Points inside a Mesh Project

Objective

The goals of this project are:

- Implement a simple algorithm to determine which points within a set are inside or outside of a given mesh.
- Design and implement a 3D user interface to visualize the mesh and points.
- Write a “flashlight” shader to visualize proximity of points to the mesh surface.

Input

The following files contain the input data:

- points.txt: This contains the point cloud. Each line represents the x,y,z coordinates of a single point.
- concave_shape_ascii.stl: A manifold mesh which is not a convex hull. It contains dips and bumps.

The mesh is available in the STL (stereolithography) file format, which is documented in Wikipedia.

There are several methods for solving this problem, such as the ray/triangle intersection method. You are free to use any algorithm of your choice.

Requirements

Your program should meet the following requirements:

- Load the mesh and points data into appropriately designed classes.
- Implement an algorithm for determining inside and outside points.
- The program should visually distinguish inside and outside points in the 3D interface.
- Ability to select one point at a time in 3D. When a point, P, is selected, draw a patch on the mesh surface centered around the mesh surface point, Q, which is nearest to the selected point. The approximate radius of the patch will be inversely proportional to the length |PQ|. In other words, the patch is larger for points which are closer. Past a defined distance threshold, the patch may not appear.
- The code must be clean, modular, maintainable, and fully object-oriented.
- If time permits, make the rendering of the points and mesh visually appealing.

Deliverables

There are two outputs to this project:

1. **Working Code:** Provide C++ source code and a working binary which implements the requirements.
2. **Design Description:** A brief written description of the application design. The design description document is where you can elaborate on your implementation, suggest alternate algorithms, and explain how you verified your algorithm.

Your solution will be judged both for the quality of your implementation and your ability to clearly communicate.

At your follow-up interview, you will present your solution and your design description. Please don’t hesitate to ask us questions or engage in discussion along the way, since that too is part of the process.