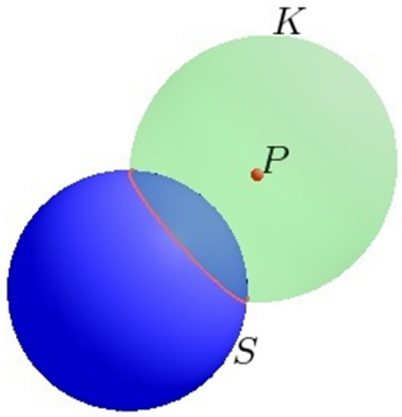
Foundations of Geometric Algebra Computing

26.10.2012

Dr.-Ing. Dietmar Hildenbrand
LOEWE Priority Program Cocoon
Technische Universität Darmstadt



Achtung Änderung!

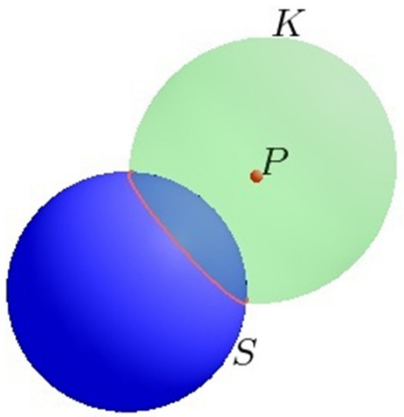
TECHNISCHE UNIVERSITÄT DARMSTADT

- Die Übung findet Montags jeweils 11:40 bis 13:10 statt !!!

Overview

TECHNISCHE UNIVERSITÄT DARMSTADT

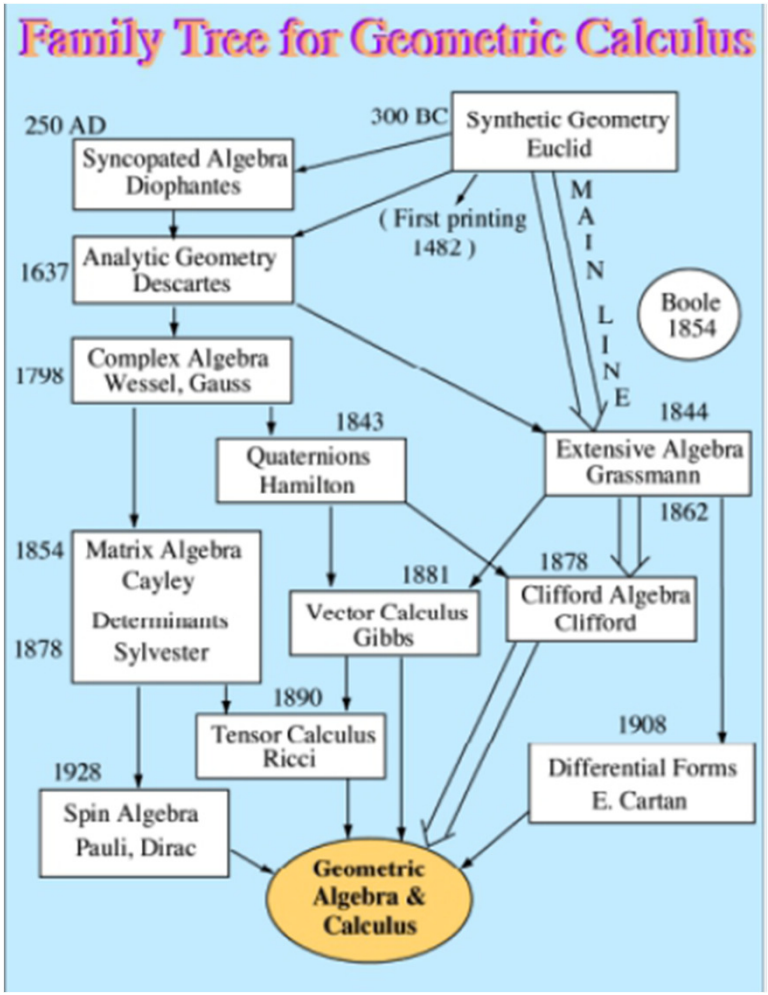
- Foundations of Geometric Algebra (GA)
- GA Applications
- One GA example
- GA Computing
  - precompiler for standard programming languages
- Molecular dynamics application



COCOON

History of Geometric Algebra&Calculus

TECHNISCHE UNIVERSITÄT DARMSTADT

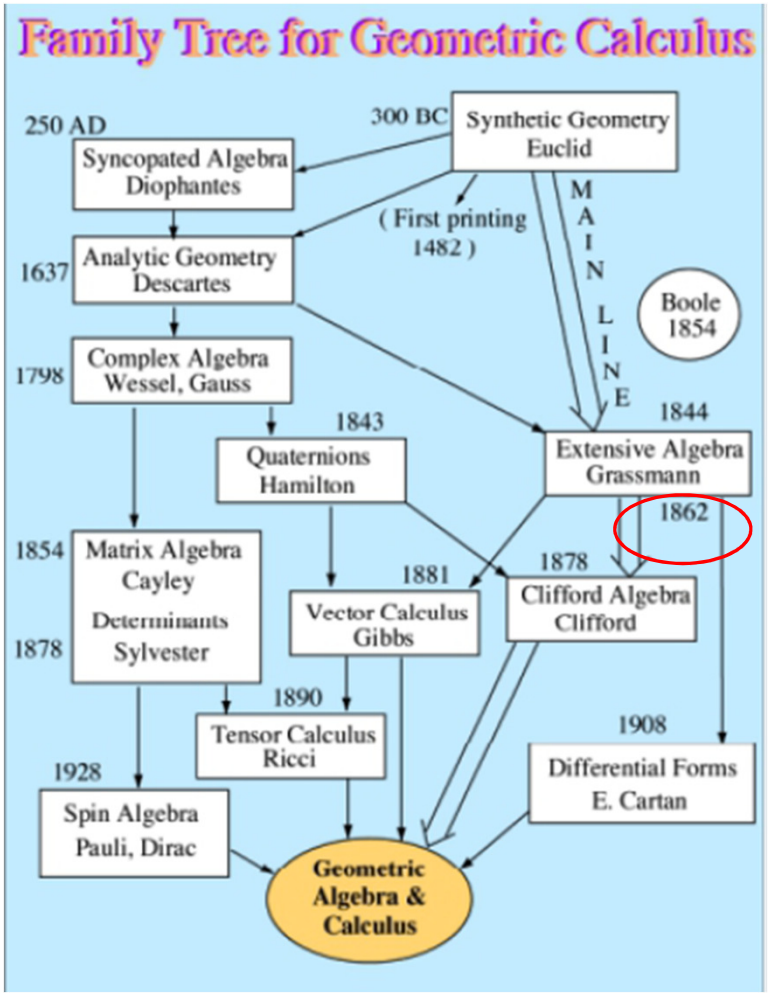


- [David Hestenes 2001]

COCOON

History of Geometric Algebra&Calculus

TECHNISCHE UNIVERSITÄT DARMSTADT

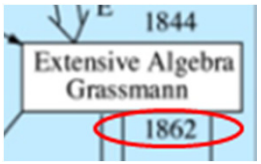


150 years

- [David Hestenes 2001]

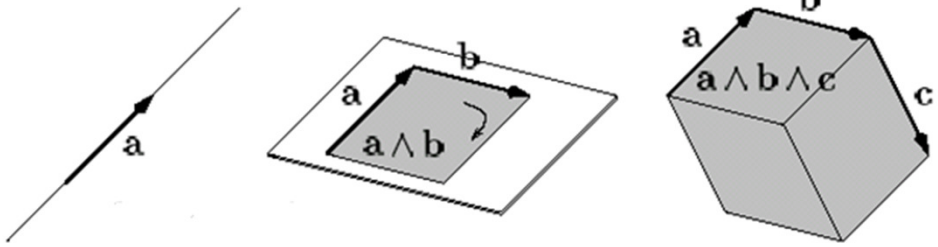
COCOON

Hermann G. Grassmann



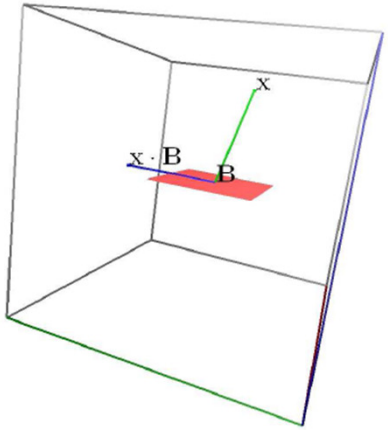
TECHNISCHE UNIVERSITÄT DARMSTADT

- Outer Product



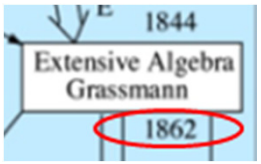
vector bivector trivector …

- Inner Product



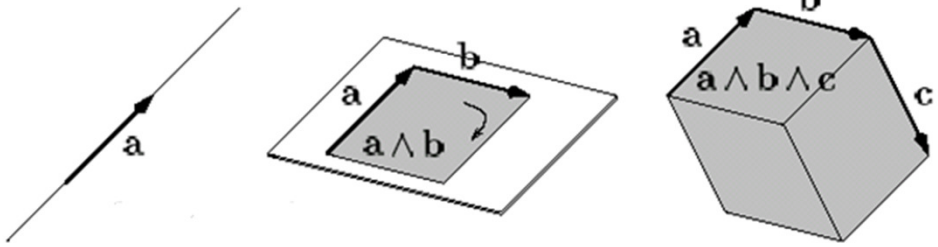
COCOON

Hermann G. Grassmann



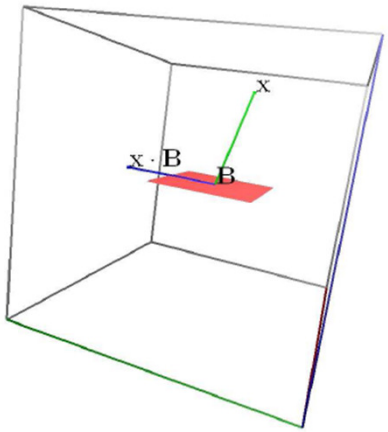
TECHNISCHE UNIVERSITÄT DARMSTADT

- Outer Product



vector bivector trivector …

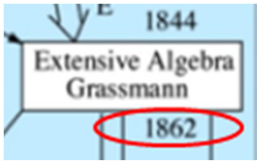
- Inner Product



cross product and scalar product are special cases of these general products

COCOON

Hermann G. Grassmann



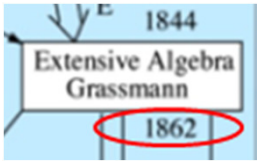
TECHNISCHE UNIVERSITÄT DARMSTADT

- Preface of „*Ausdehnungslehre*“ (Extensive Algebra) of 1862:

"I remain completely confident that the labor I have expended on the science presented here and which has demanded a signicant part of my life, as well as the most strenuous application of my powers, will not be lost. It is true that I am aware that the form which I have given the science is imperfect and must be imperfect. But I know and feel obliged to state (though I run the risk of seeming arrogant) that even if this work should again remain unused for another seventeen years or even longer, without entering into the actual development of science, still that time will come when it will be brought forth from the dust of oblivion and when ideas now dormant will bring forth fruit."
and he went on to say:
"there will come a time when these ideas, perhaps in a new form, will arise anew and will enter into a living communication with contemporary developments."

COCOON

Hermann G. Grassmann



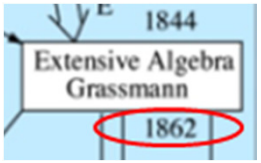
TECHNISCHE UNIVERSITÄT DARMSTADT

- Preface of „*Ausdehnungslehre*“ (Extensive Algebra) of 1862:

"I remain completely confident that the labor I have expended on the science presented here and which has demanded a signicant part of my life, as well as the most strenuous application of my powers, will not be lost. It is true that I am aware that the form which I have given the science is imperfect and must be imperfect. But I know and feel obliged to state (though I run the risk of seeming arrogant) that even if this work should again remain unused for another seventeen years or even longer, without entering into the actual development of science, still that time will come when it will be brought forth from the dust of oblivion and when ideas now dormant will bring forth fruit."
and he went on to say:
"there will come a time when these ideas, perhaps in a new form, will arise anew and will enter into a living communication with contemporary developments."

COCOON

Hermann G. Grassmann



TECHNISCHE UNIVERSITÄT DARMSTADT

- Preface of „*Ausdehnungslehre*“ (Extensive Algebra) of 1862:

"I remain completely confident that the labor I have expended on the science presented here and which has demanded a signicant part of my life, as well as the most strenuous application of my powers, will not be lost. It is true that I am aware that the form which I have given the science is imperfect and must be imperfect. But I know and feel obliged to state (though I run the risk of seeming arrogant) that even if this work should again remain unused for another seventeen years or even longer, without entering into the actual development of science, still that time will come when it will be brought forth from the dust of oblivion and when ideas now dormant will bring forth fruit."
and he went on to say:
"there will come a time when these ideas, perhaps in a new form, will arise anew and will enter into a living communication with contemporary developments."

COCOON

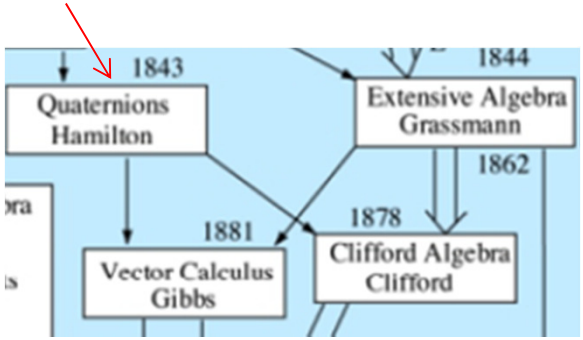
William K. Clifford

TECHNISCHE UNIVERSITÄT DARMSTADT

- Geometric Product
- For vectors

$$uv = u \wedge v + u \cdot v.$$

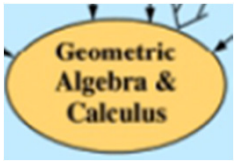
- Quaternions of Hamilton



- Normally called Clifford algebra in honor of Clifford
- He called it geometric algebra

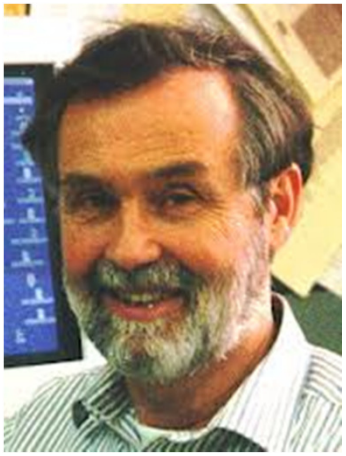
COCOON

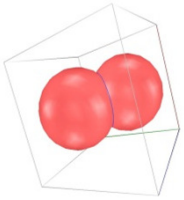
David Hestenes



TECHNISCHE UNIVERSITÄT DARMSTADT

- realized geometric algebra as a general language for physics
  („New Foundations of Classical Mechanics“, …)
- developed calculus
  ("Clifford Algebra to Geometric Calculus:
  A Unified Language for
  Mathematics and Physics")
- developed the Conformal Geometric Algebra





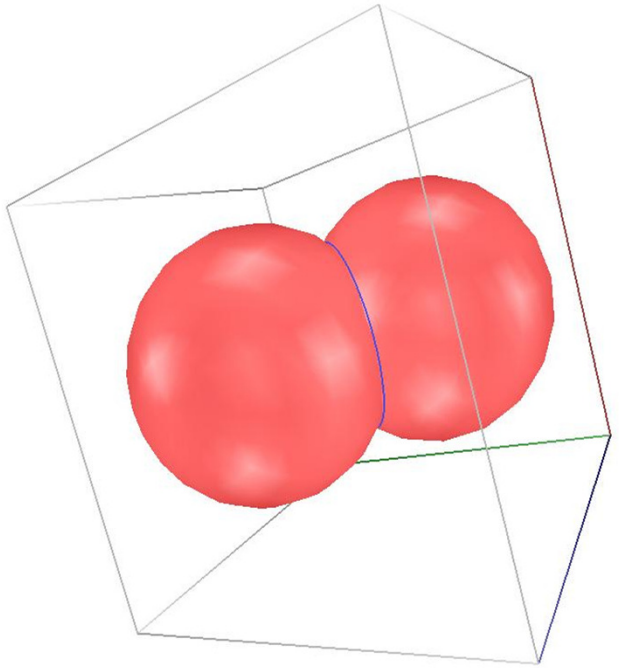
- Geometric Algebra <-> Clifford Algebra?

COCOON

Goal of geometric algebra

TECHNISCHE UNIVERSITÄT DARMSTADT

- Mathematical language close to the geometric intuition

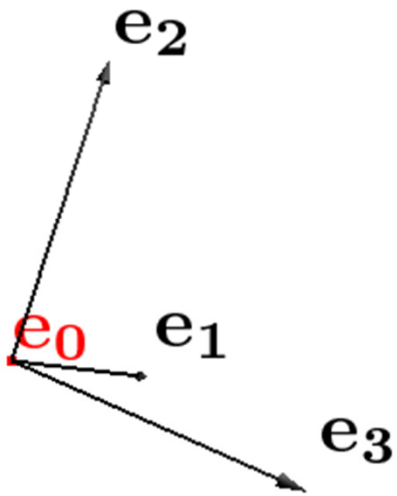


Conformal Geometric Algebra (CGA)

TECHNISCHE UNIVERSITÄT DARMSTADT

- 5 basis vectors:

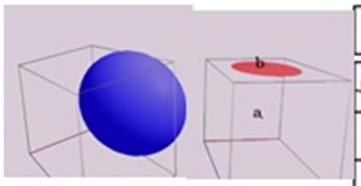
  - e_1, e_2, e_3
  - e_0 : origin
  - e_∞ : point at infinity

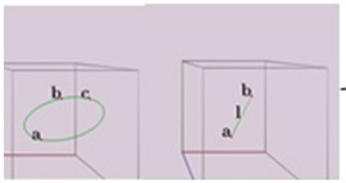


COCOON

Basis elements of CGA

TECHNISCHE UNIVERSITÄT DARMSTADT





Grade	Term	Blades	No.
0	Scalar	1	1
1	Vector	$e_1, e_2, e_3, e_\infty, e_0$	5
2	Bivector	$e_1 \wedge e_2$, $e_1 \wedge e_3$, $e_1 \wedge e_\infty$, $e_1 \wedge e_0$, $e_2 \wedge e_3$, $e_2 \wedge e_\infty$, $e_2 \wedge e_0$, $e_3 \wedge e_\infty$, $e_3 \wedge e_0$, $e_\infty \wedge e_0$	10
3	Trivector	$e_1 \wedge e_2 \wedge e_3$, $e_1 \wedge e_2 \wedge e_\infty$, $e_1 \wedge e_2 \wedge e_0$, $e_1 \wedge e_3 \wedge e_\infty$, $e_1 \wedge e_3 \wedge e_0$, $e_1 \wedge e_\infty \wedge e_0$, $e_2 \wedge e_3 \wedge e_\infty$, $e_2 \wedge e_3 \wedge e_0$, $e_2 \wedge e_\infty \wedge e_0$, $e_3 \wedge e_\infty \wedge e_0$	10
4	Quadvector	$e_1 \wedge e_2 \wedge e_3 \wedge e_\infty$, $e_1 \wedge e_2 \wedge e_3 \wedge e_0$, $e_1 \wedge e_2 \wedge e_\infty \wedge e_0$, $e_1 \wedge e_3 \wedge e_\infty \wedge e_0$, $e_2 \wedge e_3 \wedge e_\infty \wedge e_0$	5
5	Pseudoscalar	$e_1 \wedge e_2 \wedge e_3 \wedge e_\infty \wedge e_0$	1

i, j, k

$$\begin{pmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

...

COCOON

Plane and sphere as vector expression

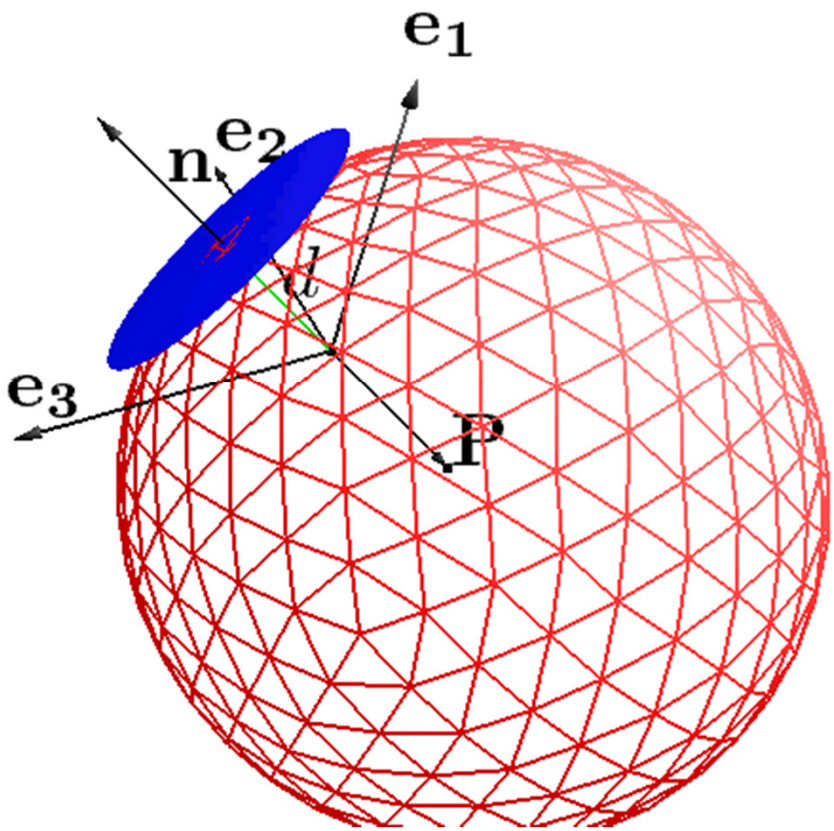
TECHNISCHE UNIVERSITÄT DARMSTADT

- **sphere**

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

- **plane**

$$\pi = \mathbf{n} + de_\infty$$

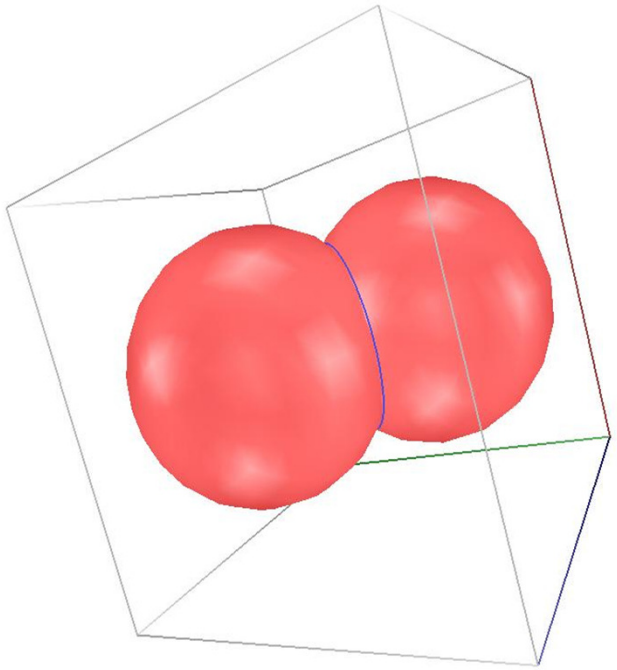


COCOON

Intersection of geometric objects

TECHNISCHE UNIVERSITÄT DARMSTADT

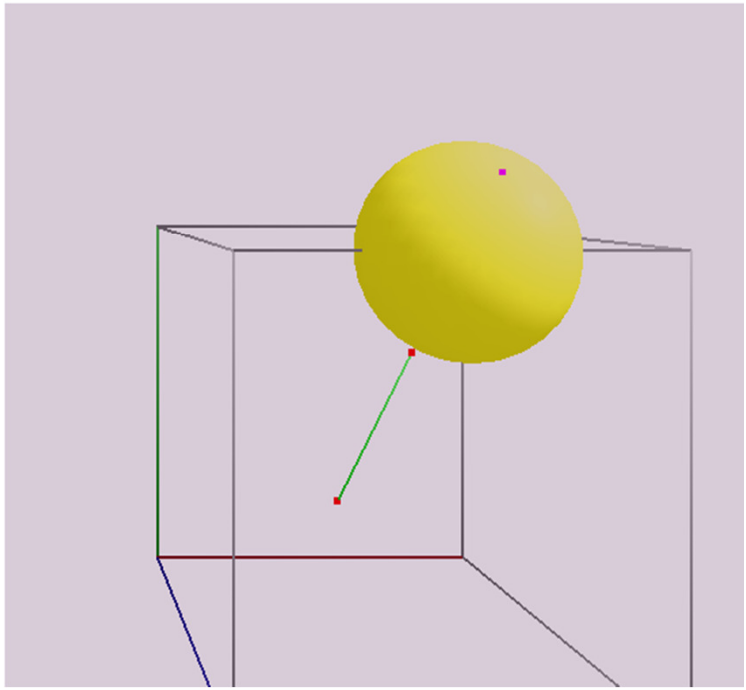
- a,b are sphere
- $a \wedge b$
  - describes the intersection of the spheres
  - represents a circle



Intersection of sphere and line

TECHNISCHE UNIVERSITÄT DARMSTADT

- $Pp = s_1 \wedge l$

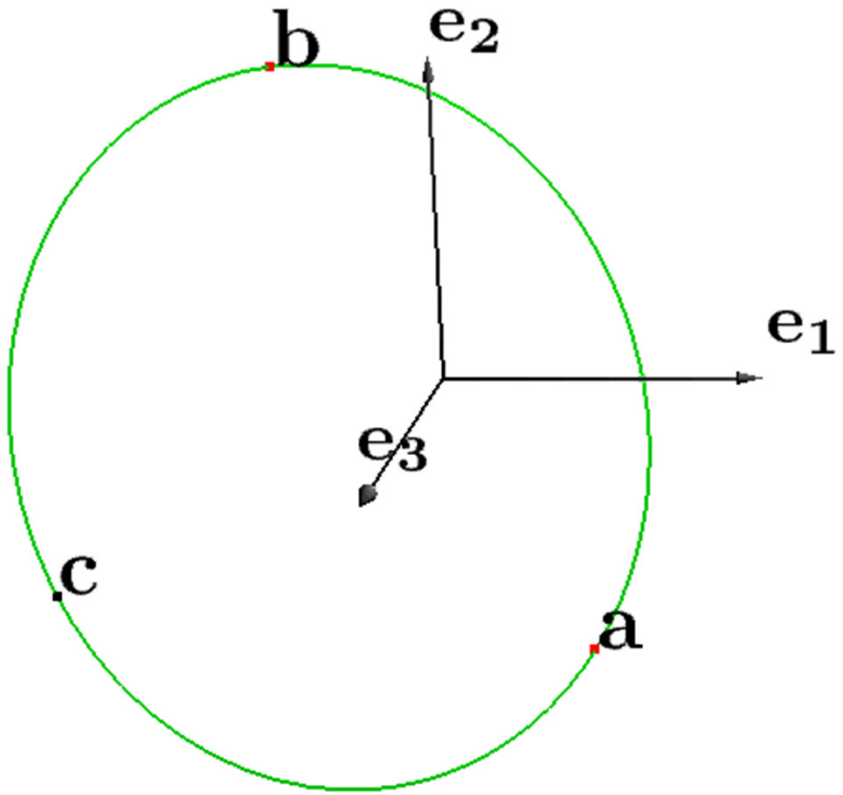


Circle and line

TECHNISCHE UNIVERSITÄT DARMSTADT

- Circle (a^b^c)
- Circle -> Line

($c \longrightarrow e_\infty$)



COCOON

Geometric Operations

TECHNISCHE UNIVERSITÄT DARMSTADT

- Rotation about L (through the origin)
- **Rotor**

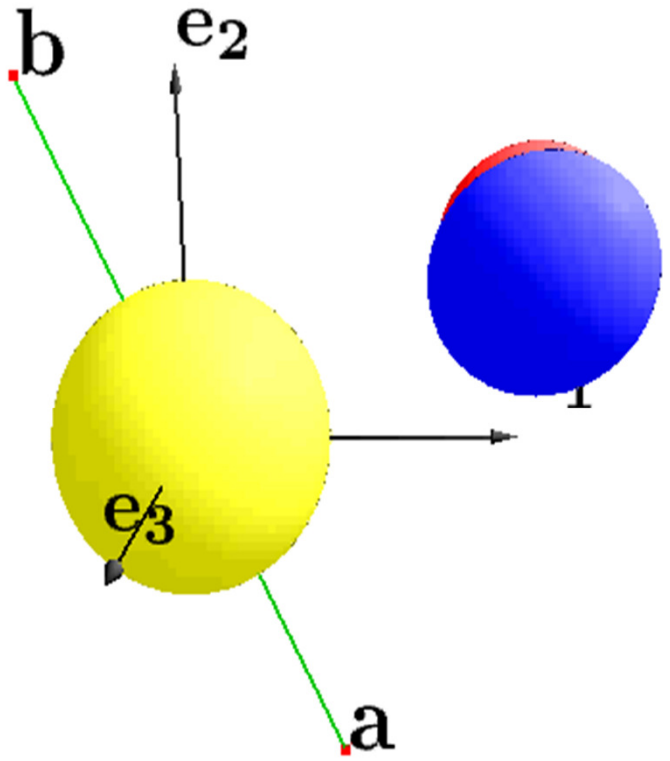
$$e^{-\frac{\phi}{2}\mathbf{l}}$$

= Quaternion

$$Q = \cos(\frac{\phi}{2}) + L\sin(\frac{\phi}{2})$$

- Note.: the line can be arbitrary

-> Rotation about an arbitrary axis(dual quaternion)



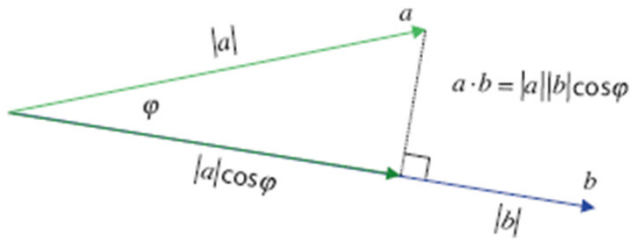
COCOON

The inner product

TECHNISCHE UNIVERSITÄT DARMSTADT

$$a \cdot b$$

- a,b 3D vectors
  -> scalar product

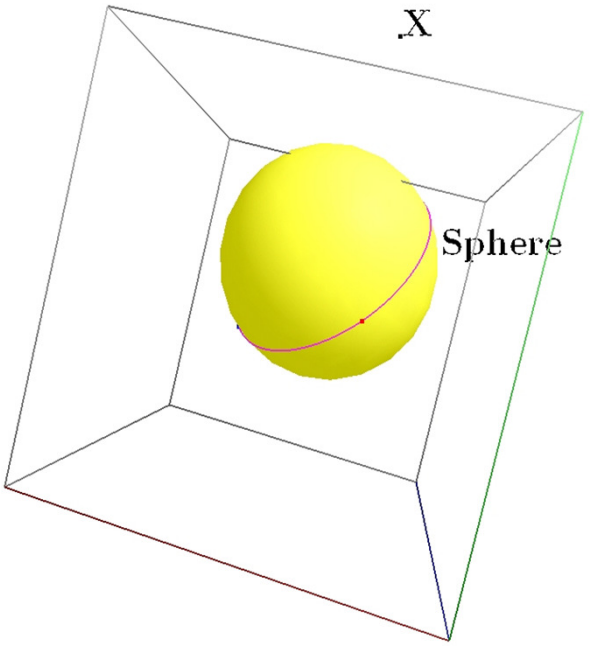


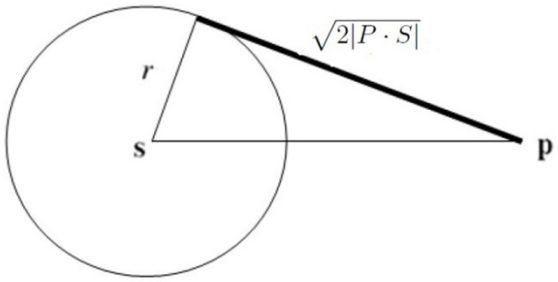
Distances

$a \cdot b$

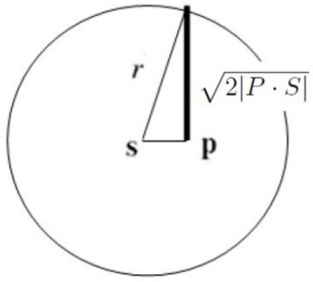
One formula for

- distance of two points
- distance between point and plane
- is a point inside or outside of a sphere? $a^2 = -2S \cdot P$





a)



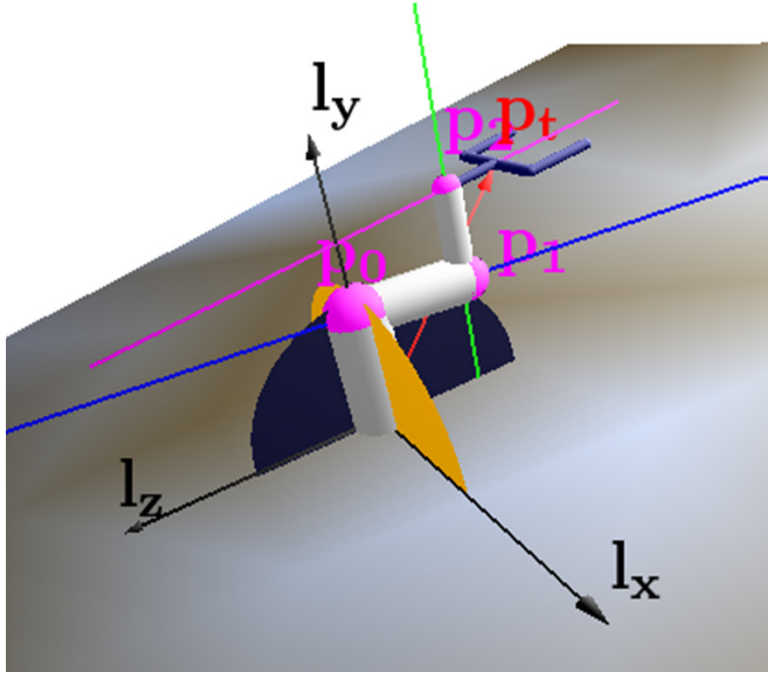
b)

Angles

$$\cos(\theta) = a \cdot b$$

One formula for the angle between

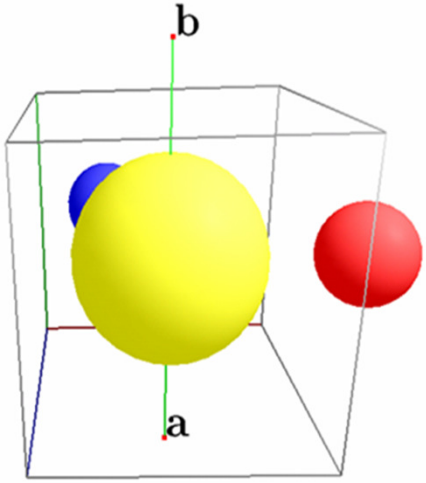
- 2 lines
- 2 planes
- 2 circles
- …



Properties of Conformal Geometric Algebra

TECHNISCHE UNIVERSITÄT DARMSTADT

- Easy calculations with geometric objects and transformations
  - Geometric intuitiveness
  - Simplicity
  - Compactness
- Unification of mathematical systems
  - Complex numbers
  - Vector algebra
  - Quaternions
  - Projective geometry
  - Plücker coordinates
  - ….



COCOON

First commercial product

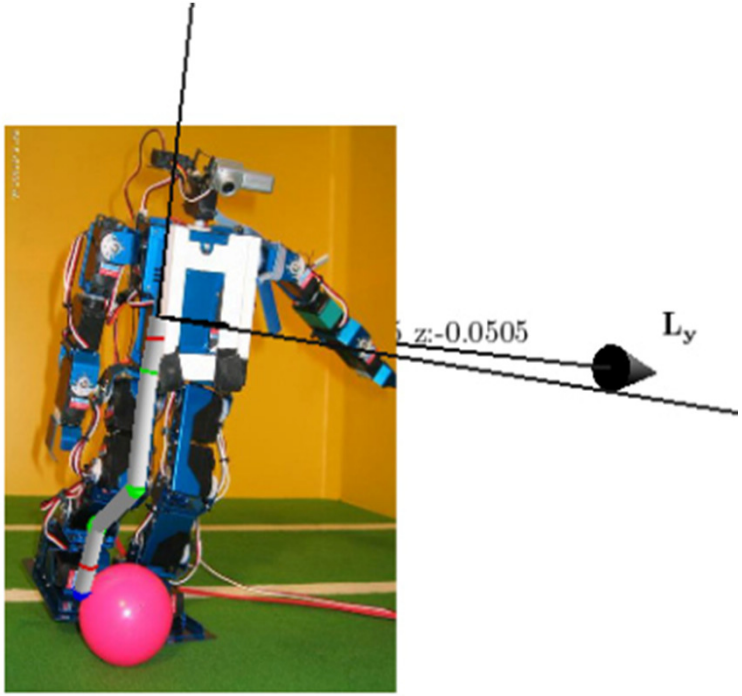
TECHNISCHE UNIVERSITÄT DARMSTADT

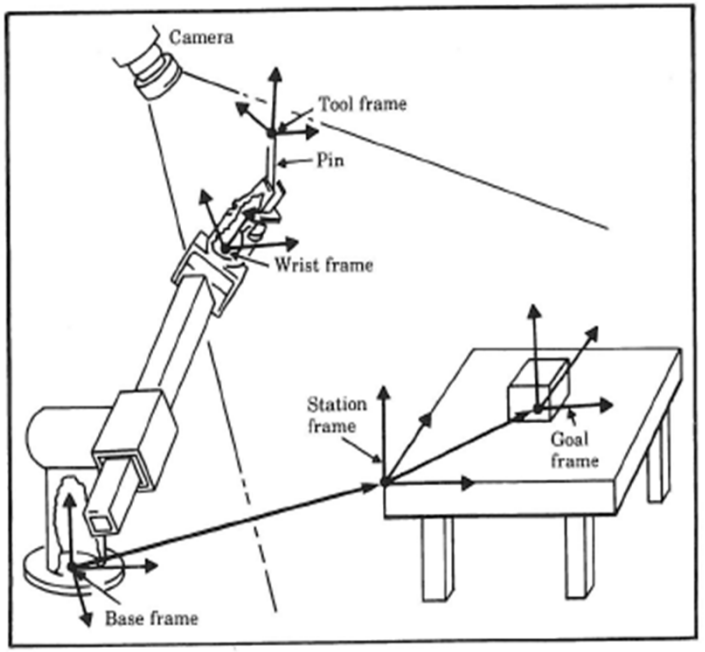
- since 2007:
  - Real time lighting engine Enlighten
  - Geomerics, Cambridge UK
  - Unreal Game Engine

Our Applications

TECHNISCHE UNIVERSITÄT DARMSTADT

- Robot kinematics



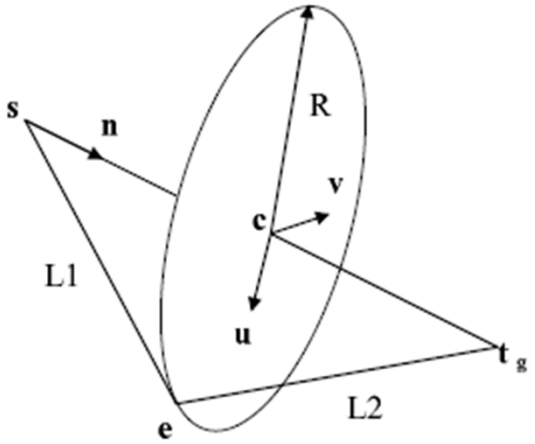


COCOON

Application

TECHNISCHE UNIVERSITÄT DARMSTADT

- Computer animation/
  Virtual reality

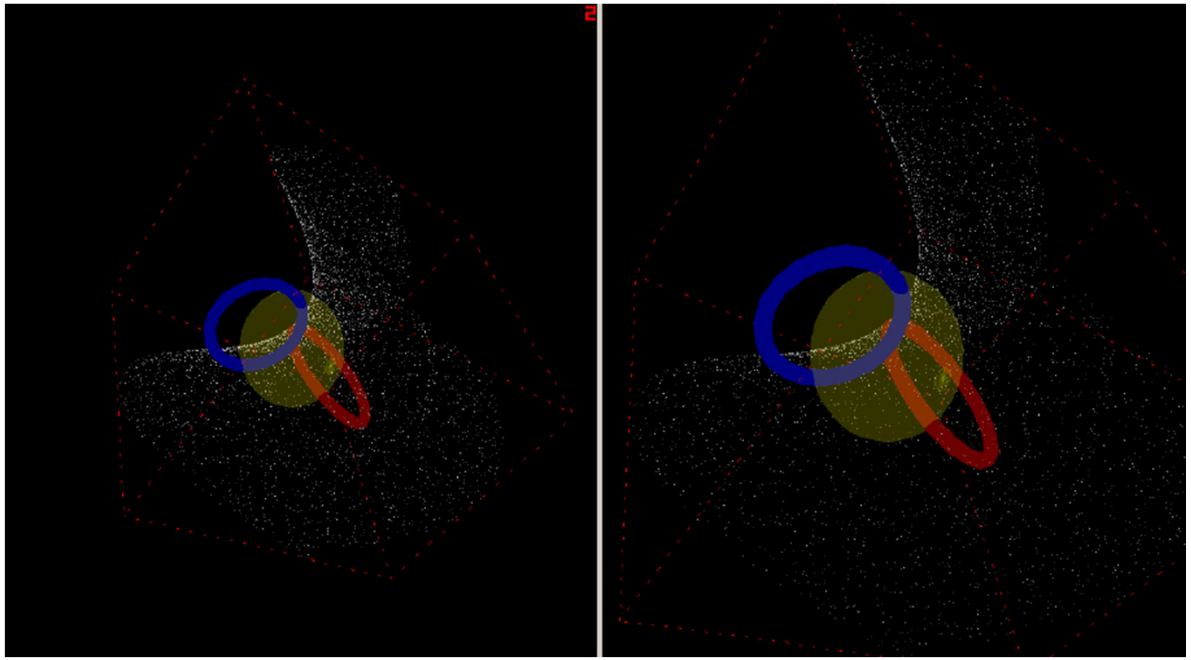


[Tolani et al. 2000]

Application

TECHNISCHE UNIVERSITÄT DARMSTADT

- Approximation of geometric objects to point clouds of laser scans



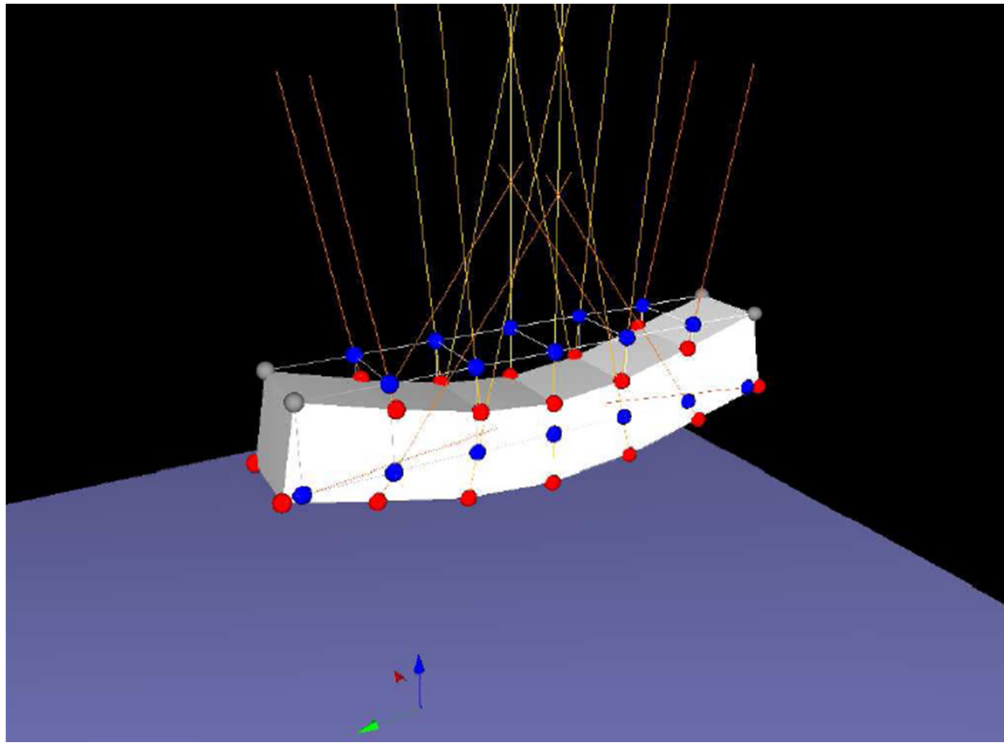
COCOON

Application

TECHNISCHE UNIVERSITÄT DARMSTADT

- Finite Element Solver
  - compact expressions for
    - velocities
    - forces
  - combining rotational and linear parts

  [Elmar Brendel et al]

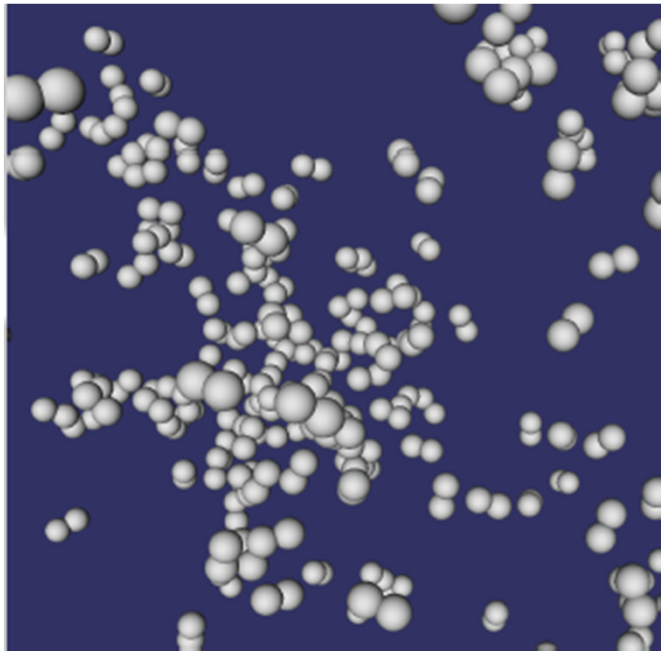


cocoon

Application

TECHNISCHE UNIVERSITÄT DARMSTADT

- Molecular dynamics simulation



Application within Cocoon

TECHNISCHE UNIVERSITÄT DARMSTADT

- EM simulation

Vector Algebra

$$\nabla \cdot \boldsymbol{E} = \rho$$
$$\nabla \times \boldsymbol{E} = -\partial_t \boldsymbol{B}$$
$$\nabla \cdot \boldsymbol{B} = 0$$
$$\nabla \times \boldsymbol{B} = \boldsymbol{J} + \partial_t \boldsymbol{E}$$

Geometric Algebra

$$\nabla \cdot \boldsymbol{E} = \rho$$
$$\nabla \wedge \boldsymbol{E} = -\partial_t (I\boldsymbol{B})$$
$$\nabla \cdot \boldsymbol{B} = 0$$
$$\nabla \wedge \boldsymbol{B} = I(\boldsymbol{J} + \partial_t \boldsymbol{E})$$

Replace:
scalar product with *inner product*
cross product with *outer product*

Geometric Algebra

$$\nabla \boldsymbol{E} = \rho - \partial_t (I\boldsymbol{B})$$
$$\nabla \boldsymbol{B} = 0 + I(\boldsymbol{J} + \partial_t \boldsymbol{E})$$
$$\Leftrightarrow$$
$$\nabla (I\boldsymbol{B}) = -(\boldsymbol{J} + \partial_t \boldsymbol{E})$$

Geometric Product:
Combine inner and outer product

$$\boldsymbol{a}\,\boldsymbol{b} = \boldsymbol{a} \cdot \boldsymbol{b} + \boldsymbol{a} \wedge \boldsymbol{b}$$

Finally combine them to a **single multivector**:

$$\nabla (\boldsymbol{E} + I\boldsymbol{B}) + \partial_t (\boldsymbol{E} + I\boldsymbol{B}) = \rho - \boldsymbol{J}$$

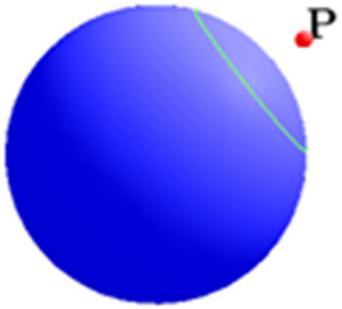
- One formula for Maxwell equations

COCOON

Horizon Example

TECHNISCHE UNIVERSITÄT DARMSTADT

- Compute the horizon circle with
  - observer *P*
  - Earth *S*

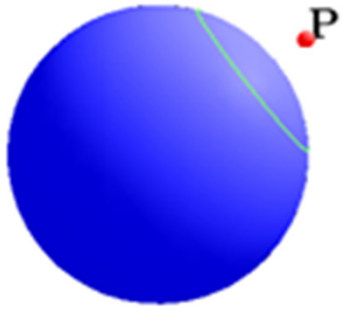


COCOON

Warum Horizont-Aufgabe?

TECHNISCHE UNIVERSITÄT DARMSTADT

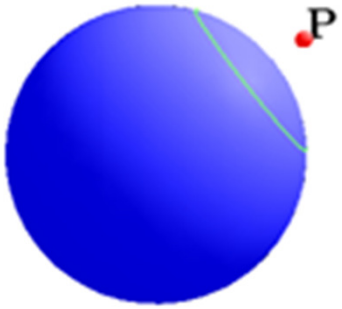
- Nachweis von
  - Nähe der algebraischen Beschreibung zur geometrischen Vorstellung
  - Einfachheit/Kompaktheit der Algorithmen



The Solution

TECHNISCHE UNIVERSITÄT DARMSTADT

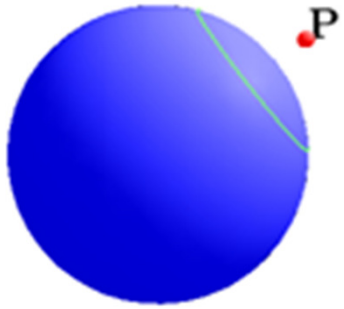
- *P*, *S*, … *C*?



The Solution

TECHNISCHE UNIVERSITÄT DARMSTADT

- Which sphere intersects with the sphere *S* at the horizon circle?

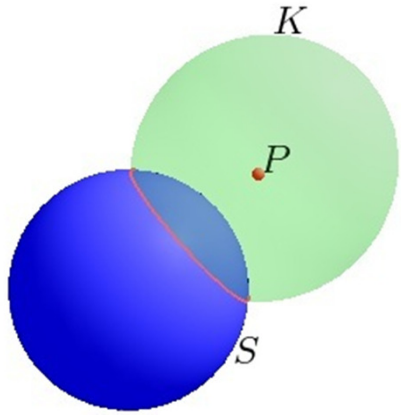


The Solution

TECHNISCHE UNIVERSITÄT DARMSTADT

- Which sphere intersects with the sphere *S* at the horizon circle?

  Sphere *K* with center *P* and radius such that it touches *S*



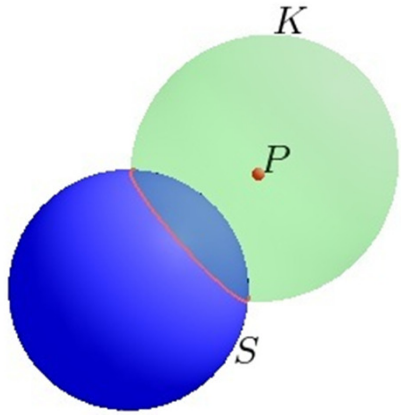
COCOON

The solution

TECHNISCHE UNIVERSITÄT DARMSTADT

- Which sphere intersects with the sphere *S* at the horizon circle?
  Sphere *K* with center *P* and radius such that it touches *S*

- The radius of *K*?



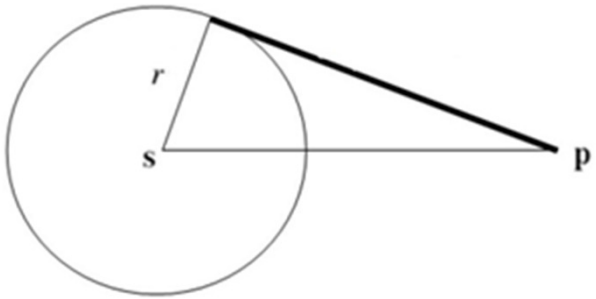
The solution

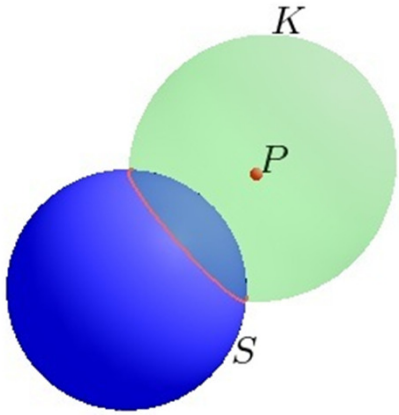
TECHNISCHE UNIVERSITÄT DARMSTADT

- Which sphere intersects with the sphere *S* at the horizon circle?
  Sphere *K* with center *P* and radius such that it touches *S*

- The radius a of *K*?

$$a^2 = -2S \cdot P$$





The solution

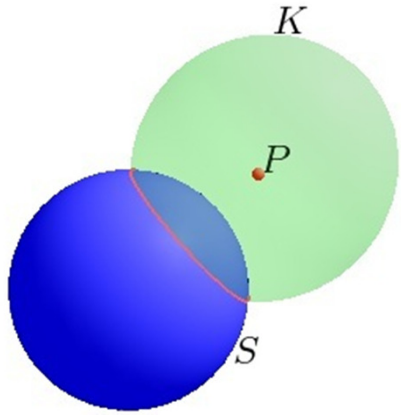
TECHNISCHE UNIVERSITÄT DARMSTADT

- Which sphere intersects with the sphere *S* at the horizon circle?
  Sphere *K* with center *P* and radius such that it touches *S*

- The radius a of *K*?

$$a^2 = -2S \cdot P$$

- How to describe the sphere *K*?



The solution

TECHNISCHE UNIVERSITÄT DARMSTADT

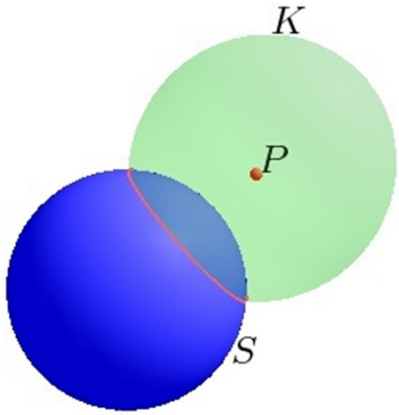
- Which sphere intersects with the sphere *S* at the horizon circle?
  Sphere *K* with center *P* and radius such that it touches *S*

- The radius a of *K*?

$$a^2 = -2S \cdot P$$

- How to describe the sphere *K*?

$$K = P - \frac{1}{2}a^2 e_\infty = P + (S \cdot P)e_\infty$$



COCOON

The solution

TECHNISCHE UNIVERSITÄT DARMSTADT

- Which sphere intersects with the sphere *S* at the horizon circle?
  Sphere *K* with center *P* and radius such that it touches *S*

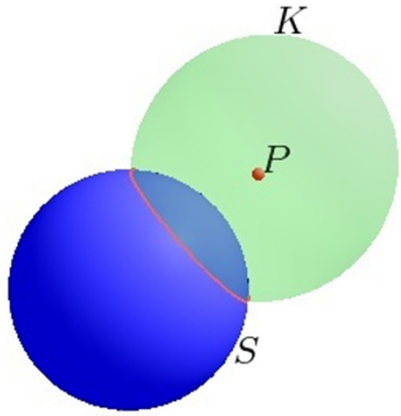
- The radius a of *K*?

$$a^2 = -2S \cdot P$$

- How to describe the sphere *K*?

$$K = P - \frac{1}{2}a^2 e_\infty = P + (S \cdot P)e_\infty$$

- The formula for the horizon circle *C*?



The solution

TECHNISCHE UNIVERSITÄT DARMSTADT

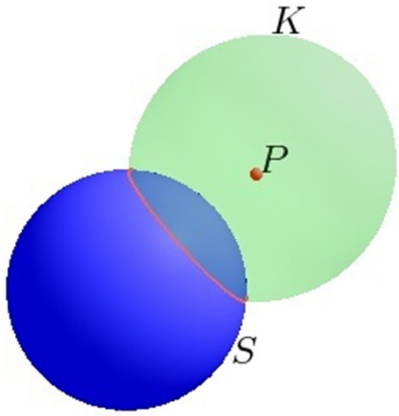
- Which sphere intersects with the sphere *S* at the horizon circle?
  Sphere *K* with center *P* and radius such that it touches *S*

- The radius a of *K*?

$$a^2 = -2S \cdot P$$

- How to describe the sphere *K*?

$$K = P - \frac{1}{2}a^2 e_\infty = P + (S \cdot P)e_\infty$$



- The formula for the horizon circle *C*?

$$C = S \wedge K$$

$$C = S \wedge (P + (S \cdot P)e_\infty).$$

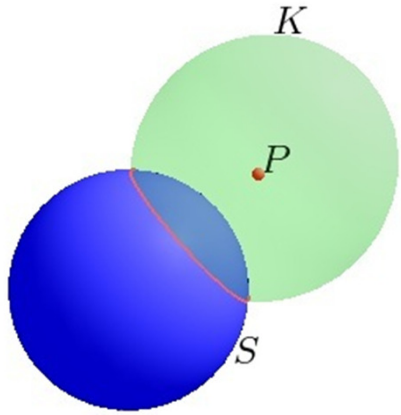
COCOON

Properties of Geometric Algebra

TECHNISCHE UNIVERSITÄT DARMSTADT

- geometrically intuitive?
- simple?
- compact?
- role of coordinates?

$$C = S \wedge (P + (S \cdot P)e_\infty).$$



How to support the widespread use of geometric algebra?

TECHNISCHE UNIVERSITÄT DARMSTADT

- What form of geometric algebra do we need today?

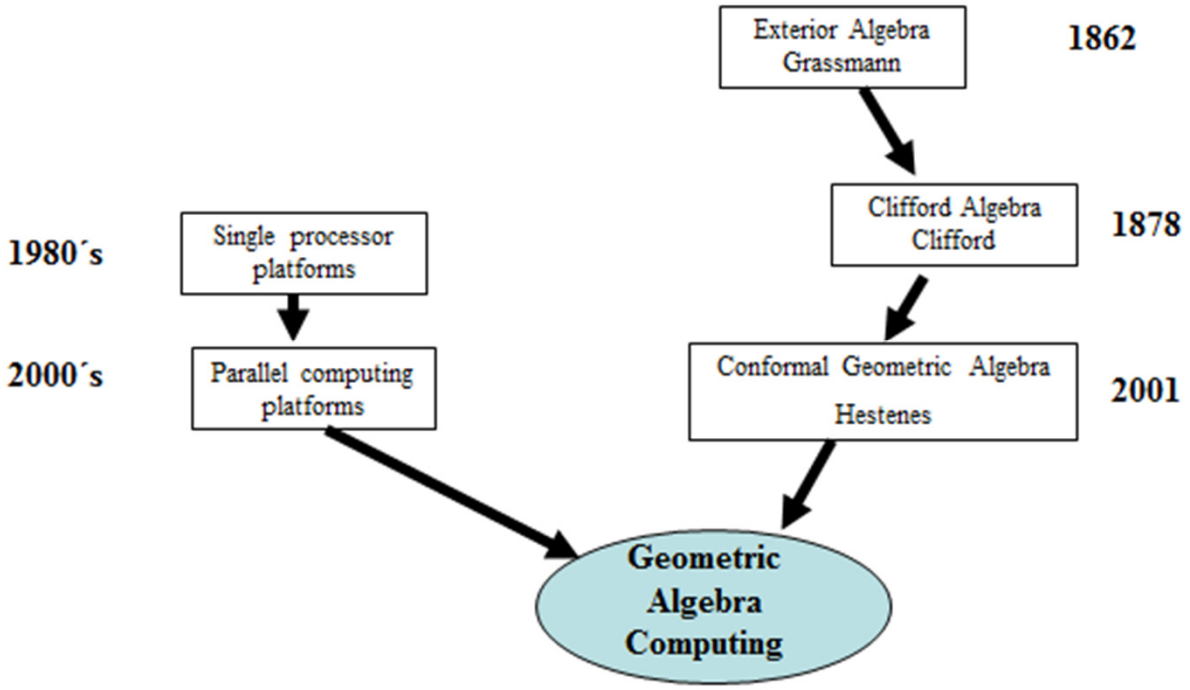
"there will come a time when these ideas, perhaps in a new form, will arise anew and will enter into a living communication with contemporary developments.“ [Grassmann 1862]

- Making it accessible for as many people as possible and their applications
- -> provide a suitable computing technology

COCOON

Geometric Algebra Computing

TECHNISCHE UNIVERSITÄT DARMSTADT

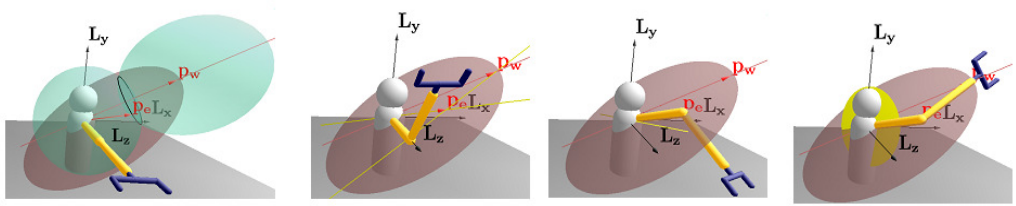


Goals of Geometric Algebra Computing

- **G**eometric **a**lgebra **al**gorithms ***op***timizer (www.gaalop.de)

- Easy development process
  - integration in standard programming languages
  - Intuitive and compact descriptions based on geometric algebra
  - leading to
    - reduction in development time
    - better maintainability
- High performance and robustness of the implementation

Proof-of-Concept Application



- Visual and interactive development
- Compact algorithms (about 30%)
- First implementation:
  - 50 times slower than conventional implementation
- Eurographics 2006:
  - Paper " Competitive runtime performance for inverse kinematics algorithms using conformal geometric algebra " by Dietmar Hildenbrand, Daniel Fontijne, Yusheng Wang, Marc Alexa and Leo Dorst
  - First geometric algebra implementation that was faster than the conventional implementation

Geometric Algebra Computing Architecture

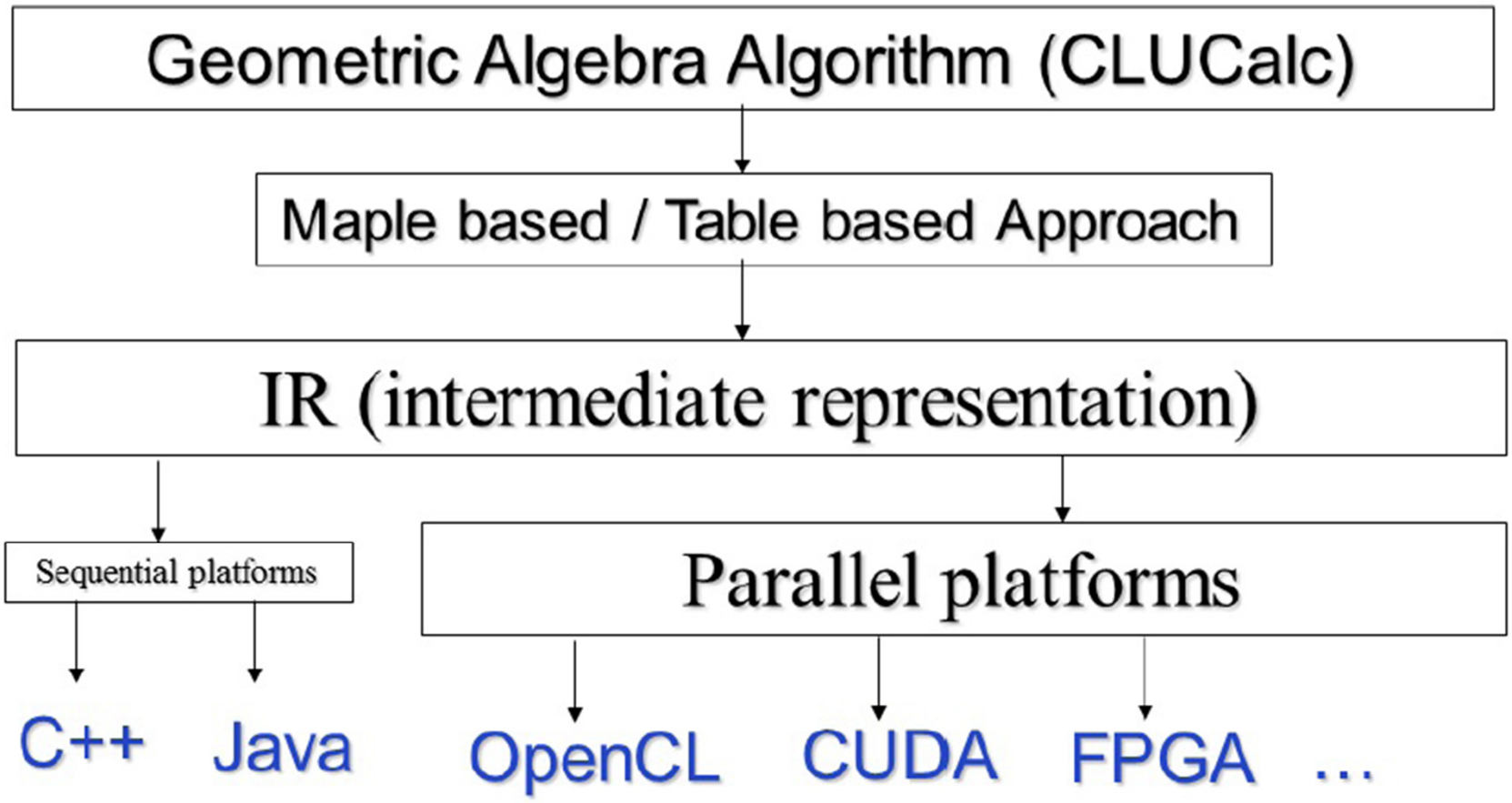


Table based Compilation Example

TECHNISCHE UNIVERSITÄT DARMSTADT

- Geometric product multiplication table in 3D GA:

		b		$\mathbf{b_1}$	$\mathbf{b_2}$	$\mathbf{b_3}$				
			E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8
a			1	e_1	e_2	e_3	e_{12}	e_{23}	e_{13}	e_{123}
	E_1	1	0	0	0	0	0	0	0	0
$\mathbf{a_1}$	E_2	e_1	0	E_1	E_5	E_7	0	0	0	0
$\mathbf{a_2}$	E_3	e_2	0	$-E_5$	E_1	E_6	0	0	0	0
$\mathbf{a_3}$	E_4	e_3	0	$-E_7$	$-E_6$	E_1	0	0	0	0
	E_5	e_{12}	0	0	0	0	0	0	0	0
	E_6	e_{23}	0	0	0	0	0	0	0	0
	E_7	e_{13}	0	0	0	0	0	0	0	0
	E_8	e_{123}	0	0	0	0	0	0	0	0

- GA algorithm:

```
a=a1*e1+a2*e2+a3*e3;
b=b1*e1+b2*e2+b3*e3;
?c=a*b;
```

resulting C code:

```
c[1]=a1*b1+a2*b2+a3*b3;
c[5]=a1*b2-a2*b1;
c[6]=a2*b3-a3*b2;
c[7]=a1*b3-a3*b1;
```


COCOON

Table based Compilation Example

TECHNISCHE UNIVERSITÄT DARMSTADT

- Geometric product multiplication table in 3D GA:

		b		$\mathbf{b_1}$	$\mathbf{b_2}$	$\mathbf{b_3}$				
			E_1	E_2	E_3	E_4	E_5	E_6	E_7	E_8
a			1	e_1	e_2	e_3	e_{12}	e_{23}	e_{13}	e_{123}
	E_1	1	0	0	0	0	0	0	0	0
$\mathbf{a_1}$	E_2	e_1	0	E_1	E_5	E_7	0	0	0	0
$\mathbf{a_2}$	E_3	e_2	0	$-E_5$	E_1	E_6	0	0	0	0
$\mathbf{a_3}$	E_4	e_3	0	$-E_7$	$-E_6$	E_1	0	0	0	0
	E_5	e_{12}	0	0	0	0	0	0	0	0
	E_6	e_{23}	0	0	0	0	0	0	0	0
	E_7	e_{13}	0	0	0	0	0	0	0	0
	E_8	e_{123}	0	0	0	0	0	0	0	0

- GA algorithm:

```
a=a1*e1+a2*e2+a3*e3;
b=b1*e1+b2*e2+b3*e3;
?c=a*b;
```

resulting C code:

```
c[1]=a1*b1+a2*b2+a3*b3;
c[5]=a1*b2-a2*b1;
c[6]=a2*b3-a3*b2;
c[7]=a1*b3-a3*b1;
```


COCOON

Horizon Example in Gaalop

TECHNISCHE UNIVERSITÄT DARMSTADT

- based on the CLUCalc language (www.clucalc.info)

```
P = VecN3(px,py,pz);          // view point
M = e0;                       // center point of earth set to origin
S = M-0.5*r*r*einf;           // sphere representing earth
K = P+(P.S)*einf;             // sphere around P
?C=S^K;                       // intersection circle
```


COCOON

Horizon Example in Gaalop

TECHNISCHE UNIVERSITÄT DARMSTADT

- 32-dim multivector array with optimized coefficient computations

```
C[8] = 0.5f * px * r * r;   // e1^einf
C[9] = -px;                 // e1^e0
C[11] = 0.5f * py * r * r;  // e2^einf
C[12] = -py;                // e2^e0
C[13] = 0.5f * pz * r * r;  // e3^einf
C[14] = -pz;                // e3^e0
C[15] = -r * r;             // einf^e0
```

- … to be further optimized from the storage point of view

COCOON

Horizon Example in Gaalop

TECHNISCHE UNIVERSITÄT DARMSTADT

- 32-dim multivector array with optimized coefficient computations

```
C[8] = 0.5f * px * r * r;   // e1^einf
C[9] = - px;                // e1^e0
C[11] = 0.5f * py * r * r;  // e2^einf
C[12] = - py;               // e2^e0
C[13] = 0.5f * pz * r * r;  // e3^einf
C[14] = - pz;               // e3^e0
C[15] = - r * r;            // einf^e0
```

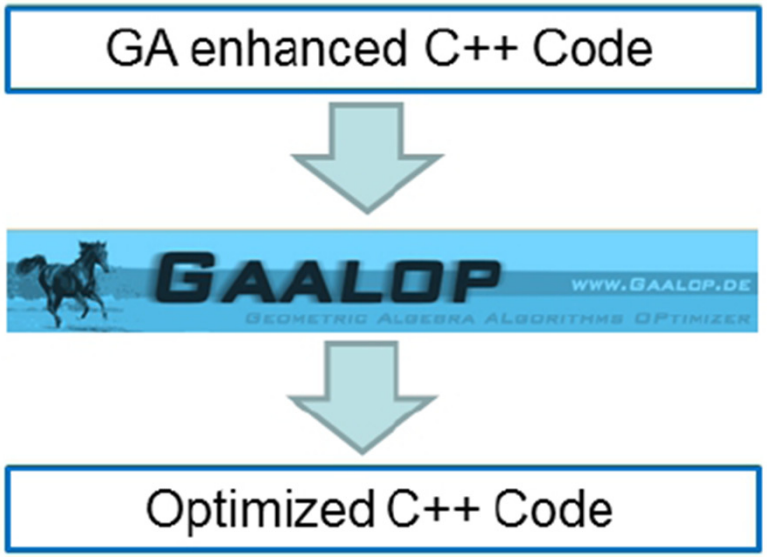
- … to be further optimized from the storage point of view

This kind of very easy arithmetic expressions lead to high runtime performance and robustness

Gaalop Precompiler

TECHNISCHE UNIVERSITÄT DARMSTADT

```
#pragma gpc begin
  ...
  Import of multivectors (if needed)
  ...
  #pragma clucalc begin
    ...
    Geometric Algebra code based on CLUCalc
    ...
  #pragma clucalc end
  ...
  Export of multivectors (if needed)
  ...
#pragma gpc end
```

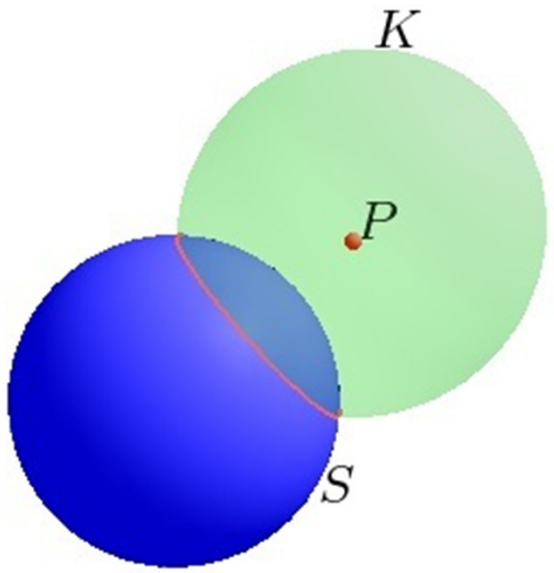


Gaalop Precompiler

TECHNISCHE UNIVERSITÄT DARMSTADT

- Visualization of the horizon example

```
void horizon() {
  #pragma clucalc begin
     :Black;
     :P = VecN3(1,1,0);
      r = 1;
     :Blue;
     :S = e0-0.5*r*r*einf;
     :Color(0,1,0,0.2);
     :K = P+(P.S)*einf;
     :Red;
     :C = S^K;
  #pragma clucalc end
}
```



Gaalop Precompiler

TECHNISCHE UNIVERSITÄT DARMSTADT

- Horizon computations in C++

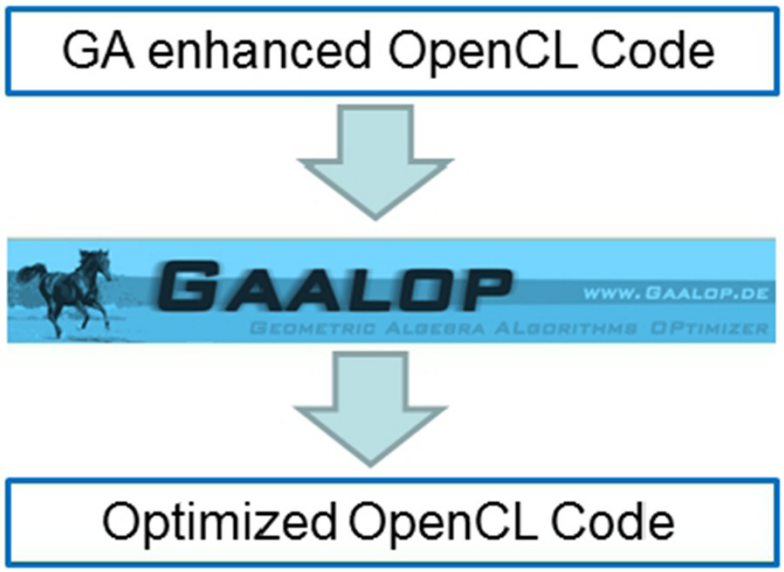
```
#include <iostream>
#include <gp.h>

int main() {
  #pragma gpc begin
    #pragma clucalc begin
      P = VecN3(1,1,0);
      r = 1;
      S = e0-0.5*r*r*einf;
      C = S^(P+(P.S)*einf);
      ?homogeneousCenter = C*einf*C;
      ?scale = -homogeneousCenter.einf;
      ?EuclideanCenter = homogeneousCenter / scale;
    #pragma clucalc end
    std::cout << mv_get_bladecoeff(homogeneousCenter,e0) << std::endl;
    std::cout << mv_get_bladecoeff(EuclideanCenter,e0) << std::endl;
    std::cout << mv_get_bladecoeff(EuclideanCenter,e1)
    << "," << mv_get_bladecoeff(EuclideanCenter,e2)
    << "," << mv_get_bladecoeff(EuclideanCenter,e3);
  #pragma gpc end
  return 0;
}
```


COCOON

Gaalop Precompiler

TECHNISCHE UNIVERSITÄT DARMSTADT

- Gaalop GPC for OpenCL



Gaalop Precompiler

TECHNISCHE UNIVERSITÄT DARMSTADT

- Horizon example
- in OpenCL

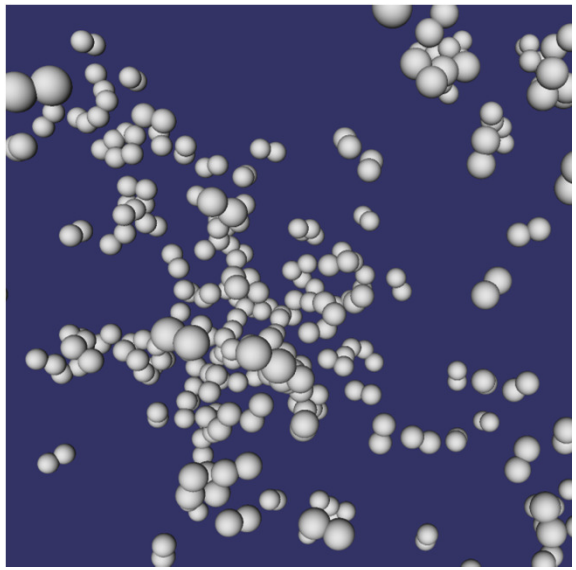
```
__kernel void horizonKernel(__global
float* circleCenters, __global const float* points,
const unsigned int num_points)
{
  const int id = get_global_id(0);
  #pragma gpc begin
    P = VecN3(points[id],
              points[id+num_points],
              points[id+2*num_points]);
    #pragma clucalc begin
      r = 1;
      S = e0-0.5*r*r*einf;
      C = S^(P+(P.S)*einf);

      ?homogeneousCenter = C*einf*C;
      ?scale = -homogeneousCenter.einf;
      ?EuclideanCenter = homogeneousCenter / scale;
    #pragma clucalc end
    circleCenters = mv_to_stridedarray(EuclideanCenter,
                                       id, num_points, e1,e2,e3);
  #pragma gpc end

}
```


Gaalop GCD for OpenCL …

TECHNISCHE UNIVERSITÄT DARMSTADT

- … available for Windows and Linux (www.gaalop.de).

- Application example: Molecular dynamics …

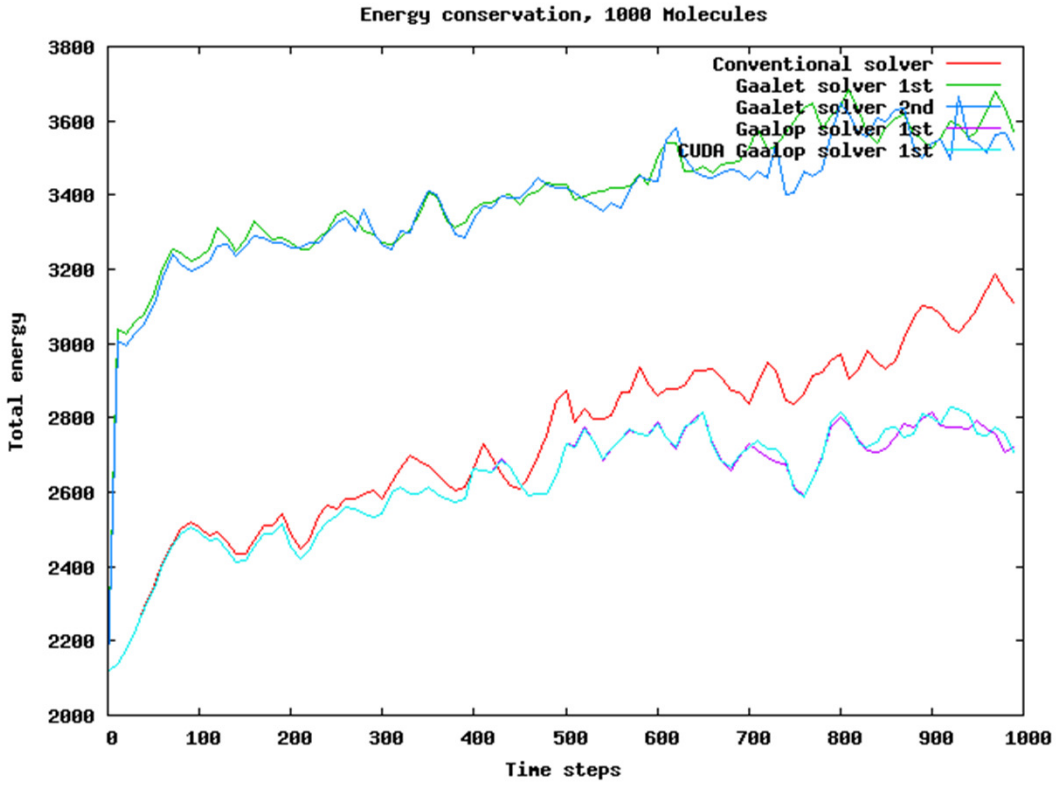


COCOON

Results

TECHNISCHE UNIVERSITÄT DARMSTADT

- Better runtime performance
- Better Numerical Stability (Energy Conservation)

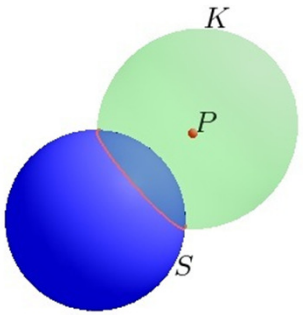


cocoon

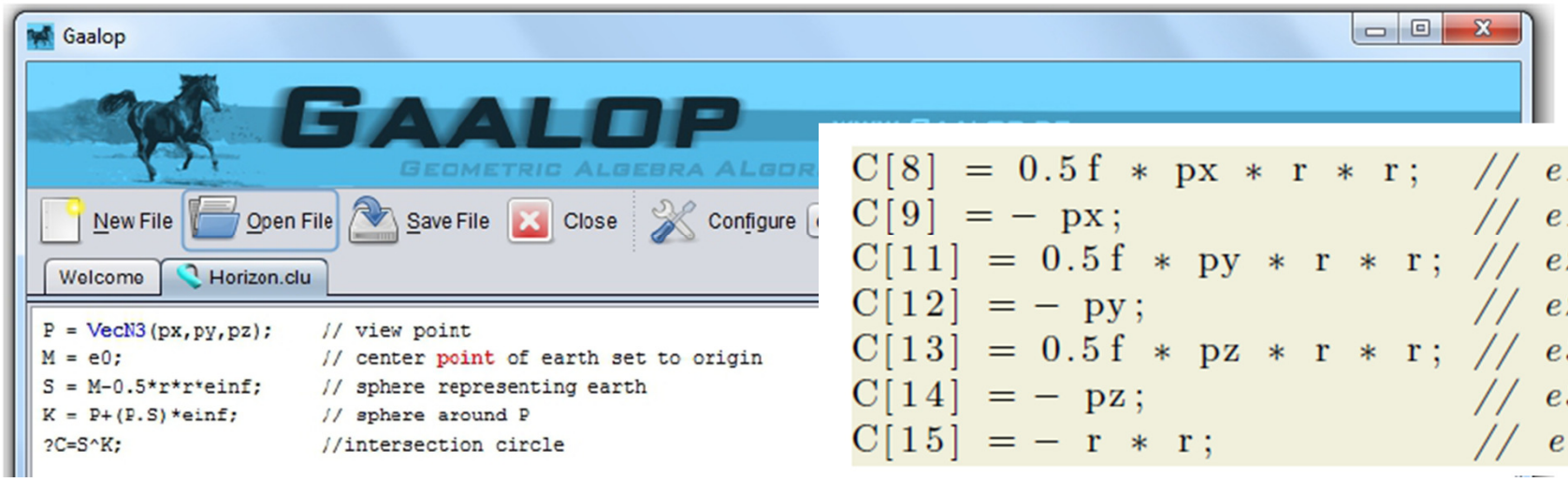
Conclusion

TECHNISCHE UNIVERSITÄT DARMSTADT

- Geometrically intuitive



- Fast and robust implementations



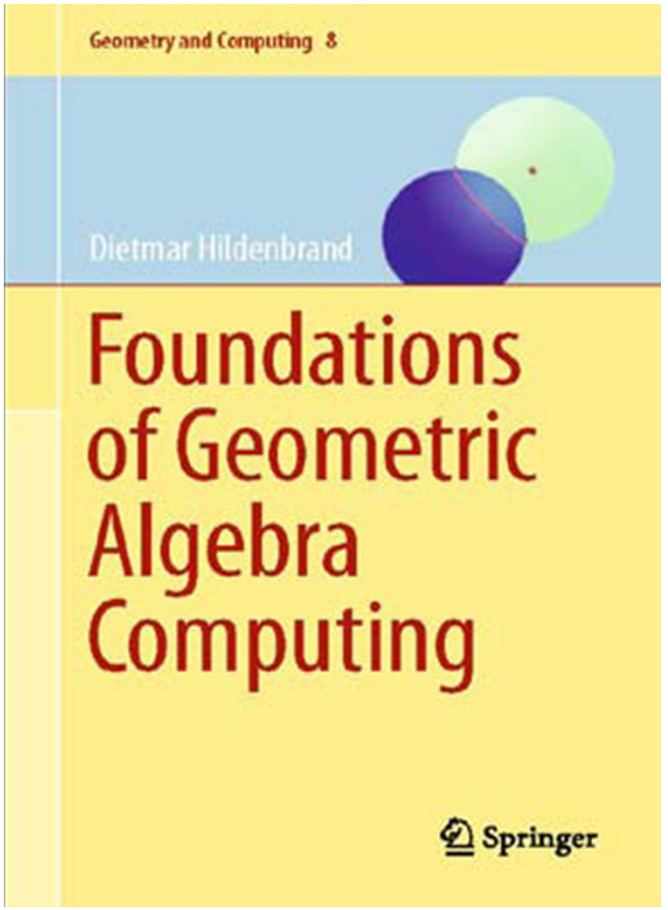

```
C[8] = 0.5f * px * r * r;   // e1^einf
C[9] = - px;                // e1^e0
C[11] = 0.5f * py * r * r;  // e2^einf
C[12] = - py;               // e2^e0
C[13] = 0.5f * pz * r * r;  // e3^einf
C[14] = - pz;               // e3^e0
C[15] = - r * r;            // einf^e0
```


COCOON

Reference

TECHNISCHE UNIVERSITÄT DARMSTADT

- „Foundations of Geometric Algebra Computing“
- Dietmar Hildenbrand
- Springer, Nov. 2012



- Article in Elektronik-Praxis:

http://www.elektronikpraxis.vogel.de/grundlagenwissen/articles/376044/

COCOON

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Übungsaufgaben

Übungsthemen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Mögliche Aufgaben:
  - Kegelschnitte und Quadriken in GA: Kapitel 6d/8d von Christian + 9D von Julio
  - Paper „Coordinate Free Perspective Projection of Points in the conformal Model“ von Stephen Mann
  - Beamforming-Paper
  - Selig-Paper zur Dynamik mit 8D
  - …

COCOON

Übungsthemen

TECHNISCHE UNIVERSITÄT DARMSTADT

- Mögliche Aufgaben:
  - Bsp.-Algorithmen für compass-ruler algebra mit Vergleichen zu herkömmlichen Algorithmen
  - Raytracer (Basis: OpenCL-Raytracer)
  - Game-Engine-Algorithmen mit Test und Vergleich
  - EM-Simulation mit OpenCL
  - Parallele Pi-Suche mit OpenCL
  - eigene Ideen?

  - Gaalop-Themen …

COCOON

Gaalop-Erweiterungen I

TECHNISCHE UNIVERSITÄT DARMSTADT

- Precompiler-Erweiterung
  - Visualisierung konformer Objekte
  - Visualisierung für 6d/8d
  - OpenMP / Performance?
  - CluCalc-Visualisierungs-Integration (in C- und Java-Version) in Gaalop?
  - Precompiler für Mathematica?
  - Precompiler für Java?
  - Gaalop -> LaTeX in lineare Algebra-Form (Skalar/Kreuzprodukte etc.)

COCOON

Gaalop-Erweiterungen II

TECHNISCHE UNIVERSITÄT DARMSTADT

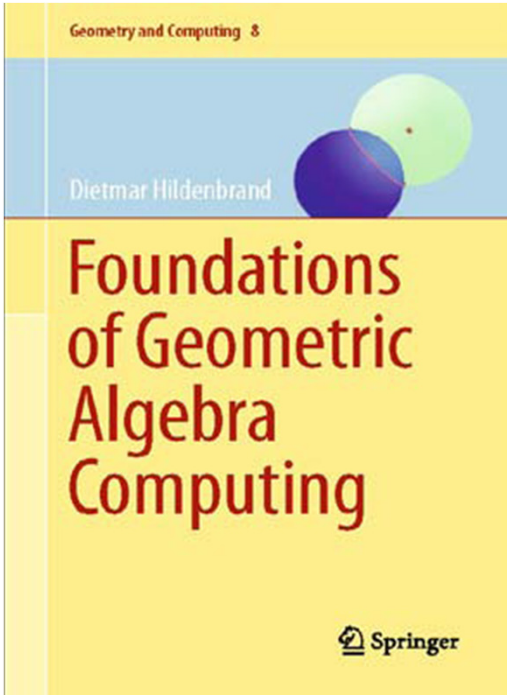
- Precompiler-Erweiterung
  - Gaalop for OpenACC
    - Cmake-Anpassung
    - OpenACC-Backend

COCOON

TECHNISCHE UNIVERSITÄT DARMSTADT

Thanks a lot

Google Dietmar Hildenbrand



COCOON