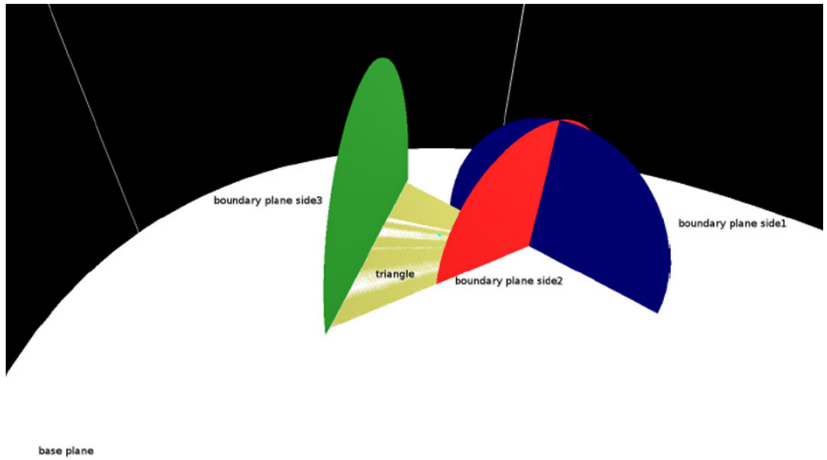
Geometric Algebra Computing

13.11.2014

Dr.-Ing. Dietmar Hildenbrand
Technische Universität Darmstadt

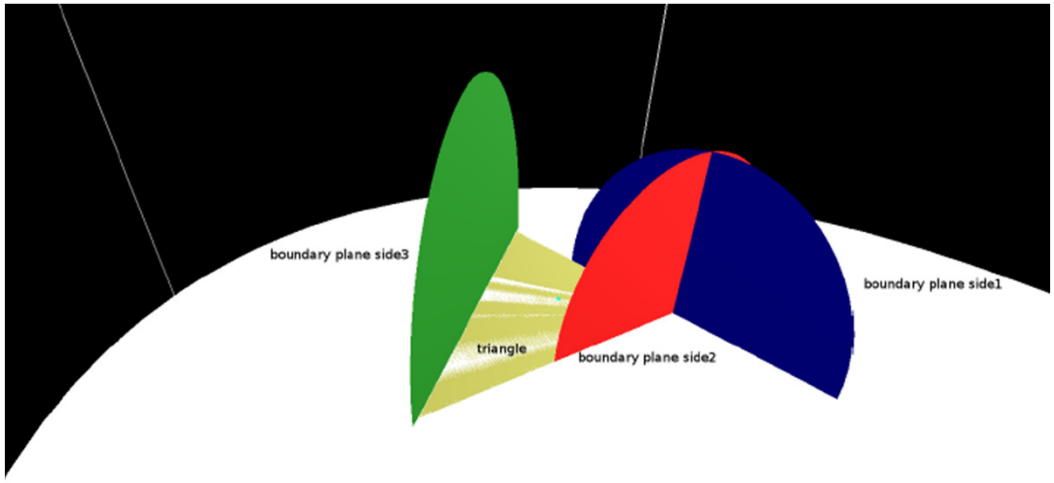


Gaalop GPC for C++ …

TECHNISCHE UNIVERSITÄT DARMSTADT

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

- Application example: Collision Detection …



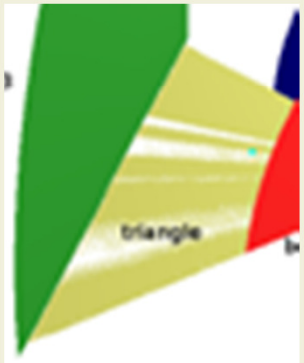
Point-Triangle-Test:

Point-Triangle-Test

TECHNISCHE UNIVERSITÄT DARMSTADT

1. Define the three "triangle points" (the vertices of the triangle) and the test point based on the corresponding C++ variables (t1x, t1y, ...).

```
bool pointTriangleTest(const float t1x, const float t1y,
const float t1z, const float t2x, const float t2y, const float t2z,
const float t3x, const float t3y, const float t3z, const float px,
const float py, const float pz, const float h) {

  #pragma gpc begin
    #pragma clucalc begin
      TrianglePoint1 = VecN3(t1x, t1y, t1z);
      TrianglePoint2 = VecN3(t2x, t2y, t2z);
      TrianglePoint3 = VecN3(t3x, t3y, t3z);
      TestPoint = VecN3(px, py, pz);
```



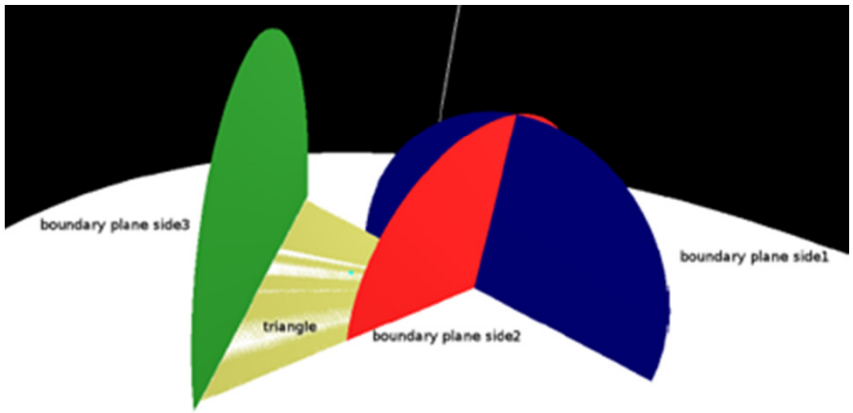
Point-Triangle-Test

TECHNISCHE UNIVERSITÄT DARMSTADT

2. Construct the base plane based on the triangle points (see Sect. 3.1.3).

3. Compute the signed distance d between the base plane and the test point based on the inner product (see Sect. 3.4.1).

4. Compute the normal vector to the plane.




```
// construct base plane
plane=*(TrianglePoint1 ^ TrianglePoint2 ^ TrianglePoint3 ^ einf);
// compute signed distance of TestPoint to base plane
?d = plane . TestPoint;
// extract triangle normal
?normal_ = plane - (plane . e0) ^ einf;
```

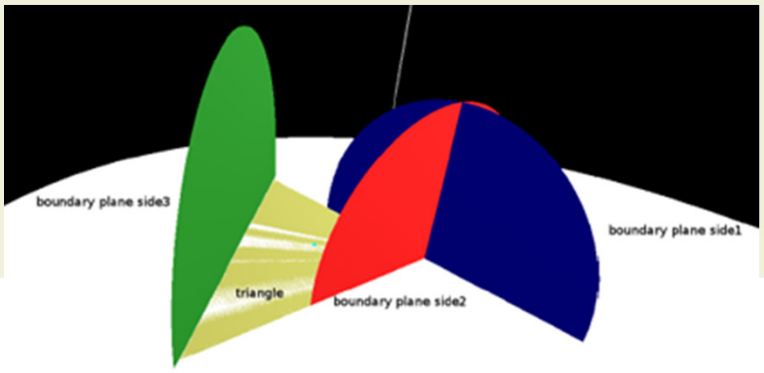
Point-Triangle-Test

TECHNISCHE UNIVERSITÄT DARMSTADT

5. Using this normal vector and the triangle points, compute the boundary planes, for example, the planes that are perpendicular to the base plane and pass through every combination of the three points.

6. Compute the signed distances d_1, d_2, and d_3 between the test point and the three boundary planes and assign them to C++ variables.

```
// construct boundary planes
side1 = *(TrianglePoint1 ^ TrianglePoint2 ^ normal_ ^ einf);
side2 = *(TrianglePoint2 ^ TrianglePoint3 ^ normal_ ^ einf);
side3 = *(TrianglePoint3 ^ TrianglePoint1 ^ normal_ ^ einf);

// compute distances
?d1 = side1 . TestPoint;
?d2 = side2 . TestPoint;
?d3 = side3 . TestPoint;
#pragma clucalc end
```



Point-Triangle-Test

TECHNISCHE UNIVERSITÄT DARMSTADT

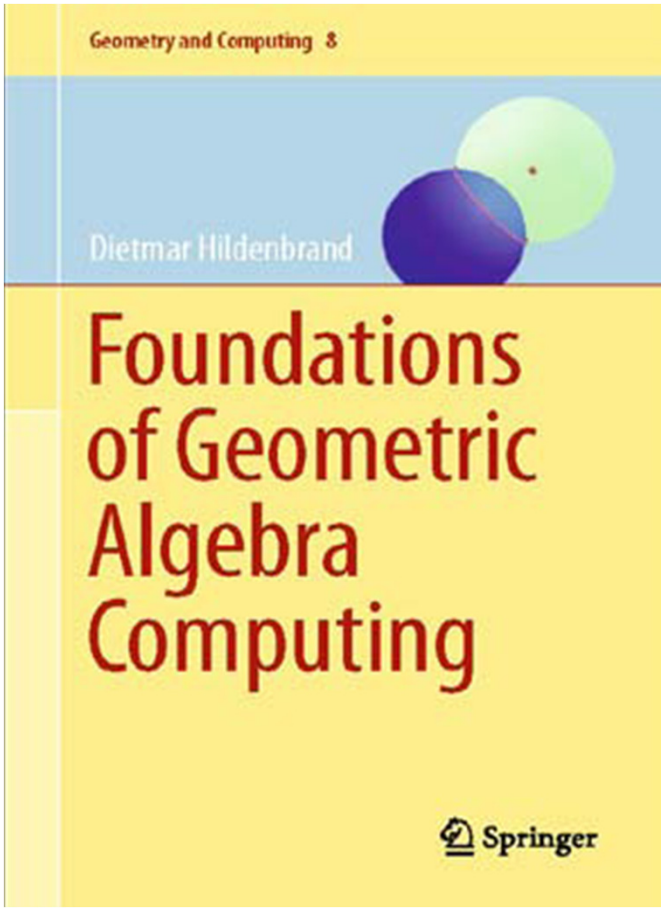
```
    const float d_SCALAR = mv_get_bladecoeff(d,1);
    const float d1_SCALAR = mv_get_bladecoeff(d1,1);
    const float d2_SCALAR = mv_get_bladecoeff(d2,1);
    const float d3_SCALAR = mv_get_bladecoeff(d3,1);
  #pragma gpc end

  if (d_SCALAR * d_SCALAR > h * h)
        return false;
  if (d1_SCALAR <= 0.0f && d2_SCALAR <= 0.0f
                && d3_SCALAR <= 0.0f || d1_SCALAR >= 0.0f
                && d2_SCALAR >= 0.0f && d3_SCALAR >= 0.0f)
        return true;
  return false;
}
```

7. The condition for a collision is satisfied if d^2 is less than or equal to the square of the thickness h of the triangle, and all signed distances d_1, d_2, and d_3 have the same sign.

Reference

TECHNISCHE UNIVERSITÄT DARMSTADT

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



Thanks a lot

Google Dietmar Hildenbrand