Simulation of Elastic Rods using conformal Geometric Algebra

Elmar Brendel[1], Thomas Kalbe[2], Dietmar Hildenbrand[3], Michael Schäfer[4]

[1,2,3,4]University of Technology Darmstadt, Germany

[1,3]Faculty of Interactive Graphics Systems Group
dietmar.hildenbrand@gris.informatik.tu-darmstadt.de

[2]Faculty of Interactive Graphics Systems Group
thomas.kalbe@gris.informatik.tu-darmstadt.de

[4]Department for Numerical Methods in Mechanical Engineering
schaefer@fnb.tu−darmstadt.de

Abstract. In this paper we present how Geometric Algebra can be used for deformation simulations. The aim is to capture the elastic behavior of simple components like rods. First we will review some other works in this field. Later we present an extended Finite Element Method, which has recently been developed and investigated. Our goal is a proof of concept that Geometric Algebra is able to improve Finite Element Methods. All algorithms presented in this paper work in real-time.

1 Introduction

Geometric Algebra (GA) is a mathematical framework to easily describe geometric aspects. It allows us to develop algorithms fast and in an intuitive way. It unifies many other mathematical concepts like quaternions and dual quaternions and is appropriate to model solutions for many different disciplines. GA is based on the work of Hermann Grassmann and William Clifford. Pioneering work has been done by David Hestenes, who firstly applied GA to problems in mechanics and physics. Rigid body motions can be described with the help of one linear expression, called *screw*, including both the rotational and the linear part. In the same way velocities and forces are represented. The combinations of rotational and linear velocities as well as of forces and torques are also described with the help of one linear expression including both the rotational and the linear parts.

2 Related Work

GA is used in different fields, e.g. physics, simulation and engineering. As a consequence, various introductions, intended for readers with different backgrounds exist. In computer graphics there are [3] [7], for example. The conformal model is emphasized there, which has useful attributes, as we will see in the following section. David Hestenes is one of the first, who applied GA for calculations of deformable objects. Based on his theoretical work [6], a simulation of an elastic coupling between rigid bodies has been developed [8]. A numeric solver using expressions in GA has been presented, which we have used in our approach as well. It is a generic tool and can be used for solving equations of motions of various kinds. The method is called *Iterated Commutators* [8] [2] and operates with compact terms in GA, representing the needed physical entities. This allows us to process linear and rotational motions at once without splitting them up into two separated expressions. This simplifies the implementation of the solver. The different cases, where only a translational movement or only a rotational movement are considered, do not need to be treated separately. So, only one uniform term has to be integrated. In our modified Finite Element Method (FEM), the location of the element vertices are given by screws. These screw displacements allow us to manipulate the position and orientation of the body in an easy and intuitive way. We can define an arbitrary axis in space and rotate the vertices, respectively the body around it. There is no need of an external application of transformations, for example with matrices. Velocities of the nodes are expressed by so-called *twists*. Forces and torques can be applied to the nodes by so-called *wrenches*. In our simulation we numerically integrate these entities with the already mentioned solver. We obtain the equations of motion from physical laws, which we discretized with the FEM. We use a simplified version of the FEM [4], that can be used in interactive real-time simulations. This type is based on the standard FEM [9], which is rather used in different engineering fields and often only works offline. Distances and intersections between objects can also easily be calculated with the products of GA, no matter how the objects are oriented in space. This improves the implementation of collision detection and collision response.

3 Introduction to conformal Geometric Algebra

Conformal GA uses five base vectors $e_1, e_2, e_3, e_0, e_\infty$. The first three vectors represent the three spatial directions, e_0 is an additional vector for the origin, e_∞ is an additional vector for the point at infinity. So, in conformal GA we do not have to treat these two special points separately. The algebra contains three products. The *outer product* is a grade increasing operation, which spans the *blades* and is also used for intersection calculations. The *inner product* is a grade decreasing operation and is used for distance and angle measures. The *geometric product* is a combination of outer and inner product and is mainly used to apply transformations.

3.1 Blades

The basic elements of this algebra are the so-called blades. These blades are spanned by the basis vectors described above with the help of the outer product. A 2-blade or *bivector* is an oriented area element spanned by two vectors, a 3-blade or *trivector* is an oriented volume element and so on. The highest grade of blades in conformal GA is a 5-blade or pseudoscalar. In total there are 32 unique vector combinations and therefore 32 basic blades.

3.2 Multivectors

The general elements of GA are *multivectors*. They are combinations of blades and generated by the geometric product. They have a geometric meaning and can be represented in two ways: the IPNS (inner product null space) or OPNS (outer product null space) representation, which are dual to each other.

3.3 Geometric Objects

Conformal GA contains some basic geometric objects. The user can create and work with them in an intuitive way and also can apply transformations directly to them.

Table 1. The basic geometric objects of conformal Geometric Algebra.

Object	IPNS	OPNS
Point	$P = \mathbf{x} + \frac{1}{2}\mathbf{x}^2 e_\infty + e_0$	
Sphere	$S = P - \frac{1}{2}r^2 e_\infty$	$S^* = x_1 \wedge x_2 \wedge x_3 \wedge x_4$
Plane	$\pi = \mathbf{n} + d e_\infty$	$\pi^* = x_1 \wedge x_2 \wedge x_3 \wedge e_\infty$
Circle	$Z = s_1 \wedge s_2$	$Z^* = x_1 \wedge x_2 \wedge x_3$
Line	$L = \pi_1 \wedge \pi_2$	$L^* = x_1 \wedge x_2 \wedge e_\infty$
Point Pair	$Pp = s_1 \wedge s_2 \wedge s_3$	$Pp^* = x_1 \wedge x_2$

3.4 Transformations

A transformation V can be applied to an object with the help of the geometric product in the following way.

$$Object' = V\ Object\ \tilde{V} \tag{1}$$

The *translator* T translates an object along a specific vector t.

$$T = 1 - \frac{te_\infty}{2} \tag{2}$$

The *rotor* R rotates an object around an arbitrary axis l in space by a certain angle ϕ.

$$R = \cos\frac{\phi}{2} + l\sin\frac{\phi}{2} \tag{3}$$

The *motor* (or screw) M is a combination of a rotor and translator. M rotates an object around an arbitrary axis in space by a certain angle and translates it along this axis at the same time.

$$M = R\,T \tag{4}$$

3.5 Physical Entities

The screw can be seen as a transformation, that displaces points in space. So, it can be used to represent a location in space instead of a position vector. This location is additionally equipped with an implicit orientation information because of the rotational component of the screw. This is the reason why screws are very useful for rigid body applications, because they can determine compactly all needed information for describing the body's state. A conventional definition of a screw is based on Plücker coordinates. A line described with these coordinates is extended by a ratio h between a translation along and a rotation around it. A screw can be expressed in this conventional form by a 6x1 vector [8]:

$$\mathcal{S} = \begin{pmatrix} \widehat{\mathbf{s}}^T \\ \mathbf{b} \times \widehat{\mathbf{s}} + h\widehat{\mathbf{s}} \end{pmatrix} \tag{5}$$

In conformal GA a screw can be expressed by a bivector, which we have already seen in equation (4).

The twist determines the velocity of a screw motion. It extends a screw in conventional representation by a scalar ω, which is the ratio between radians and time. A twist can be defined by a 6x1 vector [8]:

$$\mathcal{T} = \begin{pmatrix} \omega\widehat{\mathbf{s}} \\ \omega(\mathbf{b} \times \widehat{\mathbf{s}} + h\widehat{\mathbf{s}}) \end{pmatrix} \tag{6}$$

In conformal GA a twist can be expressed by a bivector:

$$\mathcal{T} = -(i\boldsymbol{\omega} + e_\infty\mathbf{v}) \tag{7}$$

Now $\boldsymbol{\omega}$ represents the rotational velocity vector and vector $\mathbf{v}$ denotes the linear velocity.

The wrench combines the force and torque acting on a body in one term. It is simply expressed by a 6x1 vector:

$$\mathcal{W} = \begin{pmatrix} \boldsymbol{\tau} \\ \mathbf{f} \end{pmatrix} \tag{8}$$

In conformal GA a wrench can be written in bivector form:

$$\mathcal{W} = i\boldsymbol{\tau} + e_0\mathbf{f} \tag{9}$$

In both forms the vector $\boldsymbol{\tau}$ is the torque vector and $\mathbf{f}$ is the force vector.

4 Modified Finite Element Approach

In our approach [2] we use a real-time version of the FEM. Those algorithms have a wide range of application and can be used in interactive simulations, game engines or medical training devices. The FEM divides a body into discrete elements, for example tetrahedra or cubes. For each element the stiffness matrix K_e is determined. After that the global stiffness matrix K can be assembled out of the single K_e and the arrangement of the elements. In the real-time FEM this only has to be done once before the simulation starts. The elastic forces are then calculated with $f = K \cdot \Delta x$, where Δx denote the displacements of the elements' vertices. Concepts of conformal GA have been used for position, velocity and force. So, the position of a vertex is defined by a screw, the velocity by a twist and the force acting at the vertex by a wrench. Then equations of motions for each vertex can be solved by the Iterated Commutators scheme, which works with these GA expressions.

4.1 Time Integration with Iterated Commutators

The Iterated Commutators solver [8] [2] is capable to process linear and rotational expressions in one term. It uses the screw representation of positions. The velocity represented in the body frame is given by a twist. Calculating the acceleration $\dot{V}(t)$ depends on the type of solver being used. The Iterated Commutators scheme is defined as follows:

$$\mathcal{T}(t + \Delta t) = \mathcal{T}(t) + \dot{V}(t)\ \Delta t \tag{10}$$

$$\mathcal{S}(t + \Delta t) = \mathcal{S}(t)\ e^{\frac{1}{2}\ \mathcal{T}(t)\ \Delta t} \tag{11}$$

The acceleration $\dot{V}$ is also given in form of a twist. Using an implicit scheme, $\dot{V}$ depends on the solution of a linear equation system. In an explicit scheme $\dot{V}$ can be calculated by the following equation:

$$\dot{V} = M^{-1} \cdot \mathcal{W}_{vec} \tag{12}$$

Note, that the wrench is given in the conventional 6x1 vector form (cp. equation (12)). The resulting twist describes the acceleration with reference to the body frame. The 6x6 Matrix M is denoted as generalized mass matrix. If it is defined in the body's center of mass, it has a diagonal shape and can be written as follows:

$$M = \begin{pmatrix} I & \mathbf{0} \\ \mathbf{0} & m\mathbf{1} \end{pmatrix} \tag{13}$$

The 3x3 submatrix I is the inertia tensor of the body. It describes the distribution of mass inside the body and determines its behavior under rotations. The scalar m denotes the mass of the body and is multiplied with the 3x3 identity matrix $\mathbf{1}$. It is easy to calculate the inversion M^{-1} because of the diagonal shape of M. Simply the scalar values on the diagonal have to be replaced by their reciprocals.

$$M^{-1} = \begin{pmatrix} I^{-1} & \mathbf{0} \\ \mathbf{0} & \frac{1}{m}\mathbf{1} \end{pmatrix} \tag{14}$$

The equation (12) can now be written as:

$$\dot{V} = \begin{pmatrix} I^{-1} & \mathbf{0} \\ \mathbf{0} & \frac{1}{m}\mathbf{1} \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{\tau} \\ \mathbf{f} \end{pmatrix} \tag{15}$$

The first three lines of this linear equation system determine the rotational dynamic behavior and the last three lines determine the linear dynamic behavior. In our FEM simulation all bodies are simple mass points *(the vertices of the elements)*. Thus we do not have a mass distribution and can use the identity matrix as M. In a rigid body application we have to adapt M to the properties of the body, of course. A slight improvement of the Iterated Commutator scheme (equations (10), (11)) is given in the next formulas. An additional Verlet step has been inserted to obtain a better accuracy.

$$\mathcal{T}(t + \frac{1}{2}\Delta t) = \mathcal{T}(t) + \frac{1}{2}\dot{V}(t)\ \Delta t \tag{16}$$

$$\mathcal{S}(t + \Delta t) = \mathcal{S}(t)\ e^{\frac{1}{2}\ \mathcal{T}(t+\frac{1}{2}\Delta t)\ \Delta t} \tag{17}$$

$$\mathcal{T}(t + \Delta t) = \mathcal{T}(t + \frac{1}{2}\Delta t) + \frac{1}{2}\dot{V}(t + \Delta t)\ \Delta t \tag{18}$$

4.2 Collision Detection

Conformal GA can also support the collision detection of objects. A collision between two objects can be detected very simply. The inner product of two objects of conformal GA results in the distance between them. Consider a plane π representing the rigid ground. A deformable body can collide with this plane because of gravity. The distance d between all vertices i of the body and the ground π can be calculated by the next equation:

$$d = \pi \ . \ (\mathcal{S}_i \ e_0 \ \tilde{\mathcal{S}}_i) \tag{19}$$

Generally, we cannot prevent a point from penetrating the ground because of the inaccuracy of the numeric time integration method. So, one has to define a threshold, that determines whether a point is colliding with the plane or not. The smaller the time step Δt of the numeric solver the less deep the penetration of the particle in the ground will be. The advantage of the GA approach is, that the calculations are very compact and independent of the position and orientation of the colliding objects. Other types of objects (cp. table 1) can be taken without or only with slight changes of equation (19). Collisions between more complex objects can be efficiently modeled with bounding sphere hierarchies. Operators for intersection determination and distance measurements are already embedded in conformal GA. If a collision has been detected, it must trigger an event to simulate the impact. A very simple collision handling between a fixed rigid ground and a deformable body can be done in the following way. The method keeps waiting until a particle of the deformable body penetrates the ground due to numeric reasons. If the particle is inside the ground, a force acts on it. The direction of the force is the same as the direction of the ground's surface normal **n**. Its magnitude depends on the distance ϵ between the point and the surface. The factor k can be seen as the hardness of the ground.

$$\mathbf{f}_{coll} = \mathbf{n}\epsilon \cdot k \tag{20}$$

5 Conclusion and Discussion

In the future, GA will more and more improve algorithms and geometric calculations. At the first view, GA algorithms often tend to be slower than conventional algorithms. Without optimizations this is usually true. One have to take into account, that there is no hardware support for calculations in GA. For example, matrix computations in computer graphics are highly supported by the hardware. Recently much effort has been put into optimizing expressions of conformal GA. For our simulation source files we used a software named Gaalop to convert expressions in GA into optimized C code. A main drawback of our simulation is the unrealistic behavior of the body under large bendings. The body unnaturally gains volume due to the usage of the real-time FEM. There the Cauchy strain tensor is being used, which allows us to calculate the stiffness matrix K only once, at startup. Unfortunately, this tensor is not invariant up to rotations, which leads to this incorrect behavior. A technique called stiffness warping successfully overcomes this problem. There is even a GA implementation of this method [5], which would be interesting to compare with conventional algorithms in terms of accuracy and performance. At last we want to review the main benefits of our approach.

5.1 Benefits

Applying rotations. The locations of the vertices of the elastic body are determined by conformal screws. These screw displacements allow us to manipulate the position and orientation of the body in an easy and intuitive way. We can define an arbitrary axis in space and rotate the vertices, respectively the body around it. Also a rotation around the coordinate axes can be performed. There is no need of an external application of transformations, for example with matrices.

Time integration. The numerical time integration is done by the Iterated Commutators solver. This solver is directly applicable to expressions in conformal GA. This allows us to process linear and rotational motions at once without splitting them up into two separated expressions. This simplifies the implementation of the solver. The different cases, where only a translational movement or only a rotational movement are considered, do not need to be treated separately. So, only one uniform term has to be integrated.

Collision detection. The collision detection and collision response are a big issue since the beginning of interactive computer graphics. GA can improve the simplification of some aspects of collision detection. The main advantage is the easy distance measure between objects. It is completely independent from the position and orientation of the objects and works almost identical for various types of objects. Recently GA has find one's way into commercial products in this field. The company Geomerics [1] uses GA successfully for collision detection as well as dynamic light calculations.

References

1. The homepage of geomerics ltd. HTML document http://www.geomerics.com.
2. Elmar Brendel. Simulation of elastic rods in conformal geometric algebra, bachelor thesis, TU Darmstadt, 2008.
3. L. Dorst, D. Fontijne, and S. Mann. *Geometric Algebra for Computer Science, An Object-Oriented Approach to Geometry.* Morgan Kaufman, 2007.
4. Andrew Nealen et al. Physically based deformable models in computer graphics. Technical report, Discrete Geometric Modeling Group, TU Darmstadt, 2005.
5. Matthias Müller et al. Stable real-time deformations. In *Proceedings of ACM SIGGRAPH Symposium on Computer Animation (SCA) 2002, pp 49-54*, 2002.
6. D. Hestenes and E. Fasse. Homogeneous rigid body mechanics with elastic coupling. In Leo Dorst, Chris Doran, and Joan Lasenby, editors, *Applications of Geometric Algebra in Computer Science and Engineering.* Birkhäuser, 2002.
7. D. Hildenbrand. Geometric computing in computer graphics using conformal geometric algebra. *Computers & Graphics*, 29(5):802–810, 2005.
8. Thomas Kalbe. Beschreibung der dynamik elastisch gekoppelter koerper in konformaler geometrischer algebra. Master's thesis, TU Darmstadt, 2006.
9. Michael Schäfer. *Computational Engineering, Introduction to Numerical Methods.* Springer, 2006.