Engineering Graphics in Geometric Algebra

Alyn Rockwood, Dietmar Hildenbrand

Abstract We illustrate the suitability of geometric algebra for representing structures and developing algorithms in computer graphics, especially for engineering applications. A number of example applications are reviewed. Geometric algebra unites many underpinning mathematical concepts in computer graphics such as vector algebra and vector fields, quaternions, kinematics and projective geometry, and it easily deals with geometric objects, operations and transformations. Not only are these properties important for computational engineering, but also for the computational point-of-view they provide. We also include the potential of geometric algebra for optimizations and highly efficient implementations.

1 Introduction

Computer graphics relies heavily on geometric models and methods. Geometric algebra is a mathematical framework to easily describe geometric concepts and operations. It allows us to develop algorithms fast and in an intuitive way. Geometric algebra is based on the work of Hermann Grassmann (see the conference [47] celebrating his 200th birthday in 2009) and William Clifford ([14], [15]). Pioneering work has been done by David Hestenes, who first applied geometric algebra to problems in mechanics and physics [27] [25].

Alyn Rockwood
King Abdullah University of Science and Technology, Saudi Arabia
Geometric Modeling and Scientific Visualization Research Center
e-mail: alyn.rockwood@kaust.edu.sa

Dietmar Hildenbrand
University of Technology Darmstadt, Germany
Interactive Graphics Systems Group
e-mail: dhilden@gris.informatik.tu-darmstadt.de

2 Benefits of geometric algebra for computational engineering

We first highlight some of the properties of geometric algebra that make it advantageous for graphics engineering applications.

2.1 Unification of mathematical systems

In the wide range of engineering applications many different mathematical systems are currently used. One notable advantage of geometric algebra is that it subsumes mathematical systems like vector algebra, complex analysis, quaternions, Plucker coordinates and tensor analysis. Applications described in Section 3 will illustrate this advantage.

2.2 Uniform handling of different geometric primitives

Conformal geometric algebra, the geometric algebra of conformal space we focus on, is able to treat different geometric objects such as points, vectors, lines, circles, spheres, and planes as the same entities algebraically. Consider the spheres of Fig. 1,

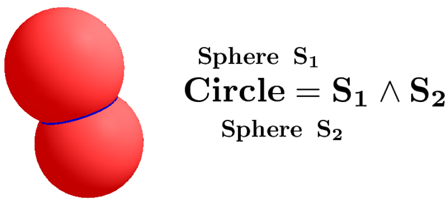

Fig. 1 Spheres and circles are basic entities of geometric algebra. Operations like the intersection of two spheres are easily expressed.

for instance. These spheres are simply represented by

$$S = P - \frac{1}{2}r^2 e_\infty \tag{1}$$

based on their center point P, their radius r and the basis vector e_∞ which represents the point at infinity. The circle of intersection of the spheres is then easily computed using the outer product to operate on the spheres as simply as if they were vectors.

$$Z = S_1 \wedge S_2 \tag{2}$$

This way of computing with geometric algebra clearly benefits applications like kinematics, pose estimation and other computer graphics applications as seen in Section 3.

2.3 Simplified rigid body motion

Rigid body motions in geometric algebra can be described with one compact linear expression, the so-called *screw*

$$S = \mathrm{i}\mathbf{m} + e_\infty \mathbf{n} \tag{3}$$

with the Euclidean pseudoscalar $\mathrm{i} = e_1 \wedge e_2 \wedge e_3$ includes both rotational and linear parts described with the 3D vectors **m** and **n** (see [26]). The combinations of rotational and linear velocities as well as of forces and torques are also described with the help of one linear expression.

One result of this property is the improvement of Finite Element methods [9].

2.4 Curl, vorticity and rotation

The vector algebra concepts of curl, vorticity and rotation as expressed in geometric algebra are defined in any dimension, whereas the cross-product in classical vector algebra is restricted to three-dimensions. Thus geometric calculus enables vector algebra applications to be considered in any dimensions [13].

2.5 More efficient implementations

Geometric algebra as a mathematical language often suggests a clearer structure and greater elegance in understanding methods and formulae. This regularly results in more efficiency and lower runtime performance for derived algorithms. In section 5 we present a dramatically improved optimization approach for kinematics. We will see there that geometric algebra inherently has a large potential for creating optimizations leading to more highly efficient implementations.

3 Some applications

Computer graphics, and the related areas of robotics and computer vision are active areas of research in geometric algebra. In this section we survey some of these applications in more detail.

For about a decade, researchers at the University of Cambridge, UK have applied geometric algebra to a number of graphics related projects. They started with ideas in computer vision.. Lasenby et al. and Perwass et al. present some applications dealing with structure and motion estimation as well as with the trifocal tensor in the articles [32], [34] and [44, 45, 39]. Rigid-body pose and position interpolation, mesh deformation and catadioptric cameras articles using geometric algebra are presented by Cameron et al. [12] and Wareham et al. [54], [55]. Geomerics [1] is a start-up company in Cambridge specializing in simulation software for physics and lighting, which presented its new technology allowing real-time radiosity in videogames utilizing commodity graphics processing hardware. The technology is based on geometric algebra wavelet technology.

Dorst et al. at the University of Amsterdam, the Netherlands, are applying their fundamental research on geometric algebra [16, 18, 20, 35, 36] mainly to 3D computer vision. Zaharia et al. investigated modeling and visualization of 3D polygonal mesh surfaces using geometric algebra [56]. Currently D. Fontijne is primarily focusing on the efficient implementation of geometric algebra. He investigated the performance and elegance of five models of 3D Euclidean geometry in a ray tracing, an archetypical computer graphics application [24]. It summarized the investigation by noting that 5D conformal space was the most elegant, but required appropriate hardware to become the most efficient as current hardware supported the 4D affine model. Along this line, research into hardware for geometric algebra continues. The Amsterdam group developed a code generator for geometric algebras [23]. Also, there is a book with applications of geometric algebra edited by Dorst et al. [17]. A new book was published recently [19], which dedicates a major portion of the book to the issue of geometric algebra calculation.

The first time geometric algebra was introduced to a wider Computer Graphics audience, was through a couple of courses at the SIGGRAPH conferences 2000 and 2001 (see [37]).

Bayro-Corrochano et al. from Guadalajara, Mexico are primarily dealing with the application of geometric algebra in the field of computer vision, robot vision and kinematics. They are using geometric algebra for instance for tasks like visual guided grasping, camera self-localization and reconstruction of shape and motion [4]. Their methods for geometric neural computing are used for tasks like pattern recognition ([21], [2]). Registration, the task of finding correspondences between two point sets, is solved based on geometric algebra methods in [33]. Some of their kinematics algorithms can be found in [5] for the 4D motor algebra and in the conformal geometric algebra papers [7, 8] dealing with inverse kinematics, fixation and grasping as well as with kinematics and differential kinematics of binocular robot heads. Books from Bayro-Corrochano et al. with geometric algebra applications are, for instance, [3] and [6].

At the University of Kiel, Germany, Sommer et al. are applying geometric algebra to robot vision [52], e.g. Rosenhahn et al. concerning pose estimation [49, 50] and Sommer et al. regarding the twist representation of free-form objects [53]. Perwass et al. are applying conformal geometric algebra to uncertain geometry with circles, spheres and conics [42], to geometry and kinematics with uncertain data [43] or concerning the inversion camera model [46]. There is a book with applications of geometric algebra edited by Sommer [51] and a new book about the application of geometric algebra in engineering applications by Christian Perwass [41]. Sven Buchholz together with Kanta Tachibana from the university of Nagoya and Eckhard Hitzer from the university of Fukui, Japan do some interesting research dealing for instance with neural networks based on geometric algebra ([11], [10]).

In addition to these examples there are many other applications like geometric algebra fourier transforms for the visualization and analysis of vector fields [22] or classification and clustering of spatial patterns with geometric algebra [48] showing the wide area of possibilities of advantageously using this mathematical system in engineering applications.

4 The geometric primitives in more detail

Table 1 List of the basic geometric primitives provided by the 5D conformal geometric algebra. The bold characters represent 3D entities ($\mathbf{x}$ is a 3D point, $\mathbf{n}$ is a 3D normal vector and $\mathbf{x}^2$ is the scalar product of the 3D vector $\mathbf{x}$). The two additional basis vectors e_0 and e_∞ represent the origin and infinity. Based on the outer product, circles and lines can be described as intersections of two spheres, respectively two planes. The parameter r represents the radius of the sphere and the parameter d the distance of the plane to the origin.

entity	representation
Point	$P = \mathbf{x} + \frac{1}{2}\mathbf{x}^2 e_\infty + e_0$
Sphere	$s = P - \frac{1}{2}r^2 e_\infty$
Plane	$\pi = \mathbf{n} + d e_\infty$
Circle	$z = s_1 \wedge s_2$
Line	$l = \pi_1 \wedge \pi_2$

Here, we look into some details of the basic geometric primitives of conformal geometric algebra as introduced in section 2.2 and listed in table 1. We especially look into the representations of spheres and planes and will see that planes are specific spheres with infinite radius. Increasing the radius of a sphere to infinity, the resulting plane is described by

$$\pi = \mathbf{n} + d e_\infty \tag{4}$$

with $\mathbf{n}$ being the 3D unit normal vector of the plane, and d the distance of the plane from the origin. This limit process can be used in order to **fit the best suitable**

object into a set of points , whether it is a plane or a sphere. A locally estimated sphere can be used in order to describe local curvature of point clouds [29] while an estimation of a plane describes vanishing curvature.

4.1 Planes as a limit of spheres

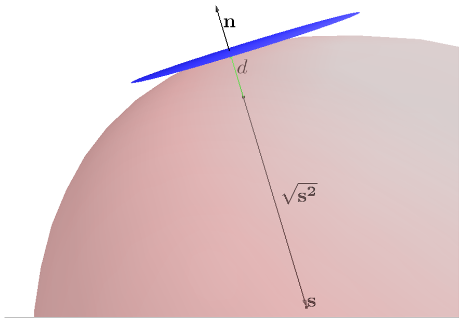

Fig. 2 A sphere with a center point **s** (in the opposite direction of a normal vector **n**) going to infinity (and always adapting its radius), at the end, results in a plane with normal vector **n** and distance d to the origin.

Spheres and planes, both, are vectors in conformal geometric algebra. In this section, we will see how a sphere

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

with Euclidean center point **s** and radius r degenerates to a plane as the result of a limit process.
According to the construction of Fig. 2, the minimum distance from the origin to the sphere, having its center in the opposite direction of a normal vector **n**, is

$$d = r - \sqrt{\mathbf{s}^2} \tag{6}$$

and the radius as the sum of the length of the 3D vector **s** and d

$$r = \sqrt{\mathbf{s}^2} + d \tag{7}$$

or

$$r^2 = \mathbf{s}^2 + 2d\sqrt{\mathbf{s}^2} + d^2. \tag{8}$$

Now, the sphere can be written as

$$S = \mathbf{s} + \frac{1}{2}(\mathbf{s}^2 - \mathbf{s}^2 - 2d\sqrt{\mathbf{s}^2} - d^2)e_\infty + e_0 \tag{9}$$

or equivalently

$$S = \mathbf{s} + \frac{1}{2}(-2d\sqrt{\mathbf{s}^2} - d^2)e_\infty + e_0. \tag{10}$$

Now, we introduce S' as a scaled version of the algebraic expression of sphere S representing geometrically the same sphere as

$$S' = -\frac{S}{\sqrt{\mathbf{s}^2}} = -\frac{\mathbf{s}}{\sqrt{\mathbf{s}^2}} + \frac{1}{2}(2d + \frac{d^2}{\sqrt{\mathbf{s}^2}})e_\infty - \frac{e_0}{\sqrt{\mathbf{s}^2}}. \tag{11}$$

Since the ratio of the 3D vector $\mathbf{s}$ and its length $\sqrt{\mathbf{s}^2}$ corresponds to the negative normal vector $\mathbf{n}$ (see the construction in figure 2)

$$\lim_{\mathbf{s}^2\to\infty} -\frac{S}{\sqrt{\mathbf{s}^2}} = \mathbf{n} + \lim_{\mathbf{s}^2\to\infty} \frac{1}{2}(2d + \frac{d^2}{\sqrt{\mathbf{s}^2}})e_\infty - \lim_{\mathbf{s}^2\to\infty} \frac{e_0}{\sqrt{\mathbf{s}^2}.} \tag{12}$$

This is equivalent to

$$\lim_{\mathbf{s}^2\to\infty} -\frac{S}{\sqrt{\mathbf{s}^2}} = \mathbf{n} + de_\infty \tag{13}$$

which is a representation of a plane with normal vector $\mathbf{n}$ and distance d to the origin.

4.2 Distances based on the inner product

Points, planes and spheres are represented as vectors (as listed in table 1). We will see that the inner products of these vectors describe distances between these geometric objects. The inner product between a vector P and a vector S is defined by

$$P \cdot S = (\mathbf{p} + p_4 e_\infty + p_5 e_o) \cdot (\mathbf{s} + s_4 e_\infty + s_5 e_o) \tag{14}$$

This corresponds to

$$\begin{aligned} P \cdot S = \mathbf{p}\cdot\mathbf{s} &+ s_4 \underbrace{\mathbf{p}\cdot e_\infty}_{0} + s_5 \underbrace{\mathbf{p}\cdot e_o}_{0} \\ &+ p_4 \underbrace{e_\infty\cdot\mathbf{s}}_{0} + p_4 s_4 \underbrace{e_\infty^2}_{0} + p_4 s_5 \underbrace{e_\infty\cdot e_o}_{-1} \\ &+ p_5 \underbrace{e_o\cdot\mathbf{s}}_{0} + p_5 s_4 \underbrace{e_o\cdot e_\infty}_{-1} + p_5 s_5 \underbrace{e_o^2}_{0} \end{aligned}$$

and based on the rules of conformal geometric algebra to

$$P \cdot S = \mathbf{p} \cdot \mathbf{s} - p_5 s_4 - p_4 s_5 \tag{15}$$

or

$$P \cdot S = p_1 s_1 + p_2 s_2 + p_3 s_3 - p_5 s_4 - p_4 s_5 \tag{16}$$

4.2.1 Distances between points

In the case of P and S being points we get

$$p_4 = \frac{1}{2}\mathbf{p}^2, \quad p_5 = 1$$

$$s_4 = \frac{1}{2}\mathbf{s}^2, \quad s_5 = 1$$

The inner product of these points is according to equation (15)

$$P \cdot S = \mathbf{p} \cdot \mathbf{s} - \frac{1}{2}\mathbf{s}^2 - \frac{1}{2}\mathbf{p}^2$$

$$= p_1 s_1 + p_2 s_2 + p_3 s_3 - \frac{1}{2}(s_1^2 + s_2^2 + s_3^2) - \frac{1}{2}(p_1^2 + p_2^2 + p_3^2)$$

$$= -\frac{1}{2}(s_1^2 + s_2^2 + s_3^2 + p_1^2 + p_2^2 + p_3^2 - 2p_1 s_1 - 2p_2 s_2 - 2p_3 s_3)$$

$$= -\frac{1}{2}((s_1 - p_1)^2 + (s_2 - p_2)^2 + (s_3 - p_3)^2)$$

$$= -\frac{1}{2}(\mathbf{s} - \mathbf{p})^2$$

We recognize that the square of the Euclidean distance of the inhomogenous points corresponds to the inner product of the homogenous points multiplied by -2.

$$(\mathbf{s} - \mathbf{p})^2 = -2(P \cdot S) \tag{17}$$

4.2.2 Distance between points and planes

For a vector P representing a point we get

$$p_4 = \frac{1}{2}\mathbf{p}^2, \quad p_5 = 1$$

For a vector S representing a plane with normal vector $\mathbf{n}$ and distance d we get

$$\mathbf{s} = \mathbf{n}, \quad s_4 = d, \quad s_5 = 0$$

The inner product of point and plane is according to equation (15)

$$P \cdot S = \mathbf{p} \cdot \mathbf{n} - d \tag{18}$$

representing the Euclidean distance of a point and a plane.

4.2.3 Distance between point and sphere

We will see now that the inner product of a point and a sphere can be used as a measure of distance between a point and a sphere even if it does not correspond to the minimal Euclidean distance between them.

For a vector P representing a point we get

$$p_4 = \frac{1}{2}\mathbf{p}^2, \quad p_5 = 1$$

For a vector S representing a sphere we get

$$s_4 = \frac{1}{2}(s_1^2 + s_2^2 + s_3^2 - r^2), \qquad s_5 = 1$$

The inner product of point and sphere is according to equation (15)

$$\begin{aligned} P \cdot S &= \mathbf{p} \cdot \mathbf{s} - \frac{1}{2}(\mathbf{s}^2 - r^2) - \frac{1}{2}\mathbf{p}^2 \\ &= \mathbf{p} \cdot \mathbf{s} - \frac{1}{2}\mathbf{s}^2 + \frac{1}{2}r^2 - \frac{1}{2}\mathbf{p}^2 \\ &= \frac{1}{2}r^2 - \frac{1}{2}(\mathbf{s}^2 - 2\mathbf{p} \cdot \mathbf{s} - \mathbf{p}^2) \\ &= \frac{1}{2}r^2 - \frac{1}{2}(\mathbf{s} - \mathbf{p})^2 \end{aligned}$$

Finally, we get

$$2(P \cdot S) = r^2 - (\mathbf{s} - \mathbf{p})^2 \tag{19}$$

Twice the inner product $P \cdot S$ equals to the square of the radius minus the square of the distance between the point **p** and the center point **s** of the sphere. Figure 3 describes this relation geometrically. Equation 19 can be rearranged to

$$(\mathbf{s} - \mathbf{p})^2 = r^2 - 2(P \cdot S) \tag{20}$$

describing the relations of the right angle triangle in the case a) with the point **p** being outside of the sphere, while the equation

$$r^2 = 2(P \cdot S) + (\mathbf{s} - \mathbf{p})^2 \tag{21}$$

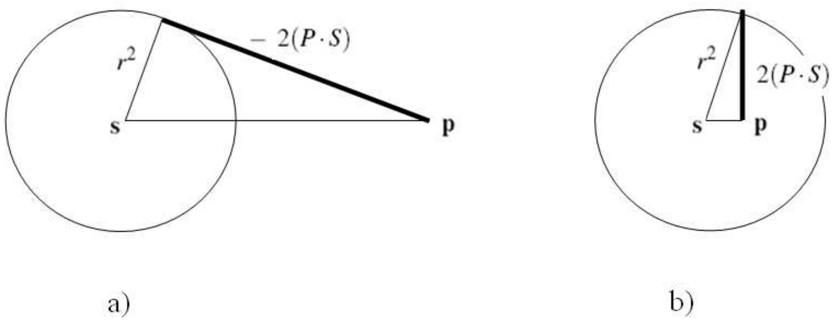

Fig. 3 The inner product of point and sphere [19], the bold segments describe the square root of the inner product depending on a) $(\mathbf{s}-\mathbf{p})^2 = r^2 - 2(P\cdot S)$, the point **p** lies outside of the sphere b) $r^2 = 2(P\cdot S) + (\mathbf{s}-\mathbf{p})^2$, the point **p** lies inside of the sphere

describes the relations of the right angle triangle in the case b) with the point **p** being inside of the sphere.

Please notice that, based on these observations, we can see that

$P\cdot S > 0$: **p** is inside of the sphere
$P\cdot S = 0$: **p** is on the sphere
$P\cdot S < 0$: **p** is outside of the sphere

4.3 Approximation of points with the help of planes or spheres

In this section, a point set $\mathbf{p_i} \in \mathbb{R}^3,\ i \in \{1,...,n\}$ will be approximated with the help of the best fitting plane or sphere. Please find also an approach for the fitting of circles into point sets in [42].

Plane and sphere in conformal space are vectors of the form

$$S = s_1e_1 + s_2e_2 + s_3e_3 + s_4e_\infty + s_5e_0 \tag{22}$$

while the points are specific vectors of the form

$$P_i = \mathbf{p_i} + \frac{1}{2}\mathbf{p_i}^2 e_\infty + e_0 \tag{23}$$

In order to solve the approximation problem we

- use the distance measure of the previous section between point and sphere/plane with the help of the inner product.

- make a least squares approach to minimize the squares of the distances between the points and the sphere/plane.
- solve the resulting eigenvalue problem.

4.3.1 Distance measure

From section 4.2.3 we already know that a distance measure between a point P_i and the sphere/plane S can be defined with the help of their inner product

$$P_i \cdot S = (\mathbf{p_i} + \frac{1}{2}\mathbf{p_i}^2 e_\infty + e_0) \cdot (\mathbf{s} + s_4 e_\infty + s_5 e_0) \tag{24}$$

According to equation 15, this results in

$$P_i \cdot S = \mathbf{p_i} \cdot \mathbf{s} - s_4 - \frac{1}{2} s_5 \mathbf{p_i}^2$$

or equivalently

$$P_i \cdot S = \sum_{j=1}^{5} w_{i,j} s_j \tag{25}$$

with

$$w_{i,k} = \begin{cases} p_{i,k} & : \ k \in \{1,2,3\} \\ -1 & : \ k = 4 \\ -\frac{1}{2}\mathbf{p_i}^2 & : \ k = 5 \end{cases}$$

4.3.2 Least squares approach

In the least-squares sense we consider the minimum of the sum of the squares of the distances (in terms of the inner product) between all the points and the plane/sphere

$$min \sum_{i=1}^{n} (P_i \cdot S)^2 \tag{26}$$

In order to obtain the minimum this can be rewritten in bilinear form to

$$min(s^T B s) \tag{27}$$

with

$$s^T = (s_1, s_2, s_3, s_4, s_5)$$

and the 5x5 matrix

$$B = \begin{pmatrix} b_{1,1} & b_{1,2} & b_{1,3} & b_{1,4} & b_{1,5} \\ b_{2,1} & b_{2,2} & b_{2,3} & b_{2,4} & b_{2,5} \\ b_{3,1} & b_{3,2} & b_{3,3} & b_{3,4} & b_{3,5} \\ b_{4,1} & b_{4,2} & b_{4,3} & b_{4,4} & b_{4,5} \\ b_{5,1} & b_{5,2} & b_{5,3} & b_{5,4} & b_{5,5} \end{pmatrix}$$

with entries

$$b_{j,k} = \sum_{i=1}^{n} w_{i,j} w_{i,k}$$

The matrix B is symmetric since $b_{j,k} = b_{k,j}$. We consider only normalized results $s^T s = 1$. A conventional approach to such a constrained optimization problem is to introduce

$$L = s^T Bs - 0 = s^T Bs - \lambda(s^T s - 1),$$
$$s^T s = 1,$$
$$B^T = B$$

Necessary conditions for a minimum are

$$0 = \nabla L = 2 \cdot (Bs - \lambda s) = 0$$

$$\rightarrow Bs = \lambda s$$

The solution of the minimization problem is given as the eigenvector of B that corresponds to the smallest eigenvalue.

The figures 4 and 5 discuss two properties of the distance measure of this approach dealing with the double squaring of the distance as well as with the limit process of the distance in the case of a plane as a sphere with infinite radius

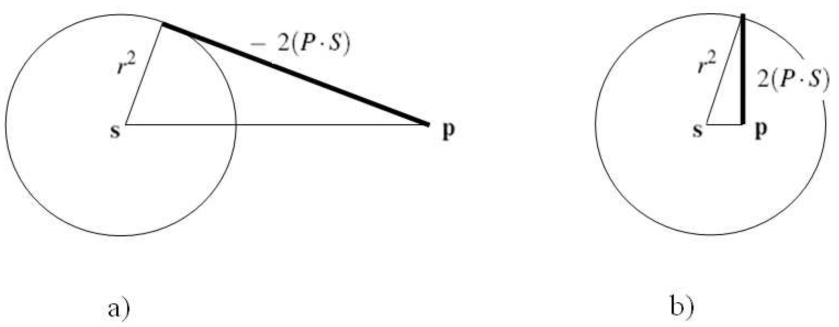

Fig. 4 The inner product of point and sphere on one hand describes already the square of a distance, but on the other hand has to be squared again in the least squares sense since the inner product can be positive or negative depending on a) the point **p** lies outside of the sphere b) the point **p** lies inside of the sphere

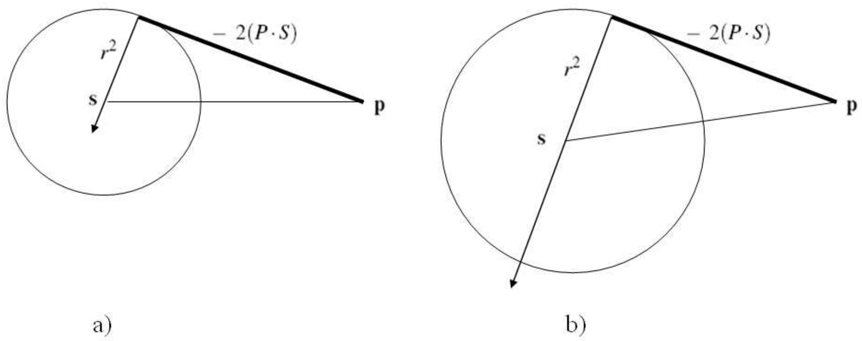

Fig. 5 The constraint $s^T s = 1$ leads implicitly to a scaling of the distance measure in order that it gets smaller with increasing radius, leading to a plane as a sphere with infinite radius

4.3.3 Example

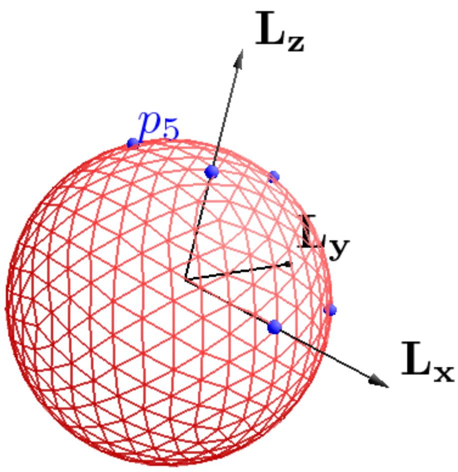

Fig. 6 Fitting a sphere into a set of 5 points

Three distinct (not co-linear) points are needed to describe a plane while four distinct (not co-planar) points exactly describe a sphere. In this example we use five points in order to demonstrate that our approach is really able to fit the best fitting objects, whether it is a sphere or a plane. First, let us have a look on an example with the following five points with four of them being co-planar:

Point	x	y	z
$\mathbf{p_1}$	1	0	0
$\mathbf{p_2}$	1	1	0
$\mathbf{p_3}$	0	0	1
$\mathbf{p_4}$	0	1	1
$\mathbf{p_5}$	-1	0	1

The least squares calculation results in

$$S = -0.301511e_1 + 0.301511e_2 - 0.301511e_3$$
$$-0.603023e_\infty + 0.603023e_0$$

Another scaled representation describing the same object is

$$S = -\frac{1}{2}e_1 + \frac{1}{2}e_2 - \frac{1}{2}e_3 - e_\infty + e_0$$

This corresponds to a sphere with the center point $\mathbf{s} = (0.5, 0.5, -0.5)$ and the square of the radius $r^2 = 2.75$ (see figure 6). Let us now change the fifth point in order that

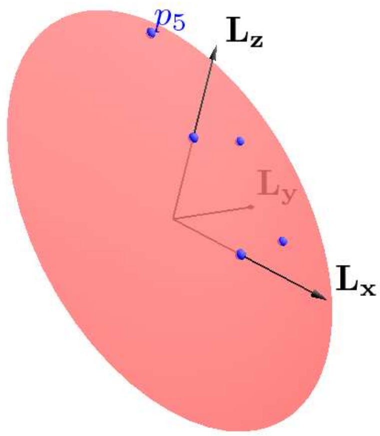

Fig. 7 Fitting a plane into a set of 5 points

all the points are within one plane.

Point	x	y	z
$\mathbf{p_1}$	1	0	0
$\mathbf{p_2}$	1	1	0
$\mathbf{p_3}$	0	0	1
$\mathbf{p_4}$	0	1	1
$\mathbf{p_5}$	-1	0	2

Now, the result is

$$S = 0.57735e_1 + 0.57735e_3 + 0.57735e_\infty$$

representing a plane according to figure 7.

5 Computational efficiency of geometric algebra using Gaalop

Since many of the applications depend on an appropriate calculation platform for geometric algebra, it is worth investigating one approach in some detail. Gaalop ([30],[31]) uses a two-stage approach for the automatic optimization of geometric algebra algorithms. In a first step they optimize geometric algebra algorithms with the help of symbolic computing. This kind of optimization results in very basic algorithms leading to high efficient software implementations. These algorithms, foster a high degree of parallelization which are then used for hardware optimizations in a second step.

They investigated performance issues with an inverse kinematics algorithm. Naively implemented, the first algorithm was slower than the conventional one. However, with the symbolic computation optimization approach the software implementation became three times faster [28] and with a hardware implementation about 300 times faster [30] (3 times by software optimization and 100 times by additional hardware optimization) than the conventional software implementation. This result served as a proof-of-concept for Gaalop [31].

Figure 8 shows an overview over the architecture of Gaalop. Its input is a geometric algebra algorithm written in CLUCalc (see [40]). Via symbolic simplification it is transformed into a generic intermediate representation (IR) that can be used for the generation of different output formats. Gaalop supports sequential platforms with the automatic generation of C and JAVA code while its main focus is on supporting parallel platforms like reconfigurable hardware as well as modern accelerating GPUs. FPGAs (field programmable gate arrays) are currently supported as a structural hardware description, written in the Verilog language. Thanks to the lower prices of powerful GPUs for instance based on the CUDA technology [38] from NVIDIA or on the future Larrarbee technology of INTEL, one can expect impressive results using the powerful language of geometric algebra .

One focus of Gaalop will lie on mixed solutions handling reasonable combinations of software and hardware implementations.

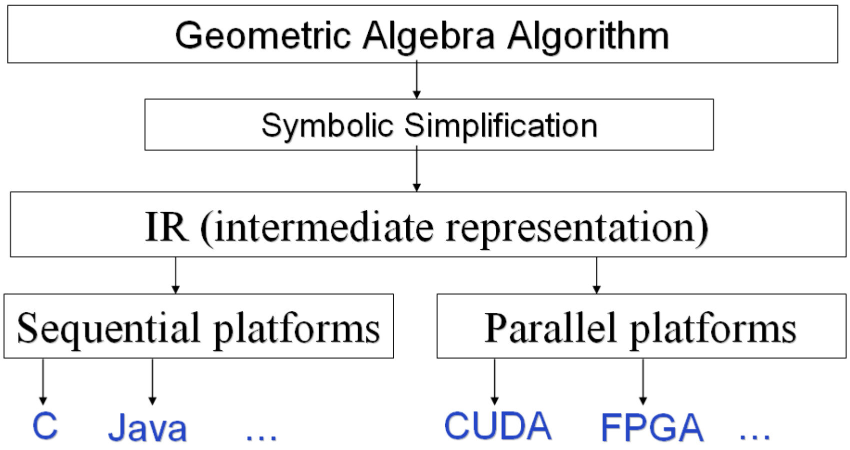

Fig. 8 Architecture of Gaalop

6 Conclusion

In this paper, we observed some properties of geometric algebra that have already proven helpful in computer graphics engineering applications. With these properties, together with the potential of being the base for highly efficient implementations using tools like Gaalop, we are convinced that geometric algebra will become more and more fruitful in a great variety of computational engineering applications. As a consequence, it is worth noting the benefits for students, researchers and practitioners with geometric algebra. From the educational point of view, students do not have to learn the different mathematical systems and the translations between them, rather they learn one global mathematical system. Researchers gain new insights into their research area using geometric algebra. Practitioners in the field of computational engineering benefit from the easy development, testing and maintenance of algorithms based on geometric algebra.

References

1. The homepage of geomerics ltd. HTML document http://www.geomerics.com
2. Bayro-Corrochano, E.: Geometric neural computing. IEEE Transactions on Neural Networks **12**(5), 968–986 (2001)
3. Bayro-Corrochano, E.: Robot perception and action using conformal geometry. In: E. Bayro-Corrochano (ed.) the Handbook of Geometric Computing. Applications in Pattern Recognition, Computer Vision, Neurocomputing and Robotics, chap. 13, pp. 405–458. Springer Verlag, Heidelberg (2005)
4. Bayro-Corrochano, E., Banarer, V.: A geometric approach for the theory and applications of 3d projective invariants. Journal of Mathematical Imaging and Vision **16**, 131–154 (2001)

5. Bayro-Corrochano, E., Daniilidis, K., Sommer, G.: Motor algebra for 3d kinematics : The case of the hand-eye calibration. Journal of Mathematical Imaging and Vision **13**, 79–99 (2000)
6. Bayro-Corrochano, E., Sobczyk, G. (eds.): Geometric Algebra with Applications in Science and Engineering. Birkhäuser (2001)
7. Bayro-Corrochano, E., Zamora-Esquivel, J.: Inverse kinematics, fixation and grasping using conformal geometric algebra. In: IROS 2004, September 2004, Sendai, Japan (2004)
8. Bayro-Corrochano, E., Zamora-Esquivel, J.: Kinematics and differential kinematics of binocular robot heads. In: proceedings of ICRA conference, Orlando, USA (2006)
9. Brendel, E., Kalbe, T., Hildenbrand, D., Schaefer, M.: Simulation of elastic rods using conformal geometric algebra. In: International Symposium on Frontiers of Computational Science, Nagoya, Japan (2008)
10. Buchholz, S., Hitzer, E., Tachibana, K.: Coordinate independent update formulas for versor Clifford neurons. In: Proc. Joint 4th International Conference on Soft Computing and Intelligent Systems and 9th International Symposium on advanced Intelligent Systems (SCIS and ISIS 2008). Nagoya, Japan (2008)
11. Buchholz, S., Tachibana, K., Hitzer, E.: Optimal learning rates for Clifford neurons. In: International Conference on Artificial Neural Networks, vol. 1, pp. 864–873. Porto, Portugal (2007)
12. Cameron, J., Lasenby, J.: Oriented conformal geometric algebra. Proceedings of ICCA7 (2005)
13. Cibura, C., Hildenbrand, D.: Geometric algebra approach to fluid dynamics. In: AGACSE conference Leipzig (2008)
14. Clifford, W.K.: Applications of grassmann's extensive algebra. In: R. Tucker (ed.) Mathematical Papers, pp. 266–276. Macmillian, London (1882)
15. Clifford, W.K.: On the classification of geometric algebras. In: R. Tucker (ed.) Mathematical Papers, pp. 397–401. Macmillian, London (1882)
16. Dorst, L.: Honing geometric algebra for its use in the computer sciences. In: G. Sommer (ed.) Geometric Computing with Clifford Algebra. Springer-Verlag (2001)
17. Dorst, L., Doran, C., Lasenby, J. (eds.): Applications of Geometric Algebra in Computer Science and Engineering. Birkhäuser (2002)
18. Dorst, L., Fontijne, D.: 3d euclidean geometry through conformal geometric algebra (a gaviewer tutorial). available from http://www.science.uva.nl/ga (2003)
19. Dorst, L., Fontijne, D., Mann, S.: Geometric Algebra for Computer Science, An Object-Oriented Approach to Geometry. Morgan Kaufman (2007)
20. Dorst, L., Mann, S.: Geometric algebra: a computational framework for geometrical applications (part i: algebra). Computer Graphics and Application **22**(3), 24–31 (2002)
21. E. Bayro-Corrochano, R.V.A., Arana-Daniel, N.: Geometric preprocessing, geometric feedforward neural networks and Clifford support vector machines for visual learning. Special issue of Journal Neurocomputing **67**, 54–105 (2005)
22. Ebling, J.: Clifford fourier transform on vector fields. IEEE Transactions on Visualization and Computer Graphics **11**(4), 469–479 (2005)
23. Fontijne, D., Bouma, T., Dorst, L.: Gaigen: A geometric algebra implementation generator. Available at http://www.science.uva.nl/ga/gaigen
24. Fontijne, D., Dorst, L.: Modeling 3D euclidean geometry. IEEE Computer Graphics and Applications **23**(2), 68–78 (2003)
25. Hestenes, D.: New Foundations for Classical Mechanics. Dordrecht (1986)
26. Hestenes, D., Fasse, E.: Homogeneous rigid body mechanics with elastic coupling. In: L. Dorst, C. Doran, J. Lasenby (eds.) Applications of Geometric Algebra in Computer Science and Engineering. Birkhäuser (2002)
27. Hestenes, D., Sobczyk, G.: Clifford Algebra to Geometric Calculus: A Unified Language for Mathematics and Physics. Dordrecht (1984)
28. Hildenbrand, D., Fontijne, D., Wang, Y., Alexa, M., Dorst, L.: Competitive runtime performance for inverse kinematics algorithms using conformal geometric algebra. In: Eurographics conference Vienna (2006)

29. Hildenbrand, D., Hitzer, E.: Analysis of point clouds using conformal geometric algebra. In: GRAPP conference Madeira (2008)
30. Hildenbrand, D., Lange, H., Stock, F., Koch, A.: Efficient inverse kinematics algorithm based on conformal geometric algebra using reconfigurable hardware. In: GRAPP conference Madeira (2008)
31. Hildenbrand, D., Pitt, J.: The Gaalop home page. HTML document http://www.gaalop.de (2008)
32. J. Lasenby E. Bayro-Corrochano, A.L., Sommer, G.: A new methodology for computing invariants in computer vision. In: Proceedings of ICPR 96 (1996)
33. L. Reyes-Lozano, G.M., Bayro-Corrochano: Registration of 3d points using geometric algebra and tensor voting. Journal of Computer Vision **75**(3), 351–369 (2007)
34. Lasenby, J., Fitzgerald, W.J., Lasenby, A., Doran, C.: New geometric methods for computer vision: An application to structure and motion estimation. International Journal of Computer Vision **3**(26), 191–213 (1998)
35. Mann, S., Dorst, L.: Geometric algebra: a computational framework for geometrical applications (part ii: applications). Computer Graphics and Application **22**(4), 58–67 (2002)
36. Mann, S., Dorst, L., Bouma, T.: The making of GABLE, a geometric algebra learning environment in matlab pp. 491–511 (2001)
37. Naeve, Rockwood, A.: Course 53 geometric algebra. In: Siggraph conference Los Angeles (2001)
38. NVIDIA: The CUDA home page. HTML document http://www.nvidia.com/object/cuda_home.html (2009)
39. Perwass, C.: Applications of geometric algebra in computer vision. Ph.D. thesis, Cambridge Universtiy (2000)
40. Perwass, C.: The CLU home page. HTML document http://www.clucalc.info (2005)
41. Perwass, C.: Geometric Algebra with Applications in Engineering. Springer (2009)
42. Perwass, C., Förstner, W.: Uncertain geometry with circles, spheres and conics. In: R. Klette, R. Kozera, L. Noakes, J. Weickert (eds.) Geometric Properties from Incomplete Data, *Computational Imaging and Vision*, vol. 31, pp. 23–41. Springer-Verlag (2006)
43. Perwass, C., Gebken, C., Sommer, G.: Geometry and kinematics with uncertain data. In: A. Leonardis, H. Bischof, A. Pinz (eds.) 9th European Conference on Computer Vision, ECCV 2006, May 2006, Graz, Austria, no. 3951 in LNCS, pp. 225–237. Springer-Verlag, Berlin Heidelberg (2006)
44. Perwass, C., Lasenby, J.: A Geometric Analysis of the Trifocal Tensor. In: R.K. R. Klette G. Gimel'farb (ed.) Image and Vision Computing New Zealand, IVCNZ'98, Proceedings, pp. 157–162. The University of Auckland (1998)
45. Perwass, C., Lasenby, J.: A Unified Description of Multiple View Geometry. In: G. Sommer (ed.) Geometric Computing with Clifford Algebra. Springer-Verlag (2001)
46. Perwass, C., Sommer, G.: The inversion camera model. In: 28. Symposium für Mustererkennung, DAGM 2006, Berlin, 12.-14.09.2006. Springer-Verlag, Berlin, Heidelberg (2006)
47. Petsche, H.J.: The Grassmann Bicentennial Conference home page. HTML document http://www.uni-potsdam.de/u/philosophie/grassmann/Papers.htm (2009)
48. Pham, M., Tachibana, K., Hitzer, E., Yoshikawa, T., Furuhashi, T.: Classification and clustering of spatial patterns with geometric algebra. In: AGACSE conference Leipzig (2008)
49. Rosenhahn, B.: Pose estimation revisited. Ph.D. thesis, Inst. f. Informatik u. Prakt. Mathematik der Christian-Albrechts-Universität zu Kiel (2003)
50. Rosenhahn, B., Sommer, G.: Pose estimation in conformal geometric algebra. Journal of Mathematical Imaging and Vision **22**, 27–70 (2005)
51. Sommer, G. (ed.): Geometric Computing with Clifford Algebra. Springer Verlag (2001)
52. Sommer, G.: Applications of geometric algebra in robot vision. In: H. Li, P. Olver, G. Sommer (eds.) Computer Algebra and Geometric Algebra with Applications, *LNCS*, vol. 3519, pp. 258–277. 6th International Workshop IWMM 2004, Shanghai, China and International Workshop GIAE 2004, Xian, China, Springer-Verlag, Berlin Heidelberg (2005)

53. Sommer, G., Rosenhahn, B., Perwass, C.: The twist representation of free-form objects. In: R. Klette, R. Kozera, L. Noakes, J. Weickert (eds.) Geometric Properties from Incomplete Data, *Computational Imaging and Vision*, vol. 31, pp. 3–22. Springer-Verlag (2006)
54. Wareham, R., Cameron, J., Lasenby, J.: Applications of conformal geometric algebra in computer vision and graphics. Lecture Notes in Computer Science **3519**, 329–349 (2005)
55. Wareham, R., Lasenby, J.: Applications of conformal geometric algebra in computer vision and graphics. submitted to ACM Transactions on Graphics (2004)
56. Zaharia, M., Dorst, L.: Modeling and visualization of 3d polygonal mesh surfaces using geometric algebra. Computers & Graphics **29**(5), 802–810 (2003)