Tracking Objects on a Deformable Surface using Displacement and Orientation Information

Zachary DeStefano[*1], Kyle Cutler[†1], Gopi Meenakshisundaram[‡1], and Bruce Tromberg[§1]

[1]University of California, Irvine

1 Mesh Traversal Algorithms

1.1 Following a Deformable mesh

Even after calibration, due to measurement error and deformability, we have a path that mostly follows the surface but there will be segments that end up lying above or below the surface. The path was still recorded on the surface and we thus want to have a visualization of where the path was located on the surface itself. We thus need a method of converting the inaccurate path in the virtual world into a path that follows the surface and shows the locations on the surface that the probe traversed.

A path is just a series of connected segments and the mesh is just a series of triangles. Thus the input of our algorithm is a segment with a start and end point as well as the triangle where it originates and the triangles neighbors. We need to make sure the projection preserves the length of the segment. We are restricted to rotating the segment along the plane that it makes with the triangle's normal.

If we let v be the segment vector, N be the normal, and E be the resultant vector that is the projection of v onto the plane described by N, then the following equation will get us E

$$E = v - proj_N(v)$$

This can be furthur simplified to say

$$E = v - N(v.N)$$

INSERT FIGURE OF THE VECTORS We then find the intersection of the vector E with the current triangle. If the segment is entirely in the triangle, then we move onto the next segment starting from the endpoint of E. If the segment leaves the triangle, then we find which edge the segment intersects and repeat the projection procedure for the end part of the segment that is not in the triangle. This procedure of course relied on finding the intersection of the triangle with the segment which proved to be non-trivial.

When finding the triangle and segment intersection, we had a triangle in a 3D space as well as a segment that was supposed to be coplanar to the triangle but could be slightly off due to floating-point errors with the coordinates. To find the intersection point, I converted the segment and triangle coordinates to the coordinate system where one of the vertices of the triangle is the origin and the basis vectors are the two vectors made by the segments coming off that vertex as well as the cross product of those vectors. In this coordinate system, the third coordinate should be zero. In practice, it was near zero due to floating point errors. Once in the new coordinate system, we just had to find the 2D intersection of the triangle and the segment.

The above procedure describes what we did with a single segment. With paths, we just iterated this procedure for each segment of the path. We assumed that the important aspect of each segment is its vector and not its origin point, since that is what we get from the probe. This meant that the end point of a projected segment was treated as the start point for the rest of the path when doing the loop to project the entire path onto the mesh.

1.2 Justification

In Theorem 9.10.11 of **??**, the geodesic is defined to be the curve of shortest length between two points on the surface.

References

[*]zdestefa@uci.edu
[†]kbcutler58@gmail.com
[‡]gopi.meenakshisundaram@gmail.com
[§]bjtrombe@uci.edu