D. Hildenbrand, E. Hitzer, Analysis of Point Clouds - Using conformal geometric algebra, in Braz, J. (ed.), GRAPP 2008, 3rd Int. Conf. on Computer Graphics Theory and Applications. Proc.: Funchal, Madeira, Portugal, January 22-25, 2008, Porto: INSTICC Press, pp. 99-106 (2008). DOI: 10.1.1.151.7539

ANALYSIS OF POINT CLOUDS
Using Conformal Geometric Algebra

Dietmar Hildenbrand
Research Center of Excellence for Computer Graphics, University of Technology, Darmstadt, Germany
Dietmar.Hildenbrand@gris.informatik.tu-darmstadt.de

Eckhard Hitzer
Department of Applied Physics, University of Fukui, Japan
hitzer@mech.fukui-u.ac.jp

Keywords: geometric algebra, geometric computing, point clouds, osculating circle, fitting of spheres, bounding spheres.

Abstract: This paper presents some basics for the analysis of point clouds using the geometrically intuitive mathematical framework of conformal geometric algebra. In this framework it is easy to compute with osculating circles for the description of local curvature. Also methods for the fitting of spheres as well as bounding spheres are presented. In a nutshell, this paper provides a starting point for shape analysis based on this new, geometrically intuitive and promising technology.

1 INTRODUCTION

The main contribution of this paper are properties and basic algorithms based in conformal geometric algebra (CGA) promising for the analysis of point clouds. We refer e.g. to (Schnabel et al., 2007) for current research with this application.

CGA has shown some advantages in recent years. It is very easy to directly calculate with geometric objects like spheres, circles and planes and to transform them. CGA unifies a lot of mathematical systems like vector algebra, projective geometry, quaternions and Plücker coordinates, etc.

While CGA formerly had the problem of efficiency, new approaches provide CGA algorithms that can even be faster than conventional algorithms (Hildenbrand et al., 2006), (Hildenbrand et al., 2008).

Like implementations of quaternions can be more robust than rotation matrices CGA promises to deliver more robust algorithms. As an example, planes can be represented as specific spheres allowing to fit both into point sets (Hildenbrand, 2005).

For the foundations of CGA and its application to computer graphics we refer to (Dorst et al., 2007), (Rosenhahn, 2003), (Hitzer, 2004) and to tutorials (Hildenbrand et al., 2004), (Hildenbrand, 2005).

2 CONFORMAL GEOMETRIC ALGEBRA FOUNDATIONS

2.1 Geometric Algebra of 3D Space

Geometric algebra (GA) uses a dimension independent way of vector multiplication, adding inner and outer products.

$$e_i e_j = e_i \cdot e_j + e_i \wedge e_j, \ 1 \leq i, j \leq n, \tag{1}$$

where the vectors $\{e_i : 1 \leq i \leq n\}$ form a vector space basis. The outer product is antisymmetric

$$e_i \wedge e_j = -e_j \wedge e_i, \quad 1 \leq i, j \leq n. \tag{2}$$

In three dimensions the $\mathcal{R}^3$ basis $\{e_1, e_2, e_3\}$ with

$$e_1^2 = e_2^2 = e_3^2 = 1, \ e_1 \cdot e_2 = e_2 \cdot e_3 = e_3 \cdot e_1 = 0 \tag{3}$$

yields an eight dimensional GA of subspace blades

$$\{1, e_1, e_2, e_3, e_1 e_2, e_2 e_3, e_3 e_1, e_1 e_2 e_3\}, \tag{4}$$

of a grade 0 scalar, three grade 1 vectors, three grade 2 bivector areas, and a grade 3 unit volume trivector $e_{123} = e_1 e_2 e_3$. Adding blades of the same grade k gives a k-vector, a linear combination of blades of different grades gives a multivector.

A vector e_1 can represent a Euclidean point. But via the inner product e_1 can also represent a normal plane (standard representation) through the origin

$$\mathbf{x} \cdot e_1 = 0 \Leftrightarrow \mathbf{x} \in \mathit{Plane}. \tag{5}$$

If we want to *reflect* a general position vector $\mathbf{y}$ at the plane (5) we simply reverse the component of $\mathbf{y}$ parallel to e_1 (perpendicular to the plane) by

$$-e_1\mathbf{y}e_1 = -e_1(\mathbf{y}\cdot e_1 + \mathbf{y}\wedge e_1) = \mathbf{y} - 2(\mathbf{y}\cdot e_1)e_1, \quad (6)$$

because $\mathbf{y}\wedge e_1 = -(e_1\wedge\mathbf{y}) = -(e_1\mathbf{y} - e_1\cdot\mathbf{y})$.

It is well known that the product of two reflections is a *rotation* about the axis in which their planes intersect, by twice the smaller angle between the planes. So using a second plane normal to $\mathbf{n}$, $\angle(e_1,\mathbf{n}) = \theta/2$, we get a rotation by θ in the $e_1,\mathbf{n}$-plane by

$$-\mathbf{n}(-e_1\mathbf{y}e_1)\mathbf{n} = (\mathbf{n}e_1)\mathbf{y}(e_1\mathbf{n}) = \widetilde{R}\mathbf{y}R. \quad (7)$$

The rotation is discribed in (7) by the (scalar + bivector) rotation operator $R = e_1\mathbf{n} = e_1\cdot\mathbf{n} + e_1\wedge\mathbf{n}$, $\widetilde{R} = \mathbf{n}e_1$, completely equivalent to *quaternions*.

Via the outer product e_1 may further represent a line through the origin with direction e_1 (direct repr.)

$$\mathbf{x}\wedge e_1 = 0 \Leftrightarrow \mathbf{x}\in \mathit{Line}^*. \quad (8)$$

Bivectors like $e_2e_3 = e_2\wedge e_3$ represent lines in the standard repr. (intersection of normal planes)

$$\mathbf{x}\cdot(e_2e_3) = (\mathbf{x}\cdot e_2)e_3 - (\mathbf{x}\cdot e_3)e_2 = 0 \Leftrightarrow \mathbf{x}\in \mathit{Line}. \quad (9)$$

and planes in the direct representation

$$\mathbf{x}\wedge(e_2e_3) = \mathbf{x}\wedge e_2\wedge e_3 = 0 \Leftrightarrow \mathbf{x}\in \mathit{Plane}^*. \quad (10)$$

We therefore see that planes and lines in the two representations are *dual* to each other, because e.g.

$$(e_2e_3)(e_3e_2e_1) = e_2(e_3e_3)e_2e_1 = e_2e_2e_1 = e_1. \quad (11)$$

The (dihedral) angle between (planes) lines o_1, o_2 can be calculated representation independent as

$$cos(\theta) = \frac{o_1\cdot o_2^{-1}}{|o_1||o_2^{-1}|} \quad (12)$$

2.2 Conformal Geometric Algebra

In order to model objects away from the origin and to describe curved objects, conformal GA (CGA) adds an origin-infinity plane to 3D Euclidean space. In order to retain as much Euclidean structure as possible we demand for origin vector e_0, infinity vector e_∞, and e_1, e_2, e_3 of (4)

$$e_i\cdot e_0 = e_i\cdot e_\infty = 0,\ e_\infty^2 = e_0^2 = 0,\ e_\infty\cdot e_0 = -1. \quad (13)$$

The origin-infinity plane is given by its bivector blade $E = e_\infty\wedge e_0$.

Euclidean points, planes and spheres can all be embedded (normed stand. repr.) in CGA by 5D vectors:

$$P = \mathbf{p} + \frac{1}{2}\mathbf{p}^2e_\infty + e_0, \quad (14)$$

$$S = \mathbf{s} + \frac{1}{2}(\mathbf{s}^2 - r^2)e_\infty + e_0, \quad (15)$$

$$\pi = \mathbf{n} + de_\infty. \quad (16)$$

Table 1: Representations of the conf. geometric objects.

object	**standard repr.**	**direct repr.**
Point	$P = \mathbf{x} + \frac{1}{2}\mathbf{x}^2e_\infty + e_0$	
Sphere	$S = P - \frac{1}{2}r^2e_\infty$	$S^* = P_1\wedge P_2\wedge P_3\wedge P_4$
Plane	$\pi = \mathbf{n} + de_\infty$	$\pi^* = P_1\wedge P_2\wedge P_3\wedge e_\infty$
Circle	$Z = S_1\wedge S_2$	$Z^* = P_1\wedge P_2\wedge P_3$
Line	$L = \pi_1\wedge\pi_2$	$L^* = P_1\wedge P_2\wedge e_\infty$
P-Pair	$P_p = S_1\wedge S_2\wedge S_3$	$P_p^* = P_1\wedge P_2$

Where $\mathbf{p}$ is the 3D point location, $\mathbf{s}$ and r the 3D sphere center and the sphere radius, $\mathbf{n}$ the 3D unit normal vector of the plane, and d the distance of the plane from the origin. A point is thus a sphere with $r = 0$.

Conformal objects are homogeneous, but in (14) and (15) we have *normed* them by keeping the coefficient of e_0 one. This is not necessay, but helpful regarding numerical implementations. We can always norm them by division with $-P\cdot e_\infty$ or $-S\cdot e_\infty$, respectively. The plane π can be normed by division with $|\mathbf{n}|$.

The standard representation of a conformal subspace V uses conformal blades A_V via the inner product

$$X\cdot A_V = 0 \Leftrightarrow X\in V. \quad (17)$$

The direct representation is based on the outer product and is *dual* to the standard representation via the division with the unit 5D volume $I_5 = e_{123}E$ (similar to (11))

$$X\wedge A_V^* = 0 \Leftrightarrow X\in V, \quad A_V^* = A_VI_5^{-1} = -A_Ve_{123}E. \quad (18)$$

In the conformal (conf.) standard repr. the intersection of two spheres (planes) gives a circle (line) and the intersection of three spheres a point pair, see table 1. In the conf. direct repr. all these objects can be constructed by joining conf. object points by outer products (table 1).

In CGA two reflections at parallel planes yield a translation by twice the 3D distance vector $\mathbf{t}/2$ of the planes

$$\pi_2\pi_1P\pi_1\pi_2 = \widetilde{T}PT,\ T = \pi_1\pi_2 = 1 + \frac{1}{2}\mathbf{t}e_\infty. \quad (19)$$

The blade E can be used to extract pure 3D Eucl. parts from conf. multivectors M

$$M_{3D} = (M\wedge E)\cdot E, \quad (20)$$

e.g. $P_{3D} = \mathbf{p}$, $S_{3D} = \mathbf{s}$ and $\pi_{3D} = \mathbf{n}$.

3 DISTANCES AND ANGLES

The inner product of conf. vectors for points, spheres and planes results in a scalar, and can be used as an (oriented) distance (or angle) measure between these geometric objects. Minus two times the inner product of two (normed) sphere vectors gives according to (3), (13) and (15)

$$-2(S_1 \cdot S_2) = (\mathbf{s}_1 - \mathbf{s}_2)^2 - r_1^2 - r_2^2, \qquad (21)$$

which is the square of the 3D distance of the centers minus the squares of the radii. Based on (21) we observe that for

$S_1 \cdot S_2 > 0 : S_1$ intersects S_2 (P_1 inside S_2)
$S_1 \cdot S_2 = 0 : S_1$ touches S_2 from outside (P_1 on S_2)
$S_1 \cdot S_2 < 0 : S_1, S_2$ do not intersect (P_1 outside S_2)

The relations in parenthesis result from (21) if S_1 degenerates ($r_1 = 0$) to a point P_1, i.e.

$$-2(P_1 \cdot S_2) = (\mathbf{p}_1 - \mathbf{s}_2)^2 - r_2^2. \qquad (22)$$

If S_2 also degenerates ($r_2 = 0$) to a point P_2, we get from (22) the 3D distance square of the two points. If the two spheres become identical $S_1 = S_2$ we get from from (21) the radius square itself

$$S_1^2 = r_1^2. \qquad (23)$$

The inner product of a sphere S and a plane π gives the oriented 3D distance of the sphere center $\mathbf{s}$ from the plane

$$S \cdot \pi = \mathbf{s} \cdot \mathbf{n} - d. \qquad (24)$$

The sign of (24) shows on which side of the plane the sphere center lies (relative to the origin). If the sphere S degenerates ($r = 0$) to a point P, (24) results in the oriented 3D distance of the point P from the plane π.

Finally the inner product of two conf. lines (or planes) results like in (12) in the cosine of their (dihedral) angle. The same is also true for two circles (or point pairs). Then the inner product yields the cosine of the angle of the respective carrier planes (lines) of the two circles (point pairs).

4 DIFFERENTIAL GEOMETRY

In order to analyze point clouds some properties of CGA are very helpful. Assuming some kind of curvature information of point clouds the easy handling of geometric objects like circles and lines can be used for the local description of curvature. Based on these local properties the existence of geometric objects like cylinder, sphere, cone or torus can be investigated.

(Adamson, 2007) already developed algorithms to estimate local curvature information including principal curvatures. These can be very easily transferred into algebraic expressions for locally fitting objects. These objects include

- osculating circle (fig. 1)
- line describing vanishing curvature (fig. 2)
- osculating circle with vanishing radius (fig. 3)

and can be treated very consistently since all these objects are easy to compute with CGA bivectors (stand. repr. of table 1).

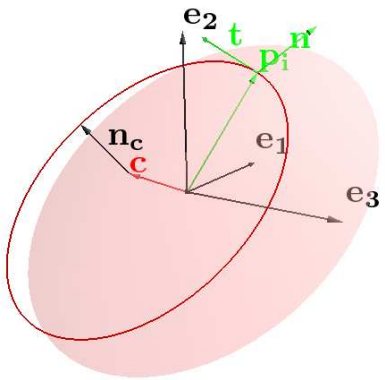

Figure 1: Osculating circle describing curvature at point $\mathbf{p}_i$ and local coordinate system (see Figure 3).

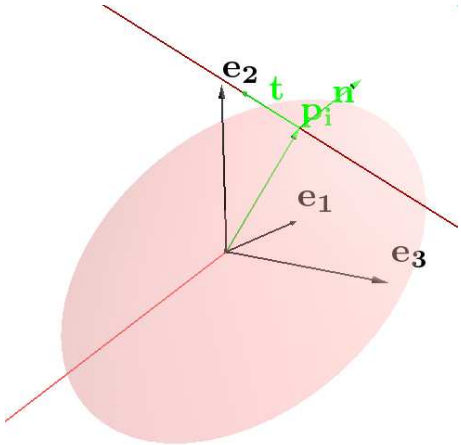

Figure 2: Line describing vanishing curvature at point $\mathbf{p}_i$ and local coordinate system (see Figure 3).

Circles (3D center $\mathbf{c}$, radius r) can be described with the help of the outer product of 3 conf. points lying on the the circle or as the intersection of a sphere and a plane (normal $\mathbf{n_c}$) resulting in the following for-

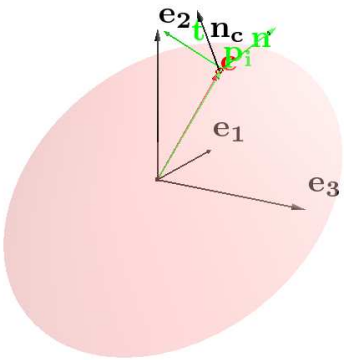

Figure 3: Local coordinate system at point p_i based on the tangent vector **t** and the normal vector **n**.

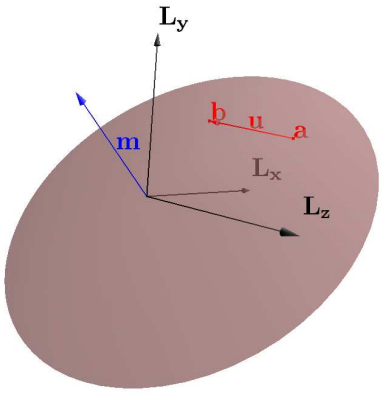

Figure 4: The line through **a**, **b** and its plücker parameters **u** and **m** of equation (25).

mula

$$Z = \mathbf{c} \wedge \mathbf{n_c} - \mathbf{n_c} \wedge e_0 - (\mathbf{c} \cdot \mathbf{n_c})E$$
$$+[(\mathbf{c} \cdot \mathbf{n_c})\mathbf{c} - \frac{1}{2}(\mathbf{c}^2 - r^2)\mathbf{n_c}] \wedge e_\infty.$$

Lines can be described with the help of the outer product of 2 planes (i.e. intersection in stand. repr.) which explicitly leads to (dir. repr. $L^* = A \wedge B \wedge e_\infty$)

$$L = \mathbf{u}e_{123} + \mathbf{m} \wedge e_\infty, \qquad (25)$$

with **a**, **b** two 3D points on the line, $\mathbf{u} = \mathbf{b} - \mathbf{a}$ as 3D direction vector, and $\mathbf{m} = \mathbf{a} \times \mathbf{b}$ as the 3D moment vector (relative to origin). The corresponding six Plücker coordinates (componentes of **u** and **m**) are (see fig. 4)

$$(\mathbf{u} : \mathbf{m}) = (u_1 : u_2 : u_3 : m_1 : m_2 : m_3). \qquad (26)$$

5 Fitting of Points With a Sphere

While in (Hildenbrand, 2005) fitting of spheres or planes into point clouds is described, in this section a point cloud $\mathbf{p_i} \in \mathbb{R}^3$, $i \in \{1, ..., n\}$ will be approximated specifically with the help of a sphere. We will use the conf. representation (14) for the points and (15) for the sphere S.

The inner product (22) provides us with a distance measure, we will therefore seek the *least square minimum* of

$$\sum_{i=1}^{n} [P_i \cdot S]^2 = \sum_{i=1}^{n} \left[\mathbf{p}_i \cdot \mathbf{s} - \frac{1}{2}\mathbf{p}_i^2 - \frac{1}{2}(\mathbf{s}^2 - r^2) \right]^2 \qquad (27)$$

The four coordinates of the sphere (15) are $s = (s_1, s_2, s_3, s_4)$, where $s_4 = (\mathbf{s}^2 - r^2)/2$. In the minimum the derivatives of (27) with respect to the components s_1, s_2, s_3, and s_4 will be zero. We thus get a linear system of four equations

$$\begin{pmatrix} \sum p_{i,1}p_{i,1} & \sum p_{i,2}p_{i,1} & \sum p_{i,3}p_{i,1} & -\sum p_{i,1} \\ \sum p_{i,1}p_{i,2} & \sum p_{i,2}p_{i,2} & \sum p_{i,3}p_{i,2} & -\sum p_{i,2} \\ \sum p_{i,1}p_{i,3} & \sum p_{i,2}p_{i,3} & \sum p_{i,3}p_{i,3} & -\sum p_{i,3} \\ -\sum p_{i,1} & -\sum p_{i,2} & -\sum p_{i,3} & \sum 1 \end{pmatrix} \cdot s$$
$$= \begin{pmatrix} \frac{1}{2}\sum \mathbf{p}_i^2 p_{i,1} \\ \frac{1}{2}\sum \mathbf{p}_i^2 p_{i,2} \\ \frac{1}{2}\sum \mathbf{p}_i^2 p_{i,3} \\ -\frac{1}{2}\sum \mathbf{p}_i^2 \end{pmatrix}$$

with $p_{i,1}, p_{i,2}, p_{i,3}$ the 3D coordinates of the points $\mathbf{p}_i$. We sum over the whole cloud $\sum = \sum_{i=1}^{n}$. The result s represents the center point of the approximation sphere (s_1, s_2, s_3) and its radius as $r^2 = s_1^2 + s_2^2 + s_3^2 - 2s_4$.

6 BOUNDING SPHERE ALGORITHM

The problem of defining a bounding sphere of a point cloud can be subdivided into three sub-problems:

1. How to enclose a set of points by a minimal sphere.
2. How to minimally expand an existing bounding sphere when adding more points.
3. How to merge existing bounding spheres.

Because points can be treated as spheres of zero radius, case 2 becomes part of case 3 if the latter is solved for bounding spheres of general radii.

6.1 Clouds of one, two or three Points

If the cloud consists of only one point (14), then this point defines its own bounding sphere with conf. center P and radius $r = 0$.

If the cloud consists of two (normed) conf. points P_1, P_2, then the minimal bounding sphere has P_1 and

P_2 as its poles. The conformal sphere vector (see left side of Fig. 5) is then given by

$$S = \frac{1}{2}(P_1 + P_2). \tag{28}$$

The factor one half is convenient for norming S such that $r^2 = S^2$, because of (23). The Euclidean center vector of the normed sphere (28) is given according to (20) by

$$\mathbf{s} = (S \wedge E) \cdot E. \tag{29}$$

In the case of three conformal points P_1, P_2 and P_3, we can first define an initial sphere with two points (e.g. P_1, P_2) as in Equ. (28). Then we can regard the third point as a second sphere with zero radius and center P_3 and apply the method for the bounding of two spheres described in subsection 6.3. Or we can directly expand the sphere to minimally include the third point in the following way.

6.2 Minimally Including a New Point

We describe this alternative (compared to subsection 6.3) way in order to show that CGA offers a variety of algorithmic constructions, some of which may be preferable for specific tasks and for numerical optimization.

We show two variants in the form of CLUCalc Scripts. The first more from a geometric algebra principle point of view, the second based on further code performance optimization with Maple.

```
DefVarsN3(); // Use of conformal model
:IPNS; // Use of standard representation
C = VecN3(c1,c2,c3);
  // Conformal sphere center
r = r0; // Sphere radius
S = C-0.5*r*r*einf;
  // Definition of initial sphere S
P = VecN3(p1,p2,p3);
  // New point outside S
d = sqrt(-2*P.C);
  // Distance from sphere center C to P
if (d>r){
  CP = (p1-c1)*e1+(p2-c2)*e2+(p3-c3)*e3;
    // 3D vector from C to P
  T  = 1 + 0.5*(0.5*(1-r/d)*CP)*einf;
    // Translator to new sphere center
  C1 = ~T*C*T;  // Center of new sphere
  r1 = (d+r)/2; // Radius of new sphere
  S1 = C1 - 0.5*r1*r1*einf;
    // Definition of new sphere          }
```

We see in the CLUCalc Script that the initial sphere $S = C - \frac{1}{2}r_0^2 e_\infty$ is only expanded if the new point P is outside the sphere S ($d > r$). C_1 is the center of the expanded sphere S_1, obtained by shifting

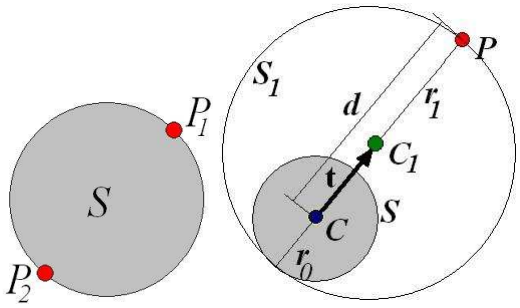

Figure 5: Left: Sphere vector constructed from two pole points. Right: Minimally adding a point P to a sphere S by adjusting center and radius.

C in the direction of P by $\mathbf{t} = \frac{1}{2}(d-r)\frac{CP}{|CP|}$, because $d = |CP|$. $r_1 = (d+r)/2$ is the radius of the minimally expanded sphere S_1. Compare right side of Fig. 5.

For numerical optimization we can replace the definition of d and the if loop by shorter Maple optimized code

```
d = sqrt((p1-c1)*(p1-c1)+(p2-c2)*(p2-c2)
         +(p3-c3)*(p3-c3));
if (d>r){
c1x = (-r*p1+r*c1+p1*d+c1*d)/d/2;
c1y = ( r*c2+p2*d+c2*d-r*p2)/d/2;
c1z = ( p3*d-r*p3+r*c3+c3*d)/d/2;
r1  = (d+r)/2;
S1  = VecN3(c1x,c1y,c1z)-0.5*r1*r1*einf; }
```

6.3 Bounding Two Spheres

We can merge (bound) two spheres by defining a straight line g which connects the centers of the two (normed) spheres S_1 and S_2

$$g^* = (S_1 \wedge S_2 \wedge e_\infty)^*. \tag{30}$$

The Eulidean unit vector in the direction of g is then

$$\mathbf{d} = \frac{g^* \cdot e_{123}}{|g^* \cdot e_{123}|}. \tag{31}$$

We now calculate the poles of the minimal bounding sphere. The first pole is the point of intersection of g with S_1 away from S_2

$$\mathbf{q}_1 = \mathbf{s}_1 - r_1 \mathbf{d}, \tag{32}$$

with $\mathbf{s}_1$ and r_1 calculated according to (29) and (23). We similarly obtain the second pole of the bounding sphere as the point of intersection of g with S_2 away from S_1

$$\mathbf{q}_2 = \mathbf{s}_2 + r_2 \mathbf{d}, \tag{33}$$

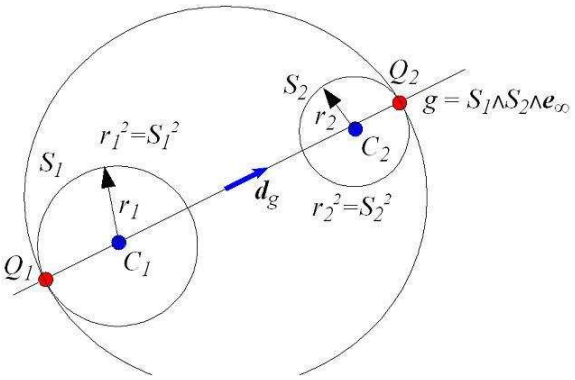

Figure 6: Minimal bounding sphere of two spheres.

The corresponding conformal poles (14) give according to (28) the mininal bounding sphere (see Fig. 6)

$$S_{12} = \frac{1}{2}(Q_1 + Q_2). \tag{34}$$

In this section we developed the algebraic expressions for the minimal bounding sphere [eqs. (30) to (34)] for easy numerical implementation. It would be possible to completely carry out the calculation of S_{12} on the conf. level. For that purpose we would first calculate the conf. point pairs (direct repr.) by intersecting line g with spheres S_1 and S_2, respectively

$$pair_1 = g \cdot S_1, \qquad pair_2 = g \cdot S_2. \tag{35}$$

With

$$Q_1 = \frac{pair_1 + |pair_1|}{pair_1 \cdot e_\infty}, \quad Q_2 = \frac{pair_2 - |pair_2|}{pair_2 \cdot e_\infty} \tag{36}$$

we can then directly pick the conformal points Q_1 and Q_2 out of the point pairs $pair_1$ and $pair_2$.

If (like in subsection 6.2) the second sphere S_2 happens to be only a point (sphere with zero radius), we can omit the calculation of Q_2 in Equs. (33), (35) and (36). We simply replace $Q_2 = S_2$ in Equ. (34).

6.4 Comparison with Welzl's Bounding Sphere Algorithm

The iteration of the method suggested in subsection 6.3 (test of inclusion, and if necessary the calculation of the new bounding sphere) results in the final bounding sphere of n points in linear time $O(n)$. The proposed method is easy to understand and with given routines for inner and outer products easy to implement. In average Welzl's algorithm (Welzl, 1991) also runs in asymptotically linear time, but the recursion in Welzl's algorithm makes it harder to examine and guarantee the performance time. As demonstrated in subsection 6.2 our algorithm can be further optimized with Maple as well as based on reconfigurable hardware with the potential of dramatically reducing the constant factor of our $O(n)$ algorithm as described for an inverse kinematics algorithm in (Hildenbrand et al., 2008).

7 CONCLUSION

We introduced the framework of conformal geometric algebra (CGA). We highlighted representation and manipulation of geometric objects in CGA. We are convinced that in CGA the easy handling of objects like spheres, circles or planes, the seemless computation of distances and angles between them, as well as the new possibilities for the fitting and bounding of geometric objects will provide a promising foundation for the analysis of point clouds.

REFERENCES

Adamson, A. (2007). *Computing Curves and Surfaces from Points*. PhD thesis, Darmstadt University of Technology.

Dorst, L., Fontijne, D., and Mann, S. (2007). *Geometric Algebra for Computer Science, An Object-Oriented Approach to Geometry*. Morgan Kaufman.

Hildenbrand, D. (2005). Geometric computing in computer graphics using conformal geometric algebra. *Computers & Graphics*, 29(5):802–810.

Hildenbrand, D., Fontijne, D., Perwass, C., and Dorst, L. (2004). Tutorial geometric algebra and its application to computer graphics. In *Eurographics conference Grenoble*.

Hildenbrand, D., Fontijne, D., Wang, Y., Alexa, M., and Dorst, L. (2006). Competitive runtime performance for inverse kinematics algorithms using conformal geometric algebra. In *Eurographics conference Vienna*.

Hildenbrand, D., Lange, H., Stock, F., and Koch, A. (2008). Efficient inverse kinematics algorithm based on conformal geometric algebra using reconfigurable hardware. In *GRAPP conference Madeira*.

Hitzer, E. (2004). Euclidean geometric objects in the clifford geometric algebra of Origin, 3-Space, Infinity. *Bulletin of the Belgian Mathematical Society - Simon Stevin*, 11(5):653–662.

Rosenhahn, B. (2003). *Pose Estimation Revisited*. PhD thesis, Inst. f. Informatik u. Prakt. Mathematik der Christian-Albrechts-Universität zu Kiel.

Schnabel, R., Wahl, R., and Klein, R. (2007). Efficient ransac for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226.

Welzl, E. (1991). Smallest enclosing disks (balls and ellipsoids. In *Lecture Notes in Computer Science*, pages 555:359–370.

D. Hildenbrand, E. Hitzer, Analysis of Point Clouds - Using conformal geometric algebra, in Braz, J. (ed.), GRAPP 2008, 3rd Int. Conf. on Computer Graphics Theory and Applications. Proc.: Funchal, Madeira, Portugal, January 22-25, 2008, Porto: INSTICC Press, pp. 99-106 (2008). DOI: 10.1.1.151.7539