Tracking Objects on a Deformable Surface using displacement and orientation data

Zachary DeStefano, Kyle Culter, Gopi Meenakshisundaram, Bruce Tromberg
University of California, Irvine

August 18, 2014

Abstract

Previous attempts at tracking objects as they move have used magnetic fields or a camera array. With deformable surfaces trying to track on them is especially difficult because the path is not directly following the original shape for the mesh. For this paper, we present an algorithm for tracking on a deformable surface as well as a way of tracking using only displacement and orientation sensors. We developed a probe that contained an optical mouse sensor for displacement on the surface and an accelerometer, gyroscope, and compass to find out the orientation of each displacement. We used this data to record paths and developed an algorithm for doing calibrations as well as following the surface.

Introduction

In various medical applications, there is a need to track a probe as it moves across a patient's body and takes measurements. Currently they attach a grid to the patient and painstakingly take measurements at each point in the grid. This led us to explore ways of automating that process. Using a Kinect to try and track probe movement can be impractical because the probe needs to always be in view of the Kinect camera. Magnetic grids are expensive and cumbersome so that option would not have worked well. We then thought that using an ordinary optical mouse sensor on the surface itself could work well. Because the surface is not flat, we will need to know the orientation so we put a gyroscope, compass, and accelerometer into the probe. Using these simple tools as well as software that implemented the calibration algorithm shown in this paper, we were able to get a reasonable approximation of the probe's path on the surface itself.

Our process started with taking a 3D scan of the surface to get the mesh. We use a Kinect to get the data and the point cloud architecture **INSERT MORE DETAIL** to take the data and get a mesh and texture from it. We then take out some of the noise and use the QSlim **INSERT MORE DETAIL** algorithm to simplify the mesh. Once the mesh is simplified, we load it into our tracking environment. Our tracking environment is an application that takes in the mesh as well as probe data readings and gives live updates as to the probe's position and data. It operates similarly to a video game and was developed using JMonkeyEngine **INSERT DETAIL**, a video game library written in Java that is similar to Ogre **INSERT REFERENCE**.

The first step is to make sure the scale is calibrated. This is done in two steps. We move the probe a certain distance and see what the total displacement outputted by the mouse sensor was. We then measure the distance between known points on the surface in the virtual world and the real world and see what the ratio is. We then combine the two ratios so that we have a scale factor to go from displacement units read by the probe to displacement units in the virtual world. Once the scale is calibrated, we start recording paths on the mesh itself. We then have a problem. Because the rotation has not been calibrated, it is likely that the paths will have a shape similar to the part of the mesh they were on but they will not actually be on the mesh **INSERT PICTURE OF THIS**. We thus need to figure out what rotation to apply to the probe reading's rotation in order to get the rotation in the virtual world. After that though, there is still a problem. Because

the surface is deformable, the path would still not perfectly follow the mesh. We thus need an algorithm that will approximate the probe's position on the mesh given the deformability.

This paper presents two algorithms. The first one takes a path that almost follows the mesh and projects it onto the mesh in such a way that preserves its orientation along the mesh as well as its arc length. The second one takes a recorded 3D path between two known points on the mesh and figures out how to rotate it and project it onto the mesh so that its start and end points match the ones on the mesh. The rotation found by the second algorithm gives us the rotation calibration we need. Once the rotation is calibration is found, we can then live track the probe and use the first algorithm to keep it along the mesh.

Related Work

Commercial motion tracking for professional film makers use image-based tracking techniques [1]. This method has some serious disadvantages for our application. Being that is designed for use by artists, it does not have a guarantee of measurement accuracy, which is necessary for our case. Additionally, the fact that it is image based means the probe would always have to be in the view of the camera and in a clinical setting that can be hard to ensure. The post-processing of the images can be time consuming and in a clinical setting the results should come quickly. Finally, training the medical technicians taking the data to do all the steps that would be required is impractical.

Using the Kinect for tracking would have some advantages over image-based tracking. Because the Kinect gives us a depth map, we could more easily ensure measurement accuracy. Additionally, there would be much less processing required to get a final motion path. It still has the disadvantage that the probe would always have to be in view of the camera. It also has the disadvantage that the result is dependent on the shape of the probe. For this application, we do not know the shape of the final probe and it is impractical to try and accommodate all the different probe shapes.

Motion Capture is a popular technique for film making that tracks a character's motion in 3D space. Using a motion capture system we would get 3D coordinates for the probe's position as it moves across a patient. Current motion capture systems however are expensive and cumbersome to set up. Both of these make it impractical for the clinical setting. **INSERT REFERENCE**

There is work being done on using magnets for motion tracking [4]. While we could get measurement accuracy with this method and the setup would not be too difficult, the system would be cumbersome for patients. It would also only be limited to tracking across an area as big as the magnetic coils. The method we are proposing can be used on any mesh that is scanned in 3D. Additionally, we want to easily be able to track across the entire mesh that was scanned.

Current work on deformable surface tracking is focused on tracking the deformations of the surface itself [3, 2]. This paper focuses on deformable surfaces that deform when pressure is applied but then retain

their shape afterwards. It is also focused on tracking objects on the surface rather than the surface itself.

Being that we are tracking on a 2D surface embedded into 3D space, we thought about trying to flatten the mesh first and then track on it. We found the following paper that described a mesh flattening algorithm. **INSERT CITATION**. Even though the mesh flattening does not have any holes or gaps when using this algorithm, there is no way to assure that our path will not leave the boundary of the flattened mesh. Additionally, we need an algorithm to work on arbitrary meshes and the mesh flattening was only proven to work with a handful of meshes. Due to these difficulties, we decided not to flatten the entire mesh and only do local flattening when we want the path to follow the mesh, which means we care about the triangle where the path is currently as well as its neighboring triangles.

Following a Deformable mesh

For the purposes of our application, we need to have paths on the mesh itself so that we know the location on the mesh of particular data points. Due to the deformability, the paths ends up being close to the mesh, but not on the mesh itself, no matter how accurate the calibration was. Additionally, in order to do the calibration, we need the path to follow the mesh itself so that the end points line up. We thus came up with an algorithm for projecting a path onto the triangles of the mesh.

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 also want to preserve the orientation along the surface thus 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

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

Calibration of Rotation

Experimental Results

Conclusion and Future Work

Acknowledgments

References

[1] Adobe Systems Incorporated. Tracking and stabilizing motion.

[2] Salzmann M. Hartley R. Fua P. Convex optimization for deformable surface 3-d tracking.

[3] Schulman J. Lee A. Ho J. Abbeel P. Tracking deformable objects with point clouds. *ICRA 2013*.

[4] Huang J. Takashima K. Hashi S. Kitamura Y. Im3d: Magnetic motion tracking system for dextrous 3d interactions.

Appendix