A GEOMETRIC ALGEBRA APPROACH TO SOME PROBLEMS OF ROBOT VISION

Gerald Sommer
Institut für Informatik und Praktische Mathematik
Christian-Albrechts-Universität zu Kiel, Kiel, Germany
gs@ks.informatik.uni-kiel.de

Abstract Geometric algebra has been proved to be a powerful mathematical language for robot vision. We give an overview of some research results in two different areas of robot vision. These are signal theory in the multidimensional case and knowledge based neural computing. In both areas a considerable extension of the modeling has been achieved by applying geometric algebra.

Keywords: Clifford algebra, Clifford spinor neuron, geometric algebra, hypersphere neuron, monogenic function, neural computing, Poisson scale-space, robot vision, signal theory.

1. Introduction

In this tutorial paper we present a survey of some results contributed by the Kiel Cognitive Systems Group to the applications of geometric algebra in robot vision. Geometric algebra makes available a tremendous extension of modeling capabilities in comparison to the classical framework of vector algebra. It is our experience that application of geometric algebra should be strictly controlled by the geometric nature of the problem being considered. To demonstrate that tight relation between the problem at hand, its algebraic formulation and the way to find solutions will be our principle matter of concern. We i will do that by considering some key problems in robot vision. More details of the results reported here can be found in the papers and reports of the Kiel Cognitive Systems Group at our website (http://www.ks.informatik.uni-kiel.de). In addition, an extended version of this paper will be available [Sommer, 2003].

1.1 Robot Vision

Robot vision is a demanding engineering discipline emerging from several contributing scientific disciplines. It aims at designing mobile technical systems which are able to take actions in its environment by using visual sensory information.

Although a matter of research for three decades, we are far away from having available seeing robots which are able to act comparable to humans in real world conditions. There are several reasons. First, many different disciplines such as image processing and signal theory, pattern recognition including learning theory, robotics, computer vision, and computing science are required for robot vision. Each of these have their own problems caused by limited modeling capabilities. Second, each of them has been developed to a great extent, isolated from the other ones, by using quite different mathematical languages of modeling. Thus, the fusion of all these disciplines within one framework is demanding by itself. Third, the most difficult problem is the design of the cognitive architecture. This concerns e.g. the gathering and use of world knowledge, controlling the interplay of perception and action, the representation of equivalence classes, invariants, and conceptions. Besides, such a system has to cope with hard real-time conditions.

The design of perception-action cycles cooperating and competing for solving a task is a demanding challenge. Of special interest is to enable the system to learn the required competence [Pauli, 2001] from experience. From a mathematical point of view the equivalence of visual and motor categories is remarkable. Both are mutually supporting [Sommer, 1997]. Of practical importance are the following two projections of a perception-action cycle. "Vision for action" means to control actions by vision and "action for vision" means the control of gaze for making vision easier.

1.2 Motivations of Geometric Algebra

Representing geometry in a general sense is a key problem of system design. But only those geometric aspects have to be represented which are of pragmatic importance. This opportunistic minimalism is tightly related to the so-called stratification of space as introduced into computer vision by Faugeras [Faugeras, 1995]. The system should be able to purposively switch from, e.g., metric to projective or kinematic aspects. We come back to that point in section 4 of [Sommer, 2003].

Another interesting topic is to ensure pragmatic completeness of the representations. But signal theory which supports image processing operations fails in that respect [Sommer, 1992, Zetzsche and Barth, 1990].

Local operations have to represent intrinsically multi-dimensional structure. The search of solutions for that problem was our original motivation for considering geometric algebra as a modeling language. We will survey our efforts in striving for a linear theory of intrinsically multi-dimensional signals in section 2.

Our aim of representing geometry in a general sense means thinking in a Kleinian sense, thus taking advantage of the tight relationship between geometry and algebra. All we have done in our work so far is based on choosing a useful geometry by embedding the problem into a certain geometric algebra. This is in fact a knowledge based approach to system design. In section 3 we will demonstrate this way of modeling in the context of neural computing. There we will profit from the chosen embedding because it results in a transformation of a non-linear problem to a linear one.

The problem of converting non-linear problems in Euclidean vector space to linear ones by embedding into a certain geometric algebra is related to another basic phenomenon which makes geometric algebra so useful in robot vision and beyond. From a geometrical point of view points are the basic geometric entities of a Euclidean vector space. Instead, a cognitive system is operating on geometric objects as a whole unique entity, e.g. a tea pot. An algebraic framework is wanted in which any object concept and transformations of it may be represented in a linear manner. Regrettably, this is an illusion. But geometric algebra enables the extension from point concepts to rather complex ones. In [Sommer, 2003] we demonstrate their linear construction and how to model their motion in a linear manner.

We will abstain from presenting a bird's eye view of geometric algebra in this contribution. Instead, we recommend the following introduction to geometric algebra [Hestenes et al., 2001]. In this paper we will use several geometric algebras as well as linear vector algebra. We will mostly write the product in the chosen algebra simply by juxtaposition of its factors. In some cases we will use a special notation for the chosen geometric product to emphasize its difference to a scalar product. Special products will be noted specially. We will use the notation $\mathbb{R}_{p,q,r}$ for the geometric algebra derived from the vector space $\mathbb{R}^{p,q,r}$ with $p+q+r=n$. These indices mark the signature of the vector space. Hence, (p,q,r) means we have $p/q/r$ basis vectors which square to $+1/-1/0$. If possible we reduce the set of signature indices, as in the case of the Euclidean vector space $\mathbb{R}^n$.

2. Local Analysis of Multi-dimensional Signals

Image analysis is a fundamental part of robot vision. We do not understand image analysis as interpreting the whole visual data with respect to the complete scene or recognizing certain objects mapped to an image. This in fact is subject of computer vision. Instead, the aim of image analysis is to derive features from visual data for further processing and analysis. Its theoretical basis in called signal theory.

In (linear) signal theory we find the framework for handling linear shift invariant operators (LSI operators) and spectral representations of both signals and operators which are computed by Fourier transform. Both are tightly related. Although both are widely used in image analysis, there is a serious problem with respect of supporting recognition of intrinsically multi-dimensional and especially two-dimensional structure. This is a problem of incomplete representation.

We have to distinguish between two different conceptions of dimensionality of image data. The first one is the embedding dimension of data which is two in case of images and three in case of image sequences. The other one is the intrinsic dimension of local image structures. It expresses the number of degrees of freedom necessary to describe local structure. There exists structure of intrinsic dimensions zero (i0D) which is a constant signal, one (i1D) which are lines and edges and two (i2D) which are all the other possible patterns. As the meaning of "local" is depending on the considered scale, the intrinsic dimension at a chosen image position is scale-dependent too.

In this section we will show that by embedding image analysis into geometric algebra the mentioned representation gap hopefully will be closed.

2.1 Local Spectral Representations

Image analysis in robot vision means to derive locally structural hints by filtering. From the filter outputs certain useful features can be computed. A well known but naive way of filtering is template matching. Template matching filters are detectors for image structure. Because there is no way of designing templates for each possible structure, this method will get stuck soon if i2D patterns are of interest [Rohr, 1992]. Locally the only i1D patterns are lines, edges and textures constructed from these.

What filter would enable local image analysis if it is no template of the pattern in search? The alternative is to use a set of generic features as basis functions of patterns than the patterns themselves. We call this approach the split of identity.

There are many different approaches for computing the split of identity, including multi-scale analysis, local principal component analysis and variants of local Taylor series development. All of these take into consideration only the magnitude of the filters in the Fourier domain.

Our preferred approach in the split of identity is to instead use quadrature filters h_q [Granlund and Knutsson, 1995],

$$h_q = h_e + jh_o, \tag{1}$$

that is complex valued filters, where $\{h_e, h_o\}$ is a pair of real filters with even and odd symmetry, respectively. An example is the widely used Gabor filter [Granlund and Knutsson, 1995]. This kind of filtering applies evenness and oddness as a feature basis for image analysis.

Because these filters are in quadrature phase relation, that is, their phase is shifted by $-\frac{\pi}{2}$, $\{h_e, h_o\}$ is called a quadrature pair. Then convolution of the image f with such a filter $h_\alpha,\ \alpha \in \{e, o\}$,

$$g_\alpha(\boldsymbol{x}) = (h_\alpha * f)(\boldsymbol{x}) \tag{2}$$

results in outputs g_e and g_o, respectively, which represent locally the even and odd symmetry of f. If h_q is a bandpass filter, then equation (2) enables the above mentioned multi-resolution analysis of local symmetry.

We restrict ourselves for the moment to 1D signals because for these the theory is well established.

Global even and odd symmetry is intrinsic to the Fourier representation of a real 1D signal. This property is called Hermite symmetry. But filtering a real function f with a quadrature filter h_q results in a complex valued function

$$g(x) = g_e(x) + jg_o(x) \tag{3}$$

no longer having Hermite symmetry in the Fourier domain. Instead, the Fourier transform $G(x)$ has the important property of approximating the analytic signal $F_A(u)$,

$$F_A(u) = F(u) + jF_H(u) = (1 + \operatorname{sign}(u))F(u) \tag{4}$$

very well within the passband of the quadrature filter $H_q(u)$. The analytic signal in the Fourier domain is composed of the Fourier transform of the real signal $F(u)$ and the Hilbert transformed signal $F_H(u)$,

$$F_H(u) = H_I(u)F(u) = -j \operatorname{sign}(u)F(u). \tag{5}$$

In the spatial domain the analytic signal is

$$f_A(x) = f(x) + jf_H(x), \tag{6}$$

which is not only a complex function as in equation (4) but also has the property that f and f_H are phase shifted by $-\frac{\pi}{2}$. The functions f and f_H form a Hilbert pair. This is caused by the fact that the Hilbert transform H_I changes even signals to odd ones and vice versa. There are no amplitude changes because $|H_I| = 1$ for all frequencies. H_I, being pure imaginary, causes only phase effects. The operator of the Hilbert transform in the spatial domain reads

$$h_I(x) = \frac{1}{\pi x}. \tag{7}$$

The importance of the analytic signal results from computing the local energy $e(x)$ and the local phase $\phi(x)$,

$$e(x) = f^2(x) + f_H^2(x), \tag{8a}$$

$$\phi(x) = \arg f_A(x). \tag{8b}$$

Their use in local signal analysis decomposes the split of identity into quantitative and qualitative subtasks. If there exists a threshold ϵ of significance, then $e(x) > \epsilon$ indicates a certain local variance of data. In that case it is interesting to ask for the quality of the local structure. This is given by the local phase in the following way. An edge (odd symmetry) is indicated by $\phi(x) = \pm\frac{\pi}{2}$ and a line (even symmetry) is indicated by $\phi(x) = 0$ or $\phi(x) = \pi$. The same ideas can be used to interpret the output of a quadrature filter $g(x)$, given in equation (3).

Regrettably, this method of symmetry related split of identity which we outlined so far is not easy extendable to 2D functions, although it is used in image analysis for edge and line detection. In 2D signals it is limited to intrinsically one-dimensional functions. Hence, the aim is twofold: extension in a consistent manner to a 2D embedding and extension to intrinsically two-dimensional signals. We will present our first trials in the next subsection. These results are based on the PhD thesis of Bülow [Bülow, 1999].

2.2 Quaternionic Embedding of Line Symmetry

In this section we will first show that the algebraic nature of the Fourier transform in the complex domain causes a limited representation of symmetries for 2D functions. We propose embedding the Fourier transform into an algebraic domain with more degrees of freedom. Then we will use this idea to generalize the concept of the analytic signal for embedding dimension two to cover also the case of i2D signals.

2.2.1 Quaternionic Fourier Transform. We use the well-known technique of decomposing a 1D function $f(x)$ for any location x into an even part, $f_e(x)$, and an odd part, $f_o(x)$, according to $f(x) = f_e(x) + f_o(x)r$. Because the Fourier transform is a linear integral transform it maps this decomposition in an integral manner into the frequency domain. Let $F^c(u)$ be the complex valued Fourier transform, $F^c(u) \in \mathbb{C}$, of a one-dimensional real function $f(x)$. Then

$$F^c(u) = F_R(u) + jF_I(u) = F_e(u) + F_o(u) \tag{9}$$

as a result of all possible local decompositions $f(x) = f_e(x) + f_o(x)$. Thus, the complex Fourier transform makes explicit the only symmetry concept of a 1D function. In case of 2D functions $f(\boldsymbol{x})$, $\boldsymbol{x} = (x, y)$, the complex Fourier transform

$$F^c(\boldsymbol{u}) = F_R(\boldsymbol{u}) + jF_I(\boldsymbol{u}) \ , \ \boldsymbol{u} = (u, v) \tag{10}$$

can also represent only two symmetry concepts. This contradicts the fact that in principle in two dimensions the number of different symmetries is infinite.

The 2D harmonic function

$$Q^c(\boldsymbol{u}, \boldsymbol{x}) = \exp(-j2\pi \boldsymbol{u} \cdot \boldsymbol{x}) \tag{11}$$

is the basis function of the 2D Fourier transform on a unit area

$$F^c(\boldsymbol{u}) = \iint f(\boldsymbol{x}) \, Q^c(\boldsymbol{u}, \boldsymbol{x}) \, d\boldsymbol{x}. \tag{12}$$

The 1D structure of $Q^c(\boldsymbol{u}, \boldsymbol{x})$ is the reason that in the complex Fourier domain there is no access to 2D symmetries. Looking at the decomposition

$$Q^c(\boldsymbol{u}, \boldsymbol{x}) = Q^c(u, x) \, Q^c(v, y) = \exp(-j2\pi ux) \exp(-j2\pi vy), \tag{13}$$

it is obvious that the basis functions represent a symmetry decomposition according to

$$F^c(\boldsymbol{u}) = F_{ee}(\boldsymbol{u}) + F_{oo}(\boldsymbol{u}) + F_{oe}(\boldsymbol{u}) + F_{eo}(\boldsymbol{u}) \tag{14}$$

and, thus, support in the spatial domain a decomposition

$$f(\boldsymbol{x}) = f_{ee}(\boldsymbol{x}) + f_{oo}(\boldsymbol{x}) + f_{oe}(\boldsymbol{x}) + f_{eo}(\boldsymbol{x}). \tag{15}$$

That is, the algebraic nature of equation (13) corresponds to considering products of symmetries with respect to the coordinate axes. We call

this line symmetry. But the limited degrees of freedom in the algebraic structure of complex numbers results in a partial cover of symmetry with

$$F_R(\boldsymbol{u}) = F_{ee}(\boldsymbol{u}) + F_{oo}(\boldsymbol{u}), \quad F_I(\boldsymbol{u}) = F_{eo}(\boldsymbol{u}) + F_{oe}(\boldsymbol{u}). \tag{16}$$

Hence, although all symmetries according to equation (14) are contained in the global spectral representation of the complex Fourier transform, these are hidden because of equation (16) and cannot be made explicit. Obviously this is a consequence of the complex algebra which causes the equivalence of the basis functions according to (11) and (13), respectively. With respect to local spectral representation this results in the incompleteness as discussed in subsection 2.1.

In [Zetzsche and Barth, 1990] Zetzsche and Barth argue from a diffential geometry viewpoint that the responses of LSI filters should be non-linearly combined to represent i2D structures. They developed in the sequel a non-linear filter approach based on second order Volterra series [Krieger and Zetzsche, 1996].

Instead, our approach is an algebraic extension of the degrees of freedom of a multi-dimensional Fourier transform by embedding the spectral domain into a domain of Clifford numbers. In the case of embedding a signal of dimension N in the spatial domain, the dimension of the algebra has to be 2^N [Bülow and Sommer, 2001]. We call this kind of Fourier transform a Clifford Fourier transform (CFT). The Clifford Fourier transform has been modeled already by Brackx et al. in 1982 [Brackx et al., 1982].

In the case of signal dimension $N = 2$, the following isomorphisms exist: $\mathbb{R}_{0,2} \simeq \mathbb{R}^+_{3,0} \simeq \mathbb{H}$. Hence, for that case the quaternionic Fourier transform (QFT), $F^q(\boldsymbol{u}) \in \mathbb{H}$,

$$F^q(\boldsymbol{u}) = \iint Q_i^q(u,x)\; f(\boldsymbol{x})\; Q_j^q(v,y)\; d\boldsymbol{x} \tag{17}$$

with the quaternion valued basis functions

$$Q_i^q(u,x) = \exp(-i2\pi ux)\;, \quad Q_j^q(v,y) = \exp(-j2\pi vy) \tag{18}$$

represents the symmetries of equation (14) totally uncovered in the quaternionic domain. The algebraic decomposition of F^q is given by

$$F^q(\boldsymbol{u}) = F_R^q(\boldsymbol{u}) + iF_I^q(\boldsymbol{u}) + jF_J^q(\boldsymbol{u}) + kF_K^q(\boldsymbol{u}). \tag{19}$$

For real $f(\boldsymbol{x})$ this corresponds to the symmetry decomposition (same order as in (19))

$$F^q(\boldsymbol{u}) = F_{ee}(\boldsymbol{u}) + F_{oe}(\boldsymbol{u}) + F_{eo}(\boldsymbol{u}) + F_{oo}(\boldsymbol{u}). \tag{20}$$

Instead of equation (13), the quaternionic basis functions

$$Q^q(\boldsymbol{u}, \boldsymbol{x}) = \exp(-i2\pi ux) \exp(-j2\pi vy) \tag{21}$$

cannot be fused as in equation (11). Those basis functions which are not positioned along the coordinate axes are indeed intrinsically 2D structures and, thus, are able to represent explicitly i2D signals.

The importance of Fourier phase as a carrier of geometric information is known, as well as the lack of reasonable phase concepts for 2D signals. The presented QFT overcomes this problem for the first time because of the representation

$$F^q(\boldsymbol{u}) = |F^q(\boldsymbol{u})| \exp(i\phi(\boldsymbol{u})) \exp(k\psi(\boldsymbol{u})) \exp(j\theta(\boldsymbol{u})) \tag{22}$$

of the QFT. Here the triple

$$(\phi, \theta, \psi) \in \left[-\pi, \pi \right[\times \left[-\frac{\pi}{2}, \frac{\pi}{2} \right[\times \left[-\frac{\pi}{4}, \frac{\pi}{4} \right]$$

of the quaternionic phase represents the 1D phases in axes directions (ϕ, θ) and the 2D phase (ψ), respectively.

2.2.2 Quaternionic Analytic Signal. There are different approaches, with limited success, to generalizing the analytic signal in the complex domain for representation of i2D structure. This in effect means generalization of the Hilbert transform to the multi-dimensional case [Hahn, 1996]. The quaternionic analytic signal (QAS) presented in [Bülow and Sommer, 2001] takes advantage of the additional degrees of freedom in the quaternionic domain. In the spectral domain it is

$$F_A^q(\boldsymbol{u}) = (1 + \text{sign}(u))(1 + \text{sign}(v))F^q(\boldsymbol{u}) \tag{23}$$

and in the spatial domain

$$f_A^q(\boldsymbol{x}) = f(\boldsymbol{x}) + \boldsymbol{n}^{\mathrm{T}} \boldsymbol{f}_H^q(\boldsymbol{x}), \tag{24}$$

where $\boldsymbol{n} = (i, j, k)^{\mathrm{T}}$ is the vector of quaternionic imaginary units and $\boldsymbol{f}_H^q$ is the vector of the Hilbert transformed signal,

$$\boldsymbol{f}_H^q(\boldsymbol{x}) = f(\boldsymbol{x}) * \left(\frac{\delta(y)}{\pi x}, \frac{\delta(x)}{\pi y}, \frac{1}{\pi^2 xy} \right)^{\mathrm{T}}. \tag{25}$$

Regrettably, there is a mismatch of the chosen symmetry concept in the case of an isotropic pattern. Therefore, in the next subsection we demonstrate another way of modeling a generalization of the analytic signal.

2.3 Monogenic Embedding of Point Symmetry

The drawback of the QAS is the rigid coupling of the considered symmetry concept to the coordinate axes of the image signal. In case of deviations of the patterns from that model, e.g. in case of a rotated pattern, the quaternionic signal results in wrong local spectral representations. The alternative approach of generalizing the analytic signal which is outlined in this subsection has been developed in the PhD thesis of Felsberg [Felsberg, 2002].

The aim of this approach is to generalize the analytic signal in a rotation invariant manner. While in [Bülow et al., 2000] the identification of the isotropic multi-dimensional generalization of the Hilbert transform with the Riesz transform was presented for the first time, in [Felsberg and Sommer, 2000] the relation of the corresponding multi-dimensional generalization of the analytic signal to the monogenic functions of Clifford analysis [Stein and Weiss, 1971] has been published.

As in 2D (complex domain), in 4D (quaternion domain) no rotation invariant generalization of the Hilbert transform could be formulated. This is different for embedding a 2D function into a 3D space, or more generally an nD function into an (n+1)D space. The Riesz transform then generalizes the Hilbert transform for $n > 1$. We will give a short sketch of its derivation in the 2D case in the framework of Clifford analysis, which is an extension of the complex analysis of 1D functions to higher dimensions. Furthermore, we will discuss the extension of the analytic signal to the monogenic signal. The monogenic signal will find a natural embedding into a new scale space which is derived from the Poisson kernel. All these results are complete representations of i1D signals in 2D. We finally will present a first approach of an extension to i2D signals.

2.3.1 Solutions of the 3D Laplace Equation. Complex analysis is mainly concerned with analytic or holomorphic functions. Such functions can be obtained by computing the holomorphic extension of a real 1D function. The resulting unique complex function over the complex plane fulfills the Cauchy-Riemann equations. There exists a mapping of a holomorphic function to a gradient field which fulfills the Dirac equation. Such a gradient field of a holomorphic function has zero divergence and zero curl everywhere and is, thus, the gradient field of a harmonic potential. The harmonic potential in turn fulfills the Laplace equation. What is known in signal theory as the Hilbert transform and analytic signal, respectively, corresponds to the 1D part of the mentioned holomorphic extension in the complex plane. For a real 2D function the

corresponding framework is given in Clifford harmonic analysis [Stein and Weiss, 1971, Brackx et al., 1982].

Let $\boldsymbol{x} \in \mathbb{R}^3$, $\boldsymbol{x} = x\boldsymbol{e}_1 + y\boldsymbol{e}_2 + z\boldsymbol{e}_3$, be a vector of the blades $\langle \mathbb{R}_3 \rangle_1$ of the Euclidean geometric algebra $\mathbb{R}_3$. Then a $\mathbb{R}_3$-valued analysis can be formulated, see also [Felsberg, 2002], for 3D vector fields $\boldsymbol{g}(\boldsymbol{x}), \boldsymbol{g} = g_1\boldsymbol{e}_1 + g_2\boldsymbol{e}_2 + g_3\boldsymbol{e}_3$, which fulfill the 3D Dirac equation,

$$\nabla_3 \boldsymbol{g}(\boldsymbol{x}) = 0. \tag{26}$$

Here ∇_3 is the 3D nabla operator defined in $\mathbb{R}_3$. Then $\boldsymbol{g}(\boldsymbol{x})$ is called a monogenic function and is a generalization of a holomorphic one. It can be shown that there exists a scalar valued function $p(\boldsymbol{x})$ called harmonic potential which is related to $\boldsymbol{g}(\boldsymbol{x})$ by $\boldsymbol{g}(\boldsymbol{x}) = \nabla_3 p(\boldsymbol{x})$. Therefore, $\boldsymbol{g}(\boldsymbol{x})$ is a gradient field called the harmonic potential field. The harmonic potential $p(\boldsymbol{x})$ fulfills the 3D Laplace equation,

$$\Delta_3 p(\boldsymbol{x}) = \nabla_3 \nabla_3 p(\boldsymbol{x}) = 0, \tag{27}$$

with Δ_3 as the 3D Laplace operator. If $\boldsymbol{g}(\boldsymbol{x})$ is harmonic, its three components $g_i(\boldsymbol{x})$ represent a triple of harmonic conjugates. This is the important property we want to have to generalize a Hilbert pair of functions, see subsection 2.1.

The fundamental solution of the Laplace equation (27) is given by

$$p(\boldsymbol{x}) = -\frac{1}{2\pi|\boldsymbol{x}|} \tag{28a}$$

and its derivative,

$$\boldsymbol{g}(\boldsymbol{x}) = \frac{\boldsymbol{x}}{2\pi|\boldsymbol{x}|^3} \tag{28b}$$

is the harmonic field we started with.

Before having a closer look at the fundamental solution (28) of the Laplace equation and its derivative, we will relate it to the monogenic extension $\boldsymbol{f}_m(\boldsymbol{x})$ of a real 2D function $f(x, y)$ in $\mathbb{R}_3$. The starting point is a usefull embedding of $f(x, y)$ into $\mathbb{R}_3$ as a vector field, respectively as an $\boldsymbol{e}_3$-valued function,

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{f}(x\boldsymbol{e}_1 + y\boldsymbol{e}_2) = f(x, y)\boldsymbol{e}_3. \tag{29}$$

Note that $z = 0$. Then the monogenic extension $\boldsymbol{f}_m(\boldsymbol{x})$ of the real 2D function $f(x, y)$ can be computed by solving a boundary value problem in such a way that $\boldsymbol{f}_m$ is monogenic for $z > 0$ and its $\boldsymbol{e}_3$-component corresponds to $f(x, y)$ for $z = 0$.
The boundary value problem reads

$$\Delta_3 p(\boldsymbol{x}) = 0 \quad \text{for} \quad z > 0 \tag{30a}$$

$$\boldsymbol{e}_3 \frac{\partial}{\partial z} p(\boldsymbol{x}) = f(x,y)\boldsymbol{e}_3 \quad \text{for} \quad z = 0. \tag{30b}$$

Solving equation (30a) results in the fundamental solution (28) for $z > 0$. But the boundary equation (30b) introduces the contraint $g_3(x,y,0) = f(x,y)$. Thus, the monogenic extension $\boldsymbol{f}_m(\boldsymbol{x})$ is a specific solution of (30) which in fact can be computed by convolving the function $f(x,y)$ with a set of operators derived from the fundamental solution.

The components of $\boldsymbol{g}(\boldsymbol{x})$ are derivations of $p(\boldsymbol{x})\boldsymbol{e}_3$ when $z > 0$,

$$h_P(\boldsymbol{x}) = \boldsymbol{e}_3 \frac{\partial}{\partial z} p(\boldsymbol{x})\boldsymbol{e}_3 = \frac{z}{2\pi|\boldsymbol{x}|^3} \tag{31a}$$

$$h_{CPx}(\boldsymbol{x}) = \boldsymbol{e}_1 \frac{\partial}{\partial x} p(\boldsymbol{x})\boldsymbol{e}_3 = -\frac{x}{2\pi|\boldsymbol{x}|^3}\boldsymbol{e}_{31} \tag{31b}$$

$$h_{CPy}(\boldsymbol{x}) = \boldsymbol{e}_2 \frac{\partial}{\partial y} p(\boldsymbol{x})\boldsymbol{e}_3 = \frac{y}{2\pi|\boldsymbol{x}|^3}\boldsymbol{e}_{23}. \tag{31c}$$

These functions are well-known in Clifford analysis as the Poisson kernel (31a) and the conjugate Poisson kernels (31b and 31c), respectively. While $h_P = \boldsymbol{g} \cdot \boldsymbol{e}_3$, the expressions (31b) and (31c) can be summarized by $\boldsymbol{h}_{CP} = h_{CPx} + h_{CPy} = \boldsymbol{g} \wedge \boldsymbol{e}_3$,

$$\boldsymbol{h}_{CP}(\boldsymbol{x}) = \frac{(x\boldsymbol{e}_1 + y\boldsymbol{e}_2)\boldsymbol{e}_3}{2\pi|\boldsymbol{x}|^3}. \tag{32}$$

Their Fourier transforms are

$$H_P(u,v,z) = \exp(-2\pi|u\boldsymbol{e}_1 + v\boldsymbol{e}_2|z) \tag{33a}$$

$$\boldsymbol{H}_{CP}(u,v,z) = \boldsymbol{H}_R(u,v)\exp(-2\pi|u\boldsymbol{e}_1 + v\boldsymbol{e}_2|z), \tag{33b}$$

where

$$\boldsymbol{H}_R(u,v) = -\left(\frac{u\boldsymbol{e}_{31} - v\boldsymbol{e}_{23}}{|u\boldsymbol{e}_1 + v\boldsymbol{e}_2|}\right)^* = \frac{u\boldsymbol{e}_1 + v\boldsymbol{e}_2}{|u\boldsymbol{e}_1 + v\boldsymbol{e}_2|}\boldsymbol{I}_2^{-1} \tag{34}$$

is the Riesz transform, which is the generalization of the Hilbert transform with respect to dimension. Note that in equation (34) $M^* = M\boldsymbol{I}_3^{-1}$ means the dual of the multivector M with respect to the pseudoscalar $\boldsymbol{I}_3$ of $\mathbb{R}_3$. Equations (31) formulate three harmonic conjugate functions which form a Riesz triple of operators in the half-space $z > 0$. Obviously the Riesz transform does not depend on the augmented coordinate

z. But equations (33) can be interpreted in such a way that z is a damping parameter in the frequency domain for all components of the Riesz triple. In fact, z must be interpreted as a scale parameter of a linear scale-space, called Poisson scale-space.

Here we must make a remark with respect to the Fourier transform used in this approach. Although the signal embedding is given in a 3D space, the Fourier transform is 2D. Actually we are using an isotropic transform to correspond to the model of point symmetry. Hence, for a given function $\boldsymbol{f}(\boldsymbol{x})$ the complex valued isotropic Fourier transform is

$$F(u,v) = \iint \boldsymbol{f}(\boldsymbol{x}) \exp(-\boldsymbol{I}_3 2\pi \boldsymbol{u} \cdot \boldsymbol{x})\, dxdy. \tag{35}$$

Thus, the harmonics we consider are actually spherical ones. This important feature, together with the conception of point symmetry to realize the signal decomposition into even and odd components, overcomes the restrictions of the QFT of missing rotation invariance.

In the following we will focus on the result of the fundamental solution of the Laplace equation for $z = 0$. Besides, we will consider the scale as parameter of the representation, hence, $\boldsymbol{x} = (x, y)$, and s is the scale parameter.

2.3.2 The Monogenic Signal. Let $\boldsymbol{f}_M$ be the monogenic signal [Felsberg and Sommer, 2001] of a real 2D signal $f(x, y)$. In the plane $s = 0$ the monogenic signal $\boldsymbol{f}_M$ is composed by the $\boldsymbol{e}_3$-valued representation $\boldsymbol{f}(\boldsymbol{x}) = f(x, y)\boldsymbol{e}_3$ and its Riesz transform, $\boldsymbol{f}_R(\boldsymbol{x})$,

$$\boldsymbol{f}_M(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x}) + \boldsymbol{f}_R(\boldsymbol{x}). \tag{36}$$

The monogenic signal in the frequency domain, $\boldsymbol{u} = (u, v)$,

$$\boldsymbol{F}_M(\boldsymbol{u}) = \boldsymbol{F}(\boldsymbol{u}) + \boldsymbol{F}_R(\boldsymbol{u}), \tag{37}$$

is computed by

$$\boldsymbol{F}_M(\boldsymbol{u}) = (1 + \boldsymbol{H}_R(\boldsymbol{u}))\, \boldsymbol{F}(\boldsymbol{u}) = \left(1 + \frac{\boldsymbol{u}}{|\boldsymbol{u}|}\boldsymbol{I}_2^{-1}\right) \boldsymbol{F}(\boldsymbol{u}). \tag{38}$$

In the spatial domain the Riesz transform results from convolving $\boldsymbol{f}$ with the Riesz transform kernel $\boldsymbol{h}_R$,

$$\boldsymbol{h}_R(\boldsymbol{x}) = \frac{\boldsymbol{x}\boldsymbol{e}_3}{2\pi|\boldsymbol{x}|^3}, \tag{39}$$

thus,

$$\boldsymbol{f}_R(\boldsymbol{x}) = \iint \frac{\boldsymbol{x}'\boldsymbol{e}_3}{2\pi|\boldsymbol{x}'|^3} \boldsymbol{f}(\boldsymbol{x} - \boldsymbol{x}')dx'dy'. \tag{40}$$

The Riesz transform generalizes all known Hilbert transform properties to 2D. In addition, the monogenic signal can be used to compute local spectral representations. Now the local magnitude is isotropic.

The local spectral decomposition of the monogenic signal in $\mathbb{R}^3$,

$$\boldsymbol{f}_M(\boldsymbol{x}) = |\boldsymbol{f}_M(\boldsymbol{x})| \exp(\arg(\boldsymbol{f}_M(\boldsymbol{x})\boldsymbol{e}_3)) \tag{41}$$

is given by the real local amplitude

$$|\boldsymbol{f}_M(\boldsymbol{x})| = \exp(\log(|\boldsymbol{f}_M(\boldsymbol{x})|)) = \exp(\langle \log(\boldsymbol{f}_M(\boldsymbol{x})\boldsymbol{e}_3)\rangle_0) \tag{42}$$

and the local rotation vector

$$\boldsymbol{r}(\boldsymbol{x}) = \arg(\boldsymbol{f}_M(\boldsymbol{x}))^* = \langle \log(\boldsymbol{f}_M(\boldsymbol{x})\boldsymbol{e}_3)\rangle_2^*, \tag{43}$$

see [Felsberg, 2002]. The local rotation vector $\boldsymbol{r}(\boldsymbol{x})$ lies in the plane spanned by $\boldsymbol{e}_1$ and $\boldsymbol{e}_2$. Hence, it is orthogonal to the local amplitude vector $|\boldsymbol{f}_M(\boldsymbol{x})|\boldsymbol{e}_3$. In the rotation plane the rotation vector is orthogonal to the plane spanned by $\boldsymbol{f}_M$ and $\boldsymbol{e}_3$. The length of the rotation vector is coding the angle φ between $\boldsymbol{f}_M$ and $\boldsymbol{e}_3$ which is the local phase of the 2D signal,

$$\varphi(\boldsymbol{x}) = \operatorname{sign}(\boldsymbol{r} \cdot \boldsymbol{e}_1)|\boldsymbol{r}|. \tag{44}$$

The orientation of the plane spanned by $\boldsymbol{f}_M$ and $\boldsymbol{e}_3$ expresses the local orientation of the 2D signal in the image plane, $\theta(\boldsymbol{x})$.

Local phase and local orientation are orthogonal features expressing structural and geometric information in addition to the energy information represented by the local amplitude.

Hence, from the chosen 3D embedding of a 2D signal we obtain a more complex phase representation than in the 1D case. It includes both the local phase and the local orientation of a structure. But it is limited to the case of i1D-2D structures. Nevertheless, the result is a consistent signal theory for representing lines and edges in images which tells apart the difference between the embedding dimension and the intrinsic dimension of a signal. A practical consequence is that steering of the orientation in that approach is unnecessary for i1D signals.

The phase decomposition of a monogenic signal expresses symmetries of a local i1D structure embedded in $\mathbb{R}^3$. There is an interpretation of the local phase other than that given by a rotation vector of a $\mathbb{R}_3^+$-valued signal $\boldsymbol{f}_M\boldsymbol{e}_3$ as in equation (43). Instead, the decomposition into local phase and orientation can be understood as specification of a rotation in a complex plane which is oriented in $\mathbb{R}^3$. In that case, for a given orientation θ, the well-known phase interpretation of a 1D signal (equation (8b) can be used. The embedding into $\mathbb{R}_3$ supplies the orientation of the complex plane.

2.3.3 The Poisson Scale-Space. So far we have discussed the derivation of local spectral representations, including local orientation, from the monogenic signal for the case of vanishing damping by the Poisson kernel (31a) and conjugate Poisson kernels (32), respectively. The same interpretation applies of course for all other scale parameters $s > 0$.

Because the Poisson kernel can also be derived for 1D signals by solving the Laplace equation in $\mathbb{R}_2$, there exists a similar scheme for that case [Felsberg, 2002].

The fact that $\{\boldsymbol{x}; s\}$ with s being the Poisson scale parameter represents a linear scale-space [Felsberg, 2002] is surprising at first glance. There exist several axiomatics which define a set of features a scale-space should have. The first important axiomatic of scale-space theory proposed by Iijima [Iijima, 1959] excludes the existence of other scale-spaces than that one derived from solving the heat equation, that is the Gaussian scale-space. In [Felsberg and Sommer, 2003] the reason for that wrong conclusion could be identified.

As set out in subsection 2.1 the allpass characteristics of the Hilbert transform and also of the Riesz transform hinders their application in image analysis. Instead, there is a need for quadrature filters which are in fact bandpasses. By using the Poisson kernel and its conjugate counterparts (equations (33)), it is possible to design bandpasses by building the difference of Poisson filters [Felsberg, 2002]. These are abbreviated as DOP filters and $DOCP$ filters, respectively. In order to match symmetry they are either even (DOP) or odd ($DOCPs$). As h_P, h_{CPx} and h_{CPy} these three bandpasses also form a Riesz triple. The set of filters is called a spherical quadrature filter. Interestingly, $DOCP = DOCP_x + DOCP_y$ is an odd and isotropic operator which could not be designed in another way.

The Poisson scale-space is not only new, but its conception establishes a unique framework for performing phase-based image processing in scale-space. Hence, local phase and local amplitude become inherent features of a scale-space theory, in contrast to Gaussian scale-space [Felsberg and Sommer, 2003]. In [Felsberg and Sommer, 2003] there is a first study of the properties of the monogenic scale-space.

2.3.4 The Structure Multivector. The monogenic signal only copes with symmetries of i1D structure in a rotation invariant manner and enables one to estimate the orientation in addition to phase and amplitude. How can this approach based on solving the Laplace equation be extended to i2D structure? In a general sense the answer is open yet. This results from the fact that in 2D there exist infinitely many different

symmetries. Because the monogenic signal is derived from first order harmonics as a transfer function of the Riesz transform, it follows that an increase of the order of harmonics to infinity would be necessary to cope with an arbitrary i2D structure. Hence, from this point of view we get stuck in a complexity problem similar to other knowledge-based approaches of designing pattern templates as filters.

Nevertheless, a first shot is given by the approach of the structure multivector [Felsberg and Sommer, 2002]. Here the first spherical harmonics of order zero to three are used to design a set of filters. This set implicitely assumes a model of two perpendicularly crossing i1D structures, thus representing in our approach a simple template of a special i2D structure [Felsberg, 2002].

Let h_S^i be the impulse response of a spherical harmonic filter of order i. Then $h_S^0(\boldsymbol{x}) \equiv \delta(\boldsymbol{x})$ and $h_S^1(\boldsymbol{x}) \equiv h_R(\boldsymbol{x})$. If $\boldsymbol{f}(\boldsymbol{x}) \in \mathbb{R}_3$, then

$$\boldsymbol{S}(\boldsymbol{x}) = \boldsymbol{f}(\boldsymbol{x}) + (h_S^1 * \boldsymbol{f})(\boldsymbol{x}) + \boldsymbol{e}_3(h_S^2 * \boldsymbol{f})(\boldsymbol{x}) + \boldsymbol{e}_3(h_S^3 * \boldsymbol{f})(\boldsymbol{x}) \qquad (45)$$

is a mapping of the local structure to a 7-dimensional multivector,

$$\boldsymbol{S}(\boldsymbol{x}) = s_0 + s_1\boldsymbol{e}_1 + s_2\boldsymbol{e}_2 + s_3\boldsymbol{e}_3 + s_{23}\boldsymbol{e}_{23} + s_{31}\boldsymbol{e}_{31} + s_{12}\boldsymbol{e}_{12}, \qquad (46)$$

called the structure multivector.

That response actually represents a special i2D generalization of the analytic signal. Hence, a split of identity of any i2D signal, projected to the model, can be realized in scale-space. The five independent features are local (main) orientation, two local i1D amplitudes and two local i1D phases. A major amplitude and a minor amplitude and their respective phases are distinguished. The occurrence of a minor amplitude indicates the i2D nature of the local pattern. For details the reader is referred to [Felsberg, 2002, Felsberg and Sommer, 2002].

The filter can be used to recognize both i1D and i2D structures, but in contrast to other filters which respond either to i1D or to i2D structure or mix the responses in an unspecific manner, this filter is specific to each type of structure.

3. Knowledge Based Neural Computing

Learning the required competence in perception and action is an essential feature of designing robot vision systems within the perception-action cycle. Instead of explicitly formulating the solution, implicit representation of knowledge is used. This knowledge concerns e.g. equivalence classes of objects to be recognized, actions to be performed or visuo-motor mappings.

There are plenty of different neural architectures. But most of them have in common that they are general purpose or universal comput-

ing architectures. On the other hand we know that some architecture principles are more useful for a given task than others.

If the designer is aware of the specific features of his problem, he may integrate domain/task knowledge into the very architecture of neural computing. We will call this knowledge based neural computing (KBNC).

We will focus here on algebraic approaches which are tightly related to the geometry of data. Our choice is to take geometric algebra as an embedding framework of neural computing. We developed a general scheme of embedding neural computing into Clifford algebras in the sense of a knowledge based approach [Buchholz and Sommer, 2000b, Buchholz and Sommer, 2001b, Buchholz and Sommer, 2001a]. As an outcome we could propose several special neural processing schemes based on using geometric algebra [Banarer et al., 2003b, Buchholz and Sommer, 2000a, Buchholz and Sommer, 2000c]. In our approach we are capturing higher order information of the data within a single neuron by exploiting the special multivector structure of a chosen algebra. Because we get access to the chosen Clifford group, we are thus able to learn geometric transformation groups for the first time.

Neural learning of a model can be understood as an iterative non-linear regression of a function to a set of training samples. By embedding the fitting of a model to given data into geometric algebra, we use algebraic knowledge on the nature of the problem to constrain the fit by the chosen algebraic rules. Furthermore, the chosen embedding causes a transformation of the function from a non-linear type in Euclidean space to a linear one in the special Clifford domain. This is a principle of minimal efforts which is called Occam's razor in statistical learning theory. Thus, learning the linear function in geometric algebra will be simpler than learning the non-linear one in vector algebra.

3.1 The Clifford Spinor Neuron

3.1.1 The Generic Neuron. In this section we will restrict ourselves to the algebraic embedding of neurons of perceptron type arranged in feed-forward nets. Next we will summarize the basic ideas of computing with such neurons.

Let us start with a so-called generic neuron whose output, y, reads for a given input vector $\boldsymbol{x}$ and weight vector $\boldsymbol{w}$, both of dimension n, as

$$y = g(f(\boldsymbol{x}; \boldsymbol{w})). \tag{47}$$

The given neuron is defined by a propagation function $f\colon D^n \longrightarrow D$ and by an activation function $g\colon D \longrightarrow D'$. In case of a real neuron with

$D = \mathbb{R}$ and $\boldsymbol{w}, \boldsymbol{x} \in \mathbb{R}^n$, the propagation function

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} w_i x_i + \theta, \tag{48}$$

where $\theta \in \mathbb{R}$ is a threshold, obviously computes the scalar product of $\boldsymbol{w}$ and $\boldsymbol{x}$. Because of the linear association of weight and input vectors, (48) is also called a linear associator. A neuron of that type with g being the identity operation is also called an adaline. By applying a non-linear activation function g to f, the neuron will become a non-linear computing unit called a perceptron. While an adaline may be interpreted as iteratively realizing a linear regression by learning, a trained perceptron enables the linear separation of two classes of samples. This classification performance results from the fact that the trained weight vector is perpendicular to a hyperplane in input space.

For the training of the generic neuron a supervised scheme can be used. That is, there is a teacher who knows the required answer, $r^i \in \mathbb{R}$, of the neuron to a given input $\boldsymbol{x}^i \in \mathbb{R}^n$. Hence, the training set is constituted by m pairs $(\boldsymbol{x}^i, r^i)$. Then the aim of learning is to find that weight vector $\boldsymbol{w}$ which minimizes the sum of squared error (SSE)

$$E = \frac{1}{2} \sum_{i=1}^{m} (r^i - y^i)^2. \tag{49}$$

This optimization is done by gradient descent because the weight correction at each step of the iterative procedure is given by

$$\Delta w_j = -\eta \frac{\partial E}{\partial w_j}, \tag{50}$$

where $\eta > 0$ is a suitable learning rate.

Because in a net of neurons the error has to be propagated back from the output to each neuron, this is also called back-propagation.

Finally, we will give a short sketch of combining several neurons to a (feed-forward) neural net. By arranging p (real) neurons in a single layer that is fully connected to the input vector $\boldsymbol{x}$, we will get a single layer perceptron network (SLP). If g is a non-linear function, the output vector $\boldsymbol{y}$ represents in each of its p components y_i a linear discriminent function in input space. By taking at least one of such layers of neurons and hiding it under an output layer, we will get a hierarchical architecture of neurons, which is called multilayer perceptron (MLP) architecture. The output layer is composed of at least one neuron which computes the superposition of the neurons of the preceding layer according to equation

(47). Hence, the MLP may be used either as a non-linear classification unit or for approximating any non-linear function.

If, on the other hand, g is the identity, the SLP computes an ordinary matrix multiplication,

$$\boldsymbol{y} = W\boldsymbol{x}, \tag{51}$$

where W is the weight matrix containing the single weight vectors. If W is square ($\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$), then it represents a linear transformation of the input vector.

3.1.2 The Clifford Neuron. Now we will extend the model of a real valued generic neuron to that of a Clifford valued one. We will neglect for the moment the activation function. By replacing the scalar product of the linear associator by the geometric product of an algebra $\mathbb{R}_{p,q}$, $p+q=n$, we are embedding the neuron into $\mathbb{R}_{p,q}$ according to

$$\boldsymbol{f} : \mathbb{R}_{p,q} \longrightarrow \mathbb{R}_{p,q}. \tag{52}$$

Hence, for $\boldsymbol{x}, \boldsymbol{w}, \Theta \in \mathbb{R}_{p,q}$ the propagation function is

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{w}\boldsymbol{x} + \Theta \tag{53a}$$

or

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{x}\boldsymbol{w} + \Theta, \tag{53b}$$

respectively. Note that the geometric product with respect to $\mathbb{R}_{p,q}$ now has to be computed. The splitting of the propagation function into the two variants of equation (53) follows obviously from the non-commutativity we have to assume for the geometric product of $\mathbb{R}_{p,q}$.

Having a training set $T := \{(\boldsymbol{x}^1, \boldsymbol{r}^1), \ldots, (\boldsymbol{x}^m, \boldsymbol{r}^m) | \boldsymbol{x}^i, \boldsymbol{r}^i \in \mathbb{R}_{p,q}\}$, the weight corrections

$$\Delta\boldsymbol{w} = \overline{\boldsymbol{x}}^i(\boldsymbol{r}^i - \boldsymbol{w}\boldsymbol{x}^i) \tag{54a}$$

for left-sided weight multiplication and

$$\Delta\boldsymbol{w} = (\boldsymbol{r}^i - \boldsymbol{x}^i\boldsymbol{w})\overline{\boldsymbol{x}}^i \tag{54b}$$

for right-sided weight multiplication enable a Clifford neuron to learn in a similar manner as a real one. Here $\overline{\boldsymbol{x}}$ means the conjugate of $\boldsymbol{x}$ in the Clifford algebra. By taking this involution, the appearance of divisors of zero during the gradient descent is prevented [Buchholz and Sommer, 2001b, Buchholz and Sommer, 2001a].

What is the benefit derived from computing (53) instead of (48)? We give an illustrative example. Let $\boldsymbol{x} = x_1 + ix_2$ and $\boldsymbol{y} = y_1 + iy_2$ be two fixed complex numbers, $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}$ with $\mathbb{C} \simeq \mathbb{R}_{0,1} \simeq \mathbb{R}^+_{2,0}$. The task of a complex neuron shall be learning the mapping $f\colon \boldsymbol{x} \longrightarrow \boldsymbol{y}$. This corresponds to learning a weight $\boldsymbol{w} \in \mathbb{C}$ so that $\boldsymbol{wx} = \boldsymbol{y}$, that is to learn a complex multiplication. This is in fact a simple task for the complex neuron. If instead real neurons should do the same, there is a need of a SLP with two neurons to compute y_1 and y_2, respectively. According to (51) the task is now to learn a weight matrix $W \in \mathbb{R}(2)$, which satisfies $W(x_1, x_2)^T = (y_1, y_2)^T$ with $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^2$. Here W represents a linear transformation of the vector $\boldsymbol{x}$. The SLP has to find out the constraints on W which correspond to the matrix representation of a complex number. These constraints are obviously $w_{11} = w_{22}$ and $w_{12} = -w_{21}$. As shown in [Buchholz and Sommer, 2001b] the SLP converges slower than the complex neuron. It is obviously better to use a model than to perform its simulation. Another advantage is that the complex neuron has half of the parameters (weights) to learn in comparison to the SLP.

Both of these observations can be generalized to any Clifford neuron in comparison to a corresponding SLP. Because of the $\mathbb{R}$-linearity of Clifford algebras [Porteous, 1995], any geometric product can be expressed as a special matrix multiplication. Hence, by choosing a certain Clifford algebra it is not only that a decision is made to use an algebraic model, but statistical learning will become a simpler task.

3.1.3 The Clifford Spinor Neuron. There are additional advantages to using this approach. Next we want to extend our model. The above toy example gives us a hint. The SLP has to find out that W is indeed representing an orthogonal transformation. Because each special orthogonal matrix $W \in \mathbb{R}(2)$ is a rotation matrix, W rotates $\boldsymbol{x}$ to $\boldsymbol{y}$ in $\mathbb{R}^2$. Therefore, in the case of the complex neuron the equation $\boldsymbol{wx} = \boldsymbol{y}$ can be interpreted as mapping $\boldsymbol{x}$ to $\boldsymbol{y}$, $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{C}$, by the complex number $\boldsymbol{w}$. But now $\boldsymbol{w}$ is no longer a point in the complex plane but the geometric product of $\boldsymbol{w}$ represents a special linear transformation, namely a rotation-dilation, see [Hestenes, 1993]. The representation of such an orthogonal operator in geometric algebra is given by the sum of a scalar and bivector components and is called a spinor, $\boldsymbol{S}$. In general, a spinor performs a dilation-rotation in the even subalgebra $\mathbb{R}^+_{p,q}$ of any geometric algebra $\mathbb{R}_{p,q}$.

Hence, the complex neuron learns a spinor operation, which is performing an orthogonal transformation in $\mathbb{C}$. But this is no longer true for our simple model of the Clifford neuron introduced in equation (53) if

we choose vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^3$. The spinors of $\mathbb{R}_3$ are quaternions because of the isomorphism $\mathbb{R}_3^+ \simeq \mathbb{H}$. To perform a rotation in $\mathbb{R}^3$, which maps vectors to vectors, the two-sided spinor product

$$\sigma(\boldsymbol{x}) : \boldsymbol{x} \longrightarrow \boldsymbol{S}\boldsymbol{x}\widehat{\boldsymbol{S}}^{-1} \tag{55}$$

has to be used in (53) instead. In this equation, the right half of the multiplication is performed with the inverse parity conjugated spinor, $\widehat{\boldsymbol{S}}^{-1}$. The geometric product is now the quaternion product because of the isomorphism $\mathbb{R}_{3,0}^+ \simeq \mathbb{R}_{0,2}$. Instead of the vector $\boldsymbol{x} \in \mathbb{R}^3$, we have to use its representation as a (pure) quaternion, $\boldsymbol{x} \in \mathbb{R}_{0,2}$.

The greatest advantage of embedding a geometric problem into geometric algebra is to profit from such linear realization of group actions. Another advantage is that the group members, and the set of elements the group members are acting on, belong to the same algebra.

Because in our context the spinors are representing weights of the input of a neuron, we will continue using the notation $\boldsymbol{w}$ instead of $\boldsymbol{S}$. The propagation function of a general Clifford spinor neuron, embedded into $\mathbb{R}_{p,q}^+$, is

$$\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{w}\boldsymbol{x}\widehat{\boldsymbol{w}}^{-1} + \Theta. \tag{56}$$

Such a neuron computes an orthogonal transformation by using only one weight, $\boldsymbol{w}$. Only in the case of a spinor product do we get the constraints necessary to use only one neuron with one weight for computing an orthogonal transformation. Obviously, the general Clifford spinor neuron has only half as many parameters (and half as many arithmetic operations) as the general Clifford neuron because only the even components of the weight multivector are used. But special care has to be taken to use the right coding of input and output data [Buchholz and Sommer, 2001b].

Now we will complete the neuron model by considering the activation function g, see equation (47). There is a need for generalizing an activation function on the real domain to the most popular sigmoid function

$$g_\beta(u) : \mathbb{R} \longrightarrow \mathbb{R};\ u \longrightarrow (1 + \ \exp\left(-\beta u\right))^{-1} \tag{57}$$

on a Clifford valued domain. So far there is no easy way to formulate such a generalization [Georgiou and Koutsougeras, 1992]. Therefore, we use a component-wise activation function

$$\boldsymbol{g}(\boldsymbol{u}) = g([\boldsymbol{u}]_i), \qquad i \in \{0, ..., 2^{n-1}\} \tag{58}$$

which is operating separately on all $2^n, n = p + q$, components of $\mathbb{R}_{p,q}$. This was first proposed by Arena et al. [Arena et al., 1997] for the quaternionic case.

The construction of a Clifford MLP (CMLP) or of a Clifford Spinor MLP (CSMLP) by following the principles of a real MLP is straightforward. To formulate a generalized backpropagation algorithm [Buchholz and Sommer, 2001a] is not a problem.

In principle real MLPs, CMLPs and CSMLPs have the same theoretical strength because all are universal approximators. Because they also use the same activation function, any potential advantage with respect to generalization of using the embedding of neural computation into geometric algebra should be based on the geometric product of the propagation function.

3.1.4 Learning a Euclidean 2D Similarity Transformation.

The task is to learn the plane Euclidean similarity transformation composed by a 2D rotation with $\varphi = -55^o$, a translation with $\boldsymbol{t} = (+1, -0.8)$ and a dilation by the factor $\delta = 1.5$ [Buchholz and Sommer, 2001b].

Figure 1 shows both the training data (a) and test data (b), respectively. We applied a SLP with two real neurons and four weights, one complex neuron and one complex spinor neuron. While the complex neuron has two weights, the spinor neuron has only one. Figure 1c shows the convergence of the three computing units during training. As we see, the complex neuron converges faster than the real neurons. The reason for that behaviour has been discussed above. The spinor neuron needs more training steps for learning the constraints of the spinor product. But after 60 epochs its error is the lowest one of all three models. The spinor neuron learns a spinor representation, which is indicated by the successive convergence of the odd components of the propagation function to zero.

Generalization with respect to learning of a function means keeping the approximation error low if the data have not been seen during training. The behaviour of the neurons applied to the test data should be comparable to that for the training data. In the case of a KBNC approach, the learning of the model should also be robust with respect to distortions of the model in the data caused by noise. We overlayed both the training and test input data with additive median-free uniform noise up to a level of 20 percent. The mean square errors of the outputs are shown in figures 1d and 1e. Both the Clifford neuron and the Clifford spinor neuron are slightly distorted in learning the model. But their results with respect to the test data in comparison to training data indicate a good generalization. There is no significant difference in the behaviour of both Clifford neuron models with respect to generalization. This is what we expect in the case of complex algebra. The result for

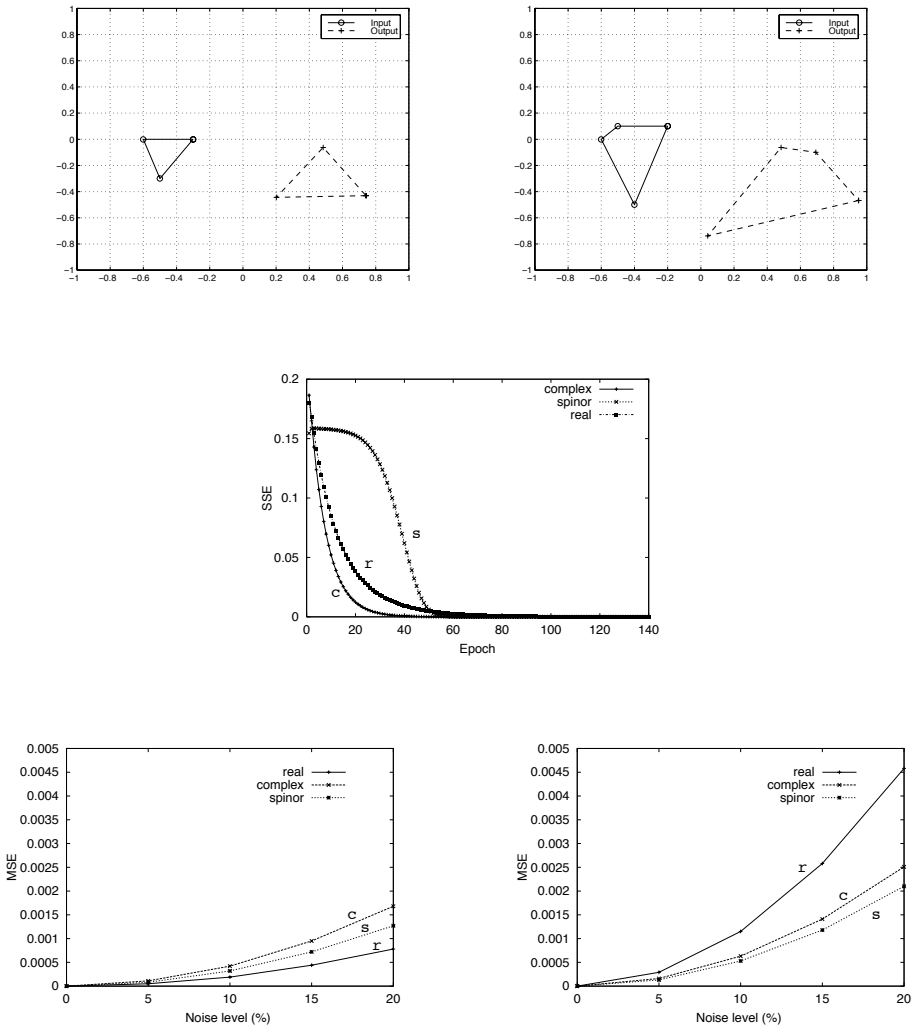

Figure 1. Learning of a plane Euclidean similarity transformation. Upper row: left (a): training data; right (b): test data; middle (c): convergence of the learning; lower row: approximation error; left (d): training errors; right (e): test errors.

the Clifford neurons contrasts with the performance of the real neurons. These are much less distorted during training but much more in the case of test data in comparison to the knowledge based neurons.

Because real neurons are learning a general linear transformation, they are able to learn noise better than the Clifford neurons. This causes both the lower errors for training data and the higher ones for test data. Because the real neurons have no constraints to separate the intrinsic

properties of the Euclidean transformation from those of the noise, they learn a transformation which only barely copes with new data. This indicates a bad generalization.

Both models of the Clifford neuron are constrained by the involved algebra to separate the properties of the geometric transformation from those of the noise. They are hindered to learn noise. Therefore, their results are worse than those of the real neurons in the case of training data but better in the case of test data.

3.2 The Hypersphere Neuron

Here we will present a special neural architecture which is based on the conformal geometric algebra. This algebra creates a non-Euclidean model of Euclidean geometry with the remarkable property of metrical equivalence [Dress and Havel, 1993]. This embedding of a Euclidean vector space results in a hypersphere decision boundary which, in the used embedding space, is a hyperplane and thus can be learned by a perceptron-like neuron.

3.2.1 The Homogeneous Model of the Euclidean Space.

It has been proposed by Hestenes in [Hestenes, 1991] that the conformal group $C(p,q)$ of $\mathbb{R}^{p,q}$ can be elegantly constructed in $\mathbb{R}_{p+1,q+1}$ in the framework of geometric algebra by applying the outer product, $\wedge$, with a unit two-blade, $\boldsymbol{E} := \boldsymbol{e} \wedge \boldsymbol{e}_0 = \boldsymbol{e}_+ \wedge \boldsymbol{e}_-$. This operation is called conformal split. It transforms the conformal group into a representation which is isomorphic to the orthogonal group $O(p+1, q+1)$ of $\mathbb{R}^{p+1,q+1}$. In [Li et al., 2001] Li et al. worked out in detail the method of representing the conformal geometry of the Euclidean space $\mathbb{R}^n$ in Minkowski space $\mathbb{R}^{n+1,1}$, respectively in the conformal geometric algebra $\mathbb{R}_{n+1,1}$.

The construction $\mathbb{R}^{n+1,1} = \mathbb{R}^n \oplus \mathbb{R}^{1,1}$, $\oplus$ being the direct sum, uses a plane with Minkowski signature whose basis $(\boldsymbol{e}_+, \boldsymbol{e}_-)$ with $\boldsymbol{e}_+^2 = 1$, $\boldsymbol{e}_-^2 = -1$ augments the Euclidean space $\mathbb{R}^n$ to realize a homogeneous stereographic projection of all points $\boldsymbol{x} \in \mathbb{R}^n$ to points $\underline{\boldsymbol{x}} \in \mathbb{R}^{n+1,1}$, see [Rosenhahn and Sommer, 2002]. By replacing the basis $(\boldsymbol{e}_+, \boldsymbol{e}_-)$ with the basis $(\boldsymbol{e}, \boldsymbol{e}_0)$ in the Minkowski plane, the homogeneous stereographic representation will become a representation as a space of null vectors. This is caused by the properties of the new basis vectors. These are related to the former ones by $\boldsymbol{e} = \boldsymbol{e}_- + \boldsymbol{e}_+$ and $\boldsymbol{e}_0 = \frac{1}{2}(\boldsymbol{e}_- - \boldsymbol{e}_+)$ with $\boldsymbol{e}^2 = \boldsymbol{e}_0^2 = 0$ and $\boldsymbol{e} \cdot \boldsymbol{e}_0 = -1$.

Any point $\boldsymbol{x} \in \mathbb{R}^{p,q}$ transforms to a point $\underline{\boldsymbol{x}} \in \mathbb{R}^{p+1,q+1}$,

$$\underline{\boldsymbol{x}} = \boldsymbol{x} + \frac{1}{2}\boldsymbol{x}^2 \boldsymbol{e} + \boldsymbol{e}_0 \tag{59}$$

with $\underline{\boldsymbol{x}}^2 = 0$.

Hence, points of the Euclidean space $\mathbb{R}^3$ are represented by null vectors in the 5-dimensional space $\mathbb{R}^{4,1}$ with Minkowski signature. Actually, they are lying on a subspace of $\mathbb{R}^{4,1}$ called horosphere, N_e^3. The horosphere, which is a cut of the null cone with a hyperplane, see [Li et al., 2001], with the remarkable property of being a non-Euclidean model of Euclidean space, has been known for a long time, see [Yaglom, 1988]. But only in geometric algebra is there a practical and relevant approach to exploit this powerful non-Euclidean representation in engineering applications. The horosphere N_e^n is metrically equivalent to a Euclidean space $\mathbb{R}^n$. It is called the homogeneous model of Euclidean space, since points in $\mathbb{R}^n$ are represented in generalized homogeneous coordinates.

3.2.2 Construction and Properties of the Hypersphere Neuron. Being metrically equivalent means that there exists a correspondence between the distance $d(\boldsymbol{x}, \boldsymbol{y})$ in $\mathbb{R}^n$ and the distance $d(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}})$ on the horosphere N_e^n of $\mathbb{R}^{n+1,1}$. With $d(\boldsymbol{x}, \boldsymbol{y}) = |\boldsymbol{x} - \boldsymbol{y}| = \sqrt{(\boldsymbol{x} - \boldsymbol{y})^2}$ this mapping reads $d(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}}) = -\frac{1}{2} d^2(\boldsymbol{x}, \boldsymbol{y})$, see [Dress and Havel, 1993]. Given two points $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, their distance $d(\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}})$ is computed simply by the scalar product $\underline{\boldsymbol{x}} \cdot \underline{\boldsymbol{y}}$ in $\mathbb{R}^{n+1,1}$. Hence [Li et al., 2001],

$$\underline{\boldsymbol{x}} \cdot \underline{\boldsymbol{y}} = -\frac{1}{2}(\boldsymbol{x} - \boldsymbol{y})^2. \tag{60}$$

Any point $\underline{\boldsymbol{c}} \in N_e^n$ can be interpreted as a degenerate hypersphere $\underline{\boldsymbol{s}}$ with center at $\underline{\boldsymbol{c}}$ and radius $r \in \mathbb{R}$ equal to zero,

$$\underline{\boldsymbol{s}} = \boldsymbol{c} + \frac{1}{2}(\boldsymbol{c}^2 - r^2)\boldsymbol{e} + \boldsymbol{e}_0. \tag{61}$$

A point $\underline{\boldsymbol{x}}$ lies on a hypersphere if $|\boldsymbol{x} - \boldsymbol{c}| = |r|$. The general relation of a point $\underline{\boldsymbol{x}}$ and a hypersphere $\underline{\boldsymbol{s}}$ may be described by evaluating their distance

$$d(\underline{\boldsymbol{x}}, \underline{\boldsymbol{s}}) = \underline{\boldsymbol{x}} \cdot \underline{\boldsymbol{s}} = \underline{\boldsymbol{x}} \cdot \underline{\boldsymbol{c}} - \frac{1}{2} r^2 \underline{\boldsymbol{x}} \cdot \boldsymbol{e} = -\frac{1}{2}(\boldsymbol{x} - \boldsymbol{c})^2 + \frac{1}{2} r^2. \tag{62}$$

Then the distance of a point, represented by the null vector $\underline{\boldsymbol{x}}$, to a hypersphere $\underline{\boldsymbol{s}} \in \mathbb{R}^{n+1,1}$ will be

$$d(\underline{\boldsymbol{x}}, \underline{\boldsymbol{s}}) : \begin{cases} > 0 \text{ if } \underline{\boldsymbol{x}} \text{ is outside } \underline{\boldsymbol{s}} \\ = 0 \text{ if } \underline{\boldsymbol{x}} \text{ is on } \underline{\boldsymbol{s}} \\ < 0 \text{ if } \underline{\boldsymbol{x}} \text{ is inside } \underline{\boldsymbol{s}} \end{cases} . \tag{63}$$

That distance measure is used for designing the propagation function of a hypersphere neuron [Banarer et al., 2003b]. If the parameters of

the hypersphere are interpreted as the weights of a perceptron, then by embedding any data points into $\mathbb{R}^{n+1,1}$, the decision boundary of the perceptron will be a hypersphere. Because a vector $\underline{\boldsymbol{x}} \in \mathbb{R}^{n+1,1}$ without an $\boldsymbol{e}_0-$component represents a hypersphere with infinite radius, that is a hyperplane, the hypersphere neuron subsumes the classical perceptron as a special case.

The implementation of the hypersphere neuron uses the equivalence of the scalar products in $\mathbb{R}^{n+1,1}$ and $\mathbb{R}^{n+2}$. If $\underline{\boldsymbol{x}}, \underline{\boldsymbol{y}} \in \mathbb{R}^{n+1,1}$ and $\overline{\boldsymbol{x}}, \overline{\boldsymbol{y}} \in \mathbb{R}^{n+2}$, then $\underline{\boldsymbol{x}} \cdot \underline{\boldsymbol{y}} = \overline{\boldsymbol{x}} \cdot \overline{\boldsymbol{y}}$. Therefore, we use the following coding for a data vector $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{x} = (x_1, ..., x_n)$, embedded in $\mathbb{R}^{n+2}$: $\overline{\boldsymbol{x}} = (x_1, ..., x_n, -1, -\frac{1}{2}\boldsymbol{x}^2)$. The coding of the hypersphere $\underline{\boldsymbol{s}} \in \mathbb{R}^{n+1,1}$, represented in $\mathbb{R}^{n+2}$, is given by $\overline{\boldsymbol{s}} = (c_1, ..., c_n, \frac{1}{2}(\boldsymbol{c}^2 - r^2), 1)$. As a result of that embedding, a hypersphere in $\mathbb{R}^n$ is represented by a hyperplane in $\mathbb{R}^{n+2}$. This maps the hypersphere neuron to a perceptron with a second bias component.

In the above coding, the components of the hypersphere are considered as independent. This makes the hypersphere unnormalized and enables one to control the assignment of the data either to the inner or to the outer class of a 2-class decision problem, see [Banarer et al., 2003b]. This is of special interest because the effective radius of the hypersphere will be influenced by the parameter β of the sigmoidal activation function, see equation (57), which is used to complete the neuron. If we remember the interpretation of points in $\mathbb{R}^{n+1,1}$ as a degenerate hypersphere, we are able to assign to the data points a confidence measure by extending the points to hyperspheres with imaginary radius. The confidence attributed to a data point is

$$\underline{\boldsymbol{x}}_{CONF} = \boldsymbol{x} + \frac{1}{2}(\boldsymbol{x}^2 + r_{CONF}^2)\boldsymbol{e} + \boldsymbol{e}_0. \tag{64}$$

From the scalar product between the hypersphere $\underline{\boldsymbol{s}}$ and $\underline{\boldsymbol{x}}_{CONF}$,

$$\underline{\boldsymbol{s}} \cdot \underline{\boldsymbol{x}}_{CONF} = \frac{1}{2}(r^2 - ((\boldsymbol{x} - \boldsymbol{c})^2 + r_{CONF}^2)), \tag{65}$$

follows a shift of the effective distance of the point to the hypersphere during training of the neuron. This leads to an adaption of the classification results to the confidence of the data.

3.2.3 The Performance of the Hypersphere Neuron. The hypersphere neuron is an elemental computing unit of a hypersphere single-layer perceptron (SLHP) or multi-layer perceptron (MLHP), respectively. Its superior performance in comparison to the classical SLP or MLP, respectively, can be demonstrated with respect to several benchmark data sets [Banarer et al., 2003a].

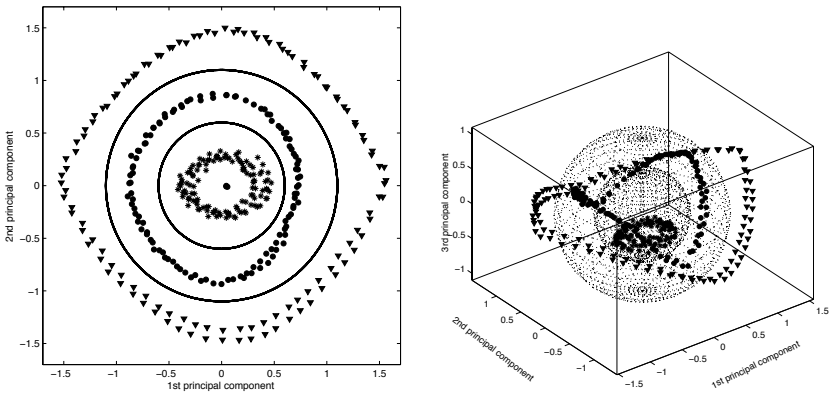

Figure 2. Classification of three toy objects. Visualization of the decision surfaces and data sets of the three first principal components.

In figure 2 we show an example from real world data. The problem is to recognize three different toy objects in a rotation invariant manner. For each of the 360 data sets (images with objects rotated each by 1°) a principal component analysis (PCA) was performed. Finally, the first three eigen-images of each class, each transformed to a data vector of length 1225, have been used for classification. Two neurons of a MLHP have been sufficient to separate the three classes. Obviously, the neuron model used is adequate for the problem.

Acknowledgments

The reported work is based on several research projects which were granted by DFG, EC and the German National Merit Foundation. Several PhD theses and diploma theses came out of that research. I have to thank all these young scientists for their enthusiasm in solving the problems I outlined in very limited space. Special thanks for the preparation of the figures to Sven Buchholz (1) and Vladimir Banarer (2).

References

[Arena et al., 1997] Arena, P., Fortuna, L., Museato, G., and Xibilia, M. (1997). Multilayer perceptrons to approximate quaternion valued functions. *Neural Networks*, 10(2):335–342.

[Banarer et al., 2003a] Banarer, V., Perwass, C., and Sommer, G. (2003a). Design of a multilayered feed-forward neural network using hypersphere neurons. In *Proc. Int. Conf. Computer Analysis of Images and Patterns, CAIP 2003, Groningen, August 2003*. accepted.

[Banarer et al., 2003b] Banarer, V., Perwass, C., and Sommer, G. (2003b). The hypersphere neuron. In *11th European Symposium on Artificial Neural Networks,*

ESANN 2003, Bruges, pages 469–474. d-side publications, Evere, Belgium.

[Brackx et al., 1982] Brackx, F., Delanghe, R., and Sommen, F. (1982). *Clifford Analysis*. Pitman Advanced Publ. Program, Boston.

[Buchholz and Sommer, 2000a] Buchholz, S. and Sommer, G. (2000a). A hyperbolic multilayer perceptron. In Amari, S., Giles, C., Gori, M., and Piuri, V., editors, *International Joint Conference on Neural Networks, IJCNN 2000, Como, Italy*, volume 2, pages 129–133. IEEE Computer Society Press.

[Buchholz and Sommer, 2000b] Buchholz, S. and Sommer, G. (2000b). Learning geometric transformations with Clifford neurons. In Sommer, G. and Zeevi, Y., editors, *2nd International Workshop on Algebraic Frames for the Perception-Action Cycle, AFPAC 2000, Kiel*, volume 1888 of *LNCS*, pages 144–153. Springer-Verlag.

[Buchholz and Sommer, 2000c] Buchholz, S. and Sommer, G. (2000c). Quaternionic spinor MLP. In *8th European Symposium on Artificial Neural Networks, ESANN 2000, Bruges*, pages 377–382.

[Buchholz and Sommer, 2001a] Buchholz, S. and Sommer, G. (2001a). Clifford algebra multilayer perceptrons. In Sommer, G., editor, *Geometric Computing with Clifford Algebra*, pages 315–334. Springer-Verlag, Heidelberg.

[Buchholz and Sommer, 2001b] Buchholz, S. and Sommer, G. (2001b). Introduction to neural computation in Clifford algebra. In Sommer, G., editor, *Geometric Computing with Clifford Algebra*, pages 291–314. Springer-Verlag, Heidelberg.

[Bülow, 1999] Bülow, T. (1999). Hypercomplex spectral signal representations for the processing and analysis of images. Technical Report Number 9903, Christian-Albrechts-Universität zu Kiel, Institut für Informatik und Praktische Mathematik.

[Bülow et al., 2000] Bülow, T., Pallek, D., and Sommer, G. (2000). Riesz transforms for the isotropic estimation of the local phase of Moiré interferograms. In Sommer, G., Krüger, N., and Perwass, C., editors, *Mustererkennung 2000*, pages 333–340. Springer-Verlag, Heidelberg.

[Bülow and Sommer, 2001] Bülow, T. and Sommer, G. (2001). Hypercomplex signals - a novel extension of the analytic signal to the multidimensional case. *IEEE Transactions on Signal Processing*, 49(11):2844–2852.

[Dress and Havel, 1993] Dress, A. and Havel, T. (1993). Distance geometry and geometric algebra. *Foundations of Physics*, 23(10):1357–1374.

[Faugeras, 1995] Faugeras, O. (1995). Stratification of three-dimensional vision: projective, affine and metric representations. *Journal of the Optical Society of America*, 12(3):465–484.

[Felsberg, 2002] Felsberg, M. (2002). Low-level image processing with the structure multivector. Technical Report Number 0203, Christian-Albrechts-Universität zu Kiel, Institut für Informatik und Praktische Mathematik.

[Felsberg and Sommer, 2000] Felsberg, M. and Sommer, G. (2000). The multidimensional isotropic generalization of quadrature filters in geometric algebra. In Sommer, G. and Zeevi, Y., editors, *2nd International Workshop on Algebraic Frames for the Perception-Action Cycle, AFPAC 2000, Kiel*, volume 1888 of *LNCS*, pages 175–185. Springer-Verlag.

[Felsberg and Sommer, 2001] Felsberg, M. and Sommer, G. (2001). The monogenic signal. *IEEE Transactions on Signal Processing*, 49(12):3136–3144.

[Felsberg and Sommer, 2002] Felsberg, M. and Sommer, G. (2002). The structure multivector. In Dorst, L., Doran, C., and Lasenby, J., editors, *Applications of Geometric Algebra in Computer Science and Engineering*, pages 437–446. Proc. AGACSE 2001, Cambridge, UK, Birkhäuser Boston.

[Felsberg and Sommer, 2003] Felsberg, M. and Sommer, G. (2003). The monogenic scale-space: A unifying approach to phase-based image processing in scale-space. *Journal of Mathematical Imaging and vision*. accepted.

[Georgiou and Koutsougeras, 1992] Georgiou, G. and Koutsougeras, C. (1992). Complex domain back propagation. *IEEE Trans. Circ. and Syst. II*, 39:330–334.

[Granlund and Knutsson, 1995] Granlund, G. and Knutsson, H. (1995). *Signal Processing for Computer Vision*. Kluwer Academic Publ., Dordrecht.

[Hahn, 1996] Hahn, S. (1996). *Hilbert Transforms in Signal Processing*. Artech House, Boston, London.

[Hestenes, 1991] Hestenes, D. (1991). The design of linear algebra and geometry. *Acta Appl. Math.*, 23:65–93.

[Hestenes, 1993] Hestenes, D. (1993). *New Foundations for Classical Mechanics*. Kluwer Academic Publ., Dordrecht.

[Hestenes et al., 2001] Hestenes, D., Li, H., and Rockwood, A. (2001). New algebraic tools for classical geometry. In Sommer, G., editor, *Geometric Computing with Clifford Algebras*, pages 3–23. Springer-Verlag, Heidelberg.

[Iijima, 1959] Iijima, T. (1959). Basic theory of pattern observation (in japanese). In *Papers of Technical Group on Automation and Automatic Control*. IECE, Japan.

[Krieger and Zetzsche, 1996] Krieger, G. and Zetzsche, C. (1996). Nonlinear image operators for the evaluation of local intrinsic dimensionality. *IEEE Trans. Image Process.*, 5:1026–1042.

[Li et al., 2001] Li, H., Hestenes, D., and Rockwood, A. (2001). Generalized homogeneous coordinates for computational geometry. In Sommer, G., editor, *Geometric Computing with Clifford Algebras*, pages 27–59. Springer-Verlag, Heidelberg.

[Pauli, 2001] Pauli, J. (2001). *Learning-Based Robot Vision*, volume 2048 of *Lecture Notes in Computer Science*. Springer-Verlag, Heidelberg.

[Porteous, 1995] Porteous, I. (1995). *Clifford Algebras and the Classical Groups*. Cambridge University Press, Cambridge.

[Rohr, 1992] Rohr, K. (1992). Recognizing corners by fitting parametric models. *International Journal Computer Vision*, 9:213–230.

[Rosenhahn and Sommer, 2002] Rosenhahn, B. and Sommer, G. (2002). Pose estimation in conformal geometric algebra, part I: The stratification of mathematical spaces, part II: Real-time pose estimation using extended feature concepts. Technical Report Number 0206, Christian-Albrechts-Universität zu Kiel, Institut für Informatik und Praktische Mathematik.

[Sommer, 1992] Sommer, G. (1992). Signal theory and visual systems. In *Measurement 92*, pages 31–46. Slovac Acad. Science, Bratislava.

[Sommer, 1997] Sommer, G. (1997). Algebraic aspects of designing behavior based systems. In Sommer, G. and Koenderink, J., editors, *Algebraic Frames for the Perception and Action Cycle*, volume 1315 of *Lecture Notes in Computer Science*, pages 1–28. Proc. Int. Workshop AFPAC'97, Kiel, Springer–Verlag, Heidelberg.

[Sommer, 2003] Sommer, G. (2003). The geometric algebra approach to robot vision. Technical Report Number 0304, Christian-Albrechts-Universität zu Kiel, Institut für Informatik und Praktische Mathematik.

[Stein and Weiss, 1971] Stein, E. and Weiss, G. (1971). *Introduction to Fourier Analysis on Euclidean Spaces*. Princeton University Press, Princeton, N.J.

[Yaglom, 1988] Yaglom, M. (1988). *Felix Klein and Sophus Lie*. Birkhäuser, Boston.

[Zetzsche and Barth, 1990] Zetzsche, C. and Barth, E. (1990). Fundamental limits of linear filters in the visual processing of two-dimensional signals. *Vision Research*, 30:1111–1117.