Delaunay Tetrahedralization in a 3-D Free-Lagrangian Multimaterial Code

Manjit S. Sahota

Computational Physics Group
Applied Theoretical Physics Division
Los Alamos National Laboratory
Los Alamos, New Mexico

Abstract: A Delaunay tetrahedralization technique for tesselating space, without any multimaterial tetrahedra for a given mass-point distribution, in a three-dimensional, multimaterial, free-Lagrangian code is described. The approach involves first connecting only the points that do not result in degeneracies, and adding the degenerate points later on, so that the degeneracies affect the mesh only locally. First, the entire Delaunay tetrahedral mesh is generated disregarding multiple materials. The multimaterial tetrahedral connections are then broken by adding new points at the multimaterial connections. The technique inherently involves $O(n^2)$ arithmetic operations, where $n$ is the number of mass points; however, we have reduced it to $O[n \log(n)]$ operations by utilizing a “binning” approach. The algorithm is fully vectorized on the Cray family of supercomputers.

1. Introduction

A good tesselation of the underlying grid is required for free-Lagrangian codes that use fixed-mass particles. We have implemented a robust Delaunay tetrahedralization algorithm that can tesselate space for any mass-point distribution. The approach involves tetrahedralizing the points by identifying their nearest neighbors by constructing Voronoi diagrams. Such a tesselation is defined as follows: Consider a random distribution of $n$ points in three-dimensional space. Select any four noncoplanar points and construct a tetrahedron. Draw the sphere circumscribing the tetrahedron. This sphere will be called the circumball of the tetrahedron. This tetrahedron is called a Delaunay tetrahedron if and only if no other $n - 4$ points lie within its circumball. Such a grid tesselating the desired convex space is called a Delaunay grid; and is unique for a given point distribution, barring degeneracies. The Delaunay tesselation is easy to accomplish in two dimensions; however, it is
much more difficult to achieve in three dimensions because of degenerate point distributions arising as a result of their usually orderly locations. These degeneracies result in nonunique meshes and even in nonunique number of tetrahedra.

Our first attempt of constructing Delaunay tetrahedra involved selecting a point in the mesh and expanding the search process, point at a time, in all directions. The process was not always successful because connecting up the degenerate points before completing the mesh affected the mesh globally, which led to nonunique meshes and resulted in wafer-thin tetrahedra (tetrahedra whose all four points were coplanar within machine precision). Field and Field and Yarnall implemented an algorithm developed by Watson in two and three dimensions by Delaunay tetrahedralization of points using a one-at-a-time strategy. Our second attempt was to try Field and Yarnall’s approach, which turned out to be very successful. We have since refined their approach to handle generacies and multimaterial connections in very robust manners. We also have reduced the \( O(n^2) \) arithmetic operations to \( O[n\log(n)] \) operations by utilizing a “binning” approach.

Section 2 delineates the details of the technique. Major conclusions are presented in Section 3.

2. Description of the Algorithm

The Delaunay mesh generator uses one-at-a-time insertion strategy, whereby, to add a new point to the existing mesh, some existing tetrahedra are removed and new tetrahedra are created. The algorithm is described by the following steps. Although the algorithm is three-dimensional, all the figures are drawn in two dimensions for simplicity.

1. Draw a tetrahedron enclosing all points (Fig. 1). This will be called the enclosing tetrahedron.
2. Select any point at random (point number 1 in the problem) and subdivide the enclosing tetrahedron into four tetrahedra by connecting the mass point to the four vertices of the enclosing tetrahedron (Fig. 2).
3. Draw circumballs associated with the four tetrahedra (Fig. 3). Maintain a master list of tetrahedra and their circumballs.
4. Pick the next point in the list and make a list of tetrahedra whose circumballs contain this point. If the point lies within an epsilon distance from any circumball, add this point to a separate list called the “fail” list, and return to the start of this step.
5. The union of all tetrahedra selected in Step 4 gives a polyhedron. This is called an “insertion polyhedron.” Delete all tetrahedra from inside this polyhedron and create new tetrahedra by connecting the point to the vertices of the surface triangles of the insertion polyhedron (Fig. 4). If the volume of any of the new tetrahedra is too small, or if the sum of the volumes of new tetrahedra is not equal to the volume of the polyhedron, add the point to the fail list and return to Step 4. Otherwise compute new circumballs for new tetrahedra (Fig. 5) and update the tetrahedron master list to reflect the change.