В связи с интенсивным ростом производительности лазерных сканеров все более актуальной становится проблема визуализации результатов лазерного сканирования. Одной из эффективных методик является отображение огромных массивов точечных данных. Для визуализации используется заранее вычисленная структура данных — восьмеричное иерархическое дерево. Практическая реализация предложенной методики показала свою эффективность при отображении массивов из десятка миллионов точек.

ВИЗУАЛИЗАЦИЯ РЕЗУЛЬТАТОВ ЛАЗЕРНОГО СКАНИРОВАНИЯ

АЛЕКСАНДР ВЕЛИЖЕВ
Московский государственный университет Геодезии и Картографии

В настоящее время производительность лазерных сканеров увеличивается с каждым годом. Современные наземные лазерные сканеры импульсного типа способны сканировать поверхность объекта со скоростью до 10 000 точек в секунду, а сканеры фазового типа — до 500 000 точек. Для обработки и анализа полученных данных необходима эффективная система визуализации точек сканирования. Большинство существующих видеокарт широкого доступа могут отображать лишь 1–3 миллиона точек без ощутимых задержек. Однако объемы данных, с которыми приходится сталкиваться на практике, существенно превышают эти цифры. Поэтому задача разработки эффективных алгоритмов визуализации данных сканирования является очень актуальной.

Любая система визуализации результатов лазерного сканирования обязана эффективно обеспечивать несколько операций. Это масштабирование, вращение сцены и ее сдвиг.

Для решения поставленной задачи был разработан алгоритм, который успешно реализует все указанные операции. Основой алгоритма являются три составляющие: (1) использование специальной структуры данных для хранения точек; (2) применение различных уровней детализации; (3) динамическое отсечение невидимой части сцены.

Построение восьмеричного иерархического дерева

Для решения задачи точечные данные однократно преобразуются в специальную структуру данных — восьмеричное иерархическое дерево, для которого заранее вычисляются уровни детализации. Построение восьмеричного дерева, которое состоит из вложенных друг в друга узлов, выполняется с целью преобразования исходного массива точек в четко организованную структуру, удобную для анализа и хранения.

Создание структуры данных включает в себя два основных этапа: прямое и обратное движение точек.

Каждая вершина дерева может содержать в себе набор точек, а также имеет четкие пространственные границы, заданные минимальной и максимальной точками описывающего параллелепипеда. Вложенные узлы называются потомками. Важно отметить, что построение дерева не прибавляет и не уменьшает количество точек, координаты которых не изменяются. Дерево — это только структура для организации хранения исходных точек.

Прямое движение точек

Данный шаг выполняется для первичной обработки всех исходных данных. К примеру, дан массив P из K исходных точек. Для построения дерева выбирается параметр N_{max}, который задает максимальное число точек в вершине дерева (например, N_{max} можно выбрать равным 1024 точкам). Вводится понятие листа дерева как узла, у которого нет потомков.

В процессе пополнения данных из файла точки считываются партиями (например, по 1 000 000), каждая из которых последовательно добавляется в иерархическое дерево. Разбивка точек на отдельные партии позволяет исключить ситуацию, когда все точки загружены в оперативную память компьютера дважды.

Обратное движение точек

Данный шаг выполняется с целью создания уровней детализации, которые используются для динамической генерализации облака точек в процессе визуализации. Основная идея алгоритма состоит в том, что для всех узлов дерева строится сильно упрощенная точечная модель из исходных точек. При этом никакого дублирования данных не происходит. Алгоритм обратного движения точек включает в себя один главный шаг — для каждого нелистового узла дерева извлекаем N_{max} точек из потомков текущего узла. Для этого N_{max} делится на равные части между всеми потомками, которые содержат точки (непустые узлы). В момент извлечения из листа точки выбираются случайным образом. В виду того, что дерево разбивает исходное пространство на области одинаковой плотности, каждый уровень детализации содержит равномерно распределенные точки с постоянной плотностью.

Динамическая визуализация точек

В процессе непосредственного просмотра точечных данных в каждый момент времени необходимо сформировать набор точек для визуализации. Этот набор определяется в зависимости от текущего положения точки обзора. Если в один пиксель экрана попадает слишком много точек, то нет никакого смысла отображать все данные. Для определения такой ситуации разработан критерий генерализации δ:

$$\delta = \begin{cases} 1, & \text{если } NodeDiagonal <- MinNodeSize \\ 0, & \text{в противном случае} \end{cases}$$

где *NodeDiagonal* — длина диагонали параллелепипеда узла, *MinNodeSize* — минимальный размер узла, который виден в камеру (значение этого параметра рассчитывается в зависимости от текущего размера пикселя экрана с учетом масштаба).

Одновременно большая часть точек может оказаться вне области видимости, поэтому их не стоит отображать — это выполняется за счет отсечения узлов дерева, которые не попадают в область видимости камеры. При этом сразу отсекается узел со всеми его потомками.

После формирования набора точек все готово к отображению.

Выводы и результаты

Реализация предложенной методики визуализации результатов лазерного сканирования показала высокую эффективность и производительность. Масштабирование, вращение набора из 60 миллионов точек, выполнялось без каких-либо заметных задержек. Большинство коммерческих продуктов при вращении точечных данных скрывает часть информации, оставляя лишь контурные точки, что может вызвать затруднение в навигации. Предложенная методика лишена данного недостатка — при выполнении операции поворота никакой «сильной» генерализации не происходит.

В предлагаемом алгоритме все точечные данные загружаются в оперативную память. Проблема нехватки памяти решается за счет динамической загрузки точек из файла данных. Структура иерархического восьмеричного дерева может быть легко разбита на пространственно локализованные группы точек, которые при необходимости будут сохраняться или загружаться независимо друг от друга.

АЛГОРИТМ ПРЯМОГО ДВИЖЕНИЯ ТОЧЕК

Для каждой точки p_i из P:

Шаг 1. Точка p_i добавляется в набор точек корневого узла дерева, который назначается текущим узлом.

Шаг 2. Если текущий узел не имеет потомков, добавляется точка текущего узла. В противном случае необходимо найти того потомка, в границы которого попадает точка p_i, перенести точку в набор этого потомка, назначить его текущим и снова перейти к шагу 2.

Шаг 3. Если число точек в наборе текущего листа превысило значение N_{max}, необходимо разбить этот узел на потомков путем деления куба текущего узла на восемь равных вложенных частей.

АЛГОРИТМ ФОРМИРОВАНИЯ НАБОРА ТОЧЕК ДЛЯ ОТОБРАЖЕНИЯ

Шаг 1. Назначается корневой узел дерева текущим узлом;

Шаг 2. Для каждого потомка текущего узла:

- **а.** Проверяется попадание в текущую область видимости камеры. Если текущий узел не виден в камеру, осуществляется переход к проверке следующего потомка;
- **б.** Проверяется условие генерализации. Если δ = 1, то добавляются точки из текущего потомка в набор точек для отображения, и осуществляется переход к проверке следующего потомка;
- **в.** Если текущий потомок является листом, то осуществляется переход к проверке следующего потомка;
- **г.** Текущий потомок назначается текущим узлом, и осуществляется переход к шагу 2;

Шаг 3. Набор точек для визуализации сформирован.