# FRACTIONAL, CANONICAL, AND SIMPLIFIED FRACTIONAL COSINE TRANSFORMS

*Soo-Chang Pei* *Jian-Jiun Ding*
Department of Electrical Engineering, National Taiwan University, Taipei, Taiwan, R.O.C
Email address: pei@cc.ee.ntu.edu.tw

## ABSTRACT

Fourier transform can be generalized into the fractional Fourier transform (FRFT), linear canonical transform (LCT), and simplified fractional Fourier transform (SFRFT). They extend the utilities of original Fourier transform, and can solve many problems that can't be solved well by original Fourier transform.

In this paper, we will generalize the cosine transform. We will derive fractional cosine transform (FRCT), canonical cosine transform (CCT), and simplified fractional cosine transform (SFRCT). We will show they are very similar to the FRFT, LCT, and SFRFT, but they are much more efficient to deal with the even, real even functions. For digital implementation, FRCT and CCT can save 1/2 of the real number multiplications, and SFRCT can save 3/4. We also discuss their applications, such as optical system analysis and space-variant pattern recognition.

## I. INTRODUCTION

**Fractional Fourier transform (FRFT)** is defined as [1]:

$$G_F^{\phi}(s) = O_F^{\phi}(g(t))$$
$$= \sqrt{\frac{1-j\cot\phi}{2\pi}}\, e^{\frac{j}{2}\cot\phi\cdot s^2} \int_{-\infty}^{\infty} e^{-j\csc\phi\cdot s\cdot t}\, e^{\frac{j}{2}\cot\phi\cdot t^2}\, g(t)\, dt \,. \quad (1)$$

It is the generalization of Fourier transform ($\phi = \pi/2$). It can be used for many applications, such as optical system analysis, filter design, phase retrieval, pattern recognition, edge detection, etc. [2]. FRFT is a useful tool for signal processing.

FRFT can be further generalized into the **linear canonical transform (LCT)** [3]. It is defined as:

$$G_F^{(a,b,c,d)}(s) = O_F^{(a,b,c,d)}(g(t))$$
$$= \sqrt{\frac{1}{j2\pi b}}\, e^{\frac{j}{2}\frac{d}{b}s^2} \int_{-\infty}^{\infty} e^{-j\frac{s}{b}t}\, e^{\frac{j}{2}\frac{a}{b}t^2}\, g(t)\, dt \,. \quad (2)$$

It satisfies the additivity property as below:

$$O_F^{(a_2,b_2,c_2,d_2)}\left(O_F^{(a_1,b_1,c_1,d_1)}(g(t))\right) = O_F^{(a_3,b_3,c_3,d_3)}(g(t)), \quad (3)$$

$$\text{where} \quad \begin{bmatrix} a_3 & b_3 \\ c_3 & d_3 \end{bmatrix} = \begin{bmatrix} a_2 & b_2 \\ c_2 & d_2 \end{bmatrix}\begin{bmatrix} a_1 & b_1 \\ c_1 & d_1 \end{bmatrix}. \quad (4)$$

The FRFT is the special case of LCT that {*a*, *b*, *c*, *d*} = {cos$\alpha$, sin$\alpha$, $-$sin$\alpha$, cos$\alpha$}:

$$O_F^{\phi}(g(t)) = \sqrt{e^{j\phi}}\; O_F^{(\cos\phi,\sin\phi,-\sin\phi,\cos\phi)}(g(t)). \quad (5)$$

And the Fresnel transform is the special case of LCT that {*a*, *b*, *c*, *d*} = {1, $\lambda z/2\pi$, 0, 1}. So LCT can be viewed as the generalization of FRFT and Fresnel transform. All the applications of FRFT and Fresnel transform are also the applications of LCT, and the LCT is more flexible for these applications.

Recently, in [4], we have introduced the **simplified fractional Fourier transform (SFRFT)**:

$$G_{SF}^{\phi}(s) = O_{SF}^{\phi}(g(t)) = \sqrt{\frac{1}{j2\pi}} \int_{-\infty}^{\infty} e^{-jst}\, e^{\frac{j}{2}\cot\phi\, t^2}\, g(t)\, dt \,. \quad (6)$$

The SFRFT is the special case of LCT that {*a*, *b*, *c*, *d*} = {cot$\phi$, 1, $-$1, 0}. We have shown it has the same effects as FRFT, but simpler for digital implementation. For many applications, such as filter design, we can use SFRFT to substitute the FRFT.

In this paper, we will generalize the cosine transform (CT):

$$G_C(w) = CT(g(t)) = \sqrt{\frac{2}{\pi}} \int_0^{\infty} \cos(wt)\cdot g(t)\cdot dt \,. \quad (7)$$

Since cosine transform is much similar to Fourier transform, so, as the Fourier transform can be generalized into FRFT, LCT, SFRFT, we expect the cosine transform can also be generalized. In this paper, we will introduce the fractional cosine transform (FRCT), canonical cosine transform (CCT), and simplified fractional cosine transform (SFRCT). We will show FRCT and CCT is very efficient to deal with the even functions, and SFRCT is very efficient to deal with the real, even functions. For all the applications of FRFT and LCT, if the input is even function, we can use FRCT and CCT instead of FRFT and LCT. And if the input is real, even, we can use SFRCT instead of FRFT and LCT.

## II. DERIVATION OF THE TRANSFORMS

We remember that the transform results of Fourier transform and cosine transform has the relation as below:

$$G_C(w) = \left(G_F(w) + G_F(-w)\right)/2 \,. \quad (8)$$

So we can derive the fractional and canonical cosine transform from the FRFT and LCT by the relations as below:

$$G_C^{\phi}(s) = \left(G_F^{\phi}(s) + G_F^{\phi}(-s)\right)/2 \,, \quad (9)$$

$$G_C^{(a,b,c,d)}(s) = \left(G_F^{(a,b,c,d)}(s) + G_F^{(a,b,c,d)}(-s)\right)/2 \,, \quad (10)$$

and change the range of integration from ($-\infty$, $\infty$) into [0, $\infty$) . So we can define the fractional and canonical transform as:

- **Fractional cosine transform (FRCT):**

$$G_C^{\phi}(s) = O_C^{\phi}(g(t))$$
$$= \sqrt{\frac{2e^{j\phi}}{j\pi\sin\phi}}\, e^{j\frac{s^2}{2}\cot\phi} \int_0^{\infty} e^{j\frac{t^2}{2}\cot\phi} \cos(\csc\phi\cdot st)\, g(t)\, dt \,. \quad (11)$$

• **Canonical cosine transform (CCT)**

$$G_C^{(a,b,c,d)}(s)=O_C^{(a,b,c,d)}\big(g(t)\big)$$
$$=\sqrt{\frac{2}{j\pi b}}\,e^{\frac{j}{2}\frac{d}{b}s^2}\int_0^\infty \cos\left(\frac{s}{b}t\right)e^{\frac{j}{2}\frac{a}{b}t^2}\,g(t)\cdot dt\,. \tag{12}$$

We can prove they all satisfy the additivity property:

$$O_C^{\phi}\Big(O_C^{\varphi}\big(g(t)\big)\Big)=O_C^{\phi+\varphi}\big(g(t)\big), \tag{13}$$
$$O_C^{(a_2,b_2,c_2,d_2)}\Big(O_C^{(a_1,b_1,c_1,d_1)}\big(g(t)\big)\Big)=O_C^{(a_3,b_3,c_3,d_3)}\big(g(t)\big) \tag{14}$$

where $\{a_1, b_1, c_1, d_1\}$, $\{a_2, b_2, c_2, d_2\}$, $\{a_3, b_3, c_3, d_3\}$ satisfy the relation of Eq. (4). It is easy to recover the original function from the transform results of SFRCT and CCT:

$$g(t)=O_C^{-\phi}\Big(G_C^{\phi}(s)\Big),\quad g(t)=O_C^{(d,-b,-c,a)}\Big(G_C^{(a,b,c,d)}(s)\Big). \tag{15}$$

In [5], they had introduced another type of FRCT. They derived it by taking the real part of the kernel of FRFT. But for the FRCT derived in [5], it is hard to recover the original function from the transform result. This problem will not exist for the FRFT and LCT defined as Eq. (11), (12).

If the input function $g(t)$ is even, then the transform result of FRCT is the same as that of the FRFT, and the transform result of CCT is the same as that of the LCT ($g(t)$ is even):

$$O_C^{\phi}\big(g(t)\big)=O_F^{\phi}\big(g(t)\big),\quad O_C^{(a,b,c,d)}\big(g(t)\big)=O_F^{(a,b,c,d)}\big(g(t)\big). \tag{16}$$

Although FRCT and CCT have additivity property, and have well mathematical definition, but there is some problem. That is, for the real input, the output will not be real function. We will introduce another type of generalized cosine transform, i.e., simplified fractional cosine transform (SFRCT). For SFRCT, if the input is real, the output is also a real function.

We remember that the SFRFT is the special case of LCT that $\{a, b, c, d\} = \{\cot\phi, 1, -1, 0\}$. So we will also derive the SFRCT from the special case of CCT that $\{a, b, c, d\} = \{\cot\phi, 1, -1, 0\}$. We derive SFRCT as:

$$O_C^{\phi}\big(g(t)\big)=\mathrm{Re}\Big(\sqrt{j}\cdot O_C^{(\cot\phi,1,-1,0)}\big(g(t)\big)\Big), \tag{17}$$

and we suppose the input function $g(t)$ is real. So

• **Simplified fractional cosine transform (SFRCT):**

$$G_{SC}^{\phi}(s)=O_{SC}^{\phi}\big(g(t)\big)$$
$$=\sqrt{\frac{2}{\pi}}\int_0^\infty \cos(s\,t)\cos\big(\cot\phi\cdot t^2/2\big)g(t)\;dt\,. \tag{18}$$

And its inverse transform is:

$$g(t)=\sqrt{\frac{2}{\pi}}\sec\big(\cot\phi\cdot t^2/2\big)\int_0^\infty \cos(s\,t)G_{SC}^{\phi}(s)ds\,. \tag{19}$$

We note, for the SFRCT, if the input $g(t)$ is a real function, then the transform result will also be real. But, not the same as the cases of FRCT and CCT, the SFRCT don't have additivity property. As the FRCT defined in [5], the SFRCT also has the property of real-input-real-output, and has no additivity property. But there are some key differences. That is, it is easy to recover the original function from the transform result of SFRCT, but this work is hard to do for the FRCT defined in [5]. Besides, the SFRCT will have much simpler digital implementation structure.

We can prove if the input $g(t)$ is a real, even function, then the transform results of SFRCT and SFRFT will have the relation as:

$$G_{SC}^{\phi}(s)=\mathrm{Re}\Big(\sqrt{j}\cdot G_{SF}^{\phi}(s)\Big)\qquad \text{if } g(t) \text{ is real, even.} \tag{20}$$

For SFRFT, if the input is real, then the output is a complex function, and the degree of freedom of the output is twice of the input. It is over-determined. But for SFRCT, it preserves the real part of $\sqrt{j}\cdot G_{SF}^{\phi}(s)$, and treats its imaginary part as redundancy. And then, for real input, the output of SFRCT is also a real function, and the degrees of freedom of input and output are the same. It is reasonable. So we can use SFRCT instead of SFRFT when the input is a real, even function.

In [4], we have stated we can use SFRFT with parameter $\phi$ instead of FRFT with parameter $\phi$ and the LCT with $a/b = \cot\phi$ for many applications. So together with the above conclusion, we can conclude when the input $g(t)$ is a real, even function, we can use SFRCT with parameter $\phi$ to substitute the FRFT with parameter $\phi$ and the LCT with $a/b = \cot\phi$ for many applications.

## III. DIGITAL IMPLEMENTION

The **most important advantage** of FRCT, CCT, and SFRCT is they are much **simpler** for **digital implementation**. We will introduce their fast algorithms, and compare their complexities with those of the FRFT and LCT. In [6], we have stated when we choose the sampling intervals to satisfy $\Delta_t\Delta_s = 2\pi b/P$, ($P$ is the total number of sampling points), then we can use two chirp multiplications and one DFT to implement the FRFT and LCT. Eqch chirp multiplication requires $3P$ real number multiplications, and DFT require $P\cdot\log_2 P$ real number multiplications. So

• **Amount of real multiplications required for FRFT, LCT:**

$$6P + P\cdot\log_2 P. \tag{21}$$

Then we discuss the digital implementation of FRCT. When we implement FRCT, we first sample $t$-axis and $w$-axis as below:

$$t = (n+n_0)\Delta_t,\quad s = (m+m_0)\Delta_s,\quad n, m = 0, 1, \ldots, N-1, \tag{22}$$

and substitute them into Eq. (11). If we choose the values of $\Delta_s$, $\Delta_t$, $n_0$, $m_0$ properly to make the term $\cos(\csc\phi\cdot st)$ in Eq. (11) becomes the kernel of DCT. And then, we can use the fast algorithm of DCT [7][8] together with the chirp multiplications to implement the FRCT. The amount of real number multiplications required for DCT is (We suppose the input is a real function):

• for 1st type DCT: $1 - N + (N/2)\cdot\log_2 N$, (23)
• for 2nd, 3rd type DCT: $(N/2)\cdot\log_2 N$, (24)
• for 4th type DCT: $N + (N/2)\cdot\log_2 N$. (25)

For example, if we choose $\Delta_s$, $\Delta_t$, $n_0$, $m_0$ as:

$$\Delta_s\cdot\Delta_t = \pi\cdot\sin\phi/(N-1),\qquad n_0 = 0,\qquad m_0 = 0, \tag{26}$$

then Eq. (11) becomes

$$G_C^{\alpha}(m\Delta_s)=\sqrt{\frac{2-j2\cot\phi}{\pi}}\,e^{j\frac{m^2\Delta_s^2}{2}\cot\phi}\,\Delta_t\sum_{n=1}^{N-1}\cos\left(\frac{\pi mn}{N-1}\right)\tilde{g}(n\Delta_t) \tag{27}$$

where $\tilde{g}(0)=g(0)/2$, $\tilde{g}(n)=e^{j\frac{n^2\Delta_t^2}{2}\cot\phi}g(n\Delta_t)$. (28)

We note, $\cos(\pi mn/(N-1))$ is just the kernel of 1st type DCT. Thus, we can follow the process as below to implement FRCT:

(1) Do the chirp multiplication as Eq. (28) to obtain $\tilde{g}(m)$.

(2) Then do the 1st type DCT for $\tilde{g}(m)$.

(3) Then multiply the outside chirp term in Eq. (27).

The 1st and 3rd steps both requires $N$-points complex multiplications, and the 2nd step requires the 1st type DCT for complex input (i.e., two 1st type DCT for real input). So if we choose $\Delta_s$, $\Delta_t$, $n_0$, $m_0$ as Eq. (26), then the amount of real number multiplications required for FRCT is:

$$6N + 2 + N\cdot\log_2 N. \tag{29}$$

Since $N$ is just the number of sampling points for $t, s \geq 0$, (see Eq. (24)), so $N \approx P/2$ ($P$ is the total number of sampling points, including positive axis and negative axis). So

• **Amount of real multiplications required for FRCT, CCT:**

$$6N + 2 + N\cdot\log_2 N \;\approx\; 3P + (P/2)\log_2(P/2). \tag{30}$$

It is just about 1/2 of Eq. (21). So when we deal with the even functions, it is much more efficient to use FRCT instead of FRFT, and about half of real number multiplications can be saved.

Similarly, we can also follow the process as above, and implement the CCT by the DCT. Then the number of multiplications required is also the same as Eq. (30), and is about 1/2 of the amount of real number multiplications required for LCT.

So **FRCT and CCT** are indeed efficient tools to process **even** functions. We can use them instead of FRFT and LCT for many of the applications of FRFT and LCT when the input is even.

Then we discuss the case of SFRCT. We also sample $t$-axis and $w$-axis as Eq. (22). If we choose $\Delta_s$, $\Delta_t$, $n_0$, $m_0$ properly, then $\cos(st)$ term in Eq. (18) will become the kernel of DCT of type 1, 2, 3, or 4, and then we can implement SFRCT by the fast algorithm of DCT of type 1, 2, 3, or 4.

For example, we can choose

$$\Delta_s\cdot\Delta_t = \pi/(N-1), \qquad n_0 = 0, \qquad m_0 = 0, \tag{32}$$

then Eq. (20) becomes:

$$G_{SC}^{\alpha}\left((m+m_0)\Delta_s\right) = \sqrt{\frac{2}{\pi}}\,\Delta_t \sum_{n=0}^{N-1} \cos\left(\frac{\pi mn}{N-1}\right)\cdot \tilde{g}(n\Delta_t) \tag{32}$$

$$\text{where}\quad \tilde{g}(0) = \frac{g(0)}{2},\quad \tilde{g}(n\Delta_t) = \cos\left(\frac{\cot\phi}{2}\, n^2\Delta_t^2\right) g(n\Delta_t). \tag{33}$$

So we can implement the SFRCT of type 1 by the multiplication of $\cos(\cot\phi\cdot n^2\Delta_t^2/2)$ and the 1st type DCT. Since the input $g(n\Delta_t)$ is real, so Eq. (33) is the product of two real functions. It requires $N$ real number multiplications. And since the input of the 1st type DCT is also a real function, so from Eq. (23), the 2nd step require $1-N+(N/2)\cdot\log_2 N$ real number multiplications. So

• **Amount of real multiplications required for SFRCT:**

$$1 + (N/2)\cdot\log_2 N \;\approx\; (P/4)\cdot\log_2(P/2). \tag{34}$$

We also use the fact that $N$ (the number of sampling points used for DCT, i.e., the sampling points for $t \geq 0$, as Eq. (22)) is about half of $P$ (total number of sampling points) for SFRCT. Eq. (34) is not only much less than the complexities of FRFT and LCT (see Eq. (21)) (about 1/4), but also less than the complexities of FRCT and CCT (see Eq. (30)) (about 1/2).

So when the input function is **real** and **even**, it is much more efficient to use the **SFRCT** to process these functions than using the FRFT and LCT.

## IV. APPLICATIONS

The applications of FRCT, CCT, and SFRCT can be summarized briefly. That is, **FRFT and CCT can replace the FRFT and LCT** when the inputs are **even functions**, and **SFRCT can replace FRFT and LCT** when the inputs are **real, even functions.**

From Eq. (16), the transform results of FRCT and CCT are the same as those of FRFT and LCT when the inputs are even. Form Eq. (20), we know for real, even inputs, SFRCT with parameter $\phi$ has very close relation with the SFRFT with parameter $\phi$, and hence has very close relation with the FRFT with parameter $\phi$ and LCT with $a/b = \cot\phi$. And together with the discussion about digital implementation in Sec. 3, we can conclude:

(1) When the input is **even**, we can use **FRCT** and **CCT** instead of FRFT and LCT, and there are about **half** of real number multiplications can be saved.

(2) When the input is **real** and **even**, we can use **SFRCT** instead of FRFT and LCT, and there are about **3/4** of real number multiplications can be saved.

So the most important utility of FRCT, CCT, and SFRCT is they can substitute the FRFT and LCT when the input is even. Thus, the applications of FRCT, CCT, SFRCT are the same as the applications of FRFT and LCT. The applications of FRFT and LCT are filter design, optical system analysis, radar system analysis, solving differential equations, phase retrieval, multiplexing, space-variant pattern recognition, edge detection, etc. They are also the applications of FRCT, CCT, and SFRCT.

We will give two examples, one is optical system analysis, and the other is space-variant pattern recognition.

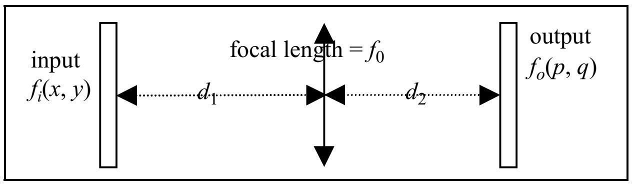

Fig. 1 Optical system with one spherical lens, two free spaces.

For the optical system as Fig. 1, the relation between the input and output can be expressed as by LCT

$$f_o(p,q) = e^{j\varphi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} K_F^{(a,b,c,d)}(p,x)K_F^{(a,b,c,d)}(q,y)f_i(x,y)dxdy \tag{35}$$

$$\text{where}\quad \begin{bmatrix} a & b \\ c & d \end{bmatrix} = \begin{bmatrix} 1-d_2/f & (d_1+d_2-d_1d_2/f_0)/k \\ -k/f_0 & 1-d_1/f_0 \end{bmatrix}, \tag{36}$$

and $\varphi$ is some constant phase, $k = 2\pi/\lambda$, and

$$K_F^{(a,b,c,d)}(p,x) = \sqrt{\frac{1}{j2\pi b}}\cdot e^{j\frac{d}{2b}p^2} e^{j\frac{1}{b}px} e^{j\frac{a}{2b}x^2}. \tag{37}$$

In the case that the input is even for both $x$-axis and $y$-axis:

$$f_i(x, y) = f_i(-x, y), \qquad f_i(x, y) = f_i(x, -y), \tag{38}$$

then we can use the CCT with the same parameters to substitute the LCT for both $x$-axis and $y$-axis, and change Eq. (35) as:

$$f_o(p,q) = e^{j\varphi}\int_0^{\infty}\int_0^{\infty} K_C^{(a,b,c,d)}(p,x)K_C^{(a,b,c,d)}(q,y)f_i(x,y)\cdot dxdy \tag{39}$$

where $\varphi$ is some constant phase, the values of $\{a, b, c, d\}$ are the same as Eq. (36), and $K_C^{(a,b,c,d)}(p, x)$ is the kernel of CCT:

$$K_C^{(a,b,c,d)}(p,x)=\sqrt{\frac{2}{j\pi b}}\,e^{j\frac{d}{2b}p^2}\cos(px/b)\,e^{j\frac{a}{2b}x^2} . \qquad (40)$$

The value of $f_o(p, q)$ calculated from Eq. (39) is the same as the value of $f_o(p, q)$ calculated from Eq. (35), but the complexity of digital implementation of Eq. (39) is much less, because the complexity of CCT is only about 1/2 of the complexity of LCT.

Then we will discuss the application of space-variant pattern recognition. In [9], they had illustrated how to use FRFT for space-variant pattern recognition. They use fractional correlation:

$$z(t)=O_{corr}^{q}\left(x(t),y(t)\right)=FT[X_F^q(s)\cdot\overline{Y_F^q(s)}] \qquad (41)$$

$$\text{where}\quad X_F^q(s)=O_F^q(x(t)),\quad Y_F^q(s)=O_F^q(y(t)). \qquad (41a)$$

Then we can choose $x(t)$ as the reference pattern, and choose $y(t)$ as the input object. And then, we can use the inequality as below:

$$\text{Max}(|z(t)|) > \text{threshold} \qquad (42)$$

to determine whether the input object matches the reference pattern. Eq. (42) will be satisfy only when

$$y(t)=x(t-t_0), \qquad \text{and } |t_0| < R. \qquad (43)$$

If the input object doesn't match the reference pattern, or the difference of locations is too large, then Eq. (42) will not satisfy, and the input object will not be recognized as the desired pattern. So FRFT can be used for space-variant pattern recognition.

In fact, if the reference pattern is real and even, we can use SFRCT instead of FRFT for the application of space-variant pattern recognition. In this case, Eq. (41) is changed as below:

$$z(t)=O_{CCR}^{q}\left(x(t),y(t)\right)=CT[X_{SC}^q(s)\cdot Y_{SC}^q(s)] \qquad (44)$$

$$\text{where}\quad X_{SC}^q(s)=O_{SC}^q(x(t)),\quad Y_{SC}^q(s)=O_{SC}^q(y(t)). \qquad (44a)$$

Although the input object $y(t)$ may not be an even function, but we can prove this won't cause any problem when we use SFRCT instead of FRFT for $y(t)$ in this application. Then, we can also use the inequality of Eq. (42) to determine whether the input matches the reference pattern. We find, as the case when we use FRFT, Eq. (43) is still the condition that Eq. (42) is satisfied. We can use $q$ to control the value of $R$. If $|\tan q|$ is large, then $R$ is also large (when $q = \pi/2$, $R \to \infty$). If $|\tan q|$ is small, then $R$ is small.

We give an example as below. We choose the reference:

$$x(t) = \Lambda(t/1.6), \qquad (45)$$

and plot it in Figs. 2(a), 3(a). Then we choose the inputs as:

$$\text{Fig. 2(b):}\quad y(t)=x(t-2), \qquad \text{Fig. 3(b):}\quad y(t)=x(t-14). \quad (46)$$

Then we choose $q = \pi/2$, and calculate $z(t)$ by Eq. (44), and plot $|z(t)|$ in Figs. 2(c), 3(c). Since in this case, the value of $R$ in Eq. (43) is infinite, so no matter how large the displacement is, the value of Max($|z(t)|$) will not be attenuated. Then we choose $q = 0.45\pi$, and plot $|z(t)|$ in Figs. 2(d), 3(d). In this case, the value of $R$ in Eq. (43) is finite. So even when the input is same as the reference, but if the displacement is too large, as the case of Fig. 3 ($t_0 = 14$), the value of Max($|z(t)|$) will be attenuated.

Thus, when $|\tan q| \to \infty$, we do the space-invariant pattern recognition. When $|\tan q|$ is finite, we do the space-variant pattern recognition. This is the same as the case when we use FRFT [9], but the complexity of computation is much less (just about 1/4).

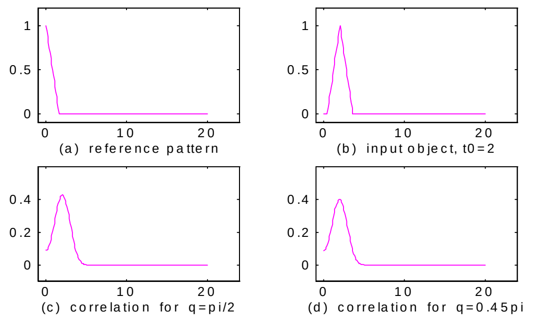

Fig. 2 Space-variant pattern recognition, displacement = 2.

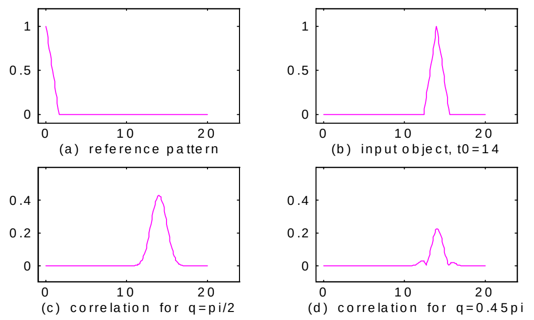

Fig. 3 Space-variant pattern recognition, displacement = 14.

## V. CONCLUSION

In this paper, we have introduced the fractional, canonical, and simplified fractional cosine transforms (FRCT, CCT, SFRFT). The complexities of FRCT and CCT are 1/2 of those of the FRFT and LCT, and the complexity of SFRCT is 1/4 of those of the FRFT and LCT. We can use FRCT and CCT to replace FRFT and LCT when the input is even, and use SFRCT to replace FRFT and LCT when the input is real and even.

We have also derived the fractional, canonical, and simplified fractional sine and Hartley transforms in [10].

## REFERENCES

[1] V. Namias, *J. Inst. Maths. Applics.*, v. 25, p 241-265, 1980.

[2] H. M. Ozaktas, M. A. Kutay, and D. Mendlovic, *Advances in Imaging and Electron Physics*, vol. 106, Ch. 4, 1999.

[3] K. B. Wolf, "*Integral Transforms in Science and Engineering*", Ch. 9, New York, Plenum Press, 1979.

[4] S. C. Pei and J. J. Ding, 'Simplified Fractional Fourier transform', to appear in *J. Opt. Soc. Am. A*, 2000.

[5] A. W. Lohmann, D. Mendlovic, Z. Zalevsky, and R. G. Dorsch, *Opt. Commun.*, vol. 125, p 18-20, Apr. 1996.

[6] S. C. Pei and J. J. Ding, *IEEE Trans. Signal Processing*, vol. 48, no. 5, p. 1338-1353, May 2000.

[7] B. G. Lee, *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-32, p. 1243-1245, Dec. 1984.

[8] C. W. Kok, vol. 45, no. 3, p. 757-760, March 1997.

[9] A. W. Lohmann, Z. Zalevsky, and D. Mendlovic, *Opt. Commun.*, vol. 128, p. 199-204, Jul. 1996.

[10] S. C. Pei and J. J. Ding, 'Fractional, canonical, and simplified fractional cosine, sine, and Hartley transforms', submitted to *IEEE Trans. Signal Processing*.