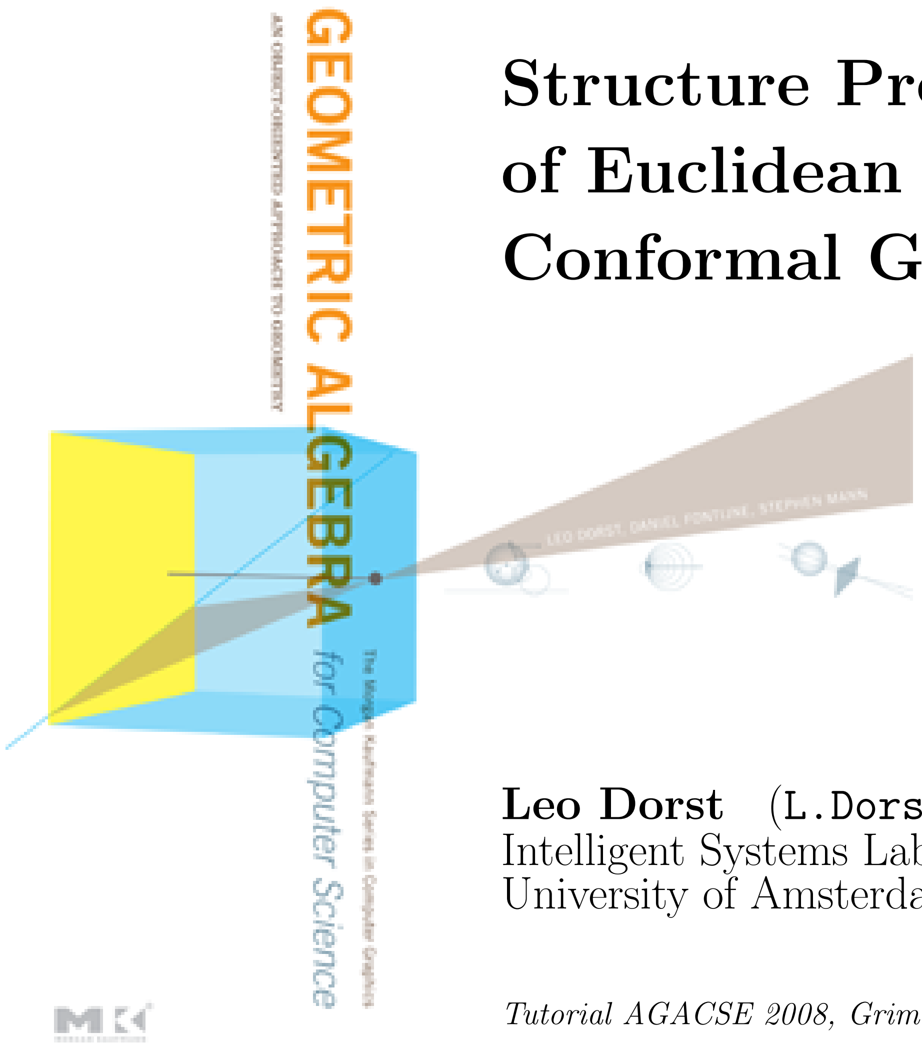
Tutorial:

Structure Preserving Representation of Euclidean Motions through Conformal Geometric Algebra


GEOMETRIC ALGEBRA for Computer Science
AN OBJECT-ORIENTED APPROACH TO GEOMETRY
LEO DORST, DANIEL FONTIJNE, STEPHEN MANN
The Morgan Kaufmann Series in Computer Graphics

Leo Dorst (`L.Dorst@uva.nl`)
Intelligent Systems Laboratory, Informatics Institute,
University of Amsterdam, The Netherlands

Tutorial AGACSE 2008, Grimma near Leipzig, Germany

1 Structural Aspects of a Language for Geometry

- *primitives*: points, lines, planes, circles, spheres, tangents
- *constructions*: connections, intersections, orthogonal complement, duality
- *motions*: translations, rotations, reflection, projection
- *properties*: size, location, orientation, cross ratio
- *numerics*: approximation, estimation, linearization

Motions are at the basis of geometry (Klein), structure preservation:

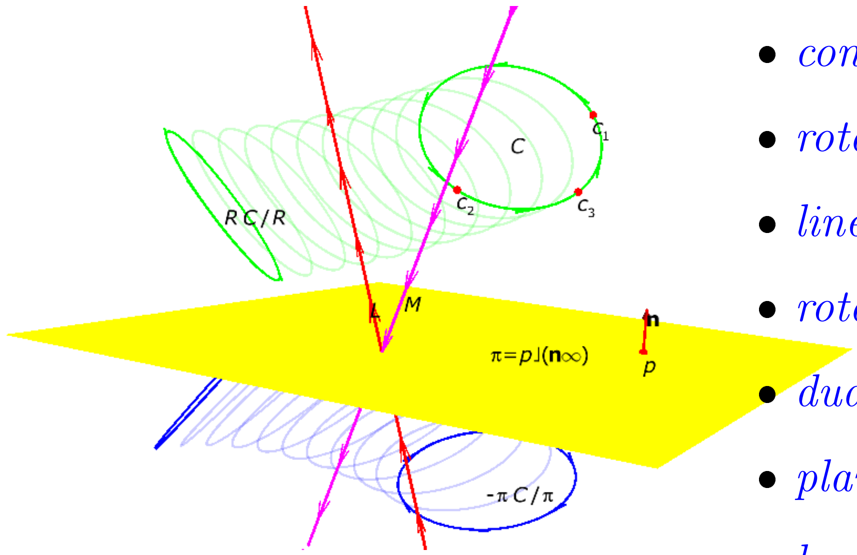
> *Constructions and properties of primitives should be covariant under motions.*

Example: covariant intersection.

$$\begin{tikzcd}
\text{line } \Lambda, \text{ plane } \Pi \arrow[r, "\text{intersect}"] \arrow[d, "\text{motion}"'] & \text{point } \Lambda \cap \Pi \arrow[d, "\text{motion}"] \\
\text{line } \Lambda', \text{ plane } \Pi' \arrow[r, "\text{intersect}"] & \Lambda' \cap \Pi' = (\Lambda \cap \Pi)'
\end{tikzcd}$$

Not automatic in standard linear algebra!
(This example would be $M\ \ker(\mathrm{span}(\Lambda, \Pi)^\top) = \ker(\mathrm{span}(M^{-\top}\Lambda, M^{-\top}\Pi)^\top)$ in hom. coordinates.)

2 An Example: Reflection of a Rotating Circle



FIG(1,1)

- *construction of a circle*: $C = c_1 \wedge c_2 \wedge c_3$
- *rotation*: $C \mapsto RC/R$
- *line representation*: $L = p_1 \wedge p_2 \wedge e_\infty = p_1 \wedge \mathbf{u} \wedge e_\infty$
- *rotation around line*: $R = \exp(\phi\, L^*/2)$
- *dual plane representation*: $\pi = p \cdot (\mathbf{n} \wedge e_\infty)$
- *plane reflection*: $X \mapsto -\pi\, X/\pi$ (for any odd-D X)
- *logarithms of motions*: $R^{1/n} = \exp(\log(R)/n)$

Note that all is specified directly in terms of the geometric elements, and some algebraic operations ($\wedge$, , /, exp, log, *, $\cdot$, fortunately all reducible to one fundamental product).

No coordinates at all in the language (just in the data).

Most figures from *Geometric Algebra for Computer Science* ©Morgan-Kaufmann 2007.
For the demos: type FIG(i,j) in GAViewer, downloadable at `www.geometricalgebra.net`.

3 Implementation Matches the Algebra (here Gaigen2)

```
// p1, p2, c1, c2, c3, pt are points
line L; circle C; dualPlane p; vector n;

L = unit_r(p1 ^ p2 ^ ni);
C = c1 ^ c2 ^ c3;
p = pt << (n^ni);

draw(L); draw(C); draw(p);

draw( - p * L * inverse(p) );                    // draw reflected line (magenta)
draw( - p * C * inverse(p) );                    // draw reflected circle (blue)

// compute rotation versor:
const float phi = (float)(M_PI / 2.0);
TRversor R;
R = exp(0.5f * phi * dual(L));

draw(R * C * inverse(R));                       // draw rotated cicle (green)
draw(-p * R * C * inverse(R) * inverse(p));     // draw reflected, rotated circle (blue)

// draw interpolated circles
pointPair LR = log(R);                          // get log of R
for (float alpha = 0; alpha < 1.0; alpha += 0.1f)
{
  TRversor iR;
  iR = exp(alpha * LR);                         // compute interpolated rotor

  draw(iR * C * inverse(iR));                   // draw rotated circle (light green)
  draw(-p * iR * C * inverse(iR) * inverse(p)); // draw reflected, rotated circle (light blue)
}
```

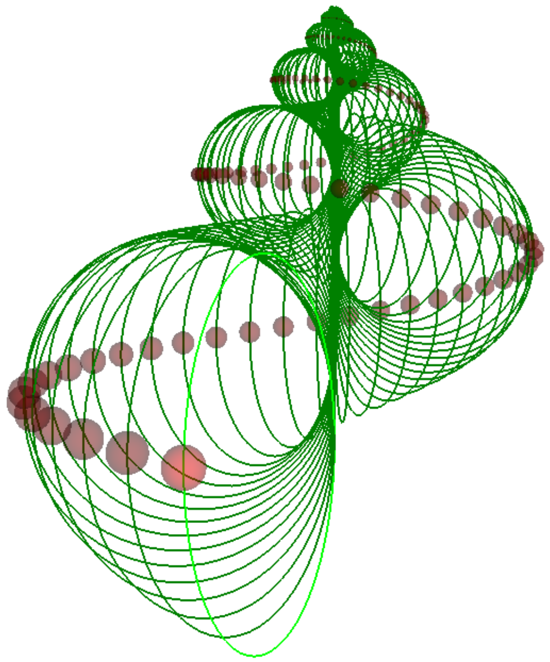
4 By Contrast, the Example in Linear Algebra

- *construction of a circle*: none, treat the points separately.
- *rotation*: by 4×4 *homogeneous coordinate matrix* $\left[\!\!\left[\begin{matrix} \mathsf{R} & (\mathsf{I}-\mathsf{R})\mathbf{t} \\ 0^T & 1 \end{matrix}\right]\!\!\right]$ acting on points $(\mathbf{x}, 1)^T$.
- *line representation*:
  - as (position vector, direction vector)-pair $(\mathbf{p}, \mathbf{u})$; each component moves differently.
  - as the kernel of two homogeneous plane equations: $[\![\ \pi_1, \pi_2\ ]\!]^T$
  - using 6D Plücker coordinates: $\{\mathbf{u}, \mathbf{p} \times \mathbf{u}\}$.
- *rotation around line*: $[\![\mathsf{R}]\!] = \mathbf{u}\mathbf{u}^T + \cos\phi\,([\![1]\!] - \mathbf{u}\mathbf{u}^T) + \sin\phi\,[\![\mathbf{u}^\times]\!]$, then move into place.
- *dual plane representation*: $\pi = [\mathbf{n}, -\mathbf{p}\cdot\mathbf{n}]$
- *plane reflection*: Use point reflection $[\![\mathsf{P}]\!] = \left[\!\!\left[\begin{matrix} \mathsf{I} - 2\mathbf{n}\mathbf{n}^T & 2\delta\mathbf{n} \\ 0^T & 1 \end{matrix}\right]\!\!\right]$. On planes as $[\![\mathsf{P}]\!]^{-T}$, on Plücker lines as more involved 6×6 matrix.
- *interpolation of general rotation*: non-elementary (done by specialized logarithm of matrix).

Linear algebra code typically consists of such coordinate tricks, applied to the points.
No direct circle rotation, or line reflection, or rotation generation at basic level.

5 The Six Tricks of Conformal Geometric Algebra

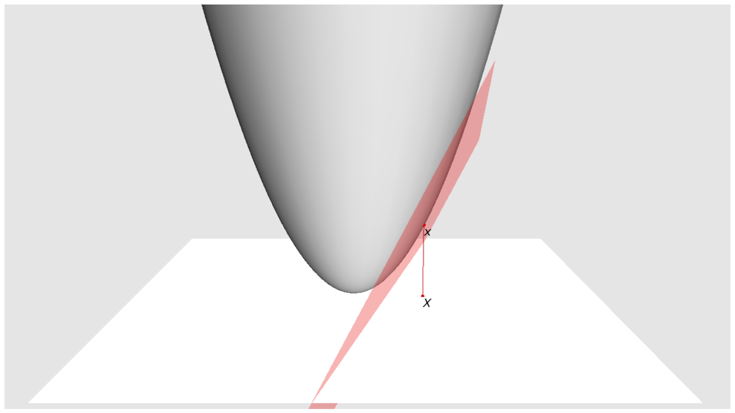
Consider Euclidean geometry not as a specific projective geometry, but as conformal geometry. Embed $\mathbb{R}^n$ isometrically into $\mathbb{R}^{n+1,1}$. Then we get a unification of techniques:



FIG(16,3)

1. Through the isometric embedding, conformal transformations of $\mathbb{R}^n$ are represented as orthogonal transformations of $\mathbb{R}^{n+1,1}$.
2. We represent orthogonal transformations as multiple reflections, and those using the geometric product of Clifford algebra as versors ('spinors'), which preserve structure.
3. We automatically get a non-metric outer product $\wedge$ as constructor for geometric primitives (points, lines, planes, spheres, circles, tangent vectors, directions etc). This gives structure.
4. We automatically get duality, providing and quantitative intersections and a metric inner product $\cdot$ to do projections.
5. Versors as exponentials of bivectors give the Lie algebra of motions. Logarithms then permit interpolation.
6. Efficient implementation uses the structural coherence to build a CGA compiler by automatic code generation.

6 Trick 1a: Euclidean Point Representation in $\mathbb{R}^{n+1,1}$ (CGA)



FIG(14,3): point

FIG(14,4): circle

FIG(14,6): circle meet

Represent point with 3D Euclidean position vector $\mathbf{x}$ in the *5D Minkowski space* $\mathbb{R}^{4,1}$ as a ray vector:

$$\boxed{x \;\sim\; e_o + \mathbf{x} + \tfrac{1}{2}\|\mathbf{x}\|^2 e_\infty}$$

where e_o is the standard point at the origin, $\mathbf{x}$ the Euclidean 'position vector', e_∞ is the point at infinity.

Basically, like *two extra homogeneous coordinates*:

$$x \sim (1, \mathbf{x}, \tfrac{1}{2}\|\mathbf{x}\|^2)^T$$

on the 5D basis $\{e_o, \mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3, e_\infty\}$.

7 Trick 1b: Inner Product Represents Squared Euclidean Distance

Metric of the representation space $\mathbb{R}^{n+1,1}$ is Minkowski. Switch to preferred basis:

$\cdot$	$\mathbf{x}$	e_+	e_-
$\mathbf{x}$	$\|\mathbf{x}\|^2$	0	0
e_+	0	1	0
e_-	0	0	-1

$$e_o = \tfrac{1}{2}(e_- + e_+)$$
$$e_\infty = e_- - e_+$$
$$\Longleftrightarrow$$

$\cdot$	e_o	$\mathbf{x}$	e_∞
e_o	0	0	-1
$\mathbf{x}$	0	$\|\mathbf{x}\|^2$	0
e_∞	-1	0	0

$$\bar{n} = -e_o$$
$$n = 2e_\infty$$
$$\Longleftrightarrow$$
(Lasenby's)

$\cdot$	$\bar{n}$	$\mathbf{x}$	n
$\bar{n}$	0	0	2
$\mathbf{x}$	0	$\|\mathbf{x}\|^2$	0
n	2	0	0

Now look what happens between two unit-weight points:

$$\begin{aligned} x \cdot y &= (e_o + \mathbf{x} + \tfrac{1}{2}\|\mathbf{x}\|^2\, e_\infty) \cdot (e_o + \mathbf{y} + \tfrac{1}{2}\|\mathbf{y}\|^2\, e_\infty) \\ &= (0 + 0 - \tfrac{1}{2}\|\mathbf{y}\|^2) + (0 + \mathbf{x} \cdot \mathbf{y} + 0) + (-\tfrac{1}{2}\|\mathbf{x}\|^2 + 0 + 0) \\ &= -\tfrac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2 \end{aligned}$$

Weird metric, nice trick: linearization of a squared distance.

The inner product in the representation space gives the squared Euclidean distance!

Therefore, Euclidean motions are represented by orthogonal transformations.

8 Inner Product Represents Squared Euclidean Distance (the Small Print)

Actually, the Euclidean distance corresponds to the inner product of unit weight vectors of the form $x = e_o + \mathbf{x} + \frac{1}{2}\|\mathbf{x}\|^2 e_\infty$. Any multiple $\alpha\mathbf{x}$ represents a point at the same location, but with a different *weight*. For the general distance formula, we need to normalize the weight to unity:

$$-\frac{1}{2} d_E(P,Q)^2 = \left(\frac{p}{-e_\infty \cdot p}\right) \cdot \left(\frac{q}{-e_\infty \cdot q}\right).$$

So Euclidean motions have to preserve

- the inner product (they are orthogonal transformations),
- the point at infinity e_∞.

The reason for the name 'conformal model' or Conformal Geometric Algebra (CGA) is:

Orthogonal transformations of $\mathbb{R}^{n+1,1}$ represent conformal transformations of $\mathbb{R}^n$.

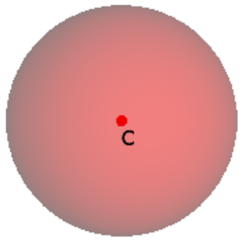
Conformal transformations generally preserve angles; for example, an isotropic scaling.

That is the context of Euclidean motions in this model. In homogeneous coordinates, the context was projective transformations. We don't have those in the conformal model.

9 Bonus: Vectors in $\mathbb{R}^{n+1,1}$ Represent Spheres and Planes in $\mathbb{R}^n$

- *general vectors of* $\mathbb{R}^{n+1,1}$ *are oriented, weighted (dual) spheres in* $\mathbb{R}^n$:

$$d_E^2(X,C) = \rho^2 \quad \Leftrightarrow \quad x \cdot c = -\tfrac{1}{2}\rho^2 \quad \Leftrightarrow \quad x \cdot (c - \tfrac{1}{2}\rho^2 e_\infty) = 0$$

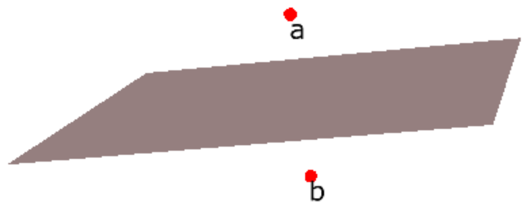


Squared norm gives radius: $(c - \frac{1}{2}\rho^2 e_\infty) \cdot (c - \frac{1}{2}\rho^2 e_\infty) = \rho^2$.
For 'imaginary' spheres, this is negative (so they are included!).
Points are (dual) spheres of zero radius.

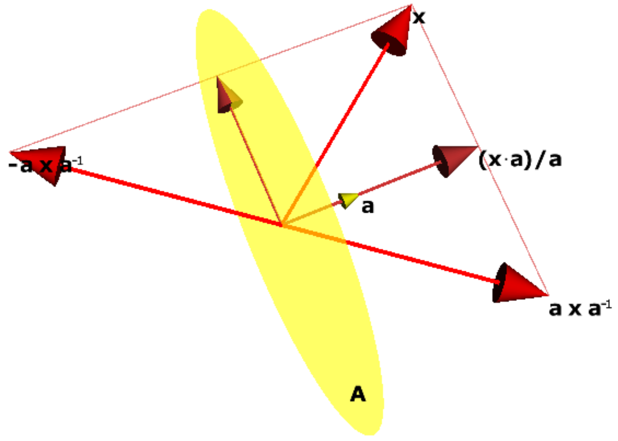
- *oriented, weighted (dual) planes of* $\mathbb{R}^n$ *are vectors in* $\mathbb{R}^{n+1,1}$ without e_o-component:

$$d_E^2(X,A) = d_E^2(X,B) \quad \Leftrightarrow \quad x \cdot a = x \cdot b \quad \Leftrightarrow \quad x \cdot (a - b) = 0$$

Note that the e_o-component satisfies $e_\infty \cdot (a - b) = 0$. (Geometrically, this means that the point at infinity is on all planes.) General dual plane is of the form $\mathbf{n} + \delta\, e_\infty$, with $\mathbf{n} \in \mathbb{R}^n$.



10 Trick 2: Geometric Reflections as Algebraic Sandwiching



FIG(7,1)

Reflection in an origin plane with unit normal **a**

$$\mathbf{x} \;\mapsto\; \mathbf{x} - 2(\mathbf{x}\cdot\mathbf{a})\,\mathbf{a}/\|\mathbf{a}\|^2 \quad \textit{(classic LA)}.$$

Now consider the dot product as the symmetric part of a more fundamental geometric product:

$$\mathbf{x}\cdot\mathbf{a} = \tfrac{1}{2}(\mathbf{x}\,\mathbf{a} + \mathbf{a}\,\mathbf{x}).$$

Then rewrite:

$$\begin{aligned}\mathbf{x} \;\mapsto\; & \mathbf{x} - (\mathbf{x}\,\mathbf{a} + \mathbf{a}\,\mathbf{x})\,\mathbf{a}/\|\mathbf{a}\|^2 \quad \textit{(GA product)} \\ & = \boxed{-\mathbf{a}\,\mathbf{x}\,\mathbf{a}^{-1}}\end{aligned}$$

with *the geometric inverse of a vector*: $\mathbf{a}^{-1} = \mathbf{a}/\|\mathbf{a}\|^2$.

Inverse works because
$\mathbf{a}^{-1}\,\mathbf{a} = \mathbf{a}\,\mathbf{a}/\|\mathbf{a}\|^2 = \frac{1}{2}(\mathbf{a}\,\mathbf{a} + \mathbf{a}\,\mathbf{a})/\|\mathbf{a}\|^2 = \mathbf{a}\cdot\mathbf{a}/\|\mathbf{a}\|^2 = 1.$

11 Bonus: Structural Transfer: Reflection Applied to Point

We should reflect the point x itself, rather than merely its Euclidean part $\mathbf{x}$:

$$\mathbf{x} \;\mapsto\; -\,\mathbf{a}\,\mathbf{x}\,\mathbf{a}^{-1}.$$

Try structural transfer to CGA:

$$\begin{tikzcd}
\boxed{\text{position } \mathbf{x}} \arrow[r, "\text{normal vector}"] \arrow[d, "\text{embed in CGA}"'] & \mathbf{a} \arrow[r, "\text{as reflector}"] \arrow[d, "\text{embed in CGA}"'] & \boxed{\text{position } -\mathbf{a}\,\mathbf{x}\,\mathbf{a}^{-1}} \arrow[d, "\text{embed in CGA}"'] \\
\boxed{\text{point } x} \arrow[r, "\text{origin plane}"] & \mathbf{a} \arrow[r, "\text{as reflector}"] & \boxed{\text{point } -\mathbf{a}\,x\,\mathbf{a}^{-1}}
\end{tikzcd}$$

Use linearity of the geometric product:

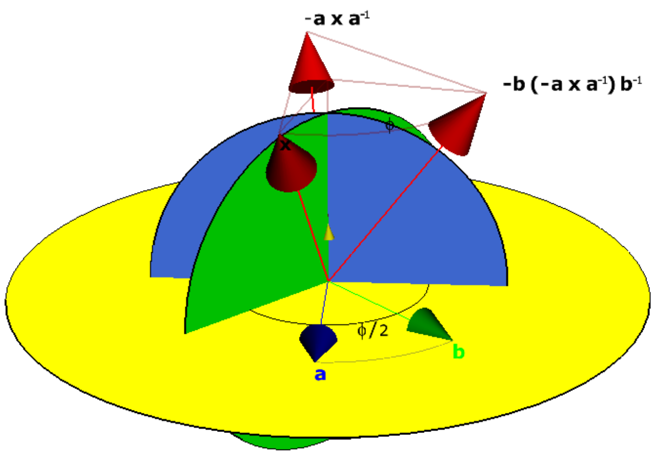
$$\begin{aligned}
x \;&\overset{?}{\mapsto}\; -\mathbf{a}\,x\,\mathbf{a}^{-1} \\
&=\; -\mathbf{a}\,(e_o + \mathbf{x} + \tfrac{1}{2}\|\mathbf{x}\|^2 e_\infty)\,\mathbf{a}^{-1} \\
&=\; -\mathbf{a}\,e_o\,\mathbf{a}^{-1} - \mathbf{a}\,\mathbf{x}\,\mathbf{a}^{-1} - \tfrac{1}{2}\|\mathbf{x}\|^2\mathbf{a}\,e_\infty\,\mathbf{a}^{-1}.
\end{aligned}$$

Now use $0 = \mathbf{a}\cdot e_o = \frac{1}{2}(\mathbf{a}\,e_o + e_o\,\mathbf{a})$ so that $-\mathbf{a}\,e_o = e_o\,\mathbf{a}$, same for e_∞. And $\|\mathbf{x}\|^2 = \|-\mathbf{a}\,\mathbf{x}\,\mathbf{a}^{-1}\|^2$.

$$\begin{aligned}
x \;&\overset{?}{\mapsto}\; e_o\,\mathbf{a}\,\mathbf{a}^{-1} - \mathbf{a}\,\mathbf{x}\,\mathbf{a}^{-1} + \tfrac{1}{2}\|\mathbf{x}\|^2 e_\infty\,\mathbf{a}\,\mathbf{a}^{-1} \\
&=\; e_o - \mathbf{a}\,\mathbf{x}\,\mathbf{a}^{-1} + \tfrac{1}{2}\|\mathbf{x}\|^2 e_\infty \\
&\overset{!}{=}\; e_o - \mathbf{a}\,\mathbf{x}\,\mathbf{a}^{-1} + \tfrac{1}{2}\|-\mathbf{a}\,\mathbf{x}\,\mathbf{a}^{-1}\|^2 e_\infty
\end{aligned}$$

So indeed a conformal point at the reflected location. Automatic in the algebra.

12 Bonus: Orthogonal Transformations as Versors



FIG(7,2)

A reflection in two successive origin planes **a** and **b**:

$$\begin{aligned} \mathbf{x} &\mapsto -\mathbf{b}\,(-\mathbf{a}\,\mathbf{x}\,\mathbf{a}^{-1})\,\mathbf{b}^{-1} \\ &= (\mathbf{b}\,\mathbf{a})\,\mathbf{x}\,(\mathbf{b}\,\mathbf{a})^{-1} \end{aligned}$$

So a rotation is represented by the geometric product of two vectors **b a**, also an element of the algebra.

(Actually, in 3D these are quaternions.)

Multiple reflections are the fundamental representation for operators in GA:

The geometric product of vectors is called a versor.
It acts as an orthogonal transformation by sandwiching.

As we will see, versors are structure-preserving.

(It is common to use normalized versors and call those rotors.)

13 The Fundamental Geometric Product

Take a real metric vector space $\mathbb{R}^n$ with inner product $\cdot : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$.

Define the *geometric product* through:

- *associative*: $X\,(Y\,Z) = (X\,Y)\,Z$.
- *linear*: $X\,(\alpha Y + \beta Z) = \alpha\, X\,Y + \beta\, X\,Z$.
- *vectors have scalar square*: $\mathbf{x}\,\mathbf{x} = \mathbf{x}\cdot\mathbf{x}$.

That's all, mathematically. Not necessarily commutative.

Starting from vectors in $\mathbb{R}^n$, one generates the geometric algebra of a 2^n-dimensional dimensional multivector space, sometimes denoted $\mathbb{R}_n$.

Alternatively, define the Clifford algebra as the quotient of a tensor algebra on the metric space (V, Q) with the two-sided ideal of $\mathbf{x} \otimes \mathbf{x} - Q(\mathbf{x}, \mathbf{x})$. If you're a mathematician, you will probably find this enlightening and reassuring.

14 3D Geometric Product of Vectors Expressed in Coordinates

Introduce coordinates on orthonormal basis$\{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3\}$. Note that

$$\begin{aligned} 1 = \mathbf{e}_1 \cdot \mathbf{e}_1 = \tfrac{1}{2}(\mathbf{e}_1\,\mathbf{e}_1 + \mathbf{e}_1\,\mathbf{e}_1), \quad \text{so} \quad \mathbf{e}_1\,\mathbf{e}_1 = 1. \\ 0 = \mathbf{e}_1 \cdot \mathbf{e}_2 = \tfrac{1}{2}(\mathbf{e}_1\,\mathbf{e}_2 + \mathbf{e}_2\,\mathbf{e}_1), \quad \text{so} \quad \mathbf{e}_1\,\mathbf{e}_2 = -\mathbf{e}_2\,\mathbf{e}_1. \end{aligned}$$

Then for $\mathbf{x} = x_1\mathbf{e}_1 + x_2\mathbf{e}_2 + x_3\mathbf{e}_3$, and $\mathbf{y} = y_1\mathbf{e}_1 + y_2\mathbf{e}_2 + y_3\mathbf{e}_3$, we get:

$$\begin{aligned} \mathbf{x}\,\mathbf{y} &= x_1y_1 + x_2y_2 + x_3y_3 + (x_2y_3 - y_2x_3)\,\mathbf{e}_2\,\mathbf{e}_3 + (x_3y_1 - y_3x_1)\,\mathbf{e}_3\,\mathbf{e}_1 + (x_1y_2 - y_1x_2)\,\mathbf{e}_1\,\mathbf{e}_2 \\ &= (\mathbf{x}\cdot\mathbf{y}) + (\mathbf{x}\times\mathbf{y})\,(\mathbf{e}_3\,\mathbf{e}_2\,\mathbf{e}_1). \end{aligned}$$

In 3D, recognizable classical products. But mixed 'grades': scalars and $\mathbf{e}_1\,\mathbf{e}_2$-elements!

The elements $\mathbf{e}_1\,\mathbf{e}_2$ etc. have the special property $(\mathbf{e}_1\,\mathbf{e}_2)^2 = -1$:

$$(\mathbf{e}_1\,\mathbf{e}_2)^2 = (\mathbf{e}_1\,\mathbf{e}_2)\,(\mathbf{e}_1\,\mathbf{e}_2) = \mathbf{e}_1\,(\mathbf{e}_2\,\mathbf{e}_1)\,\mathbf{e}_2) = -\mathbf{e}_1\,(\mathbf{e}_1\,\mathbf{e}_2)\,\mathbf{e}_2 = -(\mathbf{e}_1\,\mathbf{e}_1)\,(\mathbf{e}_2\,\mathbf{e}_2) = -1.$$

That is how complex numbers and quaternions are naturally embedded. Known to be handy for rotations in 2D and 3D, and now also beyond.

15 Bonus: Vectors, Complex Numbers and Quaternions

- The full geometric algebra of the 2D plane involves:

$$\Big\{ \underbrace{1}_{scalars}, \quad \underbrace{\mathbf{e}_1, \mathbf{e}_2}_{vector\ space}, \quad \underbrace{\mathbf{e}_1\,\mathbf{e}_2}_{bivector\ space} \Big\}$$

  Vectors are only half of this ('the elements').
  Complex numbers of the form $\alpha + \beta\mathbf{e}_1\,\mathbf{e}_2$ are the other half ('the operators').

- The full geometric algebra of 3D space involves:

$$\Big\{ \underbrace{1}_{scalars}, \quad \underbrace{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3}_{vector\ space}, \quad \underbrace{\mathbf{e}_1\,\mathbf{e}_2,\ \mathbf{e}_2\,\mathbf{e}_3,\ \mathbf{e}_3\,\mathbf{e}_1}_{bivector\ space}, \quad \underbrace{\mathbf{e}_1\,\mathbf{e}_2\,\mathbf{e}_3}_{trivector\ space} \Big\}$$

  Vectors and determinants are only half of this ('the elements').
  Quaternions of the form $\alpha + \beta_1\mathbf{e}_2\,\mathbf{e}_3 + \beta_2\mathbf{e}_3\,\mathbf{e}_1 + \beta_3\mathbf{e}_1\,\mathbf{e}_2$ are the other half ('the operators').
  (We will see that normal vectors of planes are bivectors divided by trivectors.)

You need a good representation that keeps elements and operators separate, yet makes them interact.
Characterizing elements as operators by singling out a special (real) axis is clumsy.

16 Trick 3: Constructing Elements by Anti-Symmetry

Skew-symmetric part of geometric product gives outer product (of Grassmann algebra):

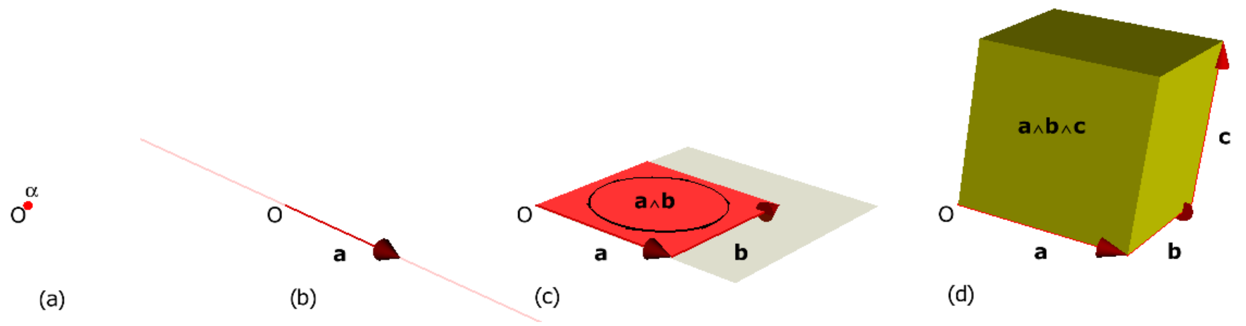
$$\mathbf{x} \wedge \mathbf{a} = \tfrac{1}{2}(\mathbf{x}\,\mathbf{a} - \mathbf{a}\,\mathbf{x})$$

It is bilinear and associative. Use it to span oriented subspaces as Grassmannians of various grades (dimensionality):

$$\mathbf{x} \wedge (\mathbf{a}_1 \wedge \cdots \wedge \mathbf{a}_k) = 0 \quad \Longleftrightarrow \quad \mathbf{x} \text{ in span}(\mathbf{a}_1, \cdots, \mathbf{a}_k)$$

For instance, $\mathbf{x} = \lambda_1\,\mathbf{a}_1 + \lambda_2\,\mathbf{a}_2$ is a vector in span$(\mathbf{a}_1, \mathbf{a}_2)$, and we verify:

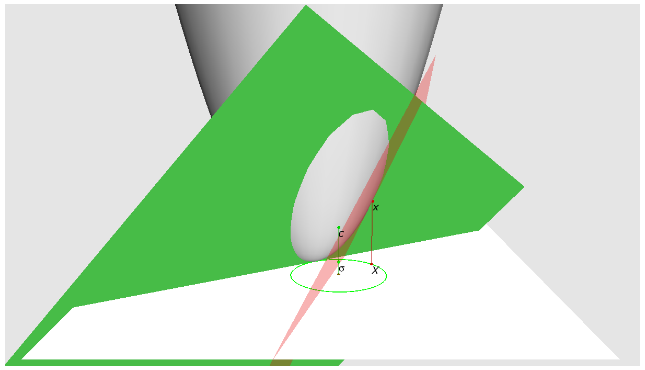
$$\begin{aligned} \mathbf{x} \wedge (\mathbf{a}_1 \wedge \mathbf{a}_2) &= \lambda_1\,\mathbf{a}_1 \wedge \mathbf{a}_1 \wedge \mathbf{a}_2 + \lambda_2\,\mathbf{a}_2 \wedge \mathbf{a}_1 \wedge \mathbf{a}_2 \\ &= \lambda_1\,(\mathbf{a}_1 \wedge \mathbf{a}_1) \wedge \mathbf{a}_2 - \lambda_2\,\mathbf{a}_1 \wedge (\mathbf{a}_2 \wedge \mathbf{a}_2) = 0 + 0 = 0. \end{aligned}$$



$$\left\{ \underbrace{1}_{\textit{scalars}}, \quad \underbrace{\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3}_{\textit{vector space}}, \quad \underbrace{\mathbf{e}_1 \wedge \mathbf{e}_2,\ \mathbf{e}_2 \wedge \mathbf{e}_3,\ \mathbf{e}_3 \wedge \mathbf{e}_1,}_{\textit{bivector space}} \quad \underbrace{\mathbf{e}_1 \wedge \mathbf{e}_2 \wedge \mathbf{e}_3}_{\textit{trivector space}} \right\}$$

17 The Outer Product in the Conformal Model

The basic elements of Euclidean geometry are all represented as $\wedge$-factorizable multivectors ('blades') in CGA. They represent proper subspaces in the representational space, but are interpretable in the Euclidean space by determining the point representatives on them.



FIG(14,4)

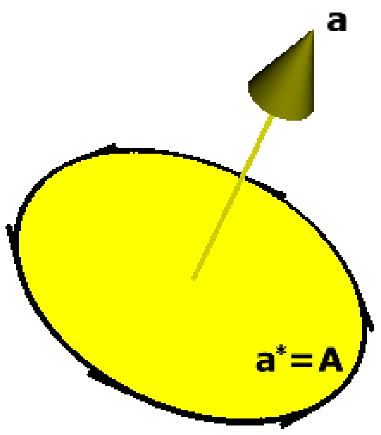
- *Rounds: spheres, circles, point pairs*
  $a \wedge b \wedge c \wedge d$ is an (oriented & weighted) sphere
  passing orthogonally through 4 given spheres (or points), etc.
- *Flats: planes, lines, flat points*
  $a \wedge b \wedge c \wedge e_\infty$ is an (oriented & weighted) plane
  passing orthogonally through 3 given spheres (or points), etc.
- *k-dimensional direction elements*
  $\mathbf{B} \wedge e_\infty$ is a pure Euclidean k-space $\mathbf{B}$, at infinity.
- *k-dimensional tangents*
  $e_o \wedge \mathbf{B}$ is the tangent k-space $\mathbf{B}$ at the origin.
  Soon: $p \cdot (p \wedge \mathbf{B} \wedge e_\infty)$ is the tangent k-space $\mathbf{B}$ at p.

18 Bonus: Dualization/Orthogonal Complement

There is also a dual representation of subspaces. It is obtained simply by dividing a blade X by the volume blade I_n of the n-D vector space (aka the *pseudoscalar*).

$$X^* \equiv X / I_n$$

If grade$(X) = k$, in n-D space, then grade$(X^*) = n - k$.



The dual of a direction vector **a** *in 3D is a 2-blade* **A** *of which* **a** *is the normal vector.*

It allows switching between dual representation, and direct (i.e., primal) representation.

direct sphere representation: $\Sigma = a \wedge b \wedge c \wedge d$ of grade 4, points satisfy $x \wedge \Sigma = 0$

$$\longleftrightarrow$$

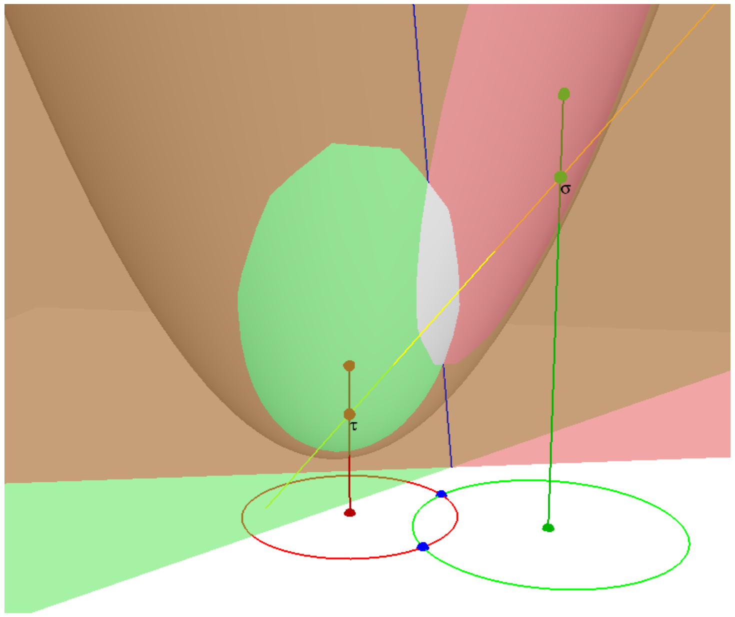
dual sphere representation: $\sigma = m - \frac{1}{2}\rho^2 e_\infty$ of grade 1, points satisfy $x \cdot \sigma = 0$

19 Bonus: Intersection of Elements

The intersection (meet) of two subspaces is dually given as the product of the duals:

$$\text{meet:} \quad (A \cap B)^* = B^* \wedge A^*.$$

with duality relative to the pseudoscalar of the smallest subspace containing A and B (their join).



FIG(14,6)

In CGA, the meet of course also applies to the Euclidean elements.

Example: Intersection of Euclidean circles in 2D is like the intersection of their subspace blades, which are general planes in the 4D conformal model. That gives a line, on which their are two point representatives. So two circles meet in a point pair (possibly imaginary).

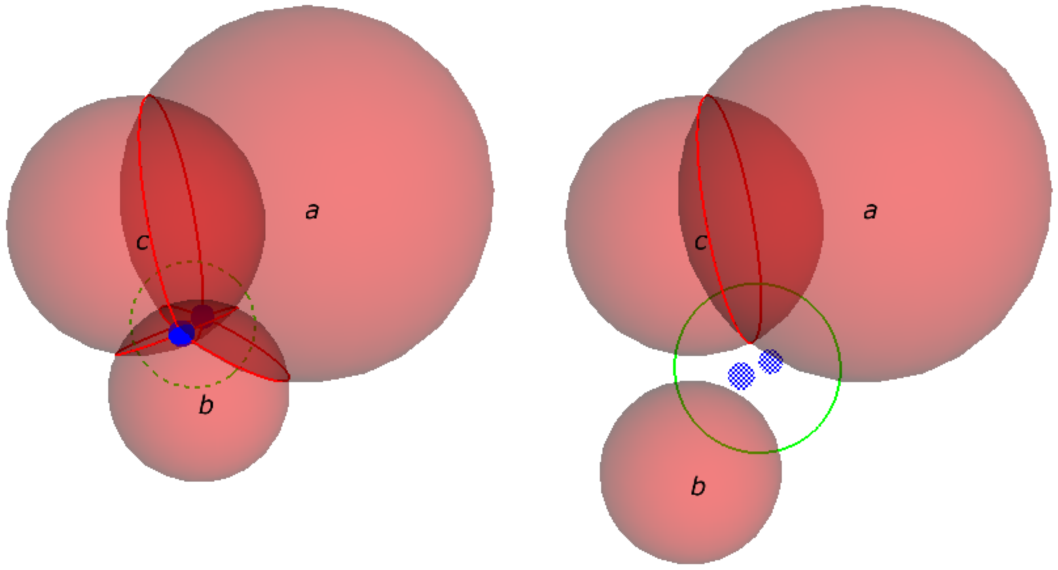
20 Extra: Plunge of Elements (A New Operation)

Also, the orthogonal element operation (plunge) can be defined:

$$\text{plunge:} \quad A \perp B = B^* \wedge A^*.$$

It determines the subspace that hits A and B orthogonally. It is dual to the meet.

For three spheres, when one is real, the other is 'imaginary' (negative squared radius).

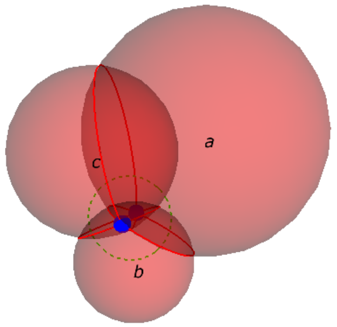


FIG(15,1)

21 Big Bonus: Euclid's Elements Through Closure

By starting from spheres or planes dually represented as vectors, and making plunges and meets, one gets the Euclidean menagerie.

FIG(15,1)



- *dual sphere:* the vector $\sigma = c - \frac{1}{2}\rho^2 e_\infty$.

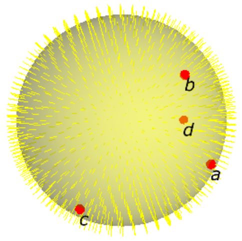
$$0 = x \cdot \sigma = x \cdot c - \tfrac{1}{2}\rho^2 x \cdot e_\infty = -\tfrac{1}{2}\big((\mathbf{x}-\mathbf{c})^2 - \rho^2\big).$$

- *dual plane:* the vector $\pi = \mathbf{n} + \delta\, e_\infty$.

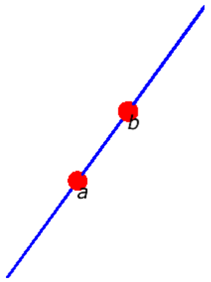
$$0 = x \cdot \pi = x \cdot \mathbf{n} + \delta\, x \cdot e_\infty = \mathbf{x} \cdot \mathbf{n} - \delta.$$

- *dual circle at origin:* the 2-blade $K = \sigma_0 \wedge \pi_0 = (e_o - \frac{1}{2}\rho^2 e_\infty) \wedge \mathbf{n}$.

$$0 = x \cdot (\sigma_0 \wedge \pi_0) = (x \cdot \sigma_0)\,\pi_0 - \sigma_0\,(x \cdot \pi_0).$$



- *(primal) sphere:* the 4-blade $\Sigma = a \wedge b \wedge c \wedge d$.
- *line:* the 3-blade $L = a \wedge b \wedge e_\infty = a \wedge \mathbf{u} \wedge e_\infty$.
- *direction blade:* the 3-blade $\mathbf{E} \wedge e_\infty$ is a 2-direction.
- *tangent vector:* the 3-blade $T = a \cdot (a \wedge \mathbf{u} \wedge e_\infty)$.



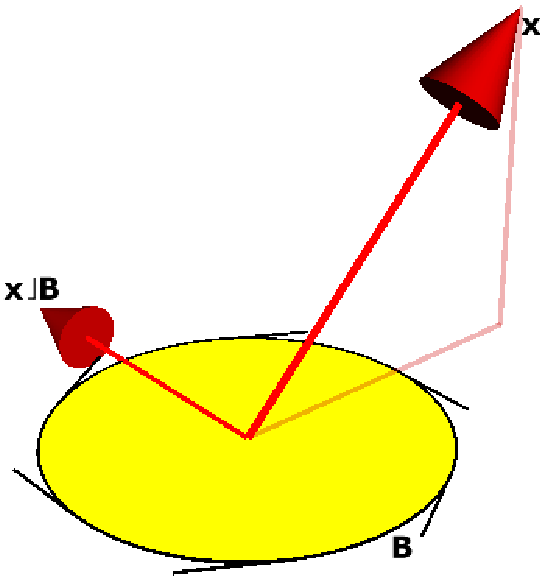
22 Trick 4: The Inner Product

We can define the inner product (aka as the contraction) as dual to the outer product:

$$A \cdot B \equiv (A \wedge B^*)^{-*},$$

extending the dot product to blades. It is bilinear, but non-associative. It has properties like:

$$\mathbf{x} \cdot (\mathbf{a} \wedge \mathbf{b}) = (\mathbf{x} \cdot \mathbf{a})\,\mathbf{b} - \mathbf{a}\,(\mathbf{x} \cdot \mathbf{b}).$$

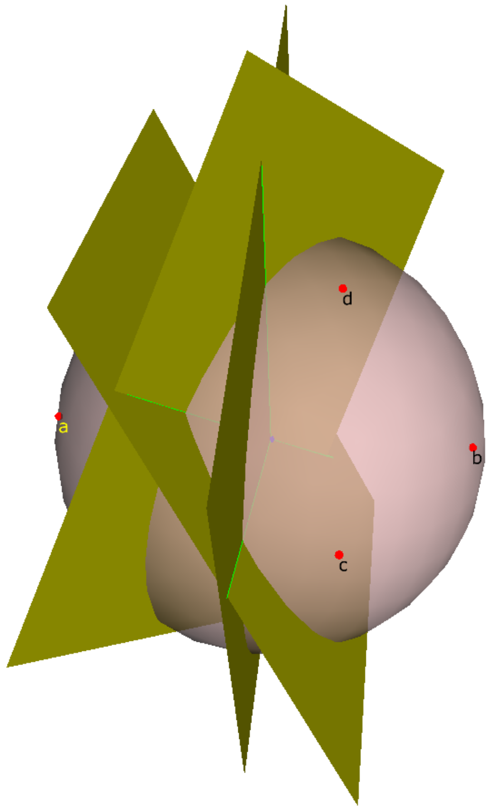


Interpretation of the inner product $\mathbf{x} \cdot \mathbf{B}$ for direction blades:

the orthogonal complement of $\mathbf{x}$ contained in $\mathbf{B}$.

23 Consistency of Dual Sphere Representations

The 4-blade $\Sigma = a \wedge b \wedge c \wedge d$ represents the sphere through the four points a, b, c, d.



DEMOspheres()

$$\begin{aligned}
\Sigma &= a \wedge b \wedge c \wedge d \\
&= a \wedge \underbrace{(b-a) \wedge (c-a) \wedge (d-a)}_{\textit{dual } \textsf{meet} \textit{ of 3 midplanes}} \\
&\propto a \wedge \underbrace{(m \wedge e_\infty)}_{\textit{flat center}}{}^{*} \\
&= \Big(a \cdot (m \wedge e_\infty)\Big)^{*} \\
&= \Big((a \cdot m)\, e_\infty - (a \cdot e_\infty) m\Big)^{*} \\
&= -(\underbrace{m - \tfrac{1}{2}\rho^2 e_\infty}_{\textit{dual sphere}})^{*} \\
&= \sigma^{-*}
\end{aligned}$$

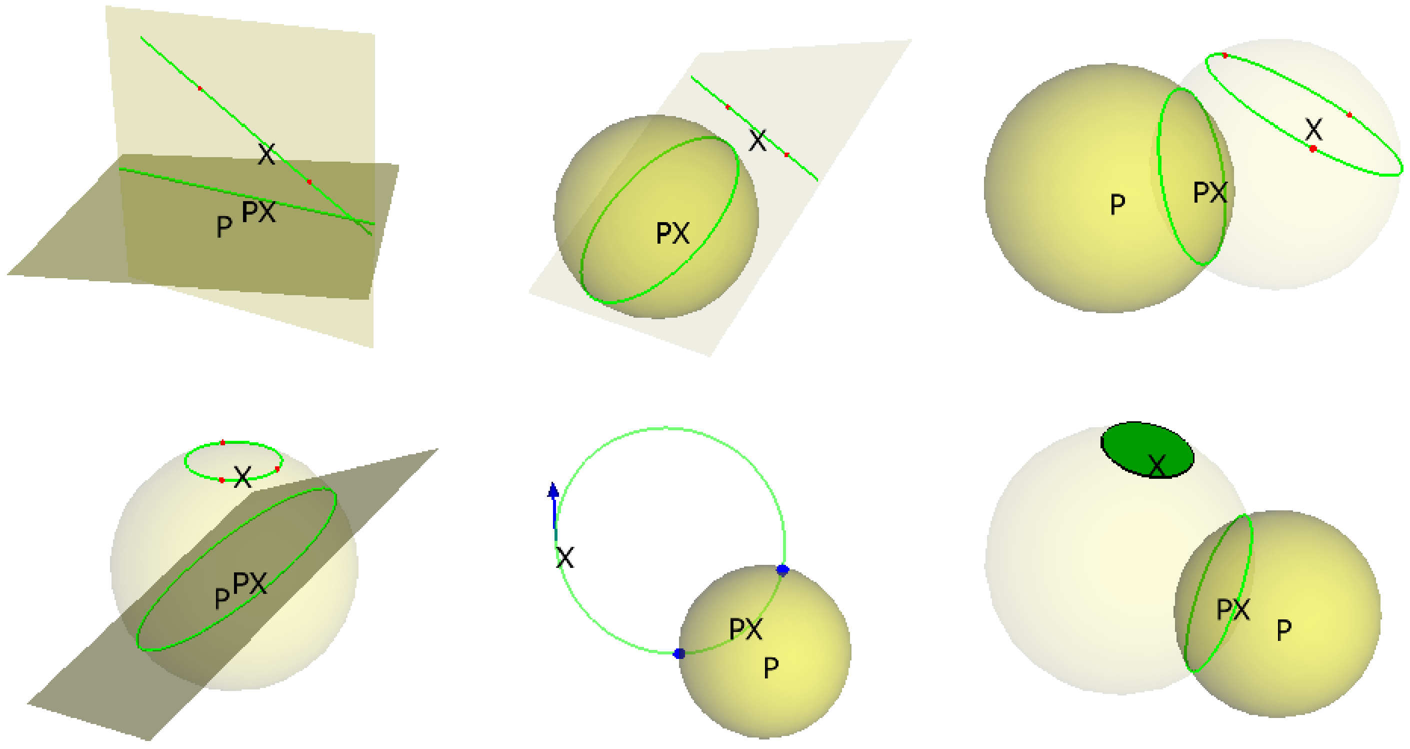
So it is easy to determine center and radius of a sphere through 4 points: they are the components of $(a \wedge b \wedge c \wedge d)^*$ (normalized).

24 Extra: Using the Inner Product for General Projections

CGA has a general projection operator, a structural generalization of $\mathbf{x} \mapsto (\mathbf{x}\cdot\mathbf{a})/\mathbf{a}$:

$$\mathsf{Proj}_P(X) = (X\cdot P)/P \qquad \Big(= P^{-1}\cap(X\wedge P^{-*})\ \Big).$$

This meets P^{-1} with a Euclidean element $X\wedge P^{-*}$ 'containing X and plunging into P orthogonally'.



DEMOprojection()

25 The Really Great Thing about CGA: Motions as Versors

Back to the motions. Make them through multiple reflections in the representative vectors of (dual) planes and spheres.

- Translation: two parallel planes, $\mathbf{t}/2$ apart.
  Computation: $(\mathbf{t} + \frac{1}{2}\mathbf{t} \cdot \mathbf{t}\, e_\infty)\, \mathbf{t} = \mathbf{t}^2\, (1 - \mathbf{t}\, e_\infty/2)$.

$$\text{translation versor: } T_{\mathbf{t}} = 1 - \mathbf{t}\, e_\infty/2$$

- Rotation at Origin: two intersecting planes, $\phi/2$ apart.
  Computation: $(\cos(\phi/2)\mathbf{e}_1 + \sin(\phi/2)\mathbf{e}_2)\, \mathbf{e}_1 = \cos(\phi/2) - \sin(\phi/2)\, \mathbf{e}_1 \wedge \mathbf{e}_2$.

$$\text{rotation versor: } R_{\mathbf{I}\phi} = \cos(\phi/2) - \sin(\phi/2)\, \mathbf{I}$$

- Uniform scaling at Origin: two concentric spheres.
  Computation: $(e_o - \frac{1}{2}\rho^2 e_\infty)\, (e_o - \frac{1}{2} e_\infty) = -\frac{1}{2}\Big((1+\rho^2) - (1-\rho^2) e_o \wedge e_\infty\Big)$.

$$\text{uniform scaling versor: } S_\gamma = \cosh(\gamma/2) + \sinh(\gamma/2)\, e_o \wedge e_\infty$$

- Transversion at Origin: two touching spheres of equal radius.

$$\text{transversion versor: } V_{\mathbf{v}} = 1 + e_o\, \mathbf{v}$$

26 Big Bonus: Structure Preservation

Reflection of vector $\mathbf{x}$ in a plane with normal $\mathbf{a}$ is: $\mathbf{x} \;\mapsto\; -\mathbf{a}\,\mathbf{x}\,\mathbf{a}^{-1}$. If X is a product of vectors $\mathbf{x}_i$, this extends to:

$$X \;\mapsto\; (-\mathbf{a}\,\mathbf{x}_1\,\mathbf{a}^{-1})\,(-\mathbf{a}\,\mathbf{x}_2\,\mathbf{a}^{-1})\,\cdots(-\mathbf{a}\,\mathbf{x}_k\,\mathbf{a}^{-1}) = \mathbf{a}\,\widehat{X}\,\mathbf{a}^{-1}$$

with $\widehat{\mathbf{x}} = (-1)^{\dim(\mathbf{x})}\,\mathbf{X}$, the 'main involution'. Then distributes over 'constructive' products $\wedge$ and $\cdot$, effectively weighted sums of geometric products:

$$\mathbf{B} \;\mapsto\; (-\mathbf{a}\,\mathbf{b}_1\,\mathbf{a}^{-1}) \wedge (-\mathbf{a}\,\mathbf{b}_2\,\mathbf{a}^{-1}) \wedge \cdots (-\mathbf{a}\,\mathbf{b}_k\,\mathbf{a}^{-1}) = (-1)^k \mathbf{a}\left(\mathbf{b}_1 \wedge \mathbf{b}_2 \wedge \cdots \mathbf{b}_k\right)\mathbf{a}^{-1} = \mathbf{a}\,\widehat{\mathbf{B}}\,\mathbf{a}^{-1}.$$

So

$$\boxed{\textit{the sandwiching product is structure preserving.}}$$

Also preserved in multiple reflections, so now all motions are universally applicable to all geometric elements using their versors:

$$\boxed{\begin{array}{rcl} V\text{even}: \; X & \mapsto & V\,X\,V^{-1} \\ V\text{odd}: \; X & \mapsto & V\,\widehat{X}\,V^{-1} \end{array}}$$

This universality is very unlike the homogeneous coordinate approach, and enormously simplifies software.

27 Transfer Principle for Reflection

For reflection $\mathbf{x} \mapsto -\mathbf{a}\,\mathbf{x}\,\mathbf{a}^{-1}$: transfer from direction vector to origin line to general line. Simultaneously, the dual origin plane becomes the general dual plane through the intersection point. Further extension to inversion possible by conformal versor.

$$\begin{tikzcd}
\boxed{\text{direction } \mathbf{u}} \arrow[r, "\text{normal vector}"] \arrow[d, "\text{embed in GA}"'] & \mathbf{a} \arrow[r, "\text{as reflector}"] \arrow[d, "\text{embed in GA}"'] & \boxed{\text{direction } \mathbf{u} - 2(\mathbf{u}\cdot\mathbf{a})\,\mathbf{a}/\|\mathbf{a}\|^2)} \arrow[d, "\text{embed in GA}"'] \\
\boxed{\text{direction } \mathbf{u}} \arrow[r, "\text{normal vector}"] \arrow[d, "\text{embed in CGA}"'] & \mathbf{a} \arrow[r, "\text{as reflector}"] \arrow[d, "\text{embed in CGA}"'] & \boxed{\text{direction } \mathbf{u}' = -\mathbf{a}\,\mathbf{u}\,\mathbf{a}^{-1}} \arrow[d, "\text{embed in CGA}"'] \\
\boxed{\text{origin line } \Lambda_o = e_o \wedge \mathbf{u} \wedge e_\infty} \arrow[r, "\text{dual plane}"] \arrow[d, "\text{Euclidean versor}"'] & \pi_o \arrow[r, "\text{as reflector}"] \arrow[d, "\text{Euclidean versor}"'] & \boxed{\text{origin line } \Lambda_o' = -\pi_o\,\Lambda_o\,\pi_o^{-1}} \arrow[d, "\text{Euclidean versor}"'] \\
\boxed{\text{general line } \Lambda = V\,\Lambda_o\,V^{-1}} \arrow[r, "\text{dual plane}"] \arrow[d, "\text{conformal versor}"'] & \pi \arrow[r, "\text{as reflector}"] \arrow[d, "\text{conformal versor}"'] & \boxed{\text{general line } \Lambda' = -\pi\,\Lambda\,\pi^{-1}} \arrow[d, "\text{conformal versor}"'] \\
\boxed{\text{general circle } K = V\,\Lambda\,V^{-1}} \arrow[r, "\text{dual sphere}"] & \sigma \arrow[r, "\text{as invertor}"] & \boxed{\text{general circle } K' = -\sigma\,K\,\sigma^{-1}}
\end{tikzcd}$$

This all just holds by structure preservation, no need to prove anything!

28 Trick 5: The Bivector Representation of Even Versors

In a rotation, we have the rotation plane $\mathbf{I}$ and angle ϕ. What is the versor?

Versors were introduced through products of invertible vectors. But an even versor V can be written as the *exponential of a bivector* B:

$$V = e^{B}$$

The bivector specification often corresponds more directly to the geometry.

Example: rotation over ϕ in plane $\mathbf{I}$ through the origin as exponential:

$$R = \cos(\phi/2) - \mathbf{I}\sin(\phi/2) = 1 + (-\mathbf{I}\phi/2) + \tfrac{1}{2!}(-\mathbf{I}\phi/2)^2 + \cdots = e^{-\mathbf{I}\phi/2},$$

since $\mathbf{I}^2 = -1$. We will show that a general 3D rotation with rotation line Λ is $R = e^{\Lambda^*\phi/2}$.

Example: translation over $\mathbf{t}$:

$$T = 1 - \mathbf{t}\wedge e_\infty/2 = 1 - \mathbf{t}\wedge e_\infty/2 + \tfrac{1}{2!}(-\mathbf{t}\wedge e_\infty/2)^2 + \cdots = e^{-\mathbf{t}\wedge e_\infty/2},$$

since $(\mathbf{t}\wedge e_\infty)^2 = (\mathbf{t}\,e_\infty)^2 = \mathbf{t}\,e_\infty\,\mathbf{t}\,e_\infty = -\mathbf{t}\,\mathbf{t}\,e_\infty\,e_\infty = -\mathbf{t}^2\,e_\infty^2 = 0.$

29 Bonus: Transfer Principle to Make General Versors

Transfer principle for exponentially represented versors:

$$\begin{aligned} V\ \exp(B)\, V^{-1} &= V\,(1 + B + \tfrac{1}{2!} B\, B + \cdots)\, V^{-1} \\ &= 1 + V\, B\, V^{-1} + \tfrac{1}{2!}(V\, B\, V^{-1})\,(V\, B\, V^{-1}) + \cdots \\ &= \exp(V\, B\, V^{-1}). \end{aligned}$$

Application: the conformal versor for rotation around an arbitrary unit line Λ, over ϕ.

$$\begin{tikzcd}
\boxed{\text{2-blade } \mathbf{I}} \arrow[r, "\text{exponentiate}"] \arrow[d, "\text{rewrite using Euclidean 3D duality}"] & \boxed{\text{‘quaternion’ } e^{-\mathbf{I}\phi/2}} \arrow[d, "\text{rewrite using Euclidean 3D duality}"] \\
\boxed{\text{axial vector } \mathbf{a} = \mathbf{I}^\star} \arrow[r, "\text{exponentiate}"] \arrow[d, "\text{line in CGA}"] & \boxed{\text{‘quaternion’ } e^{\mathbf{a}^\star\phi/2}} \arrow[d, "\text{line in CGA}"] \\
\boxed{\text{origin line } \Lambda_o = e_o \wedge \mathbf{a} \wedge e_\infty} \arrow[r, "\text{exponentiate}"] \arrow[d, "\text{Euclidean motion}"] & \boxed{\text{conformal versor } e^{\Lambda_0{}^*\phi/2}} \arrow[d, "\text{Euclidean motion}"] \\
\boxed{\text{general line } \Lambda = V\,\Lambda_o\,V^{-1}} \arrow[r, "\text{exponentiate}"] & \boxed{\text{conformal versor } e^{\Lambda^*\phi/2}}
\end{tikzcd}$$

The only thing to prove here is the duality transfer from 3D Euclidean to conformal:

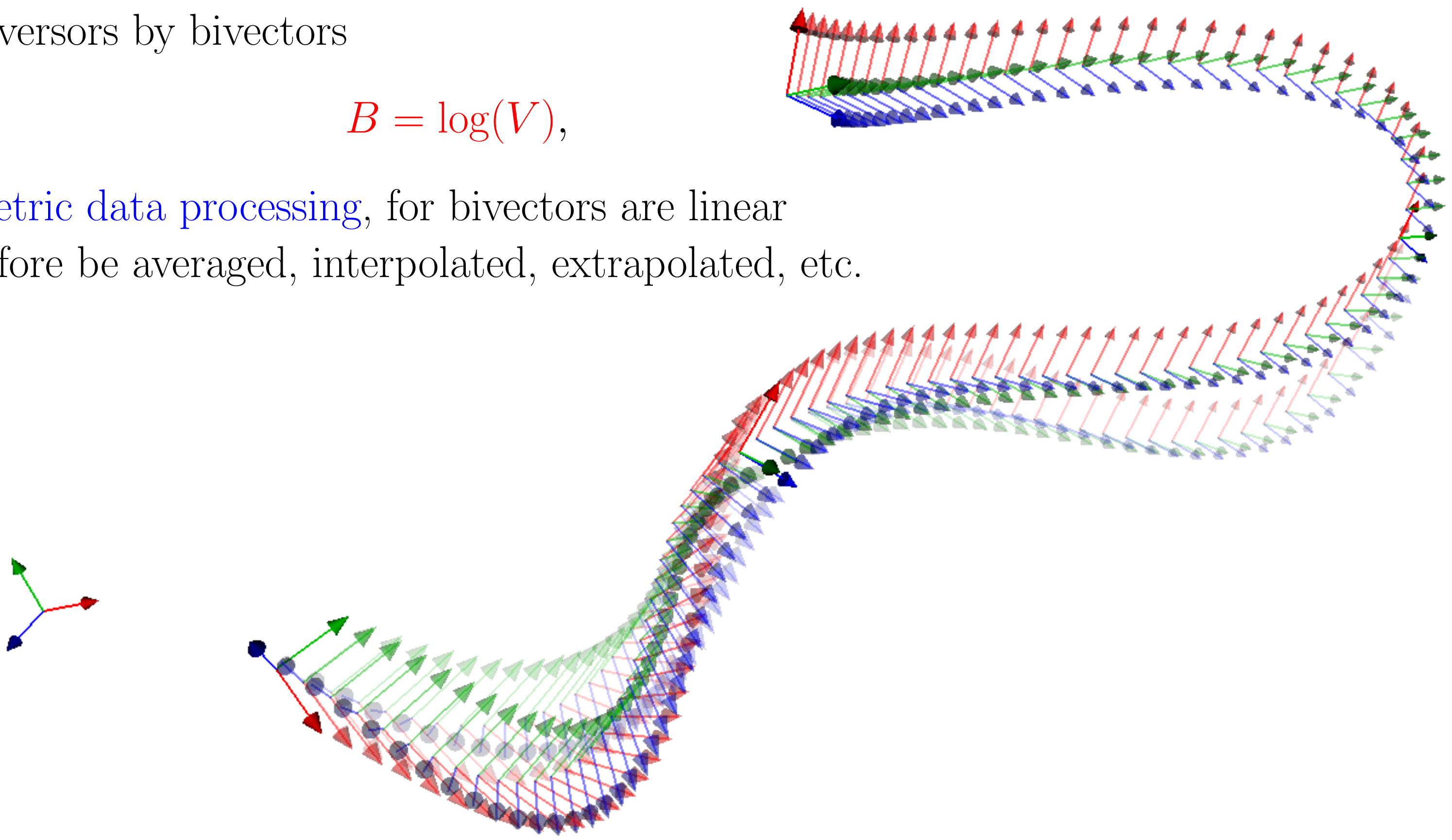
$$\Lambda_o{}^* = (e_o \wedge \mathbf{a} \wedge e_\infty)^* = (e_o \wedge \mathbf{a} \wedge e_\infty)\,(e_o \wedge \mathbf{I}_3^{-1} \wedge e_\infty) = \mathbf{a}\,\mathbf{I}_3^{-1} = \mathbf{a}^\star.$$

30 Bonus: Logarithms of Motions Behave Linearly

Representing versors by bivectors

$$B = \log(V),$$

permits geometric data processing, for bivectors are linear and can therefore be averaged, interpolated, extrapolated, etc.



Logarithms for the similarity transformations (combinations of rotation, translation, scaling) are now known (see book ‘GA for CS’). The general conformal logarithm is yet to be found.

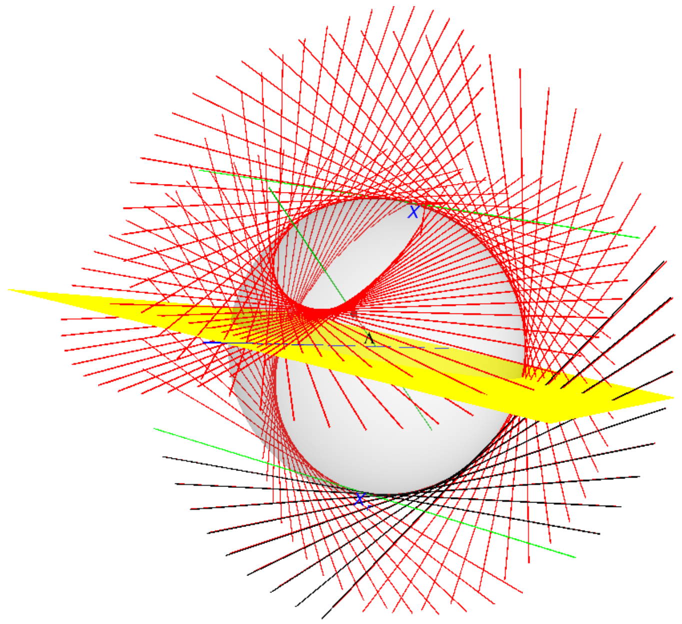
31 Geometric Calculus and Extrapolation

The elements of geometric algebra can also be differentiated with respect to each other. This allows for compact derivations of advanced results.

Perturbations are simple, and linear in B, involve commutator product (like Lie algebra):

$$e^{-\delta B/2}\, X\, e^{\delta B/2} \approx X + \tfrac{1}{2}(X\,\delta B - \delta B\, X) \equiv X + X \times \delta B.$$

First order treatment of second-order motions! Exact linearity, so apply linear data processing methods with greatly extended functionality!



Example: Yellow mirror Π rotates ϕ around Λ, where do reflected lines go?
In black: using first order bivector perturbation (i.e., second order approximation of orbit), gives rotation with versor

$$e^{-\phi\,((\Lambda\cdot\Pi)/\Pi)^*},$$

i.e. turning around the projected line, with angle $2\phi\,\cos(\Pi,\Lambda)$.

FIG(13,7)

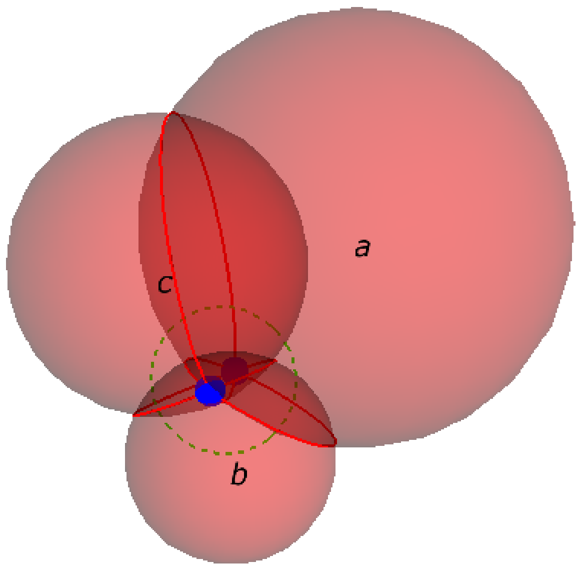
32 Trick 6: Implementation (Size Matters, But Is Not Prohibitive)

- GA can represent all 2^m subspaces of an m-D vector space as elements of computation.
- To represent Euclidean motions in $\mathbb{R}^n$ as versors, you need CGA, i.e. GA of $\mathbb{R}^{n+1,1}$-D space.
- That is a 32-dimensional representation for 3D Euclidean gometry!
- Efficient implementation is therefore an issue.
- Solved by using the strucure of GA in an automatic code generator.
  (Fontijne 2007 PhD: *Efficient Implementation of Geometric Algebra*, UvA)
- Result: high-level programming in GA available (subspace products, sandwiching).
- The actual algebraic computation takes care of the type administration; the implementation performs this at compile time. Effectively the program does hardly more than linear algebra at the lowest level, but with high-level specification using CGA products and operators.

Executive summary:

Your people can now use a high-level language (GA) to specify Euclidean geometry, at code generation level, for competitive efficiency with classical approach, but with more compact, more maintainable and less error-prone code.

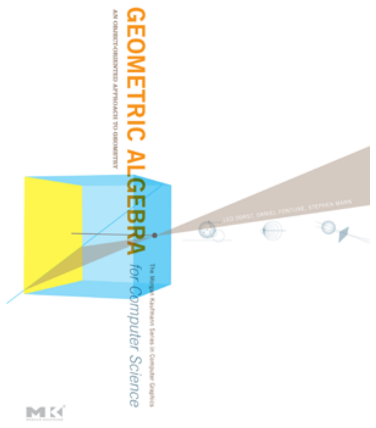
33 Summary: Euclid's Elements in the Conformal Model



FIG(15,1)

- *primitives as subspaces*:
  points, lines, planes, circles, spheres, tangents
- *constructions as subspace products*:
  connections, intersection, plunge, duality
- *motions as versors*:
  translations, rotations, reflections in planes or spheres (actually, any conformal transformation)
- *properties parametrized*:
  size, weight, location, orientation, direction
- *numerics exactly linearized*:
  linear (bivector) parametrization of motions

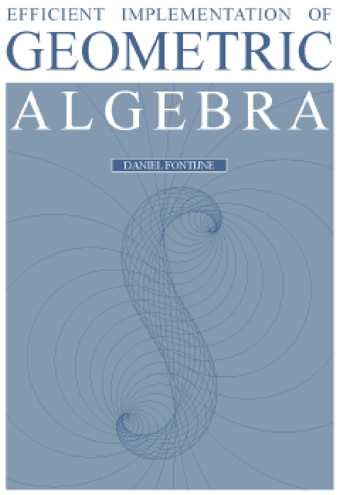
A Message from Our Sponsor: **Recommended Reading**



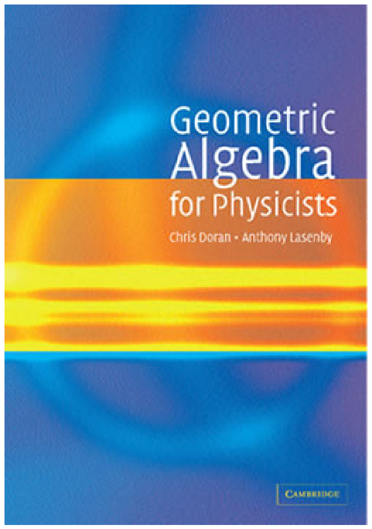
Geometric Algebra for Computer Science:
An Object-Oriented Approach to Geometry
Leo Dorst, Daniel Fontijne, Stephen Mann
(Morgan-Kaufmann Publishers 2007, ISBN 0-12-369465-5)

- 22 chapters, 4 appendices, 650 pages, 150+ full color figures, free software.
- Available everywhere, price €45-90.
- Book website, freely downloadable software and demos: `www.geometricalgebra.net`

Efficient Implementation of Geometric Algebra
Daniel Fontijne
Ph.D. thesis UvA, 2007, ISBN-13: 978-90-889-10-142,
freely available at
`www.science.uva.nl/~fontijne/phd.html`



35 Recommended Reading, Continued

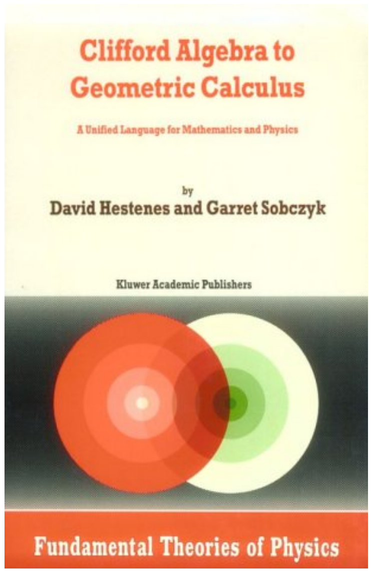


Geometric Algebra for Physicists
Chris Doran and Anthony Lasenby
(Cambridge University Press 2003)

- 14 chapters, 592 pages
- Price £80 (hardback), £40 (paperback)

Clifford Algebra to Geometric Calculus
David Hestenes and Garret Sobczyk
(Kluwer Academic Publishers 1984)

- 8 chapters, 336 pages
- Price about \$100 - \$250


Clifford Algebra to
Geometric Calculus
A Unified Language for Mathematics and Physics
by
David Hestenes and Garret Sobczyk
Kluwer Academic Publishers
Fundamental Theories of Physics

36 Appendix 1: Linear Algebra is Not Good Enough for Geometry

- *primitives*: only vectors and covectors (hyperplanes)
- *constructions*: hardly any at algebraic level, some as matrix manipulation
- *motions*: linear transformations too general; orthogonal transformations too cumbersome
- *properties*: involved non-linear parametrizations of geometric transformations
- *numerics*: strongly developed techniques for linearized estimation

The lack of algebraic constructions leads to the bad habit of *specification by coordinates*. The limited number of primitives and corresponding data structures, combined with lack of covariance, then produces *confusion and errors*.

Linear algebra is the assembly language of geometry.

Structure preservation needs to be carefully and explicitly designed and enforced.

37 Appendix 2: Geometric Algebra Is Tailored To Geometry

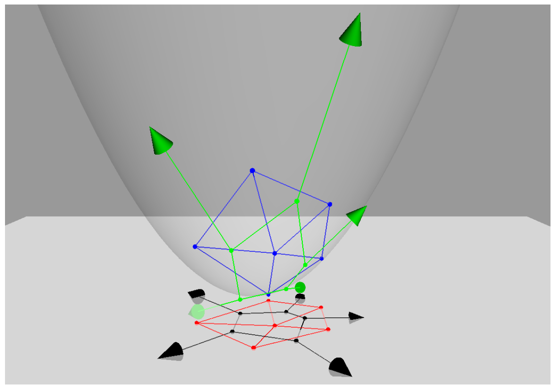
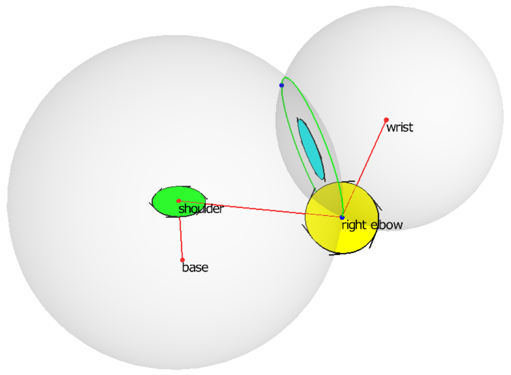
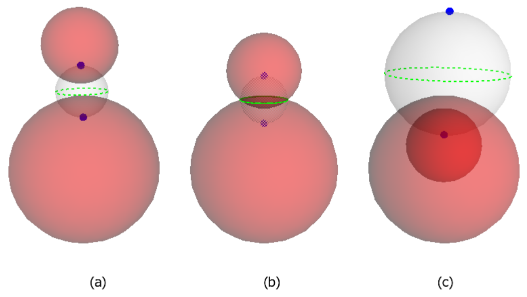
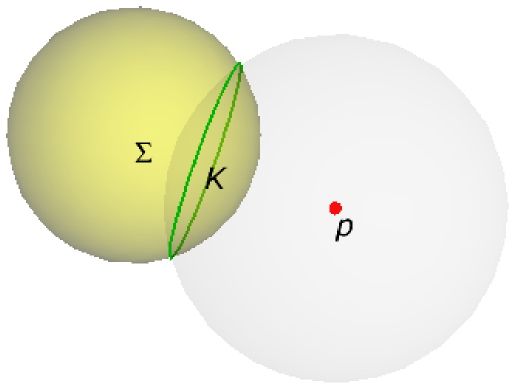
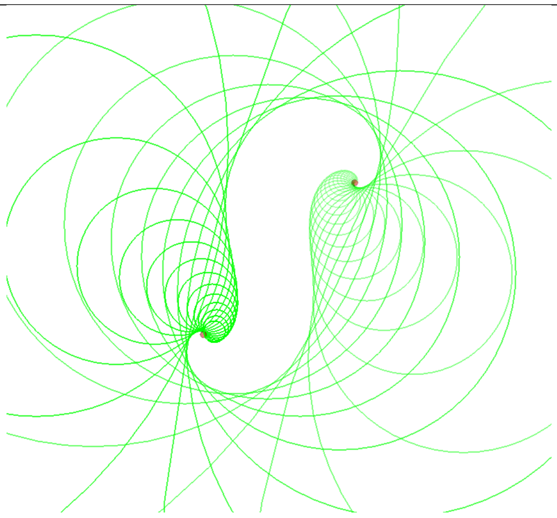
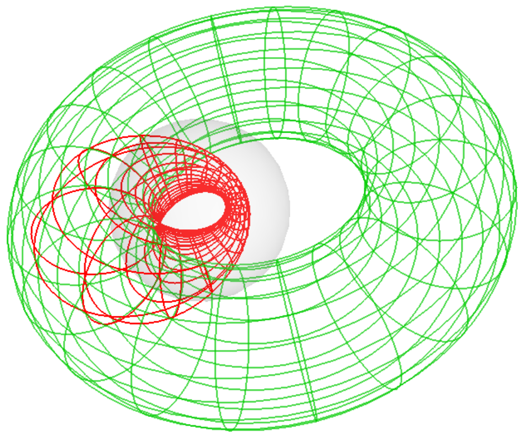
- *primitives*: general subspaces (which model points, lines, planes, spheres, tangents, etc.)
- *constructions*: algebraic spanning, intersection, orthogonality, duality
- *motions*: motions are automatically structure preserving
- *properties*: parametrization immediately in terms of geometric primitives
- *numerics*: geometric differentiation, more extended linearization, estimation (but immature)

Now everything can be specified using the geometry directly:

> *Geometric algebra is the high-level language of geometry.*

Structure preservation is automatic.

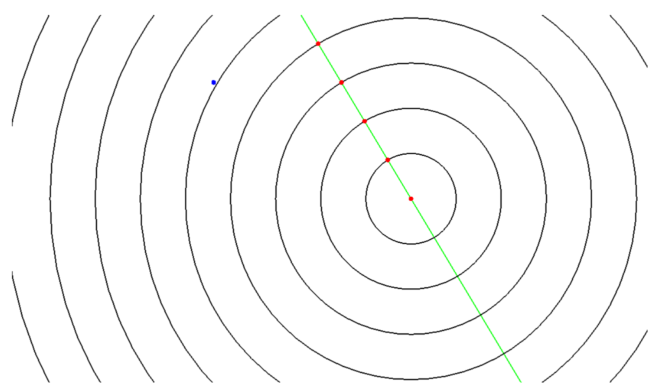
Appendix 3: Various Conformal Demos

 Voronoi diagrams FIG(14,7)	 Inverse Kinematics FIG(14,10)	 Sphere Intersection FIG(15,3)
 Contour circle FIG(15,13)	 Loxodromes FIG(16,6)	 Dupin's Cycloid FIG(16,12)

39 Appendix 4: Non-Euclidean Geometries

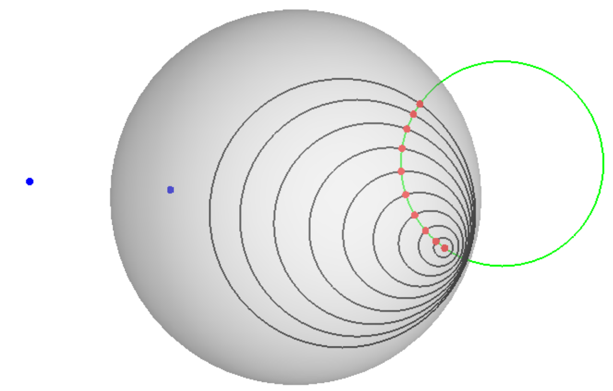
By changing the 'infinity' vector $\vec{\infty}$, we can get conformal representations of various metrics.

FIG(16,13) metrical mystery tour

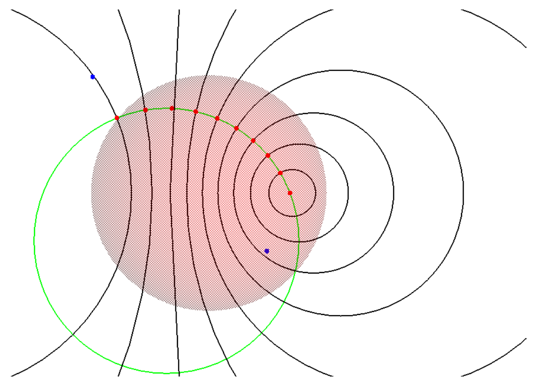


Euclidean: $\vec{\infty} = e_\infty$

FIG(16,8)

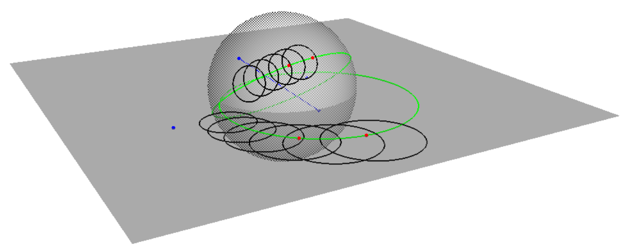


Hyperbolic: $\vec{\infty} = e_+ = e_o - e_\infty/2$



Spherical: $\vec{\infty} = e_- = e_o + e_\infty/2$

FIG(16,9)



Spherical interpretation