Handling Uncertain Data in Subspace Detection

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

Experimental data is subject to uncertainty as every measurement apparatus is inaccurate at some level. However, the design of most computer vision and pattern recognition techniques (*e.g.,* Hough transform) overlook this fact and treat intensities, locations and directions as precise values. In order to take imprecisions into account, entries are often resampled to create input datasets where the uncertainty of each original entry is characterized by as many exact elements as necessary. Clear disadvantages of the sampling-based approach are the natural processing penalty imposed by a larger dataset and the difficulty of estimating the minimum number of required samples. We present an improved voting scheme for the General Framework for Subspace Detection (hence to its particular case: the Hough transform) that allows processing both exact and uncertain data. Our approach is based on an analytical derivation of the propagation of Gaussian uncertainty from the input data into the distribution of votes in an auxiliary parameter space. In this parameter space, the uncertainty is also described by Gaussian distributions. In turn, the votes are mapped to the actual parameter space as non-Gaussian distributions. Our results show that resulting accumulators have smoother distributions of votes and are in accordance with the ones obtained using the conventional sampling process, thus safely replacing them with significant performance gains.

Key words: Hough transform, uncertain data, subspace detection, shape detection, Grassmannian, geometric algebra, parameter space

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

Voting-based techniques for fitting instances of a model to experimental data are widely used in computer vision, pattern recognition and image processing. The most popular of these approaches is undoubtedly the Hough transform (HT). It was first designed for detecting the straight lines that best fit points on the plane [1,2]. Later, the HT was specialized for detecting other analytical shapes, such as circles, parabolas and ellipses, and also generalized for detecting arbitrary non-analytical shapes in images [3,4]. More recently, a generalization of the HT concept has been presented to detect any data alignment that can be represented analytically and characterized as a linear subspace embedded in spaces with arbitrary dimensionality [5].

The HT works by mapping each primitive from the input dataset to points (in a parameter space) representing the shapes potentially passing through that primitive. Thus, shape detection is converted into the problem of identifying peaks in an accumulator array representing the discretized parameter space. The strong aspects of the HT are its robustness to noise, clutter, and missing data. However, improper discretization of the parameter space may lead to unsharp or multiple peaks of votes [6,7]. In particular, if the discretization is too fine, votes are likely to fall in neighboring bins, thus reducing the visibility of the main peaks. This problem gets aggravated when experimental data, which often contains errors due to imprecisions in the measuring instruments, are treated as exact entries. For instance, raster images captured by digital cameras are resolution dependent. Likewise, computed tomography captures images of regularly-spaced slices of a subject and hence represents inter-slice data by some approximation. Another source of uncertainty is the error associated with the estimation of the real camera parameters, which are usually obtained using image-based calibration procedures. Although uncertainty is intrinsic to experimental data, this fact has been neglected by most researchers during the development of shape and subspace detectors due to the difficulty of handling it. If the HT voting procedure does not take into account the uncertainty present in the input data, the resulting accumulator array is likely to contain spurious peaks of votes, making the identification of the actual peaks more difficult. The quality of the detection procedures can be improved by considering the role of uncertainty in the detection mechanisms.

In this paper we extend the mapping and voting procedures presented in [5] to handle input containing Gaussian distributed uncertainty. The extended mapping procedure is based on first-order error propagation analysis [8]. It propagates the uncertainty of each input element throughout the computations into an auxiliary parameter space where the uncertainty is described by a multivariate Gaussian distribution (Fig. 1b). In turn, such distribution is mapped to the actual parameter space by the extended voting procedure.

This lends to warped (non-Gaussian) distributions of votes in the accumulator array (Fig. 1a) resulting in accurate distribution of votes over the bins.

In order to handler uncertain data, existing voting-based techniques need to be applied to datasets that replace the actual uncertain input entries by as many samples as needed to characterize the distribution of uncertainty of each original entry. It is clear that such an approach imposes a processing penalty as the number of samples increase. Also, the smoothness of the resulting accumulator array may be affected even when a large number of samples is generated (Fig. 5f). Our approach, on the other hand, treats each original uncertain input entry directly and propagates its uncertainty throughout the computation chain, producing smoother distributions of votes (Fig. 5c).

Our technique can be applied without changes and using a single implementation to the detection of all kinds of data alignments representable as linear subspaces in any complete metric spaces. When the subspaces are interpreted as shapes, our formulation becomes a general analytical shape detector that outperforms conventional HTs by being able to concurrently detect multiple kinds of shapes, in datasets containing multiple types of data.

The central contribution of this paper is a more flexible voting scheme for the detection framework presented in [5]. It can handle both exact data as well as data containing Gaussian-distributed uncertainties. In addition, we define an auxiliary space where the uncertainty of p-dimensional subspaces residing in some n-dimensional space is related to the uncertainty of r-dimensional random subspace under first-order error propagation, for $0 \leq r \leq n$. This interesting property may lead to new insights on how to extend other subspace-clustering techniques to include error propagation into their formulations.

We formulate our subspace detector using geometric algebra [9–12], tensors [13], and error theory [8]. In order to validate it, we performed three set of experiments that verify that our uncertain-based voting scheme generates the expected voting distributions. As a result, our technique reduces the scattering of votes in the accumulator, lending to superior detection results.

2 Related Work

Linear subspace learning (LSL) techniques aim to find linear projections, of sets of training samples, satisfying some optimality criteria [14]. They map input data from a high-dimensional space to a lower-dimensional one where space partitioning and clustering can be performed more conveniently. Principal component analysis (PCA), linear discriminant analysis (LDA), general averaged divergence analysis (GADA) [15,16], and locality preserving projec-

tions (LPP) [17] are examples of LSL algorithms. Isomaps [18], locally linear embedding (LLE) [19], and their variations, such as the local linear transformation embedding (LLTE) [20], extend the dimensionality-reduction problem to dataset entries that lie in nonlinear-manifold structures [21]. Recently, Zhou and Tao formulated the double shrinking model (DSM) [22], which aims to compress data by simultaneously shrinking both dimensionality and cardinality. All those techniques use points or vectors in $\mathbb{R}^n$ as input. Multilinear subspace learning (MSL) techniques are higher-order generalizations of LSL [23]. They use nth-order tensorial samples as input data, thus enhancing their representation power. When used for clustering, the goal of all dimensionality-reduction techniques is to simplify the identification of nearby entries.

In contrast to dimensionality reduction techniques, our approach aims to find the p-dimensional subspaces that accommodate as many database objects as possible. Thus, it can be seen as a subspace-clustering approach (see [24] for a review of such techniques). A key difference is that, with the exception of [5], all existing subspace-clustering techniques are tailored for specific applications or input data types. Günnemann *et al.* [25] has recently proposed the first subspace clustering approach for uncertain data. Their technique is designed to detect linear subspaces aligned to the axis of the space where input uncertain points reside. Our approach, on the other hand, inherits from [5] the ability to systematically adapt itself to handle r-dimensional input subspaces ($0 \leq r \leq n$). In contrast to our previous approach [5], the proposed technique was designed to handle input entries containing Gaussian distributed uncertainty.

As pointed out in [5], our voting-based solution is a generalization of the HT. The literature covering applications and variations of HT-based techniques is vast [3,4]. The following section focus on techniques that, like ours, treat each input entry as some distribution during the voting procedure.

2.1 Uniform Distribution of Uncertainty

O'Gorman and Clowes [26] set a uniform distribution of uncertainty on the angular parameter of the normal equation of the line in order to limit the amount of bins that are incremented for each input pixel during the voting procedure. By doing so, each input point votes for a subset of the sinusoidal line resulting from its mapping to the parameter space. Kimme *et al.* [27] use a similar idea to reduce the amount of votes spread in the detection of circles.

Cyganski *et al.* [28] use unit-area squares corresponding to feature pixels instead of lattice coordinates. They show that mapping square pixels to the parameter space of the slope-intercept form of the line results in a convex region that can be efficiently stored in a new data structure proposed in [29].

In contrast to these techniques, our formulation assumes Gaussian (as opposed to uniformly) distributed uncertainty in the input. Moreover, it is not tailored to a single kind of input and, therefore, is a more general and flexible solution.

2.2 Gradient Weighted Hough Transform

Inspired by the optimization proposed in [26], van Veen and Groen [6] designed a weighting function to map an input pixel to the accumulator array taking into account the gradient of the pixel to limit the number of bins receiving votes. With their function, a different weight is given to each vote according to a Gaussian distribution centered on the angular parameter computed from the gradient angle of the point on a reference line with known normal direction.

Our approach also uses a weighting function for distributing votes in the accumulator array. For this, we use first-order error propagation to compute the appropriate vote distribution to each mapped entry.

2.3 Kernel-Based Hough Transform

The use of first-order error propagation in HT was introduced by Fernandes and Oliveira in the context of *Kernel-Based HT* (KHT) [30]. The approach operates on clusters of approximately collinear edge pixels. For each cluster, votes are cast and weighted using an oriented elliptical-Gaussian kernel that models the uncertainty of the best-fitting line with respect to the cluster. This lends to a much cleaner voting map that is robust to the detection of spurious lines, and allows a software implementation to perform in real time.

The pipeline presented in [30] is specifically designed to detect straight lines from image pixels. The present work, in contrast, extends the framework described in [5] to handle uncertainty in the detection of arbitrary shapes. It is independent of the geometric properties of the data alignments to be detected, and handles heterogeneous datasets comprised by uncertain input entries, and with possibly different dimensionalities and geometrical interpretations.

It is important to note that the approach introduced in this paper is not intended to perform in real-time like the KHT. The performance of the KHT results from an efficient clustering procedure and a culling strategy that avoids voting for kernels that are less likely to result in peak formation. Their generalization to our technique is a promising direction for future exploration.

3 Mathematical Background

In this section we briefly introduce the mathematical background used in the paper. A more detailed introduction to these topics can be found in the provided references. Supplementary Material A summarizes the notational convention.

3.1 Geometric Algebra

The Clifford algebra of a vector space over a field of real numbers endowed with a quadratic form is called a geometric algebra (GA). It is an alternative to conventional vector algebra. In additional do scalars and vectors, GA defines a new structure called k**-blade**, which resents a k-dimensional subspace. The integer value $0 \leq k \leq n$ is said the **grade** of the blade, and n is the dimensionality of the overall vector space $\mathbb{R}^n$. Thus, 0-blades and 1-blades correspond, respectively, to scalar values and vectors. A 2-blade embedded in some Euclidean vector space can be interpreted as a plane that passes through the origin, and a n-blade represents the whole space.

GA inherits from Clifford algebra its fundamental product, *i.e.,* the Clifford or geometric product (2), which is strongly motivated by geometry and can be taken between any two objects. From the geometric product of two blades one can build a new blade that represents their common space (*e.g.,* the plane that is common to two orthogonal vectors), or a blade that represents the complement of a subspace that is contained inside another (*e.g.,* the vector that is orthogonal to a plane inside a volume), or even a structure that encodes orthogonal transformations, *i.e.,* a **versor**.

The set that is closed under finite addition and geometric product multiplication is called a **multivector space** ($\bigwedge \mathbb{R}^n$). Blades and versors are numbers in $\bigwedge \mathbb{R}^n$. One may build a basis for $\bigwedge \mathbb{R}^n$ by taking the k-combinations of vectors from the set of basis vectors $\{\mathbf{e}_i\}_{i=1}^n$ of $\mathbb{R}^n$. Altogether, such a basis has 2^n basis elements (*i.e.,* $\sum_{k=0}^{n} \binom{n}{k} = 2^n$). As an example, the basis of $\bigwedge \mathbb{R}^3$ is

$$\left\{ 1, \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, \mathbf{e}_1 \wedge \mathbf{e}_2, \mathbf{e}_1 \wedge \mathbf{e}_3, \mathbf{e}_2 \wedge \mathbf{e}_3, \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3 \right\}. \tag{1}$$

The symbol $\wedge$ denotes the outer product, which will be defined latter in (3) as a special case of the geometric product. For orthogonal basis vectors $\mathbf{e}_i$ and $\mathbf{e}_j$, $\mathbf{e}_i \mathbf{e}_j$ is equal to $\mathbf{e}_i \wedge \mathbf{e}_j$ for all $i \neq j$ and *zero* for $i = j$.

A linear combination of the basis elements of $\bigwedge \mathbb{R}^n$ is called a **multivector**. For the basis (1) of $\bigwedge \mathbb{R}^3$, the multivector structure can be written as

$$M = \mu^1\, 1 + \mu^2\, \mathbf{e}_1 + \mu^3\, \mathbf{e}_2 + \mu^4\, \mathbf{e}_3 + \mu^5\, \mathbf{e}_1 \wedge \mathbf{e}_2 + \mu^6\, \mathbf{e}_1 \wedge \mathbf{e}_3 + \mu^7\, \mathbf{e}_2 \wedge \mathbf{e}_3 + \mu^8\, \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3,$$

where $\mu^i \in \mathbb{R}$ is the i-th coefficient of M. Blades and versors can be encoded as multivectors. Once the multivector space is defined, the geometric product between two numbers in $\bigwedge \mathbb{R}^n$ is completely described by the rules

$$\mathbf{e}_i\, \mathbf{e}_i = \mathsf{Q}\left(\mathbf{e}_i, \mathbf{e}_i\right), \tag{2a}$$
$$\mathbf{e}_i\, \mathbf{e}_j = -\mathbf{e}_j\, \mathbf{e}_i, \tag{2b}$$
$$\mathbf{e}_i\, \alpha = \alpha\, \mathbf{e}_i, \tag{2c}$$
$$(A\, B)\; C = A\; (B\; C) = A\, B\, C, \tag{2d}$$
$$(A + B)\; C = A\, C + B\, C \text{ and } C\; (A + B) = C\, A + C\, B, \tag{2e}$$

where $\alpha \in \mathbb{R}$ is a scalar value, A, B and $C \in \bigwedge \mathbb{R}^n$ are general multivectors and Q is the quadratic form (*i.e.,* metric) of the space. Without loss of generality, since one can always find an orthonormal basis $\{\mathbf{e}_i\}_{i=1}^n$ for $\mathbb{R}^n$, the rules in (2) assume that $\mathsf{Q}\left(\mathbf{e}_i, \mathbf{e}_j\right) = 0$ for all $i \neq j$ and $\mathsf{Q}\left(\mathbf{e}_i, \mathbf{e}_i\right) \in \{-1, 0, 1\}$ [31].

Many bilinear products in GA are special cases of the geometric product. In this paper we are concerned with three of them: the **outer product** (3), the **scalar product** (4), and the **left contraction** (5).

$$\mathbf{A}_{\langle r \rangle} \wedge \mathbf{B}_{\langle s \rangle} = \left\langle \mathbf{A}_{\langle r \rangle}\, \mathbf{B}_{\langle s \rangle} \right\rangle_{r+s} \tag{3}$$
$$\mathbf{A}_{\langle r \rangle} * \mathbf{B}_{\langle s \rangle} = \left\langle \mathbf{A}_{\langle r \rangle}\, \mathbf{B}_{\langle s \rangle} \right\rangle_{0} \tag{4}$$
$$\mathbf{A}_{\langle r \rangle} \rfloor \mathbf{B}_{\langle s \rangle} = \left\langle \mathbf{A}_{\langle r \rangle}\, \mathbf{B}_{\langle s \rangle} \right\rangle_{s-r} \tag{5}$$

In (3)–(5), $\mathbf{A}_{\langle r \rangle}$ and $\mathbf{B}_{\langle s \rangle}$ denote a r- and a s-blade, respectively. $\langle \square \rangle_t$ denotes the **take grade operation**. It retrieves the t-dimensional part of the geometric product of $\mathbf{A}_{\langle r \rangle}$ and $\mathbf{B}_{\langle s \rangle}$ (*i.e.,* the portion of the multivector resulting from $\mathbf{A}_{\langle r \rangle}\, \mathbf{B}_{\langle s \rangle}$ that is comprised of basis blades having dimensionality t). The definitions above can be extended to general multivector.

The outer product (3) is used to span a $(r + s)$-dimensional subspace from blades of dimensionality r and s. If there is at least one common (concurrent) vector factor in the multiplied terms the outcome is *zero*. In case of independent vector factors, the resulting $(r + s)$-blade is spanned by the r vectors spanning the first and by the s vectors spanning the second multiplied blades. Thus, a k-blade can be define as the outer product of k independent 1-blade.

The scalar product (4) generalizes the vector inner product to arbitrary subspaces with same dimensionality (*i.e.,* $r = s$). If $r \neq s$, the outcome is *zero*. The scalar product can be used to compute the **squared norm** of a blade

$$\left\| \mathbf{A}_{\langle k \rangle} \right\|^2 = \mathbf{A}_{\langle k \rangle} * \tilde{\mathbf{A}}_{\langle k \rangle}, \tag{6}$$

where

$$\widetilde{\mathbf{A}}_{\langle k \rangle} = (-1)^{k\,(k-1)/2}\,\mathbf{A}_{\langle k \rangle} \tag{7}$$

is the **reverse** operation. A blade is invertible if it has a nonzero norm. The **inverse** $\mathbf{A}_{\langle k \rangle}^{-1}$ of a blade $\mathbf{A}_{\langle k \rangle}$ satisfies $\mathbf{A}_{\langle k \rangle}\,\mathbf{A}_{\langle k \rangle}^{-1} = \mathbf{A}_{\langle k \rangle}^{-1}\,\mathbf{A}_{\langle k \rangle} = 1$. It is computed as

$$\mathbf{A}_{\langle k \rangle}^{-1} = \widetilde{\mathbf{A}}_{\langle k \rangle}\,\left\|\mathbf{A}_{\langle k \rangle}\right\|^{-2}. \tag{8}$$

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

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

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, and $\square^{-1}$ is given by (8). 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 (or primal) representation (*i.e.,* $(\mathbf{A}_{\langle k \rangle}^{*})^{-*} = \mathbf{A}_{\langle k \rangle}$).

From a computational point of view, this duality allows geometric algorithms to do the work of two (*e.g.,* by computing the Voronoi diagram one also gets the Delaunay tesselation). In Sections 4 and 5, we use the dual (9) and undual (10) operations of GA to define a closed-form solution for subspace detection involving input and output blades of arbitrary dimensionalities. Also, the dual and undual operations are useful for defining the **regressive product**

$$\mathbf{A}_{\langle r \rangle} \vee \mathbf{B}_{\langle s \rangle} = \left(\mathbf{B}_{\langle s \rangle}^{*} \wedge \mathbf{A}_{\langle r \rangle}^{*}\right)^{-*}, \tag{11}$$

which can be regarded as the dual operation to the outer product in some Euclidean space. When the disjoint and common parts of $\mathbf{A}_{\langle r \rangle}$ and $\mathbf{B}_{\langle s \rangle}$ span the whole space, the outcome of the regressive product is the subspace that is shared by $\mathbf{A}_{\langle r \rangle}$ and $\mathbf{B}_{\langle s \rangle}$. Otherwise, the resulting value is *zero*.

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 $\mathbb{R}^n$. The transformation encoded by a rotor $\boldsymbol{R}$ is applied to a k-blade $\mathbf{A}_{\langle k \rangle}$ using a sandwiching construction involving the geometric product $\mathbf{A}''_{\langle k \rangle} = \boldsymbol{R}\,\mathbf{A}_{\langle k \rangle}\,\widetilde{\boldsymbol{R}}$, where $\mathbf{A}''_{\langle k \rangle}$ is the transformed subspace, and $\widetilde{\boldsymbol{R}}$ denotes the reverse of $\boldsymbol{R}$. The inverse of a rotor equals its reverse, thus $\mathbf{A}_{\langle k \rangle} = \widetilde{\boldsymbol{R}}\,\mathbf{A}''_{\langle k \rangle}\,\boldsymbol{R}$.

A more practical way to define rotors is by using the exponential of 2-blades that act as rotation planes. 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

$$\boldsymbol{R} = \exp\left(-\theta/2\,\mathbf{P}_{\langle 2 \rangle}\right) = \cos\left(\theta/2\right) - \sin\left(\theta/2\right)\,\mathbf{P}_{\langle 2 \rangle}. \tag{12}$$

Using (12) one can define a rotation on an arbitrary plane without worrying about the space handedness. This exponential form is particularly useful in this work because the parameterization of blades in the general framework for subspace detection (Section 4) is based on the rotation of a canonical subspace.

The books by Dorst *et al.* [11], Perwass [10], Sommer [12] and Hestenes [31,32] provide in-depth treatments to GA. The tutorials by Fernandes and Oliveira [9] and Doran *et al.* [33,34] present a detailed introduction to the subject.

3.2 First-Order Error Propagation

Any computation propagates the error associated with its input to its output [8]. Since the exact error in the input is often unknown, the error in the computed values can only be estimated with some probability. An experimental uncertain value can be expressed by a random variable, where the uncertainty is described by some probability distribution function (PDF). We assume that the uncertainty is always Gaussian, so the PDF is fully determined by its expectation and covariance matrix. The distribution used for modeling uncertainty determines the propagation model. Since we assume Gaussian uncertainty, this work uses the **first-order error propagation model** [8].

Let $\mathrm{y} = (y_1, y_2, \cdots, y_s)^T$ be a vector calculated as a function of $\mathrm{x} = (x_1, x_2, \cdots, x_t)^T$. By expressing the uncertain data as a vector-valued random variable $\underline{\mathrm{y}}$, it follows that the expectation (mean) of $\underline{\mathrm{y}}$ is given by $\overline{\mathrm{y}} = f(\overline{\mathrm{x}})$. For the case where the input components of $\underline{\mathrm{x}}$ are correlated, the covariance matrix of $\underline{\mathrm{y}}$ can be approximated using the first-order error propagation model for vector-valued random variables with Gaussian distributed uncertainty

$$\Sigma_{\mathrm{y}} \approx \mathrm{J}_{\mathrm{y}}\,\Sigma_{\mathrm{x}}\,\mathrm{J}_{\mathrm{y}}^{\mathrm{T}}, \tag{13}$$

where Σ_{x} is the covariance matrix of the vector-valued random variables $\underline{\mathrm{x}}$, and J_{y} is the Jacobian matrix for the function f that computes the expectation $\overline{\mathrm{y}}$ of $\underline{\mathrm{y}}$ from the terms of the expectation $\overline{\mathrm{x}}$ of $\underline{\mathrm{x}}$.

It is important to comment that a function of Gaussian distributed random variables may produce a non-Gaussian distribution. First-order error propagation provides, in that case, an approximation of the resulting uncertainty. As pointed out by Cowan [8], it should be checked whether the resultant distribution can be approximated well by a Gaussian distribution. The amount of tolerable error depends on the actual application. For cases where the re-

sulting distribution is Gaussian, a better approximation of the uncertainty can be achieved by extending the first-order error propagation model (13) to higher orders. Note that (13) considers only the first-order derivatives of the function f that computes $\overline{\mathrm{y}}$. Exact results are obtained when all derivatives up to order four are considered because the derivatives of orders higher than four are equal to zero for functions producing Gaussian distributed uncertain [10].

In this paper, we show that the first-order error propagation (*i.e.* (13)) fits our subspace detection framework well. This is because input entries are processed individually by the mapping and voting procedures of our detection scheme. In this case, if an input has Gaussian distributed uncertainty, such uncertainty will be kept Gaussian up to a well-known stage of the computational flow. After that, the distribution is naturally converted to non-Gaussian through the correspondence between the auxiliary and the actual parameter space.

3.3 Tensor Representation of Geometric Algebra Operations

First-order error propagation (Section 3.2) can be applied to GA equations (Section 3.1) by expressing multivectors as component vectors and GA operations as tensor contractions [10]. Using such a representation, the Jacobian matrix in (13) can be calculated as for linear algebra equations. In order to express multivectors as component vectors, let $\{\mathbf{E}_i\}_{i=1}^{2^n}$ be the basis of a multivector space $\bigwedge \mathbb{R}^n$. A multivector $M \in \bigwedge \mathbb{R}^n$ may then be written as $M = \sum_{i=1}^{2^n} (\mu^i \, \mathbf{E}_i)$, where μ^i is the i-th component of a vector in $\mathbb{R}^{2^n}$.

The geometric product between two multivectors A and B may be written as

$$C = A \, B = \sum_{i,j,k=1}^{2^n} \left(\alpha^j \, \beta^k \, \Gamma^{i,j,k} \, \mathbf{E}_i \right), \tag{14}$$

where $\{\alpha^j\}_{j=1}^{2^n}$ and $\{\beta^k\}_{k=1}^{2^n}$ are the coefficients of A and B, respectively, and Γ is a 3rd-rank tensor encoding the geometric product (2). If $C = \sum_{i=1}^{2^n} (\gamma^i \, \mathbf{E}_i)$ then $\gamma^i = \sum_{j,k=1}^{2^n} \left(\alpha^j \, \beta^k \, \Gamma^{i,j,k} \right)$, $\forall \; i \in \{1, 2, \cdots, 2^n\}$. The tensor Γ does not depend on the arguments A and B in (14). Its entries are obtained as

$$\mathbf{E}_j \, \mathbf{E}_k = \Gamma^{i,j,k} \, \mathbf{E}_i \quad \forall \; j, k \in \{1, 2, \cdots, 2^n\}. \tag{15}$$

Thus, Γ is constant for the purpose of computing the derivatives in the Jacobian matrix (13). The principle depicted in (15) can be used to compute a different tensor for each bilinear product presented in Section 3.1. This is achieved just by replacing the geometric product in (15) by the intended product.

4 General Framework for Subspace Detection

The subspace detection scheme presented in [5] is a general procedure that systematically adapts itself to the intended detection case. The user only needs to choose a model of geometry (MOG) for which the type of data alignment to be detected is characterized by a p-dimensional linear subspace (*i.e.,* a p-blade) in some n-dimensional metric space. Given some input dataset $\mathcal{X}$, the detection is performed in three steps: (i) create an accumulator array (a discretization of the parameter space characterizing p-blades); (ii) perform a voting procedure where the input dataset is mapped to the accumulator array; and (iii) search for the peaks of votes in the accumulator, as they correspond to the p-blades that best fit the input dataset $\mathcal{X}$, and output them.

In [5] it is shown that a p-dimensional subspace $\mathbf{B}_{\langle p \rangle}$ in a n-dimensional space can be represented by a set of $m = p\,(n-p)$ rotations applied to a canonical subspace used as reference. Such a representation leads to the model function

$$\mathbf{B}_{\langle p \rangle} = \boldsymbol{T}\,\mathbf{E}_{\langle p \rangle}\,\widetilde{\boldsymbol{T}}, \tag{16}$$

where $\mathbf{E}_{\langle p \rangle}$ is the reference subspace (defined in (19)), and

$$\boldsymbol{T} = \boldsymbol{R}_m\,\boldsymbol{R}_{m-1}\cdots\boldsymbol{R}_1 \tag{17}$$

is a rotor encoding a sequence of rotations

$$\boldsymbol{R}_t = \cos\left(\theta^t/2\right) - \sin\left(\theta^t/2\right)\,\mathbf{P}^{(t)}_{\langle 2 \rangle}, \tag{18}$$

of θ^t radians on the unit planes $\mathbf{P}^{(t)}_{\langle 2 \rangle} = \mathbf{e}_{j+1} \wedge \mathbf{e}_j$ with $j = h\,(h + 2\,q - n) - t + 1$, where h is the lowest value in the strictly increasing sequence $\{1, 2, \cdots, n-q\}$ that satisfies the condition $t \leq h\,(h + 2\,q - n)$, for $q = \max(p, n-p)$. Please, see [5, Eqs. (17)-(23) and (34)] for a comprehensive definition of $\mathbf{P}^{(t)}_{\langle 2 \rangle}$.

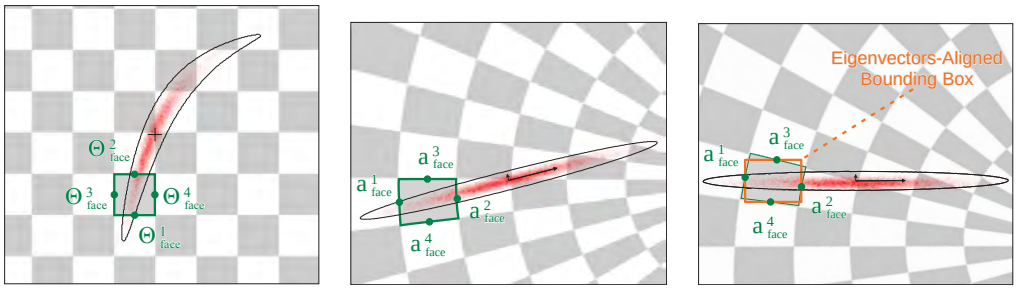
The canonical reference subspace in (16) is given by

$$\mathbf{E}_{\langle p \rangle} = \begin{cases} \bigwedge_{v \in \mathcal{V}} \mathbf{e}_v & \text{for } p \neq q \\ \bigvee_{v \in \mathcal{V}} \mathbf{e}^*_v & \text{for } p = q \end{cases}. \tag{19}$$

$\bigwedge_{v \in \mathcal{V}}$ is the outer product (3) of vectors $\mathbf{e}_v$, and $\bigvee_{v \in \mathcal{V}}$ is the regressive product (11) of pseudovectors $\mathbf{e}^*_v$ (the dual (9) of vectors $\mathbf{e}_v$), for $\mathcal{V} = \{2\,(q+i) - n\}_{i=1}^{n-q}$.

By taking $\mathbf{E}_{\langle p \rangle}$ and the rotation planes $\mathbf{P}^{(t)}_{\langle 2 \rangle}$ as constant values, it is clear that the m rotation angles (θ^t) related to the sequence of rotation operations can be used to unequivocally characterize a p-dimensional subspace in the underlying n-dimensional space. Therefore, it is possible to define a parameter space

$$\mathbb{P}^m = \{(\theta^1, \theta^2, \cdots, \theta^m) \mid \theta^t \in [-\pi/2, \pi/2)\}, \tag{20}$$



(a) (b) (c)

Fig. 1. The red spots in (a)-(c) represent the uncertainty of a given input blade. The distribution is shown as points computed by sampling the input blade according to its Gaussian distributed uncertainty and mapping each sample to $\mathbb{P}^2$ (a) and to $\mathbb{A}^2$ (b). The envelopes in (a) and (b) were obtained analytically by our approach. The points Θ^i_{face} at the center of the bin's face (a) are mapped to the open affine covering for $G(2,3)$ as points $\mathrm{a}^i_{\text{face}}$ (b). In turn, the points $\mathrm{a}^i_{\text{face}}$ are mapped to the basis defined by the eigenvectors of the probability distribution (c). An axis-aligned bounding box is computed for $\mathrm{a}^i_{\text{face}}$ in such a basis. The number of votes to be incremented to a given bin of the accumulator array is proportional to the weight of the input blade and the probabilities of an intended p-blade be inside of the box.

where each parameter vector $(\theta^1, \theta^2, \cdots, \theta^m) \in \mathbb{P}^m$ characterizes a p-blade.

In the original formulation of our subspace detection framework [5], the voting procedure essentially takes each r-dimensional subspace $\mathbf{X}_{\langle r \rangle}$ in the input dataset $\mathcal{X}$ and identifies the parameters (coordinates in $\mathbb{P}^m$, (20)) of all p-dimensional subspaces related to it. When $r \leq p$, the mapping process identifies in $\mathbb{P}^m$ all p-dimensional subspaces containing $\mathbf{X}_{\langle r \rangle}$. By duality, when $r \geq p$, the procedure identifies in $\mathbb{P}^m$ all p-dimensional subspaces contained in $\mathbf{X}_{\langle r \rangle}$. As the input entries are mapped, the bins of the accumulator related to such a mapping are incremented by some importance value of the entry.

In conventional voting-based approaches, such as the HTs, the input data type is known *a priori*. Thus, conventional mapping procedures predefine which parameters of the related parameter vectors must be arbitrated and which ones must be computed. The general approach, on the other hand, does not have prior information about input data. It decides at runtime how to treat each parameter. Such a behavior is key for the generality of this detection framework, providing a closed-form solution for the detection of subspaces of a given dimensionality p on datasets that may be heterogeneous and contain elements (*i.e.,* subspaces) with arbitrary dimensionalities ($0 \leq r \leq n$).

The last step of the subspace detection framework consists in identifying the bins that correspond to local maxima in the accumulator array. It returns a list with all detected peaks, sorted by number of votes. The coordinates of such bins (*i.e.,* parameter vectors) identify the most significant p-blades.

5 Extended Framework to Handle Uncertain Data

First-order error propagation (Section 3.2) provides a good approximation for Gaussian distributed uncertainty [8,10]. However, Fig. 1a clearly shows that the resulting distribution of votes in the parameter space is non-Gaussian. For instance, it is not symmetric around the mean (indicated by a + in Fig. 1a), and the main axes are bent. Hence, standard first-order error propagation cannot be applied directly to the computation chain of the mapping procedure discussed in Section 4. The technique described in Sections 5.2 and 5.3 provides an alternative computation flow to propagate the uncertainty through the algorithms shown in Fig. 2 (*Mapping Procedure*) and Fig. 3 (*Calculate Parameters*). These algorithms represent a resulting parameter vector $\Theta^{(0)}$ mapped as the point at the origin of an m-dimensional open affine covering $\mathbb{A}^m$ for the Grassmannian $G(p,n)$ (Section 5.1 introduces the concept of Grassmannian). This way, the uncertainty of $\Theta^{(0)}$ is described by a multivariate Gaussian distribution at the origin of $\mathbb{A}^m$. Fig. 1b illustrates the affine space and the probability distribution for the example in Fig. 1a.

5.1 A Coordinate Chart for the Grassmannian

The multivector representation of k-blades resides in $\bigwedge^k \mathbb{R}^n$ (*i.e.,* the portion of $\bigwedge \mathbb{R}^n$ with k-dimensional basis elements). However, not every number in $\bigwedge^k \mathbb{R}^n$ is a blade, except for $k \in \{0, 1, n-1, n\}$. As pointed out in Section 3.1, blades can be built as the outer product of k independent vectors. The set of all k-blades of $\mathbb{R}^n$ defines a projective variety of dimension $k\,(n-k)$ in the $\binom{n}{k}$-dimensional space of $\bigwedge^k \mathbb{R}^n$ [35]: the **Grassmannian** $G(k,n)$.

A natural consequence of the dimensionality of $G(k,n)$ is that an arbitrary k-blades requires at least $k\,(n-k)$ coordinates to be addressed in such variety. By choosing a reference blade, one may define an open affine covering $\mathbb{A}^{k\,(n-k)}$ for $G(k,n)$. The covering is open because the k-blades orthogonal to the reference are not properly represented in $\mathbb{A}^{k\,(n-k)}$ (*i.e.,* they reside at infinity). The remaining k-blades in $G(k,n)$, on the other hand, are represented uniquely as points in $\mathbb{A}^{k\,(n-k)}$, where the reference blade is related to the point at the origin. To see this in coordinates, consider the subspace spanned by $\{\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_k\}$ as the reference. It follows that any k-blade in the open affine covering of $G(k,n)$ may be represented as the row space of a unique matrix

$$\begin{pmatrix} 1 & \cdots & 0 & \alpha^{1,1} & \cdots & \alpha^{1,n-k} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 1 & \alpha^{k,1} & \cdots & \alpha^{k,n-k} \end{pmatrix}, \tag{21}$$

where the entries $\alpha^{i,j}$ define a location in $\mathbb{A}^{k\,(n-k)}$ [35]. Thus, a subspace may be mapped from a point in $\mathbb{A}^{k\,(n-k)}$ to a blade $\mathbf{B}_{\langle k \rangle} \in \bigwedge^k \mathbb{R}^n$ through

$$\mathbf{B}_{\langle k \rangle} = \bigwedge_{i=1}^{k} \left(\mathbf{e}_i + \sum_{j=1}^{n-k} \left(\alpha^{i,j}\, \mathbf{e}_{k+j} \right) \right).$$

$\mathbf{B}_{\langle k \rangle}$ may be mapped from $\bigwedge^k \mathbb{R}^n$ to $\mathbb{A}^{k\,(n-k)}$ by decomposing $\mathbf{B}_{\langle k \rangle}$ into vector factors and computing the row reduced echelon form of its $k \times n$ matrix representation. This lends to (21) when $\mathbf{B}_{\langle k \rangle}$ is not orthogonal to $\mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \cdots \wedge \mathbf{e}_k$.

According to [5], the parameter space $\mathbb{P}^m$ (20) provides a coordinate chart for $G(p,n)$. In such a coordinate system, a p-blade is addressed by a set of $p\,(n-p)$ rotation angles in the $[-\pi/2, \pi/2)$ range (a parameter vector). In contrast to the open affine covering $\mathbb{A}^{k\,(n-k)}$ of $G(k,n)$, this parameterization can represent *all* the p-blade in $\bigwedge^p \mathbb{R}^n$.

5.2 Mapping Procedure for $r \geq p$

We extend the mapping procedure described in [5] from exact to uncertain input data. Due to space limitations, we only discuss the changes to the original algorithms. Please, refer to [5] for more details about our previous work.

The procedure that maps input r-blades with uncertainty to the parameter space $\mathbb{P}^m$ characterizing p-dimensional subspaces is presented in Fig. 2 as the *Mapping Procedure* algorithm, for $r \geq p$. The function called in line 6 is presented in the algorithm *Calculate Parameters* (Fig. 3). The *Mapping Procedure* algorithm takes as input a random multivector variable $\mathbf{X}_{\langle r \rangle}$, whose expectation $\overline{\mathbf{X}_{\langle r \rangle}}$ is a blade and whose covariance matrix is $\Sigma_{\mathbf{X}_{\langle r \rangle}}$. The procedure returns a set of 2-tuples comprised by a parameter vector $\Theta^{(0)} \in \mathbb{P}^m$ (Fig. 2, line 10), and a vector-valued random variable $\underline{\mathrm{a}}$. By definition, the expectation of $\underline{\mathrm{a}}$ is at the origin of $\mathbb{A}^m$ (*i.e.,* $\bar{\mathrm{a}} = (0, 0, \cdots, 0)^{\mathrm{T}} \in \mathbb{A}^m$). The covariance matrix of $\underline{\mathrm{a}}$ is computed with the first-order error propagation model (13)

$$\Sigma_{\mathrm{a}} = \mathrm{J_a}\, \Sigma_{\mathbf{X}_{\langle \mathrm{r} \rangle}}\, \mathrm{J_a^T}. \tag{22}$$

In order to evaluate (22), one needs to compute the Jacobian matrix $\mathrm{J_a}$ for the equation that calculates the mean point $\bar{\mathrm{a}}$ in terms of the input mean blade $\overline{\mathbf{X}_{\langle r \rangle}}$. However, it is impractical to obtain a single equation that expresses the entire computation chain and from it compute $\mathrm{J_a}$. Note that intermediate variables can be combined in different ways. The combination depends on which parameters must be arbitrated and which ones must be computed while mapping the input data to the parameter space. As a result, the definition of

Algorithm: *Mapping Procedure*

Require: A random multivector variable $\underline{\mathbf{X}_{\langle r\rangle}}$ described by $\overline{\mathbf{X}_{\langle r\rangle}}$ and $\Sigma_{\mathbf{X}_{\langle \mathrm{r}\rangle}}$

1: $\mathcal{P}^{(m)} \leftarrow \left\{(\overline{\mathbf{X}_{\langle r\rangle}}, \mathrm{I}, 1, 0, \varnothing)\right\}$
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 (16)
4: $\mathcal{P}^{(t-1)} \leftarrow \varnothing$
5: **for all** $(\mathbf{X}^{(t)}_{\langle r\rangle}, \mathrm{J}_{\mathbf{X}^{(t)}_{\langle r\rangle}}, \boldsymbol{K}^{(t)}, \mathrm{J}_{\boldsymbol{K}^{(t)}}, \Theta^{(t)}) \in \mathcal{P}^{(t)}$ **do**
6: $\mathcal{T} \leftarrow$ the result of *Calculate Parameters* having $\mathbf{X}^{(t)}_{\langle r\rangle}$ and $\mathrm{J}_{\mathbf{X}^{(t)}_{\langle r\rangle}}$ as input
7: $\mathcal{P}^{(t-1)} \leftarrow \mathcal{P}^{(t-1)} \cup \big\{(\mathbf{X}^{(t-1)}_{\langle r\rangle}, \mathrm{J}_{\mathbf{X}^{(t-1)}_{\langle r\rangle}}, \boldsymbol{K}^{(t-1)}, \mathrm{J}_{\boldsymbol{K}^{(t-1)}}, (\theta^t, \Theta^{(t)}_1, \Theta^{(t)}_2, \cdots, \Theta^{(t)}_{m-t}))$
$\mid (\theta^t, \mathrm{J}_{\theta^t}) \in \mathcal{T}$, and $\mathbf{X}^{(t-1)}_{\langle r\rangle}$, and $\boldsymbol{K}^{(t-1)}$ are defined in (26) and (29)$\big\}$
8: **end for**
9: **end for**
10: **return** $\big\{(\Theta^{(0)}, \underline{\mathrm{a}}) \mid \overline{\mathrm{a}}$ is computed according to (33),
$\Sigma_{\mathrm{a}} = \mathrm{J}_{\mathrm{a}}\, \Sigma_{\mathbf{X}_{\langle \mathrm{r}\rangle}}\, \mathrm{J}_{\mathrm{a}}^{\mathrm{T}}$, and $(\mathbf{X}^{(0)}_{\langle r\rangle}, \mathrm{J}_{\mathbf{X}^{(0)}_{\langle r\rangle}}, \boldsymbol{K}^{(0)}, \mathrm{J}_{\boldsymbol{K}^{(0)}}, \Theta^{(0)}) \in \mathcal{P}^{(0)}\big\}$

Fig. 2. The procedure that extends the algorithm presented in [5] to blades with uncertainty. It takes as input an random r-blade $\underline{\mathbf{X}_{\langle r\rangle}}$ and returns a set of pairs comprised by a parameter vector $\Theta^{(0)} \in \mathbb{P}^m$ characterizing a p-blade in $\overline{\mathbf{X}_{\langle r\rangle}}$, and a vector-valued random variable $\underline{\mathrm{a}}$ describing the Gaussian uncertainty of the p-blade represented as the origin of the open affine covering of the Grassmannian.

J_{a} must handle all possible computation flows. The solution for this problem is to solve the partial derivatives in J_{a} using the chain rule, step-by-step, until the final result is found. In Figs. 2 and 3 the derivatives of intermediate computation steps are kept as the Jacobian matrices of intermediate variables. The following derivations show how these matrices are evaluated.

The mapping starts by initializing a set $\mathcal{P}^{(m)}$ (Fig. 2, line 1) with a 5-tuple

$$(\mathbf{X}^{(m)}_{\langle r\rangle}, \mathrm{J}_{\mathbf{X}^{(m)}_{\langle r\rangle}}, \boldsymbol{K}^{(m)}, \mathrm{J}_{\boldsymbol{K}^{(m)}}, \Theta^{(m)}) \in \mathcal{P}^{(m)}, \tag{23}$$

where $\mathbf{X}^{(m)}_{\langle r\rangle} = \overline{\mathbf{X}_{\langle r\rangle}}$ is the input (mean) blade, $\mathrm{J}_{\mathbf{X}^{(m)}_{\langle r\rangle}} = \mathrm{I}$ is the Jacobian matrix of $\mathbf{X}^{(m)}_{\langle r\rangle}$ with respect to $\underline{\mathbf{X}_{\langle r\rangle}}$ (*i.e.,* an identity matrix), and $\boldsymbol{K}^{(m)} = 1$ is an identity rotor denoting that no rotor $\boldsymbol{R}_t$ was computed yet. In subsequent steps of the algorithm, $\boldsymbol{K}^{(t)}$ is a rotor obtained as the geometric product of the last $(m-t)$ rotors $\boldsymbol{R}_t$ applied to $\mathbf{E}_{\langle p\rangle}$ in (16), *i.e.,* $\boldsymbol{K}^{(t)} = \boldsymbol{K}^{(t+1)}\, \boldsymbol{R}_{t+1}$, for $1 \leq t \leq m$, and $\boldsymbol{K}^{(m+1)} = 1$. At the end of the mapping process, $\boldsymbol{K}^{(0)} = \boldsymbol{T}$ ($\boldsymbol{T}$ is defined in (17)) is the rotor used to transform the reference blade $\mathbf{E}_{\langle p\rangle}$ into the blade characterized by the resulting parameter vector $\Theta^{(0)}$ (line 10). In (23), $\mathrm{J}_{\boldsymbol{K}^{(m)}} = 0$ is the Jacobian matrix of $\boldsymbol{K}^{(m)}$ (*i.e.,* a zero row vector), and $\Theta^{(m)} = \varnothing$ is an empty set denoting that no parameter was calculated yet.

At each iteration of the procedure (lines 2 to 9), the function called in line 6 and defined in Fig. 3 (*Calculate Parameters*) returns a set $\mathcal{T}$ of 2-tuples $(\theta^t, \mathrm{J}_{\theta^t}) \in \mathcal{T}$, where θ^t is the t-th parameter of some p-blade related to $\overline{\mathbf{X}_{\langle r \rangle}}$, and J_{θ^t} is its Jacobian matrix, whose definition is presented later in (47). The rotation angle θ^t is used in line 7 of Fig. 2 to compute the rotor $\boldsymbol{R}_t$ as

$$\boldsymbol{R}_t = \cos\left(\theta^t/2\right) - \sin\left(\theta^t/2\right) \mathbf{P}^{(t)}_{\langle 2 \rangle} = \cos\left(\theta^t/2\right) - \sin\left(\theta^t/2\right) \sum_{i=1}^{2^n} \left(\phi^i_t \, \mathbf{E}_i\right). \tag{24}$$

$\mathbf{P}^{(t)}_{\langle 2 \rangle}$ is a constant rotation plane with coefficients $\{\phi^i_t\}^{2^n}_{i=1}$, leading to

$$\mathrm{J}^{i,z}_{\boldsymbol{R}_t} = \frac{\partial \rho^i_t}{\partial \chi^z} = -\frac{1}{2} \mathrm{J}^{i,z}_{\theta^t} \begin{cases} \sin\left(\theta^t/2\right) & \text{for } i = 1 \\ \phi^i_t \cos\left(\theta^t/2\right) & \text{otherwise} \end{cases}. \tag{25}$$

Following the tensor representation introduced in Section 3.3, $\{\rho^i_t\}^{2^n}_{i=1}$ and $\{\chi^z\}^{2^n}_{z=1}$ in (25) denote the coefficients of, respectively, $\boldsymbol{R}_t$ and $\overline{\mathbf{X}_{\langle r \rangle}}$. The rotor $\boldsymbol{R}_t$ is used in line 7 to compute

$$\mathbf{X}^{(t-1)}_{\langle r \rangle} = \widetilde{\boldsymbol{R}}_t \, \mathbf{X}^{(t)}_{\langle r \rangle} \, \boldsymbol{R}_t = \sum_{i,j,k,l=1}^{2^n} \left(\rho^j_t \, \lambda^k_t \, \rho^l_t \, \Psi^{i,j,k,l} \, \mathbf{E}_i\right), \tag{26}$$

where $\{\rho^i_t\}^{2^n}_{i=1}$ and $\{\lambda^i_t\}^{2^n}_{i=1}$ denote the coefficients of $\boldsymbol{R}_t$ and $\mathbf{X}^{(t)}_{\langle r \rangle}$, respectively. The tensor

$$\Psi^{i,j,k,l} = \Upsilon^{j,j} \sum_{h=1}^{2^n} \left(\Gamma^{h,j,k} \, \Gamma^{i,h,l}\right) \tag{27}$$

is comprised by constant values computed from the tensors Γ and Υ encoding the geometric product and the reverse operation, respectively. The derivatives in the Jacobian matrix of $\mathbf{X}^{(t-1)}_{\langle r \rangle}$ (Fig. 2, line 7) are given by

$$\mathrm{J}^{i,z}_{\mathbf{X}^{(t-1)}_{\langle r \rangle}} = \frac{\partial \lambda^i_{t-1}}{\partial \chi^z} = \sum_{j,k,l=1}^{2^n} \left(\left(\rho^j_t \, \mathrm{J}^{k,z}_{\mathbf{X}^{(t)}_{\langle l \rangle}} \, \rho^l_t + \lambda^k_t \left(\mathrm{J}^{j,z}_{\boldsymbol{R}_t} \, \rho^l_t + \rho^j_t \, \mathrm{J}^{l,z}_{\boldsymbol{R}_t}\right)\right) \Psi^{i,j,k,l}\right). \tag{28}$$

The Jacobian matrix of $\boldsymbol{R}_t$ ($\mathrm{J}_{\boldsymbol{R}_t}$, in (28)) is defined in (25). The rotor $\boldsymbol{R}_t$ is also used in Fig. 2 (line 7) to compute

$$\boldsymbol{K}^{(t-1)} = \boldsymbol{K}^{(t)} \, \boldsymbol{R}_t = \sum_{i,j,k=1}^{2^n} \left(\kappa^j_t \, \rho^k_t \, \Gamma^{i,j,k} \, \mathbf{E}_i\right). \tag{29}$$

The coefficients of $\boldsymbol{K}^{(t-1)}$ are denoted by $\{\kappa^i_{t-1}\}^{2^n}_{i=1}$ and its Jacobian matrix is

$$\mathrm{J}^{i,z}_{\boldsymbol{K}^{(t-1)}} = \frac{\partial \kappa^i_{t-1}}{\partial \chi^z} = \sum_{j,k=1}^{2^n} \left(\left(\mathrm{J}^{j,z}_{\boldsymbol{K}^{(t)}} \, \rho^k_t + \kappa^j_t \, \mathrm{J}^{k,z}_{\boldsymbol{R}_t}\right) \Gamma^{i,j,k}\right). \tag{30}$$

After all θ^t parameters of the p-blades related to $\overline{\mathbf{X}_{\langle r\rangle}}$ have been calculated, one also has computed their respective rotors $\boldsymbol{K}^{(0)} = \boldsymbol{T}$. Recall from (16) that a rotor $\boldsymbol{T}$ transforms the reference blade $\mathbf{E}_{\langle p\rangle}$ into the blade $\mathbf{C}_{\langle p\rangle}$ related to a given parameter vector $\Theta^{(0)}$. The last step of the mapping procedure is to define the open affine covering $\mathbb{A}^m$ for the Grassmannian (Section 5.1) in such a way that $\mathbf{C}_{\langle p\rangle}$ (and $\Theta^{(0)}$) is represented as the point at the origin of $\mathbb{A}^m$. Such origin point is denoted by $\bar{\mathrm{a}}$. The computation of its coordinates leads to the Jacobian matrix $\mathrm{J_a}$ (see (34)) used in (22) to compute the covariance matrix of a vector-valued random variable $\underline{\mathrm{a}}$ (line 10).

The coordinates of $\bar{\mathrm{a}}$ are equal to *zero*. According to (21), the origin of $\mathbb{A}^m$ is actually related to the blade $\mathbf{A}_{\langle p\rangle} = \mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \cdots \wedge \mathbf{e}_p$, and not to an arbitrary blade $\mathbf{C}_{\langle p\rangle}$. Thus, a mapping from $\mathbf{C}_{\langle p\rangle}$ to $\mathbf{A}_{\langle p\rangle}$ must be defined. The rotor

$$\boldsymbol{W} = \boldsymbol{T}_A \, \widetilde{\boldsymbol{K}}^{(0)} = \sum_{i,j,k=1}^{2^n} \left(\zeta^j \, \kappa_0^k \, \Upsilon^{k,k} \, \Gamma^{i,j,k} \, \mathbf{E}_i \right) \tag{31}$$

performs a change of basis, mapping $\mathbf{C}_{\langle p\rangle}$ to $\mathbf{A}_{\langle p\rangle}$ (*i.e.,* $\boldsymbol{W}\,\mathbf{C}_{\langle p\rangle}\,\widetilde{\boldsymbol{W}} = \mathbf{A}_{\langle p\rangle}$). In (31), $\boldsymbol{T}_A$ is the rotor that transforms $\mathbf{E}_{\langle p\rangle}$ into $\mathbf{A}_{\langle p\rangle}$. Its coefficients are denoted as $\{\zeta^j\}_{j=1}^{2^n}$. Notice that $\boldsymbol{T}_A$ may be precomputed from the parameter vector returned by the procedure in Fig. 2 when $\mathbf{A}_{\langle p\rangle}$ is given as input.

The Jacobian matrix of $\boldsymbol{W}$ in (31) is computed as ($\mathrm{J}_{\boldsymbol{K}^{(0)}}$ is given by (30)):

$$\mathrm{J}_{\boldsymbol{W}}^{i,z} = \frac{\partial \omega^i}{\partial \chi^z} = \sum_{j,k=1}^{2^n} \left(\zeta^j \, \mathrm{J}_{\boldsymbol{K}^{(0)}}^{k,z} \, \Gamma^{i,j,k} \right). \tag{32}$$

Finally, the coordinates α^t of $\bar{\mathrm{a}}$ (denoted by $\alpha^{i,j}$ in (21)) are computed as

$$\alpha^t = \left(\boldsymbol{W}\,\mathbf{c}_i\,\widetilde{\boldsymbol{W}} \right) * \mathbf{e}_{p+j} = 0, \tag{33}$$

where $t = (i-1)\,(n-p) + j$, for $i \in \{1, 2, \cdots, p\}$ and $j \in \{1, 2, \cdots, n-p\}$. In (33), the vector $\boldsymbol{W}\,\mathbf{c}_i\,\widetilde{\boldsymbol{W}} = \mathbf{e}_i + \sum_{j=1}^{n-p} (\alpha^{i,j}\,\mathbf{e}_{p+j}) = \mathbf{e}_i$ is at the i-th row of the matrix representation of $\mathbf{A}_{\langle p\rangle}$ in row reduced echelon form (21), and $\mathbf{c}_i = \widetilde{\boldsymbol{W}}\,\mathbf{e}_i\,\boldsymbol{W}$ is the i-th vector spanning $\mathbf{C}_{\langle p\rangle} = \mathbf{c}_1 \wedge \cdots \wedge \mathbf{c}_i \wedge \cdots \wedge \mathbf{c}_p$.

From (33), the coordinates $\{\alpha^t\}_{t=1}^m$ of $\bar{\mathrm{a}}$ can be rewritten in tensor form as

$$\alpha^t = \sum_{h,i=1}^{2^n} \left(\epsilon_{\ell_2}^h \, \Lambda^{1,i,h} \sum_{j,k,l=1}^{2^n} \left(\omega^j \, \gamma_{\ell_1}^k \, \omega^l \, \Xi^{i,j,k,l} \right) \right),$$

where Λ is a 3rd-rank tensor encoding the left contraction (5), leading to the

Jacobian matrix

$$\mathrm{J}_{\mathrm{a}}^{t,z} = \frac{\partial \alpha^t}{\partial \chi^z} = \sum_{h,i=1}^{2^n} \left(\epsilon_{\ell_2}^h \, \Lambda^{1,i,h} \sum_{j,k,l=1}^{2^n} \left(\gamma_{\ell_1}^k \, \left(\mathrm{J}_{\boldsymbol{W}}^{j,z} \, \omega^l + \omega^j \, \mathrm{J}_{\boldsymbol{W}}^{l,z} \right) \Xi^{i,j,k,l} \right) \right), \tag{34}$$

where $\ell_1 = \lceil \frac{t}{n-p} \rceil$, $\ell_2 = t + n - \lceil \frac{t}{n-p} \rceil \, (n-p)$, and $\lceil \square \rceil$ is the ceiling function. $\mathrm{J}_{\boldsymbol{W}}$ is defined in (32). Constants $\{\gamma_{\ell_1}^k\}_{k=1}^{2^n}$ and $\{\epsilon_{\ell_2}^h\}_{h=1}^{2^n}$ are the coefficients of $\mathbf{c}_{\ell_1}$ and $\mathbf{e}_{\ell_2}$, respectively. $\Xi^{i,j,k,l} = \Upsilon^{l,l} \sum_{h=1}^{2^n} \left(\Gamma^{h,j,k} \, \Gamma^{i,h,l} \right)$ is also a constant.

5.2.1 Function Calculate Parameters

Fig. 3 complements the procedure in Fig. 2, taking as input blade $\mathbf{X}_{\langle r \rangle}^{(t)}$ ($\mathbf{Y}^{(t)}$) and the Jacobian matrix $\mathrm{J}_{\mathbf{X}_{\langle r \rangle}^{(t)}}$ ($\mathrm{J}_{\mathbf{Y}^{(t)}}$), computed in Fig. 2 (see Fig. 2, line 6). The meet operation $\cap$ in Fig. 3 (line 5) is analogous to intersection in set theory. It returns the subspace shared by $\mathbf{Y}^{(t)}$ and the space of possibilities $\mathbf{F}_l^{(t)}$ (37). In the new algorithm, meet is evaluated in terms of the pseudoscalar $\mathbf{I}_l^{(t)}$ (*i.e.*, $\mathbf{I}_l^{(t)} = \mathbf{Y}^{(t)} \cup \mathbf{F}_l^{(t)}$, where $\cup$ denotes the join operation in GA, analogous to union). Thus, meet reduces to the application of two left contractions

$$\mathbf{M}_l^{(t)} = \mathbf{Y}^{(t)} \cap \mathbf{F}_l^{(t)} = \left(\mathbf{F}_l^{(t)} \rfloor \, (\mathbf{I}_l^{(t)})^{-1} \right) \rfloor \, \mathbf{Y}^{(t)} = \sum_{i,j,k=1}^{2^n} \left(\delta_{t,l}^j \, \eta_t^k \, \Lambda^{i,j,k} \, \mathbf{E}_i \right). \tag{35}$$

In our implementation, we compute $\mathbf{I}_l^{(t)}$ using the join algorithm described in [36].

The **spaces of possibilities** are the regions of $\mathbb{R}^n$ that can be reached by vectors $\mathbf{v}_l \in \mathcal{E}$ as the rotation operations are applied to them, for $1 \leq l \leq |\mathcal{E}|$, where $|\mathcal{E}| = p$ denotes the cardinality of the set $\mathcal{E}$ (see [5] for details):

$$\mathcal{E} = \begin{cases} \{\mathbf{e}_v\}_{v \in \mathcal{V}} & \text{for } p \neq q \\ \{\mathbf{e}_v\}_{v \in \mathcal{V}} \backslash \{\mathbf{e}_i\}_{i=1}^n & \text{for } p = q \end{cases}, \tag{36}$$

where $\mathcal{A} \backslash \mathcal{B}$ denotes the relative complement of $\mathcal{A}$ in $\mathcal{B}$. Once $\mathcal{E}$ is defined, the spaces of possibilities can be computed as

$$\mathbf{F}_l^{(t)} = \begin{cases} \mathbf{F}_l^{(t-1)} \cup \mathbf{P}_{\langle 2 \rangle}^{(t)} & \text{for grade} \, (\mathbf{F}_l^{(t-1)} \cap \mathbf{P}_{\langle 2 \rangle}^{(t)}) = 1 \\ \mathbf{F}_l^{(t-1)} & \text{otherwise} \end{cases}, \tag{37}$$

where $\mathbf{F}_l^{(t)}$ is the space reachable by vector $\mathbf{v}_l \in \mathcal{E}$ after the application of the first t rotations. Therefore, $\mathbf{F}_l^{(0)} = \mathbf{v}_l$. $\mathbf{P}_{\langle 2 \rangle}^{(t)}$ is the plane where the t-th rotation happens, $\cup$ and $\cap$ denote, respectively, the join and the meet operations, and the grade function retrieves the dimensionality of a subspace.

In (35), $\left(\mathbf{F}_l^{(t)} \rfloor (\mathbf{I}_l^{(t)})^{-1}\right)$ results a constant blade. The derivatives in the Jacobian of $\mathbf{M}_l^{(t)}$ are given by

$$\mathrm{J}^{i,z}_{\mathbf{M}_l^{(t)}} = \frac{\partial \mu^i_{t,l}}{\partial \chi^z} = \sum_{j,k=1}^{2^n} \left(\delta^j_{t,l}\, \mathrm{J}^{k,z}_{\mathbf{Y}^{(t)}}\, \Lambda^{i,j,k}\right). \tag{38}$$

In (35) and (38), the coefficients of $\mathbf{M}_l^{(t)}$, $\left(\mathbf{F}_l^{(t)} \rfloor (\mathbf{I}_l^{(t)})^{-1}\right)$ and $\mathbf{Y}^{(t)}$ are denoted by $\{\mu^i_{t,l}\}_{i=1}^{2^n}$, $\{\delta^j_{t,l}\}_{j=1}^{2^n}$ and $\{\eta^k_t\}_{k=1}^{2^n}$, respectively.

$\mathcal{N}$ is a subset of $\mathcal{M}$ (Fig. 3, line 6). When $\mathcal{N}$ is empty (line 7), θ^t can assume any value in the $[-\pi/2,\pi/2)$ range. Hence, in Fig. 3 line 8, the Jacobian matrix of θ^t (*i.e.,* the second element in resulting tuples) is a zero row vector because the θ^t values do not depend on the input blade. When $\mathcal{N}$ is not empty, $\mathcal{O}$ is computed as a subset of $\mathcal{N}$ (line 10). In turn, $\mathcal{Q}$ (line 11) is defined as the set of 2-tuples $(\mathbf{q}_l^{(t)}, \mathrm{J}_{\mathbf{q}_l^{(t)}})$, where $\mathbf{q}_l^{(t)}$ is a nonzero vector resulting from the contraction of vectors $\mathbf{m}_l^{(t)} \in \mathcal{O}$ onto the rotation plane $\mathbf{P}^{(t)}_{\langle 2\rangle}$

$$\mathbf{q}_l^{(t)} = \mathbf{m}_l^{(t)} \rfloor \mathbf{P}^{(t)}_{\langle 2\rangle} = \sum_{i,j,k=1}^{2^n} \left(\mu^j_{t,l}\, \phi^k_t\, \Lambda^{i,j,k}\, \mathbf{E}_i\right). \tag{39}$$

The Jacobian matrix $\mathrm{J}_{\mathbf{q}_l^{(t)}}$ of $\mathbf{q}_l^{(t)}$ is computed from $\mathrm{J}_{\mathbf{m}_l^{(t)}}$ (38), and from the coefficients of $\mathbf{P}^{(t)}_{\langle 2\rangle}$ (denoted by $\{\phi^k_t\}_{k=1}^{2^n}$), and the tensor Λ

$$\mathrm{J}^{i,z}_{\mathbf{q}_l^{(t)}} = \frac{\partial \beta^i_{t,l}}{\partial \chi^z} = \sum_{j,k=1}^{2^n} \left(\mathrm{J}^{j,z}_{\mathbf{m}_l^{(t)}}\, \phi^k_t\, \Lambda^{i,j,k}\right). \tag{40}$$

When $\mathcal{Q}$ is not empty, one of the tuples in $\mathcal{Q}$ is used to compute the parameter θ^t (Fig. 13 line 13). However, in order to simplify the definition of the Jacobian of θ^t, the parameter is computed from $\tau_{t,l}$ and $\nu_{t,l}$

$$\theta^t = \tan^{-1}\left(\frac{\left(\mathbf{q}_l^{(t)} \wedge \mathbf{r}_l^{(t)}\right) * \mathbf{P}^{(t)}_{\langle 2\rangle}}{\mathbf{q}_l^{(t)} * \mathbf{r}_l^{(t)}}\right) = \tan^{-1}\left(\frac{\tau_{t,l}}{\nu_{t,l}}\right), \tag{41}$$

$$\tau_{t,l} = \left(\mathbf{q}_l^{(t)} \wedge \mathbf{r}_l^{(t)}\right) * \mathbf{P}^{(t)}_{\langle 2\rangle} = \mathbf{q}_l^{(t)} * \left(\mathbf{r}_l^{(t)} \rfloor \mathbf{P}^{(t)}_{\langle 2\rangle}\right) = \sum_{j,k=1}^{2^n} \left(\beta^j_{t,l}\, \Omega^{j,k}\right), \text{ and} \tag{42}$$

$$\nu_{t,l} = \mathbf{q}_l^{(t)} * \mathbf{r}_l^{(t)} = \sum_{j,k=1}^{2^n} \left(\beta^j_{t,l}\, \psi^k_{t,l}\, \Lambda^{1,j,k}\right), \tag{43}$$

with Ω in (42) being a constant defined as

$$\Omega^{j,k} = \Lambda^{1,j,k} \sum_{h,l=1}^{2^n} \left(\psi^h_{t,l}\, \phi^l_t\, \Lambda^{k,h,l}\right). \tag{44}$$

Algorithm: *Calculate Parameters*

Require: $\mathbf{Y}^{(t)}$, the current input blade
Require: $\mathrm{J}_{\mathbf{Y}^{(t)}}$, the Jacobian matrix of $\mathbf{Y}^{(t)}$ with respect to $\underline{\mathbf{X}_{\langle r \rangle}}$

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 (16)
2: Let $\mathbf{F}^{(t)}_{l}$ be a space of possibilities as defined in (37)
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: $\quad \mathcal{M} \leftarrow \left\{(\mathbf{M}^{(t)}_{l}, \mathrm{J}_{\mathbf{M}^{(t)}_{l}}) \mid \mathbf{M}^{(t)}_{l} = \mathbf{Y}^{(t)} \cap \mathbf{F}^{(t)}_{l},\ l \in \mathbb{Z}, \text{ and } 1 \leq l \leq |\mathcal{E}|\right\}$
6: $\quad \mathcal{N} \leftarrow \left\{(\mathbf{M}^{(t)}_{l}, \mathrm{J}_{\mathbf{M}^{(t)}_{l}}) \mid (\mathbf{M}^{(t)}_{l}, \mathrm{J}_{\mathbf{M}^{(t)}_{l}}) \in \mathcal{M}, \text{ and } \mathrm{grade}\,(\mathbf{M}^{(t)}_{l}) = |\mathcal{S}|\,,\right.$
$\qquad\qquad \left. \text{where } \mathcal{S} \leftarrow \left\{\mathbf{M}^{(t)}_{h} \mid (\mathbf{M}^{(t)}_{h}, \mathrm{J}_{\mathbf{M}^{(t)}_{h}}) \in \mathcal{M}, \text{ and } \mathbf{M}^{(t)}_{l} * \mathbf{M}^{(t)}_{h} \neq 0\right\}\right\}$
7: $\quad$ **if** $\mathcal{N} = \varnothing$ **then**
8: $\quad\quad$ **return** $\left\{(\theta^{t}, 0) \mid \theta^{t} \in [-\pi/2, \pi/2)\right\}$
9: $\quad$ **end if**
10: $\quad \mathcal{O} \leftarrow \left\{(\mathbf{M}^{(t)}_{l}, \mathrm{J}_{\mathbf{M}^{(t)}_{l}}) \mid (\mathbf{M}^{(t)}_{l}, \mathrm{J}_{\mathbf{M}^{(t)}_{l}}) \in \mathcal{N}, \text{ and } \mathrm{grade}\,(\mathbf{M}^{(t)}_{l}) = \mathrm{grade}\,(\mathbf{r}^{(t)}_{l}) = 1\right\}$
11: $\quad \mathcal{Q} \leftarrow \left\{(\mathbf{q}^{(t)}_{l}, \mathrm{J}_{\mathbf{q}^{(t)}_{l}}) \mid \mathbf{q}^{(t)}_{l} = (\mathbf{m}^{(t)}_{l} \rfloor \mathbf{P}^{(t)}_{\langle 2 \rangle}), \text{ and } (\mathbf{m}^{(t)}_{l}, \mathrm{J}_{\mathbf{m}^{(t)}_{l}}) \in \mathcal{O}, \text{ and } \mathbf{q}^{(t)}_{l} \neq 0\right\}$
12: $\quad$ **if** $\mathcal{Q} \neq \varnothing$ **then**
13: $\quad\quad$ **return** $\left\{(\theta^{t}, \mathrm{J}_{\theta^{t}}) \mid \theta^{t} = \tan^{-1}\left(((\mathbf{q}^{(t)}_{l} \wedge \mathbf{r}^{(t)}_{l}) * \mathbf{P}^{(t)}_{\langle 2 \rangle}) \,/\, (\mathbf{q}^{(t)}_{l} * \mathbf{r}^{(t)}_{l})\right),\right.$
$\qquad\qquad\qquad \left. \text{where } (\mathbf{q}^{(t)}_{l}, \mathrm{J}_{\mathbf{q}^{(t)}_{l}}) \text{ is one tuple in } \mathcal{Q}\right\}$
14: $\quad$ **end if**
15: $\quad \mathbf{Y}^{(t)} \leftarrow (\mathbf{M}^{(t)}_{l})^{-1} \rfloor \mathbf{Y}^{(t)}$, where $(\mathbf{M}^{(t)}_{l}, \mathrm{J}_{\mathbf{M}^{(t)}_{l}}) \in \mathcal{N}$ and $\mathbf{M}^{(t)}_{l}$ has the highest dimensionality in the set $\mathcal{N}$
16: $\quad \mathrm{J}_{\mathbf{Y}^{(t)}} \leftarrow$ the Jacobian matrix of $\mathbf{Y}^{(t)}$ (line 15) with respect of $\underline{\mathbf{X}_{\langle r \rangle}}$
17: **end loop**

Fig. 3. Function used in line 6 of the *Mapping Procedure* in Fig. 2. It extends the procedure presented in [5] by computing the Jacobian matrix of the intermediate variables with respect to the coefficients of the input variable $\underline{\mathbf{X}_{\langle r \rangle}}$ in Fig. 2.

In (42), (43) and (44), $\{\beta^{j}_{t,l}\}_{j=1}^{2^n}$, $\{\psi^{k}_{t,l}\}_{k=1}^{2^n}$ and $\{\phi^{l}_{t}\}_{l=1}^{2^n}$ denote the coefficients of $\mathbf{q}^{(t)}_{l}$ (39), $\mathbf{r}^{(t)}_{l} = \mathbf{F}^{(t-1)}_{l} \rfloor \mathbf{F}^{(t)}_{l}$ and $\mathbf{P}^{(t)}_{\langle 2 \rangle}$, respectively. The derivatives of $\tau_{t,l}$ (42) and $\nu_{t,l}$ (43) are, respectively

$$\mathrm{J}^{1,z}_{\tau_{t,l}} = \frac{\partial \tau_{t,l}}{\partial \chi^{z}} = \sum_{j,k=1}^{2^n} \left(\mathrm{J}^{j,z}_{\mathbf{q}^{(t)}_{l}}\, \Omega^{j,k}\right), \text{ and} \tag{45}$$

$$\mathrm{J}^{1,z}_{\nu_{t,l}} = \frac{\partial \nu_{t,l}}{\partial \chi^{z}} = \sum_{j,k=1}^{2^n} \left(\mathrm{J}^{j,z}_{\mathbf{q}^{(t)}_{l}}\, \psi^{k}_{t,l}\, \Lambda^{1,j,k}\right). \tag{46}$$

Given $\tau_{t,l}$ (42), $\nu_{t,l}$ (43), $\mathrm{J}_{\tau_{t,l}}$ (45), and $\mathrm{J}_{\nu_{t,l}}$ (46), the Jacobian of θ^t (41) is

$$\mathrm{J}^{1,z}_{\theta^t} = \frac{\partial \theta^t}{\partial \chi^z} = \frac{1}{(\tau_{t,l})^2 + (\nu_{t,l})^2} \left(\mathrm{J}^{1,z}_{\tau_{t,l}} \, \nu_{t,l} - \tau_{t,l} \, \mathrm{J}^{1,z}_{\nu_{t,l}} \right). \tag{47}$$

At each iteration of the loop in Fig. 3, blade $\mathbf{Y}^{(t)}$ is updated (line 15) removing from it the blade $\mathbf{M}^{(t)}_l \in \mathcal{N}$ with highest dimensionality. The new $\mathbf{Y}^{(t)}$ becomes

$$\mathbf{Y}^{(t)} = (\mathbf{M}^{(t)}_l)^{-1} \rfloor \, \mathbf{Y}^{(t)}_{\text{old}} = \mathbf{N}^{(t)}_l \rfloor \, \mathbf{Y}^{(t)}_{\text{old}} = \sum_{i,j,k=1}^{2^n} \left(\xi^j_{t,l} \, \eta^k_{t,old} \, \Lambda^{i,j,k} \, \mathbf{E}_i \right), \text{ and} \tag{48}$$

$$\mathbf{N}^{(t)}_l = (\mathbf{M}^{(t)}_l)^{-1} = \frac{\widetilde{\mathbf{M}}^{(t)}_l}{\mathbf{M}^{(t)}_l * \widetilde{\mathbf{M}}^{(t)}_l} = \frac{1}{\sum_{h,k=1}^{2^n} \left(\mu^h_{t,l} \, \mu^k_{t,l} \, \Upsilon^{k,k} \, \Lambda^{1,h,k} \right)} \sum_{j=1}^{2^n} \left(\mu^j_{t,l} \, \Upsilon^{j,j} \, \mathbf{E}_j \right) \tag{49}$$

is an intermediate multivector with coefficients $\{\xi^j_{t,l}\}_{j=1}^{2^n}$, whose derivatives are

$$\mathrm{J}^{j,z}_{\mathbf{N}^{(t)}_l} = \frac{\partial \xi^j_{t,l}}{\partial \chi^z} = \sum_{h,k=1}^{2^n} \Big(\Big(\mathrm{J}^{j,z}_{\mathbf{M}^{(t)}_l} \, \mu^h_{t,l} \, \mu^k_{t,l} - \mu^j_{t,l} \, \mathrm{J}^{h,z}_{\mathbf{M}^{(t)}_l} \, \mu^k_{t,l} - \mu^j_{t,l} \, \mu^h_{t,l} \, \mathrm{J}^{k,z}_{\mathbf{M}^{(t)}_l} \Big) \, \Upsilon^{j,j} \, \Upsilon^{k,k} \, \Lambda^{1,h,k} \Big). \tag{50}$$

The Jacobian of the new $\mathbf{Y}^{(t)}$ (48) is

$$\mathrm{J}^{i,z}_{\mathbf{Y}^{(t)}} = \frac{\partial \eta^i_t}{\partial \chi^z} = \sum_{j,k=1}^{2^n} \left(\left(\mathrm{J}^{j,z}_{\mathbf{N}^{(t)}_l} \, \eta^k_{t,old} + \xi^j_{t,l} \, \mathrm{J}^{k,z}_{\mathbf{Y}^{(t)}_{\text{old}}} \right) \Lambda^{i,j,k} \right). \tag{51}$$

The coefficients of $\mathbf{Y}^{(t)}_{\text{old}}$ and $\mathbf{M}^{(t)}_l$ are denoted in (48), (49), (50) and (51) by $\{\eta^k_{t,old}\}_{k=1}^{2^n}$ and $\{\mu^i_{t,l}\}_{i=1}^{2^n}$, respectively. The reverse operation is encoded by Υ, and the left contraction and the scalar product are encoded by Λ.

Tensors encoding GA operations are sparse structures. Also, multivectors encoding k-blades have at most $\binom{n}{k}$ nonzero coefficients (*i.e.,* the ones related to basis blades having dimensionality k), and rotors use only the coefficients related to even-dimensional basis blades of $\bigwedge \mathbb{R}^n$. It follows that the summations in all preceding derivations only need to be evaluated for possible nonzero multivector's coefficients and tensor's entries. Such a feature is used to reduce the computational load of obtaining the Jacobian matrices.

5.3 Mapping Procedure for $r \leq p$

When the dimensionality of an input blade is less or equal than the dimensionality of the intended type of subspace, one can take the dual (9) of the

input ($\mathbf{X}^{*}_{\langle r \rangle}$) and reference ($\mathbf{E}^{*}_{\langle p \rangle}$) blades in order to reduce the mapping procedure to the case where $r \geq p$ (Section 5.2). Thus, the dual of a random multivector variable $\underline{\mathbf{X}_{\langle r \rangle}}$ must be considered. In this case, the set $\mathcal{P}^{(m)}$ in Fig. 2 (line 1) is initialized (see (23)) with a 5-tuple having

$$\left(\overline{\mathbf{X}_{\langle r \rangle}}\right)^{*} = \mathbf{X}_{\langle r \rangle} \rfloor \mathbf{I}^{-1}_{\langle n \rangle} = \mathbf{X}_{\langle r \rangle} \rfloor \widetilde{\mathbf{I}_{\langle n \rangle}} = \sum_{i,j,k=1}^{2^n} \left(\chi^j \, \iota^k \, \Upsilon^{k,k} \, \Lambda^{i,j,k} \, \mathbf{E}_i\right) \tag{52}$$

as its first entry. The second entry of the 5-tuple is the Jacobian matrix of $\left(\overline{\mathbf{X}_{\langle r \rangle}}\right)^{*}$, whose derivatives are computed as

$$\frac{\partial \lambda^i_m}{\partial \chi^z} = \sum_{k=1}^{2^n} \left(\iota^k \, \Upsilon^{k,k} \, \Lambda^{i,i,k}\right). \tag{53}$$

The other three entries of the 5-tuple are, respectively, $\boldsymbol{K}^{(m)} = 1$, $\mathrm{J}_{\boldsymbol{K}^{(m)}} = 0$ and $\varnothing$. In (52) and (53), $\{\lambda^i\}_{i=1}^{2^n}$ denotes the coefficients of $\left(\overline{\mathbf{X}_{\langle r \rangle}}\right)^{*}$, and $\{\iota^k\}_{k=1}^{2^n}$ are the coefficients for the (constant) pseudoscalar $\mathbf{I}_{\langle n \rangle}$. It is important to notice that, when $r \leq p$, the derivatives in the Jacobian matrices computed by the mapping procedure are related to the coefficients $\{\chi^z\}_{z=1}^{2^n}$ of the direct representation of the mean input blade $\overline{\mathbf{X}_{\langle r \rangle}}$.

5.4 The Voting Procedure

The subspace detection framework presented in this paper identifies the most likely p-blades in a given dataset by performing a voting procedure using an accumulator array as the discrete representation of $\mathbb{P}^m$ (20). The mapping procedure described in Sections 5.2 and 5.3 is key for such a voting. It takes an uncertain r-blade ($\underline{\mathbf{X}_{\langle r \rangle}}$) and decomposes it as parameter vectors $\Theta^{(0)} \in \mathbb{P}^m$ and vector-valued random variables $\underline{\mathrm{a}}$. The resulting parameter vectors are computed from the expectation of $\overline{\mathbf{X}_{\langle r \rangle}}$. Thus, they characterize the most probable p-blades related to the input entry. The p-blades related to the uncertainty around the expectation of $\underline{\mathbf{X}_{\langle r \rangle}}$ are represented in an auxiliary space $\mathbb{A}^m$ by $\underline{\mathrm{a}}$.

For a given pair $(\Theta^{(0)}, \underline{\mathrm{a}})$ (Fig. 2, line 10), the number of votes to be incremented to each accumulator's bin can be computed by: (i) mapping the bin's region from $\mathbb{P}^m$ to $\mathbb{A}^m$; and, in turn, (ii) weighting the importance value ω of the input $\underline{\mathbf{X}_{\langle r \rangle}}$ by the probability of a related p-blade be in the mapped region. Ideally, such a probability should be computed in $\mathbb{A}^m$ as the hypervolume under the portion of the multivariate "bell" curve contained by the mapped region. However, rectangular regions in the actual parameter space (Fig. 1a) map to warped regions in the auxiliary parameter space (Fig. 1b). It is a challenging and computationally intensive task to evaluate the probability in such warped regions [8]. Our solution to this problem involves defining,

for each bin in $\mathbb{A}^m$, a representative box aligned to the eigenvectors of the covariance matrix Σ_{a} of $\underline{\mathrm{a}}$. As depicted in Fig. 1c, in the space defined by the orthogonal eigenvectors of Σ_{a}, the eigenvalues represent the variances of an axis-aligned Gaussian distribution (in Fig. 1c, the unit eigenvectors are scaled by the eigenvalues), and the covariances are equal to zero. Hence, the resulting probability can be efficiently computed as the product of the probabilities of the intervals defined by the representative box.

The representative box of a bin having coordinates Θ_{bin} is built from points $\{\Theta^i_{\mathrm{face}}\}_{i=1}^{2\,m}$ placed at the center of bin's faces in the parameter space (Fig. 1a). By using Δ_{θ^t} radians as step in the linear discretization of the t-th dimension of the parameter space, the center of the faces are computed as

$$\Theta^i_{\mathrm{face}} = \Theta_{\mathrm{bin}} + \Theta_{\mathrm{offset}_i} \quad \forall\, i \in \{1, 2, \cdots, 2\,m\},$$

where

$$\Delta_{\mathrm{offset}_i} = \begin{cases} (0, \cdots, -\Delta_{\theta^{\lfloor \frac{i+1}{2} \rfloor}}, \cdots, 0) & \text{for odd } i \\ (0, \cdots, +\Delta_{\theta^{\lfloor \frac{i+1}{2} \rfloor}}, \cdots, 0) & \text{for even } i \end{cases}$$

is the translation vector from the center of a bin to the center of a bin's face. $\lfloor \square \rfloor$ denotes the floor function. Each point Θ^i_{face} is mapped from $\mathbb{P}^m$ to $\mathbb{A}^m$ (Fig. 1b) using the following procedure:

(1) The vectors spanning the reference blade $\mathbf{E}_{\langle p \rangle}$ (36) are transformed by the rotor $\boldsymbol{W}\,\boldsymbol{T}^i_{\mathrm{face}}$, where $\boldsymbol{T}^i_{\mathrm{face}}$ is computed according to (17) using the coordinates (rotation angles) of Θ^i_{face}. The rotor $\boldsymbol{W}$ is given by (31) for current pair $(\Theta^{(0)}, \underline{\mathrm{a}})$;
(2) The vectors resulting from step 1 are used to define a $p \times n$ matrix representation the subspace related to Θ^i_{face};
(3) The location of Θ^i_{face} in $\mathbb{A}^m$ (denoted by points $\mathrm{a}^i_{\mathrm{face}}$ in Fig. 1b) is retrieved from the row reduced echelon form of the matrix build in step 2.

Once the points $\{\mathrm{a}^i_{\mathrm{face}}\}_{i=1}^{2\,m}$ are known, each $\mathrm{a}^i_{\mathrm{face}}$ is transformed to the basis defined by the (orthonormal) eigenvectors of Σ_{a} (Fig. 1c). The transformation is achieved by premultiplying $\mathrm{a}^i_{\mathrm{face}}$ by the transpose of a matrix having such eigenvectors as its columns. The eigenvectors-aligned bounding box including $\{\mathrm{a}^i_{\mathrm{face}}\}_{i=1}^{2\,m}$ is the representative box of the bin's region in $\mathbb{A}^m$. Each dimension of such a box defines an interval $[\min_t, \max_t]$ in one of the axis related to an eigenvector (see Fig. 1c). The number of votes to be incremented in the bin is

$$votes = \omega \prod_{t=1}^{m} \left(\Phi\left(\frac{\max_t}{\sigma_t}\right) - \Phi\left(\frac{\min_t}{\sigma_t}\right) \right), \tag{54}$$

where σ_t is the square root of the eigenvalue related to the t-th eigenvector of Σ_{a}, ω is the importance of the input entry $\mathbf{X}_{\langle r \rangle}$, and Φ is the cumulative distribution function of the standard normal distribution [8].

As depicted in Fig. 1, one needs to compute votes only for the bins intersecting the footprint of the Gaussian distribution in $\mathbb{A}^m$. It is because the number of votes for the bins beyond three standard deviations (the ellipses in Figs. 1b and 1c) are negligible. Intuitively, (54) can be evaluated starting at the bin containing $\Theta^{(0)}$ (*i.e.,* the parameter vector related to the expectation $\bar{\mathrm{a}}$) and moving outwards (in parameter space, Fig. 1a) in a flood fill fashion, until the returned values are lower than a threshold ϵ_{votes}. For the examples presented in this paper, ϵ_{votes} was set to 10^{-6} (a value defined empirically).

It is important to notice that the vector-valued random variable $\underline{\mathrm{a}}$ defines a k-dimensional multivariate Gaussian distribution in $\mathbb{A}^m$, for $0 \leq k \leq m$. This means that its distribution of uncertainty may have arbitrary dimensionality in $\mathbb{A}^m$, and hence also in $\mathbb{P}^m$. The dimensionality of the Gaussian distribution (k) will be equal to m if and only if two conditions are satisfied:

(1) A given input entry $\underline{\mathbf{X}_{\langle r \rangle}}$ maps to only one pair $(\Theta^{(0)}, \underline{\mathrm{a}})$ (*i.e.,* all parameter values were computed and returned by Fig. 3, line 13); and
(2) The uncertainty in $\underline{\mathbf{X}_{\langle r \rangle}}$ does not restrict the distribution of $\underline{\mathrm{a}}$ to a linear subspace in $\mathbb{A}^m$ having a dimensionality smaller than m.

When condition (1) fails then the distribution of $\underline{\mathrm{a}}$ will have at most $k = (m - s)$ dimensions, where s is the number of arbitrated parameters (Fig. 3, line 8). In such a case, the distributions of votes centered at the coordinates of each parameter vector $\Theta^{(0)} \in \mathbb{P}^m$ define (warped) parallel profiles in $\mathbb{P}^m$.

The distribution of $\underline{\mathrm{a}}$ also loses dimensions when condition (2) fails. It happens when there is no uncertainty in some of the degrees of freedom of the input entry $\underline{\mathbf{X}_{\langle r \rangle}}$. In an extreme case, the input blade does not have uncertainty at all. In this situation, the extended voting scheme presented in this paper reduces to the approach presented in [5].

6 Validation and Discussion

The prerequisites for using first-order error propagation is to have Gaussian distributed uncertainty in the input data entries and also in the resulting mappings [8,10]. The first condition is intrinsic to the input data. We evaluate our approach by carrying three sets of experiments in order to assert the second condition for the proposed mapping procedure.

The first two sets of experiments (Section 6.1 and 6.2) aim to verify whether the skewness and kurtosis of the resulting vote distributions in the the auxiliary parameter space characterize Gaussian distributions. Such experiments are based on sampling-based procedures. The third set of experiments (Sec-

tion 6.3) verifies the equivalence between the sampling-based distributions computed in Section 6.2 and the distributions computed using our procedure. Sections 6.4 and 6.5 provide results on real datasets.

6.1 Verifying Gaussian Distributions by Checking Skewness and Kurtosis

Each experiment of the set assumes a detection case (*i.e.,* r-dimensional subspaces in a n-dimensional space), and one input uncertain r-blade having a given covariance matrix. The covariance matrices were defined by rotating the principal axes of a reference axis-aligned Gaussian distribution according to its degrees of freedom (*i.e.,* the $k\,(k-1)/2$ generalized Euler angles [37] characterizing the orientation of a k-dimensional Gaussian distribution). In this way, the experiments cover a wide range of settings for Gaussian distributions. For each experiment, a set of 1,000 random samples of the input uncertain blade was generated according to the assumed Gaussian distribution. The uncertain blade was then mapped to $\mathbb{A}^m$ with the procedure presented in Sections 5.2 and 5.3. In turn, the samples related to the uncertain blade where mapped to the same auxiliary space $\mathbb{A}^m$. Recall from Section 5.2 that $\mathbb{A}^m$ is defined in such a way that the expectation of the uncertain blade is at its origin. Thus, the samples are mapped to points around the origin of $\mathbb{A}^m$. This distribution of points must be Gaussian in order to validate the assumption that first-order error propagation can be used to predict a distribution of uncertainty in $\mathbb{A}^m$. The distribution of points was verified with the statistical hypothesis test proposed by Mardia [38], which computes a P-value for the skewness, and another one for the kurtosis of a given set of points. If these P-values are greater than or equal to the significance level $\alpha = 0.05$, then the results are statistically not-significant (*i.e.,* the distribution of points is probably Gaussian because there is only a 5% chance that the expected measures of skewness and kurtosis have happened by coincidence). Otherwise, for P-values lower that α, there is a reasonable chance of the distribution being non-Gaussian.

Tables 1 and 2 summarize the P-values computed, respectively, for skewness and kurtosis in the first and second sets of experiments. These experiments (1,530 altogether) are grouped by table entries according to a detection case (see the n and p values on the column headers), and a dimensionality of input uncertain blades (see the r value for each row). Each table entry shows a relative frequency histogram of P-values (the abscissa, in logarithmic scale) for the respective detection/input case. Notice that almost all computed P-values had fallen in the bins to the right of (and including) the significance level $\alpha = 0.05$ (denoted by the dashed lines). These results suggest that samples mapped to $\mathbb{A}^m$ define Gaussian distributions for different values of variance and covariance in the input uncertain blades. However, some of the P-values presented in Tables 1 and 2 are smaller than α. Thus, it is necessary to verify whether

Table 1
Relative frequency histograms of P-values computed for the skewness of distributions of points in the auxiliary space $\mathbb{A}^m$. The most frequent P-values are greater than or equal to the significance level $\alpha = 0.05$ (the dashed lines), indicating that the distributions of points have the skewness of a Gaussian distribution.

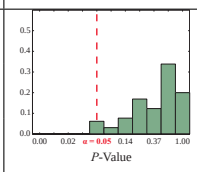
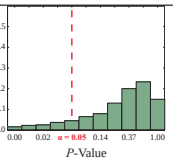
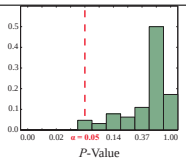
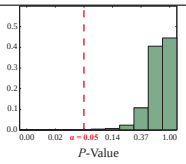
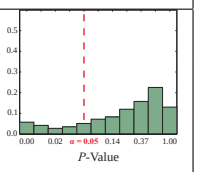
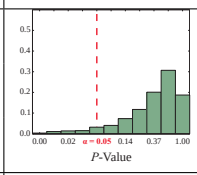
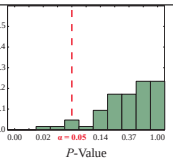
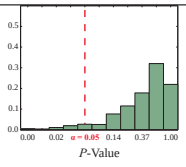
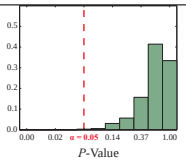
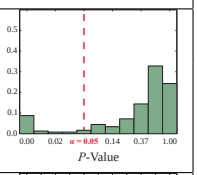
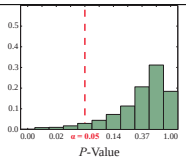
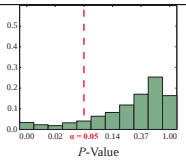
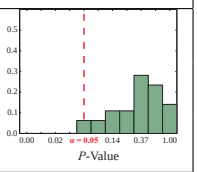
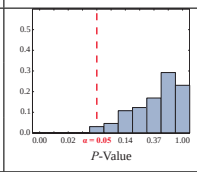
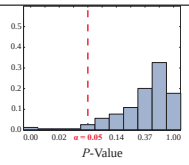
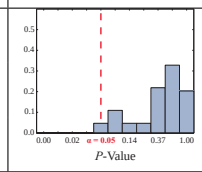
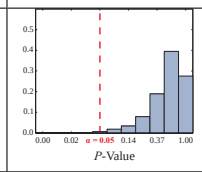
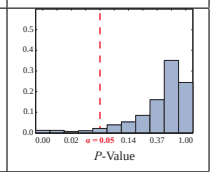
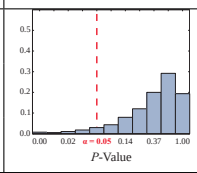
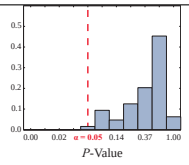
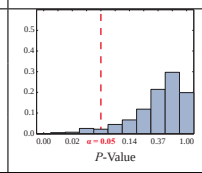
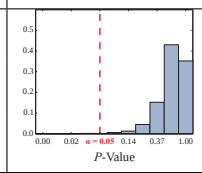
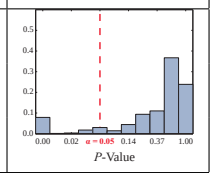
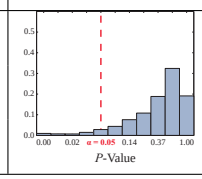
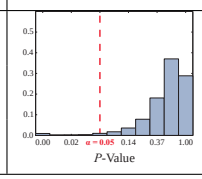
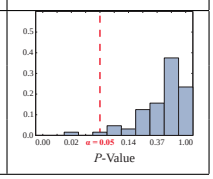
	$n = 3,\ p = 1$ $(m = 2)$	$n = 3,\ p = 2$ $(m = 2)$	$n = 4,\ p = 1$ $(m = 3)$	$n = 4,\ p = 2$ $(m = 4)$	$n = 4,\ p = 3$ $(m = 3)$
$r = 1$					
$r = 2$					
$r = 3$	—	—			

Table 2
Relative frequency histograms of P-values computed for the kurtosis of distributions of points in the auxiliary space $\mathbb{A}^m$. It shows that distributions of points related to almost all the experiments have the kurtosis of a Gaussian distribution.

	$n = 3,\ p = 1$ $(m = 2)$	$n = 3,\ p = 2$ $(m = 2)$	$n = 4,\ p = 1$ $(m = 3)$	$n = 4,\ p = 2$ $(m = 4)$	$n = 4,\ p = 3$ $(m = 3)$
$r = 1$					
$r = 2$					
$r = 3$	—	—			

these P-values are related to some sensitivity of Mardia's procedure, or to some bounding limit for input uncertainty. To verify these conditions another set of statistical experiments was carried out to observe how the "quality" of skewness and kurtosis change as the uncertainty of input blades changes.

In the second set of experiments it was assumed the detection of straight lines in the 2-dimensional homogeneous MOG (refer to [9] for details about the MOG). Thus, $p = 2$ and $n = 2 + 1$, leading to $m = 2\,(3 - 2) = 2$. Initially, a set of 1,225 blades was created by choosing 35×35 parameter vectors linearly distributed over the parameter space $\mathbb{P}^2$ for 2-blades in $\bigwedge \mathbb{R}^{2+1}$. Through (16), each one of the parameter vectors is related to a blade, which is regarded here as the expectation ($\overline{\mathbf{X}_{\langle 2\rangle}}$) of a random multivector variable $\underline{\mathbf{X}_{\langle 2\rangle}}$. By converting $\overline{\mathbf{X}_{\langle 2\rangle}}$ to the parameterization defined by the normal equation of the line

$$x\,\cos\phi + y\,\sin\phi - \rho = 0, \tag{55}$$

and by assigning standard deviations σ_ρ and σ_ϕ to ρ and ϕ, respectively, one can define the covariance matrix of $\underline{\mathbf{X}_{\langle 2\rangle}}$ from σ_ρ and σ_ϕ. This way, it is possible to verify changes on the skewness and kurtosis of distributions of samples mapped to $\mathbb{A}^2$ as an effect of changing the uncertainty in parameters having a clear geometrical interpretation: ρ defines the distance from the origin point to the line, and ϕ is the angle between the x-axis and the normal to the line. The mean normal parameters of the line are computed from $\overline{\mathbf{X}_{\langle 2\rangle}}$ as

$$\overline{\rho} = \|\mathbf{s}\| \quad \text{and} \quad \overline{\phi} = \tan^{-1}\left(\frac{\mathbf{n} * \mathbf{e}_2}{\mathbf{n} * \mathbf{e}_1}\right), \tag{56}$$

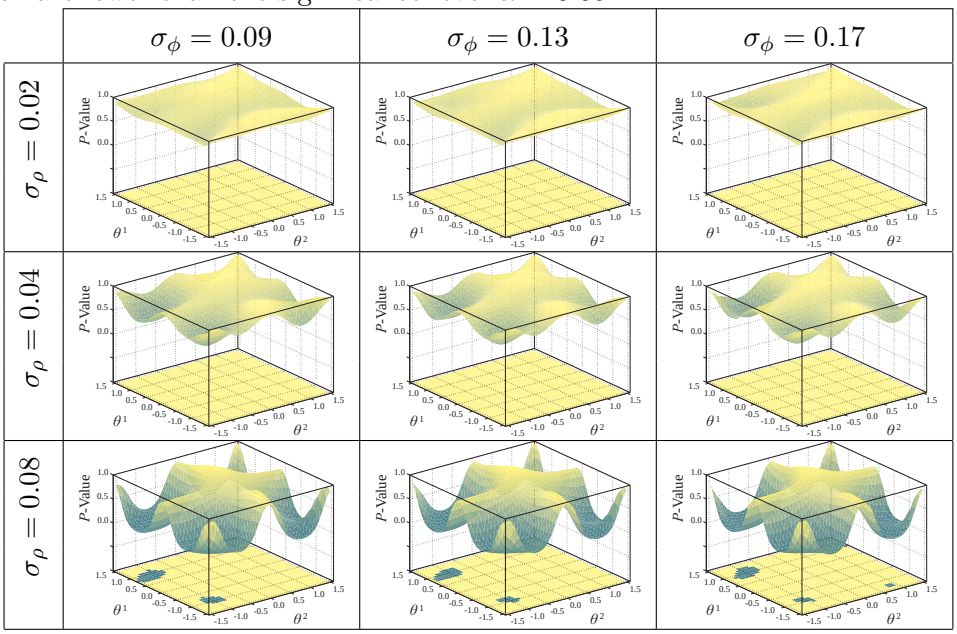
where $\mathbf{s} = \left(\mathbf{e}_0^{-1} \rfloor \left(\mathbf{e}_0 \wedge \overline{\mathbf{X}_{\langle 2\rangle}}\right)\right) / \mathbf{d}$ is the support vector of the line, $\mathbf{d} = \mathbf{e}_0^{-1} \rfloor \overline{\mathbf{X}_{\langle 2\rangle}}$ is its direction, $\mathbf{n} = \mathbf{e}_0^{-1} \rfloor \left(\mathbf{e}_0 \wedge \left(\overline{\mathbf{X}_{\langle 2\rangle}}\right)^*\right)$ is the normal to the line, with the condition that $\mathbf{s} * \mathbf{n} \geq 0$, and $*$ denotes de scalar product (4). In the homogeneous MOG, the basis vector $\mathbf{e}_0$ is interpreted as the point at the origin. Vectors $\mathbf{e}_1$ and $\mathbf{e}_2$ are related to x and y in (55), respectively. In the homogenous MOG, we assume Euclidean metric for the basis vectors, *i.e.,* $\mathbf{e}_i \cdot \mathbf{e}_j$ is *one* for $i = j$ and *zero* otherwise, where $\cdot$ denotes the inner product of vectors. It is important to comment that, in an implementation, one should evaluate the arctangent in (56) with the function ATAN2 in order to make $\overline{\phi} \in [-\pi, \pi)$. In such a case, one should assume $\overline{\rho} \in [0, \infty)$. ATAN2 is available in many programming languages. The expectation of $\underline{\mathbf{X}_{\langle 2\rangle}}$ can be computed from $\overline{\rho}$ and $\overline{\phi}$ (56) as

$$\overline{\mathbf{X}_{\langle 2\rangle}} = \left(\overline{\rho}\,\left(\mathbf{e}_0^{-1}\right) - \cos\left(\overline{\phi}\right)\mathbf{e}_1 - \sin\left(\overline{\phi}\right)\mathbf{e}_2\right)^{-*}. \tag{57}$$

It follows that the covariance matrix of $\underline{\mathbf{X}_{\langle 2\rangle}}$ is defined as

$$\Sigma_{\mathbf{X}_{\langle \mathrm{r}\rangle}} = \mathrm{J}_{\mathbf{X}_{\langle r\rangle}} \begin{pmatrix} \sigma_\rho^2 & 0 \\ 0 & \sigma_\phi^2 \end{pmatrix} \mathrm{J}^{\mathrm{T}}_{\mathbf{X}_{\langle r\rangle}}, \tag{58}$$

Table 3
P-values computed for the skewness of distributions of points in the auxiliary spaces $\mathbb{A}^2$ related to uncertain 2-blades. The heights of the surfaces represent the P-values, while the dark spots on the plane at the bottom of the charts denote those P-values which are lower than the significance level $\alpha = 0.05$.



where the derivatives in the Jacobian matrix $\mathrm{J}_{\mathbf{X}_{\langle r \rangle}}$ are computed as

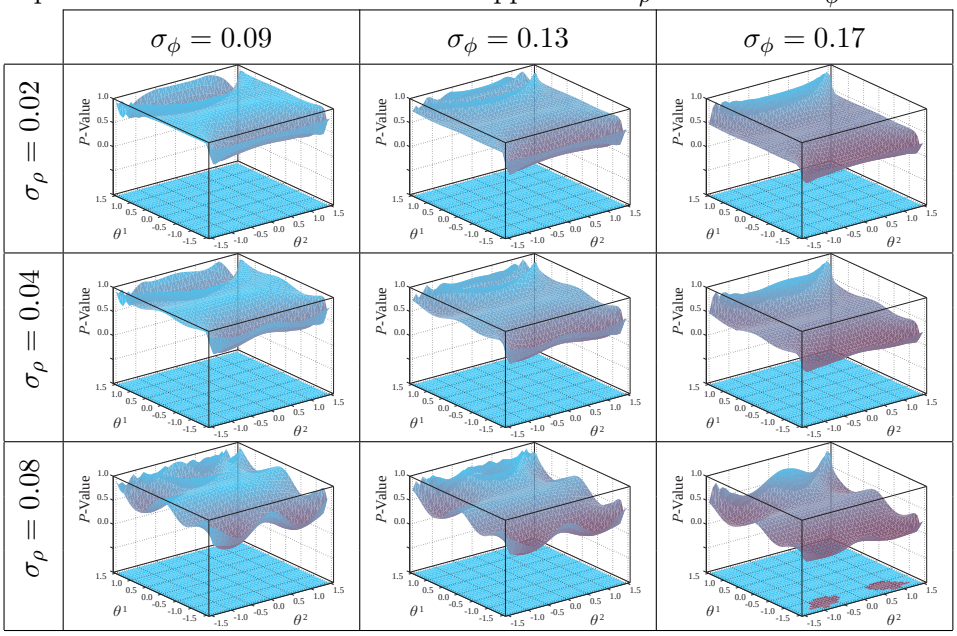
$$\mathrm{J}^{i,1}_{\mathbf{X}_{\langle r \rangle}} = \frac{\partial \chi^i}{\partial \rho} = \sum_{k=1}^{2^n} - \left(\delta_0^k \, \Lambda^{i,1,k} \right), \text{ and} \tag{59a}$$

$$\mathrm{J}^{i,2}_{\mathbf{X}_{\langle r \rangle}} = \frac{\partial \chi^i}{\partial \phi} = \sin\left(\overline{\phi}\right) \sum_{k=1}^{2^n} \left(\delta_1^k \, \Lambda^{i,1,k} \right) - \cos\left(\overline{\phi}\right) \sum_{k=1}^{2^n} \left(\delta_2^k \, \Lambda^{i,1,k} \right). \tag{59b}$$

In (59), the coefficients of the constant subspaces $\left(\mathbf{e}_0^{-1}\right)^{-*}$, $\mathbf{e}_1^{-*}$, and $\mathbf{e}_2^{-*}$ are denoted by $\{\delta_0^k\}_{k=1}^{2^n}$, $\{\delta_1^k\}_{k=1}^{2^n}$, and $\{\delta_2^k\}_{k=1}^{2^n}$, respectively. Recall form Section 3.1 that $\square^{-*}$ denotes the undual operation (10).

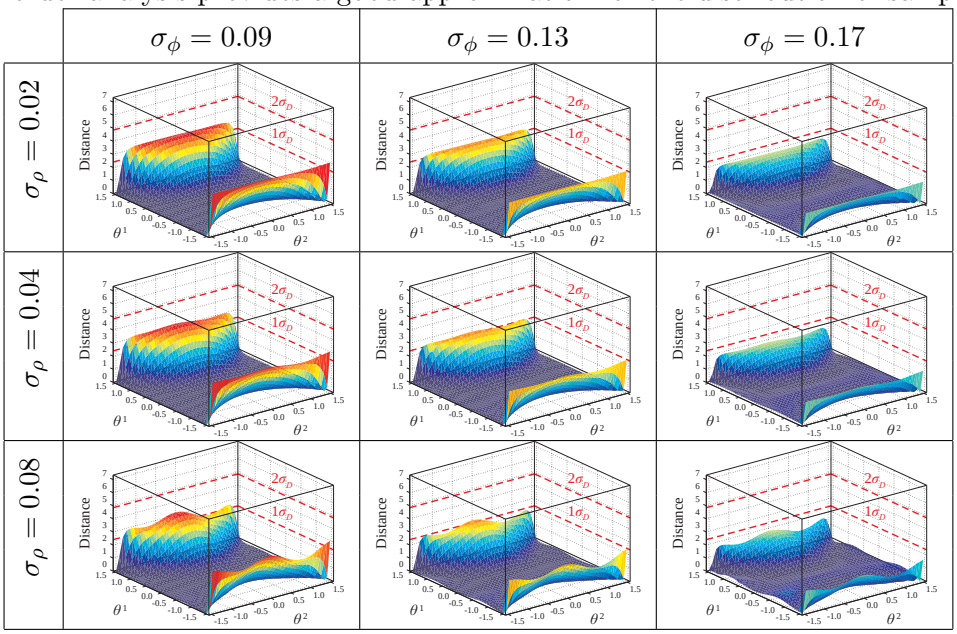
After the 1,225 mean blades $\overline{\mathbf{X}_{\langle 2 \rangle}}$ had been written as a function of their mean normal parameters $\overline{\rho}$ and $\overline{\phi}$ (57), a value for σ_ρ and another one for σ_ϕ was chosen, and the covariance matrices $\Sigma_{\mathbf{X}_{\langle 2 \rangle}}$ were computed using (58). Then, two canonical sets with 500 random real values each were generated following a standard Normal distribution. A copy of these canonical samples was assigned to each input uncertain blade $\mathbf{X}_{\langle 2 \rangle}$, and converted to the Gaussian distribution of its respective $\underline{\rho}$ and $\underline{\phi}$ variables. The use of canonical samples is important because they make possible the comparison of skewness and kurtosis of distributions related to different input uncertain subspaces $\mathbf{X}_{\langle 2 \rangle}$. Finally, as well as in the first set of experiments presented in Section 6.1, each uncertain blade was mapped to $\mathbb{A}^2$. In turn, its respective samples where mapped to the same auxiliary space, and the normality of the distribution of points

Table 4
P-values computed for the kurtosis of distributions of points in the auxiliary spaces $\mathbb{A}^2$ related to uncertain 2-blades. Kurtosis is less sensitive to variations in the input uncertain than skewness. The first distributions having kurtosis different from the one expected for a Gaussian distribution appears at $\sigma_\rho = 0.08$ and $\sigma_\phi = 0.17$.



in $\mathbb{A}^2$ was verified with Mardia's procedure [38]. Tables 3 and 4 present the P-values produced for, respectively, the skewness and kurtosis of the distributions of points in $\mathbb{A}^n$. The table entries group the experiments according to the values assumed for σ_ρ (rows) and σ_ϕ (columns). The P-values are shown as heights of a surface on a 3-dimensional visualization of the parameter space $\mathbb{P}^2$ (*i.e.,* the height at the parameter vector related to a $\overline{\mathbf{X}_{\langle 2\rangle}}$ is the P-value estimated from the samples of the respective $\underline{\rho}$ and $\underline{\phi}$ variables). The P-values lower than $\alpha = 0.05$ are distinguished as darker colors on the plane at the bottom of the charts. Notice that the samples mapped to $\mathbb{A}^2$ define Gaussian distributions even for larger values of σ_ρ and σ_ϕ (*i.e.,* there are a few small dark spots in Tables 3 and 4). The first asymmetric distributions only appeared for $\sigma_\rho = 0.08$ (see the dark spots in the third row of Table 3), while the tails of a few distributions differ from a Gaussian distribution only for $\sigma_\rho = 0.08$ and $\sigma_\phi = 0.17$ (see the two dark spots in the last entry of Table 4). By assuming an image with coordinates in the $[-1, +1] \times [-1, +1]$ range and the homogeneous MOG, it follows that the higher uncertainty values ($\sigma_\rho = 0.08$ and $\sigma_\phi = 0.17$) define a confidence interval of ± 0.24 units for the distance from the center of the image to a given line (*i.e.,* almost 1/4 of image's size), and a confidence interval of ± 0.51 radians for the direction of the line (*i.e.,* almost $\pi/3$ radians). These results show that, as far the uncertainty are kept bellow these limits, input random multivector variables define Gaussian distributions in $\mathbb{A}^2$. Therefore, P-values smaller than $\alpha = 0.05$ observed in Tables 1 and 2 can be related to very high uncertainty in the input blades.

Table 5
Measure of the similarity between the covariance matrices of Gaussian distributions approximated with first-order error propagation analysis and from points obtained through a sampling-based approach while mapping uncertain blades $\underline{\mathbf{X}_{\langle 2 \rangle}}$ to $\mathbb{A}^2$. The heights of the surfaces represent the distance between the two sets of convariances. Notice that the computed distances are lower than $1.5\sigma_D$. These results show that first-order analysis provides a good approximation for the distribution of samples.



6.3 Validating the Propagated Uncertainties

From Tables 1 to 4 it is possible to conclude that error propagation can be used to predict the uncertainty of blades mapped to $\mathbb{A}^m$. But it is also important to verify if first-order analysis is sufficient to approximate the expected Gaussian distributions. Such an analysis was performed by comparing the covariance matrix computed for random samples mapped to $\mathbb{A}^m$ with the covariance matrix estimated by processing the respective uncertain blade $\underline{\mathbf{X}_{\langle r \rangle}}$ with the proposed mapping procedure (Sections 5.2 and 5.3). These two covariance matrices were compared with the distance function described by Bogner [39]. Such a function receives as input a true (or estimated) distribution (*i.e.,* in this case, the covariance matrix computed with first-order error propagation) and a set of observations (*i.e,* the points resulting from mapping random samples to $\mathbb{A}^m$). Bogner's function [39] returns a real value D as a measure of the distance between the two distributions and the expected theoretical variance for the distance, computed from the size of the matrix. This variance can be used to interpret the resulting values as a similarity measurement.

Table 5 shows the distances computed for the examples depicted in Tables 3 and 4. Here, the heights of the surfaces represent the distance D computed by Bogner's procedure [39]. For these examples, the theoretical variance is

$\sigma_D^2 = m\,(m+1) = 2\,(2+1) = 6$, where m is the dimensionality of the distribution (in this case, the same as the dimensionality of $\mathbb{A}^2$). The charts in Table 5 denote one and two standard deviations of D by dashed red lines. Notice that all the distances are below $1.5\,\sigma_D$. These results show that the Gaussian distribution estimated with first-order error propagation is consistent with the Gaussian distribution of samples in $\mathbb{A}^2$. Further investigation of why the propagated distribution fits better the samples for larger values of σ_ρ and σ_ϕ is an interesting direction for future work. We conjecture that differences between the expected and the estimated distributions are relative to the rate between the small amount of inaccuracy introduced by the propagation process and the area occupied by the mapped samples. Since the inaccuracy is small, lack of fit is observed only in narrow uncertainty.

6.4 Detection Results on Real Datasets

This section illustrates the use of our technique for feature detection on real datasets. These examples include the detection of lines in an image obtained from electron backscatter diffraction, line detection in a photograph, and the detection of circles in a clonogenic assay image. Fig. 4a shows an electron backscatter diffraction image (445×445 pixels) taken from a particle of wulfenite ($PbMoO_4$). The straight line detection (Fig. 4b) was performed using the 2-dimensional homogeneous MOG. Thus, $p = 2$ and $n = 2 + 1 = 3$, leading to a parameter space with $m = 2\,(3-2) = 2$ dimensions. The 395,248 edge pixels on Fig. 4b where computed using Canny edge detector. The edge pixels were used as input uncertain blades characterized as uncertain vectors ($r = 1$). In this example, the standard deviation for coordinates of a given edge pixel is $2/(445\,\sqrt{12})$, where 2 is the size of the image after normalizing its coordinates to the $[-1, +1]$ range, 445 is the number of pixels in each dimension of the image, and 12 come from the second central moment of a pixel with unit side length. The discretization step for defining the accumulator array was set to $\pi/900$, and the importance value of each input is the magnitude of the gradient computed by the edge detector.

Fig. 4d shows the straight line detection in a given chess board image with 512×512 pixels of resolution. As in Fig. 4b, the procedure was performed using the 2-dimensional homogeneous MOG, leading to $m = 2$. However, in this example, the input entries were the uncertain 2-blades encoding uncertain straight lines computed from the edge pixels of the image and their gradient vector directions. The standard deviation for coordinates of a given edge pixel is $2/(512\,\sqrt{12})$. 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 discretization step for defining the accumulator array was set to $\pi/1800$, and the importance value of each input is $\omega = 1$.

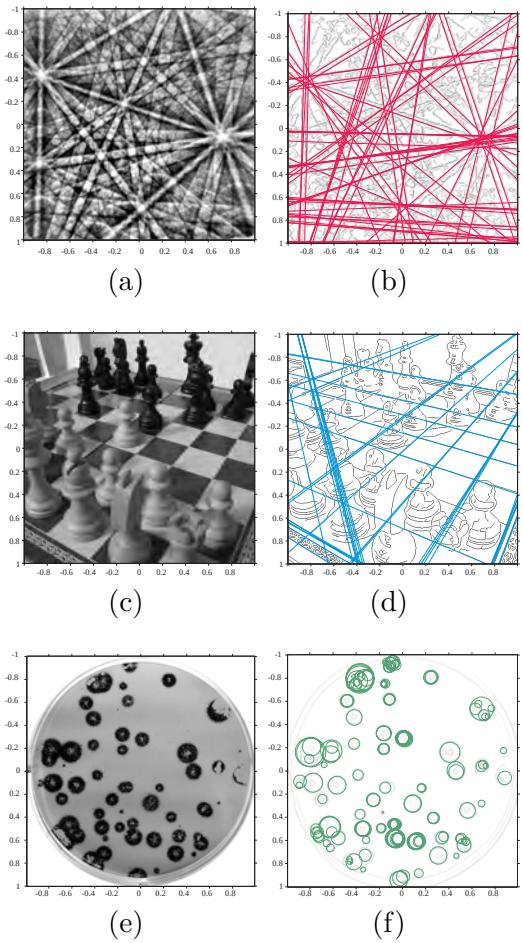


Fig. 4. Detection results (right) on real images (left): (a-b) Detection of the straight lines that best fit the edge pixels of a given electron backscatter diffraction image (left); (c-d) the most relevant detected lines in a given chess board image; and (e-f) detection of circles that best fit a given clonogenic assay image.

Fig. 4e shows the gray image (529×529 pixels) of MDCK-SIAT1 cells infected with A/Memphis/14/96-M (H1N1). The circle detection (Fig. 4f) was performed using the 2-dimensional conformal MOG. In such a case, $p = 3$ and $n = 2 + 2 = 4$, leading to $m = 3\,(4 - 3) = 3$ parameters. The blades used as input encode uncertain tangent directions. 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 and their gradient vector directions. As in Fig. 4b, ω is the magnitude of gradient directions. In order to make the irregular imaged structures become more circular, the image in Fig. 4f was convolved with a pillbox filter of radius 5 pixels before Canny edge detection. Retrieved circles having radius larger 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.

For the example depicted in Fig. 4f, the standard deviation for coordinates of a given edge pixel is $2/(529\sqrt{12})$. 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.

6.5 Discussion and Limitations

A clear advantage of the voting procedure based on first-order error propagation over a sampling-based approach is the reduced computational load of the former. This is because, with first-order analysis, only one uncertain blade needs to be processed per entry of the input dataset. With a sampling-based voting procedure, on the other hand, hundreds of samples must be generated and processed in order to properly represent the uncertainty on a given input entry. Another advantage is the possibility of spreading smoother distributions of values over the bins of the accumulator array. Such a feature improves the identification of local maxima in the resulting map of votes by reducing the occurrence of spurious peaks of votes. Fig. 5 presents a comparison between the accumulator array produced for detecting straight lines with the technique described in this paper (Fig. 5b) and the sampling-based voting using repeated invocations of the technique described in [5] (Fig. 5e). These results are consistent with the observations made by van Veen and Groem in [6] for detecting straight lines in images using a weighted HT. Our approach, however, can be used in the detection of arbitrary analytical uncertain shapes.

Figs. 5c and 5f show a detailed view of the highlighted portions in Figs. 5b and 5e, respectively. Notice the smoother transitions of votes produced by the error-propagation-based technique. In this example, the input dataset is comprised by 15,605 uncertain blades encoding straight lines in the 2-dimensional homogeneous MOG. The input 2-blades were computed from the edge pixels of the image (Figs. 5a and 5d) and their gradient vector directions. The standard deviation for coordinates of a given 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 image, and 12 comes from the second central moment of a pixel with unit side length. 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 arrays where obtained as the linear discretization of $\mathbb{P}^2$, 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 sampling-based voting procedure, each one of the 15,605 uncertain 2-blades was represented by 160 random samples. Each such sample was computed from a pixel location and a gradient vector

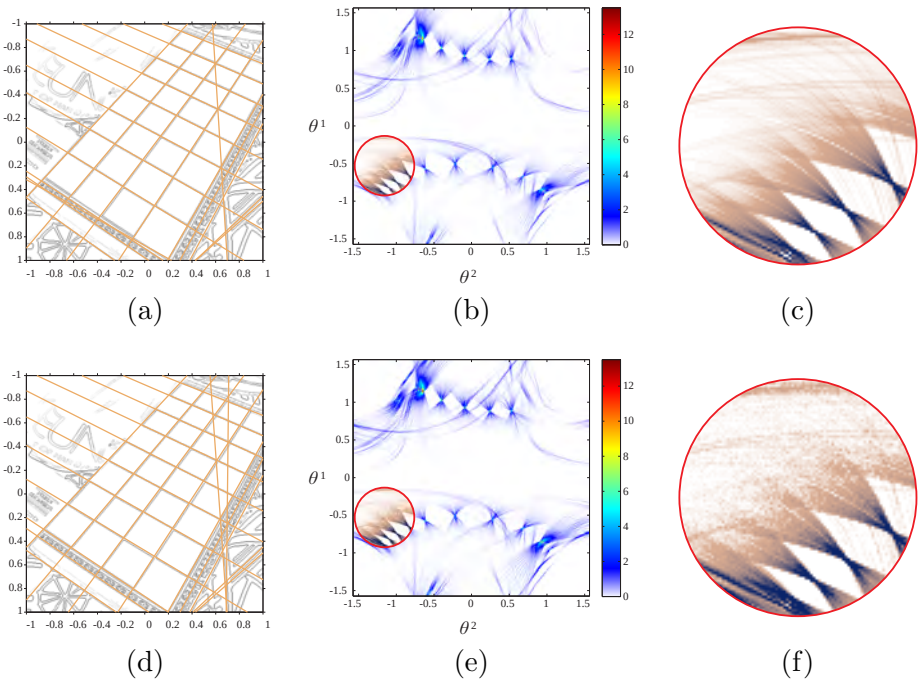


Fig. 5. The 22 most-relevant detected lines obtained using: (a) our first-order error-propagation-based voting scheme, and (d) a sampling-based approach. The accumulator arrays produced for (a) and (d) are shown in (b) and (e), respectively. The highlighted portions are presented in detail by (c) and (f). Notice that (c) presents smoother distributions of votes than (f). As a result, first-order error propagation is less prone to the detection of spurious subspaces.

taken from the distributions described above.

In our experiments it was observed that the approximation assumed for computing the number of votes to be incremented to a given bin of the accumulator array (Fig. 1c) affects the resulting distributions of votes by a scaling factor slightly bigger than one. Also, it was observed a small shift (up to one bin) in the location of some peaks. Such a displacement is not uniform in the whole parameter space. It is important to note that such a scaling and displacement do not affect the quality of detections.

The approach proposed in this paper is limited to blades with Gaussian-distributed uncertainty. This is because first-order error analysis is used to propagate the uncertainty of each input element throughout the computations. The treatment of non-Gaussian distribution would require a more cumbersome error-propagation scheme. For instance, some approach based on Monte Carlo.

The construction of the accumulator array requires some discretization criterion. It has been shown by van Veen and Groen [6], and by Lam *et al.* [7], that the linear discretization of the parameter space in conventional HTs (and hence in the generalization describe in [5]), with improper choice of discretization

values, may lead to unsharp or multiple peaks of votes when the discretization step is, respectively, too big or too small. In our experiments, the linear discretization step was chosen considering the amount of memory available in the computational system (see [5] for a discussion). However, by properly weighting the votes, our uncertain-based voting scheme helps to reduce the side effects of using small discretization values (Fig. 5c). Some authors suggest the use of non-linear discretization of the parameters for solving the tradeoff between accuracy and memory usage in specific detection cases [1]. The definition of general non-linear discretization criteria for detecting subspaces having arbitrary practical interpretations is an open problem.

7 Summary and Future Work

This paper presented a mapping and a voting procedure for subspace detection in the presence of input with Gaussian-distributed uncertainty. The mapping scheme uses first-order error propagation to transfer the uncertainty from input data to an auxiliary space defined as the open affine covering $\mathbb{A}^m$ for the Grassmannian $G(p,n)$. The propagated uncertainty defines a Gaussian profile in $\mathbb{A}^m$, which is mapped by the voting procedure to the actual parameter space $\mathbb{P}^m$ as a non-Gaussian distribution of votes.

The scope of application of our solution is guaranteed by the generality of the framework it builds upon [5]. Thus, it can be applied, without any changes, to the detection of all kinds of data alignments that can be represented as linear subspaces in datasets comprised by exact or uncertain data. In the case of uncertain data, our approach eliminates the computationally-expensive task of sampling the distribution of uncertainty of each input entry and mapping the samples to the parameter space. It is a general and valuable tool for detecting alignments in real datasets, since uncertainty is something intrinsic to data collected by all measurement devices. The generalization and integration of the clustering and culling procedures of the KHT [30] to the proposed framework is a promising direction for future exploration. We are also investigating ways to store the voting map of uncertain data by using data structures that are less memory consuming than the array of accumulators. With these improvements we hope to make the proposed framework suitable for real time applications.

Acknowledgments

This work was sponsored by CNPq-Brazil grants 142627/2007-0, 308936/2010-8 and 482271/2012-4, FAPERGS PQG 10/1322-0, and FAPERJ E-26/112.468/2012. We thank the reviewers for their comments and insightful suggestions.

References

[1] P. V. C. Hough, Machine analysis of bubble chamber pictures, in: Proc. of the International Conference on High Energy Accelerators and Instrum., 1959.

[2] 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.

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

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

[5] L. A. F. Fernandes, M. M. Oliveira, A general framework for subspace detection in unordered multidimensional data, Pattern Rec. 45 (9) (2012) 3566–3579.

[6] T. van Veen, F. Groen, Discretization errors in the Hough transform, Pattern Rec. 14 (1-6) (1981) 137–145.

[7] W. C. Y. Lam, L. T. S. Lam, K. S. Y. Yuen, D. N. K. Leung, An analysis on quantizing the Hough space, Pattern Rec. Lett. 15 (11) (1994) 1127–1135.

[8] G. Cowan, Statistical data analysis, Oxford University Press, 1998.

[9] L. A. F. Fernandes, M. M. Oliveira, Geometric algebra: a powerful tool for solving geometric problems in visual computing, in: Tutorials of the Brazilian Symposium on Computer Graphics and Image Processing, 2009, pp. 17–30.

[10] C. Perwass, Geometric algebra with applications in engineering, Springer Publishing Company, 2009.

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

[12] G. Sommer, Geometric computing with Clifford algebras: theoretical foundations and applications in computer vision and robotics, Springer, 2001.

[13] D. Danielson, Vectors and tensors in engineering and physics, 2nd Edition, Westview Press, 2003.

[14] X. Jiang, Linear subspace learning-based dimensionality reduction, IEEE Signal Process. Mag. 28 (2) (2011) 16–26.

[15] D. Tao, X. Li, X. Wu, S. J. Maybank, General averaged divergence analysis, in: Proc. of the 17th IEEE International Conference on Data Mining, 2007, pp. 302–311.

[16] D. Tao, X. Li, X. Wu, S. J. Maybank, Geometric mean for subspace selection, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2) (2009) 260–274.

[17] X. He, P. Niyogi, Locality preserving projections, in: S. Thrun, L. Saul, B. Schölkopf (Eds.), Advances in Neural Information Processing Systems 16, MIT Press, Cambridge, MA, 2004.

[18] J. B. Tenenbaum, V. de Silva, J. C. Langford, A global geometric framework for nonlinear dimensionality reduction, Science 290 (5500) (2000) 2319–2323.

[19] S. T. Roweis, L. K. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (5500) (2000) 2323–2326.

[20] C. Hou, J. Wang, Y. Wu, D. Yi, Local linear transformation embedding, Neurocomputing 72 (10–12) (2009) 2368–2378.

[21] J. Chen, Y. Liu, Locally linear embedding: a survey, Artif. Intell. Rev. 36 (1) (2011) 29–48.

[22] T. Zhou, D. Tao, Double shrinking sparse dimension reduction, IEEE Trans. Image Process. 22 (1) (2013) 244–257.

[23] H. Lu, K. N. Plataniotis, A. N. Venetsanopoulos, A survey of multilinear subspace learning for tensor data, Pattern Recognition 44 (7) (2011) 1540–1551.

[24] R. Vidal, Subspace clustering, Signal Processing Magazine 28 (2) (2011) 52–68.

[25] S. Günnemann, H. Kremer, T. Seidl, Subspace clustering for uncertain data, in: Proc. SIAM International Conference on Data Mining, 2010, pp. 385–396.

[26] F. O'Gorman, M. B. Clowes, Finding picture edges through collinearity of feature points, IEEE Trans. Comput. C-25 (4) (1976) 449–456.

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

[28] D. Cyganski, N. W. F., J. A. Orr, Analytic Hough transform, in: Proc. of SPIE, Sens. and Reconstruction of Three-Dimensional Objects and Scenes, Vol. 1260, SPIE, 1990, pp. 148–159.

[29] Y. Liu, D. Cyganski, R. F. Vaz, Efficient implementation of the analytic Hough transform for exact linear feature extraction, in: Proc. of SPIE, Intell. Robots and Computer Vis. X, Vol. 1607, SPIE, 1992, pp. 298–309.

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

[31] D. Hestenes, G. Sobczyk, Clifford algebra to geometric calculus: a unified language for mathematics and physics, Springer, 1984.

[32] D. Hestenes, New foundations for classical mechanics, Springer, 1987.

[33] C. Doran, J. Lasenby, L. Dorst, D. Hestenes, S. Mann, A. Naeve, A. Rockwood, Geometric algebra: new foundations, new insights, Course notes for the 27th SIGGRAPH (2000).

[34] C. Doran, J. Lasenby, L. Dorst, D. Hestenes, S. Mann, A. Naeve, A. Rockwood, Geometric algebra, Course notes for the 28th SIGGRAPH (2001).

[35] J. Harris, Algebraic geometry: a first course, Springer-Verlag, 1992.

[36] D. Fontijne, Efficient algorithms for factorization and join of blades, in: Proc. of the 3rd International Conference on Applications of Geometric Algebras in Computer Science and Engineering, 2008, pp. 457–476.

[37] D. K. Hoffman, R. C. Raffenetti, K. Ruedenberg, Generalization of Euler angles to n-dimensional orthogonal matrices, J. Math. Phys. 13 (4) (1972) 528–533.

[38] K. V. Mardia, Measures of multivariate skewness and kurtosis with applications, Biometrika 57 (3) (1970) 519–530.

[39] R. E. Bogner, Pattern recognition via observation correlations, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-3 (2) (1981) 128–133.