The Electronic Journal of Mathematics and Technology, Volume 9, Number 3, ISSN 1933-2823

Geometric Algebra: A Foundation of Elementary Geometry with possible Applications in Computer Algebra based Dynamic Geometry Systems

Dietmar Hildenbrand
hildenbrand@mathematik.tu-darmstadt.de
University of Technology Darmstadt

Reinhard Oldenburg
reinhard.oldenburg@math.uni-augsburg.de
University of Augsburg

Germany

Abstract

Geometric Algebra is a very general mathematical system providing simultaneously a geometrification of algebra, and also an algebrification of geometry. As an example, we present a specific Geometric Algebra, that we call Compass Ruler Algebra, which is very well suited to compute similar to working with compass and ruler. Geometric objects such as circles and lines as well as geometric operations with them can be handled very easily inside of the algebra. Gaalop is an easy to handle tool in order to compute and visualize with Compass Ruler Algebra. While a computer algebra system is responsible for the symbolic computations, its visualizing component offers basic DGS (Dynamic Geometry System) functionality.

1 Introduction

What mathematicians often call *Clifford algebra* may also be called *Geometric Algebra* if the focus is on the geometric meaning of the algebraic expressions and operators. Geometric Algebra is a mathematical framework that makes it easy to 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 [**?**] and his vision of a general mathematical language for geometry. William Clifford [1] combined Grassmann's exterior algebra and Hamilton's quaternions [1]. Pioneering work was done by David Hestenes, who applied Geometric Algebra to problems in engineering and physics [3, 4]. Please refer to Sect. 2 for an introduction to Geometric Algebra.

In this paper, we present a specific Geometric Algebra, that we call Compass Ruler Algebra, which is very well suited to compute similar to working with compass and ruler (see Sect. 3). Geometric objects such as circles and lines as well as geometric operations with them can be handled very easily inside of the algebra. A circle, for instance, can be described based on the outer product of three points of the circle. The Compass Ruler Algebra is a 4D algebra describing the 2D plane with two additional basis vectors representing the origin and infinity. You are able to directly compute with infinity, for instance, when expressing the center point of a circle as the inversion of infinity in the circle. Taken together this shows how the objects of Compass Ruler Algebra can be described in an algebraic yet syntactic fashion so that a close link between algebra and geometry is established.

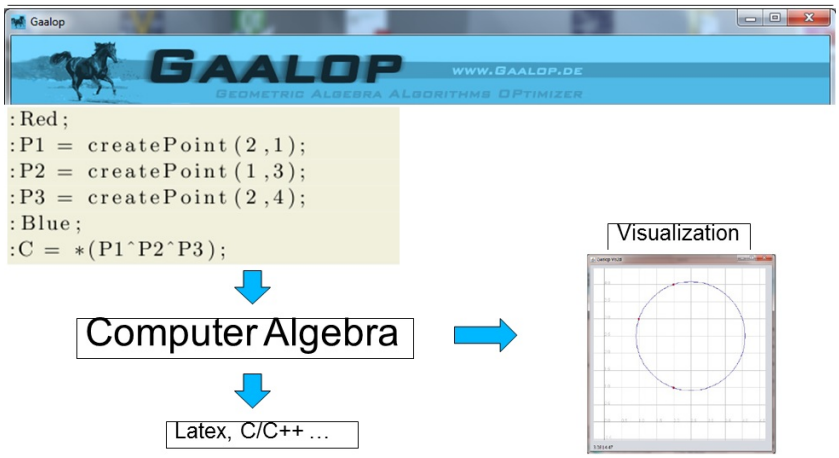

Figure 1: Gaalop symbolically computes Geometric Algebra expressions and generates geometry (visualizations) or simplified formulae (in Latex, C/C++ ... format).

We developed *Gaalop* for the computation and visualization of Geometric Algebra (please refer to the book [6] for details). It uses a computer algebra system as its main computing engine. Gaalop symbolically computes Geometric Algebra expressions and generates geometry or simplified formulae. According to Fig. 1, we use Gaalop for visualizations and for the generation of formulae expressed in Latex or C++ format. Sect. 4 presents some basic DGS functionality based on Gaalop. We present especially how geometric constructions and derivations of formulae based on Compass Ruler Algebra can be handled based on Gaalop.

2 Geometric Algebra

This section presents mainly the mathematical basics of Geometric Algebra. Please refer to Chapt. 3 of the book [8] for an axiomatic approach to this algebra.

Table 1: List of the eight blades of 3D Euclidean Geometric Algebra

Blade	Grade
1	0
e_1	1
e_2	1
e_3	1
$e_1 \wedge e_2$	2
$e_1 \wedge e_3$	2
$e_2 \wedge e_3$	2
$e_1 \wedge e_2 \wedge e_3$	3

2.1 The Basic Algebraic Elements of Geometric Algebra

While the pairwise orthogonal and normalised basis vectors $e_1, e_2, \ldots, e_n$ are the basic algebraic elements of an n-dimensional vector algebra, they are only one part of the algebraic elements of an n-dimensional Geometric Algebra. **Blades** are the basic algebraic elements of Geometric Algebra. An n-dimensional Geometric Algebra consists of blades with **grades** 0, 1, 2, …, n, where a scalar is a **0-blade** (a blade of grade 0) and the **1-blades** are the basis vectors $e_1, e_2, \ldots, e_n$. The **2-blades** $e_i \wedge e_j$ are blades spanned by two 1-blades, and so on (note that $\wedge$ is the outer product as described in section 2.2). There exists only one element of the maximum grade n, $I = e_1 \wedge e_2 \ldots \wedge e_n$. It is therefore also called the **pseudoscalar**. A linear combination of k-blades is called a k-vector (or a vector, bivector, trivector. …). The sum $e_2 \wedge e_3 + e_1 \wedge e_2$, for instance, is a bivector.

A linear combination of blades with different grades is called a **multivector**. Multivectors are the general elements of a Geometric Algebra.

There are 2^n blades in an n-dimensional Geometric Algebra. Table 1, for instance, shows the $8 = 2^3$ blades of 3D Euclidean Geometric Algebra consisting of the scalar, three (basis) vectors, three bivectors, and the pseudoscalar.

2.2 The Products of Geometric Algebra

The main product of Geometric Algebra is called the **geometric product**; many other products can be derived from it, especially the **outer** and the **inner** product. The notations of these products are listed in Table 2. We will use the outer product mainly for the construction and intersection of geometric objects, while the inner product will be used for the computation of angles and distances.

Table 2: Notations for the Geometric Algebra products

Notation	Meaning
AB	Geometric product of A and B
$A \wedge B$	Outer product of A and B
$A \cdot B$	Inner product of A and B

2.2.1 The Outer Product

Geometric Algebra provides an outer product $\wedge$ with the properties listed in Table 3.

Table 3: Properties of the outer product $\wedge$ of vectors

Property	Meaning
Anticommutativity	$u \wedge v = -(v \wedge u)$
Distributivity	$u \wedge (v + w) = u \wedge v + u \wedge w$
Associativity	$u \wedge (v \wedge w) = (u \wedge v) \wedge w$

The outer product of two parallel vectors is 0:

$$u \wedge u = -(u \wedge u) = 0. \tag{1}$$

This is the reason why the outer product can be used as a measure of parallelness.
Computation example 1 We compute the outer product of two vectors:

$$c = (e_1 + e_2) \wedge (e_1 - e_2)$$

can be transformed based on distributivity to

$$c = (e_1 \wedge e_1) - (e_1 \wedge e_2) + (e_2 \wedge e_1) - (e_2 \wedge e_2);$$

since $u \wedge u = 0$,

$$c = -(e_1 \wedge e_2) + (e_2 \wedge e_1),$$

and because of anticommutativity,

$$c = -(e_1 \wedge e_2) - (e_1 \wedge e_2)$$

or

$$c = -2(e_1 \wedge e_2).$$

2.2.2 The Inner Product

For Euclidean spaces, the inner product of two vectors is the same as the well-known Euclidean scalar product of two vectors. For perpendicular vectors, the inner product is 0; for instance, $e_1 \cdot e_2 = 0$.

The inner product of Geometric Algebra contains metric information. We use Compass Ruler Algebra and its inner product for the computation of angles and distances (see Sect. 3.2).

2.2.3 The Geometric Product

The geometric product is an amazingly powerful operation, which is used mainly for the handling of transformations (see Sect. 3.3). The geometric product of vectors is a combination of the outer

product and the inner product. The geometric product of u and v is denoted by uv. For vectors u and v, the geometric product uv can be defined as

$$uv = u \wedge v + u \cdot v. \tag{2}$$

We derive the following for the inner and outer products:

$$u \cdot v = \frac{1}{2}(uv + vu), \tag{3}$$

$$u \wedge v = \frac{1}{2}(uv - vu). \tag{4}$$

but, as noted above, these formulas apply only for vectors, in this form.
Computation examples: What is the square of a vector?

$$u^2 = uu = \underbrace{u \wedge u}_{0} + u \cdot u = u \cdot u$$

for example

$$e_1^2 = e_1 \cdot e_1 = 1.$$

The geometric product is defined for all kinds of multivectors. Let us calculate, for example, the following geometric product of two bivectors :

$$u^2 = (e_1 \wedge e_2)^2$$

Since $e_1e_2 = e_1 \wedge e_2 + \underbrace{e_1 \cdot e_2}_{0} = e_1 \wedge e_2$,

$$u^2 = (e_1 \wedge e_2)^2 = e_1e_2 \underbrace{e_1e_2}_{-e_2e_1} = -e_1 \underbrace{e_2e_2}_{1} e_1 = -\underbrace{e_1e_1}_{1} = -1$$

Because of this property, u can be used such as the imaginary unit i of complex numbers. In the case of $u = e_1 \wedge e_2$ and $v = (e_1 + e_2) \wedge e_3$,

$$\begin{aligned} uv &= (e_1 \wedge e_2)((e_1 + e_2) \wedge e_3), \\ uv &= (e_1e_2)(e_1 \wedge e_3 + e_2 \wedge e_3) \\ &= e_1e_2(e_1e_3 + e_2e_3) \\ &= e_1e_2e_1e_3 + e_1 \underbrace{e_2e_2}_{1} e_3; \end{aligned}$$

since $e_1e_2 = e_1 \wedge e_2 = -e_2 \wedge e_1 = -e_2e_1$,

$$\begin{aligned} uv &= -e_2e_1e_1e_3 + e_1e_3 \\ &= -e_2e_3 + e_1e_3 \\ &= -(e_2 \wedge e_3) + e_1 \wedge e_3. \end{aligned}$$

Table 4: The 16 basis blades of the Compass Ruler Algebra.

Index	Blade	Dimension
0	1	0
1	e_1	1
2	e_2	1
3	e_∞	1
4	e_0	1
5	$e_1 \wedge e_2$	2
6	$e_1 \wedge e_\infty$	2
7	$e_1 \wedge e_0$	2
8	$e_2 \wedge e_\infty$	2
9	$e_2 \wedge e_0$	2
10	$e_\infty \wedge e_0$	2
11	$e_1 \wedge e_2 \wedge e_\infty$	3
12	$e_1 \wedge e_2 \wedge e_0$	3
13	$e_1 \wedge e_\infty \wedge e_0$	3
14	$e_3 \wedge e_\infty \wedge e_0$	3
15	$e_1 \wedge e_2 \wedge e_\infty \wedge e_0$	4

3 Combination of Geometry and Algebra with Compass Ruler Algebra

This section introduces the Compass Ruler Algebra as a Geometric Algebra with which you are able to compute in a way similar to working with compass and ruler. We describe the algebraic representations of geometric objects as well as the geometric meaning of the main products and will see that there is a strong relation between geometry and algebra.

In the Compass Ruler Algebra, there is a correspondence between algebraic expressions and geometric objects such as circles and lines. Circles, for instance, can be defined based on the outer product of three points. This is why the circumcircle of a triangle can be expressed easily. Furthermore, with the help of the products of the algebra, distances and angles as well as geometric operations such as intersections of geometric objects can be described. The algebra uses the two Euclidean basis vectors e_1 and e_2 of the plane and two additional basis vectors e_+, e_- with positive and negative signatures, respectively, which means that they square to +1 as usual (e_+) and to -1 (e_-).

$$e_+^2 = 1, \qquad e_-^2 = -1, \qquad e_+ \cdot e_- = 0. \tag{5}$$

Another basis e_0, e_∞, with the geometric meaning (see Sect. 3.1.1)

- e_0 represents the 3D origin,
- e_∞ represents infinity,

Table 5: The representations of the geometric entities of the Compass Ruler Algebra

Entity	**IPNS representation** (inner product null space)	**OPNS representation** (outer product null space)
Point	$P = \mathbf{x} + \frac{1}{2}\mathbf{x}^2 e_\infty + e_0$	
Circle	$C = P - \frac{1}{2}r^2 e_\infty$	$C^* = P_1 \wedge P_2 \wedge P_3$
Line	$L = \mathbf{n} + d e_\infty$	$L^* = P_1 \wedge P_2 \wedge e_\infty$
Point pair	$P_p = C_1 \wedge C_2$	$P_p^* = P_1 \wedge P_2$

can be defined with the relations

$$e_0 = \frac{1}{2}(e_- - e_+), \qquad e_\infty = e_- + e_+. \tag{6}$$

These new basis vectors are null vectors:

$$e_0^2 = e_\infty^2 = 0. \tag{7}$$

Taking their inner product results in

$$e_\infty \cdot e_0 = -1, \tag{8}$$

since

$$(e_- + e_+) \cdot \frac{1}{2}(e_- - e_+) = \frac{1}{2}(\underbrace{e_- \cdot e_-}_{-1} - \underbrace{e_- \cdot e_+}_{0} + \underbrace{e_+ \cdot e_-}_{0} - \underbrace{e_+ \cdot e_+}_{1}),$$

and their geometric product is

$$e_\infty e_0 = e_\infty \wedge e_0 + e_\infty \cdot e_0 = e_\infty \wedge e_0 - 1 \tag{9}$$

or

$$e_0 e_\infty = e_0 \wedge e_\infty + e_\infty \cdot e_0 = -e_\infty \wedge e_0 - 1. \tag{10}$$

Table 4 lists all the basis blades (outer products of subsets of the 4 basis vectors). The algebraic basis elements of the algebra are the multivectors as a linear combination of these basis blades. Note that, for simplicity and practical reasons, we use e_0 and e_∞ of equation (6) for this blade description while mathematically correct the basis vectors e_+ and e_- should be used.

3.1 The Basic Geometric Entities

In the Compass Ruler Algebra, geometric objects can be represented as algebraic expressions. Multivectors representing the basic geometric entities of the algebra, namely points, circles, lines, and point pairs, are listed in Table 5.

These entities have two algebraic representations: the IPNS (inner product null space) and the OPNS (outer product null space). The IPNS of the algebraic expression A are all the points X satisfying the equation

$$A \cdot X = 0. \tag{11}$$

The OPNS of the algebraic expression A are all the points X satisfying the equation

$$A \wedge X = 0. \tag{12}$$

These representations are duals of each other (a superscript asterisk denotes the dualization operator). Below you will find some examples how to compute with these null spaces.

In the following, we present the representations of all of the basic geometric entities.

3.1.1 Points and the Meaning of e_0 and e_∞

In order to represent points in Compass Ruler Algebra, the original 2D point $\mathbf{x} = x_1 e_1 + x_2 e_2$ is extended to a 4D vector by taking a linear combination of the 4D basis vectors e_1, e_2, e_∞, and e_0 according to the equation

$$P = \mathbf{x} + \frac{1}{2}\mathbf{x}^2 e_\infty + e_0. \tag{13}$$

For example, for the 2D origin $(x_1, x_2) = (0, 0)$ we get

$$P = e_0. \tag{14}$$

In order to evaluate the geometric meaning of e_0, we are able to compute its IPNS as its null space with respect to the inner product. The IPNS of e_0 describes all the points X satisfying the equation

$$e_0 \cdot X = 0. \tag{15}$$

The following Gaalop script

Listing 1: Computation of the IPNS of e_0.

```
X = createPoint(x,y);
?Result = e0.X;
```

computes this inner product and assigns it to the variable Result (Gaalop computes all the variables indicated by a leading semicolon). This resulting multivector is equal to the scalar value $x^2 + y^2$ with the null space

$$x^2 + y^2 = 0, \tag{16}$$

describing exactly the point at the origin.

In order to evaluate the geometric meaning of e_∞, we assume an arbitrary Euclidean point $\mathbf{x} = x_1 e_1 + x_2 e_2$ (not equal to the origin) with a normalized Euclidean vector $\mathbf{n}$ in the direction of $\mathbf{x}$,

$$\mathbf{x} = t\mathbf{n},\ t > 0,\ \mathbf{n}^2 = 1 \tag{17}$$

with its representation P according to equation (13) and consider its limit $\lim_{t\to\infty}$. Another (homogeneous) representation of this point P is cP, its product with an arbitrary scalar value $c \neq 0$. In terms of the IPNS, this can be seen as follows: The IPNS equation $P \cdot X = 0$ is equivalent to the equation $cP \cdot X = c(P \cdot X) = 0$, means P and cP are representing the same geometric object.

Let us choose the arbitrary scalar value as $c = \frac{2}{\mathbf{x}^2}$ and consider $P' = \frac{2}{\mathbf{x}^2}P$,

$$P' = \frac{2}{\mathbf{x}^2}(\mathbf{x} + \frac{1}{2}\mathbf{x}^2 e_\infty + e_0), \tag{18}$$

$$P' = \frac{2}{\mathbf{x}^2}\mathbf{x} + e_\infty + \frac{2}{\mathbf{x}^2}e_0. \tag{19}$$

We use this form to compute the limit $\lim_{t\to\infty} P'$ for increasing $\mathbf{x}$. Since $\mathbf{x} = t\mathbf{n}$, we get

$$P' = \frac{2}{t^2\mathbf{n}^2}t\mathbf{n} + e_\infty + \frac{2}{t^2\mathbf{n}^2}e_0 \tag{20}$$

and, since $\mathbf{n}^2 = 1$,

$$P' = \frac{2}{t}\mathbf{n} + e_\infty + \frac{2}{t^2}e_0. \tag{21}$$

Based on this formula and the fact that P and P' represent the same Euclidean point, we can easily see that the point at infinity for any direction vector $\mathbf{n}$ is represented by e_∞:

$$\lim_{t\to\infty} P' = e_\infty. \tag{22}$$

3.1.2 Circles

A circle can, on the one hand, be represented with the help of its center point P and its radius r as

$$C = P - \frac{1}{2}r^2 e_\infty \tag{23}$$

Note that the representation of a point is simply that of a circle of radius zero.

A circle can also, on the other hand, be represented with the help of three points that lie on it, by

$$C^* = P_1 \wedge P_2 \wedge P_3. \tag{24}$$

As an example, we compute the IPNS of the expression $e_0 - \frac{1}{2}r^2 e_\infty$, means all the points X satisfying the following equation

$$\left(e_0 - \frac{1}{2}r^2 e_\infty\right) \cdot X = 0. \tag{25}$$

The following Gaalop script

Listing 2: Computation of the IPNS of an origin circle.

```
X = createPoint(x,y);
C = e0- 0.5*r*r*einf;
?Result = C.X;
```

computes this inner product. The resulting multivector is equal to the scalar value $\frac{1}{2}(x^2 + y^2 - r^2)$ with the null space

$$x^2 + y^2 - r^2 = 0, \tag{26}$$

describing all points at the same distance r from the origin, namely a circle at the origin.

3.1.3 Lines

A line is defined by

$$L = \mathbf{n} + de_\infty, \tag{27}$$

where $\mathbf{n} = n_1 e_1 + n_2 e_2$ refers to the 2D normal vector of the line L and d is the distance to the origin. A line can also be defined with the help of two points that lie on it and the point at infinity:

$$L^* = P_1 \wedge P_2 \wedge e_\infty. \tag{28}$$

Note that a line is a circle of infinite radius.

3.1.4 Point Pairs

A point pair is defined by the intersection of two circles

$$Pp = C_1 \wedge C_2 \tag{29}$$

or, directly with the help of two points, by

$$Pp^* = P_1 \wedge P_2. \tag{30}$$

3.2 Distances and Angles

We will now investigate the inner product of lines, circles and points as well as its geometric meaning as summarized in Table 6.

Table 6: Geometric meaning of the inner product of lines, circles and points

$\cdot$	**Line**	**Circle**	**Point**
Line	Angle between lines	Euclidean distance from center	Euclidean distance
Circle	Euclidean distance from center	Distance measure	Distance measure
Point	Euclidean distance	Distance measure	Distance

In Compass Ruler Algebra, points, lines and circles are represented as vectors. The inner product of this kind of objects is a scalar and can be used as a measure of distance. In the following examples, we will see that the inner product $P \cdot Q$ of two vectors P and Q can be used for tasks such as

- the distance between two points;
- the distance between a point and a line;
- the decision as to whether a point is inside or outside a circle.

3.2.1 Distance between Points

The following Gaalop script computes the inner product of two points P and Q

Listing 3: Computation of the inner product of two points.

```
P = createPoint(p1,p2);
Q = createPoint(q1,q2);
?Result = P.Q;
```

and results in

$$P \cdot Q = -\frac{1}{2}(q_2^2 - 2p_2q_2 + q_1^2 + p_2^2 - 2p_1q_1 + p_1^2)$$

$$= -\frac{1}{2}((q_1 - p_1)^2 + (q_2 - p_2)^2)$$

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

We recognize that the square of the Euclidean distance of the 2D points corresponds to the inner product of the 4D representation of the points multiplied by -2.

$$(\mathbf{q} - \mathbf{p})^2 = -2(P \cdot Q)$$

3.2.2 Distance between a Point and a Line

The following Gaalop script computes the inner product of the point P and the line L

Listing 4: Computation of the inner product of a point and a line.

```
P = createPoint(p1,p2);
L = n1*e1+n2*e2+d*einf;
?Result = P.L;
```

and results in

$$P \cdot L = p_1n_1 + p_2n_2 - d$$

$$= \mathbf{p} \cdot \mathbf{n} - d,$$

which represents the Euclidean distance between the point and the line, with a sign according to

- $P \cdot L > 0$: $\mathbf{p}$ is in the direction of the normal $\mathbf{n}$;
- $P \cdot L = 0$: $\mathbf{p}$ is on the line;
- $P \cdot L < 0$: $\mathbf{p}$ is not in the direction of the normal $\mathbf{n}$.

3.2.3 Distance between a Line and a Circle

The following Gaalop script computes the inner product of the line L and the circle C

Listing 5: Computation of the inner product of a line and a circle.

```
P = createPoint(p1,p2);
C = P - 0.5*r*r*einf;
L = n1*e1+n2*e2+d*einf;
?Result = L.C;
```

and results in

$$L \cdot C = n_1 p_1 + n_2 p_2 - d$$

$$= \mathbf{n} \cdot \mathbf{p} - d, \tag{31}$$

which represents the Euclidean distance between the center point of the circle and the line (see Sect. 3.2.2).

3.2.4 Distance between Two Circles

The following Gaalop script computes the inner product of two circles C_1 and C_2

Listing 6: Computation of the inner product of two circles.

```
P = createPoint(p1,p2);
C1 = P - 0.5*r1*r1*einf;
Q = createPoint(q1,q2);
C2 = Q - 0.5*r2*r2*einf;
?Result = C1.C2;
```

and results in

$$C_1 \cdot C_2 = \frac{1}{2}(r_2^2 + r_1^2 - q_2^2 + 2 * p_2 * q_2 - q_1^2 + 2 * p_1 * q_1 - p_2^2 - p_1^2)$$

$$= \frac{1}{2}r_1^2 + \frac{1}{2}r_2^2 - \frac{1}{2}(\mathbf{q}^2 - 2\mathbf{p} \cdot \mathbf{q} + \mathbf{p}^2)$$

$$= \frac{1}{2}(r_1^2 + r_2^2) - \frac{1}{2}(\mathbf{q} - \mathbf{p})^2.$$

We get

$$2(C_1 \cdot C_2) = r_1^2 + r_2^2 - (\mathbf{q} - \mathbf{p})^2. \tag{32}$$

This means that twice the inner product of two circles is equal to the sum of the squares of their radii minus the square of the Euclidean distance between the centers of the circles.

3.2.5 The Inner Product of a Point and a Circle

The following Gaalop script computes the inner product of a point P and a circle C

Listing 7: Computation of the inner product of a point and a circle.

```
P = createPoint(p1,p2);
Q = createPoint(q1,q2);
C=Q-0.5*r*r*einf;
Result = P.C;
```

and results in

$$P \cdot C = \frac{1}{2}r^2 - \frac{1}{2}(\mathbf{q} - \mathbf{p})^2.$$

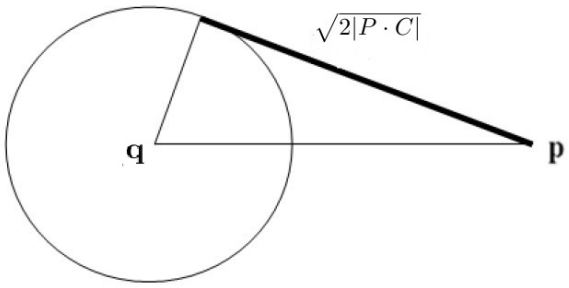

Figure 2: The inner product of a point and a circle describes the distance between the point and the tangent point of the circle according to (34).

We get

$$2(P \cdot C) = r^2 - (\mathbf{q} - \mathbf{p})^2 \tag{33}$$

or

$$-2(P \cdot C) = (\mathbf{q} - \mathbf{p})^2 - r^2. \tag{34}$$

Fig. 2 illustrates this formula. The triangle shown is right-angled. According to Pythagoras' theorem, $\sqrt{2|P \cdot C|}$ is equal to the distance between $\mathbf{p}$ and the tangent point to the circle.
Note that the inner product of a point and a circle can be used to decide whether a point is inside a circle or not:

- if $P \cdot C > 0$, $\mathbf{p}$ is inside the circle;
- if $P \cdot C = 0$, $\mathbf{p}$ is on the circle;
- if $P \cdot C < 0$, $\mathbf{p}$ is outside the circle.

3.2.6 Angles between Lines

Let us now derive an expression for the angle between two lines. The inner product of the line $L_1 = \mathbf{n_1} + d_1 e_\infty$ with normal vector $\mathbf{n_1}$ and distance d_1 and another line $L_2 = \mathbf{n_2} + d_2 e_\infty$ can be computed with the help of the following Gaalop script

Listing 8: Computation of the inner product of two lines.

```
L1 = n11*e1+n12*e2+d1*einf;
L2 = n21*e1+n22*e2+d2*einf;
?Result = L1.L2;
```

resulting in

$$L_1 \cdot L_2 = \mathbf{n_1} \cdot \mathbf{n_2} \tag{35}$$

representing the scalar product of the two normals of the lines. Based on this observation, the angle θ between the two lines can be computed as

$$\cos(\theta) = L_1 \cdot L_2. \tag{36}$$

3.3 Transformations

Transformations can easily be described in Compass Ruler Algebra. All kinds of transformations of an object o can be done with the help of the following geometric product:

$$o_{transformed} = V o \tilde{V}, \tag{37}$$

where V is a **versor** and $\tilde{V}$ is its reverse (In the reverse of a multivector the blades are with reversed order of their outer product components, for instance the reverse of $1 + e_1 \wedge e_2$ is equal to $1 + e_2 \wedge e_1$ or $1 - e_1 \wedge e_2$).

We focus here on rotations and translations.

3.3.1 Rotation

The operator

$$R = e^{-(\frac{\phi}{2})L} \tag{38}$$

describes a **rotor**. L is the normalized bivector $e_1 \wedge e_2$, and ϕ is the rotation angle. R can also be written as

$$R = \cos\left(\frac{\phi}{2}\right) - L \sin\left(\frac{\phi}{2}\right). \tag{39}$$

The rotation of a geometric object o is performed with the help of the operation

$$o_{rotated} = R o \tilde{R}.$$

There are strong relations between rotations in Compass Ruler Algebra and complex numbers.

3.3.2 Translation

In Compass Ruler Algebra, a translation can be expressed in a multiplicative way with the help of a **translator** T defined by

$$T = e^{-\frac{1}{2}\mathbf{t}e_\infty}, \tag{40}$$

where $\mathbf{t}$ is a vector

$$\mathbf{t} = t_1 e_1 + t_2 e_2.$$

Application of the Taylor series

$$T = e^{-\frac{1}{2}\mathbf{t}e_\infty} = 1 + \frac{-\frac{1}{2}\mathbf{t}e_\infty}{1!} + \frac{(-\frac{1}{2}\mathbf{t}e_\infty)^2}{2!} + \frac{(-\frac{1}{2}\mathbf{t}e_\infty)^3}{3!} + \dots$$

and the property $(e_\infty)^2 = 0$ results in the translator

$$T = 1 - \frac{1}{2}\mathbf{t}e_\infty. \tag{41}$$

3.3.3 Rigid-Body Motion

In Compass Ruler Algebra, a rigid-body motion, including both a rotation and a translation, is described by a **displacement versor** D, sometimes also called a **motor**,

$$D = RT, \tag{42}$$

where R is a rotor and T is a translator. A rigid-body motion of an object o is described by

$$o_{rigid\_body\_motion} = Do\tilde{D}.$$

4 Basic DGS functionality based on Gaalop

Compass Ruler Algebra provides a strong relation between geometry and algebra (see Sect. 3). Geometric objects, operations and transformations can be expressed very easily based on algebraic expressions. In this section we present some basic DGS functionality based on Gaalop and this algebra. Gaalop can be downloaded from [6] free of charge.

4.1 Geometric Constructions based on Compass Ruler Algebra

We use the Gaalop software package to visualize expressions of Compass Ruler Algebra. With the help of the Vis2d component of Gaalop, the script of listing 9 is transformed into the visualization of Fig. 3.

Listing 9: Script for the visualization of a circle based on the outer product of three points.

```
:Red;
:P1 = createPoint(2,1);
:P2 = createPoint(1,3);
:P3 = createPoint(2,4);
:Blue;
:C = *(P1^P2^P3);
```

First of all, three points with the 2D coordinates $(2,1),(1,3)$ and $(2,4)$ are transformed into 4D coordinates of the Compass Ruler Algebra and visualized in red (all variables with a leading colon are visualized with the currently set color). Then the circle C is computed based on the outer product of these three points, transformed into the IPNS representation via the dualization operator and visualized in blue. Note: this way the circumcircle of a triangle can be computed very easy.

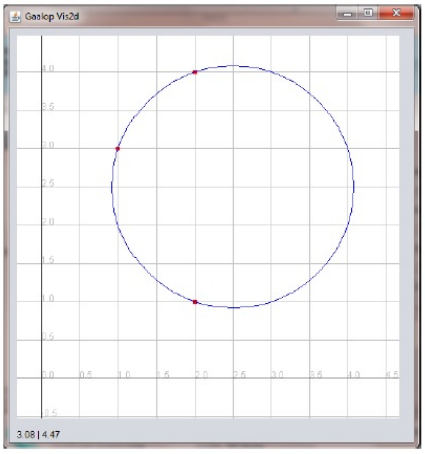

Figure 3: Visualization of a circle based on the outer product of three (red) points with the help of Gaalop Vis2D.

The following example shows how, in Compass Ruler Algebra, we are able to compute comparable to working with compass and ruler. In order to construct the perpendicular bisector of a section of a line with compass and ruler, we draw two circles with the center at the boundary points and connect the two intersection points according to Fig. 4.

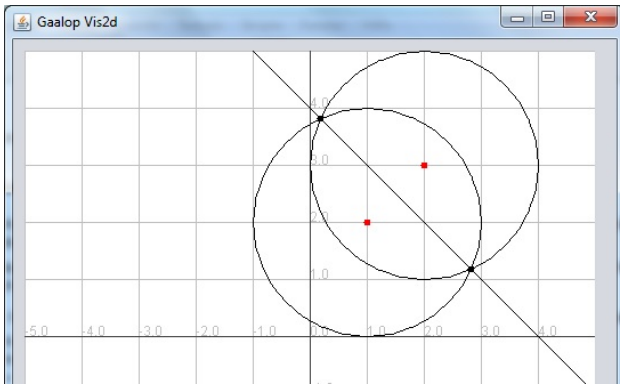

Figure 4: Visualization of the perpendicular bisector between the two red points.

How can we express this construction based on Compass Ruler Algebra?
The following Gaalop script

Listing 10: Computation of the perpendicular bisector of the section of line between the points P1 and P2 with the help of the intersection of two circles.

```
P1 = createPoint(x1,y1);
```

```
P2 = createPoint(x2,y2);
// intersect two circles with center points P1 and P2 with the same,
// but arbitrary radius
S1 = P1 - 0.5*r*r*einf;
S2 = P2 - 0.5*r*r*einf;
PP_dual = *(S1^S2);
// the line through the two points of the resulting point pair
Bisector = *(PP_dual^einf);
```

computes, first of all, two points P1 and P2 with the symbolic 2D coordinates (x1,y1) and (x2,y2). Then, the circles S1 and S2 with center at the points P1 and P2 with radius r are computed. The intersection of the circles results in a point pair (see Table 5). Its dual PP_dual describes the outer product of the two intersection points. Finally, the resulting bisector line is the dual of the outer product of these two points and e_∞ (see Table 5).

For the visualization of Fig. 4 the variables have to be equipped first with concrete input values (see listing 11)

Listing 11: concrete imput values for Fig. 4 .

```
x1 = 1;
y1 = 2;
x2 = 2;
y2 = 3;
r = 2;
```

as well as the colors for the geometric objects to be drawn have to be defined at the end (see listing 12).

Listing 12: Visualisization statements for Fig. 4 .

```
:Red;
:P1;
:P2;
:Black;
:S1;
:S2;
:Bisector;
```

The points are visualized in red and the circles and the bisector line in black.

4.2 Deriving of Formulae

Gaalop [6] can also be used to derive formulae of geometric relationships.

In case we use variable names in section 4.1 instead of concrete values for the point coordinates, Gaalop computes symbolically and the resulting formulae are presented in output formats such as C code or Latex code.

Listing 10 makes exactly that. Around the two points with the symbolic 2D coordinates (x1,y1) and (x2,y2) two circles are drawn with radius r and the perpendicular bisector is computed based on the line through the two intersecting points of the circles.

The result of Listing 10 is the multivector Bisector expressed as C code according to Listing 13.

What we can see is, that we can express the perpendicular bisector easily as the difference of the two points.

Listing 13: Result of the Gaalop script of listing 10 .

```
void calculate(float x1, float x2, float y1, float y2,
                        float Bisector[16]) {

        Bisector[1] = x2 - x1; // e1
        Bisector[2] = y2 - y1; // e2
        Bisector[3] = ((y2 * y2) / 2.0 - (y1 * y1) / 2.0
                    + (x2 * x2) / 2.0) - (x1 * x1) / 2.0; // einf
}
```

5 Conclusion and Outlook

The previous sections showed, that Gaalop is able to provide basic DGS functionality based on Compass Ruler Algebra. This algebra is able to handle basic geometric objects and operations while always combining geometry and algebra in a very consistent way.

Nevertheless, there are open items. A core question of Dynamic Geometry Systems, as stated in the PhD thesis of Kortenkamp [7] is: How should a construction behave under movements? The most natural thing would be a continuous movement of dependent elements: When a free element is moved only a little bit, then the dependent elements will move only a little bit, too. We do not want elements to jump around wildly. However, it has been proven by Kortenkamp that a moving strategy can either be continous or deterministic. So the question arises whether a geometric algebra produces continous or deterministic behaviour. However, as e.g. intersections of circles produce point pairs, there is no obvious strategy how to select a point from a point pair - and exactly this decision is the one that decides between continous or deterministic behaviour. One important situation are two moving circles. In Compass Ruler Algebra, their intersections are not two different points but a point pair as only one algebraic expression (see section 3.1.4). The intersection of two circles can be investigated based on the example of computing the perpendicular bisector (see Figure 4). This example raises questions such as

- what happens, if the radius is smaller than half the distance of the two points?
- what about an imaginary radius?
- what happens with the point pair, if the two circles touch each other?
- what happens with the point pair, if the two circles overlap each other?

With Geometric Algebra, there is a good chance of answering these questions in a consistent way.

References

[1] William Kingdon Clifford. Applications of Grassmanns Extensive Algebra, volume 1 of American Journal of Mathematics, pages 350358. The Johns Hopkins University Press, 1878.

[2] Hermann Grassmann. Die Ausdehnungslehre. Vollstaendig und in strenger Form begruendet. Verlag von Th. Chr. Fr. Enslin, Berlin, 1862.

[3] David Hestenes. New Foundations for Classical Mechanics. Springer, 1999.

[4] David Hestenes and Garret Sobczyk. Clifford Algebra to Geometric Calculus: A Unified Language for Mathematics and Physics. Springer, 1987.

[5] Dietmar Hildenbrand. Foundations of Geometric Algebra Computing. Springer, 2013.

[6] Dietmar Hildenbrand, Patrick Charrier, Christian Steinmetz, and Joachim Pitt. Gaalop home page. Available at http://www.gaalop.de, 2014.

[7] Ulrich Kortenkamp. Foundations of Dynamic Geometry. PhD thesis, ETH Zuerich, 1999.

[8] Christian Perwass. Geometric Algebra with Applications in Engineering. Springer, 2009.