16th European Signal Processing Conference (EUSIPCO 2008), Lausanne, Switzerland, August 25-29, 2008, copyright by EURASIP

# THEORY OF VECTOR FILTERS BASED ON LINEAR QUATERNION FUNCTIONS

*Todd A. Ell*†, *Stephen J. Sangwine*‡

† 5620 Oak View Court,
Savage, Minnesota 55378, USA
email: T.Ell@IEEE.org

‡ Department of Computing and Electronic Systems,
University of Essex, Wivenhoe Park,
Colchester CO4 3SQ, UK
email: S.Sangwine@IEEE.org

## ABSTRACT

Vector filtering of signals and images has many applications, but there is little theoretical framework underpinning rather *ad-hoc* approaches to the development of such filters. In this paper we make a significant step towards improving this position by showing that the geometric operations possible on samples or pixels can be expressed in a canonic form. In the formalism of quaternions, this canonic form has at most 4 quaternion coefficients. In the formalism of matrices and groups, the coefficients are $4 \times 4$ matrices, or members of the General Linear Group of order 4. We show how to combine series and parallel filters and reduce the result to the canonic form, and we discuss the set of geometric operations possible on vector samples or pixels, thus generalising the notion of sample scaling in classical DSP to a geometric concept of sample modification through linear operations.

## 1. INTRODUCTION

Vector filters have been studied for many years, particularly in the context of non-linear filtering, for example using order statistics (e.g. the classic vector median and related filters), but there have also been some linear vector filters published in the field of colour image processing. To date, the approaches taken to developing linear filters have been somewhat *ad-hoc* based on geometric operations in signal space or colour space without a sound theoretical framework. In this paper we present a theoretical framework for linear vector filters based on linear quaternion functions. This framework gives significant insight into the set of geometrical operations that can be exploited to generalise the notion of scaling a sample or pixel, and it also reveals a connection with the General Linear Group of order 4, $GL_4(\mathbb{R})$, that is the set of $4 \times 4$ matrices that transform 4-space vectors geometrically.

This paper is based on our previous work on linear quaternion filters applied to colour images. In these filters [1, 2], colour image pixels represented as pure quaternions were multiplied by left and right quaternion coefficients in a convolution. The use of left and right coefficients arises from the non-commutative nature of quaternion multiplication, and it allows geometric operations such as rotation, dilatation, reflection, *etc.* to be expressed using a quaternion difference equation. An obvious question with a non-obvious answer is whether, given a quaternion equation with an arbitrary number of first-order terms, it is possible to find a canonic form of the equation with some least upper bound on the number of coefficients. In signal or image processing terms what this means is, can an arbitrary cascade (series), or parallel, combination of such filters be reduced to a canonic minimal filter? It turns out that the answer to this question, as we show in this paper, is affirmative, and the maximum number of quaternion coefficients needed is 4.

The starting point for the theory developed in this paper is the discovery in 2007 by Ell [3] that a linear quaternion function has at most 4 quaternion coefficients when expressed in a canonic form. Therefore, any linear vector filter based on quaternion coefficients, *or a series or parallel combination of such filters*, can be reduced to a sum of four convolutions.

The paper is structured as follows. In §2 we review quaternions and the use of quaternions for representing points in projective space. In §3 we review the result that linear quaternion functions can be reduced to a canonic form with at most four quaternion coefficients, show how addition or composition of such functions (equivalent to series and parallel combinations of filters) may be calculated to yield an equivalent canonic filter, and show the equivalence of these functions to $GL_4(\mathbb{R})$. In §4 we present an overview of the set of geometric operations that are available using the canonic form. In §5 we present an initial discussion of the idea that coefficients in a filter 'mask' or impulse response may be linear quaternion systems (that is we regard the coefficients of a FIR filter as linear quaternion system operators). Finally, in §6 we discuss the practical issue of converting Cartesian data to (and from) homogeneous coordinates and the use of pre-filter point-wise linear quaternion operations to prepare data for convolution filtering.

## 2. PRELIMINARIES

### 2.1 Quaternions

This work uses the hypercomplex numbers of Hamilton [4], namely the quaternion 4-tuple $(w,x,y,z)$ denoted in hypercomplex form as: $q = w + xi + yj + zk$ where $w,x,y,z \in \mathbb{R}$. The hypercomplex operators follow the rule: $ijk = i^2 = j^2 = k^2 = -1$.

All quaternions, $q \in \mathbb{H}$, can be split into scalar and vector parts, i.e., $q = s + v$ where $s = w = S[q]$ is the scalar-part and $v = xi + yj + zk = V[q]$ is the vector-part. Conjugation is denoted with an over-bar which negates the vector part $\bar{q} = s - v$. It is usual to represent 3-space vectors as quaternions with zero scalar part; this is the set of so-called *pure* quaternions which is denoted $V[\mathbb{H}]$.

### 2.2 Quaternion Projective Space

Full quaternions, those with non-zero scalar part, can also be used to represent points in homogeneous coordinates. The quaternion identity

$$q = s + v = s[1 + v/s] \tag{1}$$

†Ell is a Visiting Fellow at the University of Essex. This work was supported in part by the U.K. Engineering and Physical Sciences Research Council under Grants GR/S58621/01 and EP/E010334/1.

illustrates this point. In this form $q$ can be used to represent a *point* located at the end of the vector $p = v/s$ from the origin with *weight* $s$. Thus in weighted-point form

$$q = s[1+p].$$

Hence the set of quaternions can be associated with the real projective space $\mathbb{P}^3$. This interpretation was discussed by Joly [5, pp.263–4] in 1905 and MacFarlane [6, p.35] in 1906.

Under this interpretation, a unit-weight point at the origin is denoted

$$q = 1[1+0i+0j+0k] = 1,$$

and a *weight-less* point at infinity in the $(x,y,z)$ direction is

$$q = [0+xi+yj+zk]$$

which can be seen by letting $s \to 0$ in equation (1). The resulting pure quaternion can also be viewed as a translation vector when added to a weighted-point.

## 3. LINEAR QUATERNION SYSTEMS

### 3.1 Quaternary Canonical Form

Real linear functions take the monomial form: $f(x) = mx$, where $x, m \in \mathbb{R}$. Linear combinations, i.e., direct sums or compositions, of such functions can always be reduced to this same form. By contrast quaternion linear functions have the multi-nomial form:

$$f(q) = \sum_{p=1}^{P} m_p q n_p, \tag{2}$$

where all factors are quaternion valued, i.e., $q, m_p, n_p \in \mathbb{H}$. One would expect under functional compositions that the number of terms in the summation can become arbitrarily large due to quaternion multiplication being non-commutative. However, Ell [3] showed in 2007 that this general multi-nomial can always be reduced to its *quaternary canonical form*

$$f(q) = Aq + Bqi + Cqj + Dqk, \tag{3}$$

where $A, B, C, D \in \mathbb{H}$. Section 3.3 explains how this reduction may be done.

So, just as a quaternion can be associated with a 4-tuple of reals as $(a,b,c,d) \Leftrightarrow a1+bi+cj+dk$, likewise the canonical linear quaternion *function* can be associated with a 4-tuple of quaternions as

$$\{A,B,C,D\} \Leftrightarrow Aq + Bqi + Cqj + Dqk.$$

The four-tuple of quaternions will be used as a shorthand notation for the canonical form.

### 3.2 Parallel & Series Combinations

The linear sum of two such functions is given by the component-wise addition. I.e., let

$$f_1(q) = A_1 q + B_1 qi + C_1 qj + D_1 qk,$$
$$f_2(q) = A_2 q + B_2 qi + C_2 qj + D_2 qk$$

then

$$f_1(q) + f_2(q) = A_3 q + B_3 qi + C_3 qj + D_3 qk$$

where

$$A_3 = A_1 + A_2, \quad B_3 = B_1 + B_2,$$
$$C_3 = C_1 + C_2, \quad D_3 = D_1 + D_2.$$

The composition of two linear functions, $f_2(f_1(q))$, is given by

$$f_2(f_1(q)) = A_3 q + B_3 qi + C_3 qj + D_3 qk$$

where

$$A_3 = A_2A_1 - B_2B_1 - C_2C_1 - D_2D_1,$$
$$B_3 = A_2B_1 + B_2A_1 - C_2D_1 + D_2C_1,$$
$$C_3 = A_2C_1 + B_2D_1 + C_2A_1 - D_2B_1,$$
$$D_3 = A_2D_1 - B_2C_1 + C_2B_1 + D_2A_1.$$

Careful examination of this composition rule reveals it has the same structure as the standard quaternion multiplication. The composition is, of course, not commutative: $f_2(f_1(q)) \neq f_1(f_2(q))$. This aligns with the fact that geometrical operations do not commute (the composition of two rotations, for example, gives different results, in general, according to the order in which the rotations are carried out).

### 3.3 Equivalence of Canonic Form and $GL_4(\mathbb{R})$

The equivalence between invertible functions in quaternary canonic form and the General Linear Group of $4 \times 4$ invertible real matrices, $GL_4(\mathbb{R})$, will be used later. This equivalence is shown in this section. Let $p = p_0 + p_1 i + p_2 j + p_3 k$ and $q = q_0 + q_1 i + q_2 j + q_3 k$ be two arbitrary quaternions. Using standard hypercomplex operator product rules, the components of the quaternion product $pq = q'_0 + q'_1 i + q'_2 j + q'_3 k = q'$ are

$$q'_0 = p_0q_0 - p_1q_1 - p_2q_2 - p_3q_3,$$
$$q'_1 = p_1q_0 + p_0q_1 - p_3q_2 + p_2q_3,$$
$$q'_2 = p_2q_0 + p_0q_2 + p_3q_1 - p_1q_3,$$
$$q'_3 = p_3q_0 + p_0q_3 - p_2q_1 + p_1q_2.$$

By gathering terms into matrix-vector notation

$$\begin{bmatrix} q'_0 \\ q'_1 \\ q'_2 \\ q'_3 \end{bmatrix} = \begin{bmatrix} p_0 & -p_1 & -p_2 & -p_3 \\ p_1 & p_0 & -p_3 & p_2 \\ p_2 & p_3 & p_0 & -p_1 \\ p_3 & -p_2 & p_1 & p_0 \end{bmatrix} \begin{bmatrix} q_0 \\ q_1 \\ q_2 \\ q_3 \end{bmatrix}$$

one can define an equivalent matrix-vector form for the quaternion product. Let $[\![p]\!]$ and $[q]$ denote the matrix and vector equivalences, respectively. Since the quaternion product is bi-linear, one can instead place the components of $p$ into the vector and the components of $q$ into the matrix, but in doing so the lower right $3 \times 3$ sub-matrix is transposed from the one given above. Denote this *transmuted* form of the matrix as $[\![q]\!]^\dagger$. Therefore, any quaternion product has two equivalent matrix-vector forms

$$pq \leftrightarrow [\![p]\!][q] = [\![q]\!]^\dagger [p].$$

Hence any triple-product $pqr$ may be arbitrarily re-ordered in matrix form as

$$pqr \leftrightarrow [\![p]\!][\![q]\!][r] = [\![p]\!][\![r]\!]^\dagger [q].$$

The linear quaternary equation $q' = Aq + Bqi + Cqj + Dqk$ can now be written in matrix-vector form as

$$[q'] = \left\{ [\![A]\!] + [\![B]\!][\![i]\!]^\dagger + [\![C]\!][\![j]\!]^\dagger + [\![D]\!][\![k]\!]^\dagger \right\} [q]. \tag{4}$$

Now, both a matrix $[\![p]\!]$ and a transmuted matrix $[\![q]\!]^\dagger$ contain only four degrees of freedom; one for each component of the quaternion. Careful examination of the four terms in the above matrix equation reveals that these matrices are independent, giving the total of sixteen degrees of freedom necessary for an arbitrary matrix. It is straightforward to start with an arbitrary matrix $\{r_{i,j}\}$ and solve for the elements of the four quaternions $\{A,B,C,D\}$. For details of this analysis see [3].

It follows from Equation 4 that any quaternion multinomial of the form given in Equation 2 may be reduced to the form in Equation 3 since the sum of the terms inside the braces in Equation 4 may be added across multiple terms in Equation 2 then solved once to find a single set of coefficients $A,B,C,D$. For details, see [3] and for a simple numerical method see [7].

## 4. PROJECTIVE GEOMETRIC TRANSFORMATIONS

### 4.1 Euclidean, Similarity, Affine & Perspective Transformations

We now discuss the various types of geometrical transformation possible using a linear quaternion system as presented in the previous section. We draw here on the classic text of Bruce Meserve [8] which clearly sets out the possibilities (although not using the quaternion formalism that we use here).

It is important to understand before we discuss the transformations that these are to be applied to signal samples or image pixels in the space of the samples. These are not transformations to be applied to an image for example, so that when we talk of rotation, we mean rotation of sample or pixel values, not rotation of the image about its coordinate or index system (in the image or focal plane). For example, in colour image processing, pixel rotation about the grey line $r = g = b$ changes the hue of a pixel, but not its saturation or luminance. In vector signal processing, an example is a signal representing vibration in mutually perpendicular directions. Each sample contains information at a sampling point about the amplitudes in each of three perpendicular directions. We may think of these sample/pixel transformations as a generalisation of the concept of *scaling*: in classical DSP with scalar sample values, the two most fundamental operations in linear time-invariant systems are scaling of the sample value by a constant, and delay of the sample value by an integral number of sample periods (time-shift). Here, with vector samples, we can obviously scale the samples, but we can also apply other linear operations to them, such as rotation about an axis, reflection about a plane, and so on. Thus we generalise the scaling concept to include the full set of linear geometric operations. This is what we are concerned with in this section.

Meserve [8, §5-11] presents a hierarchy of geometric transformations and Figure 1 depicts a simplified set of these transformations with various subsets.

The most general transformation is projective. The *affine* transformations are a subset of the *projective* transformations with the property that straight lines are preserved (that is, remain straight) under the transformation. Reflection, scaling and shear are examples of affine transformations. *Similarity* transformations are a subset of the affine transformations with the additional property that angles are preserved (thus geometric figures are transformed to similar geometric figures). Uniform scaling (that is, scaling along all axes by the same scale factor) has this property. *Euclidean* transformations are a subset of the similarity transformations. Translation and rotation are examples. They preserve angles as well as straight lines.

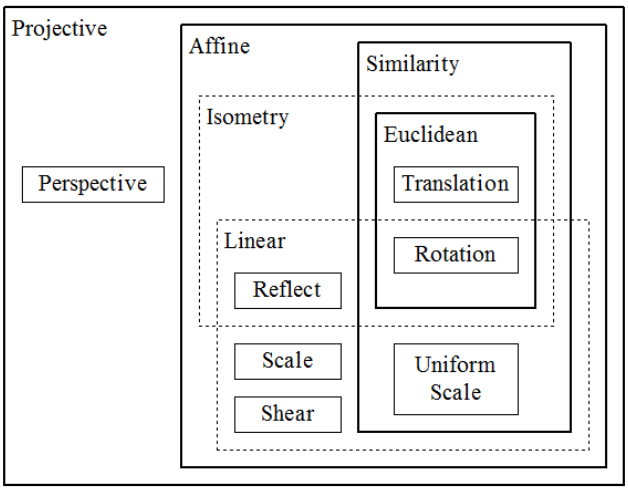


Figure 1: Hierarchy of transformations.

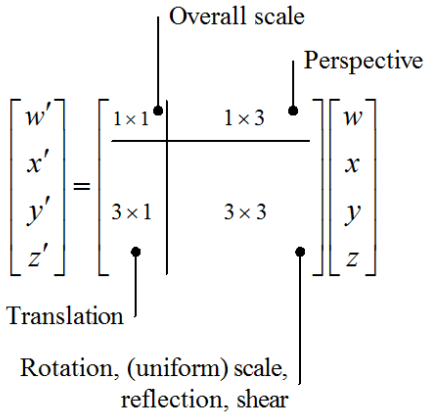


Figure 2: Transforms using $GL_4(\mathbb{R})$.

Two additional subsets of the affine transformations are shown in Figure 1 by dotted lines. These 'cut across' the other subsets. The *isometries* preserve distance between points. The *linear* transformations are those that obey superposition. This is usually well understood in the sense of signal processing – it means that a transformation applied to a sum of two signals will give the same result as applying the transformation to the two signals separately, and then summing the two results. In geometric terms it means we may decompose a vector or point into the sum of two or more vectors or points and obtain the same results by transforming the decomposed parts separately as by transforming the un-decomposed vectors or points.

Now a very significant point is that all the transformations shown in Figure 1 are linear when expressed in homogeneous coordinates, as described in §2.2, even if non-linear in the 3-dimensional Euclidean space of the signal samples

or image pixels. (It follows, of course, that the transformations between homogeneous coordinates and Euclidean coordinates is non-linear.)

The linearity in homogeneous coordinates follows from the equivalence between linear quaternion systems and the General Linear Group $GL_4(\mathbb{R})$ which we cannot cover in depth here. However, figure 2 illustrates the concept. This figure shows how the various transformations are realized using sub-matrices of a $4 \times 4$ matrix. There are many textbooks in computer vision and computer graphics that discuss the use of $GL_4(\mathbb{R})$ for projective space transformations. For example, see Ballard [9, pp. 477–480] or Yamaguchi [10].

## 5. LQS CONVOLUTION OPERATORS

In this section we explain how linear filters may be constructed using linear quaternion systems as the point operators (i.e. the operators that operate on individual samples of the signal or image being processed). We are assuming here linear time-invariant filters (signal processing) or linear shift-invariant filters (image processing) characterised by a finite impulse response, or a finite coefficient 'mask'. Making these assumptions, a filter can be represented as a convolution (we assume here a one-dimensional filter for simplicity – generalisation to two dimensions is simple):

$$y(n) = \sum_{m=1}^{N} h(m)x(n-m) = h * x$$

where $h(m)$ is the impulse response of the filter. In the case of vector signals and quaternion coefficients, we know from §3 that we need four products in the convolution:

$$\begin{aligned} y(n) &= \sum_{m=1}^{N} \begin{pmatrix} A(m)x(n-m) + B(m)x(n-m)i + \\ C(m)x(n-m)j + D(m)x(n-m)k \end{pmatrix} \\ &= A*x + B*xi + C*xj + D*xk \end{aligned}$$

where by $A(m)$ we mean the $m^{th}$ sample of $A$, and by $A$ we mean a finite quaternion-valued function with $N$ quaternion samples and similarly for $B$, $C$ and $D$. In this way, we think of the filter as consisting of the sum of four quaternion-valued convolutions, three of which are multiplied on the right by the constant values $i$, $j$ and $k$.

An alternative, and higher-level view of the filter is to consider it as the convolution of the signal $x$ with a finite linear quaternion function $F = A + Bi + Cj + Dk$, like this: $y = F * x$. Now, at each sample point in $F$, we have a linear quaternion function which implements some geometric operation, and we regard the filter as the convolution of these geometric operations with the vector signal. This is not simply a matter of notation, because we can now think of defining the frequency response of the filter – it would be the Fourier transform of $F$. (We do not know how this may be done!)

Given the preceding discussion, we may now address a more complex issue: that of filter design. There are two aspects to designing a vector filter which make the process non-trivial. The first problem, which we believe we can address using the linear quaternion system framework, is to define the set of possible geometric operations which may be applied to sample/pixel values. The second problem, which is more difficult, is to define a suitable impulse response for the filter in terms of these geometric operations. We can expand on this by way of an example: edge detectors (used mostly in image processing) operate by scaling samples either side of a putative edge in opposite ways so that samples either side of an edge will add, while those that are part of a smooth region of signal will cancel. Thus a filter with impulse response $h(n) = \{-1, 0, 1\}$ will detect sharp transitions in a signal which occur over three samples. For example, if successive samples are $a, a+\varepsilon, a+2\varepsilon$ (part of a smooth region, where $\varepsilon$ is a small change in signal value), then the filter output centred on these three samples will be $2\varepsilon$ (i.e. small). If however, successive samples are $a, a+50, a+100$, then the output of the filter centred on these samples will be 100, a large value. So, it should be clear that a filter is partially defined by the shape of its impulse response (the 'pattern' of values in the samples of $h(n)$). A vector filter, however, is also characterized by the particular arrangement of geometric operations across the filter impulse response. We can quote here a specific example by Sangwine in 1998 [1] which had the pattern of a classic image edge detector (rows of mask coefficients with opposing sign) but the geometric operation implemented was rotation, in different senses in the opposing rows or columns of the filter 'mask'. It should now be clear that the range of possibilities is enormous. We can devise patterns to the filter coefficients (the layout of significant values), and choose for each one a linear quaternion system of arbitrary geometric effect. We might imagine filters that twist in the sense that the signal values along the filter impulse response are rotated about some axis, the amount of rotation varying along the filter. Or perhaps the samples are stretched in some sense, again varying along the filter, before being summed to give an output. Thus we might detect signals with certain characteristics such as polarization in a specified sense.

We mention one further sublety that increases again the scope for defining interesting filters. In classical scalar signal processing, the scaling operation includes the limiting case where the scale factor is zero, as was mentioned above in the case of edge detectors. We have discussed the generalization of the scaling concept to geometric operations, and of course scaling by zero remains an option. However, a geometric generalization of this idea is projection of a vector onto a plane, line, or point. (Projection of a vector onto a point at the origin achieves the same effect as scaling by zero.) Projection onto a plane achieves a reduction from 3-dimensions to 2. Projection onto a line gives a reduction to 1-dimension. It may be that these operations can be usefully exploited in designing filters.

## 6. APPLICATION TO FILTER DESIGN

### 6.1 Signal or image encoding/decoding

This paper assumes that the signals or images to be processed have samples in a 3-dimensional signal space or colour space. The full power of the approach presented here, based on linear quaternion systems, requires that the samples or pixels be encoded into homogeneous coordinates. The encoding is a trivial process, since all it requires is a weight to be assigned to each sample or pixel. This can be done by arbitrarily assigning a weight of 1 to each sample. When represented in quaternion form, as described in §2.2, the weight is represented by the scalar part of the quaternion. So, for example, to encode an RGB pixel with components $(r, g, b)$ into a quaternion representation of the equivalent homoge-

neous coordinates, we simply encode the pixel as

$$1 + ri + gj + bk$$

The decoding process requires that all the pixels be normalised to the same weight. (The weights may vary as the result of filtering operations.) Therefore, taking the same example, we may have a pixel with value $w + r'i + g'j + b'k$. We must divide through by $w$ to obtain

$$1 + \frac{r'}{w}i + \frac{g'}{w}j + \frac{b'}{w}k$$

from which we can then obtain the pixel value

$$\left(\frac{r'}{w}, \frac{g'}{w}, \frac{b'}{w}\right)$$

by discarding the scalar part.

### 6.2 Pre- & Post-Filter Signal/Colour-Space Transforms

So far, we have not mentioned pre-processing or post-processing of the signals or images to be filtered. It is common in image processing to transform images from one colour space to another before processing. Many of these transformations are linear, and therefore expressible in terms of point-wise linear quaternion operators. This means that the transformation in and out of a colour space may be combined with a filtering operation, so that we may define a filter, say, with RGB input and output images, that nevertheless operates in effect in a different colour space such as $YC_bC_r$. It is possible in fact to apply translations in this way when working in homogeneous coordinates, so that offsetting the samples or removing an offset can also be combined with filtering (for example to convert from unipolar to bipolar samples).

### 6.3 Geometric operations

In a paper in 2007 [11], Ell formulated a number of affine transformations including reflections, rotations, shears and dilations (radial and axial), in quaternion terms. All of these operations were expressed using simple quaternion formulae which, not surprisingly, given the theoretical results in this paper, can be expressed as linear quaternion systems. Although these operations were designed in Cartesian coordinates (i.e., for vectors), many of them carry over to homogeneous coordinates without change. For example, radial dilations are a set of transformations that expand 3-dimensional space outwards from an invariant line defined by a unit quaternion $\mu$ using a scale factor $\alpha$. The transformation is represented by

$$D_{\alpha,\mu}(p) = \frac{2+\alpha}{2}p + \frac{\alpha}{2}\mu p \mu$$

where $\alpha > 0$ expands space, and $\alpha < 0$ compresses space. As a simple example of what this transformation achieves, if applied to an RGB encoded image, with the grey-line $r = g = b$ as the invariant line, it allows saturation to be increased or decreased without affecting hue or luminance. It is clear from the algebraic form of the transformation given above that it takes the form of Equation 2 and it may therefore be represented as a linear quaternion system as in Equation 3.

## 7. CONCLUSIONS

It has been shown in this paper that a very wide range of vector filters can be developed to operate on signals or images with samples or pixels in 3-dimensional signal or colour space. The paper has shown that linear quaternion systems operating on homogeneous coordinate representations of the samples or pixels provides a very general linear framework which is capable of representing very general geometric transformations, including many that are non-linear when expressed in the original 3-dimensional Euclidean coordinates of the samples or pixels. Thus the theory presented in this paper opens up a large set of possible filters which could be developed within a single mathematical framework. Most importantly, the paper suggests that projective transformations offer a new class of transformations that may be implemented in a linear framework using homogeneous coordinates.

## References


[1] S. J. Sangwine. Colour image edge detector based on quaternion convolution. *Electronics Letters*, 34(10): 969–971, 14 May 1998. doi: 10.1049/el:19980697.

[2] S. J. Sangwine and T. A. Ell. Colour image filters based on hypercomplex convolution. *IEE Proceedings – Vision, Image and Signal Processing*, 147(2):89–93, April 2000. doi: 10.1049/ip-vis:20000211.

[3] Todd A. Ell. On systems of linear quaternion functions. Preprint, February 2007. URL `http://www.arxiv.org/abs/math/0702084v1`.

[4] W. R. Hamilton. *Lectures on Quaternions*. Hodges and Smith, Dublin, 1853. Available online at Cornell University Library: `http://historical.library.cornell.edu/math/`.

[5] Charles J. Joly. *A manual of quaternions*. Macmillan, London, 1905. Available online at Cornell University Library: `http://historical.library.cornell.edu/math/`.

[6] Alexander MacFarlane. *Vector analysis and quaternions*. John Wiley and Sons, New York, 4th edition, 1906. Available online at Cornell University Library: `http://historical.library.cornell.edu/math/`.

[7] S. J. Sangwine. Canonic form of linear quaternion functions. Preprint, January 2008. URL `http://arxiv.org/abs/arXiv:0801.2887`.

[8] Bruce E. Meserve. *Fundamental Concepts of Geometry*. Dover, New York, 1983. ISBN 0-486-63415-9. Corrected reprint of 1959 edition.

[9] D. Ballard and C. M. Brown. *Computer Vision*. Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982. ISBN 0-13-165316-4.

[10] Fujio Yamaguchi. *Computer-Aided Geometric Design: A Totally Four-Dimensional Approach*. Springer, New York, NY, 2002. ISBN 4431703403.

[11] Todd A. Ell. Hypercomplex color affine filters. In *IEEE International Conference on Image Processing (ICIP 2007)*, volume 5, pages 249–252, San Antonio, TX, USA, 16–19 September 2007. Institute of Electrical and Electronics Engineers. ISBN 978-1-4244-1437-6. doi: 10.1109/ICIP.2007.4379812.