RIT Department of Computer Science
Thesis Pre-Proposal
Real-Time Continuous Surface Modeling of Geospatial Point Clouds with User-Selectable Entities

Student: Stephen Ranger, Advisor: Professor Reynold Bailey

Problem

Massive Point Clouds are created by scanning objects and environments using a number of technologies such as radar and lasers. However, with the detail they provide, hardware is hard pressed to display these data sets with granularity and usability in mind. There has been work done in this field with efficiencies made in rendering point clouds using pre-defined mesh structures and GPU algorithms for defining shapes. However, the issues related to these implementations don't take into account a geospatial, real-world component. I propose applying a scalable real-time filtering technique —used to do depth occlusion and surface modeling—to a context-aware selection technique in a point cloud with a fairly uniform point density. This algorithm will be built within a real-time system that will use a three-dimensional partitioning system and data server for on-demand access to sensor data.

This system will allow the user to accurately view geospatial point cloud data on a global scale and the ability to view surface information and select objects for exporting or highlighting in order to more accurately analyze these massive data sets. Currently, viewing and selecting objects is slow and inaccurate; coupled with the issues a geospatial projection adds to the mix, it can be greatly improved.

Previously, quadtree, octree [3] and kd-tree partitioning systems have been applied to a Cartesian system in order to add on-demand searching for view-dependent slices of data. However, once the data set is converted into a geospatial positioning system, either the Cartesian coordinates are now not surface aligned, or issues arise along the partitioning system boundaries where values are not continuous (such as with latitude 90 != latitude -90 but longitude 180 == longitude -180). Using a Cartesian projection from geospatial coordinates solves this issue but adds complexity when deciding what to render as it is no longer surface aligned. I will look into applying a surface-aligned data structure to a global data set such as a tetrahedral mesh. By using this structure to search for nodes within my view frustum I will be able to render progressively deeper nodes as the visualization's viewpoint moves closer or further away from the target and as the level of detail increases and pull these nodes from an on-demand point server.

Also, in recent work, real-time rendering of depth culling and surface representation [1] has been used to hide unseen data points as well as to fill surface information in on a massive unstructured point cloud. I propose to use this information and apply it to a sparse context-aware selection algorithm [2] in order to adapt it to a more uniformly dense data set. I believe that this on-the-fly surface creation can be used to augment the context-aware selection algorithm in order to give it another avenue for object separation within a uniformly-dense data set.

Methodology

I will be applying a depth buffer culling and surface generation algorithm to augment a sparse context-aware selection model so that analysis of massive point cloud data sets can be done in real time. I will be developing the model that will augment the selection algorithm by leveraging the depth and surface

data to separate objects for selection and allow a user to use the system in real-time. I will compare the output of this augmented algorithm with the original algorithm and compare the accuracy of object selection between the two. I will also be defining an on-demand point server for use with my graphical visualization and using the resulting selection to request more detailed point data where available. This will allow the user to view specific objects quickly and do analysis on the data set more easily.

Evaluation

In order to evaluate my designs, I will be developing an on-demand point server that will accept incoming requests from a client system for portions of the massive point cloud data set. The partitioning system used for the data will initially be a Cartesian octree, storing a random subset of n-points per octet and passing down the remainder to each of its eight child nodes. The system will then render each parent until a specified level of detail is reached. The visualization client will be written in Java and OpenGL (jogamp) using math and graphics libraries written by the author throughout his academic career. This visualization will display a portion of the point cloud by using frustum culling on the octree itself and applying the augmented depth culling surface selection algorithm on the visible point data.

In order to measure the success of the algorithm, I will be comparing the accuracy of the original selection algorithm with my augmented algorithm. The accuracy will be measured by comparing the output to manually selected objects in the scene for accuracy of what points are selected.

Evaluation Outcomes

I will measure success or failure when I feel that the augmented algorithm has been implemented to the best of my ability and it either succeeds or fails in surpassing the original algorithms accuracy. The possible outcomes of this proposal are that my augmented algorithm works as intended and exceeds the original, my algorithm does not exceed the original, or my algorithm is much slower as to not be realistic in a real-time scenario.

Deliverables

Deliverables will include a final report detailing the outcome of the algorithm developed along with any source code developed when testing its validity. The report will also include examples and comparisons with the baseline algorithm. Documentation for the source code will also be provided.

Schedule

2016-01-25: Initial website posted
2016-02-15: Massive Point Cloud server complete
2016-03-01: 3D Visualization complete
2016-03-15: Original algorithms implemented
2016-04-01: Early algorithm implemented (first draft)
2016-04-15: Intermediate algorithm implemented
2016-05-01: Algorithm compelte

2016-05-10: Technical Report Complete
2016-05-16: Tentative Defense Date

Current Status

The current status of my work is that I have done some research looking into previous research in the area as well as much of my own work as a full-time software developer on visualizing massive point clouds. I have also begun to write math and graphics libraries that I have been using over the past few years in academia.

References

[1] Ruggero Pintus, Enrico Gobbetti, and Marco Agus. 2011. Real-time rendering of massive unstructured raw point clouds using screen-space operators. In *Proceedings of the 12th International conference on Virtual Reality, Archaeology and Cultural Heritage* (VAST'11), Matteo Dellepiane, Sebastian Pena Serna, Holly Rushmeier, Luc Van Gool, and Franco Niccolucci (Eds.). Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 105-112. DOI=http://dx.doi.org/10.2312/VAST/VAST11/105-112

[2] Lingyun Yu, Konstantinos Efstathiou, Petra Isenberg, Tobias Isenberg. CAST: Effective and Efficient User Interaction for Context-Aware Selection in 3D Particle Clouds. *IEEE Transactions on Visualization and Computer Graphics*, Institute of Electrical and Electronics Engineers, 2016, 22 (1), pp.886-895.<10.1109/TVCG.2015.2467202>. <hal-01178051>

[3] Wenzel K, Rothermel M, Fritsch D, Haala N. An out-of-core octree for massive point cloud processing. In: Workshop on processing large geospatial data Cardiff, UK, July 8th, 2014, {http://rs.tudelft.nl/~rlindenbergh/workshop/WenzelIQmulus.pdf}; 2014.