CSCI 4972 Introduction to Visualization
Assignment #5 - Spatial Data Structures
Due: Tuesday October 5th at 11:59PM

There is no “one-size-fits-all” spatial data structure. Oftentimes experimentation is necessary to determine the most memory-efficient and/or computation-efficient data structure for a specific operation on a typical dataset for a target application. VTK provides the implementation of four different 3D spatial data structures:

- KDTree
- Octree
- OBB (Oriented Bounding Box) Tree
- Modified BSP (Binary Space Partition) Tree

We will be testing these data structures for the memory and time performance against two common classes of geometric operations:

- Closest Point Query (a.k.a. Nearest Neighbor) - Given a “point cloud” dataset and a query point, find:
  - the closest point in the dataset to the query point
  - the set of all points within a radius *r* of the query point
  - the set of *k* closest points to the query point
- Line-Polygon Intersection - Given a triangle mesh and and a line/ray/line segment, find:
  - the set of *all* intersections between the line and the triangles in the dataset
  - the *first* intersection (closest to the ray origin) along the ray with the dataset
  - *a boolean* indicating whether the line segment intersects *any* triangle in the dataset

Theoretically, all four of these data structures can be used to efficiently perform the above operations. However, some are more straightforward than others, which is reflected in the VTK implementation interface summarized below:

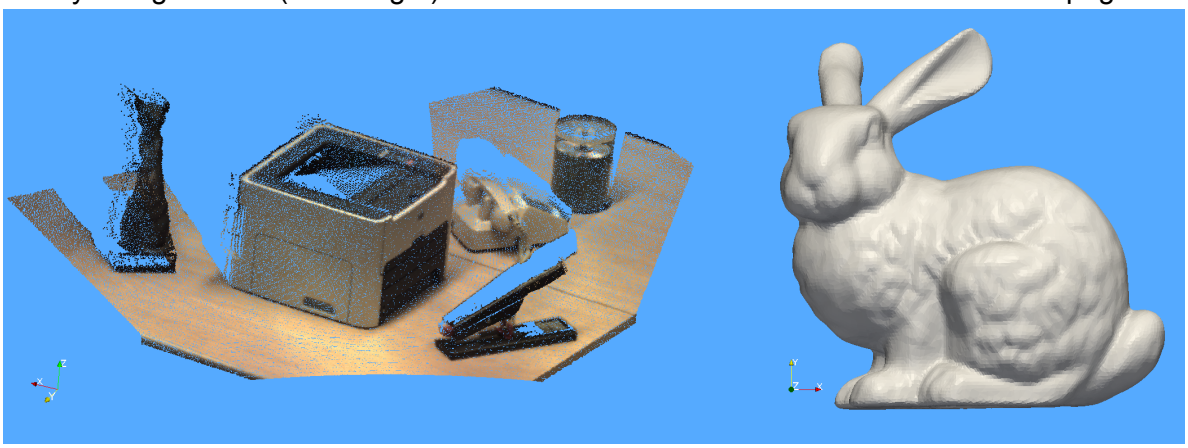
	Closest Point	Line Intersection
KD Tree	yes	no
Octree	yes	no
OBB Tree	no	yes
Modified BSP Tree	no	yes

Examples of how to use the data structures can be found here:

- KDTree closest point query: http://www.vtk.org/Wiki/VTK/Examples/Cxx/DataStructures/KdTreePointLocator/ClosestPoint
- Octree closest point query: http://www.vtk.org/Wiki/VTK/Examples/Cxx/DataStructures/Octree/ClosestPoint
- OBB Tree line intersection: http://www.vtk.org/Wiki/VTK/Examples/Cxx/DataStructures/OBBTree_IntersectWithLine
- Modified BSP Tree line intersection:

http://www.vtk.org/Wiki/VTK/Examples/Cxx/DataStructures/ModifiedBSPTree_IntersectWithLine

To test the Closest Point Query operation, we will start with a point cloud range scan of a desk (below left). And to test the Line-Polygon intersection operation, we will start with the Stanford Bunny triangle mesh (below right). Both datasets are available on the course webpage.



For this assignment the class will divide into four equal-sized teams and each team will be assigned one of the four data structures listed above. Each team is responsible for producing an implementation of the above described operations, and test them on the provided data sets. Concentrate on the implementation and testing of the operations that are straightforward for your assigned data structure. Once familiar with interface for your data structure you may *optionally* tackle the other set of operations, *for extra credit*. Since run times will vary from computer to computer, a "naive" implementation of each algorithm that does not use any spatial data structure should be written and tested as a baseline. For example, the naive closest point query will loop over *all* of the input points comparing the distance to the query point.

Each team should find or construct new datasets *and* specific queries (query point or query line/ray/line segment) that they hypothesize will leverage the strengths of their assigned data structure and perhaps identify weaknesses in the other data structures. These "adversarial" data sets should be posted (in standard VTK format) on LMS by Monday Oct 4th @ 11:59pm to allow the other teams to test their code on these datasets.

Each team should thoroughly test and analyze the performance of their data structure for the target operations. Each team should produce plots of performance (run time) over variable settings of a few parameters. Possible parameters include:

- Approximate number of leaf nodes/number of subdivisions in the tree
- The number of points in the data set
- The "volume" of the points in the data set. That is, if you imagine a uniform, volumetric grid over the data set bounds, what is the ratio of occupied cells to unoccupied cells?

Secondary Learning Outcomes

1. Understand how to write code in such a way that it is easy to vary parameters to produce these type of variable parameter analyses.
2. Work as a team and learn to use a revision control system. This is an absolutely critical

skill for working in professional teams. We suggest using git and hosting the repository at either github.com or gitorious.org.

Grading Criteria

(5 pts) Implementation of the two operations with your assigned data structure
(5 pts) Selection/construction of an adversarial dataset that demonstrates the strengths of your assigned dataset and brief explanation.
(5 pts) Thorough testing and analysis of your assigned data structure on all datasets (the provided datasets, your new datasets, and the datasets of the other teams)
(5 pts) Teamwork and division of labor

Submission Requirements

1. Submit your adversarial datasets to LMS by Monday @ 11:59pm.
2. Submit your implementation code to the homework server by Tuesday @ 11:59pm.
3. Submit a document (either plaintext writeup + images *OR* .pdf format) summarizing the performance of your dataset on the provided test cases and all adversarial datasets posted on LMS by Tuesday @ 11:59pm.
4. Present your conclusions to the class in a short (10 minute) presentation on Wednesday.