A General Framework for Subspace Detection in Unordered Multidimensional Data

Leandro A. F. Fernandes [a], Manuel M. Oliveira [b]

[a] *Instituto de Computação, Universidade Federal Fluminense (UFF)*
CEP 24210-240 Niterói, RJ, Brazil
Tel +55 21 2629-5665, Fax +55 21 2629-5669

[b] *Instituto de Informática, Universidade Federal do Rio Grande do Sul (UFRGS),*
CP 15064 CEP 91501-970, Porto Alegre, RS, Brazil

Abstract

The analysis of large volumes of unordered multidimensional data is a problem confronted by scientists and data analysts every day. Often, it involves searching for data alignments that emerge as well-defined structures or geometric patterns in datasets. For example, straight lines, circles, and ellipses represent meaningful structures in data collected from electron backscatter diffraction, particle accelerators, and clonogenic assays. Also, customers with similar behavior describe linear correlations in e-commerce databases. We describe a general approach for detecting data alignments in large unordered noisy multidimensional datasets. In contrast to classical techniques such as the Hough transforms, which are designed for detecting a specific type of alignment on a given type of input, our approach is independent of the geometric properties of the alignments to be detected, as well as independent of the type of input data. Thus, it allows concurrent detection of multiple kinds of data alignments, in datasets containing multiple types of data. Given its general nature, optimizations developed for our technique immediately benefit all its applications, regardless the type of input data.

Key words: Hough transform, geometric algebra, parameter space, subspace detection, shape detection, blade, Grassmannian, coordinate chart, line, circle, plane, sphere, conic section, flat, round, quadric

Email addresses: laffernandes@ic.uff.br (Leandro A. F. Fernandes), oliveira@inf.ufrgs.br (Manuel M. Oliveira).

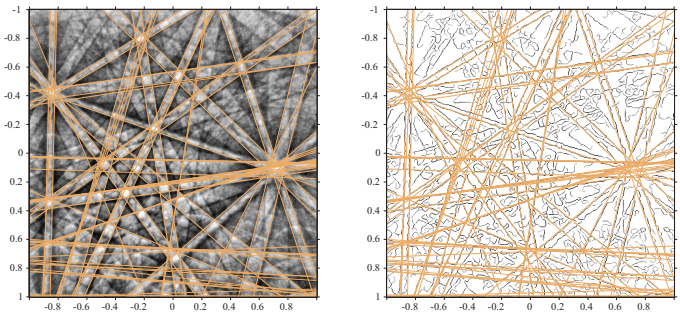
URLs: http://www.ic.uff.br/~laffernandes (Leandro A. F. Fernandes), http://www.inf.ufrgs.br/~oliveira (Manuel M. Oliveira).

1 Introduction

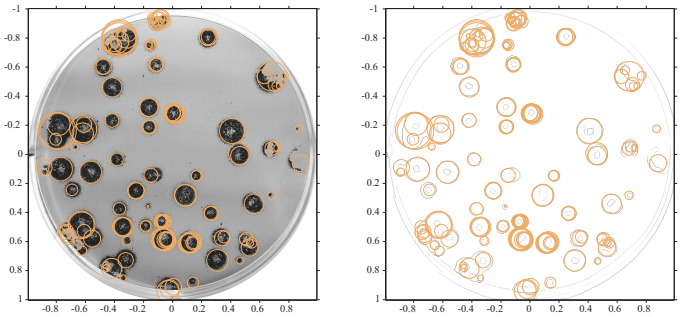
Data analysis is a fundamental element in scientific discovery and data mining. In many scientific fields, visual inspection of experimental datasets is often performed in order to identify strong local coherence in the data. Such coherence results from data alignments (in some multidimensional space), and usually emerges as geometric shapes and patterns. For instance, straight lines and circles appear as well-defined structures in the analysis of electron backscatter diffraction (Fig. 1a) and clonogenic essays (Fig. 1c), respectively. However, when large volumes of data need to be analyzed, visual inspection becomes impractical. For this reason, automatic detectors for specific types of data alignments have been broadly applied by scientists in many different areas, such as particle physics [1,2], astronomy [3,4], microbiology [5,6], crystallography [7,8], and medicine [9,10]. Such detectors are also a central component of many computer vision and image processing applications [11–13]. The goal of automatic detectors is to identify certain kinds of alignments that best fit a given unordered dataset, even in presence of noise and discontinuities.

We describe a general approach for detecting data alignments in unordered noisy multidimensional data. Our approach is based on the observation that a wide class of alignments, and also input data entries, can be represented as linear subspaces. Thus, instead of defining a different detector for each specific case and input data type, it is possible to design a unifying framework to detect the occurrences of emerging subspaces in multidimensional datasets. In our framework, these datasets may be heterogeneous and contain entries with different dimensionalities (Fig. 2).

Our approach has a broad range of applications as a pattern detection tool. For instance, it can be applied, without any changes, to all kinds of data alignments that can be represented as linear subspaces in any complete metric spaces (see Section 3). Examples include, but are not limited to, data alignments decomposed into real- or complex-valued vector spaces, orthogonal polynomials, wavelets, and spherical harmonics. For the purpose of illustration, however, we restrict the examples shown in the paper to the important problem of detecting analytic geometric shapes in real-valued spaces of arbitrary dimensionalities. By assuming a model of geometry (MOG), subspaces can be interpreted as shapes (see Supplementary Material A for examples). In such a case, the proposed formulation becomes a general analytical shape detector whose definition does not depend on the shape one wants to detect nor on the input data type.



(a) 445 × 445 pixels (b) 445 × 445 pixels



(c) 529 × 534 pixels (d) 529 × 534 pixels

Fig. 1. (a) Electron backscatter diffraction image taken from a particle of wulfenite. The detection of straight lines is key for the identification of the particle's crystalline phase. (c) Gray image of infection with H1N1 in MDCK-SIAT1 cells. The detection of circles is important for automated counting process in clonogenic assays. Our approach was used, without any changes, to automatically detect the straight lines and circles shown in (a) and (c) from the edge information shown in (b) and (d), respectively.

1.1 Contributions and Demonstrations

The main contributions of this paper include:

- A general approach for subspace detection in unordered multidimensional datasets (Section 4);
- A parameterization scheme for subspaces based on the rotation of a canonical subspace with the same dimensionality (Section 4.1); and
- An algorithm that enumerates all instances of subspaces with a given dimensionality p that either contain or are contained by an input subspace of arbitrary dimensionality (Section 4.2).

In addition, the following assertions will be demonstrated:

- The detection can be driven to a data alignment type by changing the assumed MOG where data have been encoded, while the presented formulation remains unchanged (Section 4);

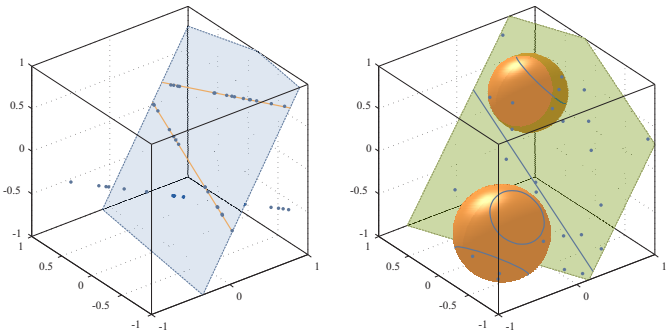


Fig. 2. Shape detection on heterogeneous synthetic datasets using our approach. (left) Detection of lines on the input plane that best fit subsets of input points. (right) Concurrent detection of plane and spheres by a single application of the proposed approach. The input dataset is comprised by points, circles, and straight line.

- The intended p-dimensional subspaces are represented with the smallest possible number of parameters (Section 4.1.2);
- It allows the detection of subspaces that best fit an input set of subspaces with different dimensionalities and different geometric interpretations (*e.g.,* the detection of straight lines that best fit points, directions, and/or planes – Fig. 2, left).
- It allows the concurrent detection of subspaces with different interpretations but with the same dimensionality in a given MOG (*e.g.,* planes and spheres in conformal MOG – Fig. 2, right);

When used as an analytic shape detector, our approach presents the following features:

- It is the generalization of the HTs for analytic shapes representable by linear subspaces;
- It leads to the most compact parameterization of analytical shapes (*e.g.,* straight lines, circles, and general conic sections in the plane are parameterized with two, three, and four parameters, respectively); and
- An approximation of the dth-order Voronoi diagram [14] of a set of points in $\mathbb{R}^d$ can be retrieved as a byproduct of the detection of subspaces geometrically interpreted as circles, spheres, and their higher-dimensional counterparts.

We formulate our subspace detector using geometric algebra (GA) [15–17]. We use GA notation instead of more conventional formalisms (*e.g.,* matrix algebra) because GA is a mathematical framework that treats subspaces as primitives for computation. As such, it is an appropriate tool for modeling the subspace-detection problem. Moreover, GA naturally generalizes and integrates useful formalisms such as complex numbers, quaternions, and Plücker coordinates, among others, into a high-level specification language for geometric operations. GA defines products with clear geometrical meaning, which lead to compact and easy to implement formulations.

1.2 Related Work

1.2.1 Hough transform

When applied as a shape detector, our approach can be seen as a generalization of the class of techniques commonly referred to as **Hough transforms** (HTs) [18,19]. A standard HT can be defined for simple shapes that can be represented by a **model function**

$$f(x_1, x_2, \cdots, x_d; p_1, p_2, \cdots, p_m) = 0, \tag{1}$$

where $\boldsymbol{x} = (x_1, x_2, \cdots, x_d) \in \mathbb{R}^d$ is a point from the input dataset, and $\boldsymbol{p} = (p_1, p_2, \cdots, p_m) \in \mathbb{P}^m$ is a vector in a **parameter space** $\mathbb{P}^m$. Each parameter vector $\boldsymbol{p}$ in $\mathbb{P}^m$ characterizes an instance of that shape. By using a **mapping procedure**, the HT takes each input point $\boldsymbol{x}$ and determines all instances of the shape potentially passing through $\boldsymbol{x}$. For each such an instance, its associated parameter vector $\boldsymbol{p}$ is used to address a bin in an **accumulator array** (*i.e.,* a discrete representation of $\mathbb{P}^m$). The value stored in the bin is then incremented by some **importance value** ω (usually *one*). At the end of the process, the bins having the largest values correspond to the parameters of the most likely shapes in the input data.

The mapping procedure is obtained from the model function (1) and consists in arbitrating a predefined subset of k parameter values from $\boldsymbol{p}$ (for $k < m$) and computing the remaining $(m - k)$ parameter values using a **mapping function** with the form:

$$\{p_{k+1}, p_{k+2}, \cdots, p_m\} = g\left(x_1, \cdots, x_d; p_1, \cdots, p_k\right). \tag{2}$$

Function (2) is evaluated for all $\{p_1, \cdots, p_k\} \in \mathbb{P}^k$.

Traditionally, each HT handles a specific detection case. As a result, a different model (1) and mapping function (2) have been designed to detect each specific geometric shape in a given type of input data. For instance, Duda and Hart [20] use the normal equation of the line as model function. The circle detection of Kimme *et al.* [21], instead, uses the center-radius parameterization. Such parameterizations are intrinsically different from each other.

Our approach differs from standard HTs [20–24] by defining a single closed-form solution that parameterizes any subspace as a rotation of a canonical subspace with the same dimensionality. Such a parameterization is independent of the (geometric) interpretation of the subspaces. Moreover, the mapping procedure is also independent of the input data type. Thus, our approach systematically adapts itself to the detection of shapes in unordered heterogeneous datasets having arbitrary dimensionality (Fig. 2). Supplementary Materials B

and C include derivations showing that standard HTs are particular cases of the proposed approach.

The *Generalized Hough Transform* (GHT) is a Hough-like method for detecting shapes (in images) that *cannot be represented analytically* [24]. The method allows the identification of the occurrences of the shape regarding changes in location, orientation, and scaling. The extension of the GHT to 3-dimensional shapes is described by Wang and Reeves [25]. We are concerned with the detection of subspaces interpreted as analytic shapes. Thus, our generalized approach targets a different problem than the GHT. As pointed out by Leavers [19], the GHT is not suitable for the detection of analytic shapes because it does not offer an efficient representation of all such shapes. In order to achieve an efficient representation, the parameterization of the GHT must be explicitly changed.

A naive implementation of standard HTs and of the proposed approach suffers from the same drawbacks: large memory requirements and high computational cost. However, as any HT, the proposed approach is robust to the presence of outliers and is suitable for implementation on massively parallel architectures. Moreover, the generality of our approach guarantees that any optimization immediately benefits the processing of all detectable shapes. The attempts to minimize drawbacks in standard HTs, on the other hand, are targeted at particular versions of HTs, due to specificities in their formulations [18,19]. Thus, optimizations to the HT need to be done on a case-by-case basis. For instance, O'Gorman and Clowes [26] pointed out that line detection from feature pixels may be optimized by the use of gradient information computed for the pixels. Kimme *et al.* [21] follow such an approach providing the same level of optimization to the HT for circles in images. Note that the same optimization took almost two years to be extended to a single case of HT.

1.2.2 Tensor Voting

The Tensor Voting (TV) framework [27] is a unified methodology for the robust inference of local features from data. Such features are retrieved in terms of emerging surfaces, curves, and labeled junctions in a given set of points, points with an associated tangent direction, points with an associated normal direction, or any combination of the above.

While TV can compete with HT in terms of robustness against noise, it is, however, inherently model-free. As a result, it cannot be efficiently applied to the detection of predefined types of data alignments. By definition, it retrieves, at the same time, all the salient structures (with any dimensionality) embedded in a dataset. A subsequent filtering step is required in order to perform the detection of some intended type of structure. Our approach detects

the occurrences of emerging subspaces with a given dimensionality p in multidimensional datasets. The detection can be driven to a specific type of data alignment just by choosing p, and by changing the assumed MOG where data have been encoded.

1.2.3 Subspace and Submanifold Clustering

The goal of subspace clustering techniques is to find, among all possible linear or affine subspaces, those that accommodate as many database objects as possible (see [28] for a review). A common practice to these techniques is to assume Euclidean metric to the ambient space and linear relations between pairs of features [29–32]. For cases where input data can be distributed along nonlinear submanifolds, one can assume that input entries are embedded in Euclidean space and use (at least locally) the Euclidean metric or a variation of it to perform clustering [33–35]. However, there are several situations where it is more natural to consider features that live in a non-Euclidean space (*e.g.,* diffusion tensor imaging segmentation). For those cases, Riemannian spaces have been used [36]. But, as pointed out by Goh and Vidal [36], even with Riemannian spaces the specific calculations vary depending on the application. In a recent work, Favaro *et al.* [37] proposed a closed-form solution for subspace estimation and clustering. However, their formulation constraints the detected elements to linear and affine subspace and the type of input data to points only.

In contrast to existing subspace and submanifold clustering approaches, the calculations performed by our closes-form detection framework are not tailored to specific applications or input data type. By definition, the proposed formulation systematically adapts itself to the MOG where data is being encoded.

2 Geometric Algebra

This section presents a brief introduction to the concepts of GA required for understanding our approach. Due to space restrictions, the primary goal of this section is to introduce the notational convention adopted in the paper rather than to introduce a complete view of the works in geometric or Clifford algebra. A more detailed introduction to GA can be found in [15]. The books by Dorst *et al.* [16], Perwass [17] and Hestenes [38], and the tutorials by Doran *et al.* [39,40] and Hildenbrand *et al.* [41] provide in-depth treatments to the subject. Supplementary Material D presents a quick reference guide to the notational convention used in this paper.

Subspaces, or **blades** (in GA notation), are the *basic computational elements* in GA. A k-blade represents a weighted k-dimensional oriented subspace spanned by the **outer product** ($\wedge$) of k independent vectors. Thus, for instance, a scalar value $\alpha \in \mathbb{R}$ is a 0-blade, a vector $\mathbf{a} \in \mathbb{R}^n$ is a 1-blade, and a 2-blade can be computed as $\mathbf{C}_{\langle 2 \rangle} = \mathbf{a} \wedge \mathbf{b}$ for linearly independent vectors $\mathbf{a}$ and $\mathbf{b}$.

In linear algebra, k-dimensional subspaces are represented as collections of k vectors in a vector space $\mathbb{R}^n$, for $0 \leq k \leq n$. Vectors are the primitive elements of linear algebra and are expressed as a weighted sum of the basis elements $\{\mathbf{e}_i\}_{i=1}^n$ assumed for $\mathbb{R}^n$, *e.g.*:

$$\mathbf{a} = \alpha_1 \, \mathbf{e}_1 + \alpha_2 \, \mathbf{e}_2 + \alpha_3 \, \mathbf{e}_3 \in \mathbb{R}^3.$$

In order to treat k-blades as computational primitives, GA decomposes them in a basis of blades comprised by the outer product of k vectors in the set $\{\mathbf{e}_i\}_{i=1}^n$ of basis vectors. For instance, the outer product of two general vectors in $\mathbb{R}^3$ naturally induces the representation of 2-blades as the weighted sum of the 2-dimensional basis blades $\mathbf{e}_1 \wedge \mathbf{e}_2$, $\mathbf{e}_1 \wedge \mathbf{e}_3$, and $\mathbf{e}_2 \wedge \mathbf{e}_3$:

$$\begin{aligned}
\mathbf{C}_{\langle 2 \rangle} &= \mathbf{a} \wedge \mathbf{b} \\
&= (\alpha_1 \, \mathbf{e}_1 + \alpha_2 \, \mathbf{e}_2 + \alpha_3 \, \mathbf{e}_3) \wedge (\beta_1 \, \mathbf{e}_1 + \beta_2 \, \mathbf{e}_2 + \beta_3 \, \mathbf{e}_3) \\
&= \quad (\alpha_1 \, \beta_2 - \alpha_2 \, \beta_1) \, \mathbf{e}_1 \wedge \mathbf{e}_2 \\
&\quad + (\alpha_1 \, \beta_3 - \alpha_3 \, \beta_1) \, \mathbf{e}_1 \wedge \mathbf{e}_3 \\
&\quad + (\alpha_2 \, \beta_3 - \alpha_3 \, \beta_2) \, \mathbf{e}_2 \wedge \mathbf{e}_3.
\end{aligned} \tag{3}$$

The algebraic manipulation in (3) is obtained by recalling that by *linearity*, the outer product is distributive over the sum, and scalar values commute. By *antisymmetry*, the outer product of two vectors commute at the cost of a sign change (*e.g.,* $\mathbf{e}_2 \wedge \mathbf{e}_1 = -\mathbf{e}_1 \wedge \mathbf{e}_2$). Thus, the outer product of a vector with itself disappears (*i.e.,* $\mathbf{e}_i \wedge \mathbf{e}_i = 0$).

The expression in (3) can be extended to represent any k-dimensional subspace as the weighted sum of $\binom{n}{k}$ k-dimensional basis blades. The space defined by the $\sum_{k=0}^{n} \binom{n}{k} = 2^n$ blades (treated as basis elements) is called the **multivector space** $\bigwedge \mathbb{R}^n$ built from a vector space $\mathbb{R}^n$. The concept of subspace and the construction of blades using the outer product are *independent of any metric properties* a vector space $\mathbb{R}^n$ or its associated multivector space $\bigwedge \mathbb{R}^n$ might have.

Section 4 presents our general approach for detecting k-dimensional subspaces (*i.e.,* k-blades). In such an approach, an arbitrary k-blade $\mathbf{B}_{\langle k \rangle}$ is characterized through the parameterization of its properties:

attitude The equivalence class $\gamma\,\mathbf{B}_{\langle k\rangle}$, for any $\gamma \in \mathbb{R}$.
weight The value of γ in $\mathbf{B}_{\langle k\rangle} = \gamma\,\mathbf{J}_{\langle k\rangle}$, where $\mathbf{J}_{\langle k\rangle}$ is a reference blade with the same attitude as $\mathbf{B}_{\langle k\rangle}$.
orientation The sign of the weight relative to $\mathbf{J}_{\langle k\rangle}$.

2.2 Geometric Product

The **geometric product** has no special symbol and it is denoted by a *thin space*. For real values (*i.e.,* 0-blades) it is equivalent to the standard multiplication operation. For vectors, it is defined as the linear combination of the vector inner product ($\cdot$) and the outer product:

$$\mathbf{a}\,\mathbf{b} = \mathbf{a}\cdot\mathbf{b} + \mathbf{a}\wedge\mathbf{b}. \tag{4}$$

The outcome of (4) is an element of mixed dimensionality. It is the sum of a scalar value computed from the inner product characterizing the metric relation of the vectors, and a 2-blade computed from the nonmetric outer product. The formulation extending (4) to arbitrary terms can be found in [15,16].

There are many products in GA, which are special cases of the geometric product. In this paper, we are concerned with three of them – the **outer product**:

$$\mathbf{A}_{\langle r\rangle}\wedge\mathbf{B}_{\langle s\rangle} = \left\langle \mathbf{A}_{\langle r\rangle}\,\mathbf{B}_{\langle s\rangle}\right\rangle_{r+s}; \tag{5}$$

the **scalar product**:

$$\mathbf{A}_{\langle r\rangle} * \mathbf{B}_{\langle s\rangle} = \left\langle \mathbf{A}_{\langle r\rangle}\,\mathbf{B}_{\langle s\rangle}\right\rangle_{0}; \tag{6}$$

and the **left contraction**:

$$\mathbf{A}_{\langle r\rangle}\rfloor\,\mathbf{B}_{\langle s\rangle} = \left\langle \mathbf{A}_{\langle r\rangle}\,\mathbf{B}_{\langle s\rangle}\right\rangle_{s-r}. \tag{7}$$

In (5), (6), and (7), $\mathbf{A}_{\langle r\rangle}$ and $\mathbf{B}_{\langle s\rangle}$ are blades, $\langle\square\rangle_k$ is used to retrieve the k-dimensional part of the geometric product of $\mathbf{A}_{\langle r\rangle}$ and $\mathbf{B}_{\langle s\rangle}$ (the outcome of $\mathbf{A}_{\langle r\rangle}\,\mathbf{B}_{\langle s\rangle}$ may have mixed dimensionality).

The outer product introduced in Section 2.1 and defined in (5) is used to span a $(r+s)$-dimensional subspace from blades of dimensionality r and s. If there is at least one common vector factor in the multiplied terms then the outcome is *zero*.

The scalar product (6) is the generalization of the vector inner product to arbitrary subspaces with same dimensionality (*i.e.,* $r = s$). Under Euclidean metric, the resulting scalar value is proportional to the cosine of the angle between the subspaces and to the weight of the multiplied terms. For blades with different dimensionality the outcome is *zero*. In Section 4.2 we use the

scalar product to check whether two blades have the same dimensionality and are not orthogonal.

The scalar product can be used to compute the **squared norm** of a blade:

$$\left\|\mathbf{A}_{\langle k\rangle}\right\|^2 = \mathbf{A}_{\langle k\rangle} * \widetilde{\mathbf{A}}_{\langle k\rangle}, \tag{8}$$

where $\widetilde{\mathbf{A}}_{\langle k\rangle} = (-1)^{k\,(k-1)/2}\,\mathbf{A}_{\langle k\rangle}$ denotes the **reverse** of the subspace. A blade is invertible if it has a nonzero norm. The **inverse** $\mathbf{A}^{-1}_{\langle k\rangle}$ of a blade $\mathbf{A}_{\langle k\rangle}$ satisfies $\mathbf{A}_{\langle k\rangle}\,\mathbf{A}^{-1}_{\langle k\rangle} = \mathbf{A}^{-1}_{\langle k\rangle}\,\mathbf{A}_{\langle k\rangle} = 1$, and is computed as:

$$\mathbf{A}^{-1}_{\langle k\rangle} = \frac{\widetilde{\mathbf{A}}_{\langle k\rangle}}{\left\|\mathbf{A}_{\langle k\rangle}\right\|^2}.$$

The geometric interpretation of the left contraction presented in (7) can be described as removing from $\mathbf{B}_{\langle s\rangle}$ the part that is "like" $\mathbf{A}_{\langle r\rangle}$, returning the $(s-r)$-dimensional subspace that is contained in $\mathbf{B}_{\langle s\rangle}$ and is "unlike" $\mathbf{A}_{\langle r\rangle}$. Under Euclidean metric, the portion of $\mathbf{A}_{\langle r\rangle}$ that is like $\mathbf{B}_{\langle s\rangle}$ is the orthogonal projection of $\mathbf{A}_{\langle r\rangle}$ onto $\mathbf{B}_{\langle s\rangle}$. Therefore, the left contraction returns the subspace in $\mathbf{B}_{\langle s\rangle}$ that is orthogonal to the projection of $\mathbf{A}_{\langle r\rangle}$, and hence orthogonal to $\mathbf{A}_{\langle r\rangle}$.

2.3 Dual Representation of Subspaces

The left contraction can be used to compute the **dual** representation of a subspace. The **dual** representation of a subspace $\mathbf{A}_{\langle k\rangle}$ is its $(n-k)$-dimensional orthogonal complement with respect to the total (n-dimensional) space:

$$\mathbf{A}^{*}_{\langle k\rangle} = \mathbf{A}_{\langle k\rangle} \rfloor \mathbf{I}^{-1}_{\langle n\rangle}, \tag{9}$$

where $\square^*$ denotes the **dual** operation, $\mathbf{I}_{\langle n\rangle} = \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \cdots \wedge \mathbf{e}_n$ is the **unit pseudoscalar** of the n-dimensional space. The complement of taking the dual is the **undual** operation:

$$\mathbf{D}^{-*}_{\langle n-k\rangle} = \mathbf{D}_{\langle n-k\rangle} \rfloor \mathbf{I}_{\langle n\rangle}. \tag{10}$$

By using this operation, the dual representation of a blade can be correctly mapped back to its direct representation (*i.e.,* $(\mathbf{A}^{*}_{\langle k\rangle})^{-*} = \mathbf{A}_{\langle k\rangle}$).

In Sections 4.1 and 4.2 we use the dual and undual operations to define a closed-form solution for subspace detection involving input and resulting blades of arbitrary dimensionalities.

2.4 Meet and Join of Subspaces

The **meet** and **join** products are the GA analogs of intersection and union operators from set theory. For any two blades $\mathbf{A}_{\langle r \rangle}$ and $\mathbf{B}_{\langle s \rangle}$ one can factor out a blade $\mathbf{M}_{\langle t \rangle}$ from both $\mathbf{A}_{\langle r \rangle}$ and $\mathbf{B}_{\langle s \rangle}$:

$$\mathbf{A}_{\langle r \rangle} = \mathbf{A}'_{\langle r-t \rangle} \wedge \mathbf{M}_{\langle t \rangle} \text{ and } \mathbf{B}_{\langle s \rangle} = \mathbf{M}_{\langle t \rangle} \wedge \mathbf{B}'_{\langle s-t \rangle}.$$

Meet returns the subspace shared by $\mathbf{A}_{\langle r \rangle}$ and $\mathbf{B}_{\langle s \rangle}$:

$$\mathbf{A}_{\langle r \rangle} \cap \mathbf{B}_{\langle s \rangle} = \mathbf{M}_{\langle t \rangle}, \tag{11}$$

while the join is the subspace spanned by the disjoint and by the common parts of $\mathbf{A}_{\langle r \rangle}$ and $\mathbf{B}_{\langle s \rangle}$:

$$\mathbf{A}_{\langle r \rangle} \cup \mathbf{B}_{\langle s \rangle} = \mathbf{A}'_{\langle r-t \rangle} \wedge \mathbf{M}_{\langle t \rangle} \wedge \mathbf{B}'_{\langle s-t \rangle}. \tag{12}$$

Both meet (11) and join (12) are independent of the particular metric since they are based on factorization by the (nonmetric) outer product.

Meet and join are nonlinear products. However, if $\mathbf{A}_{\langle r \rangle}$ and $\mathbf{B}_{\langle s \rangle}$ are disjoint, and their join is the total space (*i.e.,* $\mathbf{A}_{\langle r \rangle} \cup \mathbf{B}_{\langle s \rangle} = \mathbf{I}_{\langle n \rangle}$, the pseudoscalar), then the meet reduces to the **regressive product** ($\vee$):

$$\mathbf{A}_{\langle r \rangle} \cap \mathbf{B}_{\langle s \rangle} = \left(\mathbf{B}^*_{\langle s \rangle} \wedge \mathbf{A}^*_{\langle r \rangle}\right)^{-*} = \mathbf{A}_{\langle r \rangle} \vee \mathbf{B}_{\langle s \rangle}, \tag{13}$$

which is also linear and nonmetric. The regressive product can be regarded as the dual operation to the outer product. We use it in Section 4.1 while defining the proposed parameterization of subspaces.

2.5 Rotors

In GA a **rotor** is defined as the geometric product of an even number of unit invertible vectors. Under Euclidean metric, rotors encode rotations and generalize *quaternions* to n-dimensional spaces. The transformation encoded by a rotor $\boldsymbol{R}$ is applied to a k-blade $\mathbf{A}_{\langle k \rangle}$ by using a sandwiching construction involving the geometric product:

$$\mathbf{A}''_{\langle k \rangle} = \boldsymbol{R}\, \mathbf{A}_{\langle k \rangle}\, \widetilde{\boldsymbol{R}},$$

where $\mathbf{A}''_{\langle k \rangle}$ denotes the transformed subspace, and $\widetilde{\boldsymbol{R}}$ denotes the reverse of $\boldsymbol{R}$. In GA, the inverse of a rotor is equal to its reverse (*i.e.,* $\boldsymbol{R}^{-1} = \widetilde{\boldsymbol{R}}$), thus $\mathbf{A}_{\langle k \rangle} = \widetilde{\boldsymbol{R}}\, \mathbf{A}''_{\langle k \rangle}\, \boldsymbol{R}$.

As orthogonal transformations, rotors preserve both the inner and the outer products. This way, the structure preservation of rotors holds for the geometric product (4), and hence to all other products, *i.e.*:

$$\boldsymbol{R}\ (A \circ B)\ \widetilde{\boldsymbol{R}} = \left(\boldsymbol{R}\, A\, \widetilde{\boldsymbol{R}}\right) \circ \left(\boldsymbol{R}\, B\, \widetilde{\boldsymbol{R}}\right), \tag{14}$$

where the $\circ$ symbol represents *any* product of GA, and, as a consequence, any operation defined from the products (*e.g.,* inversion, duality, meet, and join).

An alternative (and more practical) way to define rotors is to use the exponential of 2-blades. Under Euclidean metric, the rotor $\boldsymbol{R}$ encoding a rotation of θ radians on the unit plane $\mathbf{P}_{\langle 2\rangle}$ is given by:

$$\boldsymbol{R} = \exp\left(-\frac{\theta}{2}\mathbf{P}_{\langle 2\rangle}\right) = \cos\left(\frac{\theta}{2}\right) - \sin\left(\frac{\theta}{2}\right)\mathbf{P}_{\langle 2\rangle}.$$

Using the exponential form one can easily define a rotation on an arbitrary plane without being concerned about the handedness of the space.

3 Data Alignments as Subspaces

GAs can be constructed over any type of quadratic space [17], which includes real-valued vector spaces, and also more sophisticated Hilbert spaces, such as finite Fourier basis, finite random-variable spaces, basis of orthogonal polynomials, wavelets and spherical harmonics, among others. In all cases, the concepts of blades, intersections, and combinations of subspaces are still valid, even though they may not have the same geometric meaning (see [17] for a discussion).

By assuming a MOG, one defines the quadratic space where data will be encoded and provides a practical (geometric) interpretation to blades as input data entries or resulting data alignments. For the case of real-valued vector spaces with a metric, such an interpretation may be achieved by embedding the d-dimensional **base space** $\mathbb{R}^d$ (*i.e.,* space where the geometric interpretation happens) into an n-dimensional **representational space** $\mathbb{R}^n$ (*i.e.,* the total vector space). The geometric properties of the elements in $\mathbb{R}^n$, and hence $\mathbb{R}^d$, depend on its chosen metric. See Supplementary Material A for examples of MOGs and the geometric primitives that can be represented on them.

4 The Subspace Detector

The properties of an arbitrary k-blade $\mathbf{B}_{\langle k \rangle}$ (*i.e.,* attitude, weight, and orientation) are intrinsic to the construction of the blade by the outer product of its vector factors. Therefore, they are independent of the assumed MOG and its metric. However, the attitude affects the geometric interpretation of the subspace. We use $\mathbf{C}_{\langle 3 \rangle}$ in Fig. 3 (top) to illustrate the properties of a blade. The attitude of $\mathbf{C}_{\langle 3 \rangle}$ is the stance of the circle in the surrounding space. By changing the sign of the subspace (*i.e.,* $-\mathbf{C}_{\langle 3 \rangle}$) one changes the orientation of the circle from $\mathbf{q}_1 \to \mathbf{q}_2 \to \mathbf{q}_3$ to $\mathbf{q}_3 \to \mathbf{q}_2 \to \mathbf{q}_1$. Both $\mathbf{C}_{\langle 3 \rangle}$ and $-\mathbf{C}_{\langle 3 \rangle}$ determine the same set of points defining the circumference in Fig. 3 (top) because they have the same attitude. Multiplying the subspace by a scalar value (*e.g.,* $4\,\mathbf{C}_{\langle 3 \rangle}$) produces a blade with a different weight. Again, both $\mathbf{C}_{\langle 3 \rangle}$ and $4\,\mathbf{C}_{\langle 3 \rangle}$ determine, essentially, the same circle. The later could be said to pass through its points four times faster than the former. In order to get a different circle, one has to change the attitude of $\mathbf{C}_{\langle 3 \rangle}$. Thus, **by parameterizing the attitude of subspaces we define a general parameterization that is applicable to any data alignment** that can be represented by a linear subspace (*i.e.,* a blade). In Section 4.1 we show that the attitude of a subspace $\mathbf{B}_{\langle p \rangle}$ in a n-dimensional space can be characterized by a set of $m = p\,(n - p)$ rotations applied to a canonical subspace ($\mathbf{E}_{\langle p \rangle}$) used as reference. More precisely:

$$\mathbf{B}_{\langle p \rangle} = \boldsymbol{T}\,\mathbf{E}_{\langle p \rangle}\,\widetilde{\boldsymbol{T}}. \tag{15}$$

In (15), $\boldsymbol{T}$ is the rotor encoding the sequence of m rotation operations. The computation of $\boldsymbol{T}$ is defined in (23). **The m rotation angles are the parameters characterizing the attitude of $\mathbf{B}_{\langle p \rangle}$**, and the values of p and n depend on the intended shape. For instance, by assuming the homogeneous MOG for line detection in images (Fig. 1a), $n = 3$ and $p = 2$, leading to $m = 2$.

The m rotation angles related to the sequence of rotation operations in (15) define a parameter space for p-blades. Our subspace detector uses such parameter space. The application of the proposed approach consists of three steps:

(i) Create an accumulator array as a discrete representation of the parameter space;
(ii) Perform a voting procedure where the input dataset is mapped to the accumulator array;
(iii) Search for the peaks of votes in the accumulator, as they correspond to the p-blades that best fit the input dataset.

Step (i) setups the model function for p-blades (15) and defines a parameter

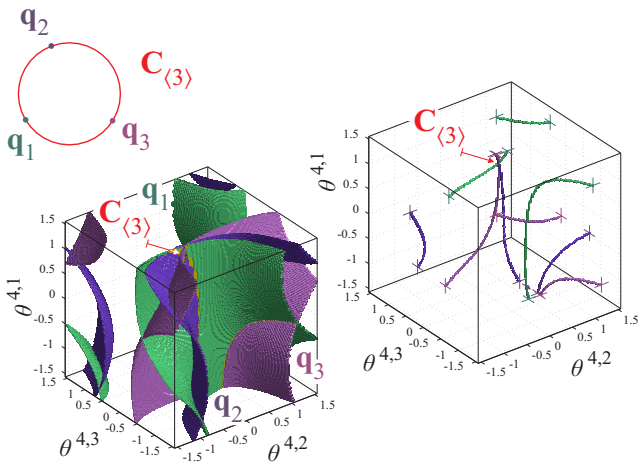


Fig. 3. Parameter spaces for simple detection case of a circle. (top) In conformal MOG, $\mathbf{C}_{\langle 3\rangle} = \mathbf{q}_1 \wedge \mathbf{q}_2 \wedge \mathbf{q}_3$ is a blade interpreted as a circle, and vectors $\mathbf{q}_1$, $\mathbf{q}_2$ and $\mathbf{q}_3$ are interpreted as points. (left) Parameter space using points from top as input. (right) Parameter space using subspaces tangent to $\mathbf{C}_{\langle 3\rangle}$ at $\mathbf{q}_i$ as input, leading to a simpler voting procedure.

space for the m degrees of freedom:

$$\mathbb{P}^m = \{(\theta^1, \theta^2, \cdots, \theta^m) \mid \theta^t \in [-\pi/2, \pi/2)\}, \tag{16}$$

where each parameter vector $(\theta^1, \theta^2, \cdots, \theta^m) \in \mathbb{P}^m$ characterizes an instance of a p-blade, and θ^t is the angle related to the t-th rotation applied to $\mathbf{E}_{\langle p\rangle}$ (15), for $1 \leq t \leq m$. In practice, we need to discretize $\mathbb{P}^m$, for which we build an accumulator array to receive "votes" and initially set its bins to zero.

Step (ii) maps the input dataset to parameter space. Essentially, our mapping procedure takes each r-blade $\mathbf{X}_{\langle r\rangle}$ in the input dataset (encoded into a MOG) and identifies the parameters (coordinates in $\mathbb{P}^m$) of all p-blades related to it. When $r \leq p$, the mapping procedure identifies in $\mathbb{P}^m$ all p-blades containing $\mathbf{X}_{\langle r\rangle}$ (*e.g.,* the lines containing an input point in Fig. 2, left). If $r \geq p$, the procedure identifies in $\mathbb{P}^m$ all p-blades contained in $\mathbf{X}_{\langle r\rangle}$ (*e.g.,* the lines on the input plane in Fig. 2, left). A detailed description of our mapping procedure is given in Section 4.2. Note that it is defined for input blades having any dimensionality (*i.e.,* $0 \leq r \leq n$). This is possible because the proposed model function (15) and its parameter space (16) are independent of the input data type.

After the voting procedure has been performed for all $\mathbf{X}_{\langle r\rangle}$, the number of votes deposited in each accumulator bin defines the importance of that bin to the input blades. Thus, the most voted bins represent the detected p-blades. The final step of our approach searches for local maxima in the accumulator array. The parameter vectors associated with such bins are used in (15) to retrieve the detected subspaces. This is achieved by applying the sequence of rotations specified by these bins to the canonical subspace $\mathbf{E}_{\langle p\rangle}$.

The algorithm described above uses the voting-based paradigm that is common to the class of techniques referred to as HTs. However, it is important to emphasize that the proposed formulation for representing intended structures (15) and the proposed mapping procedure for input entries (used in step (ii)) are different from ones presented in conventional techniques. Our definitions are conceptually different in the sense that they can be applied without changes to the detection of any primitives that can be modeled as a blade. The advantages on representing subspaces by means of the proposed parameterization instead of using the conventional parameterization scheme for p-dimensional subspaces is discussed in Section 4.1.2. Section 4.2.3 discusses the novel aspects of the proposed mapping procedure.

4.1 Parameterization of Subspaces

Let $\bigwedge \mathbb{R}^n$ be the multivector space of a metric space $\mathbb{R}^n$ with basis vectors $\{\mathbf{e}_i\}_{i=1}^n$. As pointed out in Section 2, a p-blade can be built as the outer product of p independent vectors (*i.e.,* 1-blades in $\bigwedge \mathbb{R}^n$). From the dual relationship of the outer and the regressive product (13) one can state that a p-blade can also be built as the regressive product of $n-p$ pseudovectors (*i.e.,* $(n-1)$-blades in $\bigwedge \mathbb{R}^n$). In this section, we show that the attitude of vectors and pseudovectors can be parameterized by $n-1$ parameters. The parameterization of an arbitrary p-blade is comprised by the parameters characterizing the vectors (or pseudovectors) used in its construction. The choice for one of the two constructions is based on the value of p and n, and will be discussed later in this section. Both the outer and the regressive products are independent of the metric of $\mathbb{R}^n$. Therefore, we can replace the actual metric by any convenient metric. We assume Euclidean metric for $\mathbb{R}^n$ in all computations in order to prescind the interpretation of subspaces in the actual context (*e.g.,* the geometric interpretation given by the MOG). This way, blades can be interpreted as Euclidean subspaces rather than specific structures.

An arbitrary vector $\mathbf{a}$ can be expressed in Euclidean space $\mathbb{R}^n$ by $(n-1)$ angles and a scalar value. Assuming a reference unit vector $\mathbf{e}_n$, $\mathbf{a}$ can be written as:

$$\mathbf{a} = \gamma \, \boldsymbol{S}_n \, \mathbf{e}_n \, \widetilde{\boldsymbol{S}}_n, \tag{17}$$

where

$$\boldsymbol{S}_n = \boldsymbol{R}_{n,1} \cdots \boldsymbol{R}_{n,n-2} \, \boldsymbol{R}_{n,n-1} \tag{18}$$

is a rotor encoding a sequence of rotations of $\theta^{n,j}$ radians on the unit planes $\mathbf{e}_{j+1} \wedge \mathbf{e}_j$, for

$$\boldsymbol{R}_{n,j} = \cos\left(\frac{\theta^{n,j}}{2}\right) - \sin\left(\frac{\theta^{n,j}}{2}\right) (\mathbf{e}_{j+1} \wedge \mathbf{e}_j), \tag{19}$$

and $j \in \{n-1, n-2, \cdots, 1\}$. Note that the rotors $\boldsymbol{R}_{n,j}$ in (18) are applied

to $\mathbf{e}_n$ from the right to the left. Thus, the first rotation applied is $\boldsymbol{R}_{n,n-1}$, followed by $\boldsymbol{R}_{n,n-2}$, and so on. By assuming $\theta^{n,j} \in [-\pi/2, \pi/2)$, we ensure that $\boldsymbol{S}_n \, \mathbf{e}_n \, \widetilde{\boldsymbol{S}}_n$ (17) is inside the hemisphere defined by $+\mathbf{e}_n$. Such a condition guarantees that the rotation angles encode $\mathbf{a}$'s attitude. In (17), $\gamma \in \mathbb{R}$ is the weight, and γ's sign is the orientation of $\mathbf{a}$.

The parameterization of vectors naturally extends to pseudovectors through the dual relationship between 1-dimensional and $(n-1)$-dimensional subspaces. By making the parameterized pseudovector $\mathbf{A}_{\langle n-1 \rangle} = \mathbf{a}^*$ and the reference unit blade $\mathbf{E}_{\langle n-1 \rangle} = \mathbf{e}_n^*$ ($\square^*$ is defined in (9)), (17) becomes:

$$\mathbf{A}_{\langle n-1 \rangle} = \gamma \, \boldsymbol{S}_n \, \mathbf{E}_{\langle n-1 \rangle} \, \widetilde{\boldsymbol{S}}_n. \tag{20}$$

A parameterization that is equivalent for both 1-dimensional and $(n-1)$-dimensional subspaces is convenient due the possibility to build p-blades from these subspaces while using the smallest number of parameters. For instance, when $p < (n-p)$, spanning a p-blade as the outer product of p vectors uses less parameters than spanning it as the regressive product of $(n-p)$ pseudovectors. However, when $p > (n-p)$, the best choice is to use $(n-p)$ pseudovectors. We build the reference unit subspace for p-blades as:

$$\mathbf{E}_{\langle p \rangle} = \begin{cases} \bigwedge_{v \in \mathcal{V}} \mathbf{e}_v & \text{for } p \neq q \\ \bigvee_{v \in \mathcal{V}} \mathbf{e}_v^* & \text{for } p = q \end{cases} \tag{21}$$

where $q = \max(p, n-p)$, $\bigwedge_{v \in \mathcal{V}}$ denotes the outer product of vectors $\mathbf{e}_v$, and $\bigvee_{v \in \mathcal{V}}$ is the regressive product of pseudovectors $\mathbf{e}_v^*$, for $\mathcal{V} = \{2 \, (q+i) - n\}_{i=1}^{n-q}$. As for the parameterization of vectors (17) and pseudovectors (20), the weight and orientation of an arbitrary blade $\mathbf{B}_{\langle p \rangle}$ are expressed by a scalar value (γ), while its attitude is characterized by a set of rotations ($\boldsymbol{T}$) applied to the reference blade $\mathbf{E}_{\langle p \rangle}$:

$$\mathbf{B}_{\langle p \rangle} = \gamma \, \boldsymbol{T} \, \mathbf{E}_{\langle p \rangle} \, \widetilde{\boldsymbol{T}}, \tag{22}$$

where

$$\boldsymbol{T} = \boldsymbol{S}_n \, \boldsymbol{S}_{n-2} \cdots \boldsymbol{S}_{2 \; (q+1)-n}, \tag{23}$$

and $\boldsymbol{S}_v$ are rotors computed according to (18). Each $\boldsymbol{S}_v$ is related to the parameterization of the attitude of a reference vector (or pseudovector) used in the construction of the reference blade $\mathbf{E}_{\langle p \rangle}$ (21) for $\mathbf{B}_{\langle p \rangle}$.

The interpretation of blades is not affected by γ (*i.e.,* the weight and orientation of the blade). Therefore, we can safely assume $\gamma = 1$ in (22) (leading to (15)), and define the parameterization consisting of $m = \sum_{v \in \mathcal{V}} (v-1) = p \, (n-p)$ rotation angles $\theta^{v,j}$.

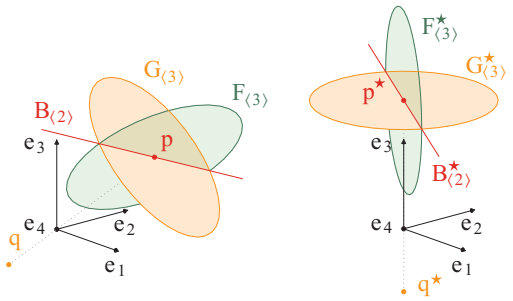


Fig. 4. Homogeneous MOG and a 3-dimensional base space: (left) $\mathbf{B}_{\langle 2\rangle}$ is the blade (interpreted as a line) resulting from the intersection of $\mathbf{F}_{\langle 3\rangle}$ and $\mathbf{G}_{\langle 3\rangle}$ (interpreted as planes). $\mathbf{G}_{\langle 3\rangle}$ defines the distance from $\mathbf{B}_{\langle 2\rangle}$ to the origin $\mathbf{e}_4$. $\mathbf{p}$ is the closest point to $\mathbf{e}_4$ for both $\mathbf{B}_{\langle 2\rangle}$ and $\mathbf{G}_{\langle 3\rangle}$. $\mathbf{F}_{\langle 3\rangle}$ includes $\mathbf{q} = \mathbf{G}^{-*}_{\langle 3\rangle}$, a vector that is orthogonal to $\mathbf{G}_{\langle 3\rangle}$ in the representational space. (right) Configuration after rolling back rotations $\boldsymbol{R}_{4,1}$ and $\boldsymbol{R}_{4,2}$ from blades $\mathbf{p}$, $\mathbf{q}$, $\mathbf{B}_{\langle 2\rangle}$, $\mathbf{F}_{\langle 3\rangle}$, and $\mathbf{G}_{\langle 3\rangle}$. The $\star$ symbol indicates that $\mathbf{A}^{\star}_{\langle k\rangle} = \widetilde{\boldsymbol{R}}_{4,2}\,\widetilde{\boldsymbol{R}}_{4,1}\,\mathbf{A}_{\langle k\rangle}\,\boldsymbol{R}_{4,1}\,\boldsymbol{R}_{4,2}$, where $\mathbf{A}_{\langle k\rangle}$ is some blade on the left.

4.1.1 Used Reference Vectors and Pseudovectors

Consecutive values for $v \in \mathcal{V}$ used in (21) and (23) are spaced two units apart. This avoids ambiguous sets of vectors/pseudovectors while defining $\mathbf{B}_{\langle p\rangle}$ (22), and leads to a more compact parameterization. The example shown in Fig. 4 illustrates this. Note that the example is described in terms of pseudovectors and, therefore, its solution immediately extends to sets of pseudovectors in spaces of any dimensionality. For this example, consider the parameterization of a straight line in 3-dimensional base space under the homogeneous MOG. In such a case, $n = (3 + 1)$ and the line $(\mathbf{B}_{\langle 2\rangle})$ is a 2-dimensional subspace embedded in the 4-dimensional representational space. The homogeneous MOG in GA is analogous to using homogeneous coordinates in projective geometry. The basis vectors are $\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_4\}$ and $\mathbf{e}_4$ is geometrically interpreted as the point at the origin.

We can write $\mathbf{B}_{\langle 2\rangle}$ in terms of the intersection (regressive product) of pseudovectors $\mathbf{F}_{\langle 3\rangle}$ and $\mathbf{G}_{\langle 3\rangle}$, geometrically interpreted as planes in Fig. 4 (left):

$$\mathbf{B}_{\langle 2\rangle} = \mathbf{F}_{\langle 3\rangle} \vee \mathbf{G}_{\langle 3\rangle}. \tag{24}$$

The choice of arbitrary pairs of planes can lead to ambiguous representations for $\mathbf{B}_{\langle 2\rangle}$. For instance, by rotating $\mathbf{F}_{\langle 3\rangle}$ and $\mathbf{G}_{\langle 3\rangle}$ around the line $\mathbf{B}_{\langle 2\rangle}$ in Fig. 4 (left), one gets different pairs of planes, and hence different parameterizations for the attitude of $\mathbf{B}_{\langle 2\rangle}$. Recall that rotations parameterizing the attitude of $\mathbf{B}_{\langle 2\rangle}$ come from the parameters of the planes (pseudovectors) defining it (see the sequence of rotors $\boldsymbol{S}_v$ in (23)). We avoid such ambiguity by choosing $\mathbf{G}_{\langle 3\rangle}$ and $\mathbf{F}_{\langle 3\rangle}$ such that they satisfy some constraints. We make $\mathbf{G}_{\langle 3\rangle}$ be the plane whose smallest distance to $\mathbf{e}_4$ is the same as the smallest distance from $\mathbf{B}_{\langle 2\rangle}$ to $\mathbf{e}_4$. Also, we choose $\mathbf{F}_{\langle 3\rangle}$ as the plane passing through $\mathbf{B}_{\langle 2\rangle}$

as well as through the point $\mathbf{q} = \mathbf{G}_{\langle 3 \rangle}^{-*}$. The undual operation (10) makes $\mathbf{q}$ be orthogonal to $\mathbf{G}_{\langle 3 \rangle}$ in the 4-dimensional representational space. Therefore, we guarantee that $\mathbf{F}_{\langle 3 \rangle}$ includes a vector factor that is orthogonal to $\mathbf{G}_{\langle 3 \rangle}$. In Fig. 4 (left), $\mathbf{p}$ is the closest point to the origin $\mathbf{e}_4$ for both $\mathbf{B}_{\langle 2 \rangle}$ and $\mathbf{G}_{\langle 3 \rangle}$. Since $\mathbf{p} \subset \mathbf{G}_{\langle 3 \rangle}$, $\mathbf{p}$ and $\mathbf{q}$ are orthogonal vectors in the representational space.

Now, let's replace $\mathbf{G}_{\langle 3 \rangle}$ in (24) by the pseudovector parameterization in (20):

$$\mathbf{B}_{\langle 2 \rangle} = \mathbf{F}_{\langle 3 \rangle} \vee \left(\beta \, \boldsymbol{S}_4 \, \mathbf{E}_{\langle 3 \rangle} \, \widetilde{\boldsymbol{S}}_4 \right). \tag{25}$$

where $\mathbf{E}_{\langle 3 \rangle} = \mathbf{e}_4^* = \mathbf{e}_4 \rfloor \mathbf{I}_{\langle 4 \rangle}^{-1} = -\mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3$ is the reference blade for $\mathbf{G}_{\langle 3 \rangle}$. Note that $\boldsymbol{S}_4$ is computed from rotations (see (18)) in such a way that any vector in the 4-dimensional space is affected by it. Therefore, one can write $\mathbf{F}_{\langle 3 \rangle} = \boldsymbol{S}_4 \, \mathbf{F}'_{\langle 3 \rangle} \, \widetilde{\boldsymbol{S}}_4$ and replace it in (25):

$$\mathbf{B}_{\langle 2 \rangle} = \left(\boldsymbol{S}_4 \, \mathbf{F}'_{\langle 3 \rangle} \, \widetilde{\boldsymbol{S}}_4 \right) \vee \left(\beta \, \boldsymbol{S}_4 \, \mathbf{E}_{\langle 3 \rangle} \, \widetilde{\boldsymbol{S}}_4 \right).$$

From the structure preservation property of rotors (14):

$$\mathbf{B}_{\langle 2 \rangle} = \beta \, \boldsymbol{S}_4 \, \left(\mathbf{F}'_{\langle 3 \rangle} \vee \mathbf{E}_{\langle 3 \rangle} \right) \, \widetilde{\boldsymbol{S}}_4. \tag{26}$$

Fig. 4 (right) shows that, rolling back $\boldsymbol{R}_{4,1}$ and $\boldsymbol{R}_{4,2}$ (the latest transformations in $\boldsymbol{S}_4$) from blades in Fig. 4 (left), one gets $\mathbf{p}^\star = (\widetilde{\boldsymbol{R}}_{4,2} \, \widetilde{\boldsymbol{R}}_{4,1}) \, \mathbf{p} \, (\boldsymbol{R}_{4,1} \, \boldsymbol{R}_{4,2})$ and $\mathbf{q}^\star = (\widetilde{\boldsymbol{R}}_{4,2} \, \widetilde{\boldsymbol{R}}_{4,1}) \, \mathbf{q} \, (\boldsymbol{R}_{4,1} \, \boldsymbol{R}_{4,2})$ in the space spanned by $\{\mathbf{e}_3, \mathbf{e}_4\}$. By also rolling back $\boldsymbol{R}_{4,3}$ (a rotation on the 2-blade $\mathbf{e}_4 \wedge \mathbf{e}_3$), all the transformations defining $\boldsymbol{S}_4$ are removed from $\mathbf{p}$ and $\mathbf{q}$. A rotation on $\mathbf{e}_4 \wedge \mathbf{e}_3$ applied to vectors in the space $\{\mathbf{e}_3, \mathbf{e}_4\}$ (such as $\mathbf{p}^\star$ and $\mathbf{q}^\star$) is interpreted as a translation along the line through $\mathbf{e}_4$ with direction $\mathbf{e}_3$. As a result, $\widetilde{\boldsymbol{R}}_{4,3} \, \mathbf{q}^\star \, \boldsymbol{R}_{4,3}$ translates $\mathbf{q}^\star$ to the origin and makes it equal to $\mathbf{e}_4$ up to a scaling factor (as one would expect from (17)). Also, $\widetilde{\boldsymbol{R}}_{4,3} \, \mathbf{p}^\star \, \boldsymbol{R}_{4,3}$ translates $\mathbf{p}^\star$ to the infinity, until it becomes $\mathbf{e}_3$ up to a scaling factor. Since $\{\mathbf{p}, \mathbf{q}\} \subset \mathbf{F}_{\langle 3 \rangle}$, we can state that $\{\mathbf{e}_3, \mathbf{e}_4\} \subset \mathbf{F}'_{\langle 3 \rangle}$ and write

$$\mathbf{F}'_{\langle 3 \rangle} = \mathbf{F}'_{\langle 1 \rangle} \wedge \mathbf{e}_3 \wedge \mathbf{e}_4, \tag{27}$$

where $\mathbf{F}'_{\langle 1 \rangle}$ is the weighted portion of $\mathbf{F}'_{\langle 3 \rangle}$ that is enclosed in the space spanned by $\{\mathbf{e}_1, \mathbf{e}_2\}$.

Using the pseudovector parameterization from (20) over $\mathbf{F}'_{\langle 1 \rangle}$, (27) becomes:

$$\mathbf{F}'_{\langle 3 \rangle} = \left(\alpha \, \boldsymbol{S}_2 \, \mathbf{E}'_{\langle 1 \rangle} \, \widetilde{\boldsymbol{S}}_2 \right) \wedge \mathbf{e}_3 \wedge \mathbf{e}_4. \tag{28}$$

$\mathbf{E}'_{\langle 1 \rangle} = \mathbf{e}_2 \rfloor \mathbf{I}_{\langle 2 \rangle}^{-1} = -\mathbf{e}_1$ is the reference blade for $\mathbf{F}'_{\langle 1 \rangle}$, a pseudovector in the $\{\mathbf{e}_1, \mathbf{e}_2\}$ space. $\mathbf{I}_{\langle 2 \rangle}$ is the pseudoscalar of such 2-D space. Replacing (28) in (26),

$$\mathbf{B}_{\langle 2 \rangle} = \beta \, \boldsymbol{S}_4 \, \left(\left(\left(\alpha \, \boldsymbol{S}_2 \, \mathbf{E}'_{\langle 1 \rangle} \, \widetilde{\boldsymbol{S}}_2 \right) \wedge \mathbf{e}_3 \wedge \mathbf{e}_4 \right) \vee \mathbf{E}_{\langle 3 \rangle} \right) \, \widetilde{\boldsymbol{S}}_4. \tag{29}$$

Note that rotation planes from $\boldsymbol{S}_2$ do not affect subspace $\mathbf{e}_3 \wedge \mathbf{e}_4$ nor $\mathbf{E}_{\langle 3 \rangle}$, because such rotation planes are orthogonal to the former and they are contained by the latter. As a result, equation (29) can be rewritten as

$$\mathbf{B}_{\langle 2 \rangle} = \gamma \left(\boldsymbol{S}_4 \, \boldsymbol{S}_2 \right) \left(\left(\mathbf{E}'_{\langle 1 \rangle} \wedge \mathbf{e}_3 \wedge \mathbf{e}_4 \right) \vee \mathbf{E}_{\langle 3 \rangle} \right) \left(\widetilde{\boldsymbol{S}}_2 \, \widetilde{\boldsymbol{S}}_4 \right). \tag{30}$$

Since $\mathbf{E}'_{\langle 1 \rangle} \wedge \mathbf{e}_3 \wedge \mathbf{e}_4 = -\mathbf{e}_1 \wedge \mathbf{e}_3 \wedge \mathbf{e}_4 = \mathbf{e}_2^*$ and $\mathbf{E}_{\langle 3 \rangle} = \mathbf{e}_4^*$ (25), (30) can be simplified to

$$\mathbf{B}_{\langle 2 \rangle} = \gamma \left(\boldsymbol{S}_4 \, \boldsymbol{S}_2 \right) \left(\mathbf{e}_2^* \vee \mathbf{e}_4^* \right) \left(\widetilde{\boldsymbol{S}}_2 \, \widetilde{\boldsymbol{S}}_4 \right). \tag{31}$$

Recall that reference pseudovectors for $\mathbf{F}_{\langle 3 \rangle}$ and $\mathbf{G}_{\langle 3 \rangle}$ are the dual representation of vectors $\mathbf{e}_2$ and $\mathbf{e}_4$, respectively (as one would expect from (21), for $\mathcal{V} = \{2, 4\}$ and $p = q = 2$), and $\gamma = \alpha \, \beta$.

In (31), $\boldsymbol{S}_2$ and $\boldsymbol{S}_4$ describe two sequences of rotations. The former consists of *one* rotation. It is similar to the case depicted in (18), but in a dimensionality lower than $n = 4$. The latter consists of three rotations, exactly like in (18). Together, the 4 rotation angles describe the attitude of $\mathbf{B}_{\langle 2 \rangle}$.

4.1.2 A Coordinate Chart for the Grassmannian

In this section we show that the proposed parameterization represents the intended p-blades with the smallest possible number of parameters.

The **Grassmannian** $G(p,n)$ is the set of all p-dimensional linear subspaces of a vector space $\mathbb{R}^n$ [42]. In GA, the representation of such subspaces resides in $\bigwedge^p \mathbb{R}^n$, *i.e.,* the portion of $\bigwedge \mathbb{R}^n$ with p-dimensional basis elements. A linear combination of basis elements in $\bigwedge^p \mathbb{R}^n$ is called a p-vector. However, an arbitrary p-vector is not necessarily a p-dimensional subspace. These are the p-vector which can be factored in terms of the outer product of p linearly independent vectors. Thus, $G(p,n)$ corresponds to a subset of p-vectors (*i.e.,* the p-blades) in $\bigwedge^p \mathbb{R}^n$.

The Grassmannian defines a projective variety of dimension $p\,(n-p)$ in the $\binom{n}{p}$-dimensional space of $\bigwedge^p \mathbb{R}^n$ [42]. Therefore, an arbitrary p-dimensional subspace requires at least $p\,(n-p)$ coordinates. By choosing a reference subspace, one may define an open affine covering $\mathbb{A}^{p\,(n-p)}$ for $G(p,n)$ [42]. The covering is open because the p-dimensional subspaces orthogonal to the reference one are not properly represented in the affine space $\mathbb{A}^{p\,(n-p)}$ as finite points (*i.e.,* they reside at infinity). The remaining p-dimensional subspaces in $G(p,n)$, on the other hand, are represented uniquely as points in $\mathbb{A}^{p\,(n-p)}$, where the reference subspace is related to the point at the origin.

The parameter space $\mathbb{P}^m$ (16) provides an alternative coordinate chart for $G(p,n)$. In such a coordinate chart, a p-dimensional subspace is addressed by

a set of $p\,(n-p)$ rotation angles in the $[-\pi/2, \pi/2)$ range (*i.e.,* the parameter vector). In contrast to the open affine covering $\mathbb{A}^{p\,(n-p)}$ of $G(p,n)$, our parameterization can represent *all* p-blades in $\bigwedge^p \mathbb{R}^n$ while using the same number of parameters defining a cyclic domain.

4.2 Mapping Procedure

Our mapping procedure is based on three key observations. The *first observation* is that the dimensionality of an arbitrary input blade $\mathbf{X}_{\langle r \rangle}$ defines a **containment relationship** between $\mathbf{X}_{\langle r \rangle}$ and $\mathbf{C}_{\langle p \rangle} \in \mathcal{C}$ (*i.e.,* $\mathbf{X}_{\langle r \rangle}$ is contained or it contains $\mathbf{C}_{\langle p \rangle}$), where $\mathcal{C}$ is the set of all p-blades related to $\mathbf{X}_{\langle r \rangle}$. Since p-blades are expressed as orthogonal transformations applied to a reference blade $\mathbf{E}_{\langle p \rangle}$ (15), we can extend such relationships through the sequence of transformations:

$$\begin{cases} \mathbf{X}^{(t)}_{\langle r \rangle} \subseteq \mathbf{C}^{(t)}_{\langle p \rangle} & \text{for } r \leq p \\ \mathbf{X}^{(t)}_{\langle r \rangle} \supseteq \mathbf{C}^{(t)}_{\langle p \rangle} & \text{for } r \geq p \end{cases} \tag{32}$$

where

$$\mathbf{X}^{(t)}_{\langle r \rangle} = \widetilde{\boldsymbol{R}}_{t+1}\, \mathbf{X}^{(t+1)}_{\langle r \rangle}\, \boldsymbol{R}_{t+1} \tag{33}$$

and

$$\mathbf{C}^{(t)}_{\langle p \rangle} = \boldsymbol{R}_t\, \mathbf{C}^{(t-1)}_{\langle p \rangle}\, \widetilde{\boldsymbol{R}}_t$$

for $\mathbf{X}^{(m+1)}_{\langle r \rangle} = \mathbf{X}_{\langle r \rangle}$, $\mathbf{C}^{(0)}_{\langle p \rangle} = \mathbf{E}_{\langle p \rangle}$, $\boldsymbol{R}_{m+1} = 1$, and $1 \leq t \leq m$. Here, $\boldsymbol{R}_t$ encodes the t-th rotation applied to $\mathbf{E}_{\langle p \rangle}$. In Section 4.1 we used a double-index notation (*i.e.,* v and j in $\boldsymbol{R}_{v,j}$) in order to emphasize that rotations $\boldsymbol{R}_{v,j}$ are related to a rotor $\boldsymbol{S}_v$, and hence to a reference vector $\mathbf{e}_v$ or pseudovector $\mathbf{e}^*_v$. In this section we have changed the notation to a single index (t) because it is more convenient for the following derivations. Thus, one can think of the model function (15) by replacing the rotor $\boldsymbol{T}$ by its component rotors $\boldsymbol{S}_v$ and, in turn, by replacing each $\boldsymbol{S}_v$ by its component rotors $\boldsymbol{R}_{v,j}$, leading to:

$$\mathbf{C}_{\langle p \rangle} = \boldsymbol{R}_m \cdots \Big(\boldsymbol{R}_2 \, \Big(\boldsymbol{R}_1\, \mathbf{E}_{\langle p \rangle}\, \widetilde{\boldsymbol{R}}_1 \Big)\, \widetilde{\boldsymbol{R}}_2 \Big) \cdots \widetilde{\boldsymbol{R}}_m. \tag{34}$$

The *second observation* is related to the rotation of basis vectors in $\mathcal{E}$ spanning $\mathbf{E}_{\langle p \rangle}$ (21):

$$\mathcal{E} = \begin{cases} \{\mathbf{e}_v\}_{v \in \mathcal{V}} & \text{for } p \neq q \\ \{\mathbf{e}_v\}_{v \in \mathcal{V}} \backslash \{\mathbf{e}_i\}_{i=1}^{n} & \text{for } p = q \end{cases}$$

where $\mathcal{A} \backslash \mathcal{B}$ denotes the relative complement or $\mathcal{A}$ in $\mathcal{B}$. **As the rotation operations are applied to vectors $\mathbf{v}_l \in \mathcal{E}$ (for $1 \leq l \leq |\mathcal{E}|$, and $|\mathcal{E}|$ denoting the cardinality of $\mathcal{E}$), the dimensionality of the regions of $\mathbb{R}^n$ that can be reached by vectors $\mathbf{v}_l$ increases**. We call these regions **spaces of possibilities**. Fig. 5 illustrates the **tree of possibilities** of reference vectors $\mathbf{v}_1 = \mathbf{e}_2$ (Fig. 5, left) and $\mathbf{v}_2 = \mathbf{e}_4$ (Fig. 5, right), for $\mathcal{E} = \{\mathbf{e}_2, \mathbf{e}_4\}$ and $n = 4$. In

Fig. 5. Trees of possibilities for $\mathbf{v}_1 = \mathbf{e}_2$ (left) and $\mathbf{v}_2 = \mathbf{e}_4$ (right) in a 4-dimensional representational space ($n = 4$ and $p = 2$).

Fig. 5, each row of the grid is related to a rotation on the plane $\mathbf{P}^{(t)}_{\langle 2 \rangle}$. The values of index t are indicated on the left side, and the rotation planes (*e.g.,* $\mathbf{e}_i \wedge \mathbf{e}_j$) are indicated on the right. The set of colored squares at each row corresponds to the region that can be reached by the reference vector after applying the rotations up to a given row (*i.e.,* the space of possibilities $\mathbf{F}^{(t)}_l$, defined in (35)). For instance, after applying the three first rotations to $\mathbf{v}_1 = \mathbf{e}_2$, it can become a vector in the space spanned by $\mathbf{F}^{(3)}_1 = \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3$.

We compute the spaces of possibilities as:

$$\mathbf{F}^{(t)}_l = \begin{cases} \mathbf{F}^{(t-1)}_l \cup \mathbf{P}^{(t)}_{\langle 2 \rangle} & \text{for } \mathit{grade}(\mathbf{F}^{(t-1)}_l \cap \mathbf{P}^{(t)}_{\langle 2 \rangle}) = 1 \\ \mathbf{F}^{(t-1)}_l & \text{otherwise} \end{cases} \tag{35}$$

where $\mathbf{F}^{(t)}_l$ is the space reachable by vector $\mathbf{v}_l \in \mathcal{E}$ after the application of the first t rotations. Therefore, $\mathbf{F}^{(0)}_l = \mathbf{v}_l$. $\mathbf{P}^{(t)}_{\langle 2 \rangle}$ is the plane where the t-th rotation happens, $\cup$ and $\cap$ denote, respectively, the join (12) and the meet (11) operations, and the *grade* function retrieves the dimensionality of a subspace. It is important to note that a parameter vector $(\theta^1, \theta^2, \cdots, \theta^m) \in \mathbb{P}^m$ defines the transformation of each $\mathbf{v}_l$ into a vector $\mathbf{c}^{(t)}_l \subseteq \mathbf{F}^{(t)}_l$, where $\mathbf{c}^{(t)}_l = \boldsymbol{R}_t \, \mathbf{c}^{(t-1)}_l \, \widetilde{\boldsymbol{R}}_t$, for $\mathbf{c}^{(0)}_l = \mathbf{v}_l$ and $\mathbf{c}^{(m)}_l = \mathbf{c}_l$.

The *third observation* is that rotations do not commute. Therefore, one needs to **respect the sequence of rotations** while computing the parameter vectors of blades in $\mathcal{C}$. Since $\mathbf{X}_{\langle r \rangle}$ and $\mathbf{E}_{\langle p \rangle}$ are the only data available to compute the elements in $\mathcal{C}$, **we calculate the parameter vectors starting from the last to the first** θ^t (*i.e.,* from θ^m to θ^1). Thus, using $\mathbf{X}^{(t)}_{\langle r \rangle}$ (33) as input, we compute the t-th rotation angle. In this case, $\mathbf{X}^{(4)}_{\langle r \rangle} = \mathbf{X}_{\langle r \rangle}$ is related to the last row of the trees of possibilities. By computing the last parameter we are able to find the rotation that takes $\mathbf{X}_{\langle r \rangle}$ into the previous row (*i.e.,* we find $\mathbf{X}^{(3)}_{\langle r \rangle}$) and so on, until we compute all the θ^t values, and finally reach $\mathbf{X}^{(0)}_{\langle r \rangle}$. $\mathbf{X}^{(0)}_{\langle r \rangle}$ is then related to the canonical reference $\mathbf{E}_{\langle p \rangle}$.

4.2.1 Mapping Procedure for $r \geq p$

The procedure for mapping an input blade $\mathbf{X}_{\langle r \rangle}$ to parameter space $\mathbb{P}^m$ is shown in Fig. 6. The algorithm assumes that $r \geq p$. The case involving $r \leq p$ is discussed in Section 4.2.2. When the t-th parameter (*i.e.,* θ^t) is computed, for $r \geq p$, the condition depicted in (32) and the second observation guarantee the existence of vectors $\mathbf{c}_l^{(t)} \subseteq (\mathbf{X}_{\langle r \rangle}^{(t)} \cap \mathbf{F}_l^{(t)})$ for all $1 \leq l \leq |\mathcal{E}|$. This holds since $\mathbf{C}_{\langle p \rangle}^{(t)}$ can be factorized as

$$\mathbf{C}_{\langle p \rangle}^{(t)} = \mathbf{c}_1^{(t)} \wedge \mathbf{c}_2^{(t)} \wedge \cdots \wedge \mathbf{c}_p^{(t)}, \tag{36}$$

where each factor $\mathbf{c}_l^{(t)}$ is related to a space of possibilities $\mathbf{F}_l^{(t)}$, and $\mathbf{X}_{\langle r \rangle}^{(t)}$ includes the entire blade $\mathbf{C}_{\langle p \rangle}^{(t)}$. In Fig. 5, it means that the transformed input blade $\mathbf{X}_{\langle r \rangle}^{(t)}$ will always share at least one vector factor with each space of possibilities at each row of the trees. At the first row of Fig. 5, $\mathbf{X}_{\langle r \rangle}^{(0)}$ must include $\mathbf{v}_1$ and $\mathbf{v}_2$.

In its first step (Fig. 6, line 1), the mapping procedure initializes a set $\mathcal{P}^{(m)}$ with a 2-tuple comprised by the input blade $\mathbf{X}_{\langle r \rangle}$ and an empty set ($\varnothing$) denoting that no parameter was calculated yet. At each iteration (lines 2 to 9) the 2-tuples in $\mathcal{P}^{(t)}$ are processed and a new set $\mathcal{P}^{(t-1)}$ is created. For each 2-tuples in $\mathcal{P}^{(t)}$ (inner loop, lines 5 to 8), the procedure $CalculateParameter$ (defined in Fig. 7) calculates the parameter θ^t for the $\mathbf{C}_{\langle p \rangle}^{(t)}$ blades related to $\mathbf{X}_{\langle r \rangle}^{(t)}$. Recall that a given input blade $\mathbf{X}_{\langle r \rangle}$ can lead to one or more parameter vectors, and hence one or more blades $\mathbf{C}_{\langle p \rangle} \in \mathcal{C}$. It depends on how many p-blades are related to $\mathbf{X}_{\langle r \rangle}$. Calculating the t-th parameter implies identifying the $\mathbf{c}_l^{(t)}$ vectors in (36), for the current t, and computing the $\boldsymbol{R}_t$ that ensures the existence of vectors $\mathbf{c}_l^{(t-1)} = \widetilde{\boldsymbol{R}}_t \, \mathbf{c}_l^{(t)} \, \boldsymbol{R}_t$ inside their respective $\mathbf{F}_l^{(t-1)}$ spaces. In other words, computing the value for θ^t consists of guaranteeing that each tree of possibilities includes at least one vector of $\mathbf{X}_{\langle r \rangle}^{(t)}$ for all t values.

When θ^t is being calculated it can assume a single value (*i.e.,* it is computed from input data) or assume all values in $[-\pi/2,\pi/2)$ (*i.e.,* it is arbitrated). Given the discrete nature of the accumulator array, to assume all values in $[-\pi/2,\pi/2)$ means replicate the current 2-tuple being processed and assign a discrete value in the $[-\pi/2,\pi/2)$ range to each one of the replicas. The possible values for θ^t define the set $\mathcal{T}$ and are computed by the $CalculateParameter$ function (line 6). Once θ^t is known, **its related rotation must be rolled back** from the input blade in order to not affect the computation of θ^{t-1} (see the sandwiching construction $\widetilde{\boldsymbol{R}}_t \mathbf{X}_{\langle r \rangle}^{(t)} \boldsymbol{R}_t$ in line 7, where $\mathbf{P}_{\langle 2 \rangle}^{(t)}$ is the rotation plane of the t-th rotation applied to $\mathbf{E}_{\langle p \rangle}$ in (34)). Also, the parameter vector must be updated (see $(\theta^t, \Theta_1^{(t)}, \cdots, \Theta_{m-t}^{(t)})$ in line 7) by including the new parameter value θ^t with the other parameters $\{\Theta_k^{(t)}\}_{k=1}^{m-t}$ computed so far. At the end of the process (line 10), $\mathbf{X}_{\langle r \rangle}^{(0)} \supseteq \mathbf{E}_{\langle p \rangle}$ for all $(\mathbf{X}_{\langle r \rangle}^{(0)}, \Theta^{(0)}) \in \mathcal{P}^{(0)}$, where

Require: An input r-blade $\mathbf{X}_{\langle r \rangle}$

1: $\mathcal{P}^{(m)} \leftarrow \left\{(\mathbf{X}_{\langle r \rangle}, \varnothing)\right\}$
2: **for** $t = m$ **down to** 1 **do**
3: Let $\mathbf{P}^{(t)}_{\langle 2 \rangle}$ be the rotation plane of the t-th rotation applied to $\mathbf{E}_{\langle p \rangle}$ in (34)
4: $\mathcal{P}^{(t-1)} \leftarrow \varnothing$
5: **for all** $(\mathbf{X}^{(t)}_{\langle r \rangle}, \Theta^{(t)}) \in \mathcal{P}^{(t)}$ **do**
6: $\mathcal{T} \leftarrow CalculateParameter(\mathbf{X}^{(t)}_{\langle r \rangle})$
7: $\mathcal{P}^{(t-1)} \leftarrow \mathcal{P}^{(t-1)} \cup \left\{(\widetilde{\boldsymbol{R}}_t \mathbf{X}^{(t)}_{\langle r \rangle} \boldsymbol{R}_t, (\theta^t, \Theta^{(t)}_1, \cdots, \Theta^{(t)}_{m-t})) \mid \theta^t \in \mathcal{T},\right.$
$\left. \text{and } \boldsymbol{R}_t = \cos\left(\frac{\theta^t}{2}\right) - \sin\left(\frac{\theta^t}{2}\right) \mathbf{P}^{(t)}_{\langle 2 \rangle}\right\}$
8: **end for**
9: **end for**
10: **return** $\left\{\Theta^{(0)} \mid (\mathbf{X}^{(0)}_{\langle r \rangle}, \Theta^{(0)}) \in \mathcal{P}^{(0)}\right\}$

Fig. 6. The algorithm used to map an input r-blade $\mathbf{X}_{\langle r \rangle}$ to $\mathbb{P}^m$ (16). The procedure returns a set of parameter vectors $\Theta^{(0)} \in \mathbb{P}^m$ characterizing the p-blades that are contained by $\mathbf{X}_{\langle r \rangle}$.

$\Theta^{(0)}$ is a parameter vector resulting from mapping $\mathbf{X}_{\langle r \rangle}$ to $\mathbb{P}^m$. All the $\Theta^{(0)}$ related to a given $\mathbf{X}_{\langle r \rangle}$ are used to tessellate a mesh of simplexes in $\mathbb{P}^m$. Voting is performed by rasterizing the mesh and incrementing the related accumulator bins by the importance ω of $\mathbf{X}_{\langle r \rangle}$.

The *CalculateParameter* function is presented in Fig. 7. It takes as input the blade $\mathbf{Y}^{(t)}$, computed in Fig. 6 as $\mathbf{X}^{(t)}_{\langle r \rangle}$. In this algorithm, the input blade is denoted by $\mathbf{Y}^{(t)}$ instead of $\mathbf{X}^{(t)}_{\langle r \rangle}$ because its dimensionality may be changed while executing the procedure.

The function in Fig. 7 is iterative (see the loop in lines 4 to 16). In line 5 it creates a set $\mathcal{M}$ containing the meet of $\mathbf{Y}^{(t)}$ with the spaces of possibilities $\mathbf{F}^{(t)}_l$. Here we are concerned with the intersections ($\mathbf{M}^{(t)}_l$) whose vector factors can be associated to well defined $\mathbf{F}^{(t)}_l$'s at current t. These intersections define the set $\mathcal{N}$ in line 6, where $\mathcal{S}$ is a set comprised by blades $\mathbf{M}^{(t)}_h$ that are not orthogonal to $\mathbf{M}^{(t)}_l$ and have the same dimensionality as $\mathbf{M}^{(t)}_l$. As a result, a 1-blade $\mathbf{M}^{(t)}_l \in \mathcal{N}$ is related (exclusively) to the l-th space of possibilities, while a 2-blade $\mathbf{M}^{(t)}_l \in \mathcal{N}$ is related to two well-defined spaces of possibilities and so on. **When $\mathcal{N}$ is empty** (line 7)**, θ^t assumes all value in** $[-\pi/2, \pi/2)$ **and the procedure stops**. Assuming all values in $[-\pi/2, \pi/2)$ means that all rotation values keep at least one vector factor of $\mathbf{X}^{(t-1)}_{\langle r \rangle} = \widetilde{\boldsymbol{R}}_t \mathbf{X}^{(t)}_{\langle r \rangle} \boldsymbol{R}_t$ (33) inside the $(t-1)$-th spaces of possibilities. Therefore, the condition depicted in (32) will be respected. However, when $\mathcal{N}$ is not empty it must be ensured that 1-blades $\mathbf{M}^{(t)}_l \in \mathcal{N}$ contained in $\mathbf{F}^{(t)}_l$ will also be contained in $\mathbf{F}^{(t-1)}_l$ after rolling back $\boldsymbol{R}_t$ from them. Thus, the set $\mathcal{O}$ (line 10) is defined as containing

Require: $\mathbf{Y}^{(t)}$, the current input blade

1: Let $\mathbf{P}^{(t)}_{\langle 2\rangle}$ be the rotation plane of the t-th rotation applied to $\mathbf{E}_{\langle p\rangle}$ in (34)
2: Let $\mathbf{F}^{(t)}_{l}$ be a space of possibilities as defined in (35)
3: Let $\mathbf{r}^{(t)}_{l} \leftarrow \mathbf{F}^{(t-1)}_{l} \rfloor \mathbf{F}^{(t)}_{l}$, *i.e.,* the vector factor in $\mathbf{F}^{(t)}_{l}$ that is not in $\mathbf{F}^{(t-1)}_{l}$
4: **loop**
5: $\quad \mathcal{M} \leftarrow \left\{(\mathbf{Y}^{(t)} \cap \mathbf{F}^{(t)}_{l}) \mid l \in \mathbb{Z}, \text{ and } 1 \leq l \leq |\mathcal{E}|\right\}$
6: $\quad \mathcal{N} \leftarrow \left\{\mathbf{M}^{(t)}_{l} \mid \mathbf{M}^{(t)}_{l} \in \mathcal{M}, \text{ and } grade(\mathbf{M}^{(t)}_{l}) = |\mathcal{S}|\,,\right.$
$\qquad\qquad\qquad\qquad\qquad\qquad \left.\text{where } \mathcal{S} \leftarrow \left\{\mathbf{M}^{(t)}_{h} \mid \mathbf{M}^{(t)}_{h} \in \mathcal{M}, \text{ and } \mathbf{M}^{(t)}_{l} * \mathbf{M}^{(t)}_{h} \neq 0\right\}\right\}$
7: $\quad$ **if** $\mathcal{N} = \varnothing$ **then**
8: $\quad\quad$ **return** $\left\{\theta^t \mid \theta^t \in [-\pi/2,\pi/2)\right\}$
9: $\quad$ **end if**
10: $\quad \mathcal{O} \leftarrow \left\{\mathbf{M}^{(t)}_{l} \mid \mathbf{M}^{(t)}_{l} \in \mathcal{N}, \text{ and } grade(\mathbf{M}^{(t)}_{l}) = 1, \text{ and } grade(\mathbf{r}^{(t)}_{l}) = 1\right\}$
11: $\quad \mathcal{Q} \leftarrow \left\{\mathbf{q}^{(t)}_{l} \mid \mathbf{q}^{(t)}_{l} = (\mathbf{m}^{(t)}_{l} \rfloor \mathbf{P}^{(t)}_{\langle 2\rangle}) \neq 0, \text{ and } \mathbf{m}^{(t)}_{l} \in \mathcal{O}\right\}$
12: $\quad$ **if** $\mathcal{Q} \neq \varnothing$ **then**
13: $\quad\quad$ **return** $\left\{\theta^t \mid \theta^t = \tan^{-1}\left(((\mathbf{q}^{(t)}_{l} \wedge \mathbf{r}^{(t)}_{l}) * \mathbf{P}^{(t)}_{\langle 2\rangle}) \,/\, (\mathbf{q}^{(t)}_{l} * \mathbf{r}^{(t)}_{l})\right),\right.$
$\qquad\qquad\qquad\qquad\qquad\qquad \left.\text{where } \mathbf{q}^{(t)}_{l} \text{ is one of the vectors in } \mathcal{Q}\right\}$
14: $\quad$ **end if**
15: $\quad \mathbf{Y}^{(t)} \leftarrow (\mathbf{M}^{(t)}_{l})^{-1} \rfloor \mathbf{Y}^{(t)}$, where $\mathbf{M}^{(t)}_{l}$ is the blade with the highest dimensionality in the set $\mathcal{N}$
16: **end loop**

Fig. 7. Function *CalculateParameter*. The algorithm takes as input a r-blade $\mathbf{Y}^{(t)}$ and determines if the t-th parameter in $\Theta^{(0)}$ (Fig. 6, line 10) can be computed from $\mathbf{Y}^{(t)}$ or if it must be arbitrated.

the vectors in $\mathcal{N}$ that can potentially "leave" $\mathbf{F}^{(t-1)}_{l}$. Notice that it may happen only when the dimensionality of $\mathbf{F}^{(t-1)}_{l}$ is smaller than the dimensionality of $\mathbf{F}^{(t)}_{l}$. The vector $\mathbf{r}^{(t)}_{l} = \mathbf{F}^{(t-1)}_{l} \rfloor \mathbf{F}^{(t)}_{l}$ in Fig. 7 (line 3) represents the additional dimension of $\mathbf{F}^{(t)}_{l}$. Thus, the left contraction ($\rfloor$) when computing $\mathbf{r}^{(t)}_{l}$ can be interpreted as removing from $\mathbf{F}^{(t)}_{l}$ the part that is not orthogonal to $\mathbf{F}^{(t-1)}_{l}$. An example of $\mathbf{r}^{(t)}_{l}$ in Fig. 5 (left) is vector $\mathbf{e}_3$ at $t = 2$ and $t = 3$.

In line 11, the set $\mathcal{Q}$ is comprised by the nonzero vectors $\mathbf{q}^{(t)}_{l}$ resulting from the contraction of vectors $\mathbf{m}^{(t)}_{l} \in \mathcal{O}$ onto the rotation plane $\mathbf{P}^{(t)}_{\langle 2\rangle}$. When $\mathbf{q}^{(t)}_{l}$ is zero, it means that $\mathbf{m}^{(t)}_{l}$ is orthogonal to $\mathbf{P}^{(t)}_{\langle 2\rangle}$ and, thus, it is not affected by a rotation in $\mathbf{P}^{(t)}_{\langle 2\rangle}$ and cannot leave $\mathbf{F}^{(t-1)}_{l}$. However, when $\mathbf{q}^{(t)}_{l}$ is not zero there is a single rotation angle θ^t that makes $\widetilde{\boldsymbol{R}}_t\, \mathbf{m}^{(t)}_{l}\, \boldsymbol{R}$ be inside $\mathbf{F}^{(t-1)}_{l}$. Such angle is computed in line 13 as the smallest angle between $\mathbf{q}^{(t)}_{l}$ and $\mathbf{r}^{(t)}_{l}$. The vector $\mathbf{q}^{(t)}_{l}$ is orthogonal to $\mathbf{m}^{(t)}_{l}$ and is contained in plane $\mathbf{P}^{(t)}_{\langle 2\rangle}$, as one would expect from the left contraction in line 11. **The rotation angle θ^t that makes $\mathbf{r}^{(t)}_{l}$ parallel to $\mathbf{q}^{(t)}_{l}$ is the only rotation angle that respects the condition**

in (32) after rolling back rotation $\boldsymbol{R}_t$ from $\mathbf{X}^{(t)}_{\langle r\rangle}$. This is because it ensures that $\widetilde{\boldsymbol{R}}_t\,\mathbf{m}^{(t)}_l\,\boldsymbol{R}_t$ will be included in $\mathbf{F}^{(t-1)}_l$. The vector $\mathbf{r}^{(t)}_l$ is included in $\mathbf{P}^{(t)}_{\langle 2\rangle}$ and $\mathbf{F}^{(t)}_l$, but it is not included in $\mathbf{F}^{(t-1)}_l$. By orthogonality, as $\mathbf{q}^{(t)}_l$ gets parallel to $\mathbf{r}^{(t)}_l$ and goes out of the tree of possibilities, $\mathbf{m}^{(t)}_l$ keeps inside of the related tree.

If the set $\mathcal{Q}$ is empty, then there is no vector $\mathbf{q}^{(t)}_l$ to be used to compute θ^t. In such a case, the input blade $\mathbf{Y}^{(t)}$ is updated by removing (see the left contraction in line 15) its highest dimensional portion that certainly will not "leave" the related spaces of possibilities after rolling back $\boldsymbol{R}_t$ for any value of θ^t.

The procedure in Fig. 7 executes p iterations in the worst case. This happens when $\mathbf{Y}^{(t)}$ is updated, at each iteration, by contracting (in line 15) only one of the factors shared with $\mathbf{C}^{(t)}_{\langle p\rangle}$.

4.2.2 Mapping Procedure for $r \leq p$

For the case involving $r \leq p$, one can explore the dual relationship between k-dimensional and $(n-k)$-dimensional subspaces in order to compute the parameter vectors of blades in $\mathcal{C}$. By taking $\mathbf{X}^{*}_{\langle r\rangle}$ as input and $\mathbf{E}^{*}_{\langle p\rangle}$ as reference blade, the containment relationship between $\mathbf{X}_{\langle r\rangle}$ and the elements in $\mathcal{C}$ changes from $\mathbf{X}_{\langle r\rangle} \subseteq \mathbf{C}_{\langle p\rangle}$ to $\mathbf{X}^{*}_{\langle r\rangle} \supseteq \mathbf{C}^{*}_{\langle p\rangle}$, reducing the mapping problem to the case described in Section 4.2.1.

Supplementary Material E presents a step-by-step example where one vector geometrically interpreted as a point under the homogeneous MOG is mapped to the parameter space defined for subspaces interpreted as straight lines in the 3-dimensional base space of the same MOG. In such a case, $r = 1$, $p = 2$, and $n = 3 + 1$.

4.2.3 On the generality of the mapping procedure

In standard HTs the input data type is known *a priori*. Thus, standard mapping procedures predefine which parameters must be arbitrated and which ones must be computed. The conventional approach to define how to treat each parameter follows the ideas presented by Ballard [24] to describe how to define a HT for analytic curves on the plane. Such an approach relies on the derivative of the chosen model function of the curve (1), which must be written with respect to the expected kind of input entry. This restricts the expected input shape and forces the specialization of the function that characterizes the intended curve. As pointed out in [24], defining the transform for analytic curves from their derivative often requires considerable algebraic

manipulation. Our approach, on the other hand, uses a general model function and does not require prior information about input data. It decides at runtime how to treat each parameter θ^t. In Section 5 we explore those features to perform concurrent detection of subspaces with different geometric interpretations in heterogeneous datasets (see Fig. 2, right) without having to worry about deriving particular mapping procedure for each type of input entry.

5 Results and Discussions

We have implemented the described algorithm using C++ and have used MATLAB® to display detection results. We have chosen to use our own GA library. However, any other library implementing the basic products of GA could be used instead (*e.g.,* Gaigen 2 [43], GluCat [44]).

From our experience, the quality of the detections performed by our approach is equivalent to the one obtained through standard HTs. It is because HTs for detecting analytic geometric shapes are particular cases of our subspace detection scheme (see Supplementary Materials B and C). Therefore, due to space restrictions, we present results illustrating the use of the proposed approach in cases that cannot be covered by a single HT designed for detecting some specific type of alignment on a given type of input data.

Fig. 1a shows the detection of subspaces geometrically interpreted as the straight lines that best fit a set of 395,248 vectors interpreted as points under homogeneous MOG. Those points were generated by supersampling (16×) each one of the 24,703 edge pixels obtained from the input image (after a Canny edge detector plus thresholding and thinning – Fig. 1b). The supersampling is used in order to treat edge pixels as area elements rather than as point elements, leading to a smoother distribution of votes in the resulting accumulator array. In this example we use homogeneous MOG, leading to $n = 3$ (the dimensionality of the representational space), $p = 2$ (the dimensionality of blades interpreted as straight lines in the MOG), and $r = 1$ (the dimensionality of input vectors interpreted as points). The discretization step for defining the accumulator array is $\pi/900$, and the importance value of each input is the magnitude of the gradient computed by the edge detector.

Fig. 8 (right) illustrates the accumulator array computed for the straight-line detection ($p = 2$) from uncertain subspaces encoding straight lines ($r = 2$) in the 2-dimensional homogeneous MOG ($n = 3$). The input 2-blades were computed from the edge pixels of the image (Fig. 8, left) and their gradient vector directions. The standard deviation for coordinates of a given edge pixel is $2/(512\sqrt{12})$, where 2 is the size of the image after normalizing its coordinates to the $[-1, +1]$ range, 512 is the number of pixels in each dimension of the

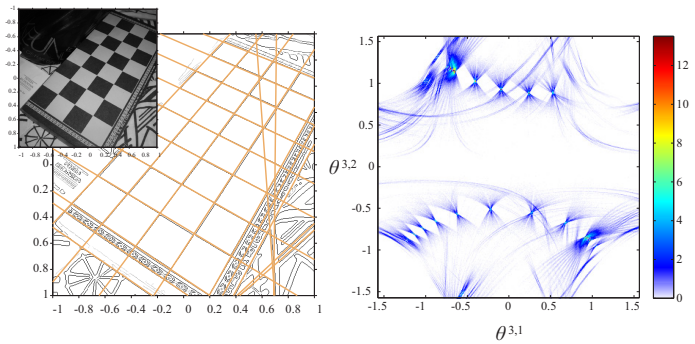


Fig. 8. The 22 most relevant detected lines obtained using our approach are presented on the left. These results were obtained from the edge information of the given image. The accumulator array on the right was obtained as the linear discretization of the parameter space $\mathbb{P}^2$, using $\pi/360$ as discretization step.

image, and 12 comes from the second central moment of a pixel with unit side. The standard deviation assumed for gradient directions was 0.13, leading to ± 0.39 radians of uncertainty on the direction normal to a given input line. The accumulator array was obtained as the linear discretization of the parameter space $\mathbb{P}^2$ ($m = 2\,(3-2) = 2$), using $\pi/360$ as discretization step. The importance value ω of each input is the magnitude of the gradient computed by the edge detector. For the voting procedure, each one of the 15,605 uncertain blades was represented by 160 random samples. Notice that Figs. 1a and 8 (left) show the detection of straight lines. However, they use different input data types. The dataset of the former is comprised by points, while the later is comprised by straight lines. In both cases the proposed approach is applied without changes.

The accumulator array for a simple case of circle detection is presented in Fig. 3 (left). For this example we represent data in the conformal MOG, leading to $n = 4$, $p = 3$, and $m = 3\,(4-3) = 3$. We use points $\mathbf{q}_1$, $\mathbf{q}_2$, and $\mathbf{q}_3$ in Fig. 3 (top) as input vectors ($r = 1$). Each input vector maps to a surface in parameter space (see Fig. 3, left). In such a mapping, the parameters $\theta^{4,1}$ and $\theta^{4,2}$ assume all values in $[-\pi/2, \pi/2)$, while $\theta^{4,3}$ is computed from $\theta^{4,1}$, $\theta^{4,2}$, and a given input vector. Note that the parameter space is defined onto a periodic domain. Therefore, surfaces resulting from mapping input data can, eventually, seem discontinuous regarding a single $[-\pi/2, \pi/2)$ period per parameter. The intersection of the three surfaces (the bins receiving three votes) defines the parameter vector related to $\mathbf{C}_{\langle 3 \rangle}$ in Fig. 3 (top). In this example, $\omega = 1$ and the step for linear discretization of $\mathbb{P}^3$ is $\pi/180$.

Fig. 1c illustrates the detection of circles that best fit a set of 1,353,760 subspaces encoding tangent directions in conformal MOG. A tangent direction is a geometric primitive encoding the subspace tangent to rounds at a given location. Therefore, tangent directions have a point-like interpretation, and also direction information assigned to them. The input tangent directions

(2-blades, leading to $r = 2$) were computed from 8,461 edge pixels (Fig. 1d) and their gradient vector directions (supersampled as 160 random samples per edge pixel to account for ± 0.35 radians of uncertainty on the gradient direction). As in Fig. 1a, ω is the magnitude of gradient directions. In order to make the irregular imaged structures become more circular, the image in Fig. 1c was convolved with a pillbox filter of radius 5 pixels before edge detection. Retrieved circles having radius bigger than 50% the image width were discarded to avoid detecting the plate. In this example, the accumulator array was defined as the linear discretization of the parameter space, using $\pi/900$ as discretization step.

The use of tangent directions (2-blades) while searching for circles allows a simpler voting procedure than when using points (1-blades without tangent information), due to the constraint imposed by the directional information. Fig. 3 (right) illustrates the result of the mapping of directions tangent to $\mathbf{C}_{\langle 3 \rangle}$ at points $\mathbf{q}_1$, $\mathbf{q}_2$, and $\mathbf{q}_3$ (Fig. 3, top) to the parameter space. By comparing the accumulator arrays in Fig. 3 (bottom), one notices that the directional information of a given tangent direction restricts the mapping of each input blade to a curve (Fig. 3, right) on the surface related to its respective point (Fig. 3, left). Such an expected behavior is a natural outcome of our mapping procedure (Section 4.2).

Our general mapping procedure allows the detection of subspaces in heterogeneous datasets. In Fig. 2 (left) we present a synthetic dataset illustrating the use of homogeneous MOG for detection of lines ($p = 2$) that best fit a heterogeneous input set comprised by 45 points ($r = 1$) and 1 plane ($r = 3$) in a 3-dimensional base space (leading to $n = 3 + 1 = 4$, and $m = 2\,(4 - 2) = 4$). In this example, we are concerned with the detection of the lines on the input plane that are also best fit for collinear input points. We set the importance of the points to $\omega = 1$ and the importance of the plane to $\omega = 45$ (the number of points). After performing the voting procedure, the bins in the accumulator array having 47 votes or more represent the lines (on the plane) defined by at least two points. Notice in Fig. 2 (left) that one subset of the points clearly defines a line, but it was not retrieved because such line is not on the plane. This example shows how our approach can be used to perform more complex coherence queries on data than standard HTs. For this example, the accumulator array was defined as the linear discretization of the parameter space, using $\pi/360$ as discretization step.

Some MOGs may represent different geometric shapes with subspaces having the same dimensionality. In conformal MOG, for instance, lines and circles are 3-dimensional subspaces, and planes and spheres are 4-dimensional subspaces. Our approach takes advantage of such a feature, allowing the concurrent detection of all shapes that have the same dimensionality on a given MOG. Fig. 2 (right) illustrates this situation, where 1 plane and 2 spheres ($p = 4$)

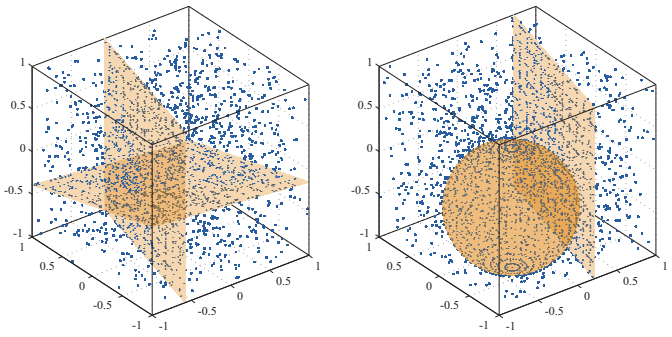


Fig. 9. Detection of data alignments from random noisy data. (left) Two subspaces interpreted as planes detected in a set of 2,855 vectors interpreted as points, from which 52.54% are outliers. (right) The concurrent detection of one plane and one sphere in a set of 3,314 input points, where 45.26% are outliers.

are detected simultaneously. In this example the heterogeneous (synthetic) dataset is comprised by 43 points ($r = 1$), 1 straight line ($r = 3$), and 3 circles ($r = 3$) in conformal MOG. The importance values of input blades was set to *one*, and the accumulator array was defined by using $\pi/360$ as discretization step.

Fig. 9 shows synthetic datasets comprised by points encoded into homogeneous ($n = 3 + 1$, left) and conformal ($n = 3 + 2$, right) MOGs. The points were randomly distributed on grids in order to improve the visualization of them as part of the data alignments to be detected. In both datasets, zero-mean Gaussian noise is added to point coordinates. In Fig. 9 (left) the dataset is comprised by 1,355 points ($r = 1$), which approximate two planes ($p = 3$), and 1,500 uniformly distributed outliers (the non-coplanar points), leading to 2,855 input entries altogether. Signal-to-noise ratio is 1.9. For the examples in Fig. 9 (left), the importance value of input blades was set to *one*, and the accumulator array was defined by using $\pi/450$ as discretization step (*i.e.,* 450 discrete angular values per axis of the 3-dimensional parameter space). Fig. 9 (right) illustrates the detection of a plane and a sphere ($p = 4$) in an input set of 3,314 points ($r = 1$), where 1,500 of such points characterize outliers with uniform distribution. Signal-to-noise ratio is 2.21. A coarse discretization step is assumed for the accumulator array ($\pi/90$). The examples in Fig. 9 show that the proposed approach can identify subspaces even in datasets having noise and outliers.

An approximation of the dth-order Voronoi diagram [14] of a set of points in $\mathbb{R}^d$ can be retrieved as byproduct of the detection of subspaces geometrically interpreted as $(d-1)$-spheres (*e.g.,* a 1-sphere is a circle, a 2-sphere is an ordinary sphere, and so on) in the conformal MOG. Fig. 10 presents an example in $\mathbb{R}^2$ (thus, $n = 2 + 2 = 4$). The points $\mathbf{p}_i$ in Fig. 10 (left) were encoded in conformal MOG and used as input for the detection of circles ($p = 3$). Each point maps to a surface in the 3-dimensional parameter space ($m = 3\,(4-3) = 3$). From the intersection of three or more surfaces one retrieves the circles pass-

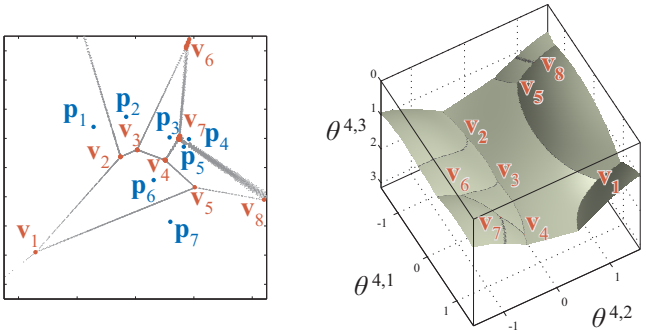


Fig. 10. (left) The vertices ($\mathbf{v}_j$) and edges (approximated by gray points) of the Voronoi diagram of points $\mathbf{p}_i$ are defined by the center of circles having no points in their interior and passing through more than two, and passing through exactly two input points, respectively. (right) These circles reside on a well defined surface at the parameter space.

ing thought three or more input points. The centers of the circles having the smaller radius correspond to the vertices of the Voronoi diagram (points $\mathbf{v}_j$ in Fig. 10, left). As Fig. 10 (right) shows, such circles reside (in parameter space) on a surface defined by the superposition of mapped input data. Thus, the vertices of the diagram can be retrieved just by looking for the bins having the largest values on that surface (*i.e.,* more than two votes). The bins having two votes correspond to the circles whose centers are at an edge of the Voronoi diagram (the gray points in Fig. 10, left). The votes accumulated by the bins below the green surface in Fig. 10 (right) are not shown for sake of clarity. In this example, the discretization step for defining the accumulator array is $\pi/720$, and $\omega = 1$. See Supplementary Material C for a detailed description of how to obtain the results presented in Fig. 10.

The Voronoi example (Fig. 10) shows that one can restrict the voting procedure to a specific region of the parameter space to restrict the detection process to a subset of geometric shapes (*e.g.,* circles with radius in a given range). Such a specialization is driven by the geometric properties of the intended shape. However, it does not affect the generality of our approach because the mapping procedure (Section 4.2) is not affected. By specifying a smaller range of interest in the parameter space, we reduce the memory and computational requirements of the technique.

We have developed proof of concept implementations of the described algorithms (Figs. 6 and 7) that keep the complete accumulator array, as well as some auxiliary data structures, allocated in the main memory. Due to practical issues, when the accumulator array has more than three dimensions, or when the discretization step is too small, the total memory required by our program can exceed the amount of memory that a process can allocate. In Windows® XP 32-bit, such a limit is 4 GB per program. By decreasing the discretization step, a higher-dimensional accumulator array may be allocated

in some acceptable amount of memory. However, the representation of the subspaces may be affected by the coarse discretization. The study of solutions for the memory-budget problem is a promising direction for future exploration.

For the results presented in this section, the discretization step of the accumulator arrays was defined according to the number of dimensions of the parameter space. In each case, the discretization step was set as the smallest value that allows the allocation of the accumulator array, while respecting the restrictions imposed by the operating system.

5.1 Time Complexity

The time complexity of our approach is $\mathcal{O}(p^2\,(m-k)\,s^k\,N)$, for $r \geq p$ and, by duality, $\mathcal{O}((n-p)^2\,(m-k)\,s^k\,N)$, for $r \leq p$, where m is the number of parameters used to characterize an instance of a p-dimensional subspace in the underlying n-dimensional space, k is the number of arbitrated parameters, s is the number of samples along one dimension of the accumulator array, and N is the number of input r-dimensional entries.

Standard HTs are usually defined assuming $r < p$ and $p = n - 1$. Under such conditions, their time complexity is $\mathcal{O}((m-k)\,s^k\,N)$. HTs are also often defined for cases where only one parameter is not arbitrated, *i.e.,* $k = m - 1$, leading to $\mathcal{O}(s^{m-1}\,N)$. It is important to notice that under the same conditions (*i.e.,* $r < p$, $p = n - 1$, and $k = m - 1$) the time complexity of the proposed approach is the same as that of standard HTs.

5.2 Limitations

A naive implementation of our approach has the same drawbacks as standard HTs regarding memory requirement and computational cost. However, as opposed to standard HTs, our approach is a closed-form solution that can be applied to all kinds of data alignments represented by linear subspaces in any complete metric spaces. As a result, any optimization developed on our framework is also generally applicable and immediately benefits the processing of all detection cases.

As one would expect, the proposed approach is limited to the detection of elements that can be represented as blades. However, it is sufficient for detecting a wide class of data alignments because GAs can be constructed over any type of quadratic spaces.

6 Summary and Future Work

We have presented a general framework for subspace detection in unordered multidimensional datasets. Our approach can be seen as a general analytical shape detector whose definition does not depend on the shape one wants to detect nor on the input data type. We have demonstrated the effectiveness of our approach as a shape detector on both real (Figs. 1 and 8) and synthetic (Figs. 2 and 9) datasets. One should note that, given its generality, our framework is not restricted to the detection of geometric shapes. It can be applied to any domain in which a problem can be cast as subspace detection. For example, the subspace clustering problem in data mining applications. Since our approach is independent of the metric space (Section 4), it can be used, without any change, for the detection of subspaces having different interpretations (*e.g.,* different MOGs), *including some that may be defined in the future.*

Mapping existing HT optimizations for the proposed approach constitutes some promising direction for future exploration. For instance, we believe that the *Statistical Hough Transform* proposed by Dahyot [45] for straight-line detection could be generalized to our subspace detector. This would overcome limitations due to the use of a discrete accumulator array, replacing it with a continuous representation of the parameter space. The use of the connectionist approach for peak detection in multidimensional parameter space proposed by Vinod *et al.* [46] is also being investigated. In some previous work [47], we have presented a HT for real-time line detection. We are currently working on the generalization of this optimization, which should benefit the detection of any pattern that can be represented as a subspace. We are also investigating the generalization of the *Probabilistic Hough Transform* proposed by Kiryati *et al.* [48], and analyzing the quality of detection achieved by using a subset of input entries chosen at random from some heterogeneous database.

The generality of our solution should enable new and exciting applications in many different areas ranging from image processing to data mining. Our approach eliminates the laborious task of defining a model and a mapping procedure for each specific case of detection. Thus, it should stimulate research on new optimization approaches for subspace detection.

Acknowledgments

This work was sponsored by CNPq-Brazil grants 142627/2007-0 and 308936/2010-8, FAPERGS PQG 10/1322-0. We thank J. A. Small and J. R. Michael, and M. Matrosovich for kindly providing the datasets used in Fig. 1, and the anonymous reviewers for their comments and insightful suggestions.

References

[1] R. Mankel, Pattern recognition and event reconstruction in particle physics experiments, Rep. Prog. Phys. 67 (4) (2004) 553–622.

[2] M. Fechner *et al.*, Kinematic reconstruction of atmospheric neutrino events in a large water Cherenkov detector with proton identification, Phys. Rev. D 79 (11) (2009) 112010.

[3] B. Krishnan, A. M. Sintes, M. A. Papa, B. F. Schutz, S. Frasca, C. Palomba, Hough transform search for continuous gravitational waves, Phys. Rev. D 70 (8) (2004) 082001.

[4] B. Abbott *et al.*, All-sky search for periodic gravitational waves in LIGO S4 data, Phys. Rev. D 77 (2) (2008) 022001.

[5] J. M. Bewes, N. Suchowerska, D. R. McKenzie, Automated cell colony counting and analysis using the circular Hough image transform algorithm (CHiTA), Phys. Med. Biol. 53 (21) (2008) 5991–6008.

[6] J. Kürner, A. S. Frangakis, W. Baumeister, Cryo-electron tomography reveals the cytoskeletal structure of Spiroplasma Melliferum, Science 307 (5708) (2005) 436–438.

[7] D. Naumović, P. Aebi, L. Schlapbach, C. Beeli, K. Kunze, T. A. Lograsso, D. W. Delaney, Formation of a stable decagonal quasicrystalline Al-Pd-Mn surface layer, Phys. Rev. Lett. 87 (19) (2001) 195506.

[8] T. Liu, D. Raabe, S. Zaefferer, A 3D tomographic EBSD analysis of a CVD diamond thin film, Sci. Technol. Adv. Mater. 9 (3) (2008) 035013.

[9] M. Ding, A. Fenster, A real-time biopsy needle segmentation technique using Hough transform, Med. Phys. 30 (8) (2003) 2222–2233.

[10] F. Oloumi, R. M. Rangayyan, Detection of the temporal arcade in fundus images of the retina using the Hough transform, in: Proc. of IEEE Eng. Med. Biol. Soc., 2009, pp. 3585–3588.

[11] W. Xiangyu, S. Ramanathan, M. Kankanhalli, A robust framework for aligning lecture slides with video, in: Proc. of IEEE Int. Conf. Image Process., 2009, pp. 249–252.

[12] O. Barinova, V. Lempitsky, P. Kohli, On detection of multiple object instances using Hough transforms, in: Proc. of IEEE Conf. Comput. Vis. Pattern Recognit., 2010, pp. 2233–2240.

[13] J. Koh, V. Govindaraju, V. Chaudhary, A robust iris llocalization method using an active contour model and Hough transform, in: Proc. of IEEE Int. Conf. Pattern Recognit., 2010, pp. 2852–2856.

[14] G. Voronoi, Nouvelles applications des paramètres continus à la théorie des formes quadratiques, J. Reine Angew. Math. (Crelle's J.) 1908 (134) (1908) 198–287.

[15] L. A. F. Fernandes, M. M. Oliveira, Geometric algebra: a powerful tool for solving geometric problems in visual computing, in: Tutorials of Brazilian Symp. Comput. Graph. Image Process., 2009, pp. 17–30.

[16] L. Dorst, D. Fontijine, S. Mann, Geometric algebra for computer science: an object oriented approach to geometry, Morgan Kaufmann, 2007.

[17] C. Perwass, Geometric algebra with applications in engineering, Springer, 2009.

[18] J. Illingworth, J. Kittler, A survey of the Hough transform, Comput. Vis. Graph. Image Process. 44 (1) (1988) 87–116.

[19] V. F. Leavers, Which Hough transform?, CVGIP: Image Underst. 58 (2) (1993) 250–264.

[20] R. O. Duda, P. E. Hart, Use of the Hough transformation to detect lines and curves in pictures, Commun. ACM 15 (1) (1972) 11–15.

[21] C. Kimme, D. Ballard, J. Sklansky, Finding circles by an array of accumulators, Commun. ACM 18 (2) (1975) 120–122.

[22] P. V. C. Hough, Machine analysis of bubble chamber pictures, in: Proc. of Int. Conf. on High Energy Accelerators and Instrum., CERN, 1959.

[23] N. Bennett, R. Burridge, N. Saito, A method to detect and characterize ellipses using the Hough transform, IEEE Trans. Pattern Anal. Machine Intell. 21 (7) (1999) 652–657.

[24] D. H. Ballard, Generalizing the Hough transform to detect arbitrary shapes, Pattern Recognit. 13 (2) (1981) 111–122.

[25] H. L. Wang, A. P. Reeves, Three-dimensional generalized Hough transform for object identification, in: Proc. of SPIE, Vol. 1192, 1990, pp. 363–374.

[26] F. O'Gorman, M. B. Clowes, Finding picture edges through collinearity of feature points, in: Proc. of Joint Conf. on Artif. Intell., 1973, pp. 543–555.

[27] G. Medioni, M.-S. Lee, C.-K. Tang, A computational framework for segmentation and grouping, Elsevier, Amsterdam, 2000.

[28] R. Vidal, Subspace clustering, IEEE Signal Process. Mag. 28 (2) (2011) 52–68.

[29] M. E. Tipping, C. M. Bishop, Mixtures of probabilistic principal component analyzers, Neural Comput. 11 (2) (1999) 443–482.

[30] R. Vidal, Y. Ma, S. Sastry, Generalized principal component analysis (GPCA), IEEE Trans. Pattern Anal. Mach. Intell. 27 (12) (2005) 1945–1959.

[31] E. Elhamifar, R. Vidal, Sparse subspace clustering, in: Proc. of IEEE Conf. Comput. Vis. Pattern Recognit., 2009, pp. 2790–2797.

[32] X. Chen, Y. Ye, X. Xu, J. Z. Huang, A feature group weighting method for subspace clustering of high-dimensional data, Pattern Recognit.(to appear).

[33] M. Polito, P. Perona, Grouping and dimensionality reduction by locally linear embedding, in: Proc. of Advances in Neural Inform. Process. Syst., 2001, pp. 1255–1262.

[34] R. Souvenir, R. Pless, Manifold clustering, in: Proc of IEEE Int. Conf. Comput. Vis., 2005, pp. 648–653.

[35] A. Goh, R. Vidal, Segmenting motions of different types by unsupervised manifold clustering, in: Proc. of IEEE Conf. Comput. Vis. Pattern Recognit., 2007, pp. 1–6.

[36] A. Goh, R. Vidal, Clustering and dimensionality reduction on Riemannian manifolds, in: Proc. of IEEE Conf. Comput. Vis. Pattern Recognit., 2008, pp. 1–7.

[37] P. Favaro, R. Vidal, A. Ravichandran, A closed form solution to robust subspace estimation and clustering, in: Proc. of IEEE Conf. Comput. Vis. Pattern Recognit., 2011, pp. 1801–1807.

[38] D. Hestenes, New foundations for classical mechanics, Reidel Publishing Company, 1987.

[39] C. Doran, J. Lasenby, L. Dorst, D. Hestenes, S. Mann, A. Naeve, A. Rockwood, Geometric algebra: new foundations, new insights, Course 31 at SIGGRAPH (2000).

[40] C. Doran, J. Lasenby, L. Dorst, D. Hestenes, S. Mann, A. Naeve, A. Rockwood, Geometric algebra, Course 53 at SIGGRAPH (2001).

[41] D. Hildenbrand, D. Fontijne, C. Perwass, L. Dorst, Geometric algebra and its application to computer graphics, Tutorial 3 at Eurographics (2004).

[42] J. Harris, Algebraic geometry: a first course, Springer, 1992.

[43] D. Fontijne, Gaigen 2: a geometric algebra implementation generator, in: Proc. of Int. Conf. Generative Progr. Component Eng., ACM Press, 2006, pp. 141–150.

[44] P. C. Leopardi, GluCat (2009).
URL `http://glucat.sourceforge.net/`

[45] R. Dahyot, Statistical Hough transform, IEEE Trans. Pattern Anal. Mach. Intell. 31 (8) (2009) 1502–1509.

[46] V. V. Vinod, S. Chaudhury, S. Ghose, J. Mukherjee, A connectionist approach for peak detection in Hough space, Pattern Recognit. 25 (10) (1992) 1253–1264.

[47] L. A. F. Fernandes, M. M. Oliveira, Real-time line detection through an improved Hough transform voting scheme, Pattern Recognit. 41 (1) (2008) 299–314.

[48] N. Kiryati, Y. Eldar, A. M. Bruckstein, A probabilistic Hough transform, Pattern Recognit. 24 (4) (1991) 303–316.