A new Selforganizing Neural Network using Geometric Algebra

Eduardo Bayro-Corrochano, Sven Buchholz, Gerald Sommer
Computer Science Institute, Cognitive Systems Group,
Christian-Albrechts University, Kiel, Germany
E-mail: `edb,sbh,gs@informatik.uni-kiel.d400.de`

Abstract

This paper presents a new selforganizing type RBF neural network and introduces the Geometric Algebra framework in the neurocomputing field. Real valued neural nets for function approximation require feature enhancement, dilation and rotation operations and are limited by the Euclidean metric. The authors believe that more general and flexible neural networks should be designed in order to capture important geometric characteristics of the manifolds. This is an important goal overlooked ever since. Geometric algebra is a system which allows the design of neural networks in a coordinate-free framework to process patterns between layers using different dimensions and desired metric. The potential of such nets working in a Clifford algebra $\mathcal{C}(V_{p,q})$ is shown by a simple application of frame coordination in robotics.

1 Introduction

Geometric algebra is a coordinate-free approach to geometry based on the algebras of Grassmann and Clifford [2] and has already been successfully applied to many areas of mathematical physics and engineering [2, 3, 4]. This paper shows that general and more flexible neural networks can be designed in the Geometric algebra framework to process patterns between layers using different dimensions and desired metric.

An outline of the algebra will be given in the next section and the reader is referred to [2] for more details. The third section involves a discussion on the metric in neural computing. The new architecture and its learning procedure is presented in the fourth section. Finally experimental results of the robotics field and the conclusion sections follow.

2 An Outline of Clifford Algebra

Clifford algebras are well-known to pure mathematicians [1]. In this work it will be used an interpretation called geometric algebra [2] which is a coordinate-free approach to geometry. In geometric algebra the elements are coordinate-independent objects called multivectors which can be multiplied together using a *geometric product*. It is thus very different to standard vector calculus.

2.1 The Geometric Product and Multivectors

The geometric or Clifford product of two vectors $\mathbf{a}$ and $\mathbf{b}$ is written $\mathbf{ab}$ and defined as

$$\mathbf{ab} = \mathbf{a}\cdot\mathbf{b} + \mathbf{a}\wedge\mathbf{b}. \tag{1}$$

Where the outer product, $\wedge$, of two vectors forms a *bivector* which is interpreted as a directed area. The geometric product $\mathbf{ab}$ is therefore the sum of a scalar, $\mathbf{a}\cdot\mathbf{b}$, and a bivector, $\mathbf{a}\wedge\mathbf{b}$. In 3 dimensions the the *trivector* $(\mathbf{a}\wedge\mathbf{b})\wedge\mathbf{c}$ is an oriented 3-dimensional volume obtained by sweeping the bivector $\boldsymbol{a}\wedge\boldsymbol{b}$ along the vector $\mathbf{c}$.

In a space of dimension n there are multivectors of grade 0 (scalars), grade 1 (vectors), grade 2 (bivectors), grade 3 (trivectors), etc... up to grade n. Any two such multivectors can be multiplied using the geometric product. Consider two multivectors $\boldsymbol{A}_r$ and $\boldsymbol{B}_s$ of grades r and s respectively. The geometric product of $\boldsymbol{A}_r$ and $\boldsymbol{B}_s$ can be written as

$$\boldsymbol{A}_r\boldsymbol{B}_s = \langle\boldsymbol{AB}\rangle_{r+s} + \langle\boldsymbol{AB}\rangle_{r+s-2} + \ldots + \langle\boldsymbol{AB}\rangle_{|r-s|} \tag{2}$$

where $\langle\boldsymbol{M}\rangle_t$ is used to denote the t-grade part of multivector $\boldsymbol{M}$, e.g. $\langle\boldsymbol{ab}\rangle = \langle\boldsymbol{ab}\rangle_0 + \langle\boldsymbol{ab}\rangle_2 = \boldsymbol{a}\cdot\boldsymbol{b} + \boldsymbol{a}\wedge\boldsymbol{b}$. In the following sections expressions of grade 0 will be written ignoring their subindex, i.e. $\langle\boldsymbol{ab}\rangle_0 = \langle\boldsymbol{ab}\rangle = \boldsymbol{a}\cdot\boldsymbol{b}$.

2.2 Geometric Algebra and Rotors

For an n-dimensional space it can be introduced an orthonormal basis of vectors $\{\sigma_i\}$ $i = 1, ..., n$ such that $\sigma_i \cdot \sigma_j = \delta_{ij}$. This leads to a basis for the entire algebra:

$$1, \quad \{\sigma_i\}, \quad \{\sigma_i \wedge \sigma_j\}, \quad \{\sigma_i \wedge \sigma_j \wedge \sigma_k\}, \quad \ldots, \quad \sigma_1 \wedge \sigma_2 \wedge \ldots \wedge \sigma_n \tag{3}$$

Note that it shall not be used bold symbols for these basis vectors. The highest grade element is called the *pseudoscalar* of the space. Any multivector can be expressed in terms of this basis, and while it is often useful to do so, it can be stressed that the main strength of geometric algebra is the ability to carry out operations in a basis-free manner. The 2^3-dimensional *Pauli-Algebra* has the following basis:

$$\underbrace{1}_{scalar}, \underbrace{\{\sigma_1, \sigma_2, \sigma_3\}}_{vectors}, \underbrace{\{\sigma_1\sigma_2, \sigma_2\sigma_3, \sigma_3\sigma_1\}}_{bivectors}, \underbrace{\{\sigma_1\sigma_2\sigma_3\} \equiv i}_{trivector}. \tag{4}$$

By straightforward multiplication it can be easily seen that the three bivectors can also be written as

$$\sigma_2\sigma_3 = i\sigma_1 = \boldsymbol{i}, \quad \sigma_1\sigma_3 = -i\sigma_2 = \boldsymbol{j}, \quad \sigma_1\sigma_2 = i\sigma_3 = \boldsymbol{k}. \tag{5}$$

Using these simple bivectors it can be proved that the quaternion algebra of Hamilton is a subset of the geometric algebra of space. If a quaternion $\mathcal{A}$ is represented by $[a_0, a_1, a_2, a_3]$, then there exists a one-to-one mapping between quaternions and rotors given by

$$\mathcal{A} = [a_0, a_1, a_2, a_3] \leftrightarrow a_0 + a_1(i\sigma_1) + a_2(i\sigma_2) + a_3(i\sigma_3) \tag{6}$$

In order to find out more about rotors in the geometric algebra we note that any rotation can be represented by a pair of reflections. It can be easily shown that the result of reflecting a vector $\mathbf{a}$ in the plane perpendicular to a unit vector $\mathbf{n}$ is $\mathbf{a}_{-} - \mathbf{a}_{||} = -\mathbf{nan}$ where $\mathbf{a}_{-}$ and $\mathbf{a}_{||}$ respectively denote parts of $\mathbf{a}$ perpendicular and parallel to $\mathbf{n}$. Thus, a reflection of $\mathbf{a}$ in the plane perpendicular to $\mathbf{n}$, followed by a reflection in the plane perpendicular to $\mathbf{m}$ results in a new vector $-\mathbf{m}(-\mathbf{nan})\mathbf{m} = (\mathbf{mn})\mathbf{a}(\mathbf{nm}) = \mathbf{Ra\tilde{R}}$. The multivector $\mathbf{R} = \mathbf{mn}$ is called a rotor. It contains only even-grade elements and satisfies $\mathbf{R\tilde{R}} = 1$. In the 3-D space we use the term 'rotor' for those even elements of the space that represent rotations. Any rotor can be written in the form $\mathbf{R} = \pm e^{\mathbf{B}/2}$, where $\mathbf{B}$ is a bivector. In particular, in 3-D we write $\mathbf{R} = e^{(-i\frac{\theta}{2}\mathbf{n})} = \cos\frac{\theta}{2} - i\mathbf{n}\sin\frac{\theta}{2}$ which represents a rotation of θ radians anticlockwise about an axis parallel to the unit vector $\mathbf{n}$.

A potentially very useful expression for the rotation operator $\mathbf{R}$ of a m-dimensional multivector $\mathbf{x}$ is

$$\mathbf{y} = \mathbf{Rx\tilde{R}} = \mathbf{e}^{-\mathbf{B}/2}\mathbf{xe}^{\mathbf{B}/2} \tag{7}$$

where now $\mathbf{B}$ is a m-bladed bivector. This equation can be further decomposed into a sequence of rotations by angles $|\theta_{2k}|$ in particular i_{2k}-planes

$$\mathbf{y} = \mathbf{R_m R_{m-1}} \ldots \mathbf{R_1 x \tilde{R}_1} \ldots \mathbf{\tilde{R}_{m-1} \tilde{R}_m}. \tag{8}$$

where $\mathbf{R_k} = \mathbf{e}^{-\mathbf{B_{2k}}/2}$ and $\mathbf{\tilde{R}_k} = \mathbf{e}^{\mathbf{B_{2k}}/2}$.

3 Metric and Clifford Neural Networks

Classic neural network models and their training algorithms are essentially dependent of the metric, scalar product and norm. It is important to remark that all these mathematical characteristics are only related to the base of the vector space of the algebra and therefore fully independent of the attributed algebraic structure. In other words the metric is exclusively defined by the space modelling using a particular vector basis. The quality of a neural network design in an algebraic framework depends of the modelling of the space involving a particular metric and the relation of this modelling with its associated algebraic product.

Recently Person et al [5] extended the complex Perceptron developed by Georgiou and Koutsougeras [6] in the Clifford algebra framework. The approach of Pearson is unfortunately also limited to the Euclidean metric. This uses as net inputs $W \cdot \bar{X}$, net outputs $\boldsymbol{o}_i = \frac{W\bar{X}}{.}|WX|$ and as learning rule $W_{k+1} = W_k + \alpha_k \bar{X}_k$, where $\bar{X}_k$ is the conjugate vector. Common rules of complex conjugation are not automatically preserved in the Clifford algebra. A difference between a more general Clifford Algebra and complex numbers is shown by the equation $\boldsymbol{x}\bar{\boldsymbol{x}} = (|\boldsymbol{x}|, 0, s_3, ..., s_n)$ where $s_i \neq 0$ stands for the resultant multivector of grade i. One operation which is similar is $\overline{\boldsymbol{xy}} = \bar{\boldsymbol{y}}\bar{\boldsymbol{x}}$ $\forall \boldsymbol{x}, \boldsymbol{y}$ multivectors.

Pearson et al [5] used the same transfer function $\boldsymbol{u}(\boldsymbol{z}) = \frac{\boldsymbol{z}}{c + \frac{1}{r}|\boldsymbol{z}|}$ (where $\boldsymbol{z}$ is any multivector) as the one used by Georgiou et al which is based in the Euclidean norm in terms of the algebra $|\boldsymbol{x}| = \left(\sum_A [\boldsymbol{x}]_A^2\right)^{1/2}$ [2]. Georgiou et al [6] used for the prove of the learning rule of the complex perceptron explicitly special characteristics of the complex numbers like the Euler's function $e^{i\phi} = \cos\phi + i\sin\phi$. Due the noncommutative multiplication of the geometric product Pearson et al [5] had to prove their learning rule in a different way. After tests using the simulated Clifford backpropagation multi-layer perceptron of Pearson, the authors of this paper believe that the network using the norm of the

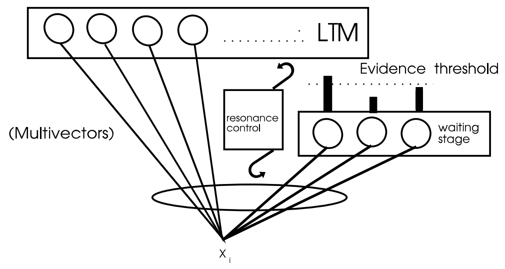

Figure 1.a Dynamic node coding.

Euclidean space behaves actually as a simple real valued backpropagation neural network and it has also convergence problems. As a result one can affirm that the design of the architecture and learning rule of the Clifford type neural networks is still an open question.

The previous brief analysis shows that one should look for more flexible structures which also allow the processing with non-Euclidean metrics. In this paper the authors present a neural network which can be formated in any Clifford Algebra $\mathcal{C}(V_{p,q})$. Therefore the network can process signals in any desired metric. The selforganization phase is implemented in terms of basic concepts of the resonance theory and in the supervised phase Clifford outstars are tuned. The design guaranties that the neural network is able to capture important characteristics of the geometric structure of the data. The authors believe that is the main motivation for geometric algebra applications in neural computing. This important scientific goal has been overlooked ever since.

4 Net Architecture and Training Algorithm

The learning procedure for the cases of the previous section has to minimize an error function $E(\vec{p})$ where $\vec{p}$ is the vector (not a multivector) which comprises all adjustable parameters.

In the Pearson's implementation [5], the vector $\vec{p}$ (weights and activation values) can be adjusted using the Clifford back propagation training rule. This procedure is unfortunately limited to the Euclidean metric. In contrast the selforganizing Clifford network allows according to the task, if necessary, a different metric. The learning procedure of the net consists basically in the first phase of an unsupervised method for the hidden layer, i.e. a multivector clustering algorithm, and a supervised one for the output layer. The second phase of learning is supervised and if it is still necessary helps to finetune all the net parameters. These phases and the recall mode are explained separately in the next subsections.

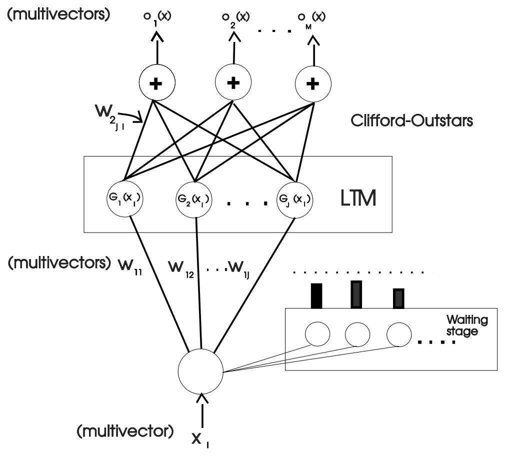

Figure 1.b Outstars labeling.

4.1 Unsupervised Learning

Figure 1.a depicts the evolution of the net architecture during selfsorganization. This process is in some aspects similar to the adaptive resonance theory selforganization [7]. At the very beginning the waiting memory and the long term memory have virgin nodes. The input patterns give information of different events and can resonate with a corresponding existing node. This capability of the net is implemented by a resonance detector and control mechanism using a task dependent metric and competitive learning. Note that the geometric algebra approach allows the use of a specific metric for a particular task. When a winner node is selected its weights are smoothly further adjusted. During the coding of a new pattern the control mechanism updates the weights of a resonant node which could be either in the long term memory or in the waiting stage. A candidate node resides in the waiting stage until it surpasses an evidence threshold, then it will be shifted to the long term memory. After the net is stable the parameters of the radial basis functions for each node are computed. These will be used for computing the resonance grade or membership grade (λ_j) of an input multivector with the jth-nodes. This factor will play an important role in the inhibition effect of the non-resonant nodes (winner-takes-all) during the training of the next layer and during the recall mode.

4.2 Supervised learning

In this stage of the training the multivector weights of the output layer have to be adjusted, see figure 1.b . Passing again the training patterns, the weights of the outstar of the resonant nodes are adapted using the following simple rule

$$\boldsymbol{w}_{2_{ji}}(k+1) = \boldsymbol{w}_{2_{ji}}(k) + \lambda_j \alpha(k)(\boldsymbol{o}_{d_i} - \sum_{j=1}^{J} \lambda_j \boldsymbol{w}_{2_{ji}}(k)) \tag{9}$$

where i is the multivector connection to the ith-output, λ_j is a constant and indicates the degree of the participation of the node j, $\alpha(k)$ is a gain factor and $\boldsymbol{o}_{d_i}$ the desired $i-th$ output multivector. All multiplications are geometric products and each output $\boldsymbol{o}_i$ supplies a multivector. Each output multivector could be composed as the geometric product of two multivectors $\boldsymbol{o}_i = \boldsymbol{e}_i \boldsymbol{y}_i$ where $\boldsymbol{e}_i$ could be set to a projective split vector [2] or to the scalar unity, i.e. $\boldsymbol{e}_i = 1$, for the case of a simple multivector association. The projective split can be used to connect the input and output spaces of different dimensions and metric. The multivector $\boldsymbol{y}_i$ could be set to $\boldsymbol{w}_{1i}$ or any other. As a result the invariant properties of the input patterns can be enhanced and made more observables for the net and in some cases the nonlinearity in one space can be easily transformed to a linear one in the representation of the other space. Here we can also appreciate the coordinate-free advantages of the geometric algebra. Once an initial solution is found after the first training phase, a supervised learning method can be additionally used in order to fine-tune all the network parameters. According to an error function $E(\vec{p})$ the vector $\vec{p}$, which comprises $\boldsymbol{w}_{1i}$ $\boldsymbol{w}_{2_{ji}}$ will be adjusted after each input and $i-th$ output values $\boldsymbol{x}_k, f_i(\boldsymbol{x}_k)$ using the descend gradient of the functional

$$e(\vec{p}, \boldsymbol{x}_k, f(\boldsymbol{x}_k)) = \frac{1}{2}|o_{i_{\vec{p}}}(\boldsymbol{x}_k) - f_i(\boldsymbol{x}_k)|^2 \tag{10}$$

as follows

$$\vec{p}_k = \vec{p}_{k-1} - \alpha_k \Big(\nabla[e(\vec{p}_{k-1}, \boldsymbol{x}_k, f_i(\boldsymbol{x}_k))]\Big) \tag{11}$$

This requires partial derivatives with respect to multivectors [2].

4.3 Recall mode

In the recall mode the outputs of the radial basis functions moderate the participation of the resonant nodes at the output energy level. This is captured by a simple equation

$$\boldsymbol{o}_i = \sum_{j=1}^{J} \lambda_j \boldsymbol{w}_{2_{ji}} \tag{12}$$

where $\boldsymbol{o}_i$ is the ith output, J is the amount of hidden nodes, λ_j is the degree of the participation of node j and is computed from the radial basis functions.

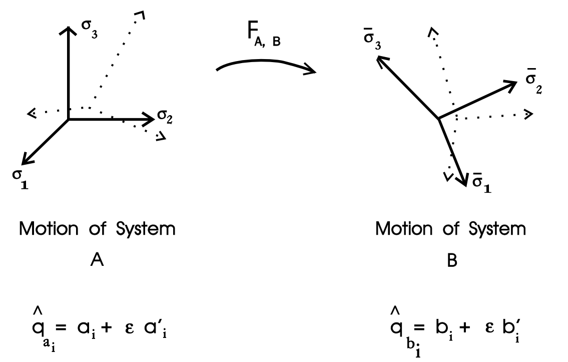

Figure 2. Mapping between two motion space.

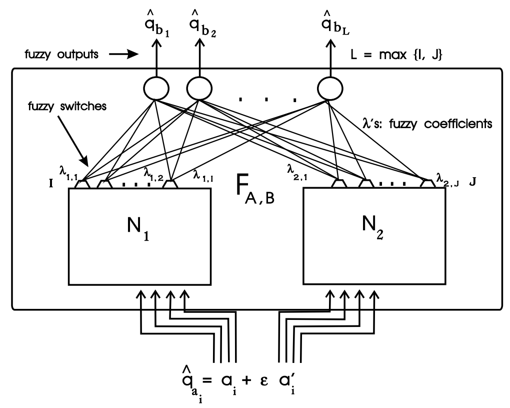

Figure 3. Combined structure for fuzzy clustering.

5 Experimental Results

The motions of reference frames of joints in robotics can be nicely represented using screws or dual quaternions. Figure 2 depicts the geometric abstraction of the problem.

For this experiment the range of movements was limited to a practical narrow area. For the approximation of this mapping a combined structure using two Clifford selforganizing neural networks was implemented. This is presented in Figures 3.

The two neural networks were set up in the Clifford algebra $\mathcal{C}(\mathbf{V}_{0,3})$ [2] accordingly and applied to approximate the mapping between the screw motions of systems A and B. After the selforganization of each network one has recognized a reduced number of long term nodes I and J, i.e clusters of the real and dual part of the dual quaternions. Here we used in fast learning [7] a moderate categorization threshold ρ. In the supervised phase the radial basis functions are tuned and then the Clifford outstarts are adjusted. The structure is full connected and the weights of the outstars are also quaternions, see Figure 3. Note that the amount of the outputs is automatically defined by the bigger number of clusters of the nets, i.e L=MAX(I,J). After supervised training the net was recalled with previously unseen patterns and due to its nice capability of fuzzy outputs the net was able to follow the deviation of the main classes as expected. Some pattern examples are presented below. The ideal dual quaternions at the output are presented in Table I. The outputs in terms of dual quaternions for each category (Cat.) and its λ_{I-MAX} and λ_{J-MAX} of the hidden layers are presented in the Table II for a combined structure with I=J=L=5. The combined network with I=J=5 has a better performance than a combined with I=5>J, because it has more nodes dedicated for clustering. When a lower ρ is used as in the case I=5>J, the coding of the dual element is more rough affecting the overall performance. For space reasons we can not include comparative experiments for this case. It may be possible for other application that a combined structure with I≠J suffices. Therefore in general it is better that the left and the right modules should use independent ρ's.

$Cat.L$	b_0	b_1	b_2	b_3	b'_0	b'_1	b'_2	b_3
1	0.998	0.023	0.031	0.046	-0.061	0.411	0.508	0.754
2	0.927	0.143	0.191	0.287	-0.374	0.528	0.460	0.635
3	0.979	0.076	0.102	0.153	-0.199	0.468	0.494	0.711
4	0.874	0.186	0.248	0.372	-0.484	0.559	0.429	0.572
5	0.771	0.244	0.325	0.488	-0.636	0.589	0.370	0.462

Table I: Expected dual quaternions at the output.

$Cat.$	b_0	b_1	b_2	b_3	b'_0	b'_1	b'_2	b_3	λ_{I-MAX}	λ_{J-MAX}
1.1	+0.998	+0.023	+0.031	+0.046	-0.060	+0.411	+0.509	+0.754	0.987	0.793
1.2	+0.999	+0.018	+0.024	+0.036	-0.047	+0.405	+0.510	+0.758	0.764	0.713
2.1	+0.927	+0.144	+0.192	+0.288	-0.374	+0.529	+0.460	+0.636	0.531	0.999
2.2	+0.936	+0.136	+0.181	+0.271	-0.353	+0.522	+0.466	+0.646	0.940	0.697
3.1	+0.981	+0.075	+0.100	+0.150	-0.195	+0.466	+0.495	+0.714	0.976	0.8850
3.2	+0.982	+0.073	+0.097	+0.146	-0.190	+0.465	+0.495	+0.715	0.952	0.708
4.1	+0.877	+0.184	+0.246	+0.369	-0.480	+0.558	+0.431	+0.576	0.988	0.979
4.2	+0.878	+0.184	+0.245	+0.368	-0.479	+0.558	+0.431	+0.576	0.987	0.975
5.1	+0.793	+0.234	+0.312	+0.468	-0.609	+0.585	+0.383	+0.485	0.950	0.790
5.2	+0.795	+0.233	+0.310	+0.466	-0.606	+0.585	+0.384	+0.487	0.945	0.787

Table II: Dual quaternions at the output structure with I=J=L=5.

6 Conclusion

This paper presents a novel selforganizing type RBF network using the Clifford algebra framework. The authors have shown that the use of geometric algebra helps enormously to improve the potential of network structures and to simplify the learning algorithms. In the network a new type of embedded processing called projective split can be added for feature enhancement and better invariants recognition. This type of neural networks can be also cascaded in order to process patterns using different space dimension and metric, the latter being possibly only due the projective split. The potential of such nets working in a specific Clifford algebra $\mathcal{C}(V_{p,q})$ is shown by a simple application of frame coordination in robotics. A combined structure consisting of two Clifford nets in parallel is applied for fuzzy clustering of dual quaternions.

References

[1] Brackx F., Delanghe R. and Sommen, F. 1982. Clifford Analysis. Research Notes in Mathematics 76, Pitman Advanced Publishing Program, London.

[2] Hestenes, D. and Sobczyk, G. 1984. Clifford Algebra to Geometric Calculus: A unified language for mathematics and physics. *D. Reidel*, Dordrecht.

[3] Lasenby, J. 1995. Engineering applications of geometric algebra. to appear in Proceedings of Banff Summer School on *Geometric Algebras in Physics.* August 1995. Birkhauser Boston.

[4] Bayro-Corrochano, E., Lasenby, J. and Sommer, G. 1996. Geometric Algebra: a framework for computing point and line correspondences and projective structure using n uncalibrated cameras. Proceedings of ICPR'96, Vienna.

[5] Pearson J.K. and Bisset D.L. 1992. Back propagation in a Clifford algebra. *Artificial Neural Networks, 2, I. Aleksander and J. Taylor (Ed.)*, 413:416, 1992.

[6] Georgiou G. M. and Koutsougeras C, 1992. Complex domain backpropagation. *IEEE Trans. on Circuits and Systems*, 330:334.

[7] Carpenter G.A. and Grossberg S, 1987. ART-2: Selforganization of stable category recognition codes for analog input patterns. *Appl. Optics*, 26(23), 4919:4930.