A Dynamic LOD Point Clouds Visualization System Using Octree-based Voxels

Togo Okubo
Department of Information Media
Hokkaido Information University
Ebetsu, Japan
s2581103@s.do-johodai.ac.jp

Masahiko Itoh
Department of Information Media
Hokkaido Information University
Ebetsu, Japan
imash@do-johodai.ac.jp

***Abstract*—Point cloud data contains enormous amounts of data. Another issue is that reducing the data degrades visibility during visualization. One solution to this issue is to use voxel representation to visualize point cloud data. In particular, by varying the voxel level of detail (LOD) depending on the level of attention to an object, it is possible to efficiently reduce data volume while maintaining visibility. In this study, we assume that areas closer to the viewpoint attract more attention, and propose a visualization method that dynamically changes voxel size depending on the distance from the viewpoint (using smaller voxels at closer distances). This method is implemented using an octree structure, and its effectiveness is verified through experiments visualizing actual snow sculpture data in 3D space.**

***Index Terms*—Point Clouds, Voxel, Octree, Visualization**

I. Introduction

In recent years, the availability of usable 3D point cloud data has increased significantly due to improvements in consumer 3D scanning technology and the progress of open data policies by local governments. The widespread use of this data has also promoted the development of applications that utilize point cloud data [1].

Point cloud data generally has a huge amount of data, and its reading and rendering process requires a huge computational cost. In addition, when visualizing, if the distance between the viewpoint and the object is not properly maintained, it becomes difficult to accurately see the shape of the object.

If the point rendering size is small, when the viewpoint is close to the object, the density between points decreases, causing gaps and making it hard to see the surface shape. On the other hand, if the rendering size is large, points overlap excessively when observed from a long distance, making the detailed shape unclear. In addition, because it is difficult to distinguish visual details from a long distance, rendering the entire point cloud is unnecessary from a computational resource perspective.

In response to the above challenges, this study focuses on voxel representation as a method for visualizing point cloud data. By using multi-resolution voxels, which gradually change resolution depending on the viewpoint distance, it is possible to reduce the amount of data while maintaining the visibility of the object. Specifically, we target point cloud data of snow sculptures from the Sapporo Snow Festival, convert them into voxel data with a hierarchical structure using an octree, and visualize them using cubes. We demonstrate that this method makes it possible to clearly view the original three-dimensional shape even when the amount of data is reduced.

To verify the validity of this hypothesis, we develop a visualization system that applies the above-mentioned method. Note that this system is designed to run on a web browser, and uses the WebGL library Three.js for its implementation.

II. Related Research

Various methods have been proposed for voxelizing point cloud data [2]. Representative voxel-based representation methods include supervoxels and boundary refined voxel structures, and the use of voxels with multiple resolutions. Among these, methods using voxels with multiple resolutions utilize an octree structure to adaptively divide space according to the geometric complexity of the object surface. This allows simple regions such as flat surfaces to be efficiently represented by a single large voxel. On the other hand, in regions that include complex shapes such as corners, the voxels are repeatedly subdivided until the internal points can be considered to be a "simple distribution close to a plane."

Martinez-Rubi et al [3]. proposed an algorithm for constructing a multi-resolution data structure for large-scale point cloud data. In this method, each point is assigned to an appropriate Octree node within a user-specified maximum depth range based on the distance between points. This process is repeated until all points are stored in a node or the Octree's maximum depth is reached. Points exceeding the maximum depth are discarded from the data structure and not stored. In visualization, LOD control is achieved by dynamically selecting the Octree reference hierarchy according to the distance from the viewpoint. This allows for high-resolution rendering at close range and low-resolution rendering at long range.

In this study, we focus on point cloud data with a hierarchical LOD structure based on Octree. For visualization, we use cubic voxels as the drawing unit instead of conventional point primitives. Furthermore, we propose a drawing method that dynamically controls the level of detail depending on the distance between the viewpoint and the object.

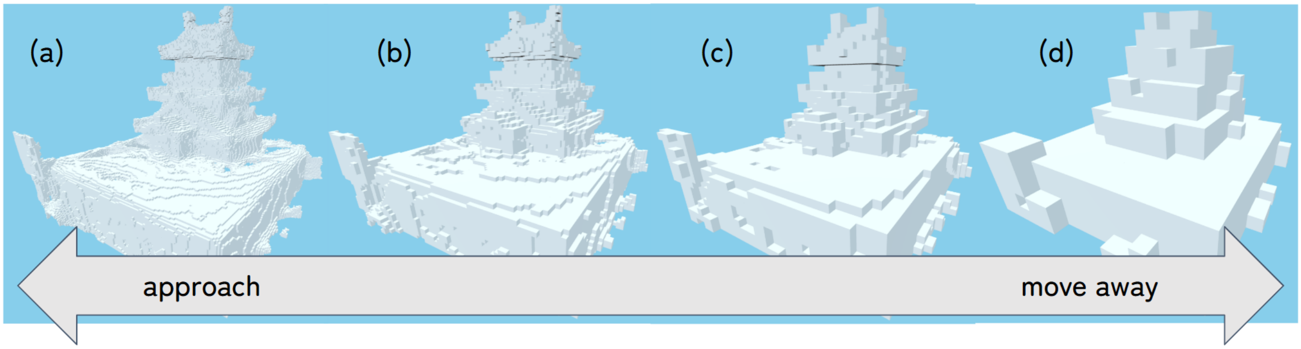

Fig. 1. Point cloud data was converted into voxel data with multiple levels of detail.

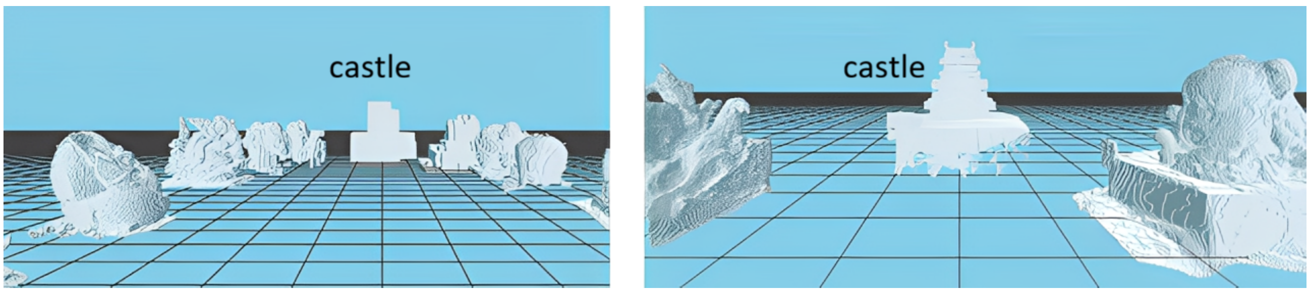

Fig. 2. Left: Object seen from a distance Right: Object seen from up close

III. Proposed Method

A. About the data

This study uses snow sculpture data from the Sapporo Snow Festival held in 2023 and 2024 as the experimental subject. While LiDAR and photogrammetry are common methods for acquiring 3D shapes, this study employed Luma AI[1], which is based on technologies such as NeRF (Neural Radiance Fields) and 3D Gaussian Splatting, to perform 3D reconstruction and generate point cloud data. Specifically, the spatial coordinates, normal vectors, and color information (RGB) of each point were extracted from the 3D model generated by Luma AI, and this was used as the point cloud data for the experiment. To prevent missing point clouds or reduced accuracy due to a lack of input images, a large number of image sets were used to acquire the data, capturing the object from multiple angles and comprehensively.

B. Convert point cloud data to voxel data

Figure 1 shows an example of an application in which point cloud data (Figure 3) obtained from snow sculptures (castles, etc.) at the Sapporo Snow Festival was converted into multi-resolution voxel data. The left side of the figure 1 shows a low-resolution model assuming a long-distance perspective, and the right side shows a high-resolution model assuming a close-up perspective. The Open3D library was used to implement the data conversion process. Octree was used as the voxelization algorithm. The specific procedure is to first define a bounding box that contains all point clouds in 3D space. Next, this cube is recursively divided into eight parts, and the center coordinates of the cubes at each level are extracted and saved as voxel data. By using this hierarchical structure, a visualization model including LOD was constructed (Figure 4).

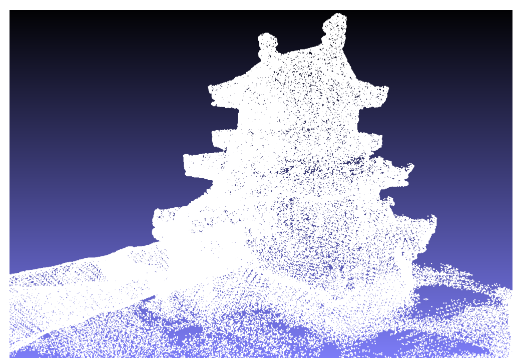

Fig. 3. Point cloud data of the castle snow sculpture

As a result of applying this method, the file size of the data shown in Figure 3 was reduced from approximately 19,000 KB before processing to approximately 9,000 KB. This is equivalent to less than 50% of the original data, demonstrating a significant reduction in data volume.

C. LOD Control based on Viewing Distance

This system uses an interface that allows users to set an arbitrary threshold for the viewpoint distance, which is the criteria for switching between LODs. The rendering system

[1] https://lumalabs.ai/dream-machine

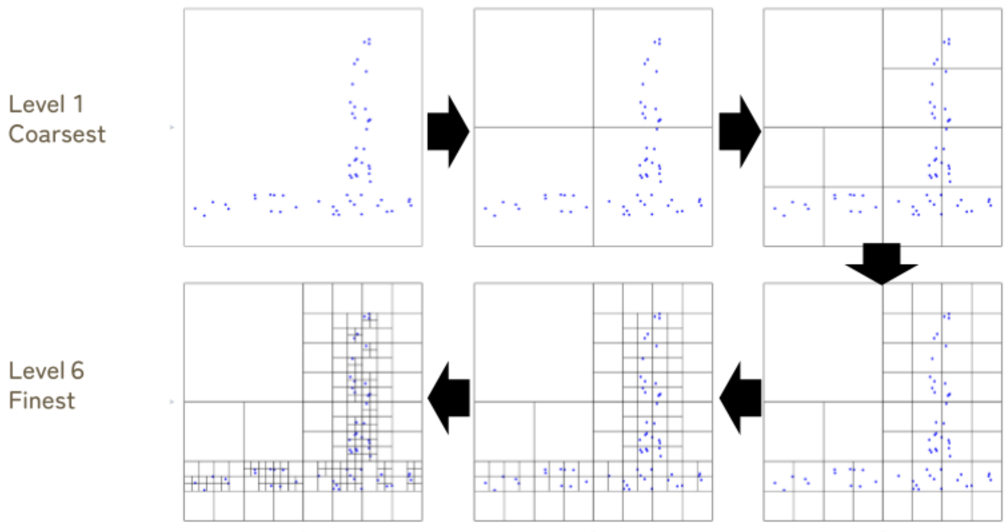

Fig. 4. A diagram of the Octree concept, replaced by a two-dimensional Quadtree.

generates and updates voxel data of an appropriate size from point cloud data based on the set distance parameter. Figure 1 shows how the change occurs as the viewpoint moves.

As the viewpoint approaches the object, the voxel representation transitions to a more detailed one as shown in Figure 1(a), and conversely, as the viewpoint moves away from the object, the representation gradually changes to a coarser voxel representation as shown in Figure 1(d). In this case, if the object has clear geometric structural features, the original three-dimensional shape can be fully recognized even in a low-resolution state like Figure 1(d).

Figure 2 shows the rendering behavior when there are multiple objects in a scene. In this system, the distance between each object and the viewpoint is calculated individually, and the LOD is switched independently for each object. As shown on the left side of Figure 2, the "castle" in the background, which is far from the viewpoint, is displayed using coarse voxels (low resolution). On the other hand, as shown on the right side of Figure 2, when the viewpoint approaches the "castle," it can be seen that the voxels are automatically switched to fined voxels (high resolution). In this way, by applying a process to each individual object that renders objects close to the viewpoint in detail and distant objects coarsely, efficient rendering control is achieved for the entire scene.

D. Voxel Rendering with Dynamic LOD Control

This system utilizes the JavaScript library Three.js and shaders (GLSL) to implement voxel visualization in which the level of detail changes dynamically depending on the viewpoint distance. Specifically, the distance from the viewpoint, which serves as the basis for switching LOD levels, can be set as an arbitrary threshold. The system updates the voxel data to be drawn in real time based on this setting. This allows the level of detail to transition gradually as the viewpoint moves, from high resolution as the object approaches, to low resolution as the object moves away.

To verify the performance of this system, we measured the operating Frames Per Second (FPS). When drawing a single object, it normally (when stationary) operates at 144 FPS, but when the voxel size was changed (LOD transition), we confirmed that the frame rate dropped from approximately 110 FPS to 80 FPS. However, because the frame rate never fell below 60 FPS, which is the benchmark for smooth drawing, we can conclude that this does not adversely affect the user experience. Furthermore, even when the number of objects to be drawn was increased to 10, we were able to maintain a rate of 60 FPS or higher, which suggests that this method is able to effectively suppress increases in drawing load.

Regarding the implementation method, when compared to the "method of generating a cube mesh centered on the coordinates of each point," the "drawing method using shaders" significantly reduced the drawing load when changing the voxel size (when switching LOD), achieving smoother operation.

IV. Case Studies

As a result of the observations, it was confirmed that when the original three-dimensional structure has clear geometric characteristics, such as the castle in Figure 3, the shape of the object can be sufficiently identified even with a low-resolution, coarse voxel representation (Figure 1).

In current methods, when changing voxel size according to the distance from the viewpoint, a uniform size is applied to all voxels. However, this method has the problem of resulting in coarse surface representation and a loss of smoothness in areas where detailed shape reproduction is required, such as curved surfaces and fine irregularities, as shown in Figure 5.

Observing the rendering results, for simple shapes such as those in Figures 6(a) and 6(b), it is possible to identify the original three-dimensional object even with coarse voxels.

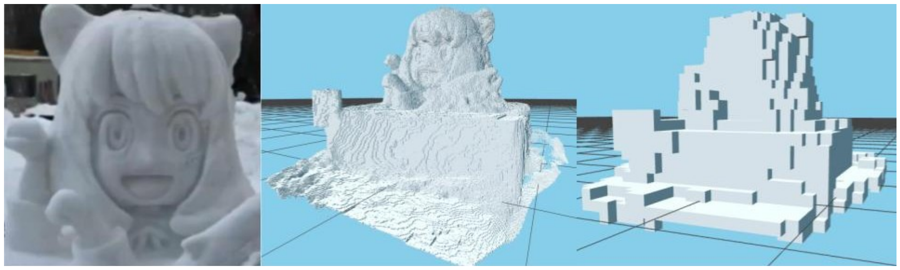

Fig. 5. Visualization comparison of snow sculpture data containing many curved surfaces. From left to right, the images show a photograph of the actual object, a fined voxel representation , and a coarse voxel representation.

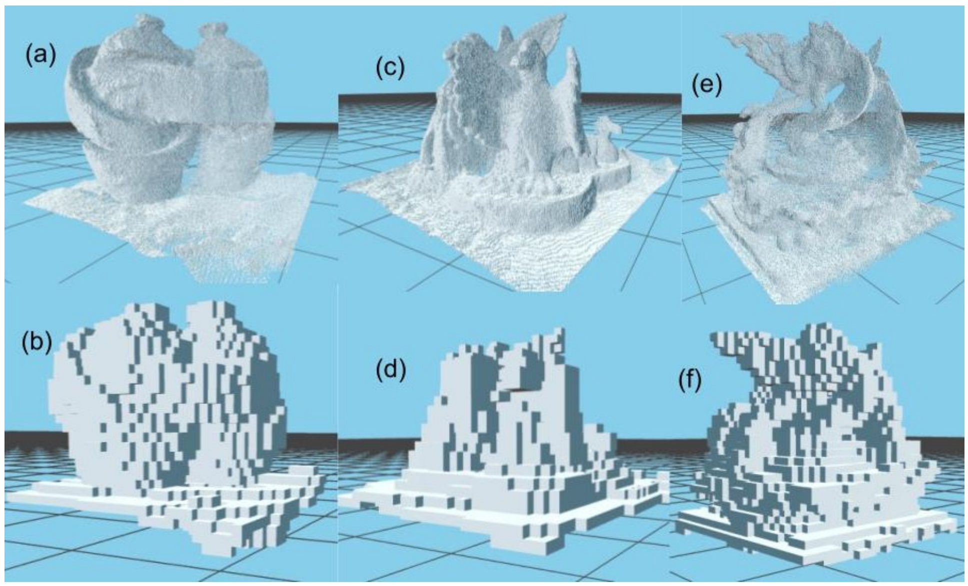

Fig. 6. A comparison of fined and coarse voxel representations using snow sculpture data from an international snow sculpture contest. The top row (a), (c), and (e) show detailed representations, while the bottom row (b), (d), and (f) show coarse voxel representations. The subjects, from left to right, are South Korea's "Harmony", Portland's "FINAL APPROACH," and Thailand's "The NAGA Fireballs".

On the other hand, for complex shapes such as those in Figures 6(c), 6(d), 6(e), and 6(f), characteristic parts that require detailed representation are missing, making it difficult to identify the original three-dimensional shape. This occurs not only when the voxel size is coarse, but also when the voxel size is somewhat fine, demonstrating the limitations of uniform resizing.

Therefore, to improve visibility, it is necessary to implement a function that locally varies the level of detail according to the geometric complexity of the shape, even within a single object. Specifically, we will utilize the octree structure and introduce "shape complexity of the point cloud" as a new LOD control criterion in addition to the standard "distance from the viewpoint." This is expected to significantly improve visibility by rendering flat areas with coarse voxels and complex areas with detailed voxels, even for objects at the same distance.

In addition, in visualization experiments using this system, some missing point cloud data was confirmed (Figures 6(a), 6(c), 6(e)). This is due to the accuracy of the input data. Therefore, improvements to the data generation process, such as preparing a more comprehensive and multi-angle image set, are also necessary.

V. Conclusion

In this study, we developed a voxel visualization system that dynamically controls the level of detail according to the distance from the viewpoint. Through evaluation of this system, we confirmed that controlling the level of detail according to distance makes it possible to reduce the amount of data while maintaining the visibility of objects. One current issue is that the quantitative basis for the optimal distance threshold for switching LOD levels is not clear. In the future, we plan to conduct quantitative evaluations, including subject experiments, and optimize these parameters.

References

[1] O.Wegen and others, "A Survey on Non-photorealistic Rendering Approaches for Point Cloud Visualization," TVCG, vol.32, issue.9, 2025.

[2] Y.Xu and others, "Voxel-based representation of 3D point clouds: Methods, applications, and its potential use in the construction industry," Automation in Construction, vol.126, 2021.

[3] O.Martinez-Rubi and others, "Taming the beast: Free and open-source massive point cloud web visualization," Capturing Reality 2015, 2015.