行政院國家科學委員會專題研究計畫　期中進度報告

數位式離散分數訊號轉換及其應用(1/3)

計畫類別：個別型計畫
計畫編號：NSC92-2213-E-002-089-
執行期間：92 年 08 月 01 日至 93 年 07 月 31 日
執行單位：國立臺灣大學電機工程學系暨研究所

計畫主持人：貝蘇章

報告類型：精簡報告

處理方式：本計畫可公開查詢

中　華　民　國 93 年 5 月 4 日

數位式離散分數訊號轉換及其應用(1/3)

Digital Discrete Fractional Signal Transforms and Its Applications (1/3)

計畫編號：NSC 92-2213-E-002-089

執行期限： 92 年 8 月 1 日 至 93 年 7 月 31 日

主持人：貝蘇章　台灣大學電機系教授

摘要

在影像處理上，一般而言，不同的影像在頻域當中會有不同的振幅及不同的相角。然而，在這裡，我們發現了，只要將原影像加上了適當的虛數部分，就可以將這些影像的振幅頻譜變得相同而只有相角頻譜不同。或者，反過來，將它們的相角頻譜變相同而只有振幅頻譜不同。我們稱前者為相位金匙演算法，稱前者為振幅金匙演算法。這些演算法可適用於分數傅氏轉換，線性完整轉換，及分數正絃餘絃轉換等等。它們對於影像或聲波處理，提供了一種有效的加密保密及編碼的方式。

關鍵詞：相位金匙，振幅金匙，離散分數傅氏轉換，離散線性完整轉換

ABSTRACT

Different images always have different amplitude spectrums. However, in this paper, we show that, with the appending of proper imaginary parts, we can make a series of natural images have the same amplitude spectrum but different phase key in the frequency domain. Since the only difference among the spectra is phase, we can use the phase spectrum to represent each of the images. Different images correspond to different phase keys. It is useful for image encryption and data compression. Our algorithm can be applied for not only the FT but also the fractional Fourier transform (FRFT) and many other operators. Since many parts of the algorithm can be treated as keys, we can effectively encrypt an image. Moreover, we can also reverse the direction of the algorithm, i.e., make a series of images have the same phase spectrum and use the amplitude spectra as the keys to distinguish different images.

Key words: phase key and amplitude key algorithms, discrete fractional Fourier transform, discrete linear canonical transform

1. INTRODUCTION

In the frequency domain, phase is usually more important than amplitude [1][2]. If two images have the same spectrum-phase but different spectrum-amplitude, they may look similar. In contrast, if two images have the same spectrum-amplitude but different spectrum-phase, they are usually quite different. Thus, phase information is critical to distinguish two different images.

In this paper, we show that, for any two or more natural images, we can modify them a little and make them have the same spectrum-amplitude in the frequency domain. Since the only difference between their spectrums is the phase, we can use the phase to represent each of the images. Different image corresponds to different phase key. This algorithm is useful for image encryption. Besides, since only the spectrum-phase is required for coding each of the images and the spectrum-amplitude is fixed, the algorithm is also useful for data compression.

Our idea comes from the concept of phase retrieval [4]. In optical systems, only the intensity of light can be observed. Phase retrieval deals with finding the phase of a light if we know its intensities at the input and the output of an optical system, i.e., to find the phase key that can transform one intensity into another intensity through the optical system.

We do some modification to convert the phase retrieval problem in optics into the phase key algorithm in image processing. First, we replace the intensity of light by the gray level of the image. Besides, we regard the optical system as some operators (such as the Fourier transform (FT)). Moreover, to avoid the time-consuming iterative process, which is required for most of the existed phase retrieval algorithm, we take the real part instead of taking the amplitude at the output.

In Sec. 2, we illustrate the phase key algorithm in detail, and show how to implement it by the FT. In Sec. 3, we show how to use phase key algorithm for image encryption, data compression, and image transformation. In Sec. 4, we show the flexibility of phase key algorithm. For example, we can use the fractional Fourier transform (FRFT) [5] and other operators instead of the FT. Since phase key algorithm is very flexible, and many parts of it can be treated as keys, it is very effective for encryption. In Sec. 5, we introduce three structures that can be used for multiple images encryption: parallel, series, and tree structures. In Sec. 6, we introduced the amplitude key algorithm. The phase key algorithm introduced in Sec. 2 makes a series of images have the same amplitude spectrum. In contrast, the amplitude key algorithm makes a

series of images have the same phase spectrum and treat the amplitude spectra as the keys to distinguish them. In Sec. 7, we make a conclusion.

2. PHASE KEY ALGORITHM BY FOURIER TRANSFORMS

The structure of phase key algorithm is shown in Fig. 1 where

$f(x, y)$, $g(x, y)$: two natural images

$\tilde{f}(x, y) = f(x, y)$ + imaginary part

$\tilde{g}(x, y) = g(x, y)$ + imaginary part

$\tilde{F}(w, s) = FT[\tilde{f}(x, y)]$, $\tilde{G}(w, s) = FT[\tilde{g}(x, y)]$,

$|\tilde{F}(w, s)| = |\tilde{G}(w, s)| = A(w, s)$. (1)

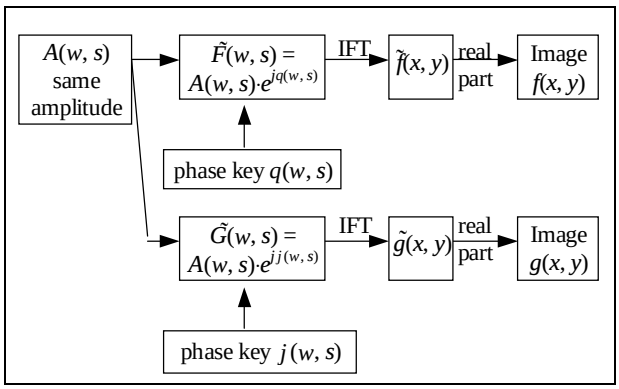


Fig. 1 The block diagram of phase key algorithm.

The complex functions $\tilde{f}(x, y)$ and $\tilde{g}(x, y)$ come from two different natural images, however, in the frequency domain, they have the same spectrum-amplitude, $A(w, s)$. If $A(w, s)$ satisfies

$A(w, s)$ is real and even, (2)

$A(w, s) \geq |F(w, s)|, |G(w, s)|$ for any w, s

where $F(w, s) = FT[f(x, y)]$, $G(w, s) = FT[g(x, y)]$, (3)

we can always find the phases $q(w, s)$ and $j(w, s)$ such that the real part of $\tilde{f}(x, y) = IFT[A(w, s)\cdot e^{jq(w, s)}]$ is $f(x, y)$, and the real part of $\tilde{g}(x, y) = IFT[A(w, s)\cdot e^{jj(w, s)}]$ is $g(x, y)$. If $A(w, s)$ has been known, $q(w, s)$ and $j(w, s)$ can be treated as the keys to recover the natural images $f(x, y)$ and $g(x, y)$.

Then, we find the solutions for $q(w, s)$ and $j(w, s)$. We apply the property that the real input of FTs corresponds to the conjugated symmetric output:

$FT[f(x, y)] = F(w, s) = [\tilde{F}(w, s) + \tilde{F}^*(-w, -s)]/2$ (4)

where $\tilde{F}(w, s) = FT[\tilde{f}(x,y)] = A(w, s)\cdot e^{jq(w,s)}$. So

$F(w, s) = A(w, s)\cdot[e^{jq(w,s)}+e^{-jq(-w,-s)}]/2$. (5)

Here we use the constraint that $A(w, s) = A(-w, -s)$. Thus,

$e^{jq(w,s)}+e^{-jq(-w,-s)} = 2F(w, s)/A(w, s)$. (6)

It is easy to see that the range of $e^{jq(w,s)}+e^{-jq(-w,-s)}$ is

$|e^{jq(w,s)} + e^{-jq(-w,-s)}| \leq 2$. (7)

Thus, the constraint that we can solve $q(w, s)$ from (7) is that $A(w, s) \geq |F(w, s)|$. This is why we use (3) as the constraint of $A(w, s)$. If (3) is satisfied, (7) can be expressed as in Fig. 2 where $2F(w, s)/A(w, s) = C\cdot e^{jf}$.

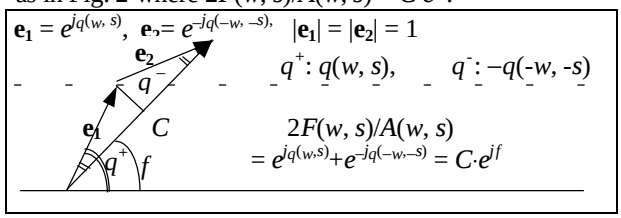


Fig. 2 Solve the phase key function $q(w, s)$ from (6).

In Fig. 2, the region surrounding by the three vectors is an isosceles triangle. Thus, $q(w, s)$ can be solved as

$q^+ = f + \cos^{-1}(C/2)$,

$q^- + (p - q^+) + 2\cos^{-1}(C/2) = p$, $q^- = f - \cos^{-1}(C/2)$. (8)

That is,

$q(w, s) = \cos^{-1}[|F(w, s)|/A(w, s)] + \arg[F(w, s)]$, (9)

$q(-w, -s) = \cos^{-1}[|F(w, s)|/A(w, s)] - \arg[F(w, s)]$.

This is the solution of phase key function $q(w, s)$. We can use the similar way to solve the phase key function $j(w, s)$:

$j(w, s) = \cos^{-1}[|G(w, s)|/A(w, s)] + \arg[G(w, s)]$, (10)

If $A(w, s)$ satisfies the constraints in (2) and (3), we can use (9) and (10) to solve the phase key functions $q(w, s)$ and $j(w, s)$ easily. We can use $q(w, s)$ to encode the image $f(x, y)$, and use $j(w, s)$ to encode the image $g(x, y)$, and use the block diagram in Fig. 1 to recover $f(x, y)$ and $g(x, y)$ from $q(w, s)$ and $j(w, s)$.

Notice that the amplitude function $A(w, s)$ in Fig. 1 is free to choose. The only requirement is that (2) and (3) should be satisfied. It makes phase key algorithm very flexible. For example, we can choose $A(w, s)$ as a constant:

$A(w, s) = B$ where $B \geq \text{Max}[|F(w, s)|], \text{Max}[|G(w, s)|]$, (11)

or choose it as an exponent decay function of radius:

$A(w, s) = C\cdot\exp(-R)$ where $R = (w^2+s^2)^{1/2}$, (12)

$C \geq \text{Max}[|F(w, s)\exp(R)|], \text{Max}[|G(w, s)\exp(R)|]$.

Moreover, we can also generate $A(w, s)$ from $F(w, s)$:

$A(w, s) = t + r\cdot|F(w, s)|$ (13)

where t and r are chosen properly such that the constraint in (3) can be satisfied. In this condition, the block diagram in Fig. 1 can be re-plotted as in Fig. 3. Since the spectrum amplitude is generated from $f(x, y)$, the phase $j(w, s)$ and the parameters t, r can be treated as the key to convert $f(x, y)$ into $g(x, y)$.

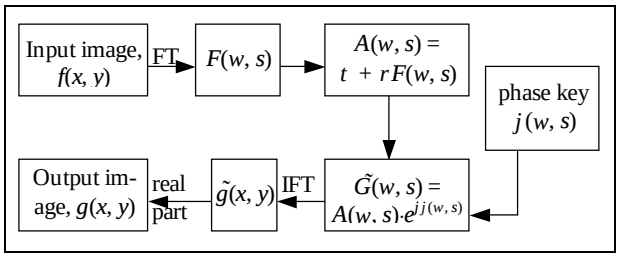


Fig. 3 The block diagram that uses phase key algorithm to convert image $f(x, y)$ into another image $g(x, y)$.

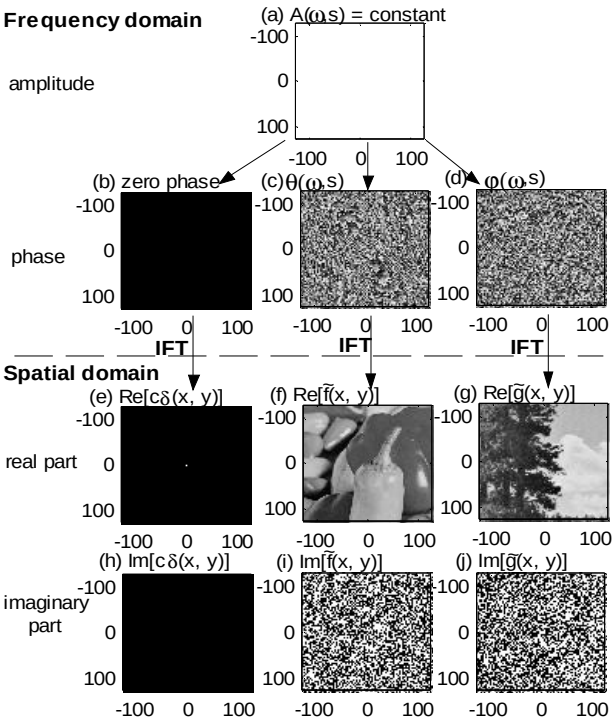


Fig. 4 Using phase key algorithm for image encoding and encryption. The spectrum amplitude is fixed as a constant.

We perform an experiment in Fig. 4. In Fig. 4(a), we use a constant function as the common spectrum-amplitude:

$$A(w, s) = 36000 \qquad \text{for all } p, q. \tag{14}$$

We try to use it together with phase key algorithm to encrypt two natural images, Fruit image and Forest image (See Fig. 4(f)(g). We denote them by $f(x, y)$ and $g(x, y)$). We use (9) and (10) to compute the phase key functions, which are plotted in Figs. 4(c)(d). Notice that both of them look like noise, however, we can use them to recover the desired image. After doing

$$\tilde{f}(x, y) = \text{IDFT}\{A(w, s)\cdot e^{jq(w, s)}\}$$
$$\tilde{g}(x, y) = \text{IDFT}\{A(w, s)\cdot e^{jj(w, s)}\}, \tag{15}$$

and taking their real part, the original images, Fruit image and Forest images, are recovered (seen Fig. 4(f)(g)).

Notice that $\tilde{f}(x, y)$ and $\tilde{g}(x, y)$ correspond to two different natural images, however, in the frequency domain, their spectrum-amplitudes are all the same. Although in the spatial domain, the following three functions are quite different:

(1) $a(x, y) = (36000\times 2p)\cdot d(x, y)$ (Fig. 4(e)(h)),
(2) $\tilde{f}(x, y)$ = Fruit image + imaginary part (Fig. 4(f)(i)),
(3) $\tilde{g}(x, y)$ = Forest image + imaginary part (Fig. 4(g)(j)), (16)

however, in the frequency domain, all of them have the spectrum-amplitude of $A(w, s) = 36000$. Since the only difference among their spectrums is phase, we can use the spectrum phases to represent each of the images. We can use $q(x, y)$ to represent Fruit image, and use $j(x, y)$ to represent Forest image. To recover the desired image, we must know its corresponding phase key. Otherwise, we may obtain a fully different image. This is why we can use phase key algorithm for image encryption.

3. APPLICATIONS

We can use phase key algorithm for **image encryption**, **data compression**, and **image transformation**. The application of image encryption has been shown by the experiment in Fig. 4.

For the application of **data compression**, suppose that there is a series of images $f_1(x, y)$, $f_2(x, y)$, ……, $f_T(x, y)$, in the frequency domain, we can choose a proper spectrum-amplitude function $A(w, s)$ that satisfies

$A(w, s)$ is real, $A(w, s) = A(-w,-s)$,
$A(w, s) \ge F_t(w, s)$ for any w, s
where $F_t(w, s) = FT[f_t(x, y)]$, $t = 1, 2, \ldots\ldots, T$. (17)

After $A(w, s)$ is chosen, we can use (9) to calculate the phase function $q_t(w, s)$, which is corresponding to $f_t(x, y)$. Then, in the frequency domain, for each of the image $f_t(x, y)$, since the spectrum-amplitude $A(w, s)$ is fixed, we only have to store their corresponding phase function $q_t(w, s)$.

- original spectrums:
  different amplitude, different phase
- with phase key algorithm:
  same amplitude, different phase.

For the application of **image transformation**, with the structure in Fig. 3, we can design a way that uses a phase function $j(w, s)$ to transform one image into another image.

Moreover, we can also use phase key algorithm to do the fractionalization of image transformation, i.e., find the intermediate between two images. Suppose that there are two natural images, $f(x, y)$ and $g(x, y)$. From the algorithm introduced in Sec. 2, if $A(w, s) > |F(w, s)|$ and $|G(w, s)|$ for all w, s, we can find the phase function $q(w, s)$ and $j(w, s)$, which correspond to $f(x, y)$ and $g(x, y)$, respectively. Then, we can define the intermediate image $h_a(x, y)$ as:

$$h_a(x, y) = \mathrm{Re}[\mathrm{IDFT}(A(w, s)\cdot\exp\{(1-a)\cdot q(w, s)+ a\cdot j(w, s)\})]. \quad (18)$$

If a =0, $h_a(x, y) = f(x, y)$. If a =1, $h_a(x, y) = g(x, y)$. If 0 < a < 1, we can show that, the image $h_a(x, y)$ is the intermediate between $f(x, y)$ and $g(x, y)$. It partially looks like $f(x, y)$, and partially looks like $g(x, y)$. When α is small, it is more similar to $f(x, y)$. When a grows larger, it is more and more similar to $g(x, y)$.

In Fig. 5, we do an experiment to find the intermediate image between Fruit and Forest. Here, we choose $A(w, s)$ = 36000, which is the same as Fig. 4(a). We vary a from 0 to 1. As a grows larger, the intermediate image becomes more and more similar to

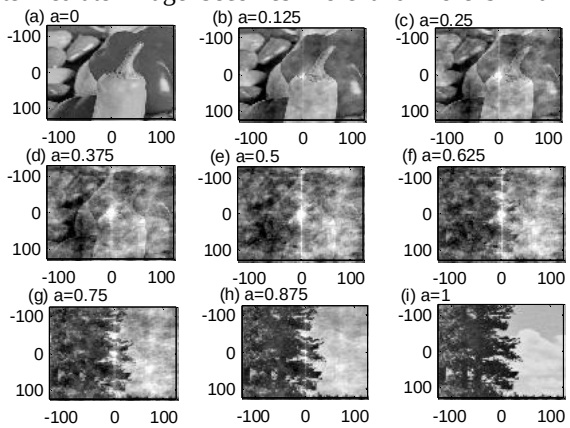


Fig. 5 Using phase key algorithm to produce intermediate images.

4. THE FLEXIBILITY OF PHASE KEY ALGORITHM

In this section, we show the flexibility of the phase key algorithm. We have shown the block diagram of the phase key algorithm in Fig. 1. In fact, many parts of it can be generalized. For example, in Fig. 1, the FT / IFT can be replaced by the fractional Fourier transform (FRFT).

4-1 Phase Key Algorithm by Fractional Fourier Transform

The fractional Fourier transform (FRFT) [5][6] is a generalized of the FT. It is defined as:

$$G_a(w, s) = \frac{1-j\cot a}{2p} e^{\frac{j}{2}\cot a\left(w^2+s^2\right)} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-j\csc a\,(w\,x+sy)} e^{\frac{j}{2}\cot a\left(x^2+y^2\right)} g(x, y)\,dx\,dy \quad . (19)$$

When a = p/2, it becomes the original FT. When a = –p/2, it becomes the original IFT. Many applications of the FT, such as filter design and pattern recognition, can be generalized by the FRFT. Similarly, the phase key algorithm can also be used in the FRFT domain.

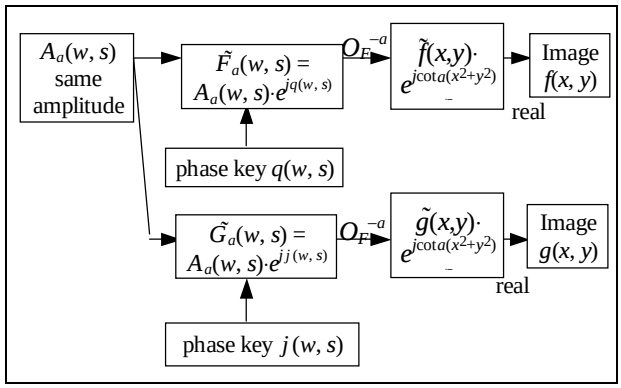


Fig. 6 The block diagram that uses the FRFT to implement the phase key algorithm.

The block diagram that uses the FRFT to implement the phase key algorithm is shown in Fig. 6 where O_F^{-a} means the FRFT with parameter $-a$. When using the FRFT, the process is very similar to that in subsection 2-1, except for the following differences:

1) The FT and IFT are changed into the FRFT with parameters a and $-a$, respectively.

2) The constraints for the amplitude spectrum becomes:

$A_a(w, s)$ is real, positive, and even,

$$A_a(w, s) \ge |\sin a\cdot F(w\cdot\csc a, s\cdot\csc a)|, \quad |\sin a\cdot G(w\cdot\csc a, s\cdot\csc a)| \quad \text{for all } w, s. \quad (20)$$

3) If the above constraints are satisfied, we use the following equations instead of (13) and (14) to determine the phase key:

$$q(w, s) = \cos^{-1}[|F(w\cdot\csc a, s\cdot\csc a)|/A_a(w, s)] + \arg[F(w\cdot\csc a, s\cdot\csc a)] + f(w, s)$$

$$j(w, s) = \cos^{-1}[|G(w\cdot\csc a, s\cdot\csc a)|/A_a(w, s)] + \arg[G(w\cdot\csc a, s\cdot\csc a)] + f(w, s)$$

$$\text{where } f(w, s) = (w^2 + s^2)\cot\alpha/2 - p/2 + a. \quad (21)$$

4) Instead of taking the real part of $\tilde{g}(x, y)$, we obtain the desired image from:

$$\text{desired image} = \mathrm{Re}\{\exp[j\cot a(x^2+y^2)/2]\cdot\tilde{g}(x, y)\}. \quad (22)$$

In Fig. 7, we show an example that uses the FRFT and the phase key algorithm to do image encoding and encryption for Forest image. We choose the amplitude spectrum as a constant:

$$A_a(w, s) = 36000. \quad (23)$$

It satisfies the constraints in (20). We then uses (21) to find the phase key $j_a(w, s)$, and plotted in Fig. 7(b). If we do the FRFT with parameter $-a$ for 36000exp[$jj_a(w, s)$], and take the real part of the chirp multiplication of the transformed result, we can successfully recover the original Fruit image, as in Fig. 7(d).

In Fig. 7(c), the phase function $j(w, s)$ is the same as that in Fig. 4(d). It is the phase key for Fruit image when we use the FT to implement the algorithm. However, now the operation we use is the FRFT. If we use it as the phase spectrum and follow the same process as that from Figs.

4(g)(j), we can't recover the desired image. Thus, besides of the phase key function, whether the FT or the FRFT we use, and the parameter a of the FRFT can also be treated as the key for image encryption.

With the FRFT, the phase key algorithm becomes more flexible. The FRFT provides an extra parameter a, and improve the security of image encryption.

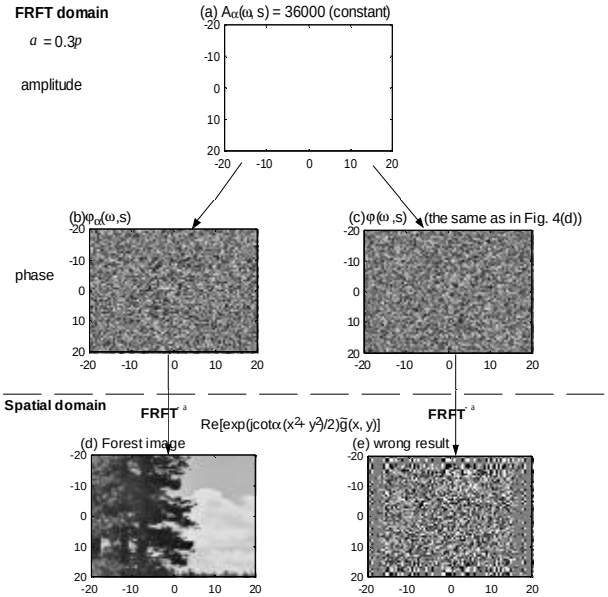


Fig. 7 Using the FRFT to implement the phase key algorithm for image encryption ($a = 0.3p$).

4-2 Phase Key Algorithm by Other Operators

The linear canonical transform (LCT) [7], which is a further generalization of the FRFT, can also be used to implement the phase key algorithm. It is defined as:

$$G_{(a,b,c,d)}(w,s)=\frac{1}{j2p\,|\,b\,|}e^{\frac{jd}{2b}\left(w^2+s^2\right)} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-\frac{j}{b}(w\,x+sy)} e^{\frac{ja}{2b}\cot a\left(x^2+y^2\right)} g(x,y)\,dx\,dy \quad (22)$$

When using the LCT instead of the FT to implement the phase key algorithm, in Fig. 1, the FT is replaced by the LCT with parameters $\{a, b, c, d\}$, and the IFT is replaced by the LCT with parameters $\{d, -b, -c, a\}$. In addition, instead of (7), the constraint of the amplitude spectrum is changed into:

$$A_{abcd}(w, s) \ge |b \cdot F(w/b, s/b)|, \quad |b \cdot G(w/b, s/b)| \quad \text{for all } w, s. \quad (23)$$

The phase key functions are calculated from the following equations:

$$q(w, s) = \cos^{-1}[|F(w/b, s/b)|/A_{abcd}(w, s)] + \arg[F(w/b, s/b)] + f(w, s) \quad (24)$$

$$j(w, s) = \cos^{-1}[|G(w/b, s/b)|/A_{abcd}(w, s)] + \arg[G(w/b, s/b)] + f(w, s)$$

where $f(w, s) = (w^2 + s^2)\cot\alpha/2 - p/2$. (25)

Besides of the FT, FRFT, and LCT, there are many other operators can be used to implement the phase key algorithm. We can classify these operations into two classes

(1) The operations that are closely related to the FT:

For example, the main reason that we can use the FRFT and LCT to implement the phase key algorithm is that they can be decomposed into the combination of the FT and chirp multiplications. In addition, the Fresnel transform [8], the Mellin transform, the Laplace transform, the Z transform, the Fraunhofer transform, the cosine transform, the sine transform, the Hartley transform, and many other Fourier-related operations can be used to implement the phase key algorithm.

(2) The linear operations whose real part and imaginary part of the kernels have some symmetry properties:

In Sec. 2, we derived the formulas of phase key functions based on the fact that the FT has the following symmetric property:

real input $f(x, y)$ $\longrightarrow$ conjugated symmetric output

$$F(w, s) = F^*(-w, -s). \quad (26)$$

Thus, if an operation has the above symmetric property, we can also use it to implement the phase key algorithm. To have the above symmetric property, the operation's kernel should satisfy some requirement. If the operation O_T can be expressed as:

$$F_T(w,s)=O_T[f(x,y)]=\int\int K_T(w,s,x,y)f(x,y)\,dxdy \quad (27)$$

and the kernel $K_T(w, s, x, y)$ satisfies:

$$\mathrm{Re}[K_T(w, s, x, y)] = \mathrm{Re}[K_T(-w,-s, x, y)]$$
$$\mathrm{Im}[K_T(w, s, x, y)] = -\mathrm{Im}[K_T(-w,-s, x, y)] \quad (28)$$

the operator also have the symmetry property as in (26), and we can use it to implement the phase key algorithm.

It's difficult to find the continuous operation whose kernel satisfies (28) and at the same time its inverse operation is not too complicated. However, in discrete case, the work becomes simpler. A two-dimensional discrete linear operation can be expressed as:

$$\mathbf{Y} = \mathbf{T_R X T_C} \qquad \mathbf{X}\text{: input}, \quad \mathbf{Y}\text{: input}. \quad (29)$$

If the row and column transform matrices $\mathbf{T_R}$ and $\mathbf{T_C}$ satisfy:

$$\mathbf{T_R}(m, n) = \mathbf{T_R}^*(M-m, n), \quad \mathbf{T_C}(m, n) = \mathbf{T_C}^*(m, N-n) \quad (30)$$

then the discrete operation has the following symmetric operation

$$\mathbf{Y}(m, n) = \mathbf{Y}^*(M-m, N-n) \quad \text{if } \mathbf{X} \text{ is real.} \quad (31)$$

So, any discrete operations whose transform matrices are reversible and satisfy the constraints in (30) can be used to implement the phase key algorithm. There are infinite matrices that satisfy the constraints in (30). In other words, there are infinite possible discrete linear operations that can be used to implement the phase key algorithm. This reflects the flexibility of the phase key algorithm.

4-3 Other Generalizations of the phase key algorithm

Besides of the operations we use, many other parts of the phase key algorithm in Fig. 1 can be generalized. For example, at the output, instead of taking the real part, we can take the linear combination of 'real part' and 'imaginary part':

$$f(x, y) = c_1\mathrm{Re}[\tilde{f}(x, y)] + c_2\mathrm{Im}[\tilde{f}(x, y)]. \quad (32)$$

In this case, the phase key function $q(w, s)$ is determined from:

$$q(w, s) = \cos^{-1}[|F(w, s)|/A(w, s)/(c_1^2+c_2^2)^{1/2}] + \arg[F(w, s)] + \tan^{-1}(c_2/c_1). \quad (33)$$

In addition, in Fig. 1, we can place a random mask function $M(w, s)$ before doing the IFT. In this case, $F(w, s)$ in (9) is changed into $F(w, s)M(w, s)$. We can also place a real mask $m(x, y)$ after $\tilde{f}(x, y)$. In this case, $F(w, s)$ in (9) is changed into $F(w, s) \otimes M(w, s)$ where $M(w, s) = FT[m(x, y)]$ and $\otimes$ means convolution.

Thus, the phase key algorithm is very flexible. If we use the phase key algorithm for encryption, the following parts of it can be treated as the keys:

j Phase function $q(w, s)$

k The amplitude spectrum $A(w, s)$.

l The operation we use and its parameters.

The possible choices are FTs, FRFTs, LCTs, the discrete linear operations that satisfy (30), ..……....., etc. If we use the FRFT, the parameter a can also be treated as the key.

m Whether the real part, or the linear combination of the real part and imaginary part of the output is treated as the desired image.

n Random mask function (if it has been used).

† If we use the block diagram in Fig. 3, the parameters t, s and the input image $f(x, y)$ can also be treated as the keys.

To recover the output image g(x, y) from f(x, y), one must know all of the above keys. If one of the keys is unknown, the desired image can't be recover. Since many parts of the phase key algorithm can be treated as keys, it is very effective and secure for image encryption.

5. PARALLEL, SERIES, AND TREE STRUCTURES

In Figs. 1, 3, we use phase key algorithm to encrypt one or two images. To encrypt a series of image, we can design a proper structure to encrypt them effectively. The two basic ways for multiple images encryption are parallel encryption (Fig. 8) and series encryption (Fig. 9). When using the parallel structure, if one of the phase keys is wrong, only the corresponding image can't be recovered. When using series structure, if the phase key $q_n(w, s)$ is wrong, then not only Image n but also Images n+1, n+2, ….., N can't be recovered. The parallel structure spends less retrieval time to recover the desired image, however, the series structure has higher security.

Besides, we can combine the parallel and series structures together, and design the 'tree' structure. The tree structure combines the advantages of parallel structure and series structure. In Fig. 10, we show an example that uses phase key algorithm to encrypt five images by tree structure.

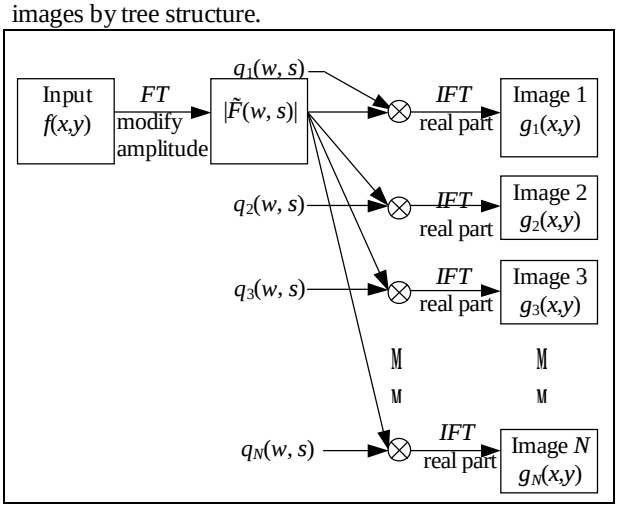


Fig. 8 Parallel structure of phase key algorithm.

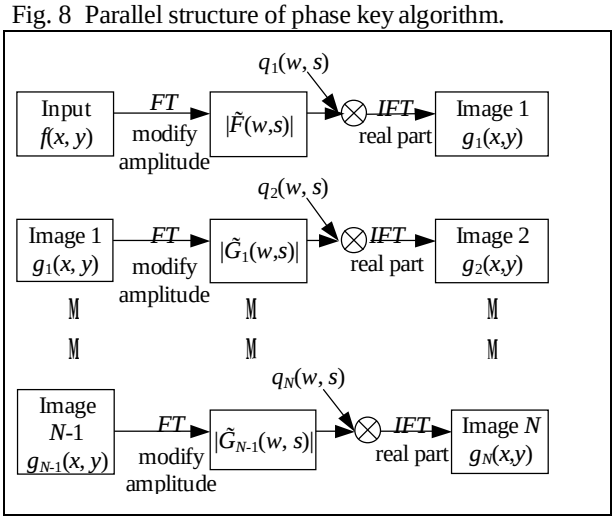


Fig. 9 Series structure of phase key algorithm.

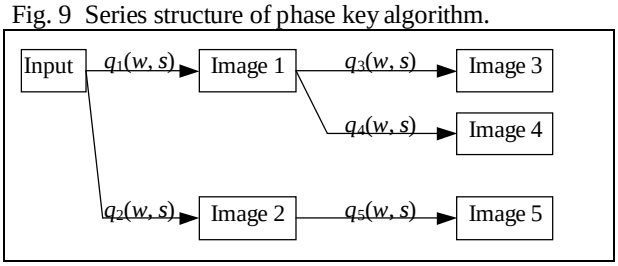


Fig. 10 Tree structure for multiple image encryption.

6. AMPLITUDE KEY ALGORITHM

In Sec. 2, we have described the way to make two real natural images have the same amplitude spectrum but different phase spectra by appending proper imaginary parts. We call it the phase key algorithm. In this section, we reverse the direction. For any two natural images, $f(x, y)$ and $g(x, y)$, we try to convert them into $\tilde{f}(x, y)$ and $\tilde{g}(x, y)$ by appending proper imaginary parts such that in the frequency domain the spectrum of $\tilde{f}(x, y)$ and $\tilde{g}(x, y)$ have the same phase but different amplitudes:

$f(x, y)$, $g(x, y)$: two real natural images

$\tilde{f}(x, y) = f(x, y)$ + imaginary part,

$\tilde{g}(x, y) = g(x, y)$ + imaginary part (34)

$\tilde{F}(w, s) = FT[\tilde{f}(x, y)]$ $\tilde{G}(w, s) = FT[\tilde{g}(x, y)]$

$\arg[\tilde{F}(w, s)] = \arg[\tilde{G}(w, s)] = q(w, s)$ (35)

$|\tilde{F}(w, s)| = A_f(w, s)$ $|\tilde{G}(w, s)| = A_g(w, s)$. (36)

We call it **the amplitude key algorithm**. Its block diagram is plotted in Fig. 11.

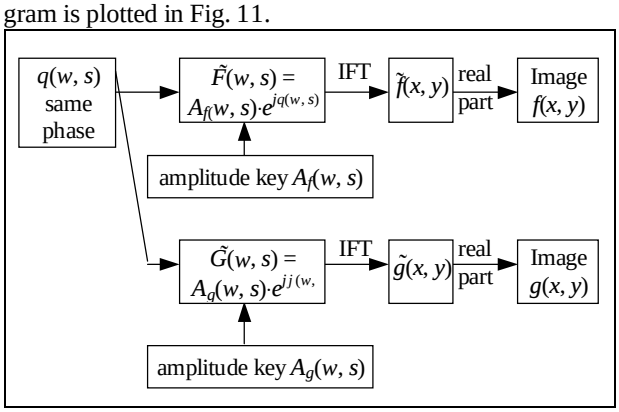


Fig. 11 Block diagram of the amplitude key algorithm

From the conventional concept of digital signal processing, the phase in the frequency domain is crucial to distinguish two different signals. Thus, it seems incredible that we can make two functions whose real parts are two different natural images have the same phase in the frequency domain. However, we indeed can do this. If the phase functions in Fig. 13 satisfies the following constraint:

$q(w, s) \neq -q(-w, -s)$ for all w, s, (37)

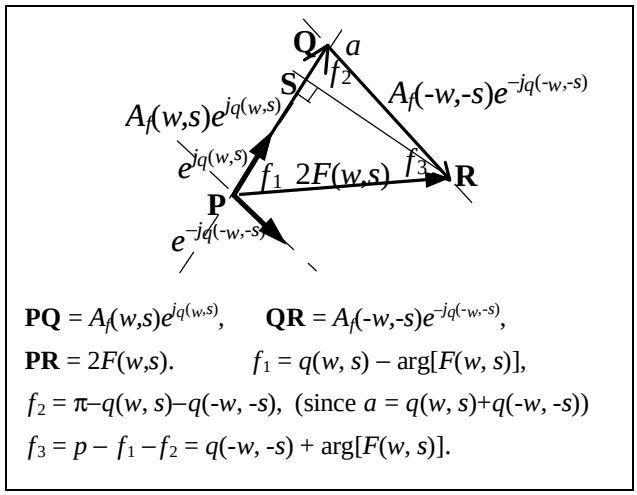
we can use the following way to determine the amplitude key $A_f(w, s)$ corresponding to $f(x, y)$ and the amplitude key $A_g(w, s)$ corresponding to $g(x, y)$.

Since $\tilde{F}(w, s) = A_f(w, s)\cdot e^{jq(w, s)}$, and $f(x, y) = \mathrm{Re}[\tilde{f}(x, y)]$, so

$F(w, s)=[\tilde{F}(w,s)+\tilde{F}^*(-w,-s)]/2$ $(w, s) = FT[f(x, y)]$ (38)

$2F(w, s) = A_f(w, s)\cdot e^{jq(w, s)} + A_f(-w, -s)\cdot e^{-jq(-w, -s)}$. (39)

If $q(w, s) \neq -q(-w, -s)$, the above equation can be plotted as in Fig. 12,



PQ = $A_f(w,s)e^{jq(w,s)}$, **QR** = $A_f(-w,-s)e^{-jq(-w,-s)}$,

PR = $2F(w,s)$. $f_1 = q(w, s) - \arg[F(w, s)]$,

$f_2 = \pi - q(w, s) - q(-w, -s)$, (since $a = q(w, s)+q(-w, -s)$)

$f_3 = p - f_1 - f_2 = q(-w, -s) + \arg[F(w, s)]$.

Fig. 12 Solve the amplitude key function $A_f(w, s)$.

From Fig. 12, we can solve $A_f(w, s)$ from

$|\mathbf{PQ}| = |\mathbf{PS}| + |\mathbf{SQ}| = |\mathbf{PR}|\cdot\cos f_1 + |\mathbf{PR}|\cdot\sin f_1\cot f_2$,

$A_f(w,s) = 2|F(w,s)|\cdot[\cos f_1+\sin f_1\cot f_2] =$

$=2|F(w,s)|\cdot[\cos f_1\sin f_2+\sin f_1\cos f_2]/\sin f_2$

$= 2|F(w,s)|\cdot\sin(f_1+f_2)]/\sin f_2$. (40)

Thus we obtain:

$$A_f(w,s) = 2|F(w,s)|\frac{\sin[\arg(F(w,s))+q(-w,-s)]}{\sin[q(w,s)+q(-w,-s)]}. \quad (41)$$

Similarly,

$$A_g(w,s) = 2|G(w,s)|\frac{\sin[\arg(G(w,s))+q(-w,-s)]}{\sin[q(w,s)+q(-w,-s)]}. \quad (42)$$

We can use (41) and (42) to solve the amplitude key functions $A_f(w, s)$ and $A_g(w, s)$.

Since in the frequency domain, the phase spectra of $\tilde{f}(x, y)$ and $\tilde{g}(x, y)$ are the same, we can use the amplitude spectra $A_f(w, s)$ and $A_g(w, s)$ to represent each of the natural images, $f(x, y)$ and $g(x, y)$. As the phase key algorithm, we can also use the amplitude key algorithm for image encryption, data compression, and image transformation.

Notice that the only constraint of the common phase spectrum $q(w, s)$ used in the amplitude key algorithm is (37). That is, any non-odd phase function can be applied. Thus, the amplitude key algorithm is very flexible. We can choose $q(w, s)$ as a constant C where $C \neq Np$, (if $C \neq Np$, $q(w, s)$ is not an odd phase function), or choose it as

$q(w, s) = pS(w, s)$ (43)

where $S(w, s)$ is any non-odd phase function satisfying $0 < S(w, s) < 1$We can even choose it from another natural image. Although from the constraint in (37), we can't choose $q(w, s)$ the same as the phase spectrum of some natural image, however, we can choose it as the 'offset form' of the phase spectrum:

$q(w, s) = \arg[H(w, s)] + f$ (44)

where $f \neq Np/2$, $H(w, s) = FT[h(x, y)]$, and $h(x, y)$ is some

natural image. In this case, we use the amplitude key algorithm to convert an image $h(x, y)$ into another image $f(x, y)$. The amplitude function $A_f(w, s)$ and the parameter f can be treated as the keys for image conversion. We can generalize it into the parallel, series, and tree structures (similar to those in Figs. 8, 9, and 10) for multiple images encryption by the amplitude key algorithm.

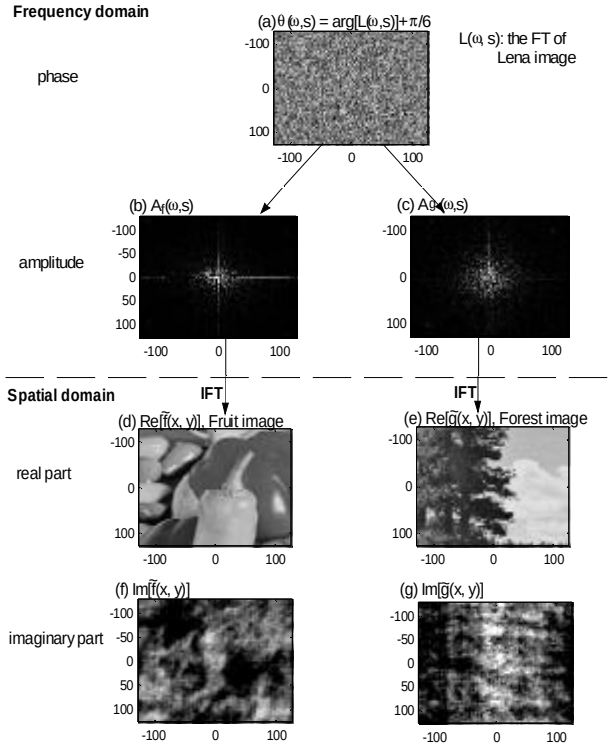


Fig. 13 Experiments of the amplitude key algorithm.

In Fig. 13, we perform an experiment that uses the amplitude key algorithm to do image encoding and encryption. We choose the phase spectrum of Lena image with some offset as the common phase functions $q(w, s)$ (shown in Fig. 13(a)):

$$q(w, s) = \arg[L(w, s)] + p/6, \quad L(w, s) = \mathrm{FT}[l(x, y)],$$
$$l(x, y): \text{Lena image.} \tag{45}$$

Then, we use (41) and (42) to calculate the amplitude key functions $A_f(w, s)$ and $A_g(w, s)$ (plotted in Fig. 13(b)(c)). After doing the IFTs for $A_f(w, s)\exp[jq(w, s)]$ and $A_g(w, s)\exp[jq(w, s)]$, and taking the real parts, we obtain the desired image, Fruit and Forest (shown in Fig. 13(d)(e)). Although in the spatial domain, the following three signals are quite different:

$$\begin{aligned} &\text{Lena image} \times \exp(j\pi/6), \\ &\text{Fruit image + imaginary part} \quad \text{(Fig. 13(d)(f))} \\ &\text{Forest image + imaginary part} \quad \text{(Fig. 13(e)(g))} \end{aligned} \tag{46}$$

however, in the frequency domain, their phase spectra are all the same (Fig. 13(a)).

7. CONCLUSION

In this paper, we introduce two image algorithms, the phase key and the amplitude key algorithms. The phase key algorithm modifies a set of images by appending proper imaginary parts and makes them have the same amplitude spectrum in the frequency domain. Since the only difference among their spectra is the phase, we can use the phase spectrum to encode each of the images. The phase key algorithm is very flexible and easy to design and implement. It provides an alternative way for image encoding. Moreover, we also introduce the amplitude key algorithm. It is similar to the phase key algorithm, but the direction is reverse. It makes the modified image have the same phase spectrum in the frequency domain, and the amplitude spectrum can be used to distinguish and encode each of the images. Both the two algorithms are helpful for image encryption, transformation, and data compression.

8. REFERENCES

[1] A. V. Oppenheim and J. S. Lim, 'The importance of phase in signals', *Proc. IEEE*, vol. 69, no. 5, p. 529-541, May 1981.

[2] A. W. Lohmann, D. Mendlovic, and G. Shabtay, 'Significance of phase and amplitude in the Fourier domain', *J. Opt. Soc. Am. A*, vol.14, no. 11, p. 2901-2904, Nov. 1997.

[3] G. Z. Yang, B. Z. Dong, B. Y. Gu, J. Y. Zhuang, and O. K. Ersoy, "*Gerchberg-Saxton and Yang-Gu algorithms for phase retrieval in a nonunitary transform system: a comparison*", Appl. Opt., Vol. 33, pp. 209-218, 1994.

[4] W. X. Cong, N, X. Chen, and B. Y. Gu, "*Recursive algorithm for phase retrieval in the fractional Fourier transform domain*", Appl. Opt., vol. 37, pp. 6906-6910, 1998.

[5] H. M. Ozaktas, Z. Zalevsky, and M. A. Kutay, "*The Fractional Fourier Transform with Applications in Optics and Signal Processing*", John Wiley & Sons, 2000.

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

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

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

[9] S. C. Pei and J. J. Ding, "Two-dimensional affine generalized fractional Fourier transform", *IEEE Trans. Signal Processing*, vol. 49, no. 4, pp. 878-897, Apr. 2001.