mathbf{AB}^{\mathrm{T}} \\ 0 & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{A} & 0 \\ 0 & \mathbf{A}^{\mathrm{T}-1} \end{bmatrix} \qquad (\det(\mathbf{A}) \neq 0). \tag{44}$$

Then together with Eqs. (30) and (32), we obtain

$$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)$$

$$= \frac{1}{2\pi\sqrt{-\det(\mathbf{A})}} \exp\left(j(\vec{\mathbf{f}} \cdot \mathbf{CA}^{-1} \cdot \vec{\mathbf{f}}^{\mathrm{T}}/2)\right) \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} r(f-x, h-y)\, g(\vec{\mathbf{x}} \cdot \mathbf{A}^{\mathrm{T}-1}) \cdot dxdy, \tag{45}$$

where $$r(x,y) = IFT\left(\exp\left(-j(\vec{\mathbf{f}} \cdot \mathbf{AB}^{\mathrm{T}} \cdot \vec{\mathbf{f}}^{\mathrm{T}})/2\right)\right). \tag{46}$$

And the explicit formula for $r(x, y)$ can be calculated from Eq. (36), (38), or (40).

(3) $\det(\mathbf{B}) = 0$, $\mathbf{B} \neq 0$, $\det(\mathbf{A}) = 0$, $\mathbf{A} \neq 0$, $\det(\mathbf{D}) \neq 0$.

Since

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{AD}^{\mathrm{T}} & \mathbf{BD}^{-1} \\ \mathbf{CD}^{\mathrm{T}} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{D}^{\mathrm{T}-1} & 0 \\ 0 & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{BD}^{-1} \\ 0 & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{AD}^{\mathrm{T}} - \mathbf{BD}^{-1}\mathbf{CD}^{\mathrm{T}} & 0 \\ \mathbf{CD}^{\mathrm{T}} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{D}^{\mathrm{T}-1} & 0 \\ 0 & \mathbf{D} \end{bmatrix},$$

and from Eqs. (13), (14),

$$\mathbf{BD}^{-1}\mathbf{CD}^{\mathrm{T}} = \mathbf{BD}^{-1}\mathbf{DC}^{\mathrm{T}} = \mathbf{BC}^{\mathrm{T}}, \quad \mathbf{AD}^{\mathrm{T}} - \mathbf{BD}^{-1}\mathbf{CD}^{\mathrm{T}} = \mathbf{AD}^{\mathrm{T}} - \mathbf{BC}^{\mathrm{T}} = \mathbf{I}, \tag{47}$$

so

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{BD}^{-1} \\ 0 & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{CD}^{\mathrm{T}} & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{D}^{\mathrm{T}-1} & 0 \\ 0 & \mathbf{D} \end{bmatrix}. \tag{48}$$

Thus, in this case,

$$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h) = \frac{\sqrt{-\det(\mathbf{D})}}{2\pi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} r(f-x, h-y) \exp\left(j\frac{(\vec{\mathbf{x}} \cdot \mathbf{CD}^{\mathrm{T}} \cdot \vec{\mathbf{x}}^{\mathrm{T}})}{2}\right) g(\vec{\mathbf{x}} \cdot \mathbf{D})\, dxdy \tag{49}$$

where $$r(x,y) = IFT\left(\exp\left(-j(\vec{\mathbf{f}} \cdot \mathbf{BD}^{-1} \cdot \vec{\mathbf{f}}^{\mathrm{T}})/2\right)\right). \tag{50}$$

And the explicit formula of $r(x, y)$ can be calculated from Eq. (36), (38), or (40).

(4) $\det(\mathbf{B}) = 0$, $\mathbf{B} \neq 0$, $\det(\mathbf{A}) = 0$, $\mathbf{A} \neq 0$, $\det(\mathbf{D}) = 0$, $\mathbf{D} \neq 0$, $\det(\mathbf{C}) \neq 0$.

Since

$$O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})} = IFT\left(O_F^{(\mathbf{C},\mathbf{D},-\mathbf{A},-\mathbf{B})}\right), \tag{51}$$

and from the Eq. (45), we obtain in this case,

$$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h) = \frac{1}{4\pi^2\sqrt{\det(\mathbf{C})}} \cdot \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \exp\left(j \cdot \vec{\mathbf{f}} \cdot \vec{\mathbf{w}} - j \cdot (\vec{\mathbf{w}} \cdot \mathbf{A}\mathbf{C}^{-1} \cdot \vec{\mathbf{w}}^T)/2\right) \cdot \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} r(w-x, v-y) \cdot g(\vec{\mathbf{x}} \cdot \mathbf{C}^{\mathbf{T}^{-1}}) \cdot dxdydwdv, \tag{52}$$

where $$r(x,y) = IFT\left(\exp\left(-j(\vec{\mathbf{f}} \cdot \mathbf{C}\mathbf{D}^{\mathbf{T}} \cdot \vec{\mathbf{f}}^{\mathbf{T}})/2\right)\right), \qquad \vec{\mathbf{w}} = (u,v). \tag{53}$$

(5) $\det(\mathbf{B}) = 0$, $\mathbf{B} \neq 0$, $\det(\mathbf{A}) = \det(\mathbf{D}) = \det(\mathbf{C}) = 0$ ($\mathbf{A} \neq 0$, $\mathbf{D} \neq 0$ must be satisfied).

This case seldom occurs. One example is the 1-D Fourier transform in the y-axis ($a_{11} = b_{22} = -c_{22} = d_{11} = 1$, others = 0). We can show in this case all the 2-D AGFFT can be decomposed as:

$$O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(g(x,y)) = O_F^{(\Lambda^{-1},0,0,\Lambda)}\left(O_F^{(\mathbf{I},0,\Phi,\mathbf{I})}\left(O_F^{(\mathbf{A}',\mathbf{B}',\mathbf{C}',\mathbf{D}')}(g(x,y))\right)\right) \tag{54}$$

where $$\mathbf{A}' = \begin{bmatrix} 1 & a \\ 0 & 0 \end{bmatrix}, \quad \mathbf{B}' = \begin{bmatrix} 0 & 0 \\ -a & 1 \end{bmatrix}, \quad \mathbf{C}' = \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix}, \quad \mathbf{D}' = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \tag{55}$$

$$O_F^{(\mathbf{A}',\mathbf{B}',\mathbf{C}',\mathbf{D}')}(g(x,y)) = FT_y(g(x-ay,y)), \tag{56}$$

and FT_y means the 1-D Fourier transform along y-axis. Thus, when $\det(\mathbf{A}) = \det(\mathbf{B}) = \det(\mathbf{C}) = \det(\mathbf{D}) = 0$ and $\mathbf{A} \neq 0$, $\mathbf{B} \neq 0$, $\mathbf{D} \neq 0$, the 2-D AGFFT can be decomposed as the combination of a 1-D Fourier transform for the variable y on $g(x-ay, y)$, a multiplication operation with the quadratic phase function (i.e., $\exp(j(s_1x^2+s_2xy+s_3y^2))$), and a geometric twisting operation.

2-3 Basic Operations

We will discuss the components of 2-D AGFFT. We find there are at most 10 independent basic operations for the 2-D AGFFT, and they correspond to the 10 free dimensions of 2-D AGFFT. We list one example of the independent basic operations set in the following, and denote them by $O_{F(1)}$, $O_{F(2)}$, ..., $O_{F(10)}$.

(1) chirp multiplication for the x–axis ($a_{11} = a_{22} = d_{11} = d_{22} = 1$, $c_{11} = \tau_x$, others = 0):

$$O_{F(1)}^{\tau_x}(g(x,y)) = e^{j\tau_x x^2/2} \cdot g(x,y). \tag{57}$$

(2) chirp convolution for the x-axis ($a_{11} = a_{22} = d_{11} = d_{22} = 1$, $b_{11} = \eta_x$, others = 0):

$$O_{F(2)}^{\eta_x}(g(x,y)) = \frac{e^{jx^2/2\eta_x}}{\sqrt{j\eta_x}} *_{\text{along } x\text{-axis}} g(x,y). \tag{58}$$

(3) scaling along the x–axis ($a_{11} = 1/d_{11} = \rho_x$, $a_{22} = d_{22} = 1$, others = 0):

$$O_{F(3)}^{\rho_x}(g(x,y)) = g(x/\rho_x, y)/\sqrt{\rho_x}. \tag{59}$$

(4) chirp multiplication for the y-axis ($a_{11} = a_{22} = d_{11} = d_{22} = 1$, $c_{22} = \tau_y$, others = 0):

$$O_{F(4)}^{\tau_y}\left(g(x,y)\right) = e^{j\tau_y y^2/2} \cdot g(x,y). \tag{60}$$

(5) chirp convolution for the y-axis ($a_{11} = a_{22} = d_{11} = d_{22} = 1$, $b_{22} = \eta_y$, others = 0):

$$O_{F(5)}^{\eta_y}\left(g(x,y)\right) = \frac{e^{jy^2/2\eta_y}}{\sqrt{j\eta_y}} *_{\text{along } y\text{-axis}} g(x,y). \tag{61}$$

(6) scaling along the y-axis ($a_{22} = 1/d_{22} = \rho_y$, $a_{11} = d_{11} = 1$, others = 0):

$$O_{F(6)}^{\rho_y}\left(g(x,y)\right) = g(x, y/\rho_y)/\sqrt{\rho_y}. \tag{62}$$

(7) multiplication of $\exp(j\tau xy)$ ($a_{11} = a_{22} = d_{11} = d_{22} = 1$, $c_{12} = c_{21} = \tau$, others = 0):

$$O_{F(7)}^{\tau}\left(g(x,y)\right) = e^{j\tau \cdot xy} \cdot g(x,y). \tag{63}$$

(8) convolution of $\sqrt{-1} \cdot |\eta|^{-1} e^{j \cdot xy/\eta}$ ($a_{11} = a_{22} = d_{11} = d_{22} = 1$, $b_{12} = b_{21} = \eta$, others = 0):

$$O_{F(8)}^{\eta}\left(g(x,y)\right) = \frac{e^{j \cdot xy/\eta}}{\sqrt{-1} \cdot |\eta|} * g(x,y). \tag{64}$$

(9) shearing along the x–axis ($a_{11} = a_{22} = d_{11} = d_{22} = 1$, $-a_{12} = d_{21} = p$, others = 0):

$$O_{F(9)}^{p}\left(g(x,y)\right) = g(x + py, y). \tag{65}$$

(10) shearing along the y–axis ($a_{11} = a_{22} = d_{11} = d_{22} = 1$, $-a_{21} = d_{12} = q$, others = 0):

$$O_{F(10)}^{q}\left(g(x,y)\right) = g(x, qx + y). \tag{66}$$

Each of the above basic operation has only 1 free parameter and hence has the free dimension of 1. They all have the additive property relates to their parameter, that is,

$$O_{F(s)}^{t_1} O_{F(s)}^{t_2} = O_{F(s)}^{t_1 + t_2}. \tag{67}$$

Some of these basic operations are exchangeable. That is,

$$O_{F(p)}^{t_1} O_{F(q)}^{t_2} = O_{F(q)}^{t_2} O_{F(p)}^{t_1} \tag{68}$$

when $p, q \in \{1, 4, 7\}$, or when $p, q \in \{2, 5, 8\}$, or when $p, q \in \{3, 6\}$.

All the 2-D AGFFT can be decomposed as the combination of the above 10 basic operations (This is why the free dimension of the 2-D AGFFT is 10). For example, when $\det(\mathbf{A}) \neq \mathbf{0}$ and $a_{22} \neq 0$, we can decompose the 2–D AGFFT as

$$O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})} = O_{F(1)}^{\tau_1} O_{F(4)}^{\tau_2} O_{F(7)}^{\tau_3} O_{F(2)}^{\eta_1} O_{F(5)}^{\eta_2} O_{F(8)}^{\eta_3} O_{F(3)}^{\rho_1} O_{F(6)}^{\rho_2} O_{F(9)}^{p} O_{F(10)}^{q} \tag{69}$$

where $\begin{bmatrix} \tau_1 & \tau_3 \\ \tau_3 & \tau_2 \end{bmatrix} = \mathbf{C}\mathbf{A}^{-1}$, $\begin{bmatrix} \eta_1 & \eta_3 \\ \eta_3 & \eta_2 \end{bmatrix} = \mathbf{A}\mathbf{B}^{\mathbf{T}}$, $\rho_1 = \det(\mathbf{A})/a_{22}$,

$$\rho_2 = a_{22}, \qquad p = -a_{12}a_{22}/\det(\mathbf{A}), \qquad q = -a_{21}/a_{22}. \tag{70}$$

Except for the 10 basic operations list above, there are also other useful basic operations for the 2-D AGFFT. For example:

(11) 1-D FRFT on the x–axis: $\{a_{11}= d_{11}= \cos\alpha,\ b_{11}= -c_{11}= \sin\alpha,\ a_{22}= d_{22}= 1,$ others $= 0\}$:

$$O^{\alpha}_{F(11)}\big(g(x,y)\big)= \sqrt{\frac{1}{j2\pi}}\cdot e^{\frac{j}{2}\cdot\cot\alpha\cdot f^2}\int_{-\infty}^{\infty} e^{-j\cdot\csc\alpha\cdot f\cdot x}\cdot e^{\frac{j}{2}\cdot\cot\alpha\cdot x^2}\cdot g(x,y)\cdot dx. \tag{71}$$

(12) 1-D FRFT on the y–axis: $\{a_{22}= d_{22}= \cos\alpha,\ b_{22}= -c_{22}= \sin\alpha,\ a_{11}= d_{11}= 1,$ others $= 0\}$:

$$O^{\alpha}_{F(12)}\big(g(x,y)\big)= \sqrt{\frac{1}{j2\pi}}\cdot e^{\frac{j}{2}\cdot\cot\alpha\cdot h^2}\int_{-\infty}^{\infty} e^{-j\cdot\csc\alpha\cdot h\cdot y}\cdot e^{\frac{j}{2}\cdot\cot\alpha\cdot y^2}\cdot g(x,y)\cdot dy. \tag{72}$$

(13) Clockwise rotation of the y-axis with angle θ_1

$\{a_{11} = 1,\ a_{12} = -\sin\theta_1,\ a_{22} = \cos\theta_1,\ d_{11} = 1,\ d_{21} = \tan\theta_1,\ d_{22} = \sec\theta_1,$ others $= 0\}$:

$$G^{\theta_1}_{(13)}(f,h)= O^{\theta_1}_{F(3')}\big(g(x,y)\big)= g\big(f+\tan\theta_1 h,\ \sec\theta_1 h\big), \tag{73}$$

or

$$g(f,h)= G^{\theta_1}_{(13)}\big(f-\sin\theta_1 h,\ \cos\theta_1 h\big). \tag{73a}$$

(14) Counterclockwise rotation of the x-axis with angle θ_2

$\{a_{11} = \cos\theta_2,\ a_{21} = \sin\theta_2,\ a_{22} = 1,\ d_{11} = \sec\theta_2,\ d_{12} = -\tan\theta_2,\ d_{22} = 1,$ others $= 0\}$:

$$G^{\theta_2}_{(14)}(f,h)= O^{\theta_2}_{F(14)}\big(g(x,y)\big)= g\big(\sec\theta_2 f,\ -\tan\theta_2 f+h\big), \tag{74}$$

or

$$g(f,h)= G^{\theta_2}_{(14)}\big(\cos\theta_2 f,\ \sin\theta_2 f+h\big). \tag{74a}$$

In fact, these 4 basic operations can all be decomposed as the combination of the former 10 basic operations, so they will not increase the free dimension of the 2-D AGFFT:

$$O^{\alpha}_{F(11)} = O^{\csc\alpha-\cot\alpha}_{F(2)} O^{-\sin\alpha}_{F(1)} O^{\csc\alpha-\cot\alpha}_{F(2)}, \tag{75}$$

$$O^{\alpha}_{F(12)} = O^{\csc\alpha-\cot\alpha}_{F(5)} O^{-\sin\alpha}_{F(4)} O^{\csc\alpha-\cot\alpha}_{F(5)}, \tag{76}$$

$$O^{\theta_1}_{F(13)} = O^{\cos\alpha}_{F(6)} O^{\sin\alpha}_{F(9)}, \tag{77}$$

$$O^{\theta_2}_{F(14)} = O^{\cos\alpha}_{F(3)} O^{-\sin\alpha}_{F(10)}. \tag{78}$$

From the basic operations, we can see the structure of the 2-D AGFFT, and realize how the 2-D AGFFT generalizes the 2-D separable FRFT and other transforms list in Fig. 1. We list Table 1 to show whether the basic operations described above exist for each of the special cases of the 2-D AGFFT. We use 'O' and '×' to indicate whether the basic operations exist for each of the transforms, and use (A) ~ (G) to indicate:

(A): 2-D separable FRFT, (B): 2-D separable Fresnel transform,

(C): multiplication of $\exp(j(\eta_1 x^2+\eta_2 xy+\eta_3 y^2))$, (D): convolution with $\exp(j(\eta_1 x^2+\eta_2 xy+\eta_3 y^2))$,

(E): 2-D separable canonical transform, (F): Sahin's 2-D unseparable FRFT,

(G): Geometric twisting operation.

The degree of freedom for each transform in Table 1 can be calculated from total number of 'O's in the corresponding column, and in each case as below, the degree of freedom must be decreased by 1: (a) The 1st, 2nd, 11th items are all 'O's. (b) The 4th, 5th, 12th items are all 'O's. (c) The 6th, 9th, 13th items are all 'O's. (d) The 3rd, 10th, 14th items are all 'O's.

The basic operations 1~6 exist for the 2-D separable canonical transform, but the basic operations 7~10 don't exist for this transform. This is because for the former 6 basic operations, the x–axis and y–axis are independent, but for the latter 4 operations, the x–axis and y–axis are dependent. And because for the remained 4 basic operations, the 7th, 8 th basic operations are the multiplication and convolution of $\exp(j\eta xy)$, the 9th, 10th basic operations exist for the geometric twisting operations, so we can say the 2–D AGFFT is the combination of

(1) 2-D separable canonical transform (E) (basic ops. (1) ~ (6)),

(2) multiplication or convolution of $\exp(j\eta xy)$ (C, D) (basic ops. (7), (8)),

(3) geometric twisting operation (G) (basic ops. (3), (6), (9), (10)).

Also, from Ref. [8], we find all the canonical transforms can be decomposed as the combination of chirp multiplication, Fourier transform, and scaling operation (one kind of geometric twisting operation). Thus, we can also view the 2-D AGFFT as the combination of

(1) 2-D Fourier transform (2-D FT),

(2) geometric twisting operation (G) (basic ops. (3), (6), (9), (10)),

(3) multiplication of quadratic phase function (i.e., $\exp(j(\eta_1 x^2+\eta_2 xy+\eta_3 y^2))$) (C)
(basic ops. (1), (4), (7)),

(4) convolution with the quadratic phase function (D) (basic ops. (2), (5), (8)).

Since the 2-D FT can be further decomposed as the combination of the chirp multiplication and chirp convolution, and can be absorbed into the third and the forth components, so we can just say the 2-D AGFFT is the combination of the latter 3 components described above.

2-4 The 2-D Affine Generalized Fractional Convolution and Correlation

Analogous to the conventional 2-D convolution defined as Eq. (35), we can define the **2-D affine generalized fractional convolution (2-D AGFCV)**. If $z(x, y)$ is the 2-D affine generalized fractional convolution of $f(x, y)$ and $g(x, y)$, then

$$\textbf{2-D AGFCV: } z(x,y)=O_F^{(\mathbf{D}^{\mathbf{T}},-\mathbf{B}^{\mathbf{T}},-\mathbf{C}^{\mathbf{T}},\mathbf{A}^{\mathbf{T}})}\left(O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f(x,y))\cdot O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(g(x,y))\right). \quad (79)$$

Also, from the similar way as the 1-D fractional correlation [9][10], we can define the **2-D affine generalized fractional correlation (2-D AGFCR)** as:

$$\textbf{2-D AGFCR: } z(x,y)=O_F^{(\mathbf{E},\mathbf{F},\mathbf{G},\mathbf{H})}\left(O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f(x,y))\cdot O_F^{(\mathbf{A}',\mathbf{B}',\mathbf{C}',\mathbf{D}')}(g^*(x,y))\right). \quad (80)$$

For convenience, we can choose {**E**, **F**, **G**, **H**} as {**0**, **−I**, **I**, **0**}, and the 2-D AGFC becomes:

2-D AGFCR: $$z(x,y)=IFT\left(O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f(x,y))\cdot O_F^{(\mathbf{A}',\mathbf{B}',\mathbf{C}',\mathbf{D}')}(g^*(x,y))\right). \quad (81)$$

The 2-D AGFCV can be used for the filter design, generalized Hilbert transform, and mask, and the 2-D AGFCR can be applied for the 2-D pattern recognition.

III. Properties and the Transform Results of the 2-D AGFFT

We will discuss the properties of the 2-D AGFFT and its kernel. We will find, most of the properties of the 1-D fractional Fourier transform [3][4], and most properties of the 2-D fractional canonical transform [1] and the 2-D unseparable transform with 4 parameters [2] can all be extended for the 2-D AGFFT. We will often use the notation described in Eqs. (24), (25).

3-1 Relations with the 2-D Wigner Distribution Function (WDF)

The **2-D Wigner distribution function (2-D WDF)** is the extension of 1-D Wigner distribution. Its formula is:

$$W_g(x,y,f,h)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-j\tau\cdot f}e^{-j\eta\cdot h}g(x+\frac{\tau}{2},y+\frac{\eta}{2})g^*(x-\frac{\tau}{2},y-\frac{\eta}{2})\cdot d\tau d\eta. \quad (82)$$

The 2-D WDF has closed relations with the 2-D AGFFT. If

$$W_{G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(x,y,f,h)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-j\tau f}e^{-j\eta h}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(x+\frac{\tau}{2},y+\frac{\eta}{2})G^*_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(x-\frac{\tau}{2},y-\frac{\eta}{2})d\tau d\eta, \quad (83)$$

then we can prove

$$W_g(x,y,f,h)=W_{G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(a_{11}x+a_{12}y+b_{11}f+b_{12}h,\quad a_{21}x+a_{22}y+b_{21}f+b_{22}h,\\ c_{11}x+c_{12}y+d_{11}f+d_{12}h,\quad c_{21}x+c_{22}y+d_{21}f+d_{22}h) \quad (84)$$

That is, if $\vec{\mathbf{v}}=[x,y,f,h]$, then

$$W_g(\vec{\mathbf{v}})=W_{G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}\left(\vec{\mathbf{v}}\cdot\begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{C} & \mathbf{D}\end{bmatrix}^{\mathbf{T}}\right). \quad (85)$$

The Eq. (85) can also be written as

$$W_{G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(\vec{\mathbf{v}})=W_g\left(\vec{\mathbf{v}}\cdot\begin{bmatrix}\mathbf{D} & -\mathbf{C}\\ -\mathbf{B} & \mathbf{A}\end{bmatrix}\right)=W_g\left(\vec{\mathbf{v}}\cdot\begin{bmatrix}\mathbf{D}^{\mathbf{T}} & -\mathbf{B}^{\mathbf{T}}\\ -\mathbf{C}^{\mathbf{T}} & \mathbf{A}^{\mathbf{T}}\end{bmatrix}^{\mathbf{T}}\right). \quad (86)$$

Except for Eqs. (84)~(86), there are also some important relations between the 2-D AGFFT and 2-D WDF. We list them in the Table 2. The proofs of these relations are list in the Appendix.

3-2 Relations with the shifting-modulation operation and the differentiation-multiplication operation

Except for the 2-D Wigner distribution function, the 2-D AGFFT also has close relations with the shifting-modulation operation pair and the differentiation-multiplication operation pair. Before discussing these relations, we first list the shifting, modulation, differentiation, and multiplication properties of the 2-D AGFFT in Table 3. The proofs of properties 3, 5 in Table 3 are described in the Appendix.

From the space shifting property, we find after the 2-D AGFFT, the space shifting will partially become the modulation operation (due to **C** or **D**), and partially remain the geometric shifting operation (due to **A** or **B**). Thus, unlike the 2-D Fourier transform, the 2-D AGFFT is not space-invariant. Combine space shifting property and modulation properties, we find

$$O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}\left(e^{i\cdot n_1 x}e^{i\cdot n_2 y}g(x-m_1,y-m_2)\right)=e^{j\zeta}e^{i\cdot s_1 f}e^{i\cdot s_2 h}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f-r_1,h-r_2) \tag{87}$$

where

$$\begin{bmatrix} r_1 \\ r_2 \\ s_1 \\ s_2 \end{bmatrix}=\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}\cdot\begin{bmatrix} m_1 \\ m_2 \\ n_1 \\ n_2 \end{bmatrix}, \tag{88}$$

and ζ is some constant phase. Thus, the 2-D AGFFT has closed relation with the shifting-modulation operations pair.

From the above discussion, we find the 2-D AGFFT defined as Eqs. (18)~(21) can be further generalized to include the space shift term and modulation term. That is,

$$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{M},\mathbf{N})}(f,h)=e^{j\cdot(n_f f+n_h h)}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f-m_f,h-m_h) \tag{89}$$

where $\mathbf{M}=[m_f,\ m_h]^{\mathrm{T}}$ represents the space shifting, and $\mathbf{N}=[n_f,\ n_h]^{\mathrm{T}}$ represents the frequency modulation. We will call it as the 2-D AGFFT with space-shifting and frequency modulation (**2-D TFAGFFT**). Together with the 10 free dimensions of the original 2-D AGFFT, there are totally 14 free dimensions for the 2-D TFAGFFT. The 2-D TFAGFFT can be represented as:

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix}\cdot\begin{bmatrix} \vec{\mathbf{x}}^{\mathrm{T}} \\ \vec{\mathbf{w}}^{\mathrm{T}} \end{bmatrix}+\begin{bmatrix} \mathbf{M} \\ \mathbf{N} \end{bmatrix}. \tag{90}$$

We can prove the following additivity property will be satisfied for the 2-D TFAGFFT:

$$O_F^{(\mathbf{A}',\mathbf{B}',\mathbf{C}',\mathbf{D}',\mathbf{M}',\mathbf{N}')}O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{M},\mathbf{N})}=O_F^{(\hat{\mathbf{A}},\hat{\mathbf{B}},\hat{\mathbf{C}},\hat{\mathbf{D}},\hat{\mathbf{M}},\hat{\mathbf{N}})} \tag{91}$$

where

$$\begin{bmatrix} \hat{\mathbf{A}} & \hat{\mathbf{B}} \\ \hat{\mathbf{C}} & \hat{\mathbf{D}} \end{bmatrix}=\begin{bmatrix} \mathbf{A}' & \mathbf{B}' \\ \mathbf{C}' & \mathbf{D}' \end{bmatrix}\cdot\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix},\qquad \begin{bmatrix} \hat{\mathbf{M}} \\ \hat{\mathbf{N}} \end{bmatrix}=\begin{bmatrix} \mathbf{A}' & \mathbf{B}' \\ \mathbf{C}' & \mathbf{D}' \end{bmatrix}\cdot\begin{bmatrix} \mathbf{M} \\ \mathbf{N} \end{bmatrix}+\begin{bmatrix} \mathbf{M}' \\ \mathbf{N}' \end{bmatrix}. \tag{92}$$

The additivity relation in Eqs. (91), (92) can also be expressed as:

$$\begin{bmatrix} \hat{\mathbf{A}} & \hat{\mathbf{B}} \\ \hat{\mathbf{C}} & \hat{\mathbf{D}} \end{bmatrix} \cdot \begin{bmatrix} \vec{\mathbf{x}}^{\mathbf{T}} \\ \vec{\mathbf{w}}^{\mathbf{T}} \end{bmatrix} + \begin{bmatrix} \hat{\mathbf{M}} \\ \hat{\mathbf{N}} \end{bmatrix} = \begin{bmatrix} \mathbf{A}' & \mathbf{B}' \\ \mathbf{C}' & \mathbf{D}' \end{bmatrix} \cdot \left(\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \cdot \begin{bmatrix} \vec{\mathbf{x}}^{\mathbf{T}} \\ \vec{\mathbf{w}}^{\mathbf{T}} \end{bmatrix} + \begin{bmatrix} \mathbf{M} \\ \mathbf{N} \end{bmatrix} \right) + \begin{bmatrix} \mathbf{M}' \\ \mathbf{N}' \end{bmatrix}. \tag{93}$$

The reversible property of the 2-D TFAGFFT is as below:

$$\left(O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D},\mathbf{M},\mathbf{N})}\right)^{-1} = O_F^{\left(\mathbf{D}^{\mathbf{T}},-\mathbf{B}^{\mathbf{T}},-\mathbf{C}^{\mathbf{T}},\mathbf{A}^{\mathbf{T}},-\mathbf{D}^{\mathbf{T}}\mathbf{M}+\mathbf{B}^{\mathbf{T}}\mathbf{N},\mathbf{C}^{\mathbf{T}}\mathbf{M}-\mathbf{A}^{\mathbf{T}}\mathbf{N}\right)}. \tag{94}$$

The 2-D TFAGFFT will be very useful for the filter design.

Also, from the multiplication differentiation properties in the Table 3, we find

$$\begin{aligned} &O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}\left[h_1 \frac{\partial g(x,y)}{\partial x} + h_2 \frac{\partial g(x,y)}{\partial y} - h_3 \cdot jx \cdot g(x,y) - h_4 \cdot jy \cdot g(x,y)\right] \\ &= k_1 \frac{\partial G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)}{\partial f} + k_2 \frac{\partial G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)}{\partial h} - k_3 jf \cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h) - k_4 jh \cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h), \end{aligned} \tag{95}$$

where the coefficients k_1, k_2, k_3, k_4 can be calculated from

$$\begin{bmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \cdot \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \end{bmatrix}. \tag{96}$$

Thus, the differentiation-multiplication operations pair also has closed relations with the 2-D AGFFT. The 2-D Wigner transform, shifting-modulation operations pair, and the differentiation-multiplication operation pair are the 3 operations closed to the 2-D AGFFT.

3-3 Other properties of the 2-D AGFFT

Except for the relations with the 2-D WDF, shifting-modulation operation, and differentiation-multiplication operation, there are also some important properties for the 2-D AGFFT. We use Table 4, Table 5 to summary these properties. We prove the property 3 in Table 4 and property 4 in Table 5 in the Appendix.

3-4 Transform Results for Some Special Signals

(1) $\delta(x, y)$:

$$O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(\delta(x,y)) = \frac{e^{\frac{j}{2\det(\mathbf{B})}\left((d_{11}b_{22}-d_{12}b_{21})\cdot f^2+2(-d_{11}b_{12}+d_{12}b_{11})\cdot f\cdot h+(-d_{21}b_{12}+d_{22}b_{11})\cdot h^2\right)}}{2\pi\sqrt{-\det(\mathbf{B})}}. \tag{97}$$

(2) Generalized transform results for $\exp[j(\eta_1 x^2+\eta_2 xy+\eta_3 y^2)+j(\tau_1 x+\tau_2 y)]$:

We note $\exp[j(\eta_1 x^2+\eta_2 xy+\eta_3 y^2)]$ can be obtained from unity equation (i.e., $g(x, y) = 1$) by the 2-D AGFFT with parameter $\{\mathbf{I}, \mathbf{0}, \Phi, \mathbf{I}\}$:

$$e^{j\left(\eta_1 x^2+\eta_2 xy+\eta_3 y^2\right)} = O_F^{(\mathbf{I},\mathbf{0},\Phi,\mathbf{I})}(1), \qquad \text{where} \quad \Phi=\begin{bmatrix} 2\eta_1 & \eta_2 \\ \eta_2 & 2\eta_3 \end{bmatrix}. \tag{98}$$

The function $\exp[j(\eta_1 x^2+\eta_2 xy+\eta_3 y^2)+j(\tau_1 x+\tau_2 y)]$ is the modulation of $\exp[j(\eta_1 x^2+\eta_2 xy+\eta_3 y^2)]$ by $\exp[j(\tau_1 x+\tau_2 y)]$. Then because the 2-D Fourier transform of $(-1)^{1/2}2\pi\cdot\delta(x, y)$ is 1, so

$$e^{j\left(\eta_1 x^2+\eta_2 xy+\eta_3 y^2\right)} = O_F^{(\mathbf{0},\mathbf{I},-\mathbf{I},\Phi)}\left(\sqrt{-1}\cdot 2\pi\delta(x, y)\right). \tag{99}$$

From Eq. (99) and the modulation property list in Table 3, we obtain

$$O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}\left(e^{j\left(\eta_1 x^2+\eta_2 xy+\eta_3 y^2+\tau_1 x+\tau_2 y\right)}\right) = \exp\left(j(\phi+\vec{\mathbf{w}}\mathbf{D}\vec{\mathbf{w}}_0^{\mathrm{T}})\right) R_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}\left(\vec{\mathbf{w}}-\vec{\mathbf{w}}_0\mathbf{B}^{\mathrm{T}}\right), \tag{100}$$

where $\phi = -\vec{\mathbf{w}}_0\mathbf{B}^{\mathrm{T}}\mathbf{D}\vec{\mathbf{w}}_0^{\mathrm{T}}/2$, $\vec{\mathbf{w}}_0 = (\tau_1, \tau_2)$, and

$$R_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f, h) = O_F^{(-\mathbf{B},\mathbf{A}+\mathbf{B}\Phi,-\mathbf{D},\mathbf{C}+\mathbf{D}\Phi)}\left(\sqrt{-1}\cdot 2\pi\delta(x, y)\right). \tag{101}$$

Then apply Eq. (97), we obtain

$$O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}\left(e^{j\left(\eta_1 x^2+\eta_2 xy+\eta_3 y^2+\tau_1 x+\tau_2 y\right)}\right)$$
$$= \frac{1}{\sqrt{\det(\mathbf{A}+\mathbf{B}\Phi)}}\exp\left(-j\frac{\vec{\mathbf{w}}_0\mathbf{B}^{\mathrm{T}}\mathbf{D}\vec{\mathbf{w}}_0^{\mathrm{T}}}{2}+j\vec{\mathbf{w}}\mathbf{D}\vec{\mathbf{w}}_0^{\mathrm{T}}+\frac{j}{2}\left(\vec{\mathbf{w}}-\vec{\mathbf{w}}_0\mathbf{B}^{\mathrm{T}}\right)(\mathbf{C}+\mathbf{D}\Phi)(\mathbf{A}+\mathbf{B}\Phi)^{-1}\left(\vec{\mathbf{w}}^{\mathrm{T}}-\mathbf{B}\vec{\mathbf{w}}_0^{\mathrm{T}}\right)\right), \tag{102}$$

where $\vec{\mathbf{w}}_0 = (\tau_1, \tau_2)$, $\Phi=\begin{bmatrix} 2\eta_1 & \eta_2 \\ \eta_2 & 2\eta_3 \end{bmatrix}$. (103)

IV. Calculation and Digital Implementation

Although the 2-D AGFFT form seems to be very complex, but in fact we can implement it in very simple way. From subsection 2-3, we have discussed all the 2-D AGFFT can be decomposed as the combinations of the 10 basic operations. And among these basic operations, we find quadratic phase term convolution (i.e., the 2nd, 5th, 8th basic operations) can be done by the combination of Fourier transform and quadratic phase term multiplication. Thus, all the 2-D AGFFT can be decomposed as the combination of the quadratic phase term multiplication (i.e., the 1st, 4th, 7th basic operations), geometric twisting operations (i.e., the 3rd, 6th, 9th, 10th basic operations), and the Fourier transform. This will be a great help for calculating the 2-D AGFFT because these operations are all easier to calculate than the integral operation in the Eq. (18).

We will discuss how to decompose the 2-D AGFFT for each case in subsection 4-1.

4-1 The calculation and digital implementation for the 2-D AGFFT

(1) $\det(\mathbf{B}) \neq 0$:

If $\det(\mathbf{B}) \neq 0$, then the 2-D AGFFT can be decomposed as:

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{D}\mathbf{B}^{-1} & \mathbf{I} \end{bmatrix} \begin{bmatrix} 0 & \mathbf{I} \\ -\mathbf{I} & 0 \end{bmatrix} \begin{bmatrix} \mathbf{I} & 0 \\ \mathbf{A}\mathbf{B}^{\mathrm{T}} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{B}^{\mathrm{T}-1} & 0 \\ 0 & \mathbf{B} \end{bmatrix}. \tag{104}$$

That is, in the case that $\det(\mathbf{B}) \neq 0$, the 2-D AGFFT can be rewritten as:

$$O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(g(x,y)) = \sqrt{\det(\mathbf{B})} \cdot e^{j(q_1 \cdot f^2 + q_2 \cdot f \cdot h + q_3 \cdot h^2)} \cdot FT\left(e^{j(v_1 \cdot x^2 + v_2 \cdot x \cdot y + v_3 \cdot y^2)} \cdot g(b_{11}x + b_{21}y,\ b_{12}x + b_{22}y)\right), \tag{105}$$

where $$\begin{bmatrix} 2q_1 & q_2 \\ q_2 & 2q_3 \end{bmatrix} = \mathbf{D}\mathbf{B}^{-1}, \qquad \begin{bmatrix} 2v_1 & v_2 \\ v_2 & 2v_3 \end{bmatrix} = \mathbf{A}\mathbf{B}^{\mathrm{T}}. \tag{106}$$

So the 2-D AGFFT can be calculated by the following 4 steps:

(1) scaling, (2) multiplication of a quadratic phase function,

(3) 2-D Fourier transform, (4) multiplication of a quadratic phase function.

For the digital implementation of 2-D AGFFT, we can also apply the Eq. (105). The sampling will not be done directly to the input function. Instead, it will be done after the quadratic phase multiplication and geometric twisting (i.e., before the 2-D FFT), as following:

$$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(r\Delta_f, s\Delta_h) = e^{j(q_1 r^2 \Delta_f^2 + q_2 r s \Delta_f \Delta_h + q_3 s^2 \Delta_h^2)} DFT\left(e^{j(v_1 m^2 \Delta_x^2 + v_2 m n \Delta_x \Delta_y + v_3 m^2 \Delta_y^2)} \cdot \sqrt{\det(\mathbf{B})} \cdot g(b_{11}m\Delta_x + b_{21}n\Delta_y,\ b_{12}m\Delta_x + b_{22}n\Delta_y)\right) \tag{107}$$

where q_1, q_2, q_3, v_1, v_2, v_3 are also the same as Eq. (106), and the sampling interval Δ_x, Δ_y, Δ_f, Δ_h must satisfy:

$$\Delta_x \cdot \Delta_f = 2\pi/M, \qquad \Delta_y \cdot \Delta_h = 2\pi/N \tag{108}$$

where M, N are the number of sampling points in the x-axis and y-axis for $g(b_{11}x+b_{21}y,\ b_{12}x+b_{22}y)$. If in Eq. (107), we set DC point at the middle, then Eq. (107) can be rewritten as:

$$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(r\Delta_f, s\Delta_h) = \frac{\Delta_x \Delta_y \sqrt{-\det(\mathbf{B})}}{2\pi} \cdot e^{j(q_1 \cdot r^2 \Delta_f^2 + q_2 \cdot r \cdot s \cdot \Delta_f \Delta_h + q_3 \cdot s^2 \cdot \Delta_h^2)/\det(\mathbf{B})} \cdot \sum_{m=(-M+1)/2}^{(M-1)/2} \sum_{n=(-N+1)/2}^{(N-1)/2} e^{j(v_1 \cdot m^2 \Delta_x^2 + v_2 \cdot m \cdot n \cdot \Delta_x \Delta_y + v_3 \cdot m^2 \cdot \Delta_y^2)} \cdot e^{-j2\pi\left(\frac{r \cdot m}{M} + \frac{s \cdot n}{N}\right)} \cdot g(b_{11}m\Delta_x + b_{21}n\Delta_y,\ b_{12}m\Delta_x + b_{22}n\Delta_y)\Big) \tag{109}$$

And the digital implementation of inverse 2-D AGFFT is

$$g(b_{11}m\Delta_x + b_{21}n\Delta_y, b_{12}m\Delta_x + b_{22}n\Delta_y))$$

$$= \frac{\Delta_f \Delta_h}{2\pi\sqrt{-\det(\mathbf{B})}} e^{-j(v1\cdot m^2\Delta_x{}^2 + v2\cdot m\cdot n\cdot\Delta_x\Delta_y + v3\cdot m^2\cdot\Delta_y{}^2)} \cdot \sum_{r=(-M+1)/2}^{(M-1)/2} \sum_{s=(-N+1)/2}^{(N-1)/2} e^{j2\pi(\frac{r\cdot m}{M}+\frac{s\cdot n}{N})} \cdot \quad (110)$$

$$e^{-j(q_1\cdot r^2\Delta_f{}^2 + q_2\cdot r\cdot s\cdot\Delta_f\Delta_h + q_3\cdot s^2\cdot\Delta_h{}^2)/\det(\mathbf{B})} \cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(r\Delta_f, s\Delta_h)$$

We note, in Eq. (109), the sampling points array will no longer align with the x-axis and y-axis for the original function $g(x, y)$. The comparing between the samplings points of $g(b_{11}m\Delta_x+b_{21}n\Delta_y, b_{12}m\Delta_x+b_{22}n\Delta_y)$ and the Cartesian grid sampling points is shown in Fig. 2.

Since usually, the available input datum are Cartesian grid sampling points:

$$g(p\Delta_{x,1}, q\Delta_{y,1}) \qquad p = ..., -1, 0, 1, 2, ..., \quad q = ..., -1, 0, 1, 2, ... \quad (111)$$

So in Eq. (109), if we want to obtain the value of $g(b_{11}m\Delta_x+b_{21}n\Delta_y, b_{12}m\Delta_x+b_{22}n\Delta_y)$, we can calculate it by the bilinear interpolation:

$$\begin{aligned} &g(b_{11}m\Delta_x + b_{21}n\Delta_y, b_{12}m\Delta_x + b_{22}n\Delta_y) \\ &= (1-h)(1-s)\, g(p\Delta_{x,1}, q\Delta_{y,1}) + h(1-s)\, g((p+1)\Delta_{x,1}, q\Delta_{y,1}) \\ &+ (1-h)s \cdot g(p\Delta_{x,1}, (q+1)\Delta_{y,1}) + hs \cdot g((p+1)\Delta_{x,1}, (q+1)\Delta_{y,1}) \end{aligned} \quad (112)$$

where $h = (b_{11}m\Delta_x+b_{21}n\Delta_y-p\Delta_x)/\Delta_x$, $s = (b_{12}m\Delta_x+b_{22}n\Delta_y-q\Delta_y)/\Delta_y$. We use Fig. 3 to illustrate the concept of bilinear interpolation. Eq. (112) can also be rewritten as:

$$\begin{aligned} &g(b_{11}m\Delta_x + b_{21}n\Delta_y, b_{12}m\Delta_x + b_{22}n\Delta_y). \\ &= (1-h)s\left[g(p\Delta_{x,1}, (q+1)\Delta_{y,1}) - g(p\Delta_{x,1}, q\Delta_{y,1})\right] + hs\left[g((p+1)\Delta_{x,1}, (q+1)\Delta_{y,1}) - g(p\Delta_{x,1}, q\Delta_{y,1})\right] \\ &+ g(p\Delta_{x,1}, q\Delta_{y,1}) + h(1-s)\left[g((p+1)\Delta_{x,1}, q\Delta_{y,1}) - g(p\Delta_{x,1}, q\Delta_{y,1})\right]. \end{aligned} \quad (113)$$

So each interpolation requires 3 multiplications. And if there are M different values of m and N different values of n, then the total number of multiplications required for the resampling is:

$$3MN. \quad (114)$$

Sometimes, we may use some more accurate resample algorithm instead of Eq. (112). For example, we can use more points for the interpolation:

$$\begin{aligned} &g(b_{11}m\Delta_x + b_{21}n\Delta_y, b_{12}m\Delta_x + b_{22}n\Delta_y) \\ &= O_{resampling}\left(\{g((p+h)\Delta_{x,1}, (q+k)\Delta_{y,1}) \mid h = -H+1 \sim H, \quad k = -K+1 \sim K\}\right) \end{aligned} \quad (115)$$

where $p\Delta_{x,1} \le b_{11}m\Delta_x+b_{21}n\Delta_y < (p+1)\Delta_{x,1}$, $q\Delta_{y,1} \le b_{12}m\Delta_x+b_{22}n\Delta_y < (q+1)\Delta_{y,1}$. That is, we first choose a window containing $2H\times2K$ original sampling points, then use the values of $g(x, y)$ on these points together with some resampling algorithm (denoted by $O_{resampling}$) to estimate the value of $g(x, y)$ on twisted grid sampling points. In these cases, the number of multiplications required for each resampling point is the function of the size of the window, and we can denote it by $f(4HK)$. So the total multiplications required for resampling is:

$$S = MN \cdot f(4HK). \quad (116)$$

Since it is unnecessary to use all the values of $g(m\Delta_x, n\Delta_y)$ to calculate one resampling, so H, K, and hence $f(4HK)$ is independent of the number of resampling points MN.

The complexity for the resampling usually can be ignored, except for the case that the resampling algorithm is too complicate and $f(4HK)$ is too large. When $f(4HK)$ is too large, the complexity of resampling must be considered when discussing the complexity of 2-D AGFFT. But since $f(4HK)$ is always independent of MN, and $S = MN \cdot f(4HK)$, so even when $f(4HK)$ is large, the amount of multiplications required for resampling is still proportional to MN, and the complexity of 2-D AGFFT is still dominated by the 2-D separable FRFT.

From Eqs. (109), the number of the multiplications for the digital implementation of 2-D AGFFT when $\det(\mathbf{B}) \neq 0$ is approximated to:

$$2MN + (MN/2)\log_2(MN) \text{ (required for 2-D separable FRFT)} + S \text{ (required for resampling)}$$
$$= (2+f(4HK))MN + (MN/2)\log_2(MN). \quad (117)$$

So the complexity of the digital implementation of 2-D AGFFT by the method of Eq. (109) is dependent on the resample algorithm we use. When we use the interpolation algorithm as Eq. (112) for the resampling, then $f(4HK) = 3$, $S = 3MN$, and the total number of the multiplication operations required for digital implementation of 2-D AGFFT is:

$$5MN + (MN/2)\log_2(MN) \approx (MN/2)\log_2(MN) \quad \text{when } M, N \text{ are large.} \quad (118)$$

We note, this is similar to the complexity of the digital implementation of 2-D separable fractional Fourier transform [1]. Thus, although the 2-D AGFFT has more parameters, but it is in fact as simple as the 2-D separable fractional Fourier transform, even almost as simple as the 2-D separable Fourier transform.

(2) $\mathbf{B} = \mathbf{0}$:

In this case, we just use Eq. (30). This is much simple, and for the digital implementation, the complexity is proportion to $M \times N$.

(3) $\det(\mathbf{B}) = \mathbf{0}$, $\mathbf{B} \neq \mathbf{0}$, and $\det(\mathbf{A}) \neq \mathbf{0}$:

In this case, we use the Eq. (45) to implement the 2-D AGFFT as,

$$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h) = \frac{1}{\sqrt{\det(\mathbf{A})}} e^{j\left(p_1 \cdot f^2 + p_2 \cdot f \cdot h + p_3 \cdot h^2\right)} \cdot IFT_{\substack{w \to f \\ v \to h}}\left[e^{j(q_1 \cdot w^2 + q_2 \cdot wv + q_3 \cdot v^2}\right.$$
$$\left. FT_{\substack{x \to w \\ y \to v}}\left(g\left(\frac{a_{22}x - a_{12}y}{\det(\mathbf{A})}, \frac{-a_{21}x + a_{11}y}{\det(\mathbf{A})}\right)\right)\right], \quad (119)$$

where $\begin{bmatrix} 2p_1 & -p_2 \\ -p_2 & 2p_3 \end{bmatrix} = \mathbf{C}\mathbf{A}^{-1}$, $\begin{bmatrix} 2q_1 & -q_2 \\ -q_2 & 2q_3 \end{bmatrix} = -\mathbf{A}\mathbf{B}^{\mathbf{T}}$. (120)

For the digital implementation, Eq. (119) becomes

$$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(r\Delta_f, s\Delta_h)$$

$$= \frac{e^{j\left(p_1 \cdot r^2 \Delta_f{}^2 + p_2 \cdot r \cdot s \cdot \Delta_f \Delta_h + p_3 \cdot s^2 \cdot \Delta_h{}^2\right)}}{MN\sqrt{\det(\mathbf{A})}} \sum_{t=\frac{-M+1}{2}}^{\frac{M-1}{2}} \sum_{k=\frac{-N+1}{2}}^{\frac{N-1}{2}} e^{j2\pi\left(\frac{r \cdot t}{M} + \frac{s \cdot k}{N}\right)} e^{j\left(q_1 \cdot t^2 \Delta_w{}^2 + q_2 \cdot t \cdot k \cdot \Delta_w \Delta_v + q_3 \cdot k^2 \cdot \Delta_v{}^2\right)} \tag{121}$$

$$\sum_{m=(-M+1)/2}^{(M-1)/2} \sum_{n=(-N+1)/2}^{(N-1)/2} e^{-j2\pi\left(\frac{t \cdot m}{M} + \frac{k \cdot n}{N}\right)} g\left(\frac{a_{22} m\Delta_x - a_{12} n\Delta_y}{\det(\mathbf{A})}, \frac{-a_{21} m\Delta_x + a_{11} n\Delta_y}{\det(\mathbf{A})}\right)$$

where

$$\Delta_x \Delta_w = 2\pi/M, \qquad \Delta_y \Delta_v = 2\pi/N, \qquad \Delta_x = \Delta_f, \qquad \Delta_y = \Delta_h. \tag{122}$$

The complexity of Eq. (121) is

$$2MN + MN \cdot \log_2(MN) + S \tag{123}$$

where S is the number of the multiplication operations required for the re-sampling. When M, N are large and S is proportion to MN, then Eq. (123) is approximated to $MN\log_2 MN$. It will be twice than the complexity of the case that $\det(\mathbf{B}) \neq \mathbf{0}$ due to 2 DFTs required.

(4) $\det(\mathbf{B}) = \mathbf{0}$, $\mathbf{B} \neq \mathbf{0}$, and $\det(\mathbf{A}) = \mathbf{0}$, $\det(\mathbf{D}) \neq \mathbf{0}$:

In this case, we use Eq. (49), and

$$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h) = \sqrt{\det(\mathbf{D})} \cdot IFT_{\substack{w \to f \\ v \to h}}\left[e^{j(p_1 \cdot w^2 + p_2 \cdot wv + p_3 \cdot v^2)} \cdot FT_{\substack{x \to w \\ y \to v}}\left[e^{j\left(q_1 \cdot x^2 + q_2 \cdot x \cdot y + q_3 \cdot y^2\right)} g(d_{11}x + d_{21}y, d_{12}x + d_{22}y)\right]\right], \tag{124}$$

where

$$\begin{bmatrix} 2p_1 & -p_2 \\ -p_2 & 2p_3 \end{bmatrix} = -\mathbf{B}\mathbf{D}^{-1}, \qquad \begin{bmatrix} 2q_1 & -q_2 \\ -q_2 & 2q_3 \end{bmatrix} = \mathbf{C}\mathbf{D}^{\mathrm{T}}. \tag{125}$$

For the digital implementation,

$$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(r\Delta_f, s\Delta_h)$$

$$= \frac{\sqrt{\det(\mathbf{D})}}{MN} \sum_{t=\frac{-M+1}{2}}^{\frac{M-1}{2}} \sum_{k=\frac{-N+1}{2}}^{\frac{N-1}{2}} e^{j2\pi\left(\frac{r \cdot t}{M} + \frac{s \cdot k}{N}\right)} e^{j\left(p_1 \cdot t^2 \Delta_w{}^2 + p_2 \cdot t \cdot k \cdot \Delta_w \Delta_v + p_3 \cdot k^2 \cdot \Delta_v{}^2\right)} \sum_{m=\frac{-M+1}{2}}^{\frac{M-1}{2}} \sum_{n=\frac{-N+1}{2}}^{\frac{N-1}{2}} e^{-j2\pi\left(\frac{t \cdot m}{M} + \frac{k \cdot n}{N}\right)} \cdot \tag{126}$$

$$e^{j\left(q_1 \cdot m^2 \Delta_x{}^2 + q_2 \cdot m \cdot n \cdot \Delta_x \Delta_y + q_3 \cdot n^2 \cdot \Delta_y{}^2\right)} g(d_{11}m\Delta x + d_{21}n\Delta y, d_{12}m\Delta x + d_{22}n\Delta y)$$

where

$$\Delta_x \Delta_w = 2\pi/M, \qquad \Delta_y \Delta_v = 2\pi/N, \qquad \Delta_x = \Delta_f, \qquad \Delta_y = \Delta_h. \tag{127}$$

must also be satisfied. The complexity of Eq. (126) is also as the Eq. (123), i.e., approximated to twice of the complexity in the case that $\det(\mathbf{B}) \neq \mathbf{0}$.

(5) $\det(\mathbf{B}) = 0$, $\mathbf{B} \neq 0$, $\det(\mathbf{A}) = 0$, $\mathbf{A} \neq 0$, $\det(\mathbf{D}) = 0$, $\mathbf{D} \neq 0$, $\det(\mathbf{C}) \neq 0$:

In this case, we can use Eq. (51), and together with Eq. (121), we obtain the formula of digital implementation as:

$$
\begin{aligned}
& G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(r\Delta_f, s\Delta_h) \\
&= \frac{\Delta_u \Delta_z}{2\pi\sqrt{-MN\det(\mathbf{C})}} \sum_{\tau=(-M+1)/2}^{(M-1)/2} \sum_{\eta=(-N+1)/2}^{(N-1)/2} e^{j2\pi\left(\frac{r\cdot\tau}{M}+\frac{s\cdot\eta}{N}\right)} e^{j\left(p_1\cdot\tau^2\Delta_u{}^2+p_2\cdot\tau\cdot\eta\cdot\Delta_u\Delta_z+p_3\cdot\eta^2\cdot\Delta_z{}^2\right)} \\
& \sum_{t=\frac{-M+1}{2}}^{\frac{M-1}{2}} \sum_{k=\frac{-N+1}{2}}^{\frac{N-1}{2}} e^{j2\pi\left(\frac{\tau\cdot t}{M}+\frac{\eta\cdot k}{N}\right)} e^{j\left(q_1\cdot t^2\Delta_w{}^2+q_2\cdot t\cdot k\cdot\Delta_w\Delta_v+q_3\cdot k^2\cdot\Delta_v{}^2\right)} \sum_{m=\frac{-M+1}{2}}^{\frac{M-1}{2}} \sum_{n=\frac{-N+1}{2}}^{\frac{N-1}{2}} e^{-j2\pi\left(\frac{t\cdot m}{M}+\frac{k\cdot n}{N}\right)} \\
& g\left(\frac{c_{22}m\Delta_x - c_{12}n\Delta_y}{\det(\mathbf{C})}, \frac{-c_{21}m\Delta_x + c_{11}n\Delta_y}{\det(\mathbf{C})}\right)
\end{aligned}
\tag{128}
$$

where

$$
\begin{bmatrix} 2p_1 & -p_2 \\ -p_2 & 2p_3 \end{bmatrix} = -\mathbf{A}\mathbf{C}^{-1}, \qquad \begin{bmatrix} 2q_1 & -q_2 \\ -q_2 & 2q_3 \end{bmatrix} = -\mathbf{C}\mathbf{D}^{\mathrm{T}},
$$

$$
\Delta_x\Delta_w = 2\pi/M, \quad \Delta_y\Delta_v = 2\pi/N, \quad \Delta_x = \Delta_u, \quad \Delta_y = \Delta_z, \quad \Delta_w = \Delta_f, \quad \Delta_v = \Delta_h . \tag{129}
$$

The complexity of Eq. (128) is

$$
2MN + (3MN/2)\cdot\log_2(MN) + S \tag{130}
$$

where S is the number of the multiplication operations required for the re-sampling. When M, N are large and S is proportion to MN, then Eq. (130) is approximated to $(3MN/2)\log_2 MN$. It is 3 times of the complexity in the case that $\det(\mathbf{B}) \neq 0$ due to 3 DFTs required.

(6) $\det(\mathbf{B}) = 0$, $\mathbf{B} \neq 0$, $\det(\mathbf{A}) = \det(\mathbf{D}) = \det(\mathbf{C}) = 0$:

We can use Eqs. (54)~(56) to implement the 2-D AGFFT for this case.

4-2 Simplified form of the 2-D AGFFT

The 2-D AGFFT defined as Eqs. (18)~(21) totally has 10 free parameters, 10 exponential terms, and 2 integration operations. It is very complex, and for the practical applications, we will use many times for design because there are 10 parameters to be adjusted. So it would be better to use the simplified form of 2-D AGFFT without losing its utilities.

From subsection 2-3, we have discussed the 2-D AGFFT is the combination of the 2-D Fourier transform, geometric twisting operation, and the multiplication and convolution of the quadratic phase term (i.e., $\exp(j(\eta_1 x^2+\eta_2 xy+\eta_3 y^2))$). We conclude the convolution of the quadratic phase term will not increase the ability of the 2-D AGFFT, since it can be replaced

by the combination of the multiplication f the quadratic phase term and the 2-D Fourier transform. So we just want the simplified 2-D AGFFT contains the 2-D Fourier transform, geometric twisting operation, and the multiplication of the quadratic phase term.

From the above discussion, we conclude the 2-D AGFFT can be simplified by extracting the outside quadratic exponent term by setting $\mathbf{D} = \mathbf{0}$ in the Eqs. (18), (20):

$$O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{0})}(g(x,y)) = \frac{1}{2\pi\sqrt{-\det(\mathbf{B})}} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{\frac{j}{\det(\mathbf{B})}((-b_{22}f+b_{12}h)x+(b_{21}f-b_{11}h)y)} e^{\frac{j}{2\det(\mathbf{B})}\left(p_1\cdot x^2+p_2\cdot x\cdot y+p_3\cdot y^2\right)} g(x,y)\cdot dxdy \quad (131)$$

where p_1, p_2, p_3 are defined as Eq. (21). We find, in Eq. (131) the multiplication of quadratic phase term has been preserved in the term of $\exp(j(p_1x^2+p_2xy+p_3y^2)/2\det(\mathbf{B}))$, and the 2-D Fourier transform together with the geometric twisting operation have been preserved in the term of $\exp(j((-b_{22}f+b_{12}h)x + (b_{21}f-b_{11}h)y)/\det(\mathbf{B}))$, and all the 3 parts of AGFFT still exist. This simplified 2-D AGFFT has only 7 free dimension because of 12 parameters ($\mathbf{D} = \mathbf{0}$) and 5 constraints (the 2nd constraint is naturally satisfied). The 7 free dimensions just correspond to the 7 exponent terms in the Eq. (131). The inverse of Eq. (131) is:

$$O_{IF}^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{0})}\left(G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{0})}(f,h)\right) = \frac{1}{2\pi\sqrt{-\det(\mathbf{B})}} e^{\frac{-j}{2\det(\mathbf{B})}\left(p_1\cdot x^2+p_2\cdot x\cdot y+p_3\cdot y^2\right)} \cdot \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{\frac{j}{\det(\mathbf{B})}((b_{22}x-b_{21}y)f+(-b_{12}x+b_{11}y)h)} G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{0})}(f,h)\cdot dfdh \,. \quad (132)$$

Although this simplified form of 2-D AGFFT only has the free dimension of 7, but in fact it is sufficient for most of the applications of the 2-D AGFFT defined as Eqs. (18)~(21), except for the optical system analysis. We note it has only 1 more free-dimension than the 2-D separable canonical transform defined as Eq. (8), but it can do many things that can't be done by the 2-D separable canonical transform.

We note, for the simplified 2-D AGFFT defined as Eq. (131), $\det(\mathbf{B}) \neq \mathbf{0}$ must be satisfied (due to $\mathbf{D} = \mathbf{0}$). Then from subsection 4-1, the complexity of the digital implementation is approximated to

$$MN + (MN/2)\cdot\log_2 MN + S. \quad (133)$$

Because $\mathbf{D} = \mathbf{0}$, so one chirp multiplication operation is saved.

For the application of filter design (see subsection 5-1), it would be better to further simplify the Eq. (131) by setting $\mathbf{B} = \mathbf{I}$, $\mathbf{C} = -\mathbf{I}$ (in this case, $a_{12} = a_{21}$), and remove the term of $\sqrt{-1}$. Then Eq. (131) becomes:

$$G_{(\mathbf{A})}(f,h)=\frac{1}{2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-j(fx+hy)}e^{j(a_{11}x^2+2a_{12}xy+a_{22}y^2)/2}\cdot g(x,y)\cdot dxdy. \tag{134}$$

In this case there are only 3 parameters, and all the 6 constraints are naturally satisfied. So the free dimension is only 3. This simplified transform is sufficient to filter out all the quadratic type noise easily. And because there are only 3 parameters, so the design of the optimal filter is easy. Its inverse is

$$g(x,y)=\frac{1}{2\pi}e^{-j(a_{11}x^2+2a_{12}xy+a_{22}y^2)/2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{j(fx+hy)}\cdot G_{(\mathbf{A})}(f,h)\cdot dfdh. \tag{135}$$

There are also another choices for the simplified 2-D AGFFT. For example, we can use the following equation:

$$O_{\text{modified }F}^{(a_1,b_1,a_2,b_2,p,q)}(g(x,y))$$
$$=\frac{1}{2\pi\sqrt{b_1b_2(pq-1)}}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-j\left(\frac{f}{b_1}x+\frac{h}{b_2}y\right)}e^{\frac{j}{2}\left(\frac{a_1}{b_1}x^2+\frac{a_2}{b_2}y^2\right)}g(x+py,qx+y)dxdy \tag{136}$$

This is the combination of 2-D separable canonical transform (with the parameters $\{a_1,\ b_1,\ -1/b_1,\ 0\}$ for x-axis and $\{a_2,\ b_2,\ -1/b_2,\ 0\}$ for y-axis) and the geometric twisting operation. It is the special case of 2-D AGFFT with the parameters that

$$\mathbf{A}=\begin{bmatrix} a_1/(1-pq) & -a_1p/(1-pq) \\ -a_2q/(1-pq) & a_2/(1-pq)\end{bmatrix},\qquad \mathbf{B}=\begin{bmatrix} b_1 & b_1q \\ b_2p & b_2\end{bmatrix},$$
$$\mathbf{C}=\begin{bmatrix} -1/b_1(1-pq) & p/b_1(1-pq) \\ q/b_2(1-pq) & -1/b_2(1-pq)\end{bmatrix},\qquad \mathbf{D}=\mathbf{0}. \tag{137}$$

It is also the special case of the simplified AGFFT defined as Eq. (131). It totally has 6 parameters. It has emphasized the combination of the 2-D separable canonical transform and the geometric twisting, and changes the 2 axes of the 2-D separable canonical transform from $\{x,\ 0\}$, $\{0,\ y\}$ into $\{x,\ qx\}$, $\{py,\ y\}$. It is suitable for many applications, such as the pattern recognition.

V. Applications

Because the 2-D AGFFT is the generalization of the 2-D separable fractional Fourier transform, so many of the applications of the fractional Fourier transform can be extended for the 2-D AGFFT. We will discuss some applications of the 2-D AGFFT in the following.

5-1 Filter design

The filter for the 2-D AGFFT acts as the following formula:

$$\hat{s}(x,y)=O_F^{(\mathbf{D}^{\mathrm{T}},-\mathbf{B}^{\mathrm{T}},-\mathbf{C}^{\mathrm{T}},\mathbf{A}^{\mathrm{T}})}\left(O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(s(x,y))\cdot Z(f,h)\right), \tag{138}$$

i.e., the 2-D affine generalized convolution of the signal $s(x, y)$ and the filter $Z(f, h)$. There are at least 2 ways to design the filter with 2-D AGFFT:

(1) Optimal filter. (2) Pass-Stop band filter.

We first discuss the optimal filter. Because the 2-D AGFFT is also an orthonormal transform, so the formula of the optimal filter for 1-D FRFT [11] can also be applied here with a little modification. We suppose

(a) the correlation between the input signal $g(x, y)$ and the received signal $s(x, y)$ (denoted by $R_{gs}(x, y, \sigma, \tau)$),

(b) the auto-correlation of the received signal (denoted by $R_{ss}(x, y, \sigma, \tau)$)

has been known in advance. Then the optimal filter can be calculated from

$$z_{opt}(f,h)=\frac{R_{G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}S_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(f,h,f,h)}{R_{S_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}S_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(f,h,f,h)}. \tag{139}$$

And $R_{G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}S_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(f,h,f,h)$, $R_{S_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}S_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(f,h,f,h)$ can be calculated from

$$R_{G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}S_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(f,h,f,h)=\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}K_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,x,y)\right.$$
$$\left.K^{*}_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,\sigma,\tau)R_{gs}(x,y,\sigma,\tau)dxdyd\sigma d\tau\right], \tag{140}$$

$$R_{S_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}S_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(f,h,f,h)=\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}K_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,x,y)\right.$$
$$\left.K^{*}_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,\sigma,\tau)R_{ss}(x,y,\sigma,\tau)dxdyd\sigma d\tau\right]. \tag{141}$$

To calculating the mean square error (MSE), suppose we also have known

(c) the auto-correlation of the input signal (denoted by $R_{gg}(x, y, \sigma, \tau)$).

Then we can calculate $R_{G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(f,h,f,h)$ from

$$R_{G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(f,h,f,h)=\left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}K_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,x,y)\right.$$
$$\left.K^{*}_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,\sigma,\tau)R_{gg}(x,y,\sigma,\tau)dxdyd\sigma d\tau\right]. \tag{142}$$

And the MSE for the optimal filter can be calculated as:

$$MSE=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left[R_{G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(f,h,f,h)-2\,\mathrm{Re}\left(z^{*}_{opt}(f,h)R_{S_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(f,h,f,h)\right)\right.$$
$$\left.+\left|z_{opt}(f,h)\right|^{2}R_{S_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}S_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}}(f,h,f,h)\right]\cdot dfdh \tag{143}$$

We can use Eq. (139) to determine the optimal filter for each choice of {**A**, **B**, **C**, **D**}, and then use Eq. (143) in iterative to determine the optimal choice of {**A**, **B**, **C**, **D**}. Because the 2-D

AGFFT defined as Eqs. (18)~(21) has totally the free dimension of 10, so it is very difficult to determine the optimal choice of {**A**, **B**, **C**, **D**}. But if we use the modified 2-D AGFFT defined as Eq. (134), then there are only 3 free parameters, and the optimal choice of the parameters is much easier to determine.

We then discuss the pass-stop band filter. That is, $Z(f, h) = 1$ in the passband, and $Z(f, h) = 0$ in the stopband. This type of filter is especially useful when the noise is as the form of:

$$n(x,y) = P_0 \cdot \exp\left(j(p_1x^2 + p_2xy + p_3y^2 + p_4x + p_5y)\right), \tag{144}$$

or in general

$$n(x,y) = \sum_{s=0}^{S-1} P_{s,0} \cdot \exp\left(j(p_{s,1}x^2 + p_{s,2}xy + p_{s,3}y^2 + p_{s,4}x + p_{s,5}y)\right). \tag{145}$$

If in Eq. (145), $p_{s,2} = 0$ for all s, then this type of noise can just be removed by the 2-D separable canonical transform. But if $p_{s,2} \neq 0$ for some s, then we must use the 2-D AGFFT introduced in this paper to remove the noise. To remove the noise as the form of Eq. (144), we can choose the parameters {**A**, **B**, **C**, **D**} of the 2-D AGFFT as:

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} -\Psi & \mathbf{I} \\ -\mathbf{I} & \mathbf{0} \end{bmatrix}, \qquad \text{where} \quad \Psi = \begin{bmatrix} 2p_1 & p_2 \\ p_2 & 2p_3 \end{bmatrix}. \tag{146}$$

This is because

$$\begin{aligned} O_F^{(-\Psi,\mathbf{I},-\mathbf{I},\mathbf{0})}\left(P_0 e^{j(p_1x^2+p_2xy+p_3y^2+p_4x+p_5y)}\right) &= FT\left(O_F^{(\mathbf{I},\mathbf{0},-\Psi,\mathbf{I})}\left(P_0 e^{j(p_1x^2+p_2xy+p_3y^2+p_4x+p_5y)}\right)\right) \\ &= FT\left(P_0 e^{j(p_4x+p_5y)}\right) \\ &= 2\pi\sqrt{-1}\cdot P_0 \cdot \delta(f-p_4, h-p_5). \end{aligned} \tag{147}$$

So the noise as Eq. (144) will become the impulse function after the 2-D AGFFT with the parameters as Eq. (146). To remove the noise as Eq. (145), we can repeat the filter operation several times, and for each time we choose the parameters of the 2-D AGFFT according to Eq. (146) ($\{p_1, p_2, p_3\}$ is changed as $\{p_{s,1}, p_{s,2}, p_{s,3}\}$) to remove one of the quadratic phase components in Eq. (145).

But when we don't know the components of the noise, we can't apply the method described above. In this case, we must assort to other ways to search the parameters {**A**, **B**, **C**, **D**} of the 2-D AGFFT. For example, we can calculate the 2-D Wigner distribution function (2-D WDF, described in subsection 3-1) for $s(x, y) + n(x, y)$ where $s(x, y)$ is the signal and $n(x, y)$ is the noise, and use the WDF to conclude we must choose which set of parameters. Since we would require much effort to discuss this method in detail, so we will not discuss it in this paper.

We will give an example as below. Here the signal is the fruit image with the size of 256×256 shown in Fig. 5 (the location (0, 0) is at the center), and the noise is as below:

$$n(x, y) = \exp(j \cdot 0.001(2x^2 - 10xy + 1.5y^2)). \quad (148)$$

In Fig. 6, we plotted the fruit plus the noise. This noise can't be moved by the separable 2-D canonical transforms because of the existence of the unseparable term $\exp(j\eta xy)$. But we can remove it by the 2-D AGFFT defined as Eqs. (18)~(21), and its simplified form of Eq. (134). We choose the parameters as $\mathbf{B} = -\mathbf{C} = \mathbf{I}$, $\mathbf{D} = \mathbf{0}$, $a_{11} = -0.004$, $a_{12} = a_{21} = 0.01$, $a_{22} = -0.003$. In Fig. 7, we show the result of the 2-D AGFFT of the Fig. 6 with the parameters described above. After the 2-D AGFFT, we use the filter as:

$$Z(f, h) = 1 - \delta(f, h). \quad (149)$$

Then, we doing the inverse 2-D AGFFT, and obtain the recovered signal as Fig. 8. We find, the noise has been completely removed.

5-2 Optical implementation for the 2-D AGFFT

The 2-D AGFFT can also be applied to the optical system analysis for the monochromic light. The 2-D separable canonical transform has been used for this application [1]. It can analyze the cylinder lens with the width-variation direction of x or y-axis. But for the **tilted cylinder lens**, the 2-D separable canonical transform will fail to analyze it, and we must use the 2-D AGFFT defined as Eqs. (18)~(21). For example, for the optical system as Fig. 4, the transform function of the first lens is:

$$t_1(x, y) = e^{-j\pi(y\cos\alpha - x\sin\alpha)^2 / \lambda \cdot f_1}, \quad (150)$$

and it corresponds to the 2-D AGFFT with the parameters:

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \Phi & \mathbf{I} \end{bmatrix}, \quad \text{where } \Phi = \begin{bmatrix} \dfrac{-k\sin^2\alpha}{f_1} & \dfrac{k\sin\alpha\cos\alpha}{f_1} \\ \dfrac{k\sin\alpha\cos\alpha}{f_1} & \dfrac{-k\cos^2\alpha}{f_1} \end{bmatrix}, \quad k = 2\pi/\lambda. \quad (151)$$

Then, the overall system in Fig. 4 will correspond to the 2-D AGFFT with the parameters as:

$$\begin{bmatrix} \mathbf{A}' & \mathbf{B}' \\ \mathbf{C}' & \mathbf{D}' \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \Omega & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{I} & \Psi \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \Phi & \mathbf{I} \end{bmatrix}, \quad (152)$$

$$\text{where } \Omega = \begin{bmatrix} 0 & 0 \\ 0 & -k/f_2 \end{bmatrix}, \quad \Psi = \begin{bmatrix} d/k & 0 \\ 0 & d/k \end{bmatrix}. \quad (153)$$

Besides, the 2-D AGFFT can also be applied for describing the monochromic light propagates in the gradient index medium. In Ref. [12], they state if the monochromic light propagates in the **elliptic gradient index (elliptic GRIN) medium** with the refractive index as:

$$n^2(x, y) = n_0^2\left[1 - (n_x / n_0)x^2 - (n_y / n_0)y^2\right], \quad (154)$$

and the propagation length is L, then

$$T(g(x,y)) = \exp(-j2\pi n_0 L/\lambda)\cdot O_{Fx}^{\alpha} O_{Fy}^{\beta}(g(x,y)), \tag{155}$$

where $O_{Fx}{}^{\alpha}$ is the 1-D SAFT along the x-axis with parameters

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sqrt{\frac{n_0}{n_x}}\frac{\sin\alpha}{k} \\ -\sqrt{\frac{n_x}{n_0}}k\sin\alpha & \cos\alpha \end{bmatrix} \quad \text{where} \quad \alpha = \frac{2L}{\pi}\sqrt{\frac{n_x}{n_0}}, \tag{156}$$

and $O_{Fy}{}^{\beta}$ is the 1-D SAFT along the y-axis with parameters similar as above, except for that α is changed as β and n_x is changed as n_y.

Thus, we can conclude if the monochromic light propagates in the elliptic GRIN medium with the refractive index as:

$$n^2(x,y) = n_0^2\left[1-(n_1/n_0)(x+py)^2-(n_2/n_0)(qx+y)^2\right], \tag{157}$$

and the propagation length is L, then the result can be represented by the 2-D AGFFT:

$$T(g(x,y)) = \exp(-j2\pi n_0 L/\lambda)\cdot O_F^{(A,B,C,D)}(g(x,y)), \tag{158}$$

where

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} = \begin{bmatrix} 1/\Delta & -p/\Delta & 0 & 0 \\ -q/\Delta & 1/\Delta & 0 & 0 \\ 0 & 0 & 1 & q \\ 0 & 0 & p & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos\alpha & 0 & \frac{\sin\alpha}{m_1 k} & 0 \\ 0 & \cos\beta & 0 & \frac{\sin\beta}{m_2 k} \\ -m_1 k\sin\alpha & 0 & \cos\alpha & 0 \\ 0 & -m_2 k\sin\beta & 0 & \cos\beta \end{bmatrix} \cdot \begin{bmatrix} 1 & p & 0 & 0 \\ q & 1 & 0 & 0 \\ 0 & 0 & 1/\Delta & -q/\Delta \\ 0 & 0 & -p/\Delta & 1/\Delta \end{bmatrix}, \tag{159}$$

and

$$m_1 = \sqrt{\frac{n_1}{n_0}}, \quad m_2 = \sqrt{\frac{n_2}{n_0}}, \quad \alpha = \frac{2L}{\pi}m_1, \quad \beta = \frac{2L}{\pi}m_2, \quad \Delta = 1-pq. \tag{160}$$

5-3 Other potential applications

In the 1-D case, we can use fractional correlation for pattern recognition [13][14]. Thus, in the 2-D case, we can also use the 2-D affine generalized fractional correlation defined as Eq. (81) for the pattern recognition. When using the generalized fractional correlation for pattern recognition, objects will be identified only when:

(1) The objects are similar as the reference pattern.

(2) The objects are in the certain region (Because 2-D AGFFT is partially space variant).

Besides, the 2-D AGFFT can also be applied for generalizing the 2-D Hilbert transform, mask, signal synthesis, beam shaping, 2-D signal compression, and phase retrieval, etc.

VI. Conclusion

We have introduced the 2-D AGFFT and some important properties, its calculation and digital implementation, and applications for the filter design, optical system analysis, etc.

The 2-D AGFFT generalizes the 2-D separable FRFT and the 2-D canonical transform introduced by [1], and mixed them with the geometric twisting operation. Thus, the 2-D AGFFT not only can extend most of the applications for 1-D fractional Fourier transform, but also be a useful tool for the pattern recognition with some spatial distortion. The main disadvantages for the generalized transform are the number of the parameter increases and the calculation will become more complex. But with the method introduced in section 4, we find if M, N are large, and the re-sampling algorithm we use is not very complex, then the complexity of digital implementation for the 2-D AGFFT is still proportion to $(MN/2)\log_2 MN$ for most of the cases, as the 2-D separable FRFT. Besides, using the simplified forms introduced in subsection 4-2, we can reduce the number of parameters with very little effect on the utility of 2-D AGFFT. Thus, in fact the 2-D AGFFT is only a little complex than the 2-D separable FRFT, but the utilities would be extended a lot. We believe, the 2-D AGFFT will be a popular tool for the 2-D signal processing in the future.

Reference

[1] A. Sahin, H. M. Ozaktas, and D. Mendlovic, 'Optical implementation of two-dimensional fractional Fourier transforms and linear canonical transforms with arbitrary parameters', *Appl. Opt.*, vol. 37, no. 11, p 2130-2141, Apr 1998.

[2] A. Sahin, M. A. Kutay, and H. M. Ozaktas, 'Nonseparable two-dimensional fractional Fourier transform`, *Appl. Opt.*, vol. 37, no. 23, p 5444-5453, Apr 1998.

[3] V. Namias, 'The fractional order Fourier transform and its application to quantum mechan-

ics', *J. Inst. Maths. Applics.*, vol. 25, p 241-265, 1980.

[4] L. B. Almeida, 'The fractional Fourier transform and time-frequency representations', *IEEE Trans. Signal Processing*, vol. 42, no. 11, p 3084-3091, Nov. 1994.

[5] M. Moshinsky and C. Quesne, 'Linear canonical transformations and their unitary representations', *J. Math. Phys.*, vol. 12, no. 8, p 1772-1783, Aug, 1971.

[6] S. Abe and J. T. Sheridan, 'Optical operations on wave functions as the Abelian subgroups of the special affine Fourier transformation', *Opt. Lett.*, v. 19, n. 22, p 1801-1803, 1994.

[7] G. B. Folland, "*Harmonic Analysis in Phase Space*", the Annals of Math. Studies vol. 122, Princeton University Press, 1989.

[8] J. J. Ding, "*Derivation and Properties of Orthogonal transform*", Master Thesis, National Taiwan University, 1997.

[9] H. M. Ozaktas, B. Barshan, D. Mendlovic, and L. Onural, 'Convolution, filtering, and multiplexing in fractional Fourier domains and their rotation to chirp and wavelet transform', *J. Opt. Soc. Am. A*, Vol. 11, No. 2, p 547-559, Feb. 1994.

[10] D. Mendlovic, H. M. Ozaktas, and A. W. Lohmann, 'Fractional correlation', *Appl. Opt.*, Vol. 34, p 303-309, 1994.

[11] M. A. Kutay, H. M. Ozaktas, O. Arikan, and L. Onural, 'Optimal filters in fractional Fourier domain', *IEEE Trans. Signal Processing*, v. 45, n. 5, p 1129-1143, May 1997.

[12] L. Yu, M. Huang, L. Wu, Y. Lu, W. Huang, M. Chen, and Z. Zhu, 'Fractional Fourier transform and the elliptic gradient-index medium', *Opt. Commun.*, v. 152, p 23-25, 1998.

[13] A. W. Lohmann, Z. Zalevsky, and D. Mendlovic, 'Synthesis of pattern recognition filters for fractional Fourier processing', *Opt. Commun.*, vol. 128, p 199-204, Jul. 1996.

[14] M. A. Kutay and H. M. Ozaktas, 'Optimal image restoration with the fractional Fourier transform', *J. Opt. Soc. Am. A*, vol. 15, no. 4, p 825-833, Apr. 1998.

[15] T. A. C. M. Classen and W. F. G. Mecklenbrauker, "The Wigner distribution – a tool for time-frequency signal analysis; Part I: continuous time signals", *Philips J. Res.*, vol. 35, p. 217-250, 1980.

[16] M. F. Erden, H. M. Ozaktas, A. Sahin, and D. Mendlovic, 'Design of dynamically adjustable anamorphic fractional Fourier transform', *Opt. Commun.*, v. 136, p. 52-60, 1997.

[17] J. J. Ding and S. C. Pei, '2-D affine generalized fractional Fourier transform', *ICASSP*, p. 3181-3184, 1999.

[18] J. W. Goodman, "*Introduction to Fourier Optics*", McGraw-Hill, 2nd ed., 1988.

Appendix

• Proof of the properties

(Proof of the properties 1, 2 in Table 2):

We have known [15]

$$|g(x,y)|^2 = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_g(x, y, f, h)\cdot dfdh, \tag{161}$$

$$g(x,y) = \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_g\left(\frac{x}{2}, \frac{y}{2}, f, h\right)\cdot e^{j\cdot(f\cdot x+h\cdot y)}\cdot dfdh\right] / \overline{g(0,0)}. \tag{162}$$

So if $W_{G_{(A,B,C,D)}}(x, y, f, h)$ is the 2-D WDF of $G_{(A,B,C,D)}(f, h)$, then

$$\left|G_{(A,B,C,D)}(x,y)\right|^2 = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_{G_{(A,B,C,D)}}(x, y, f, h)\cdot dfdh, \tag{163}$$

$$G_{(A,B,C,D)}(x,y) = \left[\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_{G_{(A,B,C,D)}}\left(\frac{x}{2}, \frac{y}{2}, f, h\right)\cdot e^{j\cdot(f\cdot x+h\cdot y)}\cdot dfdh\right] / \overline{G_{(A,B,C,D)}(0,0)}. \tag{164}$$

And then together with the Eq. (86), we can obtain the properties 1, 2 in Table 2.

(Proof of property 3 in Table 2):

If $s(x, y) = p(x, y)q(x, y)$, and $W_s(x, y, f, h)$, $W_p(x, y, f, h)$, $W_q(x, y, f, h)$ are the WDF of $s(x, y)$, $p(x, y)$, $q(x, y)$, then [15]

$$W_s(\vec{\mathbf{x}}, \vec{\mathbf{w}}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_p(\vec{\mathbf{x}}, \vec{\rho})\cdot W_q(\vec{\mathbf{x}}, \vec{\mathbf{w}} - \vec{\rho})\cdot d\tau d\rho, \tag{165}$$

where $\vec{\mathbf{x}} = [x, y]$, $\vec{\mathbf{w}} = [f, h]$, $\vec{\rho} = [\tau, \sigma]$.

If $z(x, y)$ is the 2-D affine generalized fractional convolution of $f(x, y)$ and $g(x, y)$, as Eq. (79), then $Z_{(A,B,C,D)}(f, h) = F_{(A,B,C,D)}(f, h)\cdot G_{(A,B,C,D)}(f, h)$, and from Eq. (165),

$$W_{Z_{(A,B,C,D)}}(\vec{\mathbf{x}}, \vec{\mathbf{w}}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_{F_{(A,B,C,D)}}(\vec{\mathbf{x}}, \vec{\rho}) W_{g_{(A,B,C,D)}}(\vec{\mathbf{x}}, \vec{\mathbf{w}} - \vec{\rho})\cdot d\tau d\rho. \tag{166}$$

Then from Eq. (86), we know

$$W_{Z_{(A,B,C,D)}}(\vec{\mathbf{x}}, \vec{\mathbf{w}}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_f(\vec{\mathbf{x}}\mathbf{D} - \vec{\rho}\mathbf{B}, -\vec{\mathbf{x}}\mathbf{C} + \vec{\rho}\mathbf{A})\cdot W_g(\vec{\mathbf{x}}\mathbf{D} - \vec{\mathbf{w}}\mathbf{B} + \vec{\rho}\mathbf{B}, -\vec{\mathbf{x}}\mathbf{C} + \vec{\mathbf{w}}\mathbf{A} - \vec{\rho}\mathbf{A})\cdot d\tau d\rho.$$

Then, from Eq. (85),

$$\begin{aligned} W_z(\vec{\mathbf{x}}, \vec{\mathbf{w}}) &= W_{Z_{(A,B,C,D)}}(\vec{\mathbf{x}}\mathbf{A}^{\mathrm{T}} + \vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}, \vec{\mathbf{x}}\mathbf{C}^{\mathrm{T}} + \vec{\mathbf{w}}\mathbf{D}^{\mathrm{T}}) \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_f(\vec{\mathbf{x}}\mathbf{A}^{\mathrm{T}}\mathbf{D} + \vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}\mathbf{D} - \vec{\rho}\mathbf{B}, -\vec{\mathbf{x}}\mathbf{A}^{\mathrm{T}}\mathbf{C} - \vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}\mathbf{C} + \vec{\rho}\mathbf{A})\cdot \\ &\qquad W_g(\vec{\mathbf{x}} + \vec{\rho}\mathbf{B}, \vec{\mathbf{w}} - \vec{\rho}\mathbf{A})\cdot d\tau d\sigma \end{aligned}$$

(Proof of property 3 in Table 3):

From Eqs. (18)~(21), we find

$$\frac{\partial}{\partial f}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\frac{j}{\det(\mathbf{B})}((d_{11}b_{22}-d_{12}b_{21})f+(-d_{11}b_{12}+d_{12}b_{11})h$$
$$-b_{22}x+b_{21}y)\cdot K_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,x,y)\cdot g(x,y)\cdot dxdy \quad (167)$$

$$\frac{\partial}{\partial h}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\frac{i}{\det(\mathbf{B})}((-d_{21}b_{12}+d_{22}b_{11})h+(-d_{11}b_{12}+d_{12}b_{11})f$$
$$+b_{12}x-b_{11}y)\cdot K_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,x,y)\cdot g(x,y)\cdot dxdy \quad (168)$$

Then we multiply Eq. (167) by b_{11}, multiply Eq. (168) by b_{21}, sum them together, and use the fact that $-d_{11}b_{12}+d_{12}b_{11}=d_{21}b_{22}-d_{22}b_{21}$ (the 2nd constraint of 2-D AGFFT in Eq. (22)), we will obtain the property 3.

(Proof of property 5 in Table 3):

From Eqs. (26) and (20), we obtain

$$\frac{\partial}{\partial x}g(x,y)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\frac{j}{\det(\mathbf{B})}((-a_{11}b_{22}+a_{21}b_{12})x+(a_{11}b_{21}-a_{21}b_{11})y+b_{22}f$$
$$-b_{12}h)\cdot K_{(\mathbf{D}^{\mathrm{T}},-\mathbf{B}^{\mathrm{T}},-\mathbf{C}^{\mathrm{T}},\mathbf{A}^{\mathrm{T}})}(f,h,x,y)\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)\cdot dfdh \quad (169)$$

It can be written as:

$$O_F^{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}\left(\frac{\partial}{\partial x}g(x,y)+\frac{j}{\det(\mathbf{B})}((a_{11}b_{22}-a_{21}b_{12})x+(-a_{11}b_{21}+a_{21}b_{11})y)\cdot g(x,y)\right) \quad (170)$$
$$=(b_{22}f-b_{12}h)\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)$$

Then substituting the properties 3, 4 into the above equation, we obtain the property 5.

(Proof of property 3 in Table 4):

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)S^*_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)dfdh$$
$$=\frac{1}{4\pi^2|\det(\mathbf{B})|}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-j(\vec{\mathbf{t}}_1-\vec{\mathbf{t}}_2)\mathbf{B}^{-1}\vec{\mathbf{w}}^{\mathrm{T}}}e^{\frac{j}{2}\vec{\mathbf{t}}_1\mathbf{B}^{-1}\mathbf{A}\vec{\mathbf{t}}_1^{\mathrm{T}}}e^{-\frac{j}{2}\vec{\mathbf{t}}_2\mathbf{B}^{-1}\mathbf{A}\vec{\mathbf{t}}_2^{\mathrm{T}}}\cdot g(\vec{\mathbf{t}}_1)s^*(\vec{\mathbf{t}}_2)\cdot$$
$$dfdhdx_1dy_1dx_2dy_2$$

where $\vec{\mathbf{t}}_1=(x_1,y_1)$, $\vec{\mathbf{t}}_2=(x_2,y_2)$, $\vec{\mathbf{w}}=(f,h)$. Then because

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}e^{-j(\vec{\mathbf{t}}_1-\vec{\mathbf{t}}_2)\mathbf{B}^{-1}\vec{\mathbf{w}}^{\mathrm{T}}}dfdh=4\pi^2|\det(\mathbf{B})|\cdot\delta(\vec{\mathbf{t}}_1-\vec{\mathbf{t}}_2),$$

so we obtain

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)S^*_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)dfdh=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}g(x,y)s^*(x,y)dxdy.$$

(Proof of property 4 in Table 5):

Suppose $\widetilde{G}_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)$ is the 2-D AGFFT of $g(x,y)/x$:

$$\widetilde{G}_{(\mathrm{A,B,C,D})}(\vec{\mathbf{w}}) = \frac{1}{2\pi\sqrt{-\det(\mathbf{B})}} \exp\left(\frac{j}{2}\vec{\mathbf{w}}\mathbf{D}\mathbf{B}^{-1}\vec{\mathbf{w}}^{\mathrm{T}}\right) \int \exp\left(\frac{j}{2}\left(-\vec{\mathbf{x}}\mathbf{B}^{-1}\vec{\mathbf{w}}^{\mathrm{T}} + \vec{\mathbf{x}}\mathbf{B}^{-1}\mathbf{A}\vec{\mathbf{x}}^{\mathrm{T}}\right)\right)\frac{g(\vec{\mathbf{x}})}{x} d\vec{\mathbf{x}},$$

where $\vec{\mathbf{x}} = [x, y]$, $\vec{\mathbf{w}} = [f, h]$. Then

$$\widetilde{G}_{(\mathrm{A,B,C,D})}(\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}) = \frac{1}{2\pi\sqrt{-\det(\mathbf{B})}} \exp\left(\frac{j}{2}\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}\mathbf{D}\vec{\mathbf{w}}^{\mathrm{T}}\right) \int \exp\left(\frac{j}{2}\left(-\vec{\mathbf{x}}\vec{\mathbf{w}}^{\mathrm{T}} + \vec{\mathbf{x}}\mathbf{B}^{-1}\mathbf{A}\vec{\mathbf{x}}^{\mathrm{T}}\right)\right)\frac{g(\vec{\mathbf{x}})}{x} d\vec{\mathbf{x}},$$

so

$$\exp\left(-\frac{j}{2}\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}\mathbf{D}\vec{\mathbf{w}}^{\mathrm{T}}\right)\widetilde{G}_{(\mathrm{A,B,C,D})}(\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}) = \frac{1}{2\pi\sqrt{-\det(\mathbf{B})}} \int \exp\left(\frac{j}{2}\left(-\vec{\mathbf{x}}\vec{\mathbf{w}}^{\mathrm{T}} + \vec{\mathbf{x}}\mathbf{B}^{-1}\mathbf{A}\vec{\mathbf{x}}^{\mathrm{T}}\right)\right)\frac{g(\vec{\mathbf{x}})}{x} d\vec{\mathbf{x}},$$

$$\frac{\partial}{\partial f}\left[\exp\left(-j\vec{\mathbf{w}}\cdot\mathbf{B}^{\mathrm{T}}\mathbf{D}\cdot\vec{\mathbf{w}}^{\mathrm{T}}/2\right)\cdot\widetilde{G}_{(\mathrm{A,B,C,D})}(\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}})\right]$$

$$= \frac{-j}{2\pi\sqrt{-\det(\mathbf{B})}} \int \exp\left(j\left(-\vec{\mathbf{x}}\cdot\vec{\mathbf{w}}^{\mathrm{T}} + \vec{\mathbf{x}}\cdot\mathbf{B}^{-1}\mathbf{A}\cdot\vec{\mathbf{x}}^{\mathrm{T}}/2\right)\right)\cdot g(\vec{\mathbf{x}})\cdot d\vec{\mathbf{x}}. \tag{171}$$

Since

$$G_{(\mathrm{A,B,C,D})}(\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}) = \frac{1}{2\pi\sqrt{-\det(\mathbf{B})}} \exp\left(\frac{j}{2}\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}\mathbf{D}\vec{\mathbf{w}}^{\mathrm{T}}\right) \int \exp\left(\frac{j}{2}\left(-\vec{\mathbf{x}}\vec{\mathbf{w}}^{\mathrm{T}} + \vec{\mathbf{x}}\mathbf{B}^{-1}\mathbf{A}\vec{\mathbf{x}}^{\mathrm{T}}\right)\right) g(\vec{\mathbf{x}}) d\vec{\mathbf{x}},$$

so Eq. (171) can be rewritten as:

$$\frac{\partial}{\partial f}\left[\exp\left(-\frac{j}{2}\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}\mathbf{D}\vec{\mathbf{w}}^{\mathrm{T}}\right)\widetilde{G}_{(\mathrm{A,B,C,D})}(\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}})\right] = -j\exp\left(-\frac{j}{2}\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}\mathbf{D}\vec{\mathbf{w}}^{\mathrm{T}}\right)G_{(\mathrm{A,B,C,D})}(\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}),$$

$$\exp\left(-\frac{j}{2}\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}\mathbf{D}\vec{\mathbf{w}}^{\mathrm{T}}\right)\widetilde{G}_{(\mathrm{A,B,C,D})}(\vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}) = -j\int_{-\infty}^{f}\exp\left(-\frac{j}{2}\vec{\mathbf{v}}\mathbf{B}^{\mathrm{T}}\mathbf{D}\vec{\mathbf{v}}^{\mathrm{T}}\right)G_{(\mathrm{A,B,C,D})}(\vec{\mathbf{v}}\mathbf{B}^{\mathrm{T}})dp,$$

where $\vec{\mathbf{v}} = [p, h]$. Thus, if we replace $\vec{\mathbf{w}}$ by $\vec{\mathbf{w}}(\mathbf{B}^{\mathrm{T}})^{-1}$, then

$$\widetilde{G}_{(\mathrm{A,B,C,D})}(\vec{\mathbf{w}}) = -j\exp\left(j\vec{\mathbf{w}}\cdot\mathbf{D}\mathbf{B}^{-1}\cdot\vec{\mathbf{w}}^{\mathrm{T}}/2\right)\cdot$$
$$\int_{-\infty}^{z}\exp\left(-j\vec{\mathbf{s}}\cdot\mathbf{B}^{\mathrm{T}}\mathbf{D}\cdot\vec{\mathbf{s}}^{\mathrm{T}}/2\right)\cdot G_{(\mathrm{A,B,C,D})}(\vec{\mathbf{s}}\mathbf{B}^{\mathrm{T}})\cdot dp,$$

where $z = (f\cdot b_{22} - h\cdot b_{12})/\det(\mathbf{B})$, $\vec{\mathbf{s}} = [p, q]$, $q = (-f\cdot b_{21} + h\cdot b_{11})/\det(\mathbf{B})$.

• List of the abbreviations used in this paper:

FRFT: fractional Fourier transform. (see Eq. (1)).

SAFT: special affine Fourier transform, also called as the canonical transform (Eqs. (3), (4)).

FT: Fourier transform.

IFT: inverse Fourier transform

2-D AGFFT: 2-D affine generalized fractional Fourier transform (Eqs. (18) ~ (21)).

2-D AGFCV: 2-D affine generalized fractional convolution (Eq. (79)).

2-D AGFCR: 2-D affine generalized fractional correlation (Eqs. (80), (81)).

2-D WDF: 2-D Wigner distribution function (Eq. (82)).

2-D TFAGFFT: 2-D AGFFT with space-shifting and frequency modulation (Eq. (89)).

Figures and Tables Captions

Figures and Tables

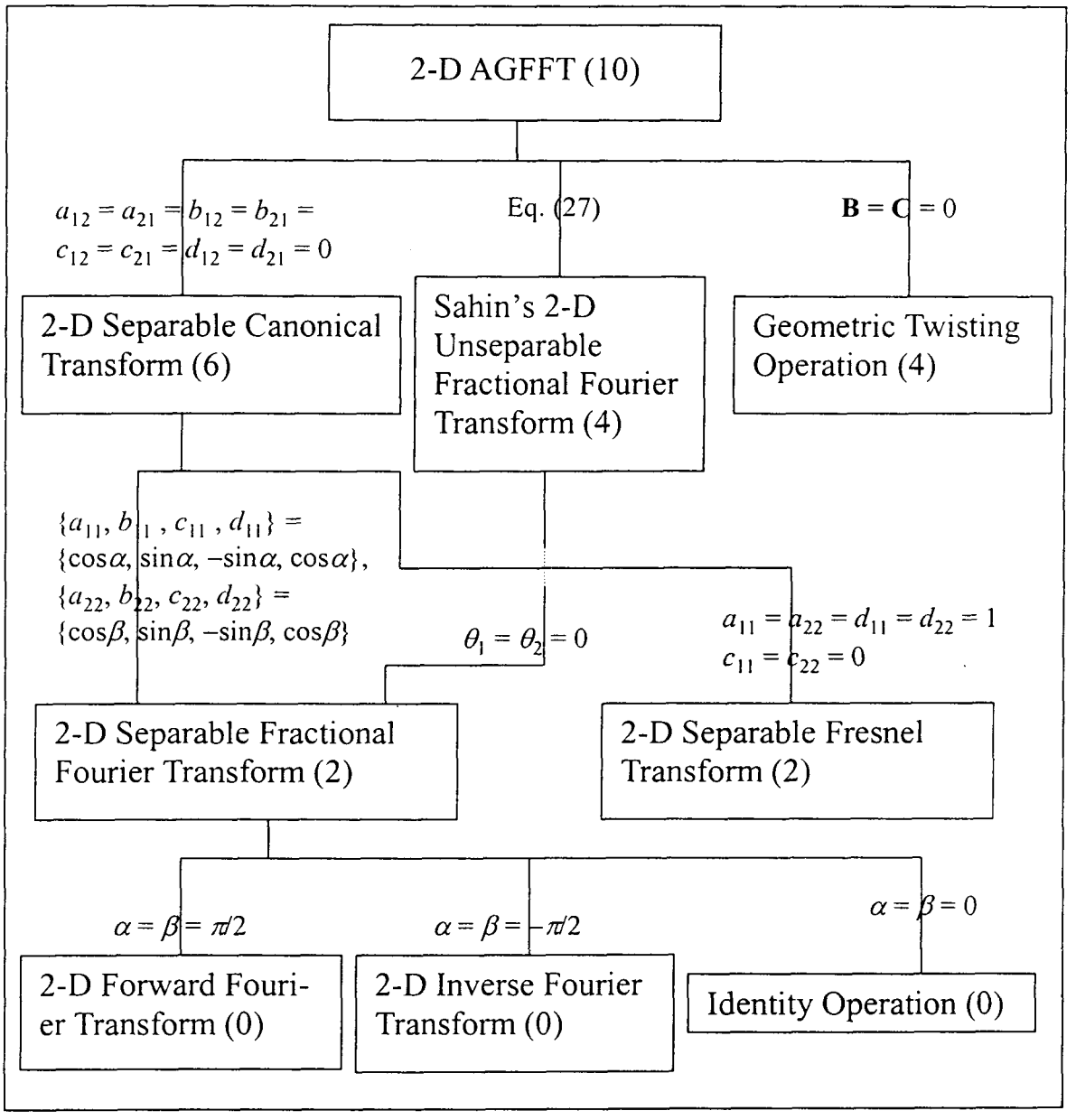


Fig. 1

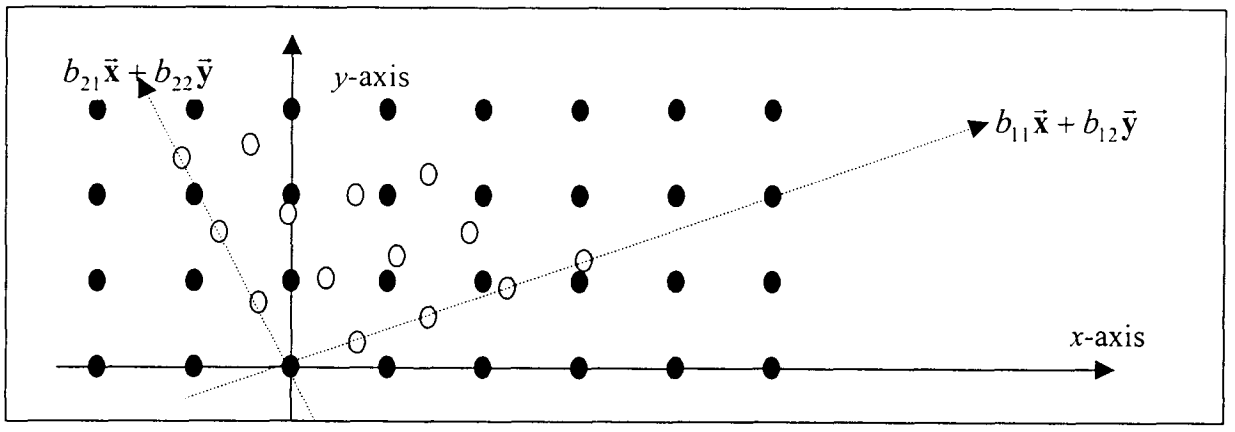


Fig. 2

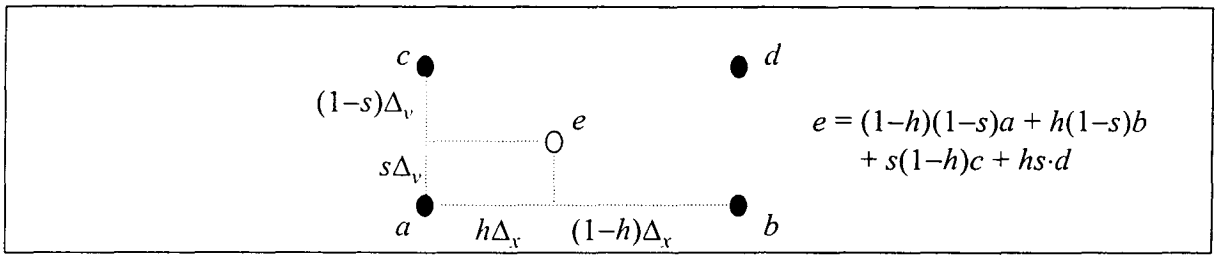


Fig. 3

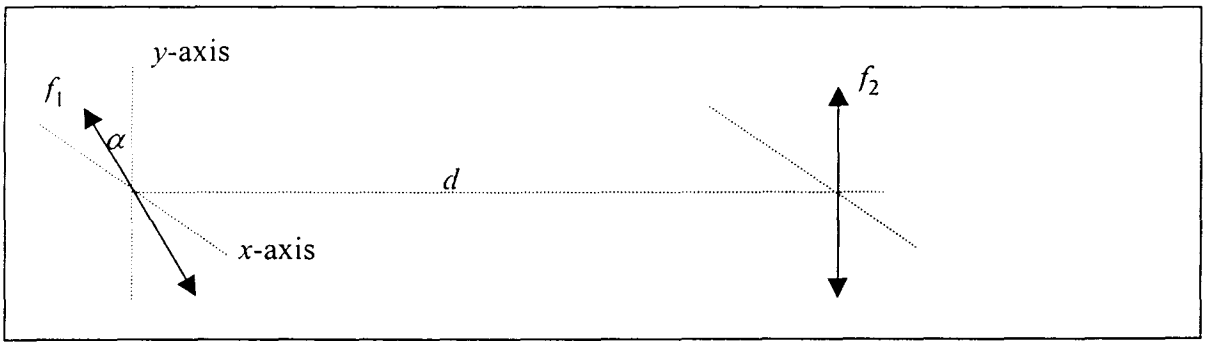


Fig. 4

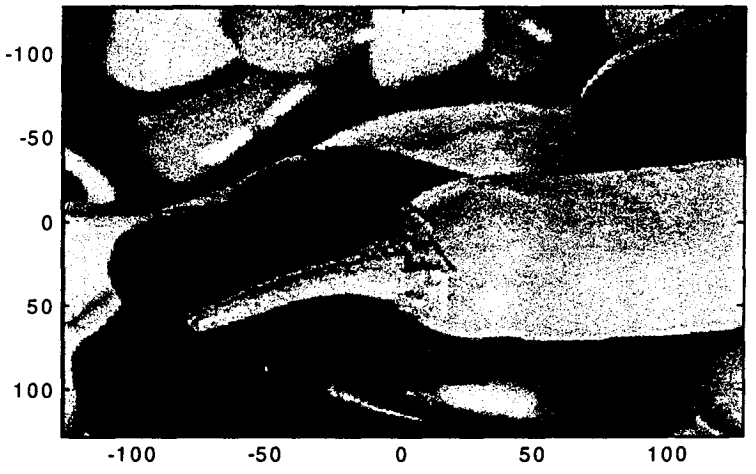


Fig. 5

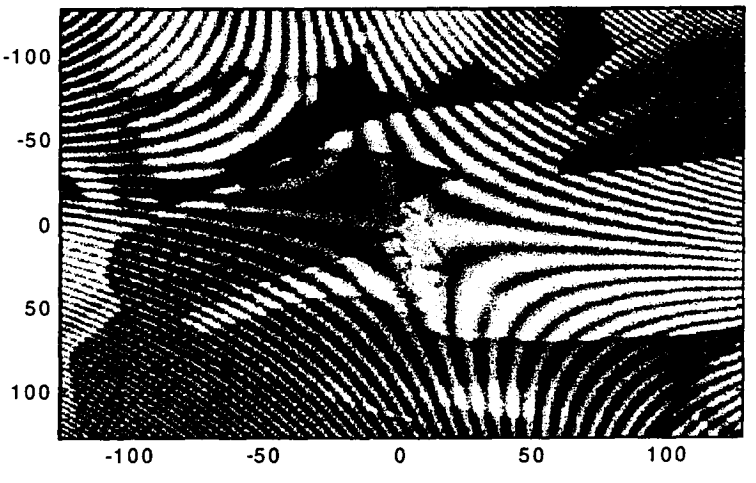


Fig. 6

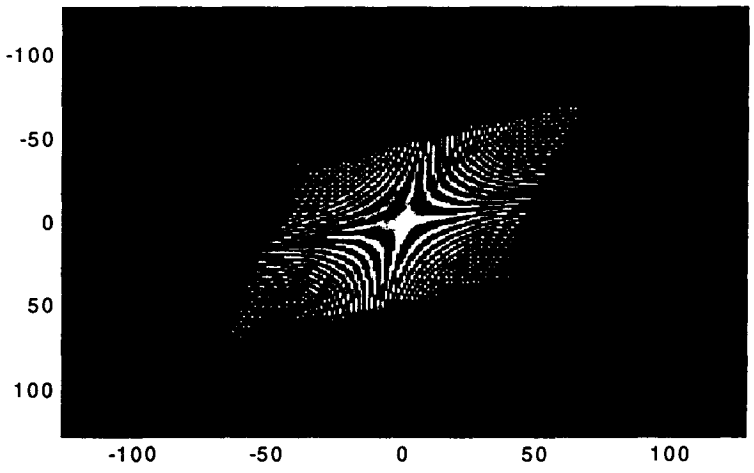


Fig. 7

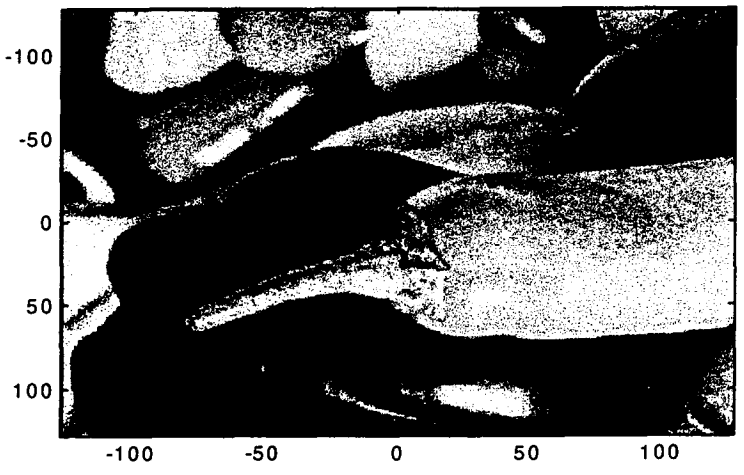


Fig. 8

Basic ops.	(A)	(B)	(C)	(D)	(E)	(F)	(G)	2-D AGFFT
(1)	×	×	O	×	O	×	×	O
(2)	×	O	×	O	O	×	×	O
(3)	×	×	×	×	O	×	O	O
(4)	×	×	O	×	O	×	×	O
(5)	×	O	×	O	O	×	×	O
(6)	×	×	×	×	O	×	O	O
(7)	×	×	O	×	×	×	×	O
(8)	×	×	×	O	×	×	×	O
(9)	×	×	×	×	×	×	O	O
(10)	×	×	×	×	×	×	O	O
(11)	O	×	×	×	O	O	×	O
(12)	O	×	×	×	O	O	×	O
(13)	×	×	×	×	×	O	O	O
(14)	×	×	×	×	×	O	O	O
Degree of freedom	2	2	3	3	6	4	4	10

Table 1

(1) Projection property	$\left\|G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(x,y)\right\|^2 = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_g\left(\vec{\mathbf{v}}\cdot\begin{bmatrix}\mathbf{D} & -\mathbf{C}\\ -\mathbf{B} & \mathbf{A}\end{bmatrix}\right) dfdh\,, \qquad \vec{\mathbf{v}} = [x, y, f, h].$
(2) Recovery property	$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(x,y)$ $= \left\{\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_g\left(\vec{\mathbf{v}}\cdot\begin{bmatrix}\mathbf{D}/2 & -\mathbf{C}/2\\ -\mathbf{B} & \mathbf{A}\end{bmatrix}\right)\cdot e^{j\cdot(f\cdot x+h\cdot y)}\cdot dfdh\right\} / \overline{G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(0,0)}$
(3) Relations with generalized fractional convolution	If $z(x, y)$ is the 2-D affine generalized fractional convolution of $f(x, y)$ and $g(x, y)$, as Eq. (79), then $W_z(\vec{\mathbf{x}},\vec{\mathbf{w}}) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} W_f\left(\vec{\mathbf{x}}\mathbf{A}^{\mathrm{T}}\mathbf{D} + \vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}\mathbf{D} - \vec{\rho}\mathbf{B}, -\vec{\mathbf{x}}\mathbf{A}^{\mathrm{T}}\mathbf{C} - \vec{\mathbf{w}}\mathbf{B}^{\mathrm{T}}\mathbf{C} + \vec{\rho}\mathbf{A}\right)\cdot W_g\left(\vec{\mathbf{x}} + \vec{\rho}\mathbf{B}, \vec{\mathbf{w}} - \vec{\rho}\mathbf{A}\right)\cdot d\tau d\sigma$ where $\vec{\mathbf{x}} = [x, y]$, $\vec{\mathbf{w}} = [f, h]$, $\vec{\rho} = [\tau, \sigma]$.

Table 2

Property	Input	Transform results
(1) Space shifting	$g(\vec{\mathbf{x}}-\vec{\mathbf{x}}_0)$, where $\vec{\mathbf{x}}_0=(x_0,y_0)$	$\exp\left(j(\varphi+\vec{\mathbf{w}}\mathbf{C}\vec{\mathbf{x}}_0^{\mathrm{T}})\right)\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(\vec{\mathbf{w}}-\vec{\mathbf{x}}_0\mathbf{A}^{\mathrm{T}})$, where $\varphi=-\vec{\mathbf{x}}_0\mathbf{C}^{\mathrm{T}}\mathbf{A}\vec{\mathbf{x}}_0^{\mathrm{T}}/2$.
(2) Modulation	$e^{j\cdot\vec{\mathbf{w}}_0\vec{\mathbf{x}}^{\mathrm{T}}}g(\vec{\mathbf{x}})$, where $\vec{\mathbf{w}}_0=(f_0,h_0)$.	$\exp\left(j(\phi+\vec{\mathbf{w}}\mathbf{D}\vec{\mathbf{w}}_0^{\mathrm{T}})\right)G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(\vec{\mathbf{w}}-\vec{\mathbf{w}}_0\mathbf{B}^{\mathrm{T}})$, where $\phi=-\vec{\mathbf{w}}_0\mathbf{B}^{\mathrm{T}}\mathbf{D}\vec{\mathbf{w}}_0^{\mathrm{T}}/2$.
(3) Multiplied by x	$x\cdot g(x,y)$	$jb_{11}\frac{\partial}{\partial f}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)+jb_{21}\frac{\partial}{\partial h}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)$ $+d_{11}f\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)+d_{21}h\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)$
(4) Multiplied by y	$y\cdot g(x,y)$	$jb_{12}\frac{\partial}{\partial f}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)+jb_{22}\frac{\partial}{\partial h}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)$ $+d_{12}f\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)+d_{22}h\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)$
(5) Differentiation for x-axis	$\frac{\partial}{\partial x}g(x,y)$	$a_{11}\frac{\partial}{\partial f}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)+a_{21}\frac{\partial}{\partial h}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)$ $-jc_{11}f\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)-jc_{21}h\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)$
(6) Differentiation for y-axis	$\frac{\partial}{\partial y}g(x,y)$	$a_{12}\frac{\partial}{\partial f}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)+a_{22}\frac{\partial}{\partial h}G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)$ $-jc_{12}f\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)-jc_{22}h\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)$

Table 3

(1) Orthogonality property	$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} K_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,x,y)\cdot K^{*}_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f',h',x,y)dxdy$ $=\delta(f-f',h-h')$
(2) Energy conservation property	$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\lvert g(x,y)\rvert^{2}dxdy=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\lvert G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)\rvert^{2}dfdh$ Also called as the Parseval's theorem
(3) Generalized energy conservation property	$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} g(x,y)\cdot s^{*}(x,y)\cdot dxdy$ $=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)\cdot S^{*}_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)\cdot dfdh$ Also called as the generalized Parseval's theorem
(4) Conjugation prop.	$K^{*}_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,x,y)=K_{(\mathbf{A},-\mathbf{B},-\mathbf{C},\mathbf{D})}(f,h,x,y)$
(5) Symmetry property of Kernel	$K_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,x,y)=K_{(\mathbf{D}^{\mathbf{T}},\mathbf{B}^{\mathbf{T}},\mathbf{C}^{\mathbf{T}},\mathbf{A}^{\mathbf{T}})}(x,y,f,h)$
(6) Another form of the inverse 2-D AGFFT	$g(x,y)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} K^{*}_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h,x,y)\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,h)\cdot dfdh$

Table 4

Property	Input	Transform Result
(1) Space reversion	$g(-x,-y)$	$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(-f,-h)$
(2) Mirror property	$g(-x,y)$, $g(x,-y)$	$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(-f,h)$, $G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(f,-h)$ when $a_{12}=a_{21}=b_{12}=b_{21}=c_{12}=c_{21}=d_{12}=d_{21}=0$, i.e., for the case of the separable 2-D canonical transform.
(3) Twisting preservation	$g(px+qy,-qx+py)$, where $p^2+q^2=1$.	$G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}(pf+qh,-qf+ph)$ when $a_{12}=a_{21}=b_{12}=b_{21}=c_{12}=c_{21}=d_{12}=d_{21}=0$.
(4) Division by x	$g(x,y)/x$	$-j\exp\left(j\vec{\mathbf{w}}\cdot\mathbf{D}\mathbf{B}^{-1}\cdot\vec{\mathbf{w}}^{\mathbf{T}}/2\right)\cdot\int_{-\infty}^{\infty}\exp\left(-j\vec{\mathbf{s}}\cdot\mathbf{B}^{\mathbf{T}}\mathbf{D}\cdot\vec{\mathbf{s}}^{\mathbf{T}}/2\right)\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}\left(\vec{\mathbf{s}}\mathbf{B}^{\mathbf{T}}\right)\cdot dp$, where $\vec{\mathbf{w}}=[f,h]$, $\vec{\mathbf{s}}=[p,q]$, $z=(f\cdot b_{22}-h\cdot b_{12})/\det(\mathbf{B})$, and $q=(-f\cdot b_{21}+h\cdot b_{11})/\det(\mathbf{B})$.
(5) Division by y	$g(x,y)/y$	$-j\exp\left(j\vec{\mathbf{w}}\cdot\mathbf{D}\mathbf{B}^{-1}\cdot\vec{\mathbf{w}}^{\mathbf{T}}/2\right)\cdot\int_{-\infty}^{\infty}\exp\left(-j\vec{\mathbf{s}}\cdot\mathbf{B}^{\mathbf{T}}\mathbf{D}\cdot\vec{\mathbf{s}}^{\mathbf{T}}/2\right)\cdot G_{(\mathbf{A},\mathbf{B},\mathbf{C},\mathbf{D})}\left(\vec{\mathbf{s}}\mathbf{B}^{\mathbf{T}}\right)\cdot dq$, where $\vec{\mathbf{w}}=[f,h]$, $\vec{\mathbf{s}}=[p,q]$, $z=(-f\cdot b_{21}+h\cdot b_{11})/\det(\mathbf{B})$, and $p=(f\cdot b_{22}-h\cdot b_{12})/\det(\mathbf{B})$.

Table 5

Pattern Recognition Using Discrete Fractional Fourier Transform and Fractional Correlation

Soo-Chang Pei and Chi-chin Lien
Department of Electrical Engineering, National Taiwan University
1 Roosevelt Road, Sec.4, Taipei 106, Taiwan, ROC
E-mail: pei@cc.ee.ntu.edu.tw

ABSTRACT

In this paper, we apply discrete fractional Fourier transform to pattern recognition. The objective of pattern recognition is to determine presence or absence of specified objects. By means of discrete fractional Fourier transform and fractional correlation, we can not only control the detecting scope but also recognize the objects with different scales.

1. Introduction

In recent years, the fractional Fourier transform (FrFT) draws a lot of attention. Researchers develop many applications in optics and signal processing. In [1], Pei develops the discrete version of FrFT (DFrFT). The full consistency between DFrFT and FrFT let us be able to implement application by digital computation. In this paper, we apply DFrFT to pattern recognition.

Pattern recognition is concerned with the determination of the presence or absence of objects in an image. The most fundamental approach of pattern recognition is correlation, in which we translate and compare the template pattern to all unknown patterns in the image. In [4], the authors use optical system to implement FrFT and apply it to pattern recognition. However, optical system is not flexible and reliable as digital system. Therefore, we use DFrFT to implement the application.

This paper is organized as follows. In Section 2, we briefly introduce the definitions of FrFT and DFrFT. We also give a simple introduction to pattern recognition. In section 3, We describe our approach of recognition, which is derived from correlation. This method shows some difference from the original correlation results. In Section 4, we present the experiment results, which suggest that pattern recognition by DFrFT shows the shift-variance property and scalable detection. We can control the detection scope by means of the order of DFrFT. In the other experiment, we use DFrFT to recognize the objects with different scales. We can use it to determine scale of objects even in fractional scale ratio. Finally, we give a conclusion in section 5.

2. Preliminary

2.1. Fractional Fourier Transform

The fractional Fourier transform (FrFT) of a signal x(t) is defined as[3]:

$$X_\phi(u) = FrFT^\phi\{x(t)\} = \sqrt{\frac{1-j\cot\phi}{2\pi}}\, e^{j\frac{u^2}{2}\cot\phi}\int_{-\infty}^{\infty} x(t)e^{j\frac{t^2}{2}\cot\phi - ju\csc\phi}dt \quad (1)$$

Where φ is the fractional angle of FrFT. When φ=0, FrFT is an identity transformation. When $\varphi=\pi$ /2, FrFT becomes Fourier transform. FrFT satisfies the additive property:

$$FrFT^\beta\{FrFT^\alpha\{x(t)\}\} = FrFT^{\beta+\alpha}\{x(t)\} \quad (2)$$

2.2. Discrete Fractional Fourier Transform

Discrete Fractional Fourier Transform (DFrFT) is the discrete version of FrFT. In [1], DFrFT is defined as fractionalization of discrete Fourier transform (DFT). N-point discrete Fourier transform can be written as:

$$\mathbf{X}_F = \mathbf{F}_N\mathbf{x} = \frac{1}{\sqrt{N}}\sum_{n=0}^{N-1} x(n)e^{-j2\pi\frac{nk}{N}} \quad (3)$$

Where $\mathbf{x}$ and $\mathbf{X}_F$ are both N-tuple vectors and $\mathbf{F}_N$ is N-point DFT.

$$\mathbf{F}_N = \frac{1}{\sqrt{N}}\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & e^{-j\frac{2\pi}{N}1} & \cdots & e^{-j\frac{2\pi}{N}(N-1)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & e^{-j\frac{2\pi}{N}(N-1)} & \cdots & e^{-j\frac{2\pi}{N}(N-1)(N-1)} \end{bmatrix} \quad (4)$$

We can diagonalize the DFT matrix as:

$$\mathbf{F}_N = \hat{\mathbf{U}}\mathbf{D}\hat{\mathbf{U}}^T \quad (5)$$

Thus the power matrix of F is:

$$\mathbf{F}_N^a = \hat{\mathbf{U}}\mathbf{D}^a\hat{\mathbf{U}}^T \quad (6)$$

Then the DFrFT is defined as

$$\mathbf{F}_N^a x = (\hat{\mathbf{U}}\mathbf{D}^a\hat{\mathbf{U}}^T)x \quad (7)$$

2.3. Pattern Recognition by Correlation

We can detect a pattern p(x, y) in an image i(x, y) by following computation:

$$d(x,y) = \sum\sum i(u,v)p^*(u-x, v-y) \quad (8)$$

It means the pattern p(x, y) is shifted and compared to the image. After some computations, correlation can be calculated by

$$d(x,y) = IFT\{FT\{i(x,y)\}FT^*\{p(x,y)\}\} \quad (9)$$

Where FT is Fourier transform and IFT is inverse Fourier transform. '*' denotes conjugate.

3. Applying DFrFT to Pattern Recognition

If we replace FT and IFT with FrFT in (9), it becomes:

$$d(x,y) = FrFT_\gamma\{FrFT_\alpha\{i(x,y)\}FrFT_\beta^*\{p(x,y)\}\} \quad (10)$$

Where α, β, and γ are fractional angles of

FrFT and can be chosen for specified applications. We call it "fractional correlation". Figure 1 schematically illustrates (10).

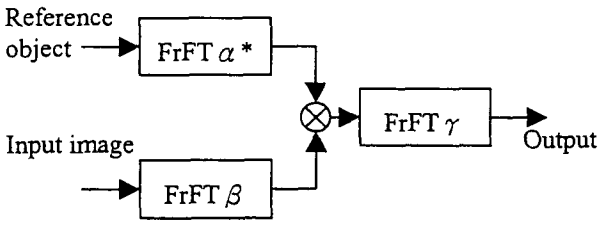


Figure 1 Fractional correlation.

4. Experimental Results

In this section, we give the experiments results by DFrFT. We divide the results into two parts. We discuss the effect of object shifting in section 4.1 and the effect of object scaling in section 4.2.

4.1. Effect of object shift-variant correlation

In this experiment, we use a 1-D rectangular window as the reference object to be recognized. The rectangular window with unity height and two units width locates at the center of the space. The parameters of fractional correlation are: $\alpha=\beta$ and $\gamma=-\pi/2$. We experiment the fractional correlation at three fractional angles, $\pi/4$, $3\pi/8$, and $\pi/2$. There are four input images for each fractional angle. One is the same as the reference object. The other three images are its shifted versions. In our experiment, we shift the image 1, 2, and 3 units respectively.

Figure 2 shows the experiment results at fractional angle $\pi/2$. In fact, when the fractional angle equals $\pi/2$, the fractional correlation is the same as the original correlation. Thus the output image is a shifted peak where the position of the correlation peak depends on the position of the input image. The peak value of the correlation does not change although the input image is shifted. We call the result is "shift invariance".

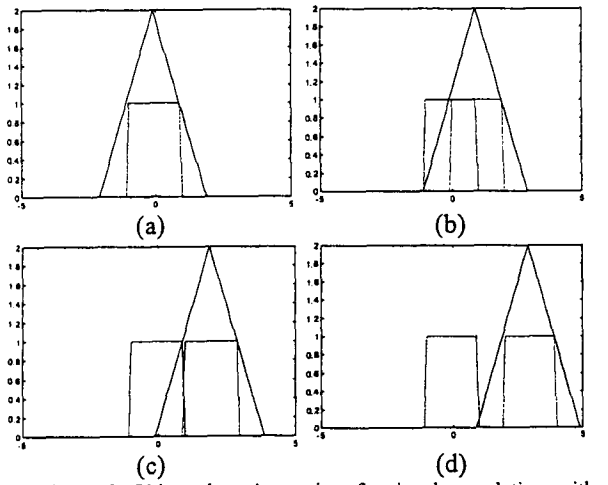


Figure 2 Object detection using fractional correlation with fractional angle $\pi/2$. The results are the same as FFT correlation. Dot-dash line: reference object, dash line: input image, solid line: correlation output. (a) The input image locates at the correct position. (b) The input image shifts 1 unit. (c) The input image shifts 2 units. (d) The input image shifts 3 units.

Figure 3 shows the experiment results when fractional angle is $\pi/4$. We see the results are quite different from figure 2. When the input image is shifted from the correct position, the correlation peak decays. Further the object is shifted, smaller the correlation peak becomes. In figure 3 (d), the reference object and the input image are not overlapped and the peak value remains only 10% of figure 3 (a)

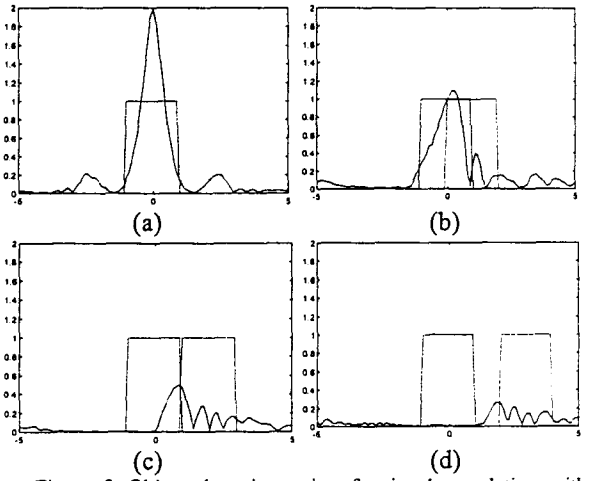


Figure 3 Object detection using fractional correlation with fractional angle $\pi/4$. Dot-dash line: reference object, dash line: input image, solid line: correlation output. (a) The input image locates at the correct position. (b) The input image shifts 1 unit. (c) The input image shifts 2 units. (d) The input image shifts 3 units.

Figure 4 show the experiment results at fractional angle $3\pi/8$. The correlation peaks are again decays as the input images are shifted away from the correct position. However, the situation have a little difference from the results at fractional angle $\pi/4$. In figure 4 (d), the peak value remains about 20% of figure 4 (a).

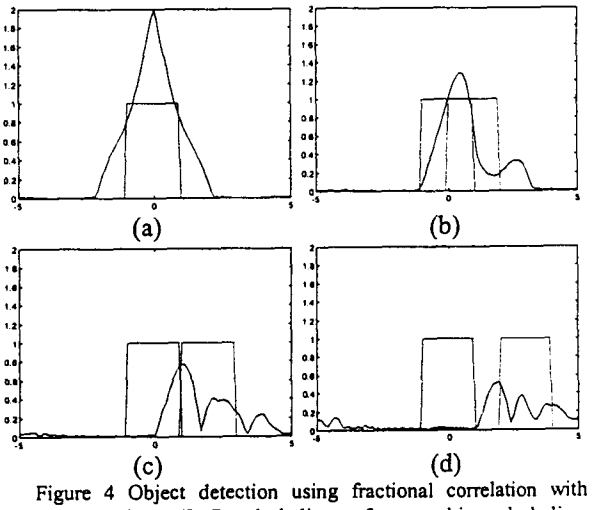


Figure 4 Object detection using fractional correlation with fractional angle $3\pi/8$. Dot-dash line: reference object, dash line: input image, solid line: correlation output. (a) The input image locates at the correct position. (b) The input image shifts 1 unit. (c) The input image shifts 2 units. (d) The input image shifts 3 units.

From above results, the correlation values reduce when the fractional angles decrease from $\pi/2$. From these results, the fractional correlation provides "shift-variance" correlation when fractional angle is not $\pi/2$. And we can control the "scale of shift-variance" by fractional angle. When the angle approaches $\pi/2$, the correlation becomes more shift-invariant. In contrast, When the angle approaches zero, the correlation becomes more shift-variant

4.2. Determine the scales of objects

In this section, we use DFrFT to detect objects with different scales. This technique is derived form the scaling property of FrFT, which is given in (11).

$$F_\phi(x(ct)) = \sqrt{\frac{1-j\cot\phi}{c^2-j\cot\phi}}\, e^{j\frac{u^2}{2}\cot\phi(1-\frac{\cos^2\phi}{\cos^2\beta})} X_\beta(u\frac{\sin\beta}{c\sin\phi}) \quad (11)$$

$$\tan\beta = c^2\tan\phi \quad (12)$$

This property means that the φ-angle FrFT of a scaled function x(ct) with scaling ratio c is related with β-angle FrFT of the original function x(t). They are different in a constant complex factor, multiplying a chirp function, and scaling in frequency axis.

For proper use in discrete fractional Fourier transform, we set

$$\frac{\sin\beta}{c\sin\phi} = 1 \quad (13)$$

Solving (12) and (13), we get angles φ and β:

$$\tan\phi = \frac{1}{c} \quad and \quad \tan\beta = c \quad (14)$$

If we only consider the magnitude of (11), it becomes

$$|F_\phi(x(ct))| = A|X_\beta(u)| \quad (15)$$

The amplitude of the φ-angle FrFT of a scaled function x(ct) is in proportion to the amplitude of β-angle FrFT of the original function x(t). Thus, two functions with different scales have the same magnitude distribution in two fractional domains.

Figure 6 illustrates the relation in (15). Figure 6 (b) and (d) show the DFrFT of functions in (a) and (c) respectively and they are approximately the same when the fractional angles satisfy (14).

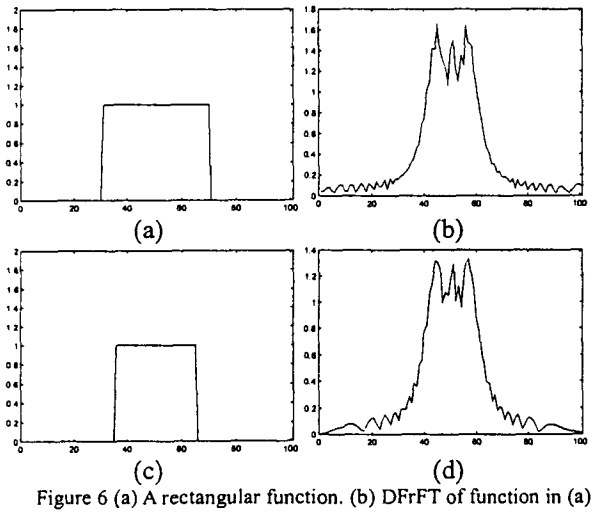


Figure 6 (a) A rectangular function. (b) DFrFT of function in (a) with fractional angle $\tan^{-1}(4/3)$ (c) Another scaled rectangular function with 3/4 width of (a). (d) DFrFT of function in (c) with fractional angle $\tan^{-1}(3/4)$

Figure 7 shows the block diagram of pattern recognition with scale determination. The reference object is first taken DFrFT with angle $\tan^{-1}(c)$, then taken magnitude as the input of the FFT-based correlation. The input image is first taken FrFT with angle $\tan^{-1}(1/c)$, then taken magnitude as the other input of the FFT-based correlation. The output of the FFT-based correlation shows the detection result of the system. The parameter c determines what scale of the reference object to be recognized. If we sweep the parameter c in some extent, we can determine the scale of .the input image.

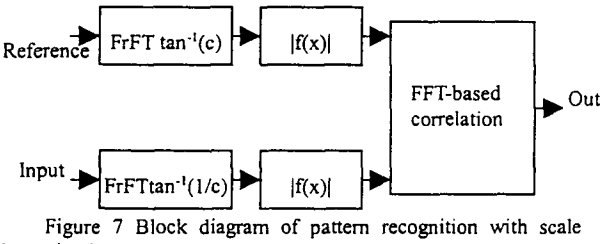


Figure 7 Block diagram of pattern recognition with scale determination

Figures 8 and 9 show 1-D examples of scale determination using system in figure 7. Figure 8(a) is the reference object, (b) and (c) are the input images with scale ratio 2 and $\sqrt{2}$ respectively. In figure 9, we sweep the parameter c of the system in figure 7 and get the profile of correlation peak value. Figure 9 (a) has maximum correlation peak value at c=2. It means that the input image is 1/2 scaling version of the reference object. Figure 9 (b) detects the maximum value at c=1.41. Both examples correctly determine the scales.

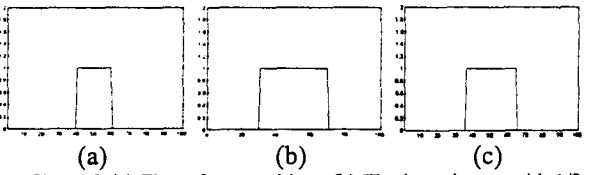


Figure 8 (a) The reference object. (b) The input image with 1/2 window width (scaling factor c=2). (c) Input image with $1/\sqrt{2}$ window width (scaling factor c=$\sqrt{2}$)

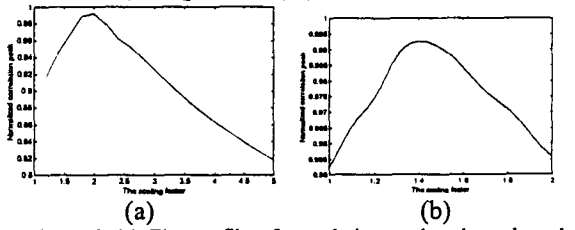


Figure 9 (a) The profile of correlation peak value when the scaling parameter sweeps from 1 to 5. The maximum value occurs at c=2. (b) The correlation peak when the scaling parameter sweeps from 1 to 2. The maximum value occurs at c=1.41.

Figures 10-13 show 2-D examples of scale determination. We use the letter 'S' (Figure 10(a)) as the reference pattern. In figures 10(b), (c), and (d), the input images are scaled versions of (a). Because now

we process 2-D objects, the system in figure 7 has two scale factors in two directions. We use 'cx' and 'cy' to represent the scale factors in horizontal and vertical directions respectively.

In figure 11, we use figure 10(a) to detect figure 10(b) and sweep both cx and cy from 0.5 to 3. We get the maximum correlation peak value=0.93 when cx=1, cy=2. In figure 12, we use figure 10(a) to detect figure 10(c). We get the maximum correlation peak value=0.937 when cx=2, cy=2. The results agree with the scale ratios of input images.

In figure 13, we use figure 10(a) to detect figure 10(d), which has the same scale with figure 10(c) but is shifted. Because shifting introduce only phase change in FrFT, the scale determination is not affected. The maximum correlation peak value 0.92 occurs at cx=2, cy=2, which agrees with what we expect. These examples illustrate the feasibility of scaled pattern recognition by DFrFT.

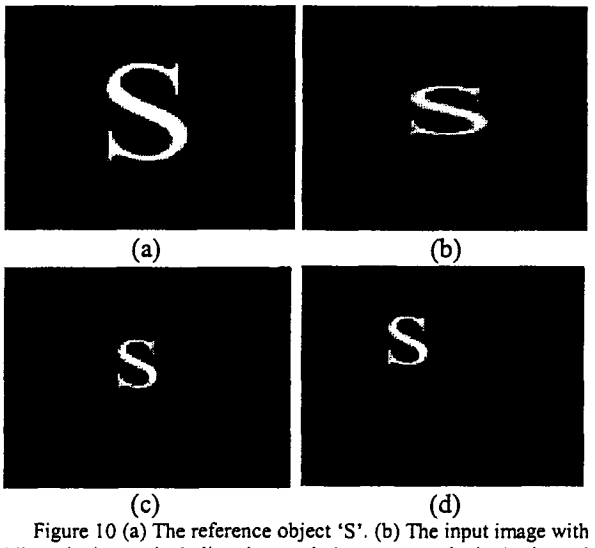


Figure 10 (a) The reference object 'S'. (b) The input image with 1/2 scale in vertical direction and the same scale in horizontal direction (scaling factor cx=1, cy=2). (c) The input image with 1/2 scale in both directions (scaling factor cx=2, cy=2). (d) shifted version of (c)

5. Conclusion

In this paper, we apply discrete fractional Fourier transform to pattern recognition. Our approach is extended from conventional FT-based correlation. We discuss the effect of object shifting and scaling. The experiment results show that pattern recognition by DFrFT has shift-variance property. We can control the detection scope by fractional angle of FrFT. In another experiment of scaled pattern recognition, We can exactly determine the object scale by DFrFT.

References

[1] S.-C Pei, C,-C Tseng, M,-H Yeh, and J,-J Shyu, "Discrete fractional Hartley and Fourier transforms" IEEE Transactions on Circuits and Systems-II: analog and digital signal processing, vol. 45, no. 6, pp.665-675, June 1998.

[2] V. Namias, "The Fractional Order Fourier Transform and Its Application to Quantum Mechanics," J. Inst. Math. Appl., vol. 25, pp. 241-265, 1980

[3] Sergio Granieri, Ricardo Arizaga, and Enrique E. Sicre, "Optical correlation based on the fractional Fourier transform," Applied Optics, Vol. 36, No. 26, Sep. 1997, pp. 6636.

[4] Z.Zalevsky, D. Mendlovic, and J.H. Caulfield, " Fractional correlator with real-time control of the space- invariant property," Applied Pptics, Vol. 36, No. 11, April 1997, pp. 2370.

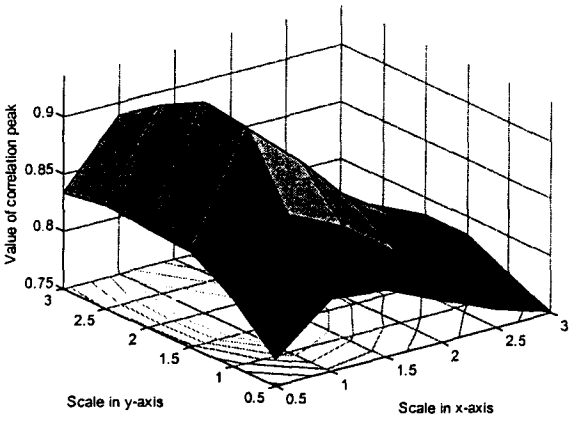


Figure 11 The value of correlation peaks when the scaling parameters sweep from 0.5 to 3. The maximum value 0.93 occurs at cx=1 and cy=2.

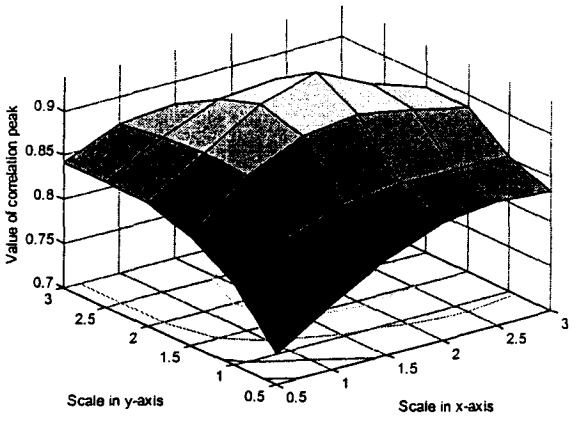


Figure 12 The value of correlation peaks when the scaling parameters sweep from 0.5 to 3. The maximum value 0.937 occurs at cx=2 and cy=2.

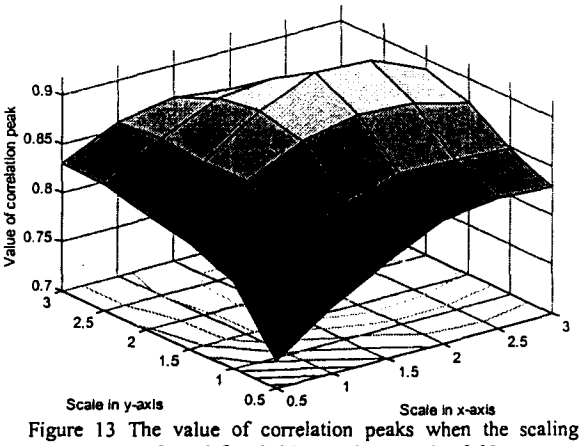


Figure 13 The value of correlation peaks when the scaling parameters sweep from 0.5 to 3. The maximum value 0.92 occurs at cx=2 and cy=2.

SP EDICS : SP 2.3.1 Time-Frequency Signal Analysis

Fractional Time-Frequency Distributions

Soo-Chang Pei* and Min-Hung Yeh

Department of Electrical Engineering
National Taiwan University
Taipei, Taiwan, R. O. C.

Abstract

Several quadratic signal representation methods have been successfully used in non-stationary signal processing. Four quadratic signal representations, Wigner distribution, Ambiguity function, signal correlation function and spectral correlation function, are the traditional ones. In this paper, we propose generalized distributions of the four elementary quadratic signal representation. The proposed distributions can have fractional time/frequency-lag and fractional time-lag/frequency characteristics of signals, and it relates the four quadratic signal representations with a more general relation.

*Author for correspondence

I. Introduction

Several quadratic signal representation methods are successfully used in nonstationary signal processing[1][2][3]. They are four elementary quadratic signal representations, and they are Wigner distribution, Ambiguity function, signal correlation function and spectral correlation function are the traditional ones. The Wigner distribution has served a useful analysis in fields as diverse as quantum mechanics, optics, acoustics, image processing and oceanography[1][2][4]. The Ambiguity function have been used extensively in the field of radar, sonar, audio, astronomy, communications and optics[5].

The generalization of Fourier transform, Fractional Fourier transform(FRFT), have been developed[6][7]. Fractional Fourier transform indicates a rotation of signal in the time-frequency plane and it becomes a very useful tool in signal analysis. The FRFT operation can have fractional time and frequency characteristics of signals.

In this paper, we develop the generalization of quadratic time-frequency distributions. They are called fractional time-frequency distributions. The fractional time-frequency distributions can have fractional time/frequency-lag and time-lag/frequency characteristics of signals. And it can related the four traditional quadratic signal representations.

This paper is organized as follows: In section II, we review the previous results about the time-frequency analysis and fractional Fourier transform. In section III, the fractional time-frequency distribution is developed. In section IV, the properties of the fractional time-frequency distribution are discussed and the fractional time-frequency distributions of several special signals are computed. An application of fractional time-frequency distribution is presented in section V. And the final conclusion are drawn in section VI.

II. Preliminary

A. Quadratic Signal Representations

Several quadratic signal representation methods are successfully used in nonstationary signal processing[2]. The Wigner distribution is the most important one, and it has served a useful analysis in fields as diverse as quantum mechanics, optics, acoustics, image processing and oceanography. The Ambiguity function have been used extensively in the field of radar and sonar, and its properties are very well understood[5]. A prominent relation of the Wigner distribution and the Ambiguity function are the Fourier transform pairs[2]. Besides Wigner distribution and ambiguity function, two correlation functions are related to the Wigner distribution and Ambiguity function.

- **Signal Correlation Function**

$$C_x(t,\tau) = x(t+\frac{\tau}{2})x^*(t-\frac{\tau}{2}) \tag{1}$$

- **Spectrum Correlation Function**

$$S_x(\eta,\omega) = \mathcal{X}(\omega+\frac{\eta}{2})\mathcal{X}^*(\omega-\frac{\eta}{2}) \tag{2}$$

- **Wigner Distribution**

$$\begin{aligned} W_x(t,\omega) &= \int_{-\infty}^{\infty} x(t+\frac{\tau}{2})x^*(t-\frac{\tau}{2})e^{-j\omega\tau}d\tau & (3)\\ &= \int_{-\infty}^{\infty} \mathcal{X}(w+\frac{\eta}{2})\mathcal{X}^*(w-\frac{\eta}{2})e^{-jt\eta}d\eta & (4) \end{aligned}$$

- **Ambiguity Function**

$$\begin{aligned} A_x(\eta,\tau) &= \int_{-\infty}^{\infty} x(t+\frac{\tau}{2})x^*(t-\frac{\tau}{2})e^{-jt\eta}dt & (5)\\ &= \int_{-\infty}^{\infty} \mathcal{X}(\omega+\frac{\eta}{2})\mathcal{X}^*(\omega-\frac{\eta}{2})e^{-j\omega\tau}d\omega & (6) \end{aligned}$$

where t is used to indicate the time variable, τ is the time-lag variable, ω is the frequency variable, and η is the frequency-lag variable. The relationships of the above four quadratic signal representation methods are presented in [2]. Figure 1 quotes that the quadratic signal representations $C_x(t,\tau)$, $S_x(\eta,\omega)$, $W_x(t,\omega)$ and $A_x(\eta,\tau)$ are interpreted by Fourier transforms.

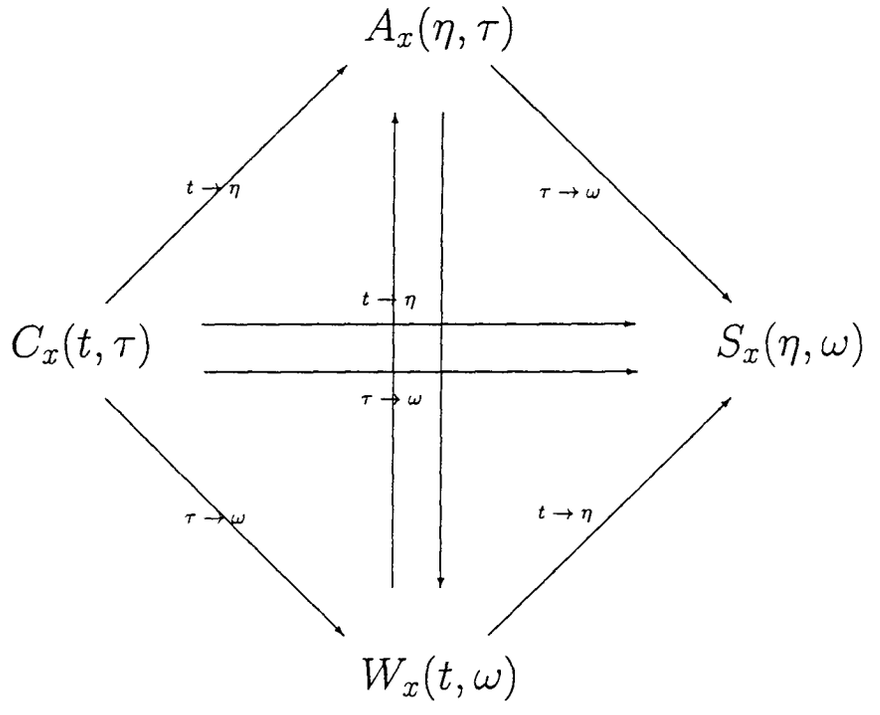


Figure 1: The quadratic signal representations represented by Fourier transforms

B. Fractional Fourier Transform

Successive applications of the forward Fourier transform $\mathcal{F}$ on the signal $x(t)$ several times, and we will get:

$$\mathcal{F}^2[x(t)] = x(-t), \qquad \mathcal{F}^3[x(t)] = \mathcal{X}(-w), \qquad \mathcal{F}^4[x(t)] = x(t) \tag{7}$$

Based upon the above notation, the Fourier transform of signal can be interpreted as a $\frac{\pi}{2}$ angle rotation of signal in the time-frequency plane. A generalization of Fourier transform, FRFT, is developed and treated as a rotation of signal to any angles in the time-frequency plane. The

transform kernel of continuous fractional Fourier transform (FRFT) is defined as follows [6][7] [8][9]:

$$K_\alpha(t,u) = \begin{cases} \sqrt{\frac{1-j\cot\alpha}{2\pi}} e^{j\frac{t^2+u^2}{2}\cot\alpha - jut\csc\alpha} & \text{if } \alpha \text{ is not a multiple of } \pi \\ \delta(t-u) & \text{if } \alpha \text{ is a multiple of } 2\pi \\ \delta(t+u) & \text{if } \alpha+\pi \text{ is a multiple of } 2\pi \end{cases} \tag{8}$$

where α indicates the rotation angle of transformed signal for FRFT. Using the kernel of FRFT, the FRFT of the signal $x(t)$ by angle α is computed as:

$$\mathcal{X}_\alpha(u) = \int_{-\infty}^{\infty} x(t) K_\alpha(t,u) dt \tag{9}$$

The signal $x(t)$ can be recovered by an FRFT operation with backward angle $(-\alpha)$:

$$x(t) = \int_{-\infty}^{\infty} \mathcal{X}(u) K_{-\alpha}(u,t) du \tag{10}$$

Some interesting result of FRFT are presented in [8], and we quote them here for our further discussion.

	Signal	fractional Fourier transform with angle α
1	$x(t-\tau)$	$\mathcal{X}_\alpha(u-\tau\cos\alpha)e^{j\frac{\tau^2}{2}\sin\alpha\cos\alpha - ju\tau\sin\alpha}$
2	$x(t)e^{jvt}$	$\mathcal{X}_\alpha(u-v\sin\alpha)e^{-j\frac{v^2}{2}\sin\alpha\cos\alpha + juv\cos\alpha}$
3	$x'(t)$	$\mathcal{X}'_\alpha(u)\cos\alpha + ju\mathcal{X}_\alpha(u)\sin\alpha$
4	$\int_a^t x(t')dt'$	$\sec\alpha e^{-j\frac{u^2}{2}\tan\alpha}\int_a^u \mathcal{X}_\alpha(z)e^{j\frac{z^2}{2}\tan\alpha}dz$
5	$tx(t)$	$u\mathcal{X}_\alpha(u)\cos\alpha + j\mathcal{X}'_\alpha(u)\sin\alpha$
6	$x(t)/t$	$-j\sec\alpha e^{j\frac{u^2}{2}\cot\alpha}\int_{-\infty}^u x(z)e^{-j\frac{z^2}{2}\cot\alpha}dz$
7	$x(-t)$	$\mathcal{X}_\alpha(-u)$
8	$x(ct)$	$\sqrt{\frac{1-j\cot\alpha}{c^2-j\cot\alpha}}e^{j\frac{u^2}{2}\cot\alpha(1-\frac{\cos^2\beta}{\cos^2\alpha})}\mathcal{X}_\beta(u\frac{\sin\beta}{c\sin\alpha})$ where $\beta=\tan^{-1}(c^2\tan\alpha)$

Table 1: Properties of the fractional Fourier transform

In [10], the 2D FRFT transform kernel with various orders in the two dimensions is defined as follows:

$$K_{\alpha,\beta}(s,t,u,v) = \frac{1}{2\pi}\sqrt{1-j\cot\alpha}\sqrt{1-j\cot\beta}e^{j\frac{s^2+u^2}{2}\cot\alpha - jsu\csc\alpha}e^{j\frac{t^2+v^2}{2}\cot\beta - jtv\csc\beta} \tag{11}$$

where α and β indicate the rotation angles of the transformed signal for 2D FRFT. Using this 2D FRFT kernel, the 2D FRFT of the signal $x(s,t)$ by angle parameter (α,β) is computed as:

$$\mathcal{X}_{\alpha,\beta}(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x(s,t)K_{\alpha,\beta}(s,t,u,v)dsdt \tag{12}$$

The signal $x(s,t)$ can be recovered by an FRFT operation with backward angle $(-\alpha,-\beta)$:

$$x(s,t) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \mathcal{X}_{\alpha,\beta}(u,v,s,t)K_{-\alpha,-\beta}(u,v,s,t)dudv \tag{13}$$

III. Development of Fractional Time-Frequency Distributions

If the Figure 1 is rotated by 45 degrees counterclockwise, a square can be obtained. Then we can find that there always exist Fourier transform relation from τ to ω in the horizontal direction and another Fourier transform relation from t to η in the vertical direction. Moreover, we can generalize the Fourier transform with the fractional Fourier transform. Thus Figure 2 will be obtained. All the points located in the square of Figure 2 can define a time-frequency distributions. These distributions can have fractional time and frequency-lag, fractional time-lag and frequency characteristics. They are called **fractional time-frequency distribution**. The fractional time-frequency distributions with parameter α_1, α_2 can be defined as:

$$\begin{aligned}\Gamma_{x,\alpha_1,\alpha_2}(\nu,\mu) &= \int\int C_x(t,\tau)K_{\alpha_1,\alpha_2}(t,\tau,\nu,\mu)dtd\tau \qquad (14)\\ &= \int\int C_x(t,\tau)\frac{1}{2\pi}\sqrt{1-j\cot\alpha_1}\sqrt{1-j\cot\alpha_2}e^{j\frac{\tau^2+\mu^2}{2}\cot\alpha_2 - j\tau\mu\csc\alpha_2}e^{j\frac{t^2+\nu^2}{2}\cot\alpha_1 - jt\nu\csc\alpha_1}dtd\tau \qquad (15)\end{aligned}$$

The traditional four quadratic signal representation are special cases of fraction time-frequency distribution, and they are located in the corners of the square. These four special cases of fractional time-frequency distributions are defined as:

$$\begin{aligned}\Gamma_{x,0,0}(\nu,\mu) &= C_x(\nu,\mu)\\ \Gamma_{x,0,\frac{\pi}{2}}(\nu,\mu) &= W_x(\nu,\mu)\\ \Gamma_{x,\frac{\pi}{2},0}(\nu,\mu) &= A_x(\nu,\mu)\\ \Gamma_{x,\frac{\pi}{2},\frac{\pi}{2}}(\nu,\mu) &= S_x(\nu,\mu)\end{aligned}$$

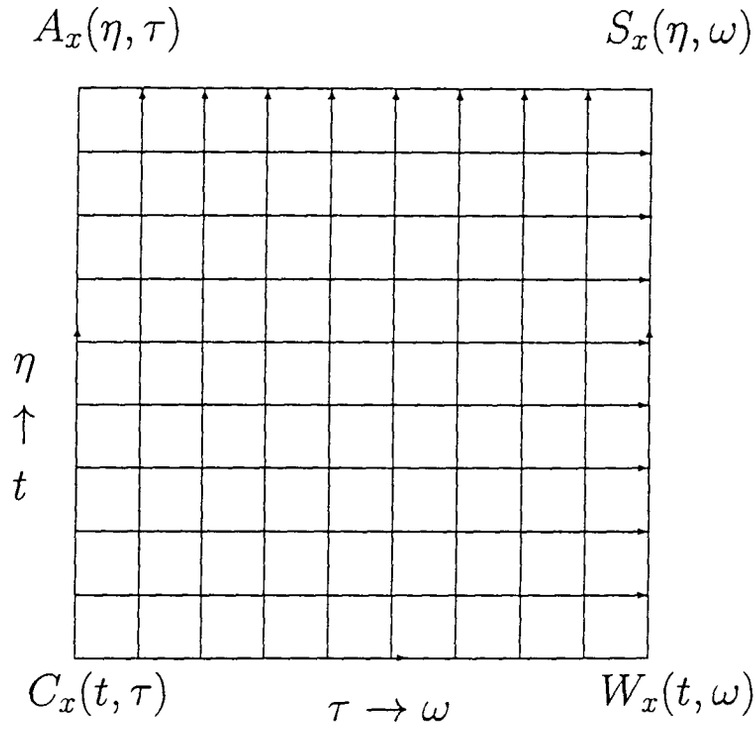


Figure 2: Quadratic signal representations and Fractional Time-Frequency distributions

In [11], a definition of fractional Wigner distribution has been defined. The fractional Wigner distribution in [11] is a rotation of Wigner distribution in the time-frequency plane. The fractional time-frequency distribution defined in this paper indicates fractional time/frequency-lag and frequency/time-lag characteristics of signals. Figure 3 shows the different concepts between our fractional time-frequency distribution and fractional Wigner distribution in 3.

IV. Properties of Fractional Time-Frequency Distributions

In this section, we will discuss the main properties of the fractional time-frequency distributions. Moreover, we will compute the fractional time-frequency distributions of several special signals.

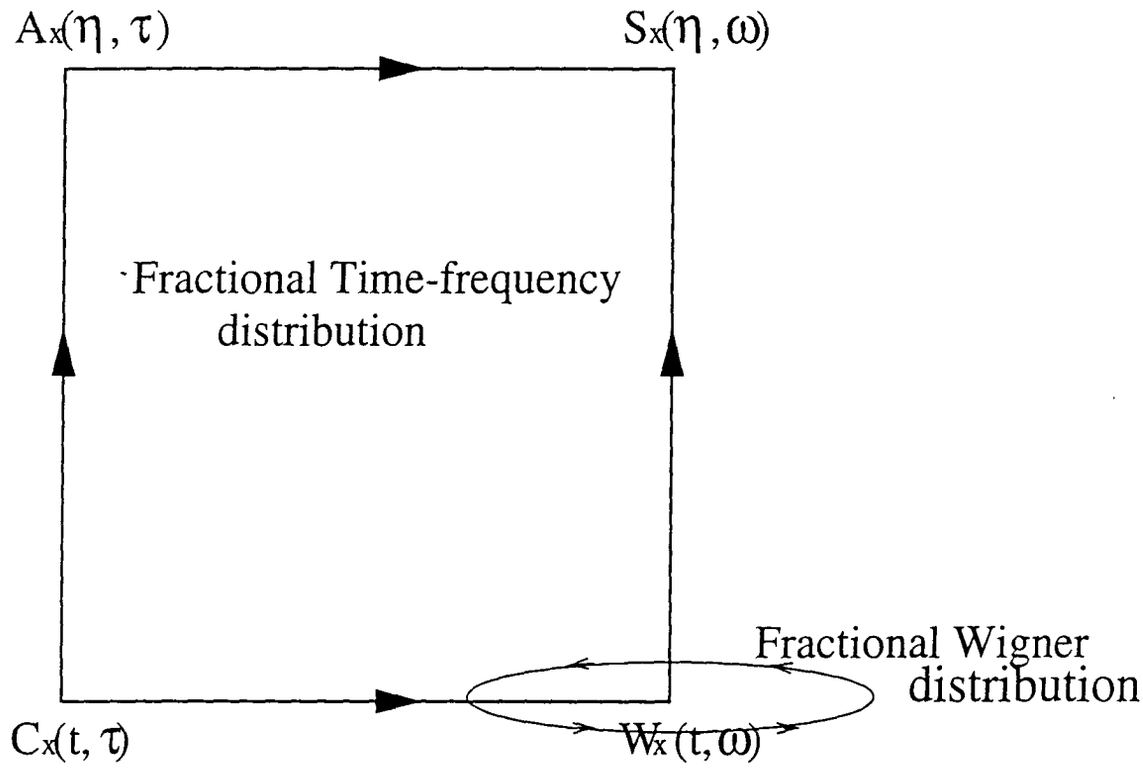


Figure 3: The different concepts of fractional Wigner distribution and fractional time-frequency distribution

A. Properties

- **Time-shift**

For any signal $x(t)$, the signal correlation function of the time-lag signal $x(t-\lambda)$ can be computed as:

$$C_x(t-\lambda,\tau) \tag{16}$$

So the fractional time-frequency distribution can be computed as:

$$\Gamma_{x,\alpha_1,\alpha_2}(\nu-\lambda\cos\alpha_1,\mu)e^{j\frac{\lambda^2}{2}\sin\alpha_1\cos\alpha_1-j\nu\lambda\sin\alpha_1} \tag{17}$$

where $\Gamma_{x,\alpha_1,\alpha_2}(\nu,\mu)$ is the fractional time-frequency distributions of signal $x(t)$ with parameters α_1, α_2. If the first angular parameter α_1 is equal to 0, the distribution becomes:

$$\Gamma_{x,0,\alpha_2}(\nu-\lambda,\mu) \tag{18}$$

Thus the time-shift invariant property is preserved. Figure 4(a) shows the angular scope of fractional time-frequency distribution that the time-shift invariant property is satisfied. It can be observed that two special cases, Wigner distribution and Signal correlation function, can have the time-invariant property. If the first parameter α_1 is equal to $\frac{\pi}{2}$, the distribution becomes:

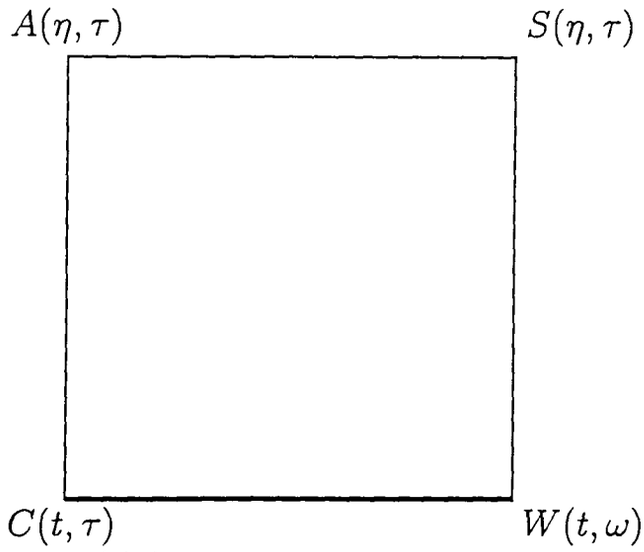
$$\Gamma_{x,\frac{\pi}{2},\alpha_2}(\nu,\mu)e^{-j\nu\lambda} \tag{19}$$

It will produce a frequency shift in the distribution. Figure 4(b) shows the angular scope of fractional time-frequency distribution that have time-shift in signal and the frequency-shift in distribution. It can be observed that two special cases, Ambiguity function and Spectrum correlation function, can preserve this property.

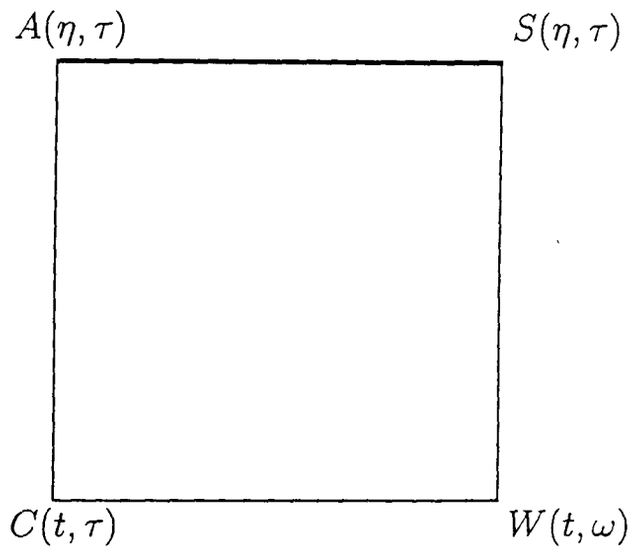
- **Frequency-shift**

For any signal $x(t)$, the signal correlation function of the frequency-shift signal $x(t)e^{j\lambda t}$ can be computed as:

$$C_x(t,\tau)e^{j\lambda\tau} \tag{20}$$



(a) Signal: time-shift
Distribution: time-shift



(b) Signal: time-shift
Distribution: frequency-shift

Figure 4: The scope of fractional time-frequency distribution for the time-shift property preserved

The fractional time-frequency distribution of the frequency-shift signal can be computed as:

$$\Gamma_{x,\alpha 1,\alpha_2}(\nu,\mu-\lambda\sin\alpha_2)e^{-j\frac{\lambda^2}{2}\sin\alpha_2\cos\alpha_2+j\mu\lambda\cos\alpha_2} \tag{21}$$

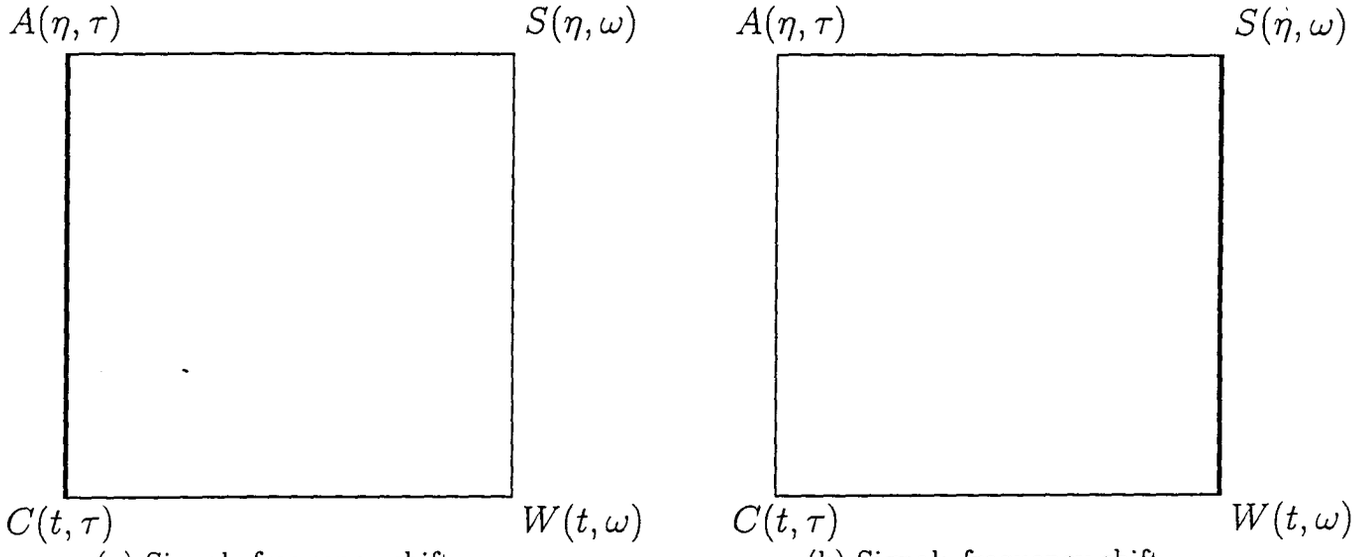
where $\Gamma_{x,\alpha_1,\alpha_2}(\nu,\mu)$ is the fractional time-frequency distributions of signal $x(t)$ with parameters α_1, α_2. If the second angular parameter α_2 is equal to zero, the distribution is computed as:

$$\Gamma_{x,\alpha_1,0}(\nu,\mu)e^{j\mu\lambda} \tag{22}$$

Thus the distribution have a frequency shift. Figure 5(a) shows the angular scope of fractional time-frequency distribution that the time-shift invariant property is satisfied. Two special cases, Ambiguity function and Spectrum correlation function, can preserve this property. If the second parameter α_2 is equal to $\frac{\pi}{2}$, the distribution becomes:

$$\Gamma_{x,\alpha_1,\frac{\pi}{2}}(\nu,\mu-\lambda) \tag{23}$$

It will produce a frequency shift in the distribution. Figure 5(b) shows the angular scope of fractional time-frequency distribution that the frequency-shift invariant property is satisfied. Two special cases, Ambiguity function and Spectrum correlation function, can preserve this frequency-shift invariant property.



(a) Signal: frequency-shift
Distribution : time-shift

(b) Signal: frequency-shift
Distribution : frequency-shift

Figure 5: The scope of fractional time-frequency distribution for the time-shift property preserved

- **Integration in time/frequency-lag parameter**

  If we calculate the integration of fractional time-frequency distribution in the time/frequency-lag variable, the following result is obtained:

$$\int \Gamma_{x,\alpha_1,\alpha_2}(\nu,\mu)d\nu \;=\; \Gamma_{x,\alpha_1\frac{\pi}{2},\alpha_2}(0,\mu) \tag{24}$$

  The equation (24) indicates that the integration of time/frequency-lag variable for a fractional time-frequency distribution with parameter (α_1,α_2) will cause the DC value of the distribution with parameter ($\alpha_1 + \frac{\pi}{2}$,α_2). In the Wigner distribution case($\alpha_1 = 0$, $\alpha_2 = \frac{\pi}{2}$),

$$\int \Gamma_{x,0,\frac{\pi}{2}}(\nu,\mu)d\nu \;=\; \Gamma_{x,\frac{\pi}{2},\frac{\pi}{2}}(0,\mu)$$

$$= S_x(0,\mu)$$
$$= ||X(\mu)||^2$$

We can verify that the Wigner distribution can preserve the frequency marginal property[2]. In the Ambiguity function case ($\alpha_1 = \frac{\pi}{2}$, $\alpha_2 = 0$), equation (24) becomes:

$$\int \Gamma_{x,\frac{\pi}{2},0}(\nu,\mu)d\nu = \Gamma_{x,\pi,0}(0,\mu)$$
$$= C_x(0,\mu)$$

- **Integration in frequency/time-lag parameter**

If we calculate the integration of fractional time-frequency distribution in the frequency/time-lag variable, the following result is obtained:

$$\int \Gamma_{x,\alpha_1,\alpha_2}(\nu,\mu)d\mu = \Gamma_{x,\alpha_1+\frac{\pi}{2},\alpha_2}(\nu,0) \quad (25)$$

The equation (25) indicates that the integration of frequency/time-lag variable for a fractional time-frequency distribution with parameter (α_1, α_2) will cause the DC value of the distribution with parameter (α_1, $\alpha_2 + \frac{\pi}{2}$). In the Wigner distribution case($\alpha_1 = 0$, $\alpha_2 = \frac{\pi}{2}$), equation (25) becomes:

$$\int \Gamma_{x,\alpha_1,\alpha_2}(\nu,\mu)d\nu = \Gamma_{x,\alpha_1,\alpha_2+\frac{\pi}{2}}(0,\mu)$$
$$= C_x(t,\tau)$$

We can verify that the Wigner distribution can preserve the time marginal property[2]. In the Ambiguity function case ($\alpha_1 = \frac{\pi}{2}$, $\alpha_2 = 0$), equation (25) becomes:

$$\int \Gamma_{x,\frac{\pi}{2},0}(\nu,\mu)d\mu = \Gamma_{x,\frac{\pi}{2},\frac{\pi}{2}}(\nu,0)$$
$$= S_x(\nu,0)$$

- **Scaling**

For any signal $x(t)$, the fractional time-frequency distributions of the scaled signal $x(ct)$ can be computed as

$$\sqrt{\frac{1-j\cot\alpha_1}{c^2-j\cot\alpha_1}}\sqrt{\frac{1-j\cot\alpha_2}{c^2-j\cot\alpha_2}}e^{j\frac{\mu^2}{2}\cot\alpha_1(1-\frac{\cos^2\beta_1}{\cos^2\alpha_1})}e^{j\frac{\mu^2}{2}\cot\alpha_2(1-\frac{\cos^2\beta_2}{\cos^2\alpha_2})}\Gamma_{x,\beta_1,\beta_2}(\nu\frac{\sin\beta_2}{a\sin\alpha_2},\mu\frac{\sin\beta_1}{a\sin\alpha_1}) \quad (26)$$

where $\beta_1 = \tan^{-1}(a^2 \tan\alpha_1)$ and $\beta_2 = \tan^{-1}(a^2 \tan\alpha_2)$. $\Gamma_{x,\beta_1,\beta_2}$ is the fractional time-frequency distribution of signal $x(t)$ with angular parameters β_1 and β_2. While $\alpha_1 = 0$ and $\alpha_2 = \frac{\pi}{2}$ (Wigner distribution), the distribution of the scaling signals becomes:

$$W_x(ct, -\frac{\omega}{c}) \tag{27}$$

While $\alpha_2 = 0$ and $\alpha_1 = \frac{\pi}{2}$ (Ambiguity function), the distribution of the scaling signals becomes:

$$A_x(-\frac{t}{c}, c\omega) \tag{28}$$

- **Inverse Problem**

  For any fractional time-frequency distribution with angular parameters (α_1, α_2), the signal correlation function $C_x(t,\tau)$ can be obtained by a 2D FRFT computation with angular parameter $(-\alpha_1, -\alpha_2)$.

  How to compute the signal from its signal correlation function? There exist a strip $\tau = 2t$ in the $C_x(t,\tau)$. Taking the strip of $C_x(t,\tau)$ will reach the original signal.

$$C_x(t, 2t) = x(2t)x^*(0)$$

  So the original signal can be recovered by the following equation:

$$x(t) = \frac{C_x(\frac{t}{2}, t)}{x^*(0)} \tag{29}$$

B. Fractional Time-Frequency Distributions of Special Signals

In this subsection, we will illustrate the fractional time-frequency distributions of several special signals.

- **Gaussian signal**

  For the normalized Gaussian signal, $x(t) = \frac{1}{\sqrt{\sqrt{\pi}}} e^{-\frac{t^2}{2}}$. Thus, the signal correlation function is computed as:

$$C_x(t,\tau) = \frac{1}{\sqrt{\pi}} e^{-(t^2 + \frac{\tau^2}{4})} \tag{30}$$

The fractional time-frequency distribution of signal $x(t)$ is obtained:

$$\Gamma_{x,\alpha_1,\alpha_2}(\nu,\mu) = \sqrt{\frac{1-j\cot\alpha_1}{1-\frac{1}{2}j\cot\alpha_1}}\sqrt{\frac{1-j\cot\alpha_2}{1-\frac{1}{2}j\cot\alpha_2}}e^{j\frac{\nu^2}{2}\cot\alpha_1(1-\frac{\cos^2\beta_1}{\cos^2\beta_1})}e^{-\frac{\nu^2}{2\rho_1^2}}e^{j\frac{\mu^2}{2}\cot\alpha_2(1-\frac{\cos^2\beta_2}{\cos^2\beta_2})}e^{-\frac{\mu^2}{2\rho_2^2}} \tag{31}$$

where

$$\rho_1 = \sqrt{2\cos^2\alpha_1 + \tfrac{1}{2}\sin^2\alpha_2}, \quad \beta_1 = \tan^{-1}(\tfrac{\tan\alpha_2}{2})$$
$$\rho_2 = \sqrt{2\cos^2\alpha_2 + \tfrac{1}{2}\sin^2\alpha_1}, \quad \beta_2 = \tan^{-1}(\tfrac{\tan\alpha_1}{2})$$

- **Sinusoidal Signals**

For a generalized sinusoidal signal $x(t) = e^{jct}$, the signal correlation function can be computed as:

$$C_x(t,\tau) = e^{jct\tau} \tag{32}$$

Thus the fractional time-frequency distribution of the generalized sinusoidal signal $x(t)$is computed as:

$$\sqrt{1+j\tan\alpha_1}e^{-j\frac{c^2+\mu^2}{2}\tan\alpha_1+jc\mu\sec\alpha_1}\sqrt{1+j\tan\alpha_2}e^{-j\nu^2 2\tan\alpha_2} \tag{33}$$

- **Chirp Signals**

For a generalized chirp signal $x(t) = e^{j(\frac{a}{2}t^2+bt)}$, the signal correlation function can be computed by the definition, and it can be computed as:

$$C_x(t,\tau) = e^{j(at+b)\tau} \tag{34}$$

Then the fractional time-frequency distribution of the generalized chirp signal $x(t)$ can be computed as:

$$\Gamma_{x,\alpha_1,\alpha_2}(\nu,\mu) = \sqrt{\frac{(1+j\tan\alpha_1)(1+j\tan\alpha_2)}{1-a^2\tan\alpha_1\tan\alpha_2}}e^{A\nu^2+B\nu\mu+C\mu^2+D\nu+E\mu+F} \tag{35}$$

where

$$A = \frac{1}{2}\frac{-a^2\tan\alpha_1 - \tan\alpha_2}{1-a^2\tan\alpha_1\tan\alpha_2}$$
$$B = \frac{a\sec\alpha_1\sec\alpha_2}{1-a^2\tan\alpha_1\tan\alpha_2}$$

$$
\begin{aligned}
C &= \frac{1}{2}\frac{1-\tan\alpha_1-a^2\tan\alpha_2}{1-a^2\tan\alpha_1\tan\alpha_2} \\
D &= ab\frac{-\tan\alpha_1\sec\alpha_2}{1-a^2\tan\alpha_1\tan\alpha_2} \\
E &= \frac{b\sec\alpha_1}{1-a^2\tan\alpha_1\tan\alpha_2} \\
F &= -\frac{1}{2}b^2\frac{\tan\alpha_1}{1-a^2\tan\alpha_1\tan\alpha_2}
\end{aligned}
$$

The above $\tan\alpha_1\tan\alpha_2 \neq 1$.

V. Applications

In this section, we will introduce an application of the fractional time-frequency distribution: Chirp Rate Detection. To begin with, we simplify the general chirp signal as: $e^{j\frac{a}{2}t^2}$. Thus the fractional time-frequency distribution will become:

$$
\sqrt{\frac{(1+j\tan\alpha_1)(1+j\tan\alpha_2)}{1-a^2\tan\alpha_1\tan\alpha_2}}e^{A\nu^2+B\nu\mu+C\mu^2} \tag{36}
$$

where

$$
\begin{aligned}
A &= \frac{1}{2}\frac{-a^2\tan\alpha_1-\tan\alpha_2}{1-a^2\tan\alpha_1\tan\alpha_2} \\
B &= \frac{a\sec\alpha_1\sec\alpha_2}{1-a^2\tan\alpha_1\tan\alpha_2} \\
C &= \frac{1}{2}\frac{1-\tan\alpha_1-a^2\tan\alpha_2}{1-a^2\tan\alpha_1\tan\alpha_2}
\end{aligned}
$$

The peaks of the distribution are occurred while its phases are equal to zero or multiples of 2π. Obviously, there existed infinite peaks in the distributions. We can find that the phase of the distribution is a quadratic function, and its shape is depended on its discriminant of the quadratic function.

$$
\begin{cases}
\Delta < 0 & \text{ellipse} \\
\Delta = 0 & \text{parabola} \\
\Delta > 0 & \text{hyperbola}
\end{cases}
$$

The discriminant of the phase of the distribution can be computed as:

$$\begin{aligned}\Delta &= B^2 - 4AC \\ &= \frac{a^2 - \tan\alpha_1 \tan\alpha_2}{1 - a^2 \tan\alpha_1 \tan\alpha_2}\end{aligned}$$

If the discriminant is greater than zero, the distribution will have ellipse shape. If the discriminant is less than zero, the distribution will have hyperbola shape. These condition can be simplify as:

- If $a > 1$

$$\begin{cases} \frac{1}{a^2} < \tan\alpha_1 \tan_2 < a^2 & \text{Ellipse} \\ \frac{1}{a^2} > \tan\alpha_1 \tan_2 < a^2 \text{ or } \tan\alpha_1 \tan\alpha_2 > a^2 & \text{Hyperbola} \end{cases}$$

- If $a = 1$

  The discriminant will be equal to 1 expect the case $\tan\alpha_1 \tan\alpha_2 = 1$, so the shape of fractional time-frequency distribution of chirp function will be with hyperbola shapes.

- If $a < 1$

$$\begin{cases} \frac{1}{a^2} < \tan\alpha_1 \tan_2 < a^2 & \text{Hyperbola} \\ \frac{1}{a^2} > \tan\alpha_1 \tan_2 < a^2 \text{ or } \tan\alpha_1 \tan\alpha_2 > a^2 & \text{Ellipse} \end{cases}$$

If the shapes of fractional time-frequency distribution are hyperbola, the asymptotes are:

$$\begin{aligned}\nu - m_1\mu &= 0 \\ \nu - m_2\mu &= 0\end{aligned}$$

where

$$\begin{aligned}m_1 &= \frac{-a\sec\alpha_1 \sec\alpha_2 + \sqrt{(a^2 - \tan\alpha_1 \tan\alpha_2)(1 - a^2 \tan\alpha_1 \tan\alpha_2)}}{a^2 \tan\alpha_1 + \tan\alpha_2} \\ m_2 &= \frac{-a\sec\alpha_1 \sec\alpha_2 - \sqrt{(a^2 - \tan\alpha_1 \tan\alpha_2)(1 - a^2 \tan\alpha_1 \tan\alpha_2)}}{a^2 \tan\alpha_1 + \tan\alpha_2}\end{aligned}$$

No matter of the shapes of fractional time-frequency distribution, the geometric center is always located in the original point of $\mu - \nu$ plane. The condition, $a^2 \tan\alpha_1 \tan\alpha_2 = 1$, will cause peaks in the output distributions, so it can be used to detect the chirp rate of chirp signals. Figure 6 shows the relationship of angular parameters α_1 and α_2 for detecting the chirp rate of the signal: e^{jat^2}, where a is the chirp rate of the chirp signal. Now, we analyze the

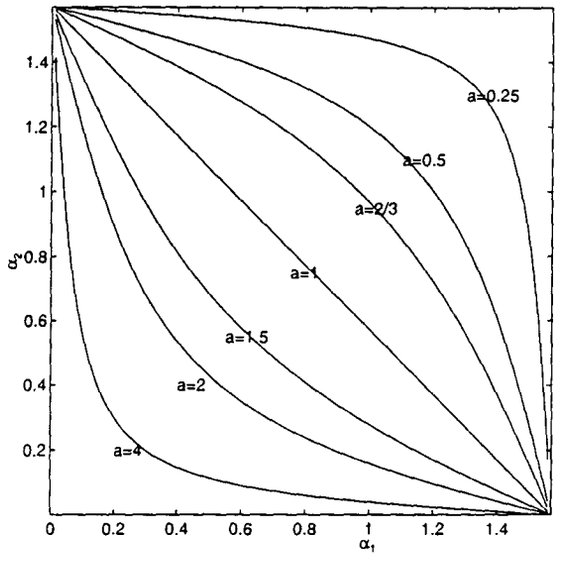


Figure 6: The relationship of angular parameters α_1 and α_2 for detecting a chirp signal

generalized chirp function. The geometric center is $(-\frac{b}{a}\sec\alpha_2, -\frac{b}{a}\tan\alpha_2)$. Asymptotes of the hyperbola are:

$$\begin{aligned} \nu - m_1\mu - p &= 0 \\ \nu - m_2\mu - q &= 0 \end{aligned}$$

where

$$\begin{aligned} m_1 &= \frac{-a\sec\alpha_1\sec\alpha_2 + \sqrt{(a^2 - \tan\alpha_1\tan\alpha_2)(1 - a^2\tan\alpha_1\tan\alpha_2)}}{a^2\tan\alpha_1 + \tan\alpha_2} \\ m_2 &= \frac{-a\sec\alpha_1\sec\alpha_2 - \sqrt{(a^2 - \tan\alpha_1\tan\alpha_2)(1 - a^2\tan\alpha_1\tan\alpha_2)}}{a^2\tan\alpha_1 + \tan\alpha_2} \\ p &= \delta + \sigma \\ q &= \delta - \sigma \end{aligned}$$

$$\delta = \frac{b \sec\alpha_1 \tan\alpha_2}{a^2 \tan\alpha_1 + \tan\alpha_2}\sqrt{\frac{a^2 \tan\alpha_1 \tan\alpha_2 - 1}{1 - a^2 \tan\alpha_1 \tan\alpha_2}}$$
$$\sigma = ab\frac{\tan\alpha_1 \sec\alpha_2}{a^2 \tan\alpha_1 + \tan\alpha_2}$$

The geometric center is always located at the point $(-\frac{ab\sec\alpha_2}{a^2-\tan\alpha_1\tan\alpha_2}, -\frac{b\sec\alpha_1\tan\alpha_2}{a^2-\tan\alpha_1\tan\alpha_2})$.

In the following examples, we will use the fractional time-frequency distributions to detect the chirp rates of chirp signals.

Example 1:

In this example, we will compute the real part of the fractional time-frequency distributions of the chirp signal: e^{jt^2+jt}. Figure 7 and 8 show the fractional time-frequency distribution of the chirp signal with different parameters. It can be found that the choice of parameters $\alpha_1 = \frac{\pi}{6}$ and $\alpha_2 = \frac{\pi}{3}$ can have one straight line as its output. Using the relation: $1 - a\tan\alpha_1\tan\alpha_2 = 0$, the chirp rate a can be detected, and it is equal to 1. Moreover, the initial frequency b can also be computed bye the center point of other fractional time-frequency distributions.

Example 2:

In this example, we will compute the real part of the fractional time-frequency distribution of the chirp signal: $e^{j\frac{t^2}{4}}$. Figure 9 10 and 11 show the fractional time-frequency distributions of the chirp signal. The geometric centers of the output distributions are always located in the original point of $\nu - \mu$ plane in this example. Figure 10 shows the fractional time-frequency distribution with hyperbolic shape. And Figure 11 shows the fractional time-frequency distribution with ellipse shape.

VI. Conclusions

The proposed fractional time-frequency distribution can have fractional time/frequency-lag and fractional time-lag/frequency characteristics of signal, and it relates the four quadratic signal representations with a more general relation. Moreover, the fractional time-frequency

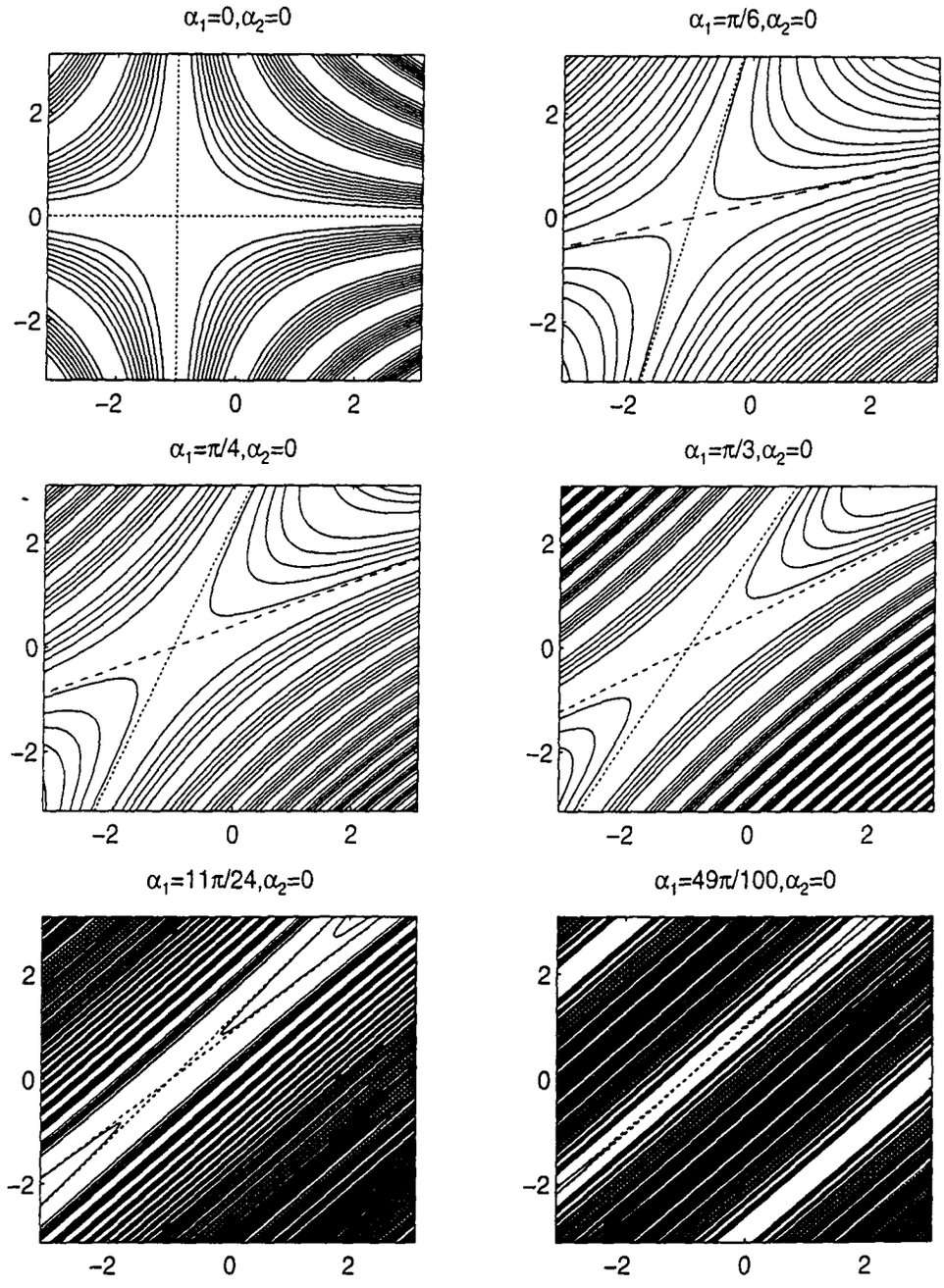


Figure 7: The fractional time-frequency distributions of the chirp signal in Example 1

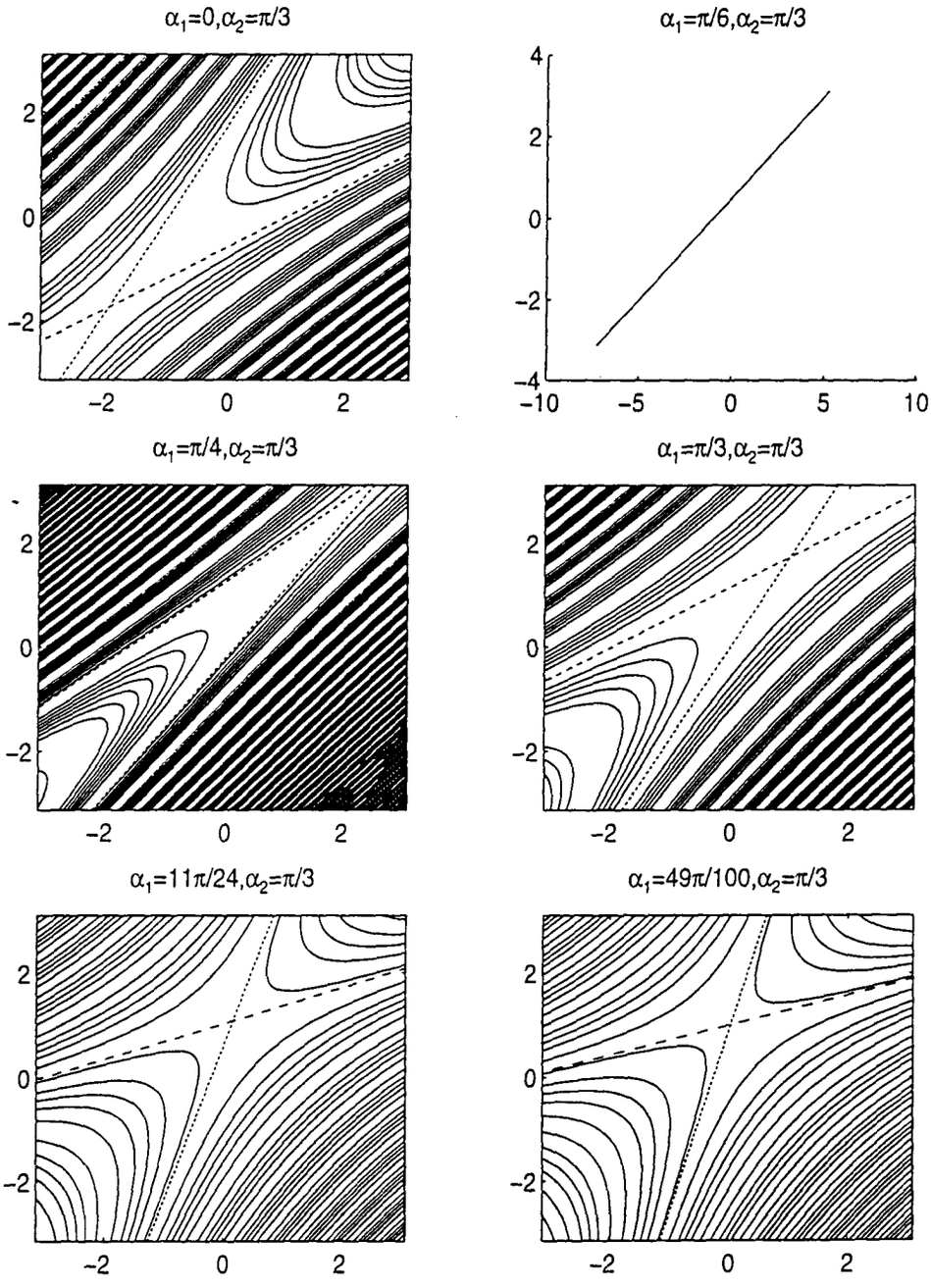


Figure 8: The fractional time-frequency distributions of the chirp signal in Example 1

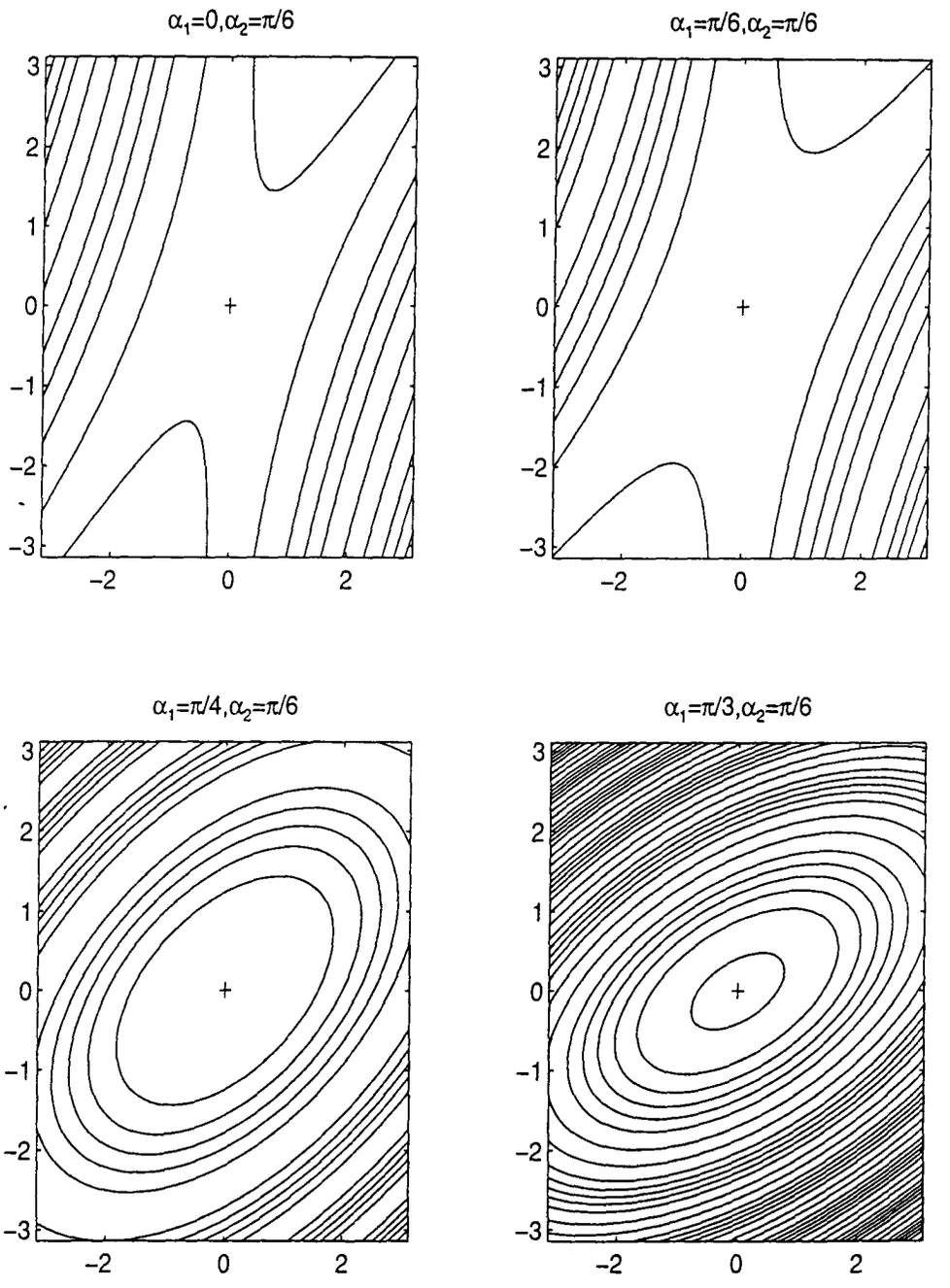


Figure 9: The fractional time-frequency distributions of the chirp signal in Example 2

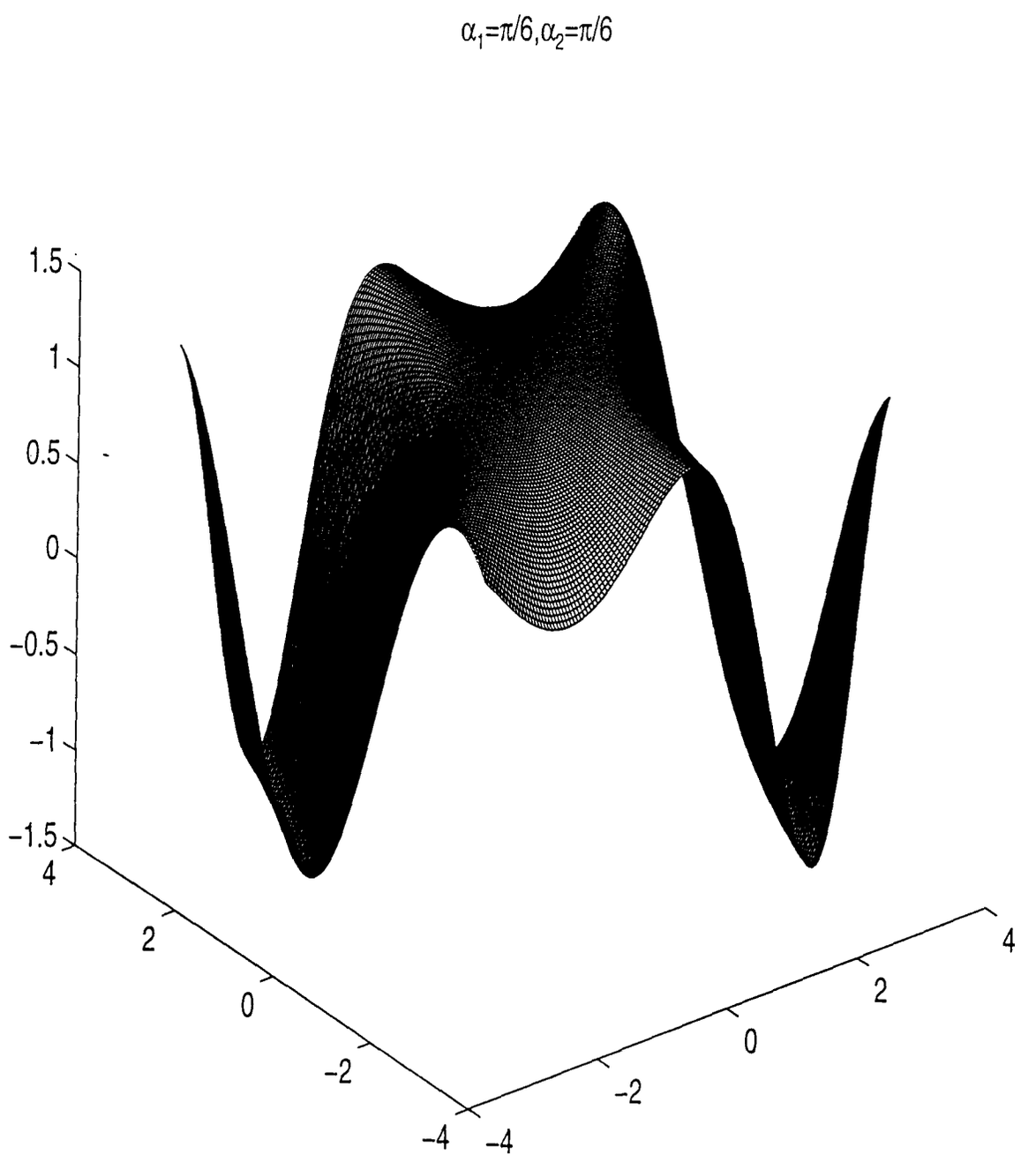


Figure 10: The fractional time-frequency distribution of the chirp signal in Example 2

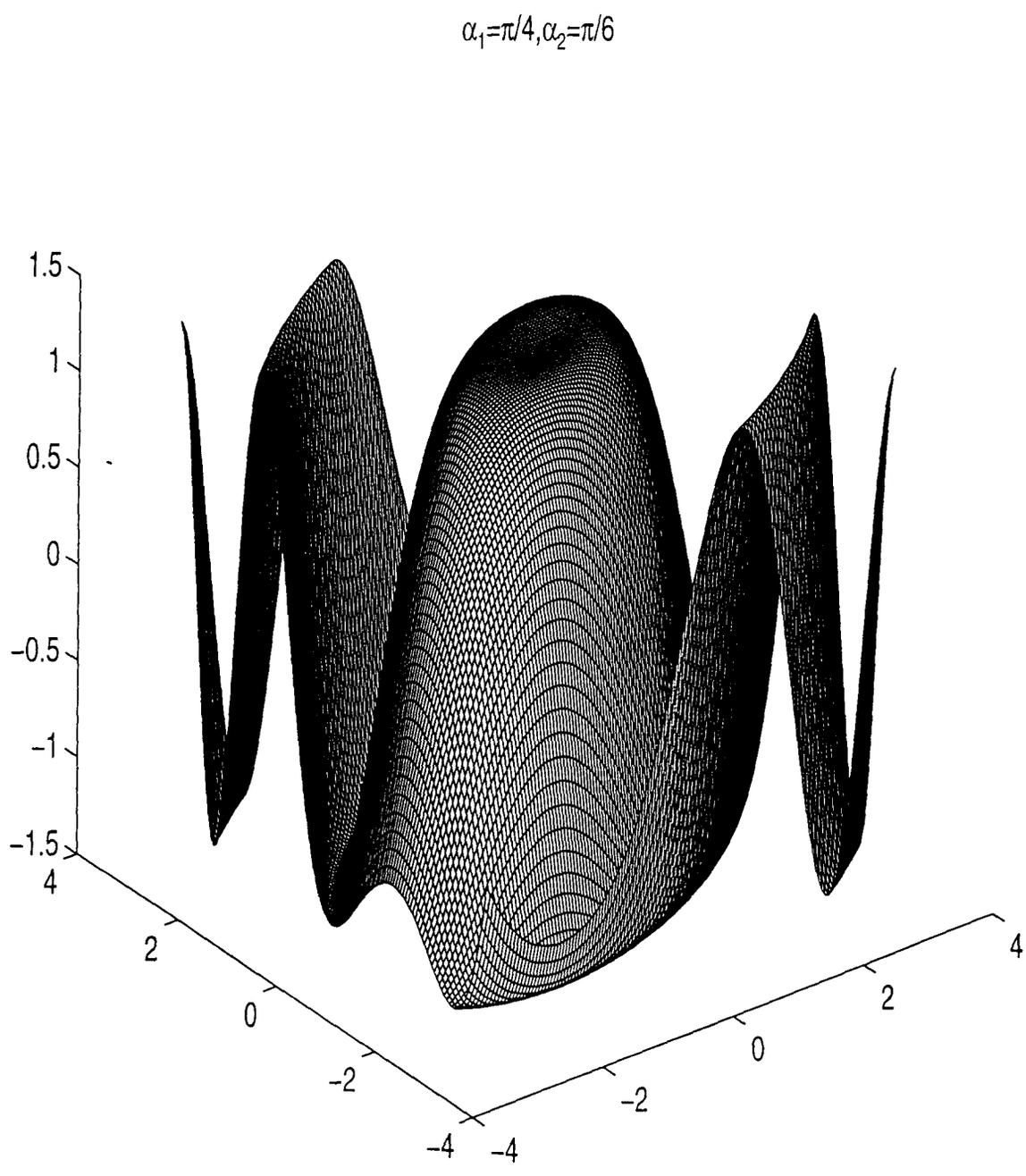


Figure 11: The fractional time-frequency distribution of the chirp signal in Example 2

distribution can be used to detect the chirp rate for chirp signals.

References

[1] L. Cohen, "Time-frequency distribution - a review," *Proc. IEEE*, vol. 77, pp. 941–981, 1989.

[2] F. Hlawatsch and G. F. Bourdeaux-Bartels, "Linear and quadratic time-frequency signal representations," *IEEE Signal Processing Mag.*, vol. 9, pp. 21–67, Apr. 1992.

[3] O. Rioul and P. Flandrin, "Time-scale energy distributions," *IEEE Trans. Signal Process*, vol. 40, pp. 1746–1757, July 1992.

[4] T. A. C. M. Claasen and W. F. G. Meklenbrauker, "The wigner distribution – a tool for time-frequency signal analysis. Part I: continuous-time signals," *Philips J. Res*, vol. 35, pp. 217–250, 1980.

[5] L. Cohen, "Generalized phase-space distribution functions," *J. Math. Phys.*, vol. 7, pp. 781–786, 1966.

[6] A. C. McBride and F. H. Kerr, "On Namias' fractional Fourier transforms," *IMA J. Appl. Math.*, vol. 39, pp. 159–175, 1987.

[7] V. Namias, "The fractional order Fourier transform and its application to quantum mechanics," *J. Inst. Maths Applics*, vol. 25, pp. 241–265, 1980.

[8] L. B. Almeida, "The fractional Fourier transform and time-frequency representation," *IEEE Trans. Signal Process.*, vol. 42, pp. 3084–3091, Nov. 1994.

[9] H. M. Ozaktas, B. Barshan, D. Mendlovic, and L. Onural, "Convolution, filtering, and multiplexing in fractional Fourier domains and their relationship to chirp and wavelet transforms," *J. Opt. Soc. Amer. A*, vol. 11, pp. 547–559, Feb. 1994.

[10] A. Sahlin, H. M. Ozaktas, and D. Mendlovic, "Optical implementation the two-dimensional fractional Fourier transform with different orders in the two dimensions," *Optics Communications*, vol. 120, pp. 134–138, 1995.

[11] D. Dragonman, "Fractional Wigner disbtribution function," *J. Opt. Soc. Amer., A*, vol. 13, pp. 474–478, Mar 1996.

[12] S. C. Pei and T. Y. Wang, "The Wigner distribution of linear time-variant systems," *IEEE Trans. Acoust., Speech, and Signal Process.*, vol. Assp-36, pp. 1681–1684, Oct. 1988.

[13] S. C. Pei and E. J. Tsai, "Cross-terms aanalysis in the modified instantaneous power spectrum," *IEEE Trans. Signal Process.*, vol. Assp-41, pp. 477–480, January 1993.

The Integer Transforms Analogous to Discrete Trigonometric Transforms

Soo-Chang Pei, *Fellow, IEEE,* and Jian-Jiun Ding

***Abstract*—Integer transform (such as the Walsh transform) is the discrete transform that all the entries of the transform matrix are integer. It is much easy to implement because the real number multiplication operations can be avoided, but the performance is usually worse. On the other hand, the noninteger transform, such as the DFT and DCT, has good performance, but the real number multiplication is required. In this paper, we will try to derive the integer transforms analogous to some popular noninteger transforms. These integer transforms remain most of the performance quality of the original transform, but the implementation will be much simpler. Especially, for two-dimensional (2-D) block transform in image/video, the saving can be huge for using integer operations. In 1989, Cham had derived the integer cosine transform. Here, we will derive the integer sine, Hartley, and Fourier transforms. We also introduce the general method to derive the integer transform from some noninteger transform. Besides, the integer transform derived by Cham still requires real number multiplication for the inverse transform. We will modify the integer transform introduced by Cham and introduce the complete integer transform. It requires no real number multiplication operation, no matter what the forward or inverse transform. The integer transform we derive would be more efficient than the original transform. For example, for the 8-point DFT and IDFT, there are in total four real numbers and eight fixed-point multiplication operations required, but for the forward and inverse 8-point complete integer Fourier transforms, there are totally 20 fixed-point multiplication operations required. However, for the integer transform, the implementation is simpler, and many of the properties of the original transform will be kept.**

***Index Terms*—Integer cosine transform, integer Fourier transform, integer Hartley transform, integer sine transform, integer transform.**

I. Introduction

SUPPOSE there is a linear discrete transform with the transform matrix A:

$$Y(m) = \sum_{n=0}^{N-1} A(m,n) \cdot X(n). \tag{1}$$

If the values of $A(m,n)$ can all be expressed as

$$A(m,n) = D \cdot 2^{-h} \quad \text{where } D, h \text{ are integer} \tag{2}$$

then we call this discrete transform the *integer transform*. For example, the Walsh transform and Hadamard transform are all integer transforms.

Manuscript received October 12, 1999; revised August 18, 2000. The associate editor coordinating the review of this paper and approving it for publication was Dr. Xiang-Gen Xia.

The authors are with the Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, R.O.C. (e-mail: pei@cc.ee.ntu.edu.tw).

Publisher Item Identifier S 1053-587X(00)10135-7.

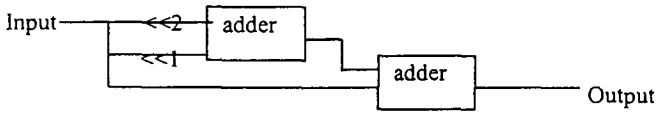


Fig. 1. Implementation of the multiplication of 7 (≪: binary shifting operation).

The most important advantage of the integer transform is that real number multiplication operations are required to implement it. We find that if the transform matrix has some entries that are not in the form of (2), then the real number multiplication operations are required to implement it. The implementation of real number multiplication is usually more complicated and time consuming. In contrast, if all the entries in the transform matrix can be expressed as the form of (2), we only require the fixed-point multiplication operations to implement it. We can implement the fixed-point multiplication operations by the addition and binary shifting operations. For example, if we want to implement the multiplication of 7, we can implement it as Fig. 1.

In addition, for the multiplication of the number not satisfying (2), we cannot implement it by the method of Fig. 1. Besides, for the computer, the multiplication of the noninteger number requires the real number processor and will be more complex and expansive. Thus, if we consider efficiency of hardware implementation and the computation by computer, the integer transform would be better than the noninteger transform.

However, if we consider performance, the noninteger transforms are usually better than the integer transforms. For example, the discrete Fourier transform (DFT) can do the frequency domain analysis, and the discrete cosine transform (DCT) has the high ability of data decorrelation [6]. Thus, although they all require the real number multiplication, they are widely applied.

In this paper, we try to find the integer transforms analogous to some popular noninteger discrete transform, such as the DFT, discrete Hartley transform (DHT), discrete sine transform (DST), and DCT. In [1]–[4], they have derived some types of integer cosine and sine transforms. In this paper, we will improve their methods to derive the integer Fourier and Hartley transforms and other types of integer cosine and sine transforms. Because these integer transforms will approximate to the original transforms, their performance will be very similar to the original transforms, but the real number multiplication can be saved.

Our goal for deriving the integer transforms is not approximating the original transforms. We just want the properties of the original transform to be kept. Then, we can use the transform with much simpler implementation to do the applications

of the original transform. We hope the integer transforms we derive can replace the original transforms in many applications.

In Section II, we will introduce the general method to derive the integer transform from a noninteger transform. We will introduce two types of integer transform. One is near-complete integer transform; it still requires some real number multiplications for the inverse transform. The other is complete integer transform, which requires no real numbers multiplications. From Sections III to V, we will introduce the integer Fourier, Hartley, Sine, and Cosine transforms. We will also compare them with the original noninteger transforms. In Section VI, we will discuss the advantages of the integer transform over the approximation directly and discuss how to choose the parameters. Finally, in Section VII, concludes the paper.

II. General Method to Derive the Integer Transform Analogous to Some Noninteger Transform

A. Conditions for the Integer Transforms

In addition to all the entries being integers, the following requirements must also be satisfied for the integer transform if we want it to be approximated to the original noninteger transform.

a) The equality relations for the entries in each row of the integer transform matrix must be kept exactly the same as the original noninteger transform matrix. That is, if in the original noninteger transform matrix $\mathbf{A}$

$$A(m,n_1) = \tau \cdot A(m,n_2), \quad \tau = 1,-1,j,-j \tag{3}$$

then the integer transform matrix $\mathbf{B}$ must have the following relations:

$$B(m,n_1) = \tau \cdot B(m,n_2). \tag{4}$$

b) In addition, the inequality relations for each row must also be the same. That is, if in the original noninteger transform matrix $\mathbf{A}$

$$\begin{aligned} \mathrm{Re}(A(m,n_1)) &\geq \mathrm{Re}(A(m,n_2)) \\ \mathrm{Im}(A(m,n_3)) &\geq \mathrm{Im}(A(m,n_4)) \end{aligned} \tag{5}$$

then in the integer transform matrix $\mathbf{B}$

$$\begin{aligned} \mathrm{Re}(B(m,n_1)) &\geq \mathrm{Re}(B(m,n_2)) \\ \mathrm{Im}(B(m,n_3)) &\geq \mathrm{Im}(B(m,n_4)). \end{aligned} \tag{6}$$

For flexibility, we allow the original inequality relation without equality in (5) becomes the inequality relation with equality in (6). With the requirements of (a) and (b), we can assure each row of the integer transform matrix will be of the same shape as the row of the original transform matrix.

c) The sign for each entry must be the same as the original. That is, if $\mathbf{A}$ is the original transform matrix, and $\mathbf{B}$ is the integer transform matrix, then

$$\begin{aligned} \mathrm{sgn}(\mathrm{Re}(A(m,n_1))) &= \mathrm{sgn}(\mathrm{Re}(B(m,n_1))) \\ \mathrm{sgn}(\mathrm{Im}(A(m,n_1))) &= \mathrm{sgn}(\mathrm{Im}(B(m,n_1))) \end{aligned} \tag{7}$$

where

$$\begin{aligned} \mathrm{sgn}(x) &= 1 \quad \text{when } x > 0 \\ \mathrm{sgn}(x) &= -1 \quad \text{when } x < 0 \\ \mathrm{sgn}(x) &= 0 \quad \text{when } x = 0. \end{aligned} \tag{8}$$

If this requirement is satisfied, then we can assure the number and the locations of the zero-crossings for each row will remain the same.

d) The basis vectors of the integer transform matrix must also be orthogonal. That is

$$\sum_{k=0}^{N-1} B(m,k) \cdot \overline{B(n,k)} = C_m \cdot \delta_{mn} \tag{9}$$

where $\bar{}$ is the symbol of conjugation. The constraint of (9) can assure that the following relation will be satisfied:

$$\mathbf{B} \cdot \mathbf{B}^{\mathbf{H}} \cdot \mathbf{C}^{-1} = \mathbf{B}^{\mathbf{H}} \cdot \mathbf{C}^{-1} \cdot \mathbf{B} = \mathbf{I} \tag{10}$$

where $^{\mathbf{H}}$ represents the Hermitian operation (transpose and conjugation), and $\mathbf{C}$ is

$$\begin{aligned} C(m,n) &= 0 \quad \text{when } m \neq n \\ C(m,m) &= C_m \quad [C_m \text{ is defined as (9)}]. \end{aligned} \tag{11}$$

Then, the inverse of the integer transform is $\mathbf{B}^{\mathbf{H}}\mathbf{C}^{-1}$. It is just the Hermitian of the forward integer transform matrix with the multiplication operation. The requirement of orthogonality is the key difference between the integer transform with the direct approximation. This will be illustrated in Section VI-A.

If all the four requirements are satisfied, then the integer transform we obtain will have the properties similar to those of the original noninteger transform.

From the above constraints, we can be assured that the forward integer transform only requires the fixed-point multiplication operations. However, since the inverse transform is $\mathbf{B}^{\mathbf{H}}\mathbf{C}^{-1}$, if the value of C_m is not as the form of 2^k where k is some integer, some floating-point multiplications are still required for the inverse transform. Thus, if the integer transform satisfies the above constraints, we can only be assured that the number of floating-point multiplications for the inverse transform is less than N (N is the number of points).

To fully avoid the floating-point multiplication operations, we will use another matrix $\mathbf{E}^{\mathbf{H}}$ other than the Hermitian of the forward integer transform matrix as the inverse transform matrix. This is because it is very hard to find the integer matrix $\mathbf{B}$ such that all the value of C_m in (9) will be the form as 2^k. Requirements a), b), and c) for the forward integer matrix $\mathbf{B}$ must also be satisfied for the inverse transform matrix $\mathbf{E}$, and the requirement d) is modified as

$$\begin{aligned} &\sum_{k=0}^{N-1} B(m,k) \cdot \overline{E(n,k)} = D_m \cdot \delta_{mn} \\ &D_m = 2^k \quad (k \text{ is an integer}). \end{aligned} \tag{12}$$

Then

$$\mathbf{B}\cdot\mathbf{E}^{\mathbf{H}}\cdot\mathbf{D}^{-1}=\mathbf{E}^{\mathbf{H}}\cdot\mathbf{D}^{-1}\cdot\mathbf{B}=\mathbf{I} \tag{13}$$

where

$$\begin{aligned} D(m,n)&=0 \quad \text{when } m\neq n\\ D(m,n)&=D_m \quad \text{when } m=n. \end{aligned} \tag{14}$$

Therefore, the inverse transform matrix is $\mathbf{E}^{\mathbf{H}}\mathbf{D}^{-1}$, and the real number multiplication will be fully not required. We call this type of integer transform the *complete integer transform* and call the former type of integer transform the *near-complete integer transform.*

Sometimes, however, the complete integer transforms are very hard to derive. In these cases, we will either only try to derive near-complete integer transform or relax some requirement in constraints a)–d). For example, we can relax some equality constraint in (4).

Thus, there are two types of integer transforms that can be derived:.

1) Near-Complete Integer Transform: In this case, the inverse of the forward transform matrix $\mathbf{B}$ will be $\mathbf{B}^{\mathbf{H}}\mathbf{C}^{-1}$, where $C(m,n)=C_m\cdot\delta_{m,n}$, and C_m is defined in (9). That is, the basis vectors of $\mathbf{B}$ are orthogonal. Some real numbers multiplications are still required (but we can assure they are less then N).

2) Complete Integer Transform: In this case, the inverse of the forward transform matrix $\mathbf{B}$ will be $\mathbf{E}^{\mathbf{H}}\mathbf{D}^{-1}$. That is, the basis vectors of $\mathbf{B}$ and $\mathbf{E}$ are *dual orthogonal.* No real number multiplication is required.

B. Process to Derive the Integer Transform

When we try to find the integer transform that is analogous to some noninteger transform, we can use the following steps.

1) Forming the Prototype Matrix—To Satisfy the Requirements a) and c): We first form the *prototype* of integer transform matrix and assign the unknowns properly to obey the relations as follows.

1) The equality relations for each row of the original noninteger transform should be kept.
2) The sign of the entries of original noninteger transform should be kept.

To be convenient, all the unknowns are assumed to be positive real integers. If the prototype matrix has too many unknowns, we can try to make some unknowns be the same or make some unknowns be 1 to reduce the number of unknowns.

If we want to derive the complete integer transform, this prototype is common for the forward transform matrix $\mathbf{B}$ and the inverse transform matrix $\mathbf{E}$, but sometimes, we must modify the prototype a little so that the complete integer transform can be derived. For example, when we derive the complete integer Fourier transform of six points, we have modified the original prototype (illustrated in Section III-D).

2) Constraints for Orthogonality (Equality Constraints)—To Satisfy Requirement d): In this step, we search for the requirements to make all the rows of the integer transform matrix be orthogonal to one another. These will be the constraints for the unknowns we have assigned.

If we want to derive the complete integer transform, these types of constraints become the constraints for dual orthogonality for the forward and inverse transform matrix.

3) Constraints for Inequality (Inequality Constraints)—To Satisfy Requirement b): In this step, we find the inequality relations among the unknowns from the inequality relations for each row of the original noninteger transform matrix. These will also be the constraints for the unknowns.

If we want to derive the complete integer transform, this type of constraint is common for the forward and inverse transform matrix.

4) Assign the Values for All the Unknowns: At last, we assign the values of unknowns. They must all be expressed as (2) and satisfy the constraints obtained from steps 2) and 3).

If we want to derive the complete integer transform, the values of unknowns assigned for the forward and inverse transform matrix would be different.

In Sections III to V, we will use this method to derive the integer transform.

We note, however, for the complete integer transform that the inverse matrix $\mathbf{E}^{\mathbf{H}}$ will not be the Hermitian of the forward transform matrix $\mathbf{B}$. However, since $\mathbf{B}$ and $\mathbf{E}$ are of the same prototype, the structures of their implementation are also basically the same, except that the direction is reversed, and the parameters are different.

III. Integer Fourier Transform (8-Points and 6-Points)

A. Near-Complete Integer Fourier Transform

In this subsection, we will derive the 8-point integer Fourier transform (ITFT) analogous to the 8-points DFT:

$$F(m,n)=\exp(-jmn\pi/4)\quad m,n\in[0,1,\ldots 7]. \tag{15}$$

We will follow the method introduced in Section II-B. Because this is the first integer transform we derive in this paper, we will discuss the derivation process in detail.

Step 1) *Forming the prototype matrix of the 8-point ITFT.*

We first list the equality relations for each row of the DFT matrix as in Table I. The values of G and C mean, for the mth row of the 8-points DFT matrix

$$F(m,((n+G))_8)=C\cdot F(m,n) \tag{16}$$

where $(( ))$ is the *modulus addition*

$$((n+G))_N=n+G+h\cdot N$$

where h is some integer and $0\le n+G+hN<N$. (17)

Then, together with the sign of the real and imaginary parts of each entry of the 8-point DFT matrix, we obtain the prototype matrix of the 8-point integer Fourier transform (ITFT) as in (18), shown at the bottom of the next page, and there are a total of 12 unknowns.

Step 2) *Searching the constraints for orthogonality of the unknowns.*

Before discussing the constraints for orthogonality, we first introduce a special way to conclude that two discrete vectors are orthogonal

to each other. Suppose there are two vectors $f_a(n), f_b(n)$ that have the length of N where N is even. In addition, suppose we use some method that make every two points form a set: $\{h_1, k_1\}, \{h_2, k_2\}, \ldots, \{h_{N/2}, k_{N/2}\}$, where $h_m, k_n \in \{0, 1, 2, \ldots, N-1\}$, and $h_m \neq k_n$ for all $m, n, h_m \neq h_n, k_m \neq k_n$ for $m \neq n$. Then, if $f_a(n), f_b(n)$ have the symmetry relations

$$f_a(k_s) = C_s \cdot f_b(h_s), \quad f_a(k_s) = D_s \cdot f_b(h_s) \quad \text{for } s = 1, 2, \ldots, N/2 \tag{19}$$

and C_s, D_s satisfies

$$C_s \cdot \overline{D_s} = -1 \quad \text{for } s = 1, 2, \ldots, N/2 \tag{20}$$

We then find that

$$\langle f_a(n), f_b(n) \rangle = \sum_{s=1}^{N/2} [f_a(h_s) \cdot \overline{f_b(h_s)} + f_a(k_s) \cdot \overline{f_b(k_s)}] \tag{21}$$

$$\langle f_a(n), f_b(n) \rangle = \sum_{s=1}^{N/2} [f_a(h_s) \cdot \overline{f_b(h_s)} + C_s \cdot \overline{D_s} \cdot f_a(h_s) \cdot \overline{f_b(h_s)}] = 0. \tag{22}$$

That is, $f_a(n), f_b(n)$ are orthogonal to each other. Thus, the relation of (19) and (20) will be the sufficient condition for the two functions to be orthogonal.

Thus, from (19)–(22) and Table I, we can conclude, except for {row 1, row 5} and {row 3, row 7}, that all pairs of rows will be orthogonal to each other.

Then, calculating the inner product of {row 1, row 5} and {row 3, row 7} directly, we find that if we want these pairs of rows to be orthogonal, the following equation must be satisfied:

$$a_1 \cdot d_1 = 2 \cdot a_2 \cdot d_2, \quad c_1 \cdot f_1 = 2 \cdot c_2 \cdot f_2. \tag{23}$$

Step 3) *Searching the inequality constraints of the unknowns.*

From the inequality relations for each row of the original DFT matrix, we obtain the inequality constraints for the ITFT as

$$a_1 \geq a_2 \quad c_1 \geq c_2 \quad d_1 \geq d_2 \quad f_1 \geq f_2 \tag{24}$$

and we have obtained all the constraints for the unknowns.

From all the constraint equations, we find that the unknowns set $\{a_1, a_2, d_1, d_2\}$ is independent of the set $\{c_1, c_2, f_1, f_2\}$, and their constraints are all the same. There is also no constraint for b_1, b_2, e_1, e_2. Thus, we can use the following equality equations:

$$a_1 = f_1, \quad a_2 = f_2, \quad d_1 = c_1 \quad d_2 = c_2, \quad b_1 = b_2 = e_1 = e_2 = 1. \tag{25}$$

We note that after we use the above equality relations, then the ITFT we obtain will satisfy

$$FI_p(m, n) = \overline{FI_p(N-m, n)}. \tag{26}$$

Then, the conjugation system property, which is the important property of the original DFT, will be kept. Although we can also use $a_1 = c_1, a_2 = c_2, d_1 = f_1, d_2 = f_2$ instead of $a_1 = f_1, a_2 = f_2, d_1 = c_1, d_2 = c_2$, when we use these equality relations, then the conjugation system relation of (26) will not exist. From (25), the number of unknowns of the prototype matrix is reduced from 12 to 4, and the prototype matrix of ITFT in (18) is simplified as in (27), shown at the bottom of the next page. The constraints then become

$$a_1 \cdot c_1 = 2 \cdot a_2 \cdot c_2, \quad a_1 \geq a_2, \quad c_1 \geq c_2. \tag{28}$$

Step 4) Assign the values of unknowns.

Then, we follow step 4 in Section II-B and assign the values of unknowns to satisfy (28) and (29), shown at the bottom of the next page. We find that there are many possible choices for the values of unknowns $\{a_1, a_2, c_1, c_2\}$. We list some possible choices of the parameters as Table II.

We can implement 8-point ITFT by the method of 1) decimation-in-frequency (as shown in Fig. 2) and 2) decimation-in-time (as shown in Fig. 3). We find that both the decimation-in-time implementation and the decimation-in-frequency implementation require eight integer multiplications. For the original DFT, we require four real number multi-

$$\mathbf{FI_P} = \begin{bmatrix} b_1 & b_1 & b_1 & b_1 & b_1 & b_1 & b_1 & b_1 \\ a_1 & a_2 - ja_2 & -ja_1 & -a_2 - ja_2 & -a_1 & -a_2 + ja_2 & ja_1 & a_2 + ja_2 \\ b_2 & -jb_2 & -b_2 & jb_2 & b_2 & -jb_2 & -b_2 & jb_2 \\ c_1 & -c_2 - jc_2 & jc_1 & c_2 - jc_2 & -c_1 & c_2 + jc_2 & -jc_1 & -c_2 + jc_2 \\ e_1 & -e_1 & e_1 & -e_1 & e_1 & -e_1 & e_1 & -e_1 \\ d_1 & -d_2 + jd_2 & -jd_1 & d_2 + jd_2 & -d_1 & d_2 - jd_2 & jd_1 & -d_2 - jd_2 \\ e_2 & je_2 & -e_2 & -je_2 & e_2 & je_2 & -e_2 & -je_2 \\ f_1 & f_2 + jf_2 & jf_1 & -f_2 + jf_2 & -f_1 & -f_2 - jf_2 & -jf_1 & f_2 - jf_2 \end{bmatrix} \tag{18}$$

TABLE I
SYMMETRY RELATION OF THE 8-POINTS DFT MATRIX

row	m=0	m=1	m=2	m=3	m=4	m=5	m=6	m=7
G=1	C=1		C=$-j$		C=-1		C=j	
G=2	C=1	C=$-j$	C=-1	C=j	C=1	C=$-j$	C=-1	C=j
G=4	C=1	C=-1	C=1	C=-1	C=1	C=-1	C=1	C=-1

TABLE II
SOME POSSIBLE CHOICES OF THE PARAMETERS OF 8-POINTS ITFT

a_1	2	3	4	5	8	10	17	99	500
a_2	1	2	3	3	5	7	12	70	353
c_1	1	4	3	6	5	7	24	140	353
c_2	1	3	2	5	4	5	17	99	500

plications. Although the 8-point ITFT requires twice the multiplication operations and since all the multiplication operations are fixed-point, it will still be more efficient.

From (10), the inverse 8-point ITFT is as in (29), where

$$\begin{aligned} C(m,n) &= 0 \quad \text{when } m \neq n \\ C(0,0) &= C(2,2) = C(4,4) = C(6,6) = 8 \\ C(1,1) &= C(7,7) = 4a_1^2 + 8a_2^2 \\ C(3,3) &= C(5,5) = 4c_1^2 + 8c_2^2. \end{aligned} \tag{30}$$

The inverse 8-point ITFT can be implemented as in Fig. 4. We note that the implementation of the inverse ITFT has almost the same structure as the implementation of the forward ITFT in Fig. 2. The difference is that the direction is reverse, and there are some multiplication operations for the input of inverse ITFT. In fact, for all of the the integer transform, the implementations of the forward and inverse transforms will have *the same structure*.

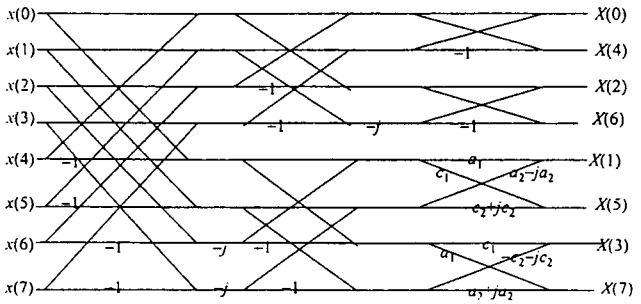


Fig. 2. Decimation-in-frequency 8-point integer Fourier transform.

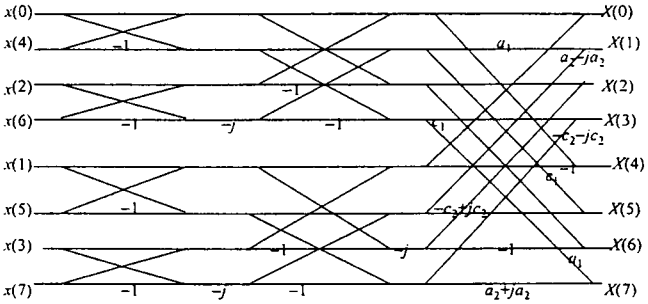


Fig. 3. Decimation-in-time 8-point integer Fourier transform.

B. Complete Integer Fourier Transform of 8-Points

After the near-complete integer Fourier transform has been derived, now we will also derive the complete integer Fourier transform. We first discuss the case of 8-points.

We will also use (27) as the prototype of the forward 8-point complete ITFT. In addition, for the inverse transform, we use a similar prototype but modify the parameters $\{a_1, a_2, c_1, c_2\}$ as

$$\mathbf{FI_P} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ a_1 & a_2 - ja_2 & -ja_1 & -a_2 - ja_2 & -a_1 & -a_2 + ja_2 & ja_1 & a_2 + ja_2 \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ c_1 & -c_2 - jc_2 & jc_1 & c_2 - jc_2 & -c_1 & c_2 + jc_2 & -jc_1 & -c_2 + jc_2 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ c_1 & -c_2 + jc_2 & -jc_1 & c_2 + jc_2 & -c_1 & c_2 - jc_2 & jc_1 & -c_2 - jc_2 \\ 1 & j & -1 & -j & 1 & j & -1 & -j \\ a_1 & a_2 + ja_2 & ja_1 & -a_2 + ja_2 & -a_1 & -a_2 - ja_2 & -ja_1 & a_2 - ja_2 \end{bmatrix}. \tag{27}$$

$$\mathbf{FI_P^{-1}} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ a_1 & a_2 - ja_2 & -ja_1 & -a_2 - ja_2 & -a_1 & -a_2 + ja_2 & ja_1 & a_2 + ja_2 \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ c_1 & -c_2 - jc_2 & jc_1 & c_2 - jc_2 & -c_1 & c_2 + jc_2 & -jc_1 & -c_2 + jc_2 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ c_1 & -c_2 + jc_2 & -jc_1 & c_2 + jc_2 & -c_1 & c_2 - jc_2 & jc_1 & -c_2 - jc_2 \\ 1 & j & -1 & -j & 1 & j & -1 & -j \\ a_1 & a_2 + ja_2 & ja_1 & -a_2 + ja_2 & -a_1 & -a_2 - ja_2 & -ja_1 & a_2 - ja_2 \end{bmatrix}^{\mathbf{T}} \cdot \mathbf{C}^{-1} \tag{29}$$

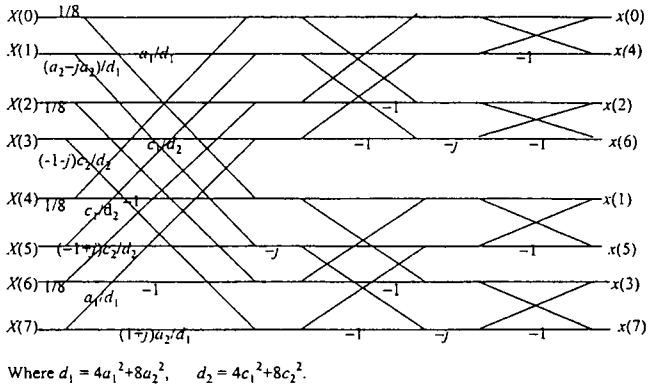


Fig. 4. Implementation of the 8-points inverse integer Fourier transform.

$\{a_3, a_4, c_3, c_4\}$, shown in (31) at the bottom of the page. Therefore, there are a total of eight parameters. The equality constraint in (28) now becomes

$$(1)\ a_1 \cdot c_3 = 2 \cdot a_2 \cdot c_4, \quad (2)\ a_3 \cdot c_1 = 2 \cdot a_4 \cdot c_2. \tag{32}$$

From the requirement of (12), we also obtain the equality constraint as

$$\begin{aligned} &(3)\ a_1 \cdot a_3 + 2 \cdot a_2 \cdot a_4 = 2^k \\ &(4)\ c_1 \cdot c_3 + 2 \cdot c_2 \cdot c_4 = 2^h \end{aligned} \tag{33}$$

where k, h are integer numbers. The inequality constraints in (28) become

$$(5)\ a_1 \geq a_2, \quad (6)\ c_1 \geq c_2, \quad (7)\ a_3 \geq a_4, \quad (8)\ c_3 \geq c_4. \tag{34}$$

Thus, there are a total of four equality constraints and four inequality constraints.

To find the values of the eight parameters $\{a_1, a_2, c_1, c_2, a_3, a_4, c_3, c_4\}$ that satisfy all the above eight constraints directly is time consuming. We introduce a convenient process to assign the values of the parameters as follows. From (1) and (2), we can set

$$c_3 = 2 \cdot a_2, \quad c_4 = a_1, \quad a_3 = 2 \cdot c_2, \quad a_4 = c_1. \tag{35}$$

Then, (3) and (4) become

$$\begin{aligned} &(3')\ 2 \cdot (a_1 \cdot c_2 + a_2 \cdot c_1) = 2^k \\ &(4')\ 2 \cdot (c_1 \cdot a_2 + c_2 \cdot a_1) = 2^h. \end{aligned} \tag{36}$$

We can then implement the following process to find the values of the parameters.

a) Choose the values of a_1, a_2 such that a_1, a_2 are integer numbers and

$$2 \cdot a_2 \geq a_1 \geq a_2. \tag{37}$$

b) Find the integer values of c_1, c_2 such that

$$2 \cdot c_2 \geq c_1 \geq c_2, \quad \text{and} \quad a_1 \cdot c_2 + a_2 \cdot c_1 = 2^n \quad \text{where } n \text{ is integer.} \tag{38}$$

c) Set the value of a_3, a_4, c_3, c_4 as

$$\begin{aligned} &c_3 = 2^{h+1}a_2, \quad c_4 = 2^h a_1, \quad a_3 = 2^{h+1}c_2 \\ &a_4 = 2^h c_1 \quad h \text{ is any integer.} \end{aligned} \tag{39}$$

Then, the parameters are all obtained.

We list some examples of the parameters as follows.

1) $\{a_1 = 2, a_2 = 1, c_1 = 2, c_2 = 1, a_3 = 1, a_4 = 1, c_3 = 1, c_4 = 1\}$
   This is the smallest, simplest integer solution for the parameters. If

$$\sum_{n=0}^{N-1} FI_P(m,n) \cdot \overline{IFI_P(m,n)} = D_m \tag{40}$$

   then $D_0 = D_2 = D_4 = D_6 = 2^3, D_1 = D_3 = D_5 = D_7 = 2^4$.
2) $\{a_1 = 7, a_2 = 5, c_1 = 13, c_2 = 9, a_3 = 18, a_4 = 13, c_3 = 10, c_4 = 7\}$.

We note that when we choose the parameters as the values list above, the ratios of $a_1 : a_2, c_1 : c_2, a_3 : a_4, c_3 : c_4$ are all near to 1.414:1, which is ratio of the original discrete Fourier transform. If D_m is defined as (82), then in this case, $D_0 = D_2 = D_4 = D_6 = 2^3$, $D_1 = D_3 = D_5 = D_7 = 2^9$.

We use Table III to list some possible choices of the values of the parameters of the 8-point ITFT.

We can also implement the forward transform as Fig. 2 or 3 and implement the inverse transform as Fig. 5.

We note that the numbers of multiplication operations required for the 8-point original and complete integer Fourier transform are

Original DFT:

four real numbers, 8 fixed-point multiplication operations

Complete ITFT:

20 fixed-points multiplication operations.

$$\mathbf{IFI_P} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ a_3 & a_4 - ja_4 & -ja_3 & -a_4 - ja_4 & -a_3 & -a_4 + ja_4 & ja_3 & a_4 + ja_4 \\ 1 & -j & -1 & j & 1 & -j & -1 & j \\ c_3 & -c_4 - jc_4 & jc_3 & c_4 - jc_4 & -c_3 & c_4 + jc_4 & -jc_3 & -c_4 + jc_4 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ c_3 & -c_4 + jc_4 & -jc_3 & c_4 + jc_4 & -c_3 & c_4 - jc_4 & jc_3 & -c_4 - jc_4 \\ 1 & j & -1 & -j & 1 & j & -1 & -j \\ a_3 & a_4 + ja_4 & ja_3 & -a_4 + ja_4 & -a_3 & -a_4 - ja_4 & -ja_3 & a_4 - ja_4 \end{bmatrix}. \tag{31}$$

TABLE III
SOME POSSIBLE CHOICES OF THE VALUES OF THE PARAMETERS OF 8-POINT COMPLETE ITFT

a_1	2	7	3	4	4	5	10
a_2	1	5	2	3	3	4	7
c_1	2	13	17	12	44	17	18
c_2	1	9	10	7	31	12	13
a_3	1	18	34	7	31	24	13
a_4	1	13	10	6	22	17	9
c_3	1	10	4	3	3	8	7
c_4	1	7	3	2	2	5	5

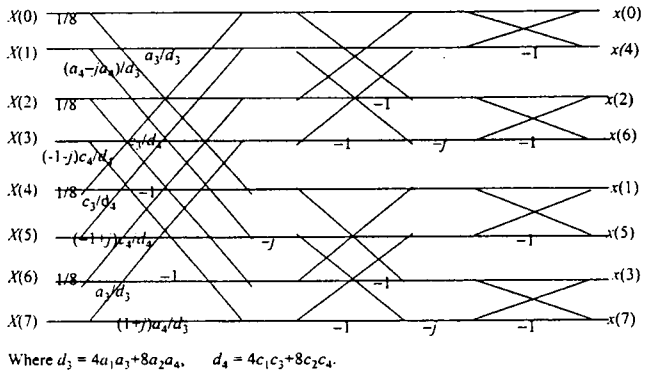


Fig. 5. Implementation of the inverse 8-point complete ITFT.

That is, we replace four real number multiplication operations with 12 fixed-point multiplication operations. The complete ITFT will be much faster than the original DFT.

C. The Properties and Performance of the 8-Point Complete ITFT

We will see some interesting properties of the 8-point complete integer Fourier transform (ITFT). Many properties of the original 8-point DFT will also exist for the 8-point ITFT.

a) Even input → even output, odd input → odd output:
If $x(n)$ is even, and $y(n)$ is odd

$$x(n) = x(N-n), \quad y(n) = y(N-m) \tag{41}$$

then the complete ITFT of $x(n)$ [denoted by $X(m)$] will also be even, and the complete ITFT of $y(n)$ [denoted by $Y(m)$] will also be odd

$$X(m) = X(N-m), \quad Y(m) = -Y(N-m). \tag{42}$$

b) Pure real input → conjugated symmetry output, pure imaginary input → conjugated asymmetry output:
If $x(n)$ is pure real, $y(n)$ is pure imaginary, i.e., $\mathrm{Im}(x(n)) = 0, \mathrm{Re}(y(n) = 0)$, and the complete ITFT of $x(n), y(n)$ are $X(m), Y(m)$, then

$$X(m) = \overline{X(N-m)}, \quad Y(m) = -\overline{Y(N-m)}. \tag{43}$$

c) Power preservation property:
If $X(m), Y(m)$ are the complete ITFT of $x(n), y(n)$, then

$$\sum_{n=0}^{N-1} x(n) \cdot \overline{y(n)} = \sum_{m=0}^{N-1} X(m) \cdot \overline{Y(m)} \cdot D_m^{-1}$$

where D_m is defined as (40). (44)

d) Time-reverse property and conjugation property:

$$\mathrm{ITFT}(x(((-n))_N)) = X(((-m))_N)$$
$$\mathrm{ITFT}(\overline{x(n)}) = \overline{X(((-m))_N)}. \tag{45}$$

e) Shift-invariant property:
For the 8-point complete ITFT, if

$$\mathrm{ITFT}(x(n)) = X(m), \quad \mathrm{ITFT}(x(((n-k))_N)) = X_k(m) \tag{46}$$

then

$$X_k(m) = e^{-j\frac{2\pi k}{N}m} X(m)$$
when k is even or when k is odd and m is even (47)
$$X_k(m) \approx e^{-j\frac{2\pi k}{N}m} X(m)$$
when k is odd and m is odd. (48)

Equation (47) is satisfied because the equality relations for each row of the original 8-point DFT matrix (listed in Table I) have been kept for the 8-point ITFT; therefore, for the 8-point ITFT matrix, the relation that $FI(m,((n+G))_N) = \exp(-j2\pi mG/N) \cdot FI(m,n)$ is still satisfied when G is even or when m is even. Thus, if the amount of time-shift k is even or m is even, then

$$\begin{aligned} X_k(m) &= \sum_{n=0}^{N-1} FI(m,n)x(((n-k))_N) \\ &= \sum_{n=0}^{N-1} FI(m,((n+k))_N)x(n) \\ &= e^{-j\frac{2\pi mk}{N}} \sum_{n=0}^{N-1} FI(m,n)x(n) \\ &= e^{-j\frac{2\pi mk}{N}} \cdot X(m) \end{aligned}$$

and the relation of (47) has been proved.

Then, we will see the performance of the 8-point ITFT. Here, the parameters we choose are $a_1 = 7, a_2 = 5, c_1 = 13, c_2 = 9, a_3 = 18, a_4 = 13, c_3 = 10, c_4 = 7$. Then, we just require 4 bits to implement the kernel of forward ITFT (since the largest entry is $c_1 = 13$) and require 5 bits to implement the kernel of inverse ITFT (since the largest entry is $a_3 = 18$).

We first see the transform result. We choose two inputs:

$$\mathbf{x_1} = [2\ 3\ 4\ 5\ 4\ 5\ 2\ 3] \tag{49}$$
$$\mathbf{x_2} = [2.8\ 4.3 - 0.6j\ 3.7 + 0.9j\ 3.1 - 0.6j\ 4.6\ 3.1 + 0.6j\ 3.7 - 0.9j\ 4.3 + 0.6j]. \tag{50}$$

We plot them in Fig. 6(a) and (b). (The real part and imaginary part are plotted in separation.). Then, we do the original 8-point DFT, and plot the transform results in Fig. 6(c) and (d). Then, we do the 8-point complete ITFT. To facilitate the comparison, we will normalize the transform results. That is, for the transform result of the complete ITFT

$$X(m) = \sum_{n=0}^{7} FI(m,n) \cdot x(n) \tag{51}$$

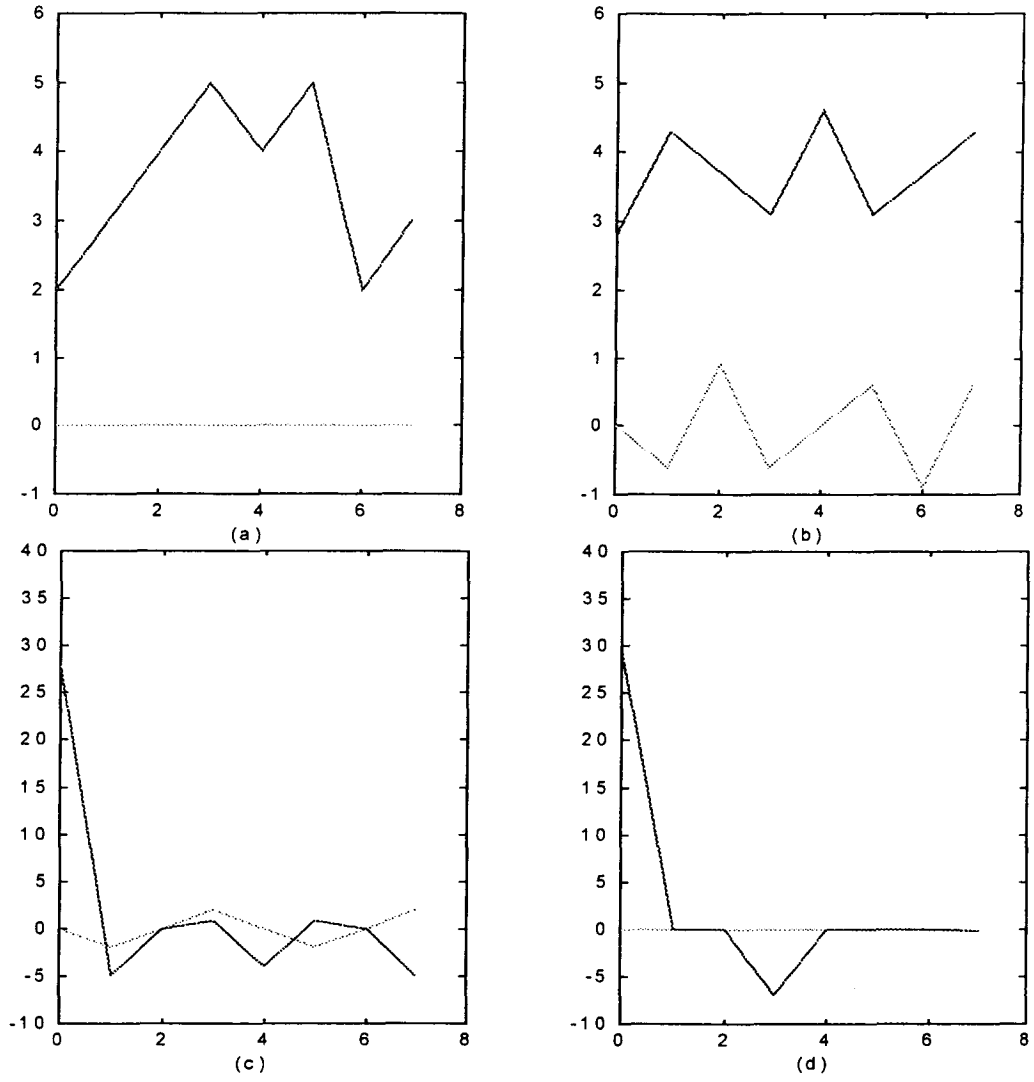


Fig. 6. Transform results of the original, approximated DFTs and complete ITFT. (a), (b) Original signals. (c), (d) Transform results of original DFT.

we will normalize $X(m)$ by the first column of the forward integer transform matrix

$$\tilde{X}(m) = X(m)/FI(m,0). \tag{52}$$

We plot the normalized transform results of the 8-point complete ITFT in Fig. 7(c) and (d).

We will compare the complete ITFT with the direct approximation of DFT. Since, in this example, we require 4 bits to implement the kernel of forward ITFT and require 5 bits to implement the kernel of inverse ITFT, we will also use 4 bits to approximate the forward DFT and use 5 bits to approximate the inverse DFT:

$$\hat{F}(m,n) \approx F(m,n) = \sum_{h=0}^{3} a_h \cdot 2^{H-h}$$

$$I\hat{F}(m,n) \approx IF(m,n) = \sum_{h=0}^{4} b_h \cdot 2^{H-h}. \tag{53}$$

Here, we use $F(m,n), IF(m,n), \hat{F}(m,n), I\hat{F}(m,n)$ to denote DFT, IDFT, approximated DFT, and approximated IDFT, respectively. Then, we use the approximated DFT to calculate the transform results of $\mathbf{x_1}, \mathbf{x_2}$ and plot the results in Fig. 7(a) and (b).

We then use the following equation to calculate the approximation error.

$$\text{err} = \sqrt{\sum_{m=0}^{N-1} |\tilde{X}(m) - X_F(m)|^2 \Big/ \sum_{m=0}^{N-1} |X_F(m)|^2} \tag{54}$$

where $X_F(m)$ is the transform result of original DFT. Then, the errors of the four results in Fig. 7 are

$$\begin{aligned} \text{for Fig. 7(a): err} &= 5.3363 \cdot 10^{-3} \\ \text{for Fig. 7(b): err} &= 4.3759 \cdot 10^{-3} \\ \text{Fig. 7(c): err} &= 3.1655 \cdot 10^{-3} \\ \text{for Fig. 7(d): err} &= 2.5958 \cdot 10^{-3}. \end{aligned} \tag{55}$$

We find that the errors are very small. The approximated results of the complete ITFT [Fig. 7(c) and (d)] are even better than the direct approximation (Fig. 7(a), (b)) and are very similar to the transform results for the original 8-point DFT.

Besides, we note that input $x_2[n]$ is the combination of a low-frequency component and a high-frequency component. These

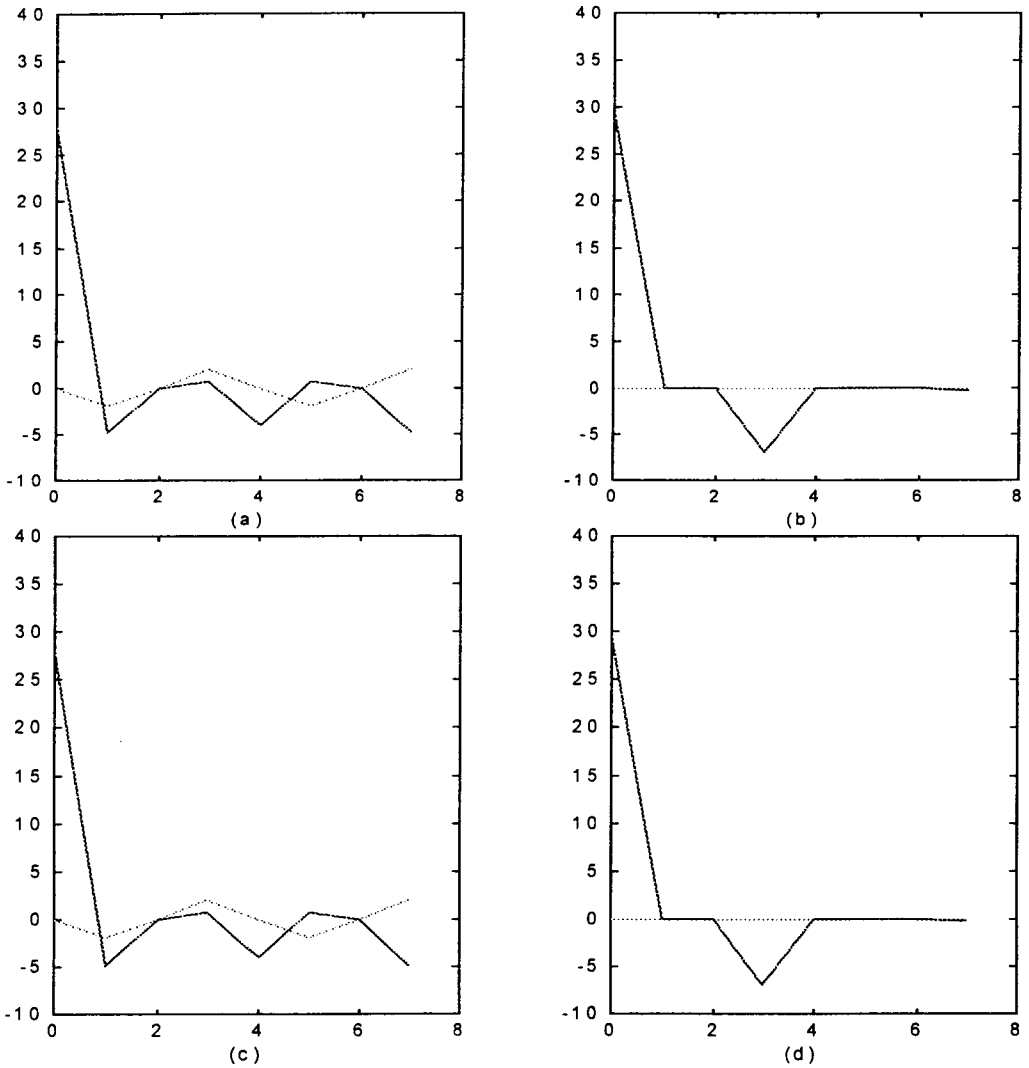


Fig. 7. Transform results of the original, approximated DFTs and complete ITFT. (a), (b) Transform results of approximated DFT. (c), (d) Transform results of complete ITFT.

two components can be separated by both original and complete ITFTs. Therefore, the complete ITFT can also be used for the filter design. We can set

$$X_f(m) = X(m)$$
$$(X(m) \text{ is the complete ITFT of } x(n)) \quad \text{when } m \neq 3,$$
$$X_f(m) = 0 \quad \text{when } m \neq 0$$

to remove the high-frequency component and then do the inverse complete ITFT for $X_f(m)$

$$x_o(n) = \sum_{n=0}^{7} \overline{IFI(n,m)} \cdot D_m^{-1} \cdot X_f(m)$$

to obtain the filter output. In Fig. 8, we show the filter results. We plot the original signal $\mathbf{x_2}$ in Fig. 8(a). Then, we use the original DFT to filter out the high-frequency component, and plot the result in Fig. 8(b). Then, we use the approximated DFT [as (53)] and plot the result in Fig. 8(c). Then, we use the complete ITFT and plot the result in Fig. 8(d). Then, we use (54) to calculate the error, but $X_F(m)$ is changed to the filter output by the original DFT, and $\tilde{X}(m)$ is changed to the filter outputs by the approximated DFT and by the complete ITFT. The errors are

$$\text{for Fig. 8(c): err} = 3.2639 \cdot 10^{-3}$$
$$\text{for Fig. 8(d): err} = 1.1952 \cdot 10^{-3}.$$

We find that the error of approximated DFT is about three times the error of the complete ITFT. This is because the complete ITFT is orthogonal and, hence, is reversible. However, the approximated DFT is not orthogonal and reversible, especially when we use fewer bits. Therefore, for the applications that we must do both the forward and inverse transforms, such as the filter design, the performance of complete ITFT will be much better than the approximated DFT.

Then, we see the displacement property of the complete ITFT. Here, we use the displacement of input $x_2[n]$ [which is defined in (50)] as the input:

$$x_3[n] = x_2[((n+1))_8], \quad x_4[n] = x_2[((n+2))_8]. \quad (56)$$

We plot $\mathbf{x_3}, \mathbf{x_4}$ in Fig. 9(a) and (b). Then, we do the complete ITFT for $\mathbf{x_3}, \mathbf{x_4}$ and plot the amplitudes of the normalized results in Fig. 10(c) and (d). We also do the original DFT and the

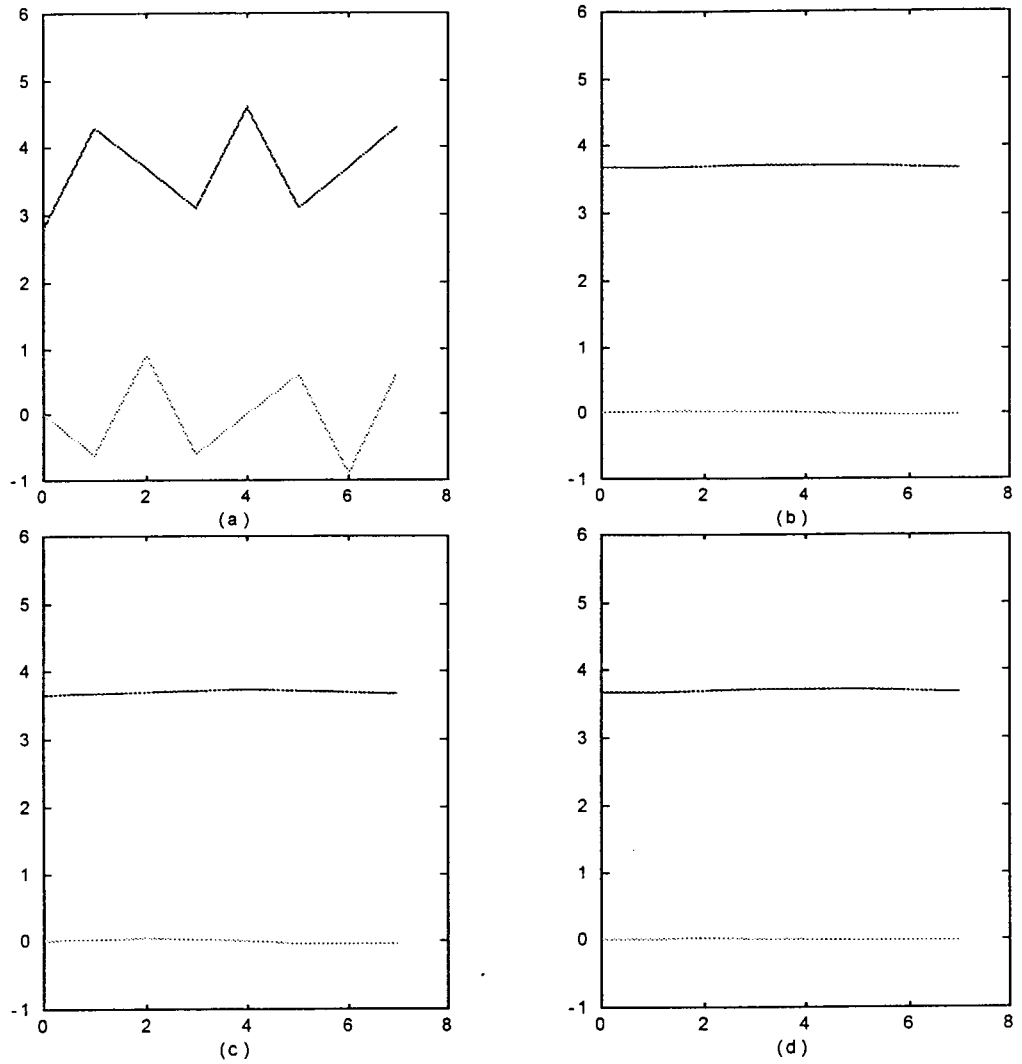


Fig. 8. Filter experiment. (a) Original signal. (b) Filter by original DFT. (c) Filter by approximated DFT. (d) Filter by complete ITFT.

approximated DFT [as in (53)] for $\mathbf{x_3}, \mathbf{x_4}$, plot the amplitudes of the results of the original DFT in Fig. 9(a) and (b), and plot the amplitudes of the results of the approximated DFT in Fig. 10(a) and (b).

We find that for the complete ITFT, the displacement property is almost kept. The signal after displacement will have the same transform amplitude as the original signal. When we compare the phase, there is an interesting phenomenon. We find that

$$\begin{aligned}&\text{angle}(\mathbf{X_3}) - \text{angle}(\mathbf{X_2})\\ &\quad = [0 \quad -45 \quad -90 \quad -135 \quad 0 \quad 135 \quad -270 \quad -315]\end{aligned} \tag{57}$$

$$\begin{aligned}&\text{angle}(\mathbf{X_4}) - \text{angle}(\mathbf{X_2})\\ &\quad = [0 \quad -90 \quad -180 \quad -270 \quad 0 \quad -90 \quad -180 \quad -270].\end{aligned} \tag{58}$$

This is exactly the same as the original DFT.

Then, we use the equation as below to calculate the difference between $X_2(m)$ (the ITFT or approximated DFT of $x_2(n)$) and $X_3(m), X_4(m)$ (the ITFT or approximated DFT of $x_3(n), x_4(n)$):

$$\text{dif} = \sqrt{\sum_{m=0}^{N-1} |X_{3\,\text{or}\,4}(m) - X_2(m)|^2 \Big/ \sum_{m=0}^{N-1} |X_2(m)|^2}. \tag{59}$$

Then

$$\begin{aligned}&\text{for Fig. 10(a): dif} = 6.3832 \cdot 10^{-3}\\ &\text{for Fig. 10(b): dif} = 0 \quad \text{(approximated DFT)}\\ &\text{Fig. 10(c): dif} = 2.3401 \cdot 10^{-3}\\ &\text{for Fig. 10(d): dif} = 0 \quad \text{(complete ITFT).}\end{aligned} \tag{60}$$

Thus, the complete ITFT will have more excellent displacement-invariant property than the approximated DFT, especially when the bits are fewer.

D. Complete Integer Fourier Transform of 6-Points

Because $6 \neq 2^k$, it seems impossible to avoid the real numbers multiplication for 6-point DFT, but in fact, we can also derive the 6-point complete ITFT.

The original 6-point DFT is

$$F(m,n) = \exp(-j \cdot m \cdot n \cdot \pi/3) \quad \text{where } m,n \in [0,1,\ldots 5]. \quad (61)$$

By a similar process as the derivation of the 8-point ITFT, we obtain the prototype of the 6-point ITFT as in (62), shown at the bottom of the page. For the prototype as in (92), it is impossible to derive the 6-point complete ITFT; therefore, we modify the prototype a little as in (63) at the bottom of the page. The prototype for the inverse transform can be set as in (64), shown at the bottom of the page. From the orthogonality, we obtain the constraints as

$$\begin{aligned} &(1)\ d_1a_2 = 2d_2b_2, \quad (2)\ d_3a_1 = 2d_4b_1 \\ &(3)\ a_1a_2 + 2b_1b_2 - 2c_1c_2 = 0. \end{aligned} \quad (65)$$

From the requirement of (12), we obtain the constraints as

$$\begin{aligned} &(4)\ d_1d_3 + 2d_2d_4 = 2^k \\ &(5)\ a_1a_2 + 2b_1b_2 + 2c_1c_2 = 2^h. \end{aligned} \quad (66)$$

From the inequality relations of the entries of the original 6-point DFT matrix, we obtain the inequality relations as

$$(6)\ b_1 \le c_1 \le a_1, \quad (7)\ b_2 \le c_2 \le a_2. \quad (67)$$

Besides, to make (93) and (94) similar to (92), we also impose the following inequality constraints:

$$(8)\ d_2/d_1 \approx 1, \quad (9)\ d_4/d_3 \approx 1. \quad (68)$$

There are a total of five equality constraints and four inequality constraints for ten parameters.

From the constraints 3, 5, we can convert these constraints as

$$(3')\ c_1c_2 = 2^{h-2}, \quad (5')\ a_1a_2 + 2b_1b_2 = 2^{h-1} \quad (69)$$

and from the constraints 1, 2, $a_2 = 2(d_2/d_1)b_2, a_1 = 2(d_4/d_3)b_1$. We can substitute them into the new constraint 5', and we obtain

$$\left(2\frac{d_2d_4}{d_1d_3} + 1\right) b_1b_2 = 2^{h-2}. \quad (70)$$

Together with constraint 4, we obtain

$$b_1b_2 = 2^{h-2-k} \cdot d_1d_3. \quad (71)$$

Thus, to find the values of the parameters, we can follow the process as follows.

1) Choose the value of d_1, d_2, and $d_1/d_2 \approx 1$ must be satisfied.
2) Find the value of d_3, d_4 such that $d_3/d_4 \approx 1$, and
$$d_1d_3 + 2d_2d_4 = 2^k, \quad \text{where } k \text{ is integer.} \quad (72)$$
3) Choose b_1, b_2 as
$$b_1 = 2^m \cdot d_3, \quad b_2 = 2^n \cdot d_1 \quad \text{where } m,n \text{ are integer numbers.} \quad (73)$$
4) Then, values of a_1, a_2 can be calculated as
$$a_1 = 2^{m+1} \cdot d_4, \quad a_2 = 2^{n+1} \cdot d_2 \quad m,n \text{ the same as above.} \quad (74)$$

$$\mathbf{FI_p} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ a_1 & b_1 - jc_1 & -b_1 - jc_1 & -a_1 & -b_1 + jc_1 & b_1 + jc_1 \\ a_1 & -b_1 - jc_1 & -b_1 + jc_1 & a_1 & -b_1 - jc_1 & -b_1 + jc_1 \\ 1 & -1 & 1 & -1 & 1 & -1 \\ a_1 & -b_1 + jc_1 & -b_1 - jc_1 & a_1 & -b_1 + jc_1 & -b_1 - jc_1 \\ a_1 & b_1 + jc_1 & -b_1 + jc_1 & -a_1 & -b_1 - jc_1 & b_1 - jc_1 \end{bmatrix}. \quad (62)$$

$$\mathbf{FI_p} = \begin{bmatrix} d_1 & d_2 & d_2 & d_1 & d_2 & d_2 \\ a_1 & b_1 - jc_1 & -b_1 - jc_1 & -a_1 & -b_1 + jc_1 & b_1 + jc_1 \\ a_1 & -b_1 - jc_1 & -b_1 + jc_1 & a_1 & -b_1 - jc_1 & -b_1 + jc_1 \\ d_1 & -d_2 & d_2 & -d_1 & d_2 & -d_2 \\ a_1 & -b_1 + jc_1 & -b_1 - jc_1 & a_1 & -b_1 + jc_1 & -b_1 - jc_1 \\ a_1 & b_1 + jc_1 & -b_1 + jc_1 & -a_1 & -b_1 - jc_1 & b_1 - jc_1 \end{bmatrix}. \quad (63)$$

$$\mathbf{IFI_p} = \begin{bmatrix} d_3 & d_4 & d_4 & d_3 & d_4 & d_4 \\ a_2 & b_2 - jc_2 & -b_2 - jc_2 & -a_2 & -b_2 + jc_2 & b_2 + jc_2 \\ a_2 & -b_2 - jc_2 & -b_2 + jc_2 & a_2 & -b_2 - jc_2 & -b_2 + jc_2 \\ d_3 & -d_4 & d_4 & -d_3 & d_4 & -d_4 \\ a_2 & -b_2 + jc_2 & -b_2 - jc_2 & a_2 & -b_2 + jc_2 & -b_2 - jc_2 \\ a_2 & b_2 + jc_2 & -b_2 + jc_2 & -a_2 & -b_2 - jc_2 & b_2 - jc_2 \end{bmatrix}. \quad (64)$$

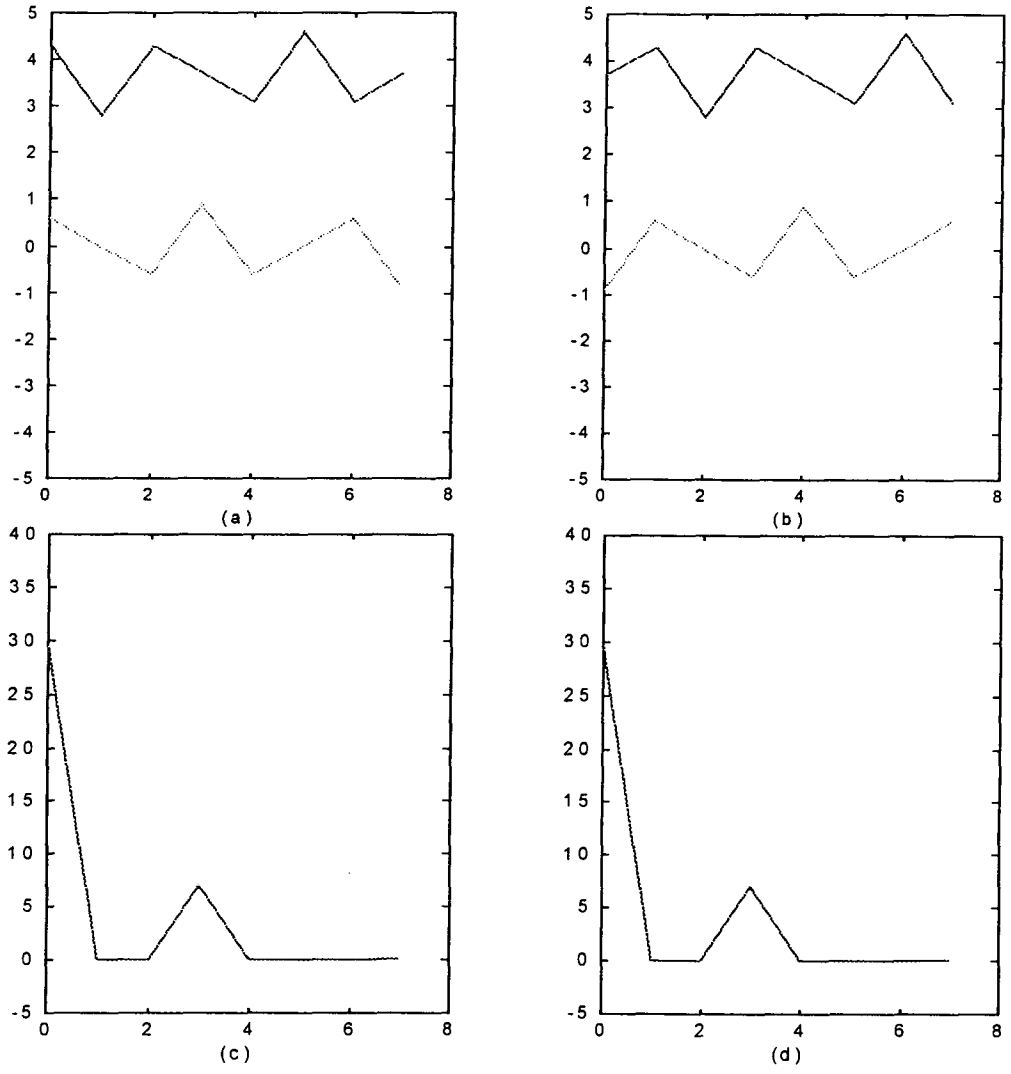


Fig. 9. Space-invariant property. (a), (b) Shifted input. (c), (d) Amplitude of the transform results of original ITFT.

5) Find the values of c_3, c_4 such that

$$c_1 c_2 = 2^{m+n+k}, \quad m, n, k \text{ are the same as above.} \tag{75}$$

In addition, the inequality constraints 6 and 7 must be satisfied.

We give some examples of the values of parameters as follows.

a) $d_1 = 1, d_2 = 1, d_3 = 6, d_4 = 5, a_1 = 5, b_1 = 3, c_1 = 4, a_2 = 2, b_2 = 1, c_2 = 2$.
This would be the smallest possible choice of the parameters. If

$$\sum_{n=0}^{5} FI_P(m,n) \cdot IFI_P(m,n) = D_m \qquad m = 0,1,2,3,4 \tag{76}$$

then $D_0 = D_3 = 2^5, D_1 = D_2 = D_4 = D_5 = 2^6$.

b) $d_1 = 9, d_2 = 10, d_3 = 36, d_4 = 35, a_1 = 35, b_1 = 18, c_1 = 32, a_2 = 20, b_2 = 9, c_2 = 16$.
In this case, the 6-point complete ITFT will be more similar to the original 6-point DFT. If D_m is defined as (107), then $D_0 = D_3 = 2^{11}, D_1 = D_2 = D_4 = D_5 = 2^{12}$.

We can implement the forward 6–point complete ITFT as Fig. 11. There are ten integer multiplication operations required. In contrast, to implement the original 6-point DFT, we need eight real multiplication operations. If we consider the overall system, that is, include the forward and inverse transform, then the numbers of multiplication operations required are as

original 6-point DFT:

22 real number multiplication operations.

(eight for forward, eight for inverse, six for normalization)

complete 6-point ITFT:

26 fixed-point multiplication operations.

(ten for forward, ten for inverse, six for normalization).

It is obvious that the complete integer 6-points DFT is much more efficient.

IV. Integer Hartley Transform

The discrete Hartley transform (DHT) [5] is defined as

$$H(m,n) = \text{cas}(mn\pi/N) \quad m,n \in [0,1,\ldots N-1] \tag{77}$$

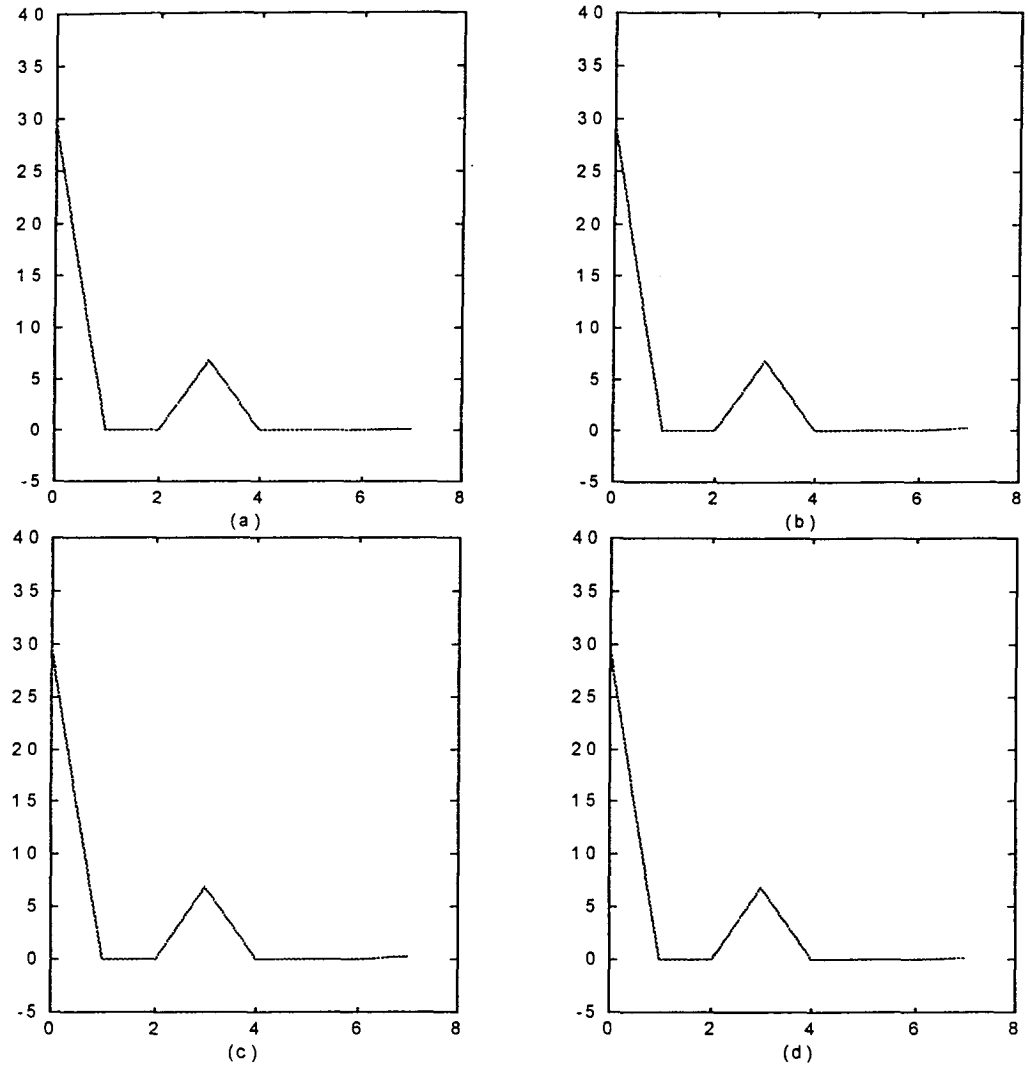


Fig. 10. Space-invariant property. (a), (b) Amplitude of the transform results of approximated DFT. (c), (d) Amplitude of the transform results of complete ITFT.

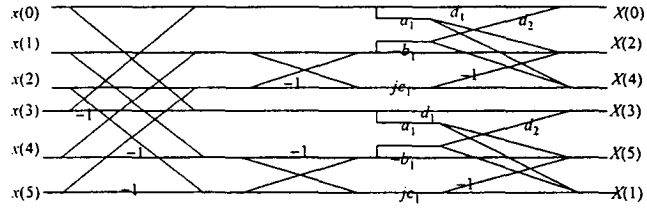


Fig. 11. Implementation of 6-point integer Fourier transform.

where $\text{cas}(x) = \cos(x) + \sin(x)$. The transform matrix of the DHT is just the summation of the real part and imaginary part of DFT

$$H(m,n) = \text{Re}(F(m,n)) - \text{Im}(F(m,n)). \tag{78}$$

The basis vectors of the DHT are orthogonal

$$\sum_{n=0}^{N-1} H(m,n) \cdot H(k,n) = N \cdot \delta_{m,k}.$$

The applications of the DHT are similar to the applications of the DFT, and the most important advantage of the DHT is when the input is pure real; then, the output is also pure real.

In this subsection, we will derive the integer Hartley transform (ITHT). We will not follow the process introduced in Section II. Instead, we will just derive the near-complete ITHT as

$$HI(m,n) = \text{Re}(FI(m,n)) - \text{Im}(FI(m,n)) \tag{79}$$

where $FI(m,n)$ is the near complete ITFT. We can just derive the complete ITHT as

$$\begin{aligned} &\text{forward ITHT: } HI(m,n) \\ &\quad = \text{Re}(FI(m,n)) - \text{Im}(FI(m,n)) \qquad (80) \\ &\text{inverse ITHT: } IHI(m,n) \\ &\quad = \text{Re}(IFI(m,n)) - \text{Im}(IFI(m,n)) \qquad (81) \end{aligned}$$

where $FI(m,n)$ is the forward complete ITFT, and $IFI(m,n)$ is the inverse complete ITFT. Here, we have constrained that the ITFT we derive should keep the conjugation system property

$$\begin{aligned} FI(m,n) &= \overline{FI(N-m,n)} \\ IFI(m,n) &= \overline{IFI(N-m,n)}. \end{aligned} \tag{82}$$

The goal of the integer transform is to keep the abilities of the original transform, but no real number multiplication operation is required. If we define the ITHT as (79) or (80) and (81), then the abilities of the DHT will be kept. That is, as the original

DHT, the ITHT defined as (79) or (80) and (81) will be another form of the ITFT, but for the real input, the output will also be real. Therefore, the ITHT may replace the ITFT for processing the real signals.

We then prove that the ITHT is reversible. We prove that the complete ITHT is reversible. From (80) and (81)

$$\begin{aligned}&\sum_{m=0}^{N-1} HI(m,n)IHI(m,k)\\&\quad=\sum_{m=0}^{N-1}\mathrm{Re}(FI(m,n))\mathrm{Re}(IFI(m,k))\\&\qquad+\sum_{m=0}^{N-1}\mathrm{Im}(FI(m,n))\mathrm{Im}(IFI(m,k))\\&\qquad-\sum_{m=0}^{N-1}\mathrm{Im}(FI(m,n))\mathrm{Re}(IFI(m,k))\\&\qquad-\sum_{m=0}^{N-1}\mathrm{Re}(FI(m,n))\mathrm{Im}(IFI(m,k)).\end{aligned}$$

Since the basis vectors of ITFT are orthogonal, we have

$$\begin{aligned}&\sum_{m=0}^{N-1} FI(m,n)\overline{IFI(m,k)}\\&\quad=\sum_{m=0}^{N-1}\mathrm{Re}(FI(m,n))\mathrm{Re}(IFI(m,k))\\&\qquad+\sum_{m=0}^{N-1}\mathrm{Im}(FI(m,n))\mathrm{Im}(IFI(m,k))\\&\qquad+j\sum_{m=0}^{N-1}\mathrm{Im}(FI(m,n))\mathrm{Re}(IFI(m,k))\\&\qquad-j\sum_{m=0}^{N-1}\mathrm{Re}(FI(m,n))\mathrm{Im}(IFI(m,k))\\&\quad=N\cdot\delta_{n,k}.\end{aligned}\tag{83}$$

Thus

$$\begin{aligned}&\sum_{m=0}^{N-1}\mathrm{Re}(FI(m,n))\mathrm{Re}(IFI(m,k))\\&\qquad+\sum_{m=0}^{N-1}\mathrm{Im}(FI(m,n))\mathrm{Im}(IFI(m,k))\\&\quad=N\cdot\delta_{n,k}\end{aligned}\tag{84}$$

$$\begin{aligned}&\sum_{m=0}^{N-1}\mathrm{Im}(FI(m,n))\mathrm{Re}(IFI(m,k))\\&\qquad-\sum_{m=0}^{N-1}\mathrm{Re}(FI(m,n))\mathrm{Im}(IFI(m,k))\\&\quad=0\end{aligned}\tag{85}$$

and (83) becomes

$$\begin{aligned}&\sum_{m=0}^{N-1} HI(m,n)IHI(m,k)\\&\quad=N\cdot\delta_{n,k}-2\sum_{m=0}^{N-1}\mathrm{Im}(FI(m,n))\mathrm{Re}(IFI(m,k)).\end{aligned}\tag{86}$$

Then since, for the ITFT, we have kept the conjugation symmetry property of (82) so that

$$\begin{aligned}\mathrm{Re}(FI(m,n))&=\mathrm{Re}(FI(N-m,n))\\\mathrm{Im}(FI(m,n))&=-\mathrm{Im}(FI(N-m,n))\end{aligned}\tag{87}$$

$$\begin{aligned}\mathrm{Re}(IFI(m,n))&=\mathrm{Re}(IFI(N-m,n))\\\mathrm{Im}(IFI(m,n))&=-\mathrm{Im}(IFI(N-m,n))\end{aligned}\tag{88}$$

we have

$$\begin{aligned}&\sum_{m=0}^{N-1}\mathrm{Im}(FI(m,n))\mathrm{Re}(IFI(m,k))\\&\quad=\sum_{m=0}^{[N/2]-1}\mathrm{Im}(FI(m,n))\mathrm{Re}(IFI(m,k))\\&\qquad+\sum_{m=0}^{[N/2]-1}\mathrm{Im}(FI(N-m,n))\mathrm{Re}(IFI(N-m,k))\\&\quad=\sum_{m=0}^{[N/2]-1}\mathrm{Im}(FI(m,n))\mathrm{Re}(IFI(m,k))\\&\qquad-\sum_{m=0}^{[N/2]-1}\mathrm{Im}(FI(m,n))\mathrm{Re}(IFI(m,k))=0\end{aligned}\tag{89}$$

and (86) becomes

$$\sum_{m=0}^{N-1} HI(m,n)IHI(m,k)=N\cdot\delta_{n,k}.\tag{90}$$

From (27) and (79), we can derive the prototype of the transform matrix of 8-point near-complete ITHT as (91), shown at the bottom of the page. The constraints are the same as (28).

$$\mathbf{HI_P}=\begin{bmatrix}1&1&1&1&1&1&1&1\\a_1&2a_2&a_1&0&-a_1&-2a_2&-a_1&0\\1&1&-1&-1&1&1&-1&-1\\c_1&0&-c_1&2c_2&-c_1&0&c_1&-2c_2\\1&-1&1&-1&1&-1&1&-1\\c_1&-2c_2&c_1&0&-c_1&2c_2&-c_1&0\\1&-1&-1&1&1&-1&-1&1\\a_1&0&-a_1&-2a_2&-a_1&0&a_1&2a_2\end{bmatrix}.\tag{91}$$

Similarly to the near-complete ITFT, we can choose the values of $\{a_1, a_2, c_1, c_2\}$ as the values listed in Table II.

For the 8-point complete ITHT, the prototype of the forward transform matrix is the same as (91). From (31) and (81), we can derive the prototype of the inverse transform matrix of the 8-point complete ITHT as (92), shown at the bottom of the page. The values of $\{a_1, a_2, c_1, c_2, a_3, a_4, c_3, c_4\}$ must also satisfy the constraints of (32)–(34). We can also choose the values of the $\{a_1, a_2, c_1, c_2, a_3, a_4, c_3, c_4\}$ as in Table III.

We draw the implementation of the forward, inverse ITHT as Figs. 12 and 13.

For the original DHT and complete ITHT, the multiplication operations we required are

original DHT:

four real number, eight fixed-point multiplication operations

complete ITHT:

20 fixed-point multiplication operations.

Although there are fewer multiplication operations for the original DHT, four of the multiplication operations are real numbers. For the complete ITHT, although there are more fixed-point multiplication operations, there are no real number multiplication operations. Thus, the complete integer Hartley transform will be simpler and faster.

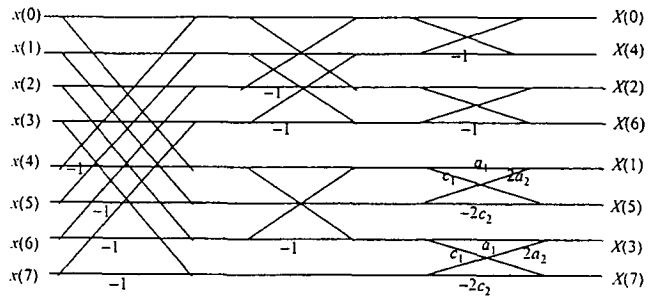


Fig. 12. Implementation of the forward 8-point complete integer Hartley transform.

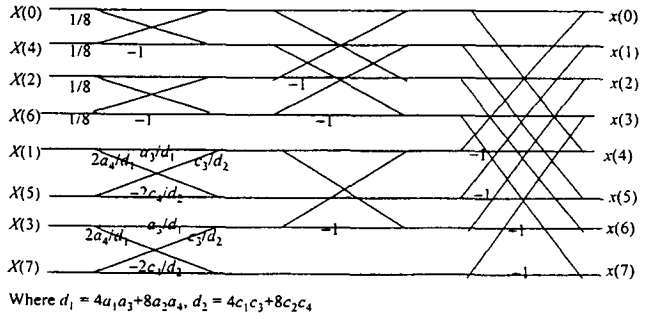


Fig. 13. Implementation of the inverse 8-point complete integer Hartley transform.

V. Integer Sine and Cosine Transforms

A. Integer Sine Transform of 8-Points

There are many types of DST [6]. In [2], they have derived the 8-point integer: even sine-1, even sine-2, even sine-3, odd sine-1, and odd sine-2 transforms. We describe the 8-point integer even sine-1 transform. Others can be seen in [2].

The original 8-point discrete even sine-1 transform is defined as

$$S(m,n) = \sin((m+1)\cdot(n+1)\pi/9) \quad \text{where } m,n \in [0,1,\ldots 7]. \tag{93}$$

In [2], they have derived the prototype of the 8-point integer even sine-1 transform (ITST-1) as

$$\mathbf{SI_P} = \begin{bmatrix} a_1 & a_2 & a_3 & a_4 & a_4 & a_3 & a_2 & a_1 \\ a_2 & a_4 & a_3 & a_1 & -a_1 & -a_3 & -a_4 & -a_2 \\ 1 & 1 & 0 & -1 & -1 & 0 & 1 & 1 \\ a_4 & a_1 & -a_3 & -a_2 & a_2 & a_3 & -a_1 & -a_4 \\ a_4 & -a_1 & -a_3 & a_2 & a_2 & -a_3 & -a_1 & a_4 \\ 1 & -1 & 0 & 1 & -1 & 0 & 1 & -1 \\ a_2 & -a_4 & a_3 & -a_1 & -a_1 & a_3 & -a_4 & a_2 \\ a_1 & -a_2 & a_3 & -a_4 & a_4 & -a_3 & a_2 & -a_1 \end{bmatrix}. \tag{94}$$

There are four unknowns a_1, a_2, a_3, a_4. The constraints of the parameters are

$$(1)\ a_1 + a_2 = a_4, \quad (2)\ a_1a_4 + a_2a_4 = a_1a_2 + a_3^2$$
$$(3)\ a_1 \le a_2 \le a_3 \le a_4. \tag{95}$$

There are many possible choices of the values of these unknowns (see [2, Tab. II]). One example is $\{a_1 = 3, a_2 = 5, a_3 = 7, a_4 = 8\}$ (This is the example where the parameters are smallest).

$$\mathbf{IHI_P} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ a_3 & 2a_4 & a_3 & 0 & -a_3 & -2a_4 & -a_3 & 0 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ c_3 & 0 & -c_3 & 2c_4 & -c_3 & 0 & c_3 & -2c_4 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ c_3 & -2c_4 & c_3 & 0 & -c_3 & 2c_4 & -c_3 & 0 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ a_3 & 0 & -a_3 & -2a_4 & -a_3 & 0 & a_3 & 2a_4 \end{bmatrix}. \tag{92}$$

The implementation of the 8-point ITST-1 has not been discussed in the previous works. We draw the implementation of the 8-point ITST-1 as Fig. 14.

The implementation of the 8-point ITST-1 requires a total of eight fixed-point multiplication operations. Since the 8-point ITST-1 introduced above is a near-complete integer transform, there are still eight real number multiplication operations required for the inverse transform. It is very hard to derive the complete integer sine transform.

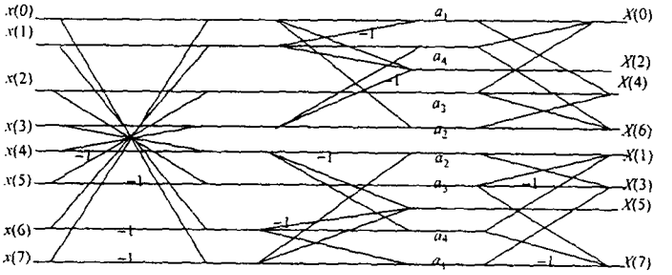


Fig. 14. Implementation of the 8-point ITST-1.

B. Integer Cosine Transform of 8-Points

In [1], they have derived the integer cosine transform (ITCT), which approximates the 8-point discrete cosine transform

$$C(m,n) = \cos(m \cdot (n+0.5)\pi/8) \quad \text{where } m,n \in [0,1,\ldots 7]. \quad (96)$$

The prototype matrix of ITCT is

$$\mathbf{CI_P} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ a_1 & a_2 & a_3 & a_4 & -a_4 & -a_3 & -a_2 & -a_1 \\ b_1 & b_2 & -b_2 & -b_1 & -b_1 & -b_2 & b_2 & b_1 \\ a_2 & -a_4 & -a_1 & -a_3 & a_3 & a_1 & a_4 & -a_2 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ a_3 & -a_1 & a_4 & a_2 & -a_2 & -a_4 & a_1 & -a_3 \\ b_2 & -b_1 & b_1 & -b_2 & -b_2 & b_1 & -b_1 & b_2 \\ a_4 & -a_3 & a_2 & -a_1 & a_1 & -a_2 & a_3 & -a_4 \end{bmatrix}. \quad (97)$$

The constraints are

$$(1)\ a_1a_2 - a_2a_4 - a_1a_3 - a_3a_4 = 0$$
$$(2)\ a_1 \geq a_2 \geq a_3 \geq a_4, \quad (3)\ b_1 \geq b_2. \quad (98)$$

Other details about ITCT can be seen in [1] and [2]. The 8-point ITCT can be implemented as Fig. 15.

In [1], they have also discussed the implementation of the 8-point integer cosine transform. In the above, we use an alternative method to implement it, and only the fixed-point multiplication operations are required. As in [1], we set $b_2 = 1$. Here, we try to make the number of multiplication operations as small as possible. If we implement the 8-point ITCT as in Fig. 15, there are 12 integer multiplication operations required for both the forward and inverse transform, but there are still six real number multiplication operations required for the inverse transform.

After the 8-point integer cosine transform (ITCT) defined in (97) has been derived in [1], in [3] and [4], the 16-point ITCT has been derived. In [2], the 8-point integer even cosine-2, odd cosine-1 transforms have been derived, and the ITCT defined as (97) is treated as the 8-point integer even cosine-1 transform.

We can also derive the complete integer cosine transform and do not require any real number multiplication operations, no matter what the forward or inverse transform. We also use (97) as the prototype for the forward ITCT. For the inverse ITCT,

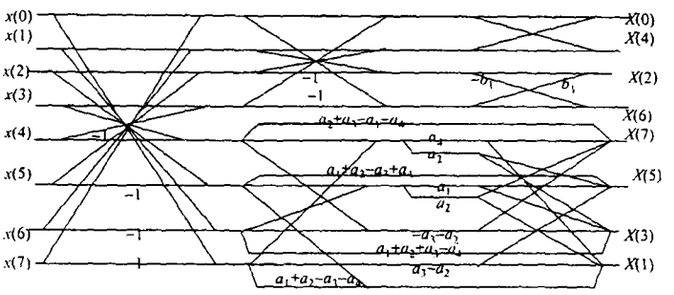


Fig. 15. Implementation of 8-point integer cosine transform.

we also use the same form of prototype, but the parameters are changed as in

$$\mathbf{ICI_P} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ c_1 & c_2 & c_3 & c_4 & -c_4 & -c_3 & -c_2 & -c_1 \\ d_1 & d_2 & -d_2 & -d_1 & -d_1 & -d_2 & d_2 & d_1 \\ c_2 & -c_4 & -c_1 & -c_3 & c_3 & c_1 & c_4 & -c_2 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ c_3 & -c_1 & c_4 & c_2 & -c_2 & -c_4 & c_1 & -c_3 \\ d_2 & -d_1 & d_1 & -d_2 & -d_2 & d_1 & -d_1 & d_2 \\ c_4 & -c_3 & c_2 & -c_1 & c_1 & -c_2 & c_3 & -c_4 \end{bmatrix}. \quad (99)$$

Thus, there are a total of 12 parameters $\{a_1, a_2, a_3, a_4, b_1, b_2\}$ $\{c_1, c_2, c_3, c_4, b_3, b_4\}$. Then, from the requirement of dual orthogonality, we obtain the constraints as

$$(1)\ a_1c_2 - a_2c_4 - a_3c_1 - a_4c_3 = 0$$
$$(2)\ a_2c_1 - a_4c_2 - a_1c_3 - a_3c_4 = 0. \quad (100)$$

From the requirement that the inner product must be the value of 2^k, we obtain

$$(3)\ a_1c_1 + a_2c_2 + a_3c_3 + a_4c_4 = 2^k$$
$$(4)\ b_1b_3 + b_2b_4 = 2^h \quad (101)$$

where k, h are integer numbers. We keep the two inequality constraints in (98) as

$$(5)\ a_1 \geq a_2 \geq a_3 \geq a_4, \quad (6)\ b_1 \geq b_2. \quad (102)$$

For flexibility, we omit the inequality constraints required for $\{c_1, c_2, c_3, c_4, b_3, b_4\}$. Thus, we obtain four equality constraints and two inequality constraints.

Although there are 12 parameters and four constraints, it seems that there would be many possible solutions for the

values of the parameters. However, these solutions are hard to find. We introduce a special method to find the solutions. We set $c_4 = 0$, and constraints 1, 2, and 3 become

$$\begin{bmatrix} -a_3 & a_1 & -a_4 \\ a_2 & -a_4 & -a_1 \\ a_1 & a_2 & a_3 \end{bmatrix} \cdot \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 2^k \end{bmatrix}$$
where k is an integer. (103)

Therefore, we want

$$\det \begin{bmatrix} -a_3 & a_1 & -a_4 \\ a_2 & -a_4 & -a_1 \\ a_1 & a_2 & a_3 \end{bmatrix} = \pm 2^n$$
where n is an integer. (104)

Then, we can assure the solution of $\{c_1, c_2, c_3\}$ will all be the values of the form as $D \cdot 2^{-h}$, where D, h are integer numbers. We then want

$$-a_1^3 - a_2^2 a_4 + a_3^2 a_4 - 2a_1 a_2 a_3 - a_1 a_4^2 = \pm 2^n$$
where n is an integer. (105)

The values of $\{a_1, a_2, a_3, a_4\}$ that satisfy (105) are hard to find. Therefore, we suppose

$$a_1 : a_3 = a_3 : a_4, \quad \text{i.e., } a_1 = r^2 a_4, \quad a_3 = r \cdot a_4. \tag{106}$$

Then, (38) becomes

$$-r^6 a_4^3 - a_2^2 a_4 - 2r^3 a_2 a_4 = \pm 2^n$$
where n is an integer. (107)

We further suppose that $a_4 = 2^s$; therefore, the requirement becomes

$$r^3 a_4 + a_2 = 2^m$$
where $m = (n - s)/2$ and must be an integer. (108)

Thus, we can do the following process to choose the parameters.

a) First, find the value of $\{b_1, b_2, b_3, b_4\}$ to satisfy constraint 4. Since, for the original cosine transform, $b_1 = 0.9238, b_2 = 0.3827$, and $b_1 : b_2 = 2.4142 : 1$, we can choose $b_1 = 5$ and $b_2 = 2$ and substitute them into constraint 4 to find the values of b_3, b_4.

b) Set $c_4 = 0$ and $a_4 = 2^s$, where s is an integer number.

c) Choose $r (r > 1)$, and set a_1, a_3 as

$$a_1 = r^2 a_4, \quad a_3 = r \cdot a_4, \quad a_1, a_3 \text{ must be an integer.} \tag{109}$$

d) Calculate a_2 from

$$a_2 = 2^m - r^3 a_4, \quad m \text{ is an integer.} \tag{110}$$

e) Check whether $a_1 \geq a_2 \geq a_3$. If not, go back to step b).

f) Solve c_1, c_2, c_3 from (103).

Then, we can obtain the values of all the parameters. However, we must be careful to satisfy the inequality constraints.

We list some other possible choices of the values of parameters as Table IV.

We note that the forward ITCT can be implemented as Fig. 15. The inverse ITCT can also be implemented in Fig. 15, but the direction is reversed with some multiplication operations at the input. There are 12 fixed-point multiplication operations both for the forward and inverse transform. There are eight multiplication operations for D_m^{-1}, where D_m is defined as

$$\sum_{n=0}^{N-1} CI(m,n) \cdot ICI(m,n) = D_m. \tag{111}$$

Four of the multiplications of D_m can be absorbed in the multiplication operations of the inverse transform; therefore, we need a total of 28 fixed-point multiplication operations for the overall system.

TABLE IV
SOME POSSIBLE CHOICES OF THE VALUES OF THE PARAMETERS OF THE 8-POINT COMPLETE ITCT

a_1	a_2	a_3	a_4	b_1	b_2	b_3	b_4	c_1	c_2	c_3	c_4
81/16	295/64	9/4	1	5	2	6	1	6817/16	256	21591/64	0
49/16	169/64	7/4	1	5	2	22	9	2657/16	128	6489/64	0
121/64	717/512	11/8	1	12	5	9	4	149896	131072	41701	0
361/64	1333/512	19/8	1	17	7	13	5	1075336	524288	403389	0

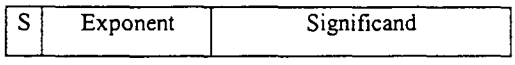


Fig. 16. Data structure including the sign, mantissa, and exponent terms.

VI. DISCUSSION ABOUT THE INTEGER TRANSFORMS

A. Advantages of the Integer Transform over Direct Approximation

In this paper, we use the integer transforms to approximate the noninteger transforms. In fact, we can also approximate the noninteger transform directly. For example, we can use the following data structure, including the mantissa and exponent terms to simulate real number operations by the fixed-point operations numerically [10] as in Fig. 16. In Fig. 16, "S" is the sign of the number, "Exponent" is the exponent term, and "Significand" is the mantissa term. For example, when we use the above structure to express the real value of $-C$, we can first approximate $-C$ as the form of

$$-C \approx -B \cdot 2^h$$
B is a positive integer number, h is an integer number. (112)

Then, we can use "$S = 1$" to express the minus sign, use "Exponent $= h$" to express the exponent of 2, and use "Significand $= B$" to express the mantissa term. After using the above data structure, we can use the fixed-point operations to simulate the real number operations.

We may ask what are the advantages of the integer transform over the direct approximation method. We list the advantages below. We will just compare the direct approximation method with the near-complete integer transform.

1) The basis vectors of the integer transform are orthogonal.

We note that in Section II-A, we have discussed the four requirements for the integer transform. We note that requirements a)–c) are also necessary for the direct approximation method. The key difference between the integer transform and the direct approximation method is requirement d). That is, for the complete integer transform, the basis vectors of the forward transform is dual orthogonal to the basis vectors of the inverse transform. However, for the direct approximation method, the basis

vectors of the transform matrix may not be orthogonal. This difference is very important, and many of the advantages of the integer transform come from it.

2) The integer transform is reversible, but the transform obtained by direct approximation will not be reversible.

Since the complete integer transform is dual orthogonal, i.e., the basis vectors of the forward transform matrix $\mathbf{B}$ are dual orthogonal to the basis vectors of some matrix $\mathbf{E}$, then $\mathbf{E}^{\mathbf{H}}\mathbf{D}^{-1}$, where $\mathbf{D}$, which is defined as (12) and (14), is the inverse of the forward transform.

However, for the direct approximation method, since the basis vectors are not orthogonal, if the forward transform matrix is $\mathbf{K}$, then the inverse of $\mathbf{K}$ (i.e., $\mathbf{K}^{-1}$) will not be as simple as the inverse of the integer transform. Furthermore, the entries of $\mathbf{K}^{-1}$ can usually just be approximated and not explicitly expressed as the data structure as Fig. 16. Thus, although the basis vectors of the original transform (which are denoted by $\mathbf{F}$) are orthogonal, when we use the data structure as Fig. 16, we will use a transform matrix $\mathbf{K}$ to approximate $\mathbf{F}$ and use another transform matrix $\mathbf{R}$ to approximate $\mathbf{F}^{-1}$. Since

$$(\mathbf{F}^{-1})\cdot\mathbf{F}=\mathbf{I},\quad \mathbf{K}\approx\mathbf{F},\quad \mathbf{R}\approx\mathbf{F}^{-1}\quad \mathbf{R}\cdot\mathbf{K}\neq\mathbf{I} \tag{113}$$

we use the data structure as Fig. 16 to approximate a reversible transform; then, the approximated transform would not be reversible.

The reversible property is very important, especially for the applications such as the filter design, data compression, and other applications that must do both the forward and inverse transforms. Although, for the directly approximation method, if the number of the bits are sufficient, then the approximated transform will be almost reversible. However, for the integer transform, we can use far fewer bits to obtain the reversible transform.

3) The integer transform can use fewer bits to obtain the accurate approximation results.

This is because when we use the data structure as in Fig. 16, some bits must be used for the exponent. Besides, the length of the data structure in Fig. 16 is usually fixed to 16 or 32 bits [10], but for the integer transform, the approximation results are sufficient, even when we use fewer bits. For example, for the two experiments about the ITFT in Section III-C, the largest parameter we choose is $a_3 = 18$, and it only requires 5 bits. Other parameters require 4 bits or less. From the experiment results, we still get accurate approximation results.

4) For the integer transform, when the input is expressed as the fixed-point representation, we do not need to convert it as the real number representation in Fig. 16. We can process it without any real number processor during the overall system.

5) More properties will be kept for the integer transform.

For example, the power preservation property will be kept for the complete ITFT (see Section III-C), but this property will not be kept when we approximate the DFT

In conclusion, even when we use fewer bits, the integer transforms can still approximate the original transforms well and retain the properties and performance of the original transform. If we use enough bits, then using the direct approximation method will also have the advantages described above. In this case, the advantages of the integer transform may not be so obvious as the case where we use fewer bits.

B. Which Set of Parameters Is the Best

From the discussion about the integer sine, cosine, Hartley, and Fourier transforms, we find that the number of the equality constraints is always less than the number of the parameters. Thus, there are always infinite possible choices for the parameters. Thus, we have a question: Among these choices of parameters, which one is the best? Here, we will discuss this question.

To discuss which set of the parameters is the best, we mainly consider two aspects.

1) *For the implementation, we want the parameters to be as small as possible.* As in Fig. 1, for binary implementation, the entire number is decomposed as

$$C = a_0 + a_1\cdot 2 + a_2\cdot 2^2 + a_3\cdot 2^3\cdots \quad \text{where } a_r = 0, \pm 1. \tag{114}$$

Thus, if the parameters we choose are small, then the number of a_n such that $a_n \neq 0$ will usually be less. Then, for the binary implementation, the number of binary-addition operations and binary-shifting operations will be less.

2) *For the performance, we want the ratios of the entries for each row of the integer transform matrix to be approximated to those of the original transform matrix.*

For example, for the 8-point ITFT that has the prototype (27), from (15), the original 8-point DFT will have the parameters a_1, a_2 as

$$a_1 = 1 \quad a_2 = 1.414. \tag{115}$$

Therefore, if we want the performance of the 8-point ITFT to be similar to the original, then we want

$$a_1 : a_2 \approx 1 : 1.414. \tag{116}$$

For the complete integer transform, it is difficult for the parameters of the forward and inverse transforms to have the ratios approximated to the original at the same time. In this case, the parameters of the forward transform will have higher priority. If the parameters of the forward transform have ratios similar to the original transform, then the transform result will be similar to the original transform.

In [1], the tradeoff between how large the values of unknowns we choose and the performance for the integer cosine transform is discussed. In fact, this tradeoff also exists for other integer transforms. If the unknowns we choose are very large, than it is possible for the integer transforms to better approximate the original transforms and make the performance better; however, the implementation will be tough. On the other hand, if the values of unknown are too small, then the implementation is

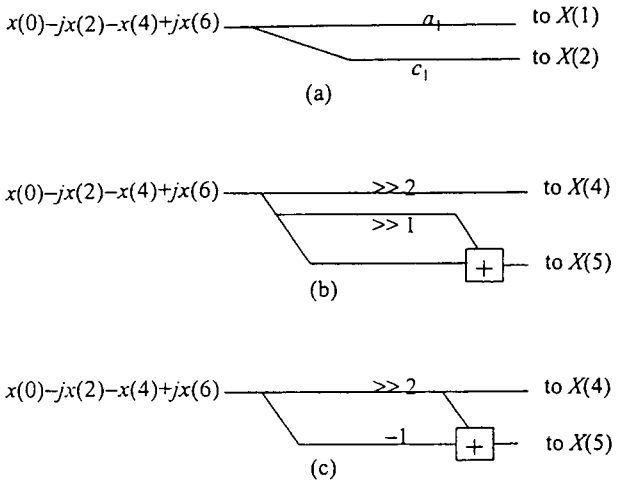


Fig. 17. (a) Implementation of 8-point ITFT (decimation-in-frequency) between the fifth position of the second stage and the fifth and sixth position of the third stage. (b) Binary implementation of Fig. 17(a) when $a_1 = 4$ and $c_1 = 3$. (c) Simplification of (b).

To find the optimal choices for the parameters is hard work. We suggest a way to find the better choices for the parameters.

1) First, find the ratios between the parameters for each row of the original transform matrix.
2) Then, assign the smaller integer values for the parameters such that the ratios are approximated well, and all the constraints for the parameters are satisfied.

   Then, although we may not obtain the optimal choices for the parameters, the integer transform we obtain will perform better with simpler implementation.

C. Some Special Methods to Implement Further More Efficient

Suppose $C = D \cdot 2^h$, where D, h are integer numbers; then if we want to multiply a number C, we can first decompose C as the linear combination of the powers of 2:

$$C = \sum_r a_r \cdot 2^r \quad \text{where } a_r = 0, \pm 1. \tag{117}$$

For the integer transform, all the multiplication operations can be implemented by the method of (117), and all use the binary shifting and the addition operations. Sometimes, the amount of multiplication operations is huge, or the numbers we want to multiply are very large. In such a case, there will be a lot of binary shifting and addition operations required. We introduce some ways to decrease the number of addition and binary shifting operations as follows.

1) Decompose C properly so that in (117), the amount of a_r that $a_r \neq 0$ will be minimum.

   For example, we can decompose 15 as $15 = 1 \cdot 2^0 + 1 \cdot 2^1 + 1 \cdot 2^2 + 1 \cdot 2^3$ or $15 = -1 \cdot 2^0 + 1 \cdot 2^4$. However, the former has 4 a_rs not equal to 0, and the latter only has 2 a_rs not equal to 0. When multiplying 15, decomposing 15 as $-1 \cdot 2^0 + 1 \cdot 2^4$ will require less binary addition operations.
2) Some multiplication operations with the same multiplicand can be merged together.

For example, for the 8-point ITFT implemented by decimation-in-frequency (Fig. 2), between the fifth position of the second stage and the fifth and sixth position of the third stage, there is the branch as Fig. 17(a). For the case that $a_1 = 4$ and $c_1 = 3$, when we implement the multiplication of 3 and 4 individually, then we must require two binary shifting operations and one addition operation, as in Fig. 17(b). In fact, we can implement these two multiplication operations together. We can decompose 3 as $3 = -1 + 2^2$, and then, there is one binary shifting operation that is common for the multiplication of 3 and 4. Therefore, when we implement the multiplication of 3 and 4 together, then one binary shifting is saved, as in Fig. 17(c).

VII. Conclusion

In Sections III–V, we only derive the integer cosine, sine, Hartley, and Fourier transform of eight points and the integer Fourier transform of six points. In fact, we can also derive the integer transforms analogous to other noninteger transforms in a similar way.

For the integer transform, we can replace the real number multiplication operations with the fixed-points multiplication operations. Especially for the complete integer transform, there are no real number multiplication operations required. Thus, the integer transform is very efficient. If the datum we want to process are all integer numbers, using the complete integer transform is especially efficient, such as in image and video processing. This is because during the whole process, there are no real number processors required.

The concept of integer transform analogous to the noninteger transform is very new, and there has been little about it until now. Because they are convenient for implementation and almost retain the quality of original noninteger transform; therefore, they can compete with the original discrete transform in many applications. We believe the integer transform will be very popular in the future.

References

[1] W. K. Cham, "Development of integer cosine transform by the principles of dyadic symmetry," *Proc. Inst. Elect. Eng.*, pt. 1, vol. 136, no. 4, pp. 276–282, Aug. 1989.
[2] W. K. Cham and P. P. C. Yip, "Integer sinusoid transforms for image processing," *Int. J. Electron.*, vol. 70, no. 6, pp. 1015–1030, 1991.
[3] W. K. Cham and Y. T. Chan, "An order 16-integer cosine transform," *IEEE Trans. Signal Processing*, vol. 39, pp. 1205–1208, May 1991.
[4] S. N. Koh, S. J. Huang, and H. K. Tang, "Development of order 16-integer transforms," *Signal Process.*, vol. 24, pp. 283–289, Sept. 1991.
[5] R. N. Bracewell, *The Hartley Transform*. New York: Oxford Univ. Press, 1986.
[6] A. K. Jain, "A sinusoid family of unitary transforms," *IEEE Trans. Pattern Anal. Machine Intel.*, vol. PAMI-4, pp. 356–365, Oct. 1979.
[7] W. K. Cham, C. S. Choy, W. K. Lam, and S. M. Chiang, "2-D integer cosine transform chip set and its application," *IEEE Trans. Consumer Electron.*, vol. 38, pp. 43–47, May 1992.
[8] T. C. J. Pang, C. S. O. Choy, C. F. Chan, and W. K. Cham, "A self-timed ICT chip for image coding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, no. 6, pp. 856–860, 1999.
[9] W. K. Cham, "Integer sinusoid transform," *Adv. Electron. Phys.*, vol. 88, pp. 1–61, 1994.
[10] W. Stallings, *Computer Organization Architecture. Principle of Structure and Function*. New York: Macmillan, 1987.

Soo-Chang Pei (F'00) was born in Soo-Auo, Taiwan, R.O.C., in 1949. He received B.S.S.E. degree from National Taiwan University (NTU), Taipei, in 1970 and the M.S.S.E. and Ph.D. degrees from the University of California, Santa Barbara, in 1972 and 1975, respectively.

He was an engineering officer with the Chinese Navy Shipyard from 1970 to 1971. From 1971 to 1975, he was a Research Assistant with the University of California, Santa Barbara. He was the Professor and Chairman with the Electrical Engineering Department, Tatung Institute of Technology from 1981 to 1983 and with NTU from 1995 to 1998. Presently, he is a Professor with the Electrical Engineering Department, NTU. His research interests include digital signal processing, image processing, optical information processing, and laser holography.

Dr. Pei received the National Sun Yat Sen Academic Achievement Award in Engineering in 1984, the Distinguished Research Award from the National Science Council from 1990 to 1998, the Outstanding Electrical Engineering Professor Award from the Chinese Institute of Electrical Engineering in 1998, and the Academic Achievement Award in Engineering from the Ministry of Education in 1998. He has been President of the Chinese Image Processing and Pattern Recognition Society in Taiwan from 1996 to 1998 and is a member Eta Kappa Nu and the Optical Society of America.

Jian-Jiun Ding was born in 1973 in Pingdong, Taiwan, R.O.C. He received both the B.S. and M.S. degree in electrical engineering from National Taiwan University (NTU), Taipei, Taiwan, in 1995 and 1997, respectively. He is currently pursuing the Ph.D. degree under the supervision of Prof. S.-C. Pei with the Department of Electrical Engineering, NTU.

He is currently a Teaching Assistant with NTU. His current research areas include fractional and affine Fourier transforms, other fractional transforms, orthogonal polynomials, integer transforms, quaternion Fourier transforms, pattern recognition, fractals, and filter design.