Int. J. Appl. Math. Comput. Sci., 2020, Vol. 30, No. 3, 415–433
DOI: 10.34768/amcs-2020-0031

IMPLEMENTATION AND EVALUATION OF MEDICAL IMAGING TECHNIQUES BASED ON CONFORMAL GEOMETRIC ALGEBRA

SILVIA FRANCHINI [a,*], ANTONIO GENTILE [b], GIORGIO VASSALLO [b], SALVATORE VITABILE [a]

[a]Department of Biomedicine, Neuroscience and Advanced Diagnostics (Bi.N.D.)
University of Palermo
Via del Vespro, 129, 90127 Palermo, Italy
e-mail: silvia.franchini@unipa.it

[b]Department of Engineering
University of Palermo
Viale delle Scienze, Edificio 6, 90128 Palermo, Italy

Medical imaging tasks, such as segmentation, 3D modeling, and registration of medical images, involve complex geometric problems, usually solved by standard linear algebra and matrix calculations. In the last few decades, conformal geometric algebra (CGA) has emerged as a new approach to geometric computing that offers a simple and efficient representation of geometric objects and transformations. However, the practical use of CGA-based methods for big data image processing in medical imaging requires fast and efficient implementations of CGA operations to meet both real-time processing constraints and accuracy requirements. The purpose of this study is to present a novel implementation of CGA-based medical imaging techniques that makes them effective and practically usable. The paper exploits a new simplified formulation of CGA operators that allows significantly reduced execution times while maintaining the needed result precision. We have exploited this novel CGA formulation to re-design a suite of medical imaging automatic methods, including image segmentation, 3D reconstruction and registration. Experimental tests show that the re-formulated CGA-based methods lead to both higher precision results and reduced computation times, which makes them suitable for big data image processing applications. The segmentation algorithm provides the Dice index, sensitivity and specificity values of 98.14%, 98.05% and 97.73%, respectively, while the order of magnitude of the errors measured for the registration methods is 10^{-5}.

Keywords: medical image segmentation, medical image registration, computational geometry, Clifford algebra, conformal geometric algebra.

1. Introduction

Modern medical imaging techniques involve massive volumes of medical data and require innovative solutions to efficiently handle big data image processing (Stefanowski *et al*., 2017). Geometry plays an important role in several aspects of medical imaging. Efficient geometric tools are required to solve different problems arising in medical image analysis and processing, such as image segmentation, shape approximation, 3D rendering, and registration of medical data (Fabijańska *et al*., 2014; Hrebień *et al*., 2008). To address these tasks, classical methods use standard geometric techniques based on linear algebra and matrix operations. A novel approach to geometric modeling is provided by conformal geometric algebra (CGA), namely, five-dimensional (5D) geometric algebra (GA) or Clifford algebra (CA) (Clifford, 1882), which is attracting a growing attention in many research fields, such as computer graphics, computer vision, and image processing (Hestenes, 1986; Hestenes and Sobczyk, 1987; Dorst *et al*., 2007). CGA is a 5D representation of the 3D space that allows for simple and intuitive modeling of complex geometric constructions. Using the mathematical framework of CGA, geometric objects such as spheres, circles, lines, and planes, become simply elements of algebra and can be easily transformed by algebra

*Corresponding author

operators. Geometric transformations, such as rotations, translations, and dilations, can all be obtained by simple universal operations in the 5D space.

1.1. Related works. Several applications of GA have been proposed in the last few years. Software and integrated software/hardware tools have been developed to efficiently implement GA objects and operators (Fontijne, 2006; Ashdown, 2018; Hitzer and Sangwine, 2018; Hildenbrand, 2018). Different dedicated hardware architectures have been also designed to natively support GA operations (Mishra *et al.*, 2015; Gentile *et al.*, 2005; Franchini *et al.*, 2008; 2011; 2012; 2013; 2015). GA has been successfully applied in different research fields, such as computer vision (Sommer, 2001; Lasenby *et al.*, 1998) and image processing (Batard *et al.*, 2010; Ebling and Scheuermann, 2005; Menneson *et al.*, 2011).

Recent studies have proposed the use of geometric methods based on CGA to solve different medical imaging tasks, including image segmentation, 3D surface modeling, and volume registration (Rivera-Rovelo and Bayro-Corrochano, 2006; 2007; Bayro-Corrochano and Rivera-Rovelo, 2009). A fully automatic segmentation method that exploits the generalized gradient vector flow (GGVF) and a self-organizing neural network with CGA translators as weights is used by Rivera-Rovelo and Bayro-Corrochano (2006; 2007) to extract the shape of objects of interest within medical images. The proposed algorithm has been validated by experiments on computer tomography (CT) and magnetic resonance (MR) medical images.

A CGA-based 3D rendering method, which allows reconstructing a 3D shape of anatomical areas of interest (organs, tumors, etc.) starting from a sequence of 2D images and their boundary points, has been presented by Bayro-Corrochano and Rivera-Rovelo (2009). While traditional rendering techniques provide 3D models composed of a collection of polygons, such as triangles, this rendering method, named Marching Spheres, integrates the ideas of the Marching Cubes (Newman and Yi, 2006) and Union of Spheres (Ranjan and Fournier, 1995) algorithms to obtain a 3D model composed of CGA spheres. The Marching Spheres technique uses a reduced number of primitives allowing for a reduction in the computational complexity with respect to the state-of-the-art methods. A CGA-based registration technique that allows for aligning two misaligned 3D models derived from two misaligned 2D medical image sequences is also presented by Bayro-Corrochano and Rivera-Rovelo (2009). The authors have adapted the thin-plate spline robust point matching (TPS-RPM) registration algorithm, usually used to align two models, each composed of a cloud of 2D/3D points, to align two 3D models based on CGA spheres. However, the paper does not provide performance comparison with traditional methods.

1.2. Our contribution. The above cited works introduce some ideas on the use of CGA in the medical imaging area and present some preliminary results. However, the practical use of the CGA-based techniques is hindered by the high dimensionality and the consequent relevant computational complexity of CGA operations so that fast and efficient implementations are needed to meet real-time processing constraints as well as accuracy and robustness requirements in medical imaging applications.

This paper proposes a novel simplified formulation of CGA operations, properly conceived to reduce the computational complexity of the 5D operators. This new formulation has been exploited to re-design the suite of CGA-based medical imaging algorithms introduced by Rivera-Rovelo and Bayro-Corrochano (2007) and Bayro-Corrochano and Rivera-Rovelo (2009). Our work proposes a medical image processing chain based on the re-formulated CGA algorithms, which includes three successive stages: segmentation of the 2D image sequence (CT or MR), 3D model extraction, and registration of two misaligned 3D models. The re-formulated CGA operators are used in each step to perform geometric calculations (rotations, translations, dilations) required to process medical images. This work also presents a reformulation of the iterative closest point (ICP) registration algorithm (Besl and McKay, 1992) in the CGA framework. An application programming interface (API), named ConformalAPI, has been designed and implemented to interface the CGA-based medical algorithms with the new fast CGA operators. Experimental results show that the new implementation of the CGA-based medical imaging algorithms leads to higher accuracy and shorter computation times compared with both the standard CGA techniques and traditional medical image processing methods, making CGA methods usable in real-time medical imaging applications. A detailed comparison between the proposed methods and the state-of-the-art approaches is reported in Table 1. We can observe that the new algebraic approach proposed in this paper improves the accuracy, speed, and robustness of the medical image processing algorithms and makes the proposed methods practically usable for big data image processing and analysis.

The rest of the paper is organized as follows: Section 2 presents CGA basic concepts, while the novel formulation of CGA operators used to improve the performance of the CGA-based medical imaging algorithms is presented in Section 3. The medical imaging techniques based on re-formulated CGA are described in Section 4, while Section 5 outlines experimental results and presents the performance comparison with the standard methods. Finally, Section 6 contains discussion and conclusions.

Table 1. Comparison between the proposed methods and the state-of-the-art methods.

	State-of-the-art methods		Proposed methods
	Rivera-Rovelo and Bayro-Corrochano (2006; 2007)	Bayro-Corrochano and Rivera-Rovelo (2009)	
Algorithms	Segmentation	3D modeling and registration	Complete medical image processing chain: segmentation, 3D modeling, and registration
Algebra used	Standard CGA formulation	Standard CGA formulation	New simplified CGA formulation to reduce computational complexity of 5D operators
New proposed registration methods	–	CGA-based TPS-RPM	CGA-based ICP and new CGA-based TPS-RPM
Experimental validation Experimental setup	Preliminary tests on a limited set of medical images	Preliminary tests on a limited set of synthetic and real medical images	Experimental tests on 8080 real medical images (CT abdominal, CT brain, and MR brain)
Performance comparison with standard methods	No	No	Comparison with both standard CGA techniques and traditional medical imaging methods
Comparison results	–	–	Better performance in terms of both precision and computation times
Quantitative analysis	No	Registration error minimum 0.72	Segmentation Dice index 98.14% Sensitivity 98.05 % Specificity 97.73% Registration error $\approx 10^{-5}$
Statistical analysis	No	No	Yes (Wilcoxon signed-rank test)

2. Conformal geometric algebra

A detailed introduction to Clifford algebra (CA) and conformal geometric algebra (CGA) can be found in (Hestenes and Sobczyk, 1987; Hestenes, 1986; Dorst *et al.*, 2007). In this section, we introduce some basic CGA concepts. CGA is based on 5D geometric (or Clifford) algebra and provides a 5D representation of the 3D space. Increasing the space dimension, CGA offers a more intuitive and compact representation of the 3D geometry, since geometric objects and transformations all can be represented by algebra elements and operators, respectively. Two extra basis vectors e_+ and e_- are added to the three basis vectors e_1, e_2, and e_3 of the 3D Euclidean space. In general, given two generic basic vectors e_i and e_j, the following axioms are introduced:

$$e_i^2 = \pm 1, \tag{1}$$

$$e_i e_j = -e_j e_i, \tag{2}$$

e_+ and e_- square to 1 and -1, respectively. The two null vectors e_0 and e_∞, which represent the point at the origin and the point at infinity, respectively, are derived from e_+ and e_- and they are defined as

$$e_0 = \frac{1}{2}(e_- - e_+), \quad e_\infty = e_- + e_+. \tag{3}$$

From (3), it follows that $e_0^2 = e_\infty^2 = 0$, while $e_0 e_\infty = -1$. In CGA, a point P is represented as a 5D vector

$$P = p + \frac{1}{2}p^2 e_\infty + e_0, \tag{4}$$

where p is a vector in the 3D space: $p = p_1 e_1 + p_2 e_2 + p_3 e_3$. Equation (4) is used to map a point p of the 3D Euclidean space to a point P of the 5D conformal space.

The basic entities in CGA are spheres. A sphere S with center c and radius r is represented as a 5D vector

$$S = c + \frac{1}{2}(c^2 - r^2)e_\infty + e_0. \tag{5}$$

Table 2. Classical formulation of CGA operators.

Reflection	$S' = -mSm$	$m =$ reflection plane
Rotation	$S' = RS\tilde{R}$	$R = e^{b\theta}$ $b =$ rotation plane $2\theta =$ rotation angle
Translation	$S' = TS\tilde{T}$	$T = e^{-te_\infty/2}$ $t =$ 3D translation vector
Dilation	$S' = DS\tilde{D}$	$D = e^{-E\log(l)/2}$ $l =$ scale factor $E = e_\infty \wedge e_0$[1]

Other geometric objects are derived from the spheres. A point is a sphere of radius zero (see Eqns. (4) and (5)), a circle is the intersection of two spheres, a plane is a sphere that passes through the infinity and a line is a circle with infinite radius. A plane Π with normal vector $n = n_1e_1 + n_2e_2 + n_3e_3$ and d as the distance from the origin is expressed as a 5D vector

$$\Pi = n + de_\infty. \tag{6}$$

Since a sphere is a 5D vector and geometric objects can be derived from spheres, we can say that, in the CGA framework, geometric objects can be reduced to 5D conformal vectors. The suite of medical imaging algorithms presented in this paper is based on the massive use of conformal (i.e., angle-preserving) geometric transformations (reflections, rotations, translations, dilations). CGA provides a universal and compact representation of conformal geometric transformations. In CGA, all these transformations are obtained by operators called versors that pre-multiply and post-multiply the object to be transformed. A versor V is the product of non-null vectors: $V = v_1v_2 \dots v_k$. Conformal transformations are all expressed in the following compact form (known as the "sandwich" product):

$$X' = (-1)^k VX\tilde{V}, \tag{7}$$

where X is the object to be transformed, $V = v_1v_2 \dots v_k$ is the versor that represents the geometric transformation, $\tilde{V} = v_k \dots v_2v_1$ is the reverse of V and X' is the transformed object. The classical CGA formulation of conformal transformations for a specific sphere S is reported in Table 2, where the operator V of Eqn. (7) takes a different form according to the specific geometric transformation, while the sphere S is a 5D vector, as defined in Eqn. (5), and S' is the transformed sphere.

[1]The symbol $\wedge$ represents the outer product. E is the so-called Minkowski plane.

3. Novel formulation of conformal geometric algebra operations

Equation (7) and Table 2 show the classical formulation of CGA operators. The low symbolic complexity of this formulation is accompanied, however, by its high numeric complexity. Since CGA operates in the 5D space, we have to deal with multiplications and sums of elements with a high number of coefficients. This numeric complexity hinders the practical use of CGA in real-time applications, such as medical image processing algorithms. To face this problem, Franchini *et al.* (2015) have introduced a new, simplified formulation of CGA with the aim to reduce the computational complexity of the 5D operators. According to this new formulation: (i) each conformal transformation (rotation, translation, dilation) is split in two consecutive reflections, and (ii) each reflection is obtained using a fast and compact formula based on a simple dot product instead of the standard "sandwich" product of Table 2.

The novel formulation of translation, dilation, and rotation operations is described in more detail in Sections 4.1.1, 4.2.1, and 4.3.3, respectively.

3.1. Computational complexity analysis. The new formulation results in a significant reduction in the computational complexity of CGA operations with respect to the standard formula of Eqn. (7). Figure 1 compares the number of primitive arithmetic operations required by the standard formulation of CGA operators with the number required by the new formulation. The reported values are related to CGA operations on vectors and therefore on points or spheres that are the objects used in the medical imaging algorithms presented in the paper. The new CGA formulation has allowed us to reduce the computational load of the algebra operators (the number of basic arithmetic operations per CGA operator) by about one order of magnitude and, therefore, to improve the performance of the medical imaging algorithms that exploit CGA operators in terms of both execution times and result precision (see Section 5 for experimental results).

4. Medical imaging techniques based on conformal geometric algebra

This section outlines a suite of techniques based on CGA for segmentation, 3D modeling and registration of medical images, which have been developed starting from the guidelines contained in the works of Rivera-Rovelo and Bayro-Corrochano (2006; 2007) as well as Bayro-Corrochano and Rivera-Rovelo (2009) and re-designed according to the novel CGA formulation described in Section 3. The proposed system is a medical

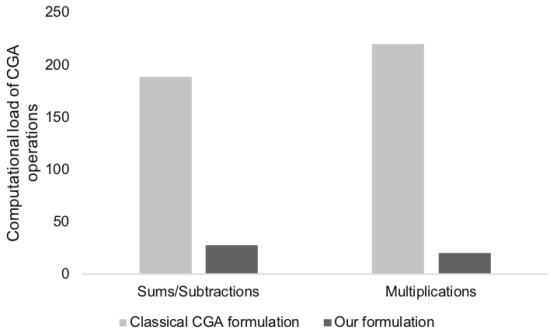


Fig. 1. Comparison between the standard CGA formulation and the new CGA formulation in terms of the number of basic arithmetic operations (multiplications and sums/subtractions) required to execute a conformal geometric transformation (rotation, translation, or dilation). Note that all conformal geometric transformations require the same number of basic arithmetic operations since they are obtained by the same formula (the "sandwich" geometric product in the standard formulation and double reflection in the new formulation used in this paper).

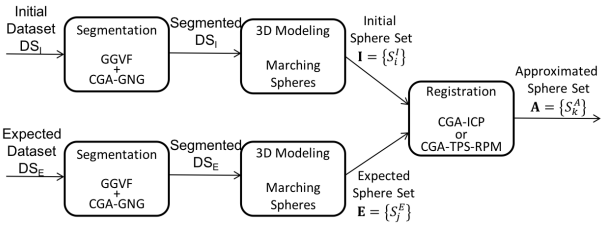


Fig. 2. 3D dataset registration: block diagram of the medical image processing chain. The registration (or alignment) task has many important applications in clinical practice including (i) aligning temporal sequences of images to compensate for motion of the patient between scans, (ii) combining images of the same subject from different imaging modalities (the multi-modality fusion facilitates diagnosis by integrating information acquired by diverse imaging devices, as CT and MR), (iii) image guidance during interventions, (iv) aligning images from the same subject or from multiple patients in longitudinal studies, which investigate temporal changes in anatomical structures (regions of interest, organs, cancers, etc.).

image processing chain whose block diagram is depicted in Fig. 2.

The goal of the whole system is to register (align) two different medical datasets (the initial dataset DS_I and the expected dataset DS_E of Fig. 2). Each dataset consists of a sequence of MR or CT images derived by successive scans. First, the reformulated CGA-based segmentation method, based on the GGVF technique and a growing neural gas (GNG) network (Rivera-Rovelo and Bayro-Corrochano, 2006; 2007), is used to extract the area of interest from each image. Then, for each input dataset, a 3D model is reconstructed starting from the sequence of the segmented 2D slices. The 3D rendering technique, named marching spheres (Bayro-Corrochano and Rivera-Rovelo, 2009), gives as output a 3D model composed of spheres defined in the CGA framework. Finally, the two sphere-based 3D models are aligned using one of the following well-known registration methods, iterative closest point (ICP) and thin-plate spline robust point matching (TPS-RPM), reformulated to operate in the CGA context and align two 3D models composed of CGA spheres. The above cited medical algorithms require millions of geometric transformations, such as translations, rotations and dilations. In our advanced implementation, all these transformations are executed using the novel fast CGA operators described in Section 3 and, in more detail, in Sections 4.1.1, 4.2.1, and 4.3.3. We have designed and implemented an application programming interface (API), named ConformalAPI, for high-level applications to interface with our new CGA operators. The main functions of the ConformalAPI library are reported in Table 3 and described in the following subsections.

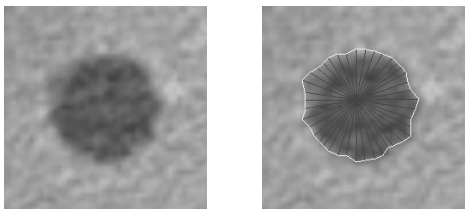
Sections 4.1, 4.2, and 4.3 describe the new enhanced formulation of the CGA-based segmentation, 3D modeling and volume registration methods, respectively.

4.1. Medical image segmentation. The segmentation algorithm consists of two steps. In the first step, a set of contour points of the area to be segmented are extracted by using an automatic technique based on the generalized gradient vector flow (GGVF). In the second step, the initial contour points derived in the first step are used as the training set of a GNG network, which has translators defined in the CGA framework as weights. Once the training stage is completed, GNG neurons will contain the proper translators, which, applied to a given initial point (the centroid of the contour points), will translate it to the object contour and will define object shape. Figure 3(b) shows the segmented object obtained after the two steps of the algorithm for the test image in Fig. 3(a) (CT liver lesion). The profiling of the segmentation algorithm (Fig. 4(a)) shows that the most computationally expensive functions are the translation operations and the distance calculations between the translated points and the initial contour points, which consume more than 70% of the execution time. In our implementation, all CGA-based geometric operations used in the GNG algorithm exploit the fast CGA operators introduced in Section 3 instead of the standard formulation of Table 2.

4.1.1. New formulation of translations. A translation in the direction given by the vector n is executed as

Table 3. Main functions of ConformalAPI.

Function	**rotation**
Syntax	`float* rotation(float a1, float a2, float a3, float alpha, float* x)`
Description	Starting from the rotation plane with normal vector a and from the rotation angle $alpha$, calculates the rotated 5D object
Arguments	a_1, a_2, a_3: coefficients of the normal vector to the rotation plane; $alpha$: rotation angle; x: 5D object to be rotated
Function	**from_rot_axis_and_angle_to_refl_vecs**
Syntax	`float* from_rot_axis_and_angle_to_refl_vecs(float* a, float alpha)`
Description	Starting from the rotation axis and rotation angle, calculates the two planes (5D vectors) with respect to which the two successive reflections have to be executed
Arguments	a: rotation axis; $alpha$: rotation angle
Function	**orthonormal_ref**
Syntax	`float* orthonormal_ref(float* a)`
Description	Starting from the rotation axis a, calculates the orthonormal basis $B = (u, v, w)$ with $w = a$
Arguments	a: rotation axis
Function	**translation**
Syntax	`float* translation(float t1, float t2, float t3, float* x)`
Description	Starting from the 3D translation vector, calculates the translated 5D object
Arguments	t_1, t_2, t_3: coefficients of the 3D translation vector; x: 5D object to be translated
Function	**from_transl_vec_to_refl_vecs**
Syntax	`float* from_transl_vec_to_refl_vecs(float t1, float t2, float t3)`
Description	Starting from the 3D translation vector, calculates the two planes (5D vectors) with respect to which the two successive reflections have to be executed
Arguments	t_1, t_2, t_3: coefficients of the 3D translation vector
Function	**dilation**
Syntax	`float* dilation(float l, float* x)`
Description	Starting from the scaling factor l, calculates the dilated 5D object
Arguments	l: scaling factor; x: 5D object to be dilated
Function	**from_scaling_factor_to_refl_vecs**
Syntax	`float* from_scaling_factor_to_refl_vecs(float l)`
Description	Starting from the scaling factor l, calculates the two spheres (5D vectors) with respect to which the two successive reflections have to be executed
Arguments	l: scaling factor
Function	**reflection**
Syntax	`float* reflection(float* m, float* x)`
Description	Starting from the 5D object x, calculates the reflected 5D object with respect to the plane m
Arguments	m: reflection plane; x: 5D object to be reflected
Function	**distance**
Syntax	`float distance(float* P1, float* P2)`
Description	Calculates the Euclidean distance between two conformal points $P1$ and $P2$ according to the formula given in equation (28)
Arguments	$P1$: first 5D point; $P2$: second 5D point
Function	**CGA_sphere**
Syntax	`float* CGA_sphere(float c1, float c2, float c3, float r)`
Description	Maps a sphere with center c and radius r in a 5D vector
Arguments	$c1$, $c2$, $c3$: coefficients of the center point c; r: radius
Function	**CGA_plane**
Syntax	`float* CGA_plane(float n1, float n2, float n3, float d)`
Description	Maps a plane with normal vector n and distance d from the origin in a 5D vector
Arguments	$n1$, $n2$, $n3$: coefficients of the normal vector n; d: distance from the origin
Function	**CGA_point**
Syntax	`float* CGA_point(float p1, float p2, float p3)`
Description	Maps a 3D point p in a 5D vector
Arguments	$p1$, $p2$, $p3$: coordinates of the 3D point p



(a) (b)

Fig. 3. Segmentation of a test image: original image (a), segmented object (b).

two successive reflections in two parallel planes Π_1 and Π_2 with unit normal vector n and distance d from each other. The translation will be twice the distance d between the two planes. A proper function in our API library (the function `from_transl_vec_to_refl_vecs` in Table 3) calculates the two planes (or 5D vectors) with respect to which the two successive reflections have to be executed. Starting from the 3D translation vector $t = t_1e_1 + t_2e_2 + t_3e_3$, the API function calculates the two 5D vectors Π_1 and Π_2, as described below. In our method, since the first reflection is executed in the plane $\Pi_1 = n + d_1e_\infty$ with normal vector n and d_1 as the distance from the origin, while the second reflection is executed in the parallel plane $\Pi_2 = n+d_2e_\infty$ with normal vector n and d_2 as the distance from the origin and with $d = d_2 - d_1$, the translator can be written as the product of the two parallel planes:

$$\begin{aligned} T &= (n + d_2e_\infty)(n + d_1e_\infty) \\ &= nn + d_1ne_\infty - d_2ne_\infty, \end{aligned} \tag{8}$$

from which

$$T = 1 - (d_2 - d_1)ne_\infty. \tag{9}$$

On the other hand, according to the formula in Table 2, the translator is expressed as

$$T = e^{-\frac{t}{2}e_\infty}. \tag{10}$$

Taking into account the Taylor series expansion, (10) can be written as

$$T \approx 1 - \frac{1}{2}te_\infty. \tag{11}$$

Starting from (9) and (11), we can write

$$1 - (d_2 - d_1)ne_\infty = 1 - \frac{1}{2}te_\infty, \tag{12}$$

from which

$$t = 2(d_2 - d_1)n = 2dn. \tag{13}$$

Since

$$t = \|t\|\, n, \tag{14}$$

we can write

$$\|t\|\, n = 2dn, \tag{15}$$

from which

$$d = \frac{\|t\|}{2}, \tag{16}$$

while n can be calculated as

$$n = \frac{t}{\|t\|}. \tag{17}$$

Finally, since the transformation depends on the offset $d = d_2 - d_1$, the two reflection planes Π_1 and Π_2 can be expressed as

$$\Pi_1 = n = n_1e_1 + n_2e_2 + n_3e_3, \tag{18}$$

$$\begin{aligned} \Pi_2 &= n + de_\infty \\ &= n_1e_1 + n_2e_2 + n_3e_3 + de_+ + de_-, \end{aligned} \tag{19}$$

where n and d are expressed by (17) and (16), respectively.

Another API function (the function `translation` in Table 3) executes the translation operation. This function receives the 3D translation vector t as the argument and returns the translated 5D object. The function `translation` uses the function `from_transl_vec_to_refl_vecs` to calculate the two 5D reflection planes starting from the 3D translation vector and then executes the translation using twice the fast reflection operator. The latter operator uses the compact formula based on the dot product and is implemented in the API function `reflection` reported in Table 3. According to this compact formula, given a vector a, the vector a', reflected in a plane with unit-normal m, can be obtained as

$$a' = a - 2(a \cdot m)m. \tag{20}$$

For the component of a collinear with m ($a_\parallel$) and the component of a orthogonal to m ($a_\perp$), the reflected vector a' can be expressed as

$$a' = a_\perp - a_\parallel. \tag{21}$$

Adding and subtracting $a_\parallel$, we obtain

$$a' = a_\perp + a_\parallel - a_\parallel - a_\parallel. \tag{22}$$

Since

$$a = a_\perp + a_\parallel, \tag{23}$$

we can write

$$\begin{aligned} a' &= a - 2a_\parallel = a - 2|a_\parallel|m \\ &= a - 2(a \cdot m)m. \end{aligned} \tag{24}$$

Regarding the distance calculations, we can observe that in CGA the Euclidean distance between two points

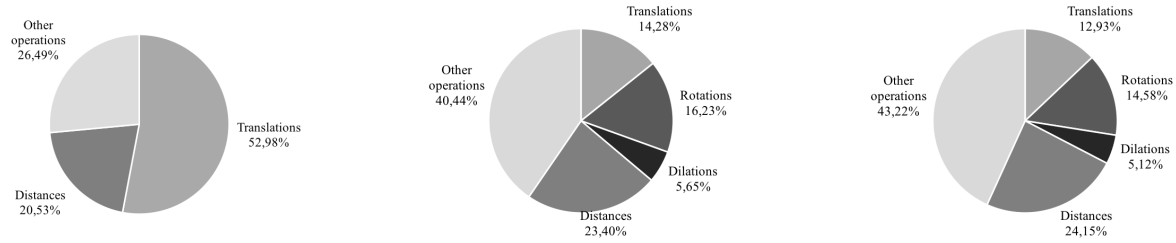


Fig. 4. Profiling: segmentation algorithm (a), ICP registration algorithm (b), TPS-RPM registration algorithm (c).

is proportional to the inner product or dot product of the two related 5D vectors. Given two points p_A and p_B, their Euclidean distance can be expressed as:

$$\begin{aligned} \text{dist}^2_{p_A p_B} &= (p_A - p_B)^2 \\ &= p_A^2 + p_B^2 - 2p_A \cdot p_B. \end{aligned} \tag{25}$$

On the other hand, starting from (4), the inner product of two 5D vectors P_A and P_B can be expressed as

$$P_A \cdot P_B = (p_A + \frac{1}{2}p_A^2 e_\infty + e_0)(p_B + \frac{1}{2}p_B^2 e_\infty + e_0), \tag{26}$$

from which

$$P_A \cdot P_B = -\frac{1}{2}p_A^2 - \frac{1}{2}p_B^2 + p_A \cdot p_B \tag{27}$$

so that, comparing (25) and (27), we can express the Euclidean distance as

$$\text{dist}^2_{p_A p_B} = -2P_A \cdot P_B. \tag{28}$$

The inner product in the representation space is proportional to the squared Euclidean distance. Therefore, to measure the Euclidean distance of two points, we have used the dot product of the two 5D vectors. The function `distance` of the ConformalAPI library (see Table 3) has been used for the distance calculations. The API functions `CGA_point`, `CGA_sphere`, and `CGA_plane` are used to map a point, a sphere, and a plane, respectively, from the 3D space into a 5D vector. To have an idea of the computational load reduction achieved by our method, we can observe that the segmentation of the medical image shown in Fig. 6, which requires 10,236 translations, needs 255,900 products and 327,552 sums/differences, whereas the standard CGA method would require 2,303,100 products and 2,026,728 sums/differences.

4.2. 3D modeling. In our implementation of the Marching Spheres algorithm, the spheres that will constitute the 3D model are generated by CGA dilation and translation operators, which are applied to a basic sphere with the center in the origin and unitary radius. The algorithm first calculates the centers and radii of all spheres and then executes the proper dilation and translation operations to generate the spheres of the final model. The needed translations and dilations are executed using the formulation presented in Section 3. The translations are executed using the API functions presented in Section 4.1.1.

4.2.1. New formulation of dilations. A dilation of a scaling factor l is obtained as two successive reflections in two spheres with the center in the origin and radii 1 and $\sqrt{l}$, respectively, represented by the two following 5D vectors:

$$S_1 = -\frac{1}{2}e_\infty + e_0, \tag{29}$$

$$S_{\sqrt{l}} = -\frac{1}{2}le_\infty + e_0. \tag{30}$$

The dilator D can be obtained starting from the product of these two vectors:

$$\begin{aligned} D &= \frac{1}{\sqrt{l}}\Big(-\frac{1}{2}le_\infty + e_0\Big)\Big(-\frac{1}{2}e_\infty + e_0\Big) \\ &= \frac{1}{\sqrt{l}}\Big(\frac{l+1}{2} - \frac{l-1}{2}e_\infty \wedge e_0\Big) \\ &= \frac{l^{\frac{1}{2}} + l^{-\frac{1}{2}}}{2} - \frac{l^{\frac{1}{2}} - l^{-\frac{1}{2}}}{2}E. \end{aligned} \tag{31}$$

Since

$$\begin{aligned} \cosh(x) &= \frac{e^x + e^{-x}}{2}, \\ \sinh(x) &= \frac{e^x - e^{-x}}{2}, \end{aligned} \tag{32}$$

the dilator D can be expressed as

$$D = \cosh\Big(\frac{\log(l)}{2}\Big) - \sinh\Big(\frac{\log(l)}{2}\Big)E, \tag{33}$$

from which

$$D = e^{-\frac{1}{2}E\log(l)}, \tag{34}$$

that is the standard formulation of the dilation as reported in Table 2.

The API function `dilation` (Table 3) is used to execute a dilation according to the proposed formulation. This function uses the API function `from_scaling_factor_to_refl_vecs` (Table 3) to calculate the two reflection vectors starting from the scaling factor, while the API function `reflection` is called twice to execute the two consecutive reflections. Figure 5 shows the result of the proposed rendering method for a sample dataset composed of 175 MR brain slices. One of the MR slices is shown in Fig. 6 along with its binarized version obtained after the segmentation process described in Section 4.1. It has to be observed that the number of spheres of the 3D model depends on the distance d between two adjacent slices. A smaller value of d leads to a higher number of spheres and, therefore, to a higher number of CGA translations and dilations, but at the same time allows us to obtain a more precise model.

4.3. Volume registration. In this work, we have redesigned both the TPS-RPM and ICP algorithms to align two 3D models composed of spheres defined in the CGA context (where each sphere is represented by a point in a 5D space or a 5D vector) and exploit the re-formulated 5D operators introduced in Section 3. The following notation will be used: $I = \{S_i^I\}, i = 1, 2, \ldots, m$ is the set of spheres of the first model (initial set), $E = \{S_j^E\}, j = 1, 2, \ldots, n$ is the set of spheres of the second model (expected model), $A = \{S_k^A\}, k = 1, 2, \ldots, n$ is the set of spheres resulting after the algorithm has been executed (estimated or approximated set).

4.3.1. CGA-based iterative closest point (ICP). In this section, we propose a novel CGA-based version of the ICP algorithm. Several variants of the ICP algorithm have been proposed over the years. We have adapted to the CGA framework the version proposed by Zhang (1994). He introduces a statistical method based on the distance distribution to reject outliers (points of either set that have no correspondences in the other) during the matching process. The goal of the registration process is to find the proper geometric transformation that aligns the initial model I with the expected model E. In the ICP algorithm, this transformation is assumed to be linear and composed of a rotation R and a translation T. The algorithm searches the optimal transformation (R, T) that minimizes the following error function:

$$E(R,T) = \frac{1}{n}\sum_{i=1}^{n}(TRS_i^I\tilde{R}\tilde{T} - S_i^E)^2 \qquad (35)$$

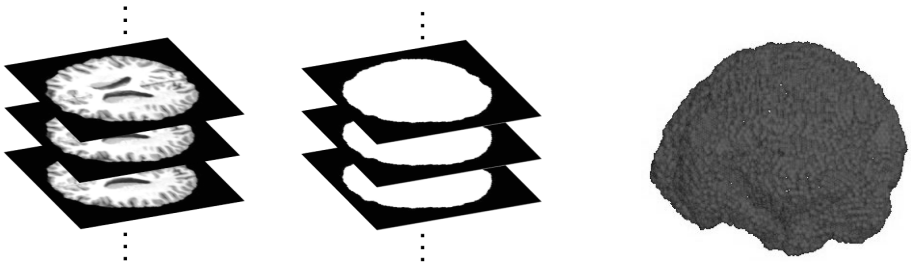
which represents a measure of the "distance" between the set of spheres of the initial model and the set of spheres of the expected model. R and T are respectively, the CGA rotator and translator expressed according to the new formulation introduced in Section 3. The ICP algorithm consists of the following steps that are iteratively repeated:

1. Find the correspondences between the spheres of the two sets, namely, compute the pairs of spheres with minimum distances (closest spheres) of the two sets.
2. Compute the transformation (R, T) that minimizes the error function $E(R, T)$.
3. Apply the transformation to the initial set of spheres $I = \{S_i^I\}$ (the radii of the corresponding spheres are also compared and, if necessary, a dilation operator is applied to correctly overlap the two spheres).

Steps 1–3 are repeated until $|E_k(R,T) - E_{k-1}(R,T)|$ becomes less than a certain threshold ϵ, where $E_k(R,T)$ and $E_{k-1}(R,T)$ are the errors measured at steps k and $k-1$, respectively.

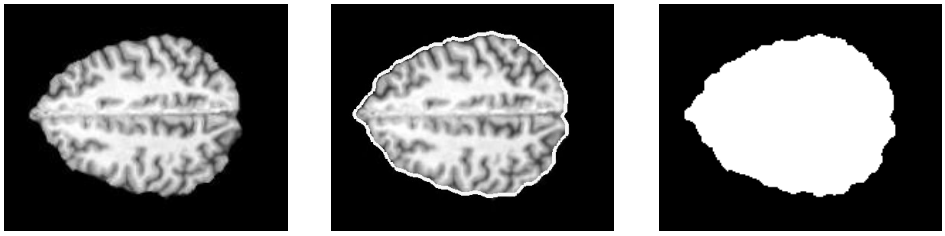
4.3.2. CGA-based thin-plate spline robust point matching (TPS-RPM). The ICP algorithm converges to the closest local minimum; therefore, it works well when the transformation is small and approximately known, while it is not appropriate for solving large motion problems since, in these cases, it may lead to false matching. Another registration method that solves the problem of the local minima and works even in the presence of a high number of outliers is the TPS-RPM. The TPS-RPM algorithm treats non-rigid transformations and can therefore be useful to solve the deformable matching problems frequently arising in medical imaging. The TPS-RPM adaptation for the non-rigid registration of two 3D models based on CGA spheres has been proposed by Bayro-Corrochano and Rivera-Rovelo (2009). The transformation between the two models takes the form of the thin-plate spline (TPS), which is a non-rigid extension of the affine map and is split in two parts: the affine transformation and the non-rigid deformation (warping), which are applied to the centers of the spheres. In our implementation of the CGA-based TPS-RPM, the affine mapping is implemented by using the conformal geometric operations (rotations, translations and dilations) defined in CGA framework. Dilation operators are used to update the radii of the spheres.

The profiling of the registration algorithms (Figs. 4(b) and (c)) shows that the most computationally expensive functions are the rotation, translation, and dilation operations as well as the distance calculations between the centers of the spheres. To reduce the computational load of the most time-consuming functions, in our implementation, the rotation, translation and dilation operators are executed according to the formulation introduced in Section 3 instead of the standard formulation of Table 2. The new formulations of translations and dilations and their related functions in



(a) (b) (c)

Fig. 5. 3D model derived from a MR brain sequence composed of 175 slices: original slices (a), slices obtained after the segmentation and binarization process (b), 3D model obtained for $d = 2$, where d is the distance between two adjacent slices (c).



(a) (b) (c)

Fig. 6. Brain MR slice: original MR brain slice (a), segmented image (b), binarized image after the segmentation process (c).

the ConformalAPI library have been already presented in Sections 4.1.1 and 4.2.1.

4.3.3. New formulation of rotations. Regarding the rotations, in our algorithm, a rotation by an angle α around the axis a is executed as two consecutive reflections in two planes with unit-normals m and n, where $m \wedge n$ (the plane in which m and n lie) represents the rotation plane, while the angle between m and n is half the rotation angle α. The rotor R can be written as the geometric product of these two vectors:

$$R = nm. \tag{36}$$

Since the geometric product is the sum of the inner product and the outer product, we can write

$$R = n \cdot m + n \wedge m \tag{37}$$

$$= \cos\frac{\alpha}{2} + \frac{n \wedge m}{||n \wedge m||} \sin\frac{\alpha}{2} \tag{38}$$

Since $n \wedge m/||n \wedge m||$ is the bivector b dual to the rotation axis (that is the rotation plane), the rotor R can be expressed as

$$R = \cos\frac{\alpha}{2} + b \sin\frac{\alpha}{2} = e^{b\frac{\alpha}{2}} \tag{39}$$

that is the standard formulation of the rotations as reported in Table 2. A specific function in the ConformalAPI library (the function `from_rot_axis_and_angle_to_refl_vecs` in Table 3) calculates the two planes or 5D vectors m and n with respect to which the two reflections have to be executed starting from the 3D rotation axis a and the rotation angle α.

The two vectors m and n have to lie in the plane orthogonal to the rotation axis a, while the angle between m and n has to be $\alpha/2$. The API function `orthonormal_ref` (Table 3) is first used to calculate an orthonormal basis $B = (u, v, w)$, where w has the same direction of a. Then, the two vectors $m_B = (1, 0, 0)$ and $n_B = (\cos(\alpha/2), \sin(\alpha/2), 0)$ with respect to the basis B are considered. These two vectors are converted into the global coordinate system by the following formulas:

$$m = A^T m_B^T, \tag{40}$$

$$n = A^T n_B^T, \tag{41}$$

where $A = [u, v, w]$. Finally, these two 3D vectors are extended to two 5D vectors setting to zero the two remaining coordinates. Another API function (the function `rotation` in Table 3) executes the rotation operation. This function receives the 3D rotation axis a and the rotation angle α as arguments and returns the rotated 5D object. The function `rotation` uses the function `from_rot_axis_and_angle_to_refl_vecs`

to calculate the two 5D reflection planes starting from the 3D rotation axis a and the rotation angle α and then executes the rotation using twice the fast reflection operator. The latter operator is implemented in the API function `reflection` reported in Table 3.

The distance calculations between the centers of the spheres are executed by exploiting the formula introduced in Section 4.1 (see Eqn. (28)) and implemented in the API function `distance` (Table 3). To have an idea of the computational load reduction achieved by our method, we can observe that the registration of the two models shown in Fig. 10, which requires 55,860 roto-translations and 24,464 dilations, needs 3,002,980 products and 4,036,592 sums/differences, whereas the standard CGA method would require 30,239,780 products and 26,241,516 sums/differences.

5. Experimental results

The proposed methods have been validated by several experimental tests executed on both MR and CT images. Several medical datasets collected at the University of Palermo, Policlinico Hospital, have been used in the experiments, including 15 CT abdominal datasets each composed of a number of slices (between 226 and 573); 15 CT brain datasets each composed of a number of slices (between 30 and 248); 20 MR brain datasets each composed of a number of slices (between 24 and 248). The total number of medical images processed in the experimental tests was 8080 (3600 MR images and 4480 CT images).

5.1. Performance comparison with standard methods. The proposed algorithms based on the new formulation of CGA have been also validated by accurate comparisons with state-of-the-art medical imaging methods, including the standard CGA-based methods. Tests have been executed on a 2.9 GHz Intel Core i7 CPU.

5.1.1. CGA-based segmentation method. The segmentation technique introduced in Section 4.1 has been compared with a standard region growing method as well as with the GGVF-snake method. Test medical images have been segmented by using both the standard techniques and the new proposed method. Figure 7 shows the results obtained by applying the CGA-based segmentation technique as well as the standard methods to a CT abdominal image with a liver lesion. Figure 8 shows the case of an object with very blurred contours in a CT liver image. The figure shows that the standard GGVF-snake algorithm is not able to correctly extract the shape of the object, neither when the initial snake is given outside the object contour (Fig. 8(d)), nor when the initial snake is given over the object contour (Fig. 8(f)); the CGA-based method gives, conversely, a well-segmented shape as output (Fig. 8(h)).

A further quantitative comparison has been performed between the proposed method and the standard techniques. The following metrics have been used for quantitative validation of the segmentation results:

$$\text{DiceIndex}(N_M, N_A) = \frac{2|N_M \cap N_A|}{|N_M| + |N_A|}, \tag{42}$$

$$\text{Sensitivity} = \frac{N_{TP}}{|N_M|}, \tag{43}$$

$$\text{Specificity} = 1 - \frac{N_{FP}}{|N_A|}, \tag{44}$$

where N_M is the manual segmented area, N_A is the automatic segmented area, N_{TP} is the number of automatic true positive pixels, and N_{FP} is the number of automatic false positive pixels. The manual segmented area was obtained starting from the manual delineation performed by agreement between three different radiologists. Table 4 lists the calculated metrics for the image of Fig. 7 as well as the average values for all the medical images used in the experiments. It can be observed that the reformulated CGA-based segmentation method shows a better performance than the standard algorithms in terms of all the metrics considered. Furthermore, our method is faster than the traditional techniques, as shown in Fig. 9 that reports a comparison of the different segmentation methods in terms of computational times.

5.1.2. Statistical analysis. In order to validate the proposed segmentation method, a statistical analysis was performed using the Wilcoxon signed-rank non-parametric test to detect significant statistical differences between the results of the segmentation methods based on the standard CGA and on the reformulated CGA, respectively. The Wilcoxon test was applied on the Dice index values related to the two above-mentioned segmentation methods for a dataset composed of 200 test images (50 CT abdominal images, 50 CT brain images, and 100 MR brain images). The statistical analysis was performed using the IBM SPSS software, while the significance level used was 5%. Statistical analysis results are listed in Tables 5 and 6. Table 5 shows results in terms of ranks, while Table 6 reports the test statistics (z-score and p-value). As reported in Table 6, the very low value of the statistical probability p ($p < 0.001$) indicates that the null hypothesis (that is, there are no significant differences among the results of the two compared methods) has to be rejected. The statistical analysis shows a significant advantage of the reformulated CGA method against the standard CGA method. From the statistical analysis

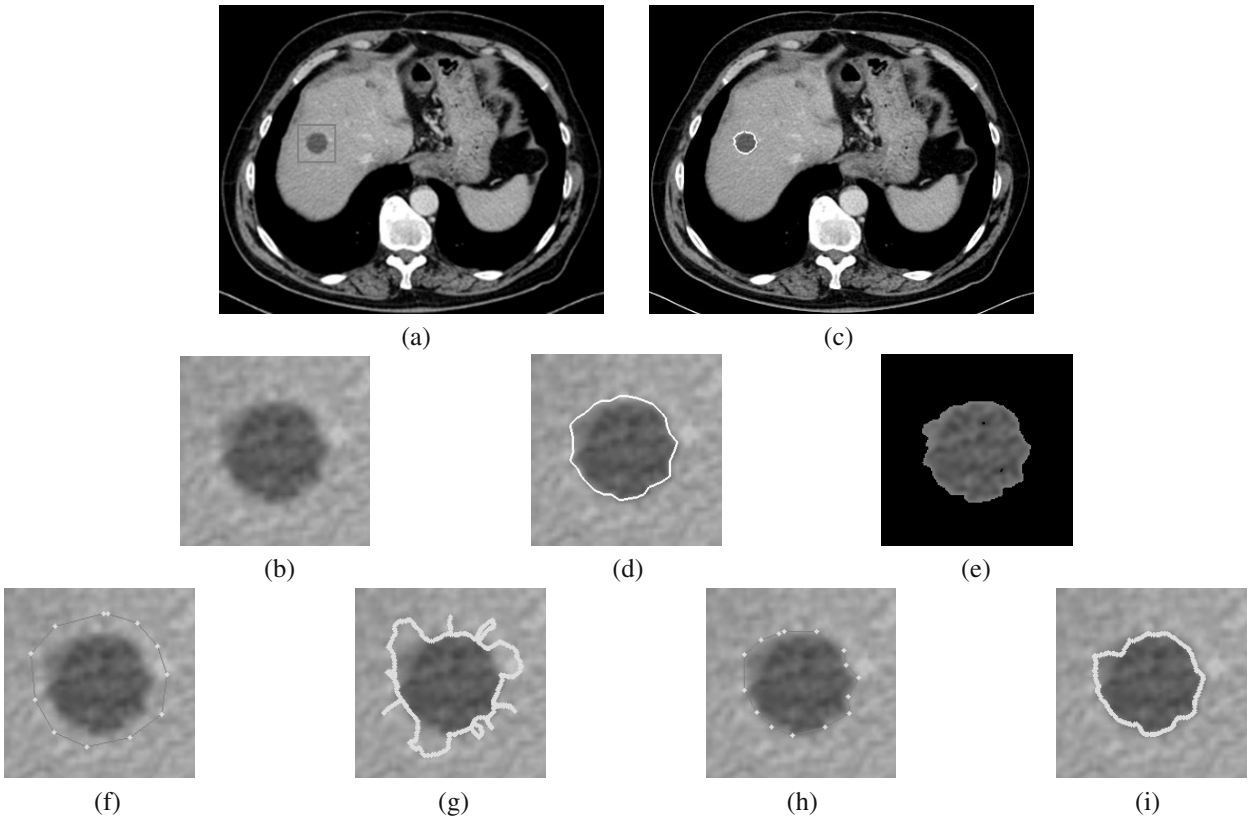


Fig. 7. Comparison between the proposed segmentation method and the standard methods for a CT abdominal image with a liver lesion: original image and ROI (a), zoom of (a) (b); CGA-based method (c)–(d): segmented object (c), zoom of (c) (d); region growing method (e); GGVF-snake method (f)–(i): initialization of the snake outside the object contour (f), segmented object with the initialization given in (f) (g), initialization of the snake over the object contour (h), segmented object with the initialization given in (h) (i).

results, it is evident that the proposed method not only produces higher quality segmented images, but also verifies its superior performance in a statistically meaningful way.

5.1.3. CGA-based registration methods. Once the 2D slices of the test medical datasets were segmented using the CGA-based segmentation technique, the 3D rendering method proposed in Section 4.2 was applied to the segmented images to obtain the related sphere-based 3D models. The obtained 3D surfaces were then used to test and validate the two registration methods proposed in Section 4.3. Several experiments were performed on different misaligned datasets with different degrees of deformation. In each experiment, the segmented 2D slices of the original CT or MR sequence and misaligned sequence, respectively, were first processed to obtain the two misaligned 3D models based on CGA spheres. Hereafter, the original model will be referred to as the expected model, while the misaligned model will be referred to as the initial model. The registration methods were then used to align the initial 3D model with the expected 3D model. For the quantitative validation of the registration methods, the following metric was used to measure the final error between the estimated model obtained after the algorithm execution and the expected model:

$$e = \sqrt{\frac{1}{n}\sum_{j,k=1}^{n} z_{jk}(S_j^E - S_k^A)^2}, \tag{45}$$

where $\{S_j^E\}$ and $\{S_k^A\}$ are the expected sphere-set and the approximated or estimated sphere-set, respectively, n is the number of pairs of the corresponding spheres of the two sphere-sets, and $z_{jk} = 1$ if S_k^A corresponds to S_j^E, and 0 otherwise.

With regard to the ICP algorithm, which treats rigid transformations, different linear transformations, each composed of a translation and a rotation, were applied to the 2D slices of the original set to obtain different misaligned sets with different degrees of misalignment to be registered with the original one. As remarked in Section 4.3, the ICP algorithm gives good results when small and approximately known transformations are considered.

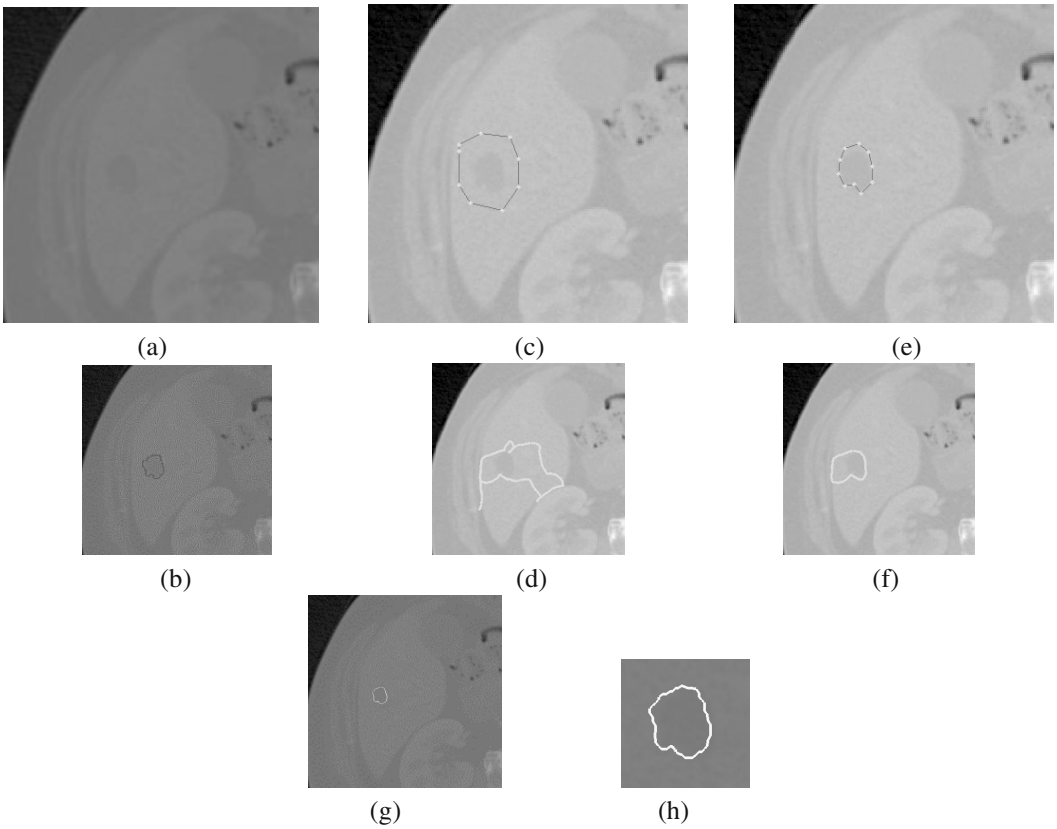


Fig. 8. Comparison between the standard GGVF-snake method (c)–(f) and the CGA-based segmentation method (g)–(h) for a CT liver image containing an object with very blurred contours: original image (a), original image with the object of interest marked by hand (b), initialization of the snake outside the object contour (c), result of the GGVF-snake method with the initialization given in (c) (d), initialization of the snake over the object contour (e), result of the GGVF-snake method with the initialization given in (e) (f), segmented object using the CGA-based method (g), zoom of (g) (h). The CGA-based method gives as result a well segmented object, while the standard GGVF-snake fails to segment the object whether the initialization is given outside or over the object contour.

As expected, the experimental tests have shown that the ICP algorithm is able to align two 3D models that differ by up to about 25 degrees in rotation and 25 mm in translation. Figure 10 shows the results obtained by applying the CGA-based ICP algorithm to register two 3D models related to a MR brain sequence composed of 175 slices that differ by 20 degrees in rotation and 20 mm in translation. The two 3D models have been obtained for $d = 8$, where d is the distance between two adjacent slices, which represents the resolution parameter of the rendered 3D model. The final error between the expected model and the approximated model after the algorithm execution, measured using Eqn. (45), is $e = 1.25 \cdot 10^{-5}$.

Figure 11 shows test results on the convergence of CGA-based ICP algorithm in the case of the same MR dataset. Each curve is related to a different value of d and shows the error e, calculated using Eqn. (45), against the number of iterations of the algorithm. It can be observed that, when d decreases, a higher number of iterations is required until the algorithm converges to the right matching. In all cases, the order of magnitude of the measured final error e is 10^{-5}.

Table 7 lists the number of spheres of the 3D models, the number of iterations of the CGA-based ICP algorithm, the error e, the number of CGA conformal operations required to align the two models, and the average execution time per iteration, for different values of the resolution parameter d. As expected, when d decreases, the number of spheres of the rendered model increases as well as the number of CGA conformal operations required to align the two models, and, consequently, the average time per iteration results to be higher.

Observing the experimental results reported in Fig. 11 and Table 7, we can conclude that a resolution parameter $d = 8$ is a good compromise between the rendering and alignment quality and the workload required by the algorithm, respectively.

The reformulated CGA-based ICP registration method shows better performance compared with the standard CGA-based ICP algorithm as well as with the

Table 4. Comparison between the reformulated CGA-based segmentation method and the standard algorithms (region growing, GGVF-snake, and the standard CGA-based method) in terms of the following statistics: Dice index, sensitivity, and specificity. The table lists the statistics for the test image in Fig. 7 (CT liver lesion) as well as average values for all test images.

		Region growing	GGVF-snake	Standard CGA	Reformulated CGA
CT liver lesion	Dice index	95.63%	96.04%	96.64%	**97.79%**
	sensitivity	97.20%	97.68%	97.51%	**98.36%**
	specificity	94.12%	95.19%	96.77%	**97.23%**
Average values for all test datasets	Dice index	95.54%	96.48%	97.10%	**98.14%**
	sensitivity	94.47%	95.89%	96.92%	**98.05%**
	specificity	96.98%	97.23%	97.41%	**97.73%**

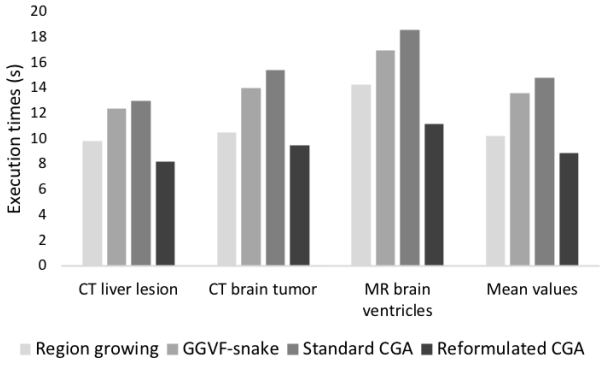


Fig. 9. Comparison between the reformulated CGA-based segmentation method and the conventional methods in terms of execution times. The graph reports the computation times related to three specific images as well as the average values over all the test medical images.

Table 5. Wilcoxon signed-rank test. Comparison between the standard CGA-based segmentation method and the reformulated CGA-based segmentation method: ranks.

	N	Mean rank	Sum of ranks
Negative ranks	15^a	74.67	1120.00
Positive ranks	185^b	102.59	18,980.00
Ties	0^c		
Total	200		

a Reformulated CGA $<$ Standard CGA
b Reformulated CGA $>$ Standard CGA
c Reformulated CGA $=$ Standard CGA

Table 6. Wilcoxon signed-rank test. Comparison between the standard CGA-based segmentation method and reformulated CGA-based segmentation method: test statistics.

z-value	-10.896^a
Exact p-value (2-tailed)	< 0.001

a Based on negative ranks

classical version of the ICP algorithm that uses linear algebra and matrix calculations for the execution of the geometric operations required during the registration process. The results of this comparison, in terms of both measured error and execution times, are summarized in Figs. 12(a) and (b), respectively, for three different MR and CT datasets. The same experiments have been repeated for all the test medical datasets showing in all cases that the reformulated CGA-based ICP algorithm allows for a reduction of about one order of magnitude of the error and a reduction of about 30% of the execution times with respect to the standard ICP method.

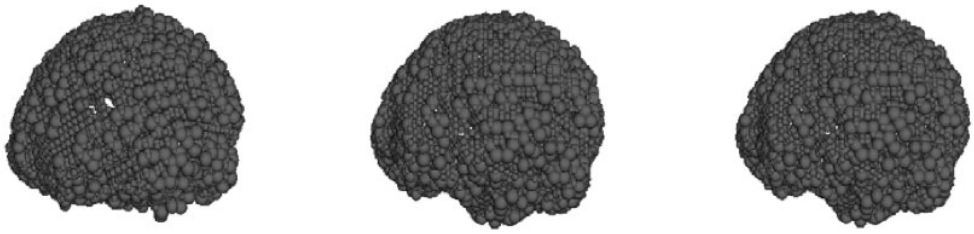
With regard to the TPS-RPM algorithm, it can be considered a non-rigid extension of the ICP algorithm. The CGA-based version of the TPS-RPM can be applied to align medical datasets that present different degrees of non-rigid deformation. Figure 13 shows the registration of two 3D models related to a liver lesion present in 15 slices within a CT abdomen scan using the CGA-based TPS-RPM method. The two 3D models have been obtained for $d = 1$. The final error between the expected model and the approximated model after the algorithm execution, measured using Eqn. (45), is $e = 2.07 \cdot 10^{-5}$.

To evaluate the CGA-based TPS-RPM performance, several experiments were performed on medical datasets with different degrees of warping. To generate the deformed datasets, we applied to each original dataset different randomly generated transformations using the Gaussian radial basis functions (RBF), instead of TPS, as non-rigid mapping. A Gaussian distribution with zero mean and standard deviation s was used to obtain the RBF coefficients. The value of s was gradually increased to generate larger deformations. The errors between the estimated model found by the algorithm and the expected model, measured for different degrees of warping, are reported in Figs. 14(a) and (b) for the MR brain dataset and the CT abdomen dataset (Figs. 10 and 13, respectively).

The same figures also show a comparison between the standard methods and our CGA-based methods: standard ICP, CGA-based ICP, standard TPS-RPM, and CGA-based TPS-RPM.

Table 7. Measured statistics on the CGA-based ICP algorithm for different values of the resolution parameter d.

d	No. of spheres	No. of iterations	Error (e)	No. of CGA roto-translations	Average time per iteration (s)
2	29,328	73	$1.16 \cdot 10^{-5}$	2,140,944	359
5	5,102	44	$1.20 \cdot 10^{-5}$	224,488	7.99
8	1,862	30	$1.25 \cdot 10^{-5}$	55,860	1.19
10	1,134	29	$1.48 \cdot 10^{-5}$	32,886	0.48
15	483	16	$1.79 \cdot 10^{-5}$	7,728	0.11
20	279	9	$1.91 \cdot 10^{-5}$	2,511	0.03



(a) (b) (c)

Fig. 10. Registration of two 3D models related to a MR brain sequence composed of 175 slices using the CGA-based ICP algorithm: initial model (a), expected model (b), approximated model (c). The 3D models have been obtained for $d = 8$, where d is the distance between two adjacent slices. The initial and expected models differ by 20 degrees in rotation and 20 mm in translation. The measured final error between the expected model and the estimated model after the algorithm execution is $e = 1.25 \cdot 10^{-5}$.

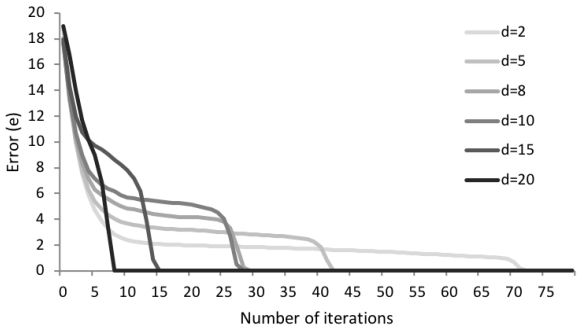


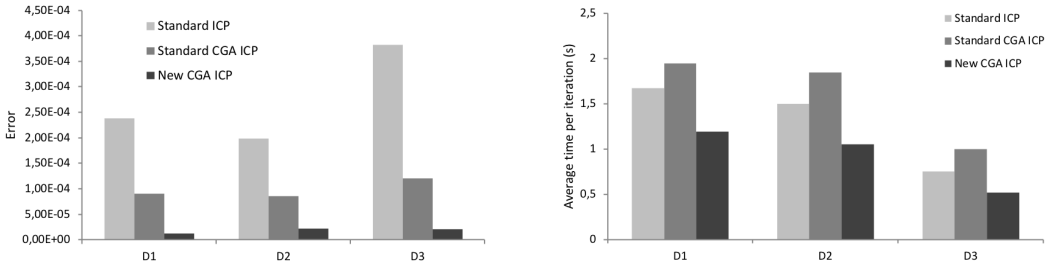
Fig. 11. CGA-based ICP algorithm: error vs. number of iterations for different values of the resolution parameter d.

It can be observed that the ICP performance deteriorates much faster than that for the TPS-RPM when the degree of warping increases. However, we can observe that the CGA-based ICP performance deteriorates slower than that for the standard ICP. A difference of about one order of magnitude is observed between the errors of the CGA-based TPS-RPM and the standard TPS-RPM, respectively. The same result can be observed comparing the CGA-based ICP with the standard ICP. Moreover, experimental tests show that the reformulated CGA-based TPS-RPM algorithm achieves better performance compared with both the standard CGA-based TPS-RPM and the conventional TPS-RPM. Figures 15(a) and (b) show the comparison in terms of both error and execution times. Results are reported for three different medical datasets. The same experiments were repeated for all the test medical datasets showing in all cases that the reformulated CGA-based TPS-RPM algorithm allows for a reduction of about one order of magnitude of the error and a reduction of about 20% of the execution times with respect to the standard TPS-RPM method.

Regarding the comparison between TPS-RPM and ICP, since the CGA-based ICP method presents shorter computation times than the CGA-based TPS-RPM method, it can be useful in time-critical applications when misalignments of the two datasets are small.

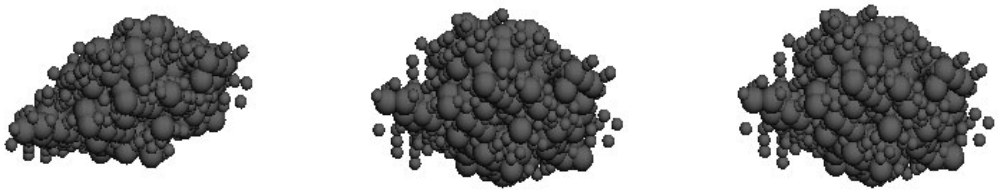
6. Discussion and conclusions

The comparative evaluation reported in the previous section shows that the reformulated CGA-based methods outperform the traditional techniques in terms of both precision and computation times. Regarding the segmentation, it has to be observed that, unlike the standard methods used for the comparative evaluation, the CGA-based segmentation method is based on fully automatic processes for both the selection of input contour points using the GGVF technique and the extraction of the object shape using a CGA-based GNG network. The only input to the algorithm is the image to be segmented and



(a) (b)

Fig. 12. Comparison between standard ICP, standard CGA-based ICP and new CGA-based ICP in terms of error (a) and execution times (b) for three different datasets (D1: MR brain with $d = 8$, D2: CT brain tumor with $d = 2$, D3: CT liver lesion with $d = 1$). Reported values are related to transformations composed of a rotation of 20 degrees and a translation of 20 mm.



(a) (b) (c)

Fig. 13. Registration of two 3D models related to a liver lesion present in 15 slices within a CT abdomen sequence using the CGA-based TPS-RPM algorithm. The two models have been derived from two CT sequences of the same patient captured in different moments: initial model (at time t_1) (a), expected model (at time t_2) (b), estimated model (c). The measured final error between the expected model and the estimated model after the algorithm execution is $e = 2.07 \cdot 10^{-5}$.

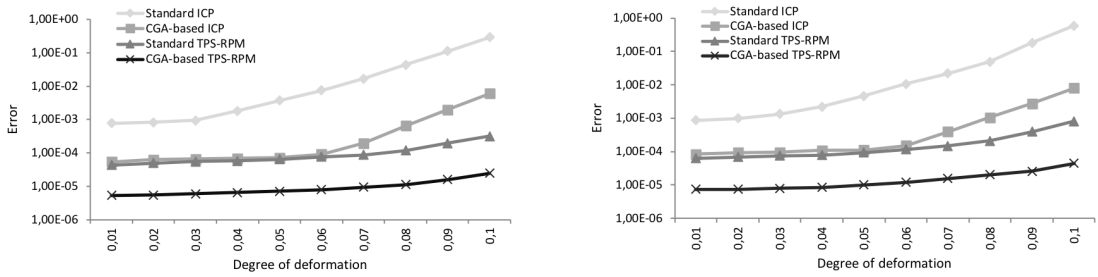
no other prior information is needed.

To experimentally validate the proposed segmentation method, we have used a large set of medical images with different features in terms of imaging modality (TC and RM), image quality and resolution (in the test set, image resolution ranges from 128×128 to 512×512 pixels). We have studied the influence of the image quality and resolution on the segmentation results, in terms of both segmentation precision and computation times. Starting from the experimental results, we have observed that, when the image resolution is higher, the computation time increases, but the segmentation quality, in terms of measured parameters (Dice index, sensitivity, and specificity) increases as well. Furthermore, the segmentation time is also affected by the particular shape to be extracted from the medical image.

Regarding the influence of the image quality on the segmentation results, we found that our CGA-based segmentation algorithm gives good results even in the presence of noisy or blurred images, as the image reported in Fig. 8, whereas the classical GGVF-snake method in most cases failed to extract the correct shape of the object of interest. Analyzing the Dice index, sensitivity and specificity values listed in Table 4, it can be observed that the reformulated CGA-based algorithm achieves an improvement also with respect to the method based on the standard CGA formulation. This result derives from the significant reduction of the number of basic arithmetic operations required to execute geometric transformations (see the computational complexity analysis reported in Fig. 1) that results in reduced propagation of rounding errors. Thanks to this higher precision, a lower number of iterations of the algorithm is required that leads to better results of the reformulated CGA technique with respect to the standard CGA method also in terms of execution times.

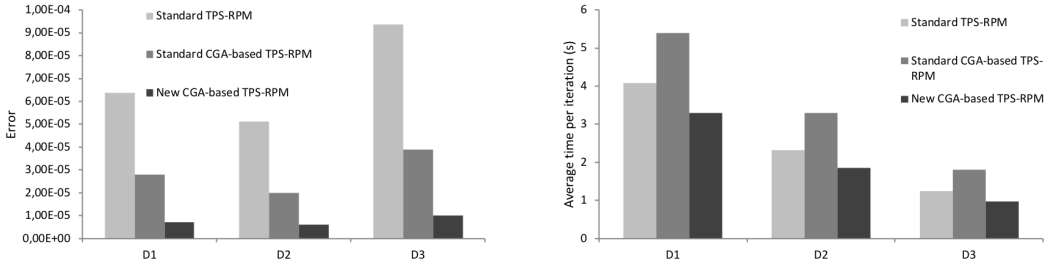
As expected and as reported in Fig. 9, the lower computational load of the reformulated CGA method leads to reduced computation times with respect to the standard CGA method, as well as with respect to the other conventional algorithms. Regarding the registration algorithms, we found that our new formulation of the ICP and TPS-RPM methods leads to better results compared with the standard CGA formulation as well as with the classical version based on the conventional linear algebra and matrix calculations.

Analyzing the results reported in Fig. 12, it can be also observed that the standard CGA ICP algorithm allows for a better precision, but requires longer computation times with respect to the conventional ICP, whereas the reformulated CGA ICP method shows better performance in terms of both precision and execution times. A similar



(a) (b)

Fig. 14. Error for different degrees of warping measured using different methods: standard ICP, CGA-based ICP, standard TPS-RPM, and CGA-based TPS-RPM: test results on the MR brain sequence (a), test results on the CT abdomen sequence (b).



(a) (b)

Fig. 15. Comparison between standard TPS-RPM, standard CGA-based TPS-RPM and new CGA-based TPS-RPM in terms of error (a) and execution times (b) for three different datasets (D1: MR brain with $d = 8$, D2: CT brain tumor with $d = 2$, D3: CT liver lesion with $d = 1$). Reported values are related to a medium degree of warping.

result has been observed also for the new formulation of the TPS-RPM algorithm. Also in this case, the standard CGA TPS-RPM is more precise, but slower than the conventional TPS-RPM, while the reformulated CGA TPS-RPM shows better results than the traditional method in terms of both precision and computation speed (see Fig. 15). We can therefore conclude that our new fast formulation of CGA operations makes the CGA-based medical imaging methods effective and practically usable in real applications.

Our experimental results demonstrate that it is possible to implement a medical image processing chain based on CGA algorithms that outperforms the conventional medical imaging approaches. The proposed methods are therefore well suited to big data analysis and processing in medical imaging. Furthermore, the ConformalAPI presented in the paper is actually a general-purpose software library that exploits the new CGA operators to offer a fast execution of the main Euclidean geometric transformations. The use of the ConformalAPI can be therefore easily extended to other geometry-based medical applications, including medical robotics, computer-assisted surgery and other medical imaging methods.

References

Ashdown, M. (2018). GA package for Maple, `http://www.mrao.cam.ac.uk/~maja1/software/GA/`.

Batard, T., Berthier, M. and Saint-Jean, C. (2010). Clifford Fourier transform for color image processing, *in* E.J. Bayro-Corrochano and G. Scheuermann (Eds), *Geometric Algebra Computing in Engineering and Computer Science*, Springer, Berlin, pp. 135–161.

Bayro-Corrochano, E. and Rivera-Rovelo, J. (2009). The use of geometric algebra for 3D modeling and registration of medical data, *Journal of Mathematical Imaging and Vision* **34**(1): 48–60.

Besl, P.J. and McKay, N.D. (1992). A method for registration of 3D shapes, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **14**(2): 239–256.

Clifford, W.K. (1882). On the classification of geometric algebras, *in* R. Tucker (Ed.), *Mathematical Papers*, Macmillian, London, pp. 397–401.

Dorst, L., Fontijne, D. and Mann, S. (2007). *Geometric Algebra for Computer Science: An Object Oriented Approach to Geometry*, Morgan Kaufmann, Burlington, MA.

Ebling, J. and Scheuermann, G. (2005). Clifford Fourier transform on vector fields, *IEEE Transactions on Visualization and Computer Graphics* **11**(4): 469–479.

Fabijańska, A., Węgliński, T., Zakrzewski, K. and Nowosławska, E. (2014). Assessment of hydrocephalus in children based on digital image processing and analysis, *International Journal of Applied Mathematics and Computer Science* **24**(2): 299–312, DOI: 10.2478/amcs-2014-0022.

Fontijne, D. (2006). Gaigen 2: A geometric algebra implementation generator, *Proceedings of the 5th International Conference on Generative Programming and Component Engineering, GPCE 2006, Portland, OR, USA*, pp. 141–150.

Franchini, S., Gentile, A., Sorbello, F., Vassallo, G. and Vitabile, S. (2008). An FPGA implementation of a quadruple-based multiplier for 4D Clifford algebra, *Proceedings of the 11th IEEE Euromicro Conference on Digital System Design—Architectures, Methods and Tools (DSD 2008), Parma, Italy*, pp. 743–751.

Franchini, S., Gentile, A., Sorbello, F., Vassallo, G. and Vitabile, S. (2011). Fixed-size quadruples for a new, hardware-oriented representation of the 4D Clifford algebra, *Advances in Applied Clifford Algebras* **21**(2): 315–340.

Franchini, S., Gentile, A., Sorbello, F., Vassallo, G. and Vitabile, S. (2012). Design space exploration of parallel embedded architectures for native Clifford algebra operations, *IEEE Design and Test of Computers* **29**(3): 60–69.

Franchini, S., Gentile, A., Sorbello, F., Vassallo, G. and Vitabile, S. (2013). Design and implementation of an embedded coprocessor with native support for 5D, quadruple-based Clifford algebra, *IEEE Transactions on Computers* **62**(12): 2366–2381.

Franchini, S., Gentile, A., Sorbello, F., Vassallo, G. and Vitabile, S. (2015). ConformalALU: A conformal geometric algebra coprocessor for medical image processing, *IEEE Transactions on Computers* **64**(4): 955–970.

Gentile, A., Segreto, S., Sorbello, F., Vassallo, G., Vitabile, S. and Vullo, V. (2005). CliffoSor: A parallel embedded architecture for geometric algebra and computer graphics, *Proceedings of the IEEE International Workshop on Computer Architecture for Machine Perception (CAMP 2005), Palermo, Italy*, pp. 90–95.

Hestenes, D. (1986). *New Foundations for Classical Mechanics*, Kluwer Academic, Dordrecht.

Hestenes, D. and Sobczyk, G. (1987). *Clifford Algebra to Geometric Calculus: A Unified Language for Mathematics and Physics*, Kluwer Academic, Dordrecht.

Hildenbrand, D. (2018). *Introduction to Geometric Algebra Computing*, Chapman and Hall/CRC, Boca Raton, FL.

Hitzer, E. and Sangwine, S. (2018). Clifford Multivector Toolbox, A toolbox for computing with Clifford algebras in Matlab, `https://sourceforge.net/projects/clifford-multivector-toolbox/`.

Hrebień, M., Steć, P., Nieczkowski, T. and Obuchowicz, A. (2008). Segmentation of breast cancer fine needle biopsy cytological images, *International Journal of Applied Mathematics and Computer Science* **18**(2): 159–170, DOI: 10.2478/v10006-008-0015-x.

Lasenby, J., Lasenby, A.N., Doran, C.J.L., and Fitzgerald, W.J. (1998). New geometric methods for computer vision: An application to structure and motion estimation, *International Journal of Computer Vision* **26**(3): 191–213.

Menneson, J., Saint-Jean, C. and Mascarilla, L. (2011). Color object recognition based on a Clifford Fourier transform, *in* L. Dorst and J. Lasenby (Eds), *Guide to Geometric Algebra in Practice*, Springer, Berlin, pp. 175—-191.

Mishra, B., Wilson, P. and Wilcock, R. (2015). A geometric algebra coprocessor for color edge detection, *Electronics* **4**(1): 94–117.

Newman, T.S. and Yi, H. (2006). A survey of the marching cubes algorithm, *Computers & Graphics* **30**(5): 854–879.

Ranjan, V. and Fournier, A. (1995). Union of Spheres (UoS) model for volumetric data, *Proceedings of the 11th Annual Symposium on Computational Geometry, Vancouver, BC, Canada*, pp. 402–403.

Rivera-Rovelo, J. and Bayro-Corrochano, E. (2006). Medical image segmentation using a self-organizing neural network and Clifford geometric algebra, *International Joint Conference on Neural Networks, IJCNN 2006, Vancovver, BC, Canada*, pp. 3538–3545.

Rivera-Rovelo, J. and Bayro-Corrochano, E. (2007). Surface approximation using growing self-organizing nets and gradient information, *Applied Bionics and Biomechanics* **4**(3): 125–136.

Sommer, G. (2001). *Geometric Computing with Clifford Algebras: Theoretical Foundations and Applications in Computer Vision and Robotics*, Springer, Berlin.

Stefanowski, J., Krawiec, K. and Wrembel, R. (2017). Exploring complex and big data, *International Journal of Applied Mathematics and Computer Science* **27**(4): 669–679, DOI: 10.1515/amcs-2017-0046.

Zhang, Z. (1994). Iterative point matching for registration of free-form curves, *International Journal of Computer Vision* **13**(2): 119–152.

Silvia Franchini holds a *laurea* degree (2004) in electrical engineering and a doctoral degree (2009) in computer engineering from the University of Palermo. She is currently a postdoctoral research assistant at the Department of Biomedicine, Neuroscience and Advanced Diagnostics of the University of Palermo. Her research interests include specialized architectures design and prototyping, embedded systems, and innovative graphic processors.

Antonio Gentile received his PhD degree in electrical and computer engineering from the Georgia Institute of Technology in 2000. He is currently an associate professor in the Department of Industrial and Digital Innovation of the University of Palermo, Italy. He is also the CEO and the founder of InformAmuse S.r.l., an academic spin-off of the same university. His research interests include high-throughput portable processing systems, image and video processing architectures, embedded systems and architectures, speech processing, human-computer interfaces, and mobile computing.

Giorgio Vassallo received his *laurea* degree in physics from the University of Palermo in 1982. He is currently a research scientist at the Department of Industrial and Digital Innovation of the University of Palermo, Italy. His research interests include innovative graphic processors, neural networks, geometric techniques for data mining, and natural language processing.

Salvatore Vitabile is currently an associate professor with the Department of Biomedicine, Neuroscience and Advanced Diagnostics, University of Palermo, Italy. He has co-authored more than 200 scientific papers in journals and conferences and chaired, organized, and served as member of the organizing committee of several international conferences and workshops. His research interests include specialized architecture design and prototyping, biometric authentication systems, driver assistance systems, and medical data processing and analysis.

Received: 14 October 2019
Revised: 14 April 2020
Accepted: 29 May 2020