Decomposition of 2D Hypercomplex Fourier Transforms into Pairs of Complex Fourier Transforms

Todd A. Ell
5620 Oak View Court,
Savage, Minnesota, USA
Email: t.ell@ieee.org

Stephen J. Sangwine
The University of Reading
Whiteknights, Reading, RG6 6AY, UK
Email: s.sangwine@ieee.org

ABSTRACT

Hypercomplex 2D Fourier transforms have been proposed by several authors with applications in image processing of both greyscale and colour images. Previously published works on hypercomplex Fourier transforms have utilized direct evaluation of a Fast Fourier transform using hypercomplex arithmetic. This paper shows that such transforms may be implemented by decomposition into two independent complex Fourier transforms and may thus be implemented by building upon existing complex code. This is a significant step because it makes available to researchers using hypercomplex Fourier transforms all the investment made by others in efficient complex FFT implementations, and requires substantially less effort than coding hypercomplex versions of existing code.

1 Introduction

The authors have previously published definitions of Fourier transforms based on hypercomplex numbers (or quaternions) and shown their applicability to colour images, thus allowing generalization of many image processing techniques dependent on Fourier transforms to colour [1, 2, 3, 4, 5]. They have also recently demonstrated the validity of hypercomplex auto- and cross-correlation [6].

The authors’ work has employed hypercomplex pixels to represent colour images as input to the Fourier transform. In addition, Bülow and Sommer have published work on hypercomplex Fourier transforms applied to greyscale images [7, 8, 9] and their implementation is described in a technical report [10] which reveals that the input image is of real-valued pixels. Bülow and Sommer utilise a hypercomplex Fourier transform because of its symmetry properties when applied to real-valued images.

All of these previous implementations of hypercomplex transforms in discrete form have been direct codings of the transform using standard FFT algorithms implemented in quaternion arithmetic. While this is not difficult, it hinders wider acceptance of the transforms because implementations are not widely available. (The authors’ implementation and that of Felsberg and Bülow [10] is limited to images which are $2^N$ pixels square, for example, whereas many commercial and public domain FFT implementations can handle arbitrary composite (and sometimes also prime) sizes of image.) In this paper we present for the first time a decomposition of a hypercomplex Fourier transform into two complex Fourier transforms, and thus make the hypercomplex transform widely available to researchers and implementors with the minimum of effort using existing commercial code, or indeed mathematics packages such as Matlab, Mathematica, or Maple.

2 Quaternions

The quaternions were discovered by Hamilton in 1843 [11]. They combine by the normal rules of algebra with the exception that multiplication is not commutative. A quaternion has four components, one real and three imaginary. The usual notation, extended from that of the complex numbers is:

$$q = w + xi + yj + zk$$

where $w, x, y$ and $z$ are real, and $i, j$ and $k$ are complex operators which obey the following rules:

$$i^2 = j^2 = k^2 = ijk = -1$$
$$ij = k \quad jk = i \quad ki = j$$
$$ji = k \quad kj = i \quad ik = -j$$

The pattern of signs in the products of different operators is easily remembered if the operators $i, j$ and $k$ are imagined on a clock face, arranged clockwise in alphabetical order. Multiplication of any pair of operators in a clockwise sequence produces a positive product, while multiplication in anti-clockwise sequence yields a negative product.

The quaternion conjugate is $\overline{q} = w - xi - yj - zk$ and the modulus of a quaternion is given by:

$$|q| = \sqrt{w^2 + x^2 + y^2 + z^2}$$

A quaternion with zero real part is called a pure quaternion, and a quaternion with unit modulus is called a unit quaternion. The imaginary part of a quaternion has three components and may be associated with a 3-space vector. For this reason, it is sometimes useful to consider the quaternion as composed of a vector part and a scalar part, thus: $q = S(q) + V(q)$, where the scalar part, $S(q)$, is the real part
(w in our notation above), and the vector part is a composite of the three imaginary components, \( V(q) = xi + yj + zk \).

The product of two quaternions expressed in terms of their scalar and vector parts is given by:

\[
qr = S(q)S(r) - V(q) \cdot V(r) + S(q)V(r) + S(r)V(q) + V(q) \times V(r)
\]

where \( \cdot \) and \( \times \) denote the vector dot and cross products respectively. It follows from this that the dot and cross products of two pure quaternions, \( u \) and \( v \) are given by:

\[
u \times v = \frac{1}{2}(uv - vu)
\]

\[
u \cdot v = -\frac{1}{2}(uv + vu)
\]

**3 Quaternion Fourier transform**

Two formulations of a quaternion Fourier transform have been published to date. The first, by Ell in 1992 [1, 2] was as follows:

\[
H[j\omega, k\nu] = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{-j\omega t} h(t, \tau) e^{-k\nu \tau} dt \, d\tau
\]

with inverse defined as:

\[
h(t, \tau) = \frac{1}{4\pi^2} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} e^{j\omega t} H[j\omega, k\nu] e^{k\nu \tau} d\nu \, d\omega
\]

A similar transform with a different pair of complex operators (\( i \) and \( j \)) was published independently by Bülow and Sommer [7, 8, 9]. Ell’s transform was the basis of the first application of quaternion Fourier transforms to colour images by Sangwine in 1996 [3] using pure quaternions to represent colour image pixels in the RGB colour space: \( f(m, n) = f(r(m, n)i + g(m, n)j + b(m, n)k \). Subsequently, the following transform pair was presented by Sangwine and Ell in 1998 [5]:

\[
F(v, u) = S \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} e^{-\mu^2\pi(\frac{m}{M} + \frac{n}{N})} f(m, n)
\]

\[
f(m, n) = S \sum_{v=0}^{M-1} \sum_{u=0}^{N-1} e^{\mu^2\pi(\frac{v}{M} + \frac{u}{N})} F(v, u)
\]

where \( S = 1/\sqrt{MN} \) and \( \mu \) is any unit pure quaternion. Note that \( \mu^2 = -1 \), so that this transform can be seen as a generalization of the standard complex Fourier transform in which the imaginary operator has been generalized to a vector imaginary operator \( \mu \). Indeed, if \( \mu = i \) and the input image is complex valued in the conventional sense (that is all pixels have values \( a + ib \)), then this transform reduces to the complex Fourier transform.

One significant difference between this hypercomplex transform and the standard complex transform is that, because multiplication is not commutative, the hypercomplex transform exists in two versions, which we call *transposes*, each obtained from the other by reversing the order of the product of the hypercomplex exponential and the function (image) to be transformed. We have shown already in [5] that the existence of the transpose transform plays an important role in generalizing at least one operational formula from the complex case to the quaternion case: the property of image reversal by conjugation of its transform requires the forward and inverse transforms to be transposes.

The choice of \( \mu \) which we call the *axis* of the transform, is arbitrary. For reasons of symmetry we usually choose \( \mu \) to correspond to the so-called ‘grey line’ in RGB space, which connects all points \( r = g = b \). Thus the value of \( \mu \) which we usually use is \( (i + j + k)/\sqrt{3} \) (recall that \( |\mu| \) must be unity, which is why the factor \( 1/\sqrt{3} \) appears).

**4 Symplectic decomposition**

In the next section we present a decomposition of the hypercomplex Fourier transform given by equation 3 above, using a notion which we call *symplectic* decomposition. The result of this decomposition is that the hypercomplex transform may be computed using two standard complex transforms.

Decomposition of a hypercomplex Fourier transform into two complex Fourier transforms is dependent on several mathematical ideas starting from the Cayley-Dickson form of a quaternion [1, p7] in which a quaternion is represented by a complex number with complex real and imaginary parts, the two types of complex number having different and orthogonal complex operators. For example:

\[
q = a + bj, \text{ where } a = w + xi, \text{ and } b = y + zi
\]

so that:

\[
q = (w + xi) + (y + zi)j
\]

where it will be seen on multiplying out (using the rules for combining the orthogonal operators \( i \) and \( j \) that the result is:

\[
q = w + xi + yj + zk
\]

Now, using the idea of a generalized complex operator \( \mu \), as employed in the hypercomplex Fourier transform of equation 3, it is possible to generalize this idea. If we take two arbitrary unit pure quaternions, \( \mu_1 \) and \( \mu_2 \), such that \( \mu_1 \perp \mu_2 \), we can represent an arbitrary quaternion as a generalized complex quantity in what we call the *symplectic* form:

\[
q = a + b\mu_2 \text{, where } a = w' + x'\mu_1, \text{ and } b = y' + z'\mu_1
\]

so that:

\[
q = (w' + x'\mu_1) + (y' + z'\mu_2)\mu_2
\]

We call the two parts of the quaternion in this form the *simplex* and *perplex* parts respectively, that is the simplex part...
is a and the perplex part is b, both isomorphic to the standard complex numbers, since they are in the same generalized complex operator space. On multiplying out it will be seen that:

\[ q = w' + x'\mu_1 + y'\mu_2 + z'\mu_3 \quad (5) \]

where \( \mu_3 = \mu_1\mu_2 \) and further, \( \mu_3 \perp \mu_1 \) and \( \mu_3 \perp \mu_2 \). We thus have a system of operators isomorphic to the standard quaternion operators \( i, j \) and \( k \), but not constrained to the coordinate axes of 3-space. Decomposition of a quaternion \( q = w + xi + yj + zk \) into symplectic form about a given pair of axes \( \mu_1 \) and \( \mu_2 \) consists of solving for the real values \( w', x', y', \text{ and } z' \). These are most concisely (and elegantly) expressed using the scalar and vector parts of \( q \) as follows:

\[
\begin{align*}
w' &= S(q) \\
x' &= \frac{1}{2}(V(q)\mu_1 + \mu_1 V(q)) \\
y' &= \frac{1}{2}(V(q)\mu_2 + \mu_2 V(q)) \\
z' &= -\frac{1}{2}(V(q)\mu_1\mu_2 + \mu_1\mu_2 V(q))
\end{align*}
\]

as may be seen by substitution into equation 5.

The symplectic decomposition resolves a quaternion into two perpendicular planes which intersect only at the origin (in 4-space). Each of these planes may be thought of as an Argand plane. One is the Argand plane of the simplex part and has a real axis identical to the scalar axis of quaternion space, and an imaginary axis \( \mu_1 \). The other plane is the Argand plane of the perplex part and is perpendicular to both axes of the Argand plane of the simplex part. Its real axis is \( \mu_2 \) and its imaginary axis is \( \mu_3 \).

If we use the axis \( \mu_1 = (i + j + k)/\sqrt{3} \) discussed in section 3, it happens that when we decompose a colour image into components parallel to and perpendicular to the axis, we are in fact decomposing the image into luminance and chrominance components. The simplex part will be purely imaginary, luminance requiring only one real number. The perplex part will be complex, chrominance requiring two real numbers.

The symplectic decomposition of a quaternion image consists of decomposing each pixel in turn. The result is a pair of complex images which may be stored interleaved using the same space as the quaternion image.

5 Transform decomposition

The previous section has shown how we may decompose an arbitrary quaternion into two orthogonal components, one parallel to an arbitrary axis and one perpendicular to this axis, using symplectic decomposition. The hypercomplex Fourier transform itself may be decomposed into components parallel to and perpendicular to its own axis \( \mu \). If we choose the axis of the transform and the axis of decomposition of an image to be the same, and use the same perpendiculars to this axis in both cases, we find that the quaternion Fourier transform may be implemented using two complex Fourier transforms.

Consider equation 3 and write the image function to be transformed in symplectic form thus:

\[ f(m, n) = c_1(m, n) + c_2(m, n)\mu_2 \]

where \( c_1(m, n) = w'(m, n) + \mu_1 x'(m, n) \) and \( c_2(m, n) = y'(m, n) + \mu_1 z'(m, n) \) (where for the sake of generality we include \( w'(m, n) \) even though it will be zero everywhere when representing an RGB image). Now rewrite equation 3 using this symplectic form for \( f(m, n) \) and using the same unit pure quaternion, \( \mu_1 \), for the transform axis and the ‘inner’ operator of the symplectic form:

\[
F(v, u) = S \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} e^{-\mu_1 \frac{2\pi}{N} \left( \frac{nu}{N} + \frac{mu}{N} \right)} \left( c_1(m, n) \right)
+ (S \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} e^{-\mu_1 \frac{2\pi}{N} \left( \frac{nu}{N} + \frac{mu}{N} \right)} c_2(m, n)) \mu_2
\]

where:

\[
C_1(v, u) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} e^{-\mu_1 \frac{2\pi}{N} \left( \frac{nu}{N} + \frac{mu}{N} \right)} c_1(m, n)
\]

where \( i \in \{1, 2\} \), which is isomorphic to the standard complex 2D Fourier transform and may be computed using complex code.

It remains to present a method for finding a unit pure quaternion perpendicular to the transform axis.

6 Generation of \( \mu_2 \) from \( \mu_1 \)

The transform axis \( \mu_1 \) defines a plane in 3-space normal to the axis. Any unit pure quaternion in this plane is a valid choice for \( \mu_2 \). We can obtain a choice for \( \mu_2 \) by making an arbitrary choice of a third pure quaternion \( p \) which is not parallel to \( \mu_1 \). (\( p \) need not have unit modulus.) The cross product between the arbitrary quaternion \( p \) and \( \mu_1 \) must be perpendicular to \( \mu_1 \) (and \( p \)). This satisfies one of the constraints on \( \mu_2 \) (its direction). The remaining constraint (unit modulus) is easily satisfied by normalising the cross product. Thus:

\[
\mu_2 = U(\mu_1 \times p) = U \left( \frac{1}{2} (\mu_1 p - p\mu_1) \right)
\]

where \( U(q) = q/|q| \). Note that if we re-order the cross-product to \( p \times \mu_1 \) we will obtain \(-\mu_2 \) which is equally valid (it points in the opposite direction).
We may choose \( p \) so that one of the three components of \( \mu_2 \) is zero, reducing the number of multiplications needed when we reconstruct a quaternion from its symplectic form, or decompose a quaternion into its symplectic form, as shown in section 4. To achieve this, we select one of the three unit vectors \( i, j, k \) for \( p \).

If \( \mu_1 = (i + j + k)/\sqrt{3} \) and we wish \( \mu_2 \) to have zero component in the direction of \( i \), we have:

\[
\mu_1 \times i = \frac{1}{2\sqrt{3}}((i + j + k)i - i(i + j + k)) = \frac{1}{\sqrt{3}}(j - k)
\]

Dividing this result by its modulus, \( \sqrt{\frac{2}{3}} \), we obtain:

\[
\mu_2 = \frac{(j - k)}{\sqrt{2}}
\]

Using a theorem due to Coxeter [12], which states that two unit pure quaternions are perpendicular if the scalar part of their product is zero, we can verify that \( \mu_1 \perp \mu_2 \) by evaluating this product:

\[
\frac{(i + j + k)(j - k)}{\sqrt{3}} = \frac{1}{6}(ij - ik + j^2 - jk + kj - k^2) = \frac{1}{6}(j + k)
\]

and observing that it has zero scalar part, as required.

7 Conclusions

We have shown that a hypercomplex Fourier transform, may be implemented by decomposition into two transforms isomorphic to the complex Fourier transform, and therefore computable using standard complex FFTs. The significance of this decomposition is that it is no longer necessary to implement hypercomplex Fourier transforms using specially written FFT code, as has been done in the past.

The generalized symplectic decomposition of a quaternion into simplex and perplex parts with generalized operators isomorphic to the three complex operators of quaternions is also a significant new step.

The results presented in this paper add to a growing body of theory regarding hypercomplex Fourier transforms which indicates significant potential for new developments in image processing, particularly of colour, and other vector, images.

References


