Для цитирования: Жуков Ю. А. Сравнение библиотек программного обеспечения, используемых для решения задачи визуализации радиолокационных 3D-изображений // Вопросы радиоэлектроники. 2019. № 9. С. 37–41.
DOI 10.21778/2218-5453-2019-9-37-41
УДК 004.43

Ю. А. Жуков[1]

[1] АО «Научно-производственное предприятие «Радар ммс»

СРАВНЕНИЕ БИБЛИОТЕК ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ, ИСПОЛЬЗУЕМЫХ ДЛЯ РЕШЕНИЯ ЗАДАЧИ ВИЗУАЛИЗАЦИИ РАДИОЛОКАЦИОННЫХ 3D-ИЗОБРАЖЕНИЙ

В статье проводится сравнительный анализ библиотек программного обеспечения с открытой лицензией, используемых для решения задач визуализации радиолокационных 3D-изображений, поддерживающих разработку ПО с использованием языка программирования C++. Дано обоснование преимущества представления результатов обработки радиолокационных данных в виде трехмерной модели вместо двумерных изображений. Описана специфика формирования трехмерного радиолокационного изображения. Приведено подробное описание каждой библиотеки, их достоинств и недостатков, выявленных при решении задачи визуализации трехмерного изображения. Результатом сравнительного анализа стало обоснование условий выбора каждой библиотеки в соответствии с возможностями вычислительной системы. Правильный выбор библиотеки программного обеспечения позволяет разработать качественный, соответствующий предъявляемым требованиям продукт для формирования трехмерного изображения, что ускорит проведение анализа радиолокационных данных.

Ключевые слова: *трехмерная визуализация, 3D-модель, графическая библиотека*

Введение

Трехмерная визуализация (3D-визуализация) является одним из наглядных способов представления большого объема данных. Большинство современных систем, работающих с большими массивами числовой или другой информации, имеют возможность представить результаты работы в виде трехмерного изображения для облегчения их восприятия человеком, что при необходимости позволяет принять решение в более короткие сроки. Эта возможность чрезвычайно актуальна в сфере радиолокации, так как существует множество систем, в которых принятие решения не автоматизировано и полностью зависит от человеческого фактора. В таких случаях необходимо предоставить всю необходимую информацию в кратчайшие сроки и в удобном для изучения виде.

Трехмерное радиолокационное изображение проще для восприятия, так как позволяет посмотреть на исследуемый объект под разными углами, оценить его физические параметры, более точно определить геометрическую форму.

3D-визуализация в радиолокации используется относительно недавно, что связано с ограничениями производительности вычислительных систем. В связи с этим большинство программных продуктов предоставляют обработанную информацию в виде изображения на плоскости, построение которого требует значительно меньших ресурсов. Однако 2D-изображение не всегда удобно для изучения из-за сложности его восприятия.

Специфика формирования трехмерного радиолокационного изображения

На рис. 1 представлено изображение карты высот в 3D. Спецификой формирования трехмерного радиолокационного изображения является необходимость обработки большого объема информации для его обновления. Использование стандартного метода формирования 3D-изображения – полигональной сетки – является ресурсозатратным решением по причине необходимости обновления структуры всей сетки несколько десятков раз в секунду.

Организовать обработку радиолокационной информации в режиме, близком к реальному времени (с частотой около 100 Гц), позволяет применение технологии вокселей [1]. Трехмерное радиолокационное изображение формируется из вокселей – кубов, размер которых зависит от разрешающей способности РЛС. Воксель является минимальной единицей трехмерного пространства аналогично пикселю 2D-изображения и хранит в себе четыре

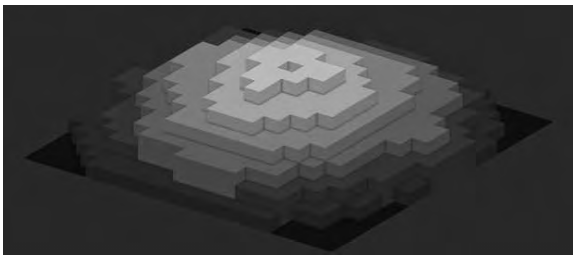

Рисунок 1. 3D-изображение карты высот

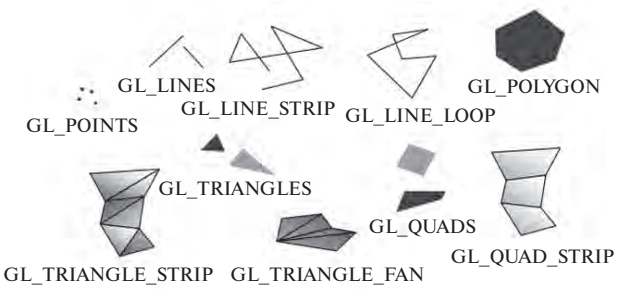

Рисунок 2. Примитивы, используемые в OpenGL

параметра: координаты вершин и цвет. Качество полученного изображения зависит от плотности вокселей, т.е. их количества на единицу объема пространства. Для получения полезных изображений плотность должна более чем в 10 раз превышать размер отображаемого объекта.

Библиотеки программного обеспечения

Одним из наиболее распространенных языков программирования при решении задач визуализации различных 3D-объектов является C++. Данный язык позволяет наиболее точно распределять ресурсы компьютера, что крайне важно при разработке ресурсоемких задач, таких как 3D-визуализация. Рассмотрим несколько инструментов, позволяющих разработать систему трехмерной визуализации с использованием языка программирования C++ и предлагаемых под открытой лицензией. Это графические библиотеки OpenGL, Visualization Library, а также фреймворк Qt, содержащий модуль Qt3D для разработки трехмерных приложений.

OpenGL

OpenGL (Open Graphic Library) – спецификация, определяющая программный интерфейс, и основная библиотека при разработке 3D-программ [2]. Остальные рассматриваемые инструменты являются надстройками над данной графической библиотекой и представляют ее возможности в более удобном формате при написании программы.

В OpenGL описаны свыше 200 функций, которые позволяют задать объект и взаимодействовать с ним [3]. Все возможности OpenGL отражены в предоставляемых функциях. Их можно разделить на пять групп:

- функции для создания простых объектов (примитивов). В качестве простых объектов используются точки, линии, плоскости (полигоны) (рис. 2) [4];
- функции для создания источников света, отвечающие за расположение света в сцене;
- функции для параметров объектов. С их помощью описываются такие параметры, как цвет, материал, а также определяются параметры света;
- функции для задания параметров камеры, отвечающие за то, как объекты в сцене будут восприниматься глазом. Задаются такие параметры, как проекция камеры, порядок отрисовки объектов и их приоритет, отсечение объектов, находящихся вне зоны видимости камеры;
- функции воздействия на объект, к которым относятся различные преобразования объекта, такие как масштабирование, перемещение, поворот.

Visualization Library

Visualization Library – библиотека для визуализации, написанная на языке C++. Библиотека основана на древовидном разделении структуры данных – она использует различные структуры данных для управления каждым конкретным компонентом визуализации [5]. Например, дерево преобразования объектов визуализации хранится в одной отдельной структуре данных древовидной системы. Объекты сцены хранятся в другой структуре и могут использовать при этом структуру преобразований. Актеры хранятся в своей собственной структуре. Шейдеры также представляют собой концепцию, которая не зависит от остальной части логики.

В Visualization Library уже реализованы функции отрисовки некоторых примитивов, таких как куб, цилиндр, сфера и т.п. Каждая функция является совокупностью команд OpenGL и позволяет в некоторых случаях экономить время на написание программы [6].

Модуль Qt3D фреймворка Qt

Qt3D является платформонезависимой C++-надстройкой над OpenGL, позволяющей в значительной мере упростить и ускорить процесс разработки 3D-приложений. Данный модуль позволяет создавать трехмерные симуляционные системы, работающие в реальном времени.

Модуль Qt3D использует принцип ECS – entity-component-system («сущность – компонент – система») для передачи информации объекту [7]. Сущность – имитируемый объект, лишенный

каких-либо свойств и характеристик. Компонент характеризует поведение объекта. Дополнительное поведение может быть привязано к объекту путем объединения одного или нескольких компонентов.

Особенностью ECS является то, что, используя агрегацию, можно динамически изменять поведение объекта путем добавления или удаления компонентов.

Агрегация в Qt позволяет независимо создавать компоненты и объекты, присоединять один и тот же компонент к разным объектам. Однако такая самостоятельность компонентов приводит к ряду проблем. Например, если не подключить компонент, отвечающий за отрисовку объекта в сцене, программа будет работать без ошибок, но при этом объект не будет отображаться в сцене. В связи с этим требуется внимательно проверять подключенные компоненты при разработке программы.

Qt3D создает настраиваемые объекты путем агрегирования компонентов, которые предоставляют дополнительные возможности [8]. Qt3D использует Renderer Aspect для обработки и обновления объектов с определенными компонентами. Renderer Aspect отвечает за рендеринг в 3D-сценах с использованием оптимальной версии OpenGL для данного графического процессора.

Сравнение библиотек программного обеспечения

Главным достоинством библиотеки OpenGL является кросс-платформенность – она поддерживается большинством производителей оборудования на разных операционных системах, таких как Windows, Mac OS, Linux. Кроме того, можно выделить следующие преимущества использования OpenGL:

- стабильность: при обновлении версии библиотеки OpenGL гарантируется работоспособность программ, написанных на более ранних версиях;
- надежность: написанные на OpenGL программы выдают одинаковые результаты на различных платформах, а также при использовании разного оборудования с условием, что производитель данного оборудования поддерживает OpenGL в своей продукции.

Среди недостатков можно выделить сложную структуру и большой объем кода при написании программ. Вследствие этого, время на разработку программы резко возрастает.

Необходимо отметить, что остальные рассматриваемые библиотеки являются надстройками над OpenGL разных версий и предоставляют готовый набор команд для упрощения создания приложений.

Visualization Library, в отличие от Qt3D, тонко интегрирует в себя функции OpenGL. Это позволяет разрабатывать приложения на более низком уровне, близком к разработке на чистом OpenGL, и лучше контролировать использование ресурсов вычислительной машины. Для работы с Visualization Library необходима поддержка оборудованием версии OpenGL не ниже 3.0. Еще одним отличием от Qt3D является то, что Visualization Library не зависит от какой-либо конкретной библиотеки графического интерфейса и предоставляет только OpenGL-контекст. Тем не менее Visualization Library имеет возможность связи с графическими интерфейсами, такими как Qt5, GLFW, MFC, Win32. Благодаря древовидному разделению структуры данных в библиотеке Visualization Library программа разделяется на отдельные блоки, независимые друг от друга и при этом взаимодействующие, что позволяет быстро добавить необходимую функциональность в ходе разработки, а также изменять отдельные компоненты системы, не затрагивая другие. Главным недостатком библиотеки является отсутствие описания большинства функций. Документация генерируется из кода, что затрудняет восприятие некоторых функций библиотеки. Кроме того, последняя версия Visualization Library была выпущена в 2017 году, после чего разработка остановилась.

Первая версия Qt3D поставлялась в виде надстройки над OpenGL 2.0. В настоящее время Qt3D может использовать любое подмножество функциональности OpenGL, поддерживаемое устройством, на котором ведется разработка приложения. Для этого был создан метод для предоставления точек входа функций OpenGL любой требуемой версии. Это позволяет разрабатывать единое приложение как для старых, так и для новых устройств. Как и Visualization Library, Qt3D также разделяет компоненты и позволяет взаимодействовать с ними по отдельности, а благодаря связи компонентов с сущностями путем агрегации появляется возможность назначить один компонент к нескольким сущностям. Однако система ECS имеет недостаток – низкая скорость создания сущностей.

Применение технологии вокселей при формировании 3D-изображения

В OpenGL при формировании вокселя используются координаты восьми вершин. Весь набор вокселей записывается в буфер оперативной памяти, который с определенной частотой помещается в видеопамять GPU для отрисовки. Visualization Library и Qt3D на нижнем уровне работают по тому же принципу, хотя и предоставляют более удобный интерфейс формирования трехмерного изображения.

В Visualization Library, кроме описанного выше способа создания объектов, существуют методы, позволяющие отрисовывать примитивы, не углубляясь в способ записи их вершин в буфер памяти: каждый воксель может задаваться координатами центра, цветом и размером. Qt3D также предоставляет разработчику оба способа создания объектов. Главное отличие заключается в использовании системы ECS, в соответствии с которой буфер вокселей закрепляется за сущностью как компонент. Следовательно, для обращения к отдельному вокселю необходимо найти сущность, к которой он относится, а затем его позицию в буфере. Такой подход несколько затрудняет изменение параметров отдельно взятого вокселя, однако оптимально подходит под концепцию воксельной графики, поддерживающую систему LOD (Level of detail) [9] на уровне сущностей.

Выводы

Каждая из рассмотренных библиотек может быть использована для визуализации радиолокационных 3D-изображений. Наиболее подходящей для визуализации 3D РЛИ в плане быстродействия и ресурсоемкости является технология вокселей. Visualization Library и Qt3D предоставляют более удобный интерфейс для создания вокселей, чем OpenGL. В Qt3D также имеется технология ECS, которая позволяет оперировать набором вокселей вместо отдельных вокселей. Однако Qt3D требует применения графического интерфейса Qt, в то время как Visualization Library дает возможность использовать целый набор графических интерфейсов для разных операционных систем. Три указанные библиотеки и воксельные технологии позволяют осуществить визуализацию в режиме, близком к реальному времени.

СПИСОК ЛИТЕРАТУРЫ

1. Voxels [Электронный ресурс]. URL: https://en.wikipedia.org/wiki/Voxels (дата обращения: 04.06.2019).
2. OpenGL Documentation. [Электронный ресурс]. URL: https://www.opengl.org/documentation/ (дата обращения: 02.06.2019).
3. OpenGL Reference Page [Электронный ресурс]. URL: https://www.khronos.org/registry/OpenGL-Refpages/gl4/ (дата обращения: 02.06.2019).
4. OpenGL Architecture [Электронный ресурс]. URL: https://www.google.com/url?sa=t&source=web&rct=j&url=https://web.cs.wpi.edu/~matt/courses/cs563/talks/OpenGL_Presentation/OpenGL_Presentation.html (дата обращения: 04.06.2019).
5. Visualization Library Documentation, v.2.0.0-b3 [Электронный ресурс]. URL: http://visualizationlibrary.org/docs/2.0/html/index.html (дата обращения: 31.05.2019).
6. Visualization Library Tutorials [Электронный ресурс]. URL: http://visualizationlibrary.org/docs/2.0/html/index.html (дата обращения: 31.05.2019).
7. Overview of Qt3D2.0 [Электронный ресурс]. URL: https://www.kdab.com/overview-qt3d-2–0-part-1/ (дата обращения: 03.06.2019).
8. Qt 4.7.0: Basic Qt Archtecture – Qt Documentation [Электронный ресурс]. URL: https://doc.qt.io/archives/qt-4.7/qt-basic-concepts.html (дата обращения: 03.06.2019).
9. LOD (Level of details) [Электронный ресурс]. URL: https://en.wikipedia.org/wiki/Level_of_Detail (дата обращения: 05.06.2019).

ИНФОРМАЦИЯ ОБ АВТОРЕ

Жуков Юрий Алексеевич, магистр, АО «Научно-производственное предприятие «Радар ммс», Российская Федерация, 197375, Санкт-Петербург, ул. Новосельковская, д. 37, лит. А, тел.: 8 (911) 757-78-30, e-mail: zhukov_jua@radar-mms.com.

For citation: Zhukov Yu. A. Software libraries comparison used for 3D radar images visualization. Voprosy radioelektroniki, 2019, no. 9, pp. 37–41.
DOI 10.21778/2218-5453-2019-9-37-41

Yu. A. Zhukov

SOFTWARE LIBRARIES COMPARISON USED FOR 3D RADAR IMAGES VISUALIZATION

This article compares open source software libraries to solve the radar image 3D visualization tasks and supporting software development using a C++ programming language. The article grounds advantages of presenting the processing radar data results in the 3D model form instead of 2D images. The article describes the 3D image formation specificity. The detailed description of each library is given, their advantages and disadvantages revealed during 3D visualization process are presented. The comparative analysis result was the library choice justification in according with the computer capabilities. Choosing of the right software library will allow develop high-quality software for generating of 3D radar image and speed up the radar data processing analysis.

Keywords: three-dimensional visualization, 3D model, graphic library

REFERENCES

1. Voxels. Available at: https://en.wikipedia.org/wiki/Voxels (accessed 04.06.2019).
2. OpenGL Documentation. Available at: https://www.opengl.org/documentation (accessed 02.06.2019).
3. OpenGL Reference Page. Available at: https://www.khronos.org/registry/OpenGL-Refpages/gl4 (accessed 02.06.2019).
4. OpenGL Architecture. Available at: https://www.google.com/url?sa=t&source=web&rct=j&url=https://web.cs.wpi.edu/~matt/courses/cs563/talks/OpenGL_Presentation/OpenGL_Presentation.html (accessed 04.06.2019).
5. Visualization Library Documentation, v.2.0.0-b3. Available at: http://visualizationlibrary.org/docs/2.0/html/index.html (accessed 31.05.2019).
6. Visualization Library Tutorials. Available at: http://visualizationlibrary.org/docs/2.0/html/index.html (accessed 31.05.2019).
7. Overview of Qt3D2.0. Available at: https://www.kdab.com/overview-qt3d-2–0-part-1 (accessed 03.06.2019).
8. Qt 4.7.0: Basic Qt Archtecture – Qt Documentation. Available at: https://doc.qt.io/archives/qt-4.7/qt-basic-concepts.html (accessed 03.06.2019).
9. LOD (Level of details). Available at: https://en.wikipedia.org/wiki/Level_of_Detail (accessed 05.06.2019).

AUTHOR

Zhukov Yury, master of science, «NPP «Radar mms» JSC, 37A, Novoselkovskaya St., Saint-Petersburg, 197375, Russian Federation, tel.: +7 (911) 757-78-30, e-mail: zhukov_jua@radar-mms.com.