Developer Documentation

1 Platform Architecture Overview

1.1 Project Aim and Architecture

This project aims to develop a high-efficiency system capable of real-time point cloud rendering using GPU parallel computing and the Vulkan API. The architecture is designed to handle and visualize large quantities of point-cloud data, with specific optimizations to manage up to 30 million data points effectively. The system is divided into four main components:

- Data Input Module: Reads and parses point cloud data files in PLY format.
- Data Processing Module: Utilizes GPU acceleration and multiple sampling anti-aliasing (MSAA), and vertex index caching and so on to optimize data for rendering.
- Data Rendering Module: Uses some method such as frustum cullingand texture mapping for efficient rendering of point cloud data.
- Rendering Interface: Developed with ImGui for real-time user interaction and rendering control.

This modular design ensures scalability, allowing for easy updates and integration of new features or optimizations.

1.2 Tools and Technologies

- Vulkan API: Chosen for its high performance and cross-platform capabilities, particularly in handling complex rendering tasks. Vulkan's multi-threading and parallel processing capabilities allow for efficient GPU utilization, essential for rendering millions of data points.

- GLFW: Used for window creation, user input management, and interfacing with the operating system. GLFW provides a stable foundation for handling user interactions and

rendering contexts.

- ImGui: A lightweight GUI library integrated for real-time adjustments and visualization during rendering. ImGui is used to create interactive interfaces that allow users to control rendering parameters dynamically.

- C++: The primary programming language for this project, chosen for its performance and control over system resources. The project uses C++ for implementing low-level graphics processing functions and high-level application logic.

2 Local Development Environment Setup

- The Operating System that the user can use must be the Windows 10 or later, During the testing phase of the project, Use is Windows 11 for the operating system; This project requires Intel Core i7-9750H or equivalent; If you need to verify the data obtained during the test phase of the project, Please use the NVIDIA GeForce GTX 1660 Ti or equivalent with Vulkan support; The Development Environment of the project is the Visual Studio 2022.

- The point cloud dataset that needs to be used is already included in the project code, please make sure that you have already installed Vulkan SDK, The Vulkan SDK can be downloaded from https://vulkan.lunarg.com. If you want to verify the data results obtained by using the Scannt dataset, please download from here: https://github.com/ScanNet/ScanNet, which is not included in the code delivered by the project.

3 Configure the Project

- Please open the project with the Visual Studio 2022, and the Set the appropriate configuration (Debug / Release) and target platform (x64).

- If you cannot run the code, the path of the dataset maybe incorrect, in main.cpp you can modify the path of the dataset and the point cloud dataset only supports the PLY file format.

Please right-click on the solution and select `Build Solution`. Ensure there are no errors during the build process.