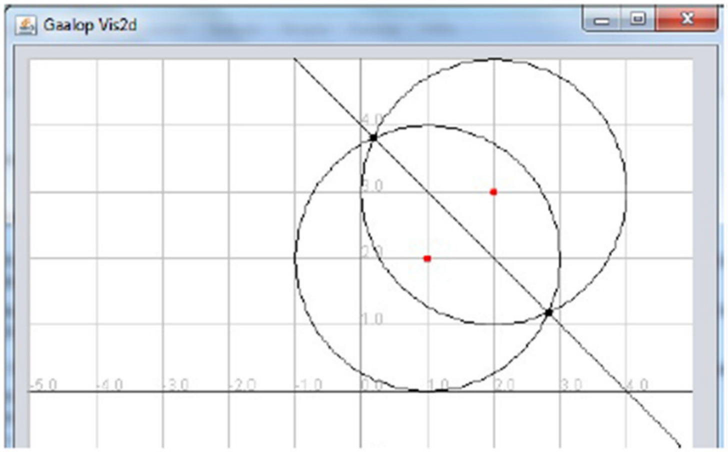
Geometric Algebra Computing
– Compass Ruler Algebra

12.02.2015

Dr.-Ing. Dietmar Hildenbrand
Technische Universität Darmstadt

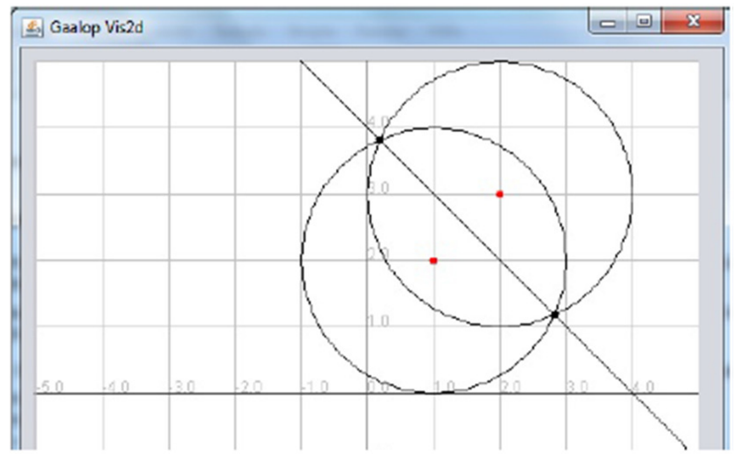


Overview

- Compass Ruler Algebra

- Visualizations with Gaalop

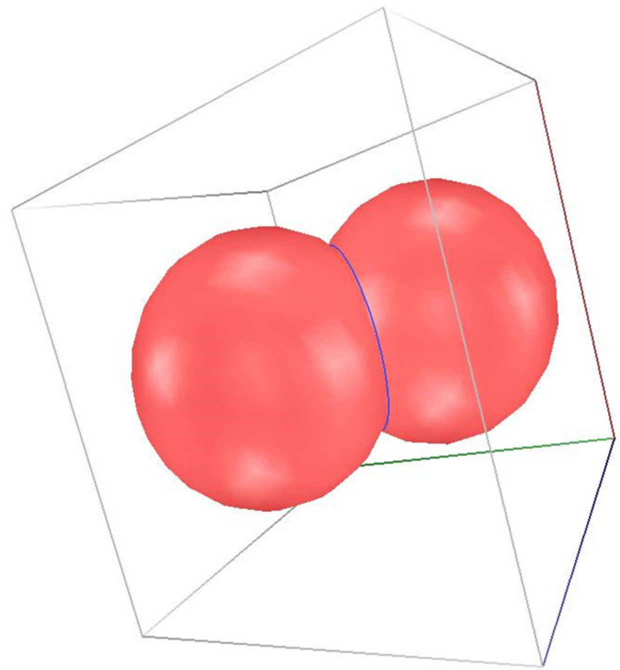
- Proofs with Gaalop



- Future Work: A foundation for the combination of Dynamic Geometry Systems with Computer Algebra Systems?

Goal of Geometric Algebra

- Mathematical language close to the geometric intuition combining geometry and algebra



Compass Ruler Algebra

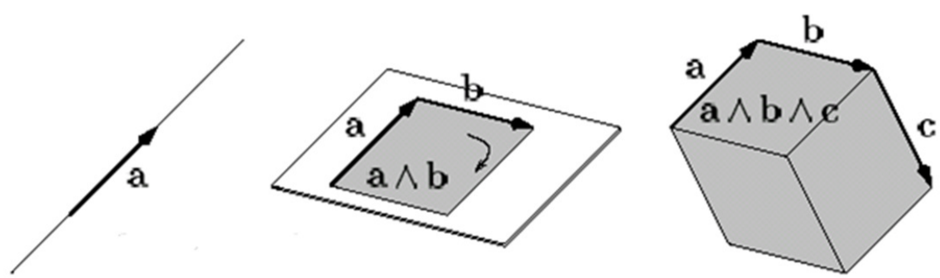
- 4 basis vectors:

  - e_1, e_2
  - e_0 : origin
  - e_∞ : point at infinity

Compass Ruler Algebra

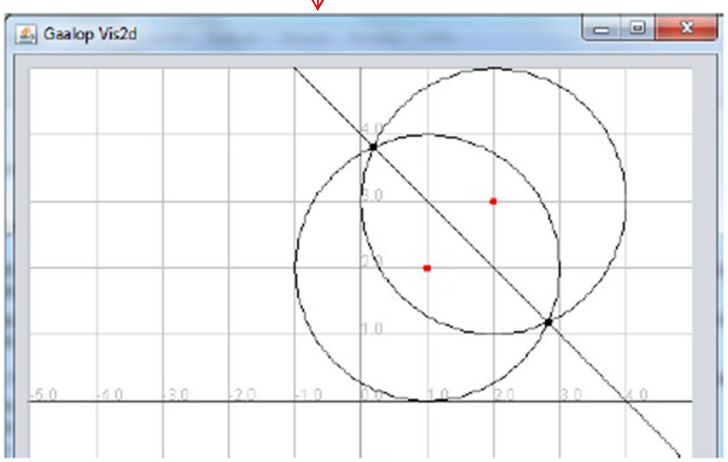
The 16 basis blades of the Compass Ruler Algebra.

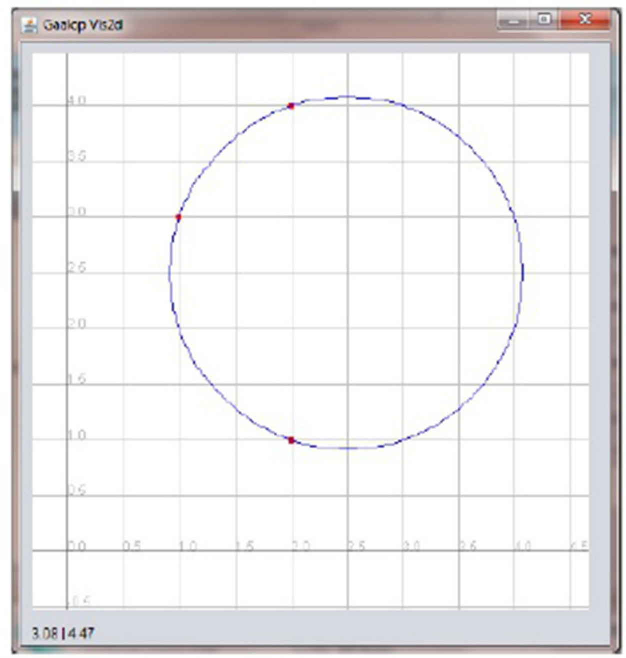
Index	Blade	Dimension
0	1	0
1	e_1	1
2	e_2	1
3	e_∞	1
4	e_0	1
5	$e_1 \wedge e_2$	2
6	$e_1 \wedge e_\infty$	2
7	$e_1 \wedge e_0$	2
8	$e_2 \wedge e_\infty$	2
9	$e_2 \wedge e_0$	2
10	$e_\infty \wedge e_0$	2
11	$e_1 \wedge e_2 \wedge e_\infty$	3
12	$e_1 \wedge e_2 \wedge e_0$	3
13	$e_1 \wedge e_\infty \wedge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



Compass Ruler Algebra

Entity	IPNS representation	OPNS representation
Point	$P = x_1 e_1 + x_2 e_2 + +\frac{1}{2}(x_1^2 + x_2^2)e_\infty + e_0$	
Circle	$C = P - \frac{1}{2}r^2 e_\infty$	$C^* = P_1 \wedge P_2 \wedge P_3$
Line	$L = \mathbf{n} + de_\infty$	$L^* = P_1 \wedge P_2 \wedge e_\infty$
Point pair	$P_p = C_1 \wedge C_2$	$P_p^* = P_1 \wedge P_2$

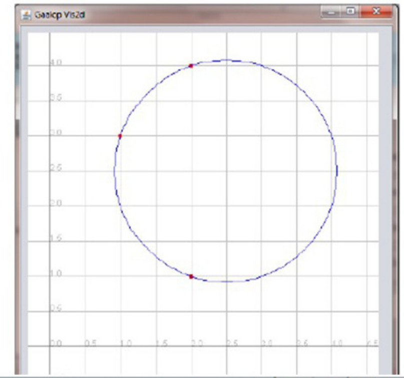


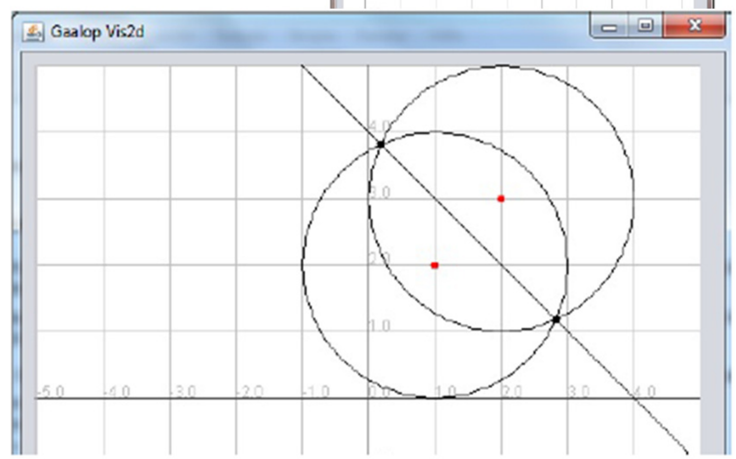


Compass Ruler Algebra

Meaning of the products:

- Outer Product
  - Generation of geometric objects
  - Intersection
- Inner Product
  - Distance Point-Point
  - Distance Point-Line
  - Angle between Line-Line
  - Distance Point-Circle
  - …
- Geometric Product
  - Rotation
  - Translation
  - Reflection
  - Inversion ($P = Ce_\infty C$ center of a circle as the inversion of infinity)
  - …





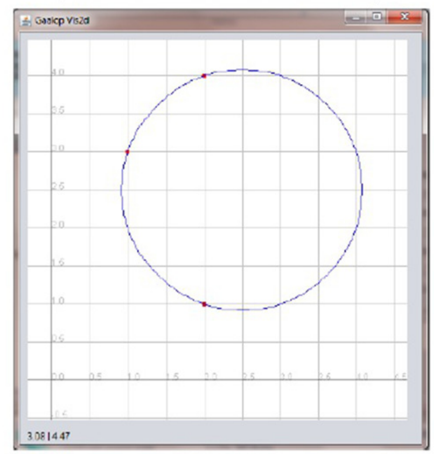
Gaalop


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

Computer Algebra

Latex, C/C++ …

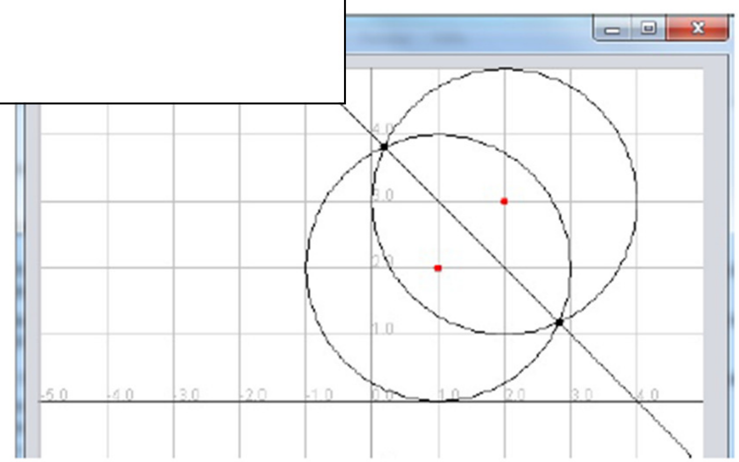
Visualization



Gaalop

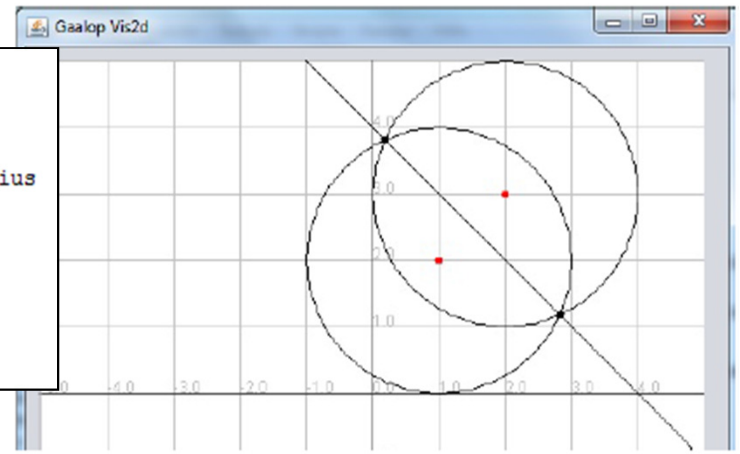

```
P1 = createPoint(x1,y1);
P2 = createPoint(x2,y2);

// 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 thru the two points of the resulting point pair
?Bisector = *(PP_dual^einf);
```

Entity	IPNS representation	OPNS representation
Point	$P = x_1e_1 + x_2e_2 + +\frac{1}{2}(x_1^2 + x_2^2)e_\infty + e_0$	
Circle	$C = P - \frac{1}{2}r^2e_\infty$	$C^* = P_1 \wedge P_2 \wedge P_3$
Line	$L = \mathbf{n} + de_\infty$	$L^* = P_1 \wedge P_2 \wedge e_\infty$
Point pair	$P_p = C_1 \wedge C_2$	$P_p^* = P_1 \wedge P_2$



Proofs with Gaalop

Proof, that the perpendicular bisector is equal to the difference of the two points



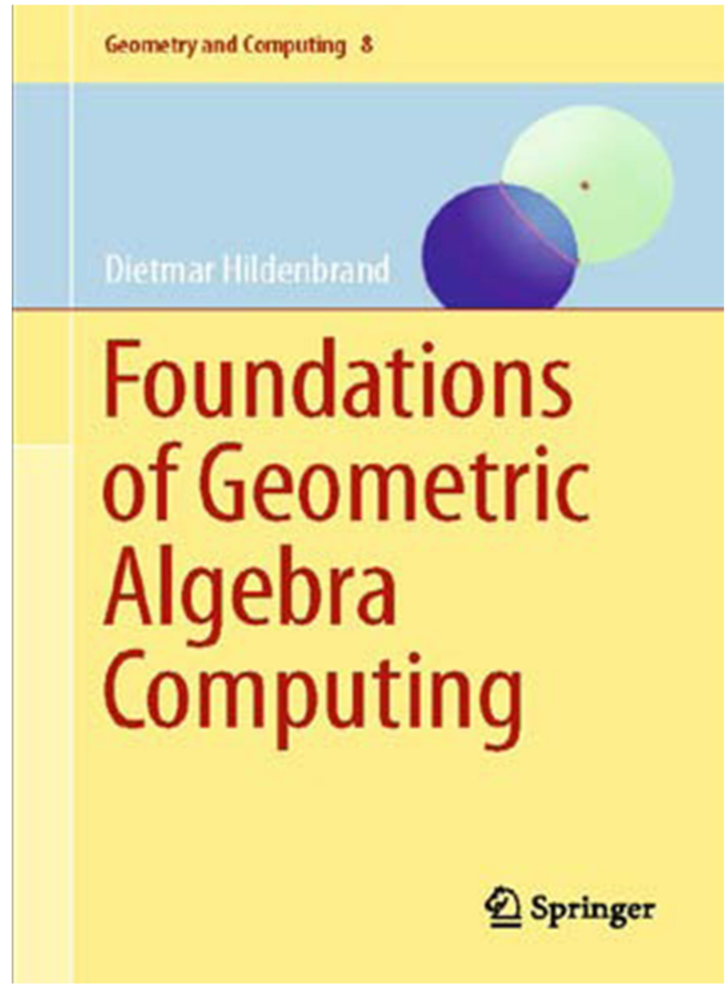

```
P1 = createPoint(x1,y1);
P2 = createPoint(x2,y2);

// 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 thru the two points of the resulting point pair
?Bisector = *(PP_dual^einf);
```

```
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
}
```

Reference

- „Foundations of Geometric Algebra Computing“
- Dietmar Hildenbrand
- Springer, 2013

- Conformal Geometric Algebra with
  - Gaalop (www.gaalop.de)

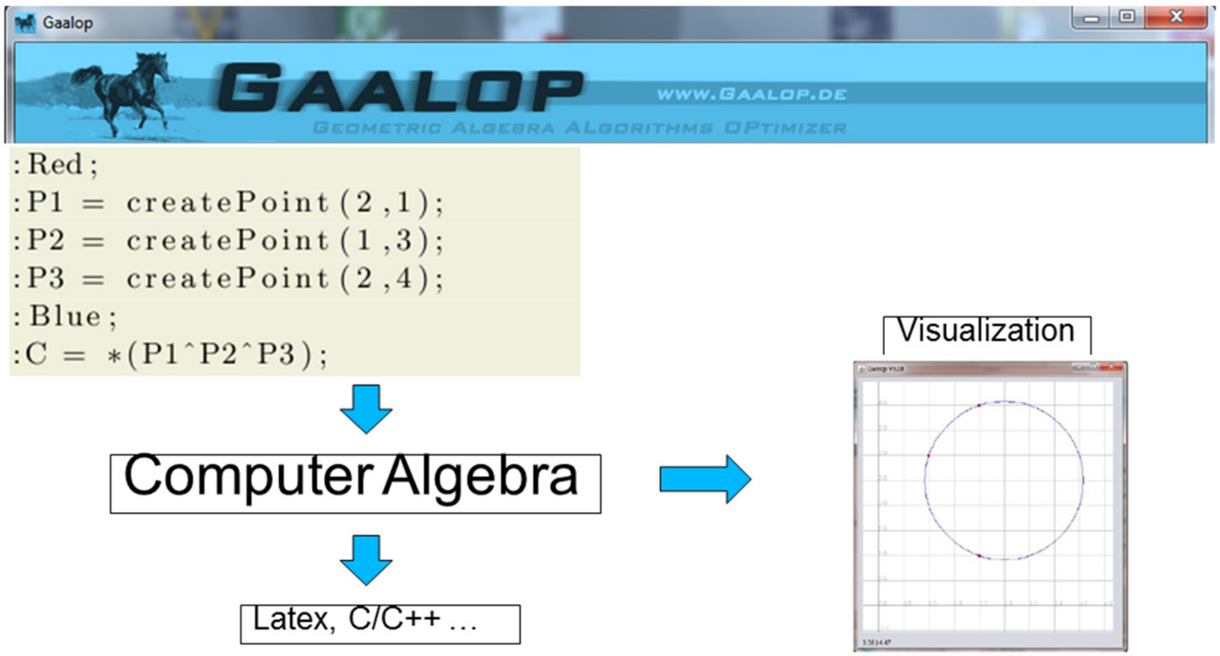


Conclusion

- Easy to visualize with Gaalop
- Easy to proove with Gaalop

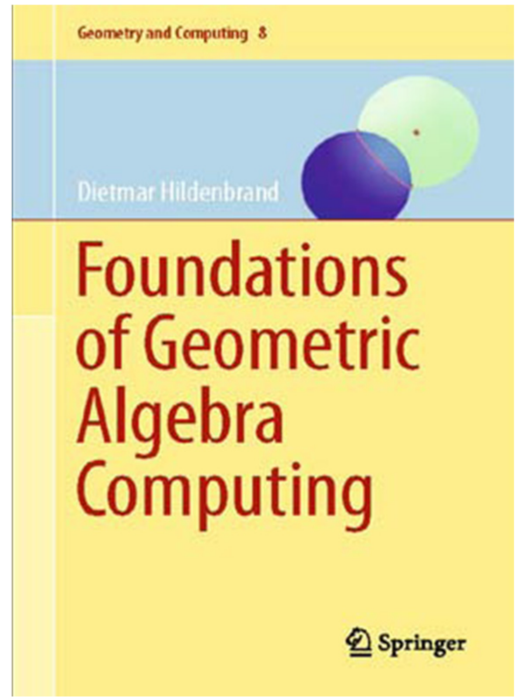
Future Work

- Compass Ruler Algebra as a foundation for the combination of Dynamic Geometry Systems with Computer Algebra Systems ?



Thanks a lot


Google
Dietmar Hildenbrand


Geometry and Computing 8
Dietmar Hildenbrand
Foundations of Geometric Algebra Computing
Springer