|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

|  |  |
| --- | --- |
| ФАКУЛЬТЕТ | Робототехника и комплексная автоматизация (РК) |
| КАФЕДРА | Системы автоматизированного проектирования (РК6) |

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ***

***НА ТЕМУ:***

***«Изучение методов рендеринга трёхмерной графики с помощью библиотеки»***

|  |  |  |
| --- | --- | --- |
| Студент РК6-74Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **Губанов Д. А.** |
|  | (Подпись, дата) | И.О. Фамилия |
| Руководитель | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **Витюков Ф.А.** |
|  | (Подпись, дата) | И.О. Фамилия |

*2023 г.*

**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой РК6

А.П. Карпенко

«\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2022 г.

**ЗАДАНИЕ**

**на выполнение научно-исследовательской работы**

по теме:  Изучение методов рендеринга трёхмерной графики с помощью библиотеки OpenGL\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент группы \_\_РК6-43Б\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Губанов Даниил Александрович\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия, имя, отчество)

Направленность НИР (учебная, исследовательская, практическая, производственная, др.) \_учебная\_\_\_

Источник тематики (кафедра, предприятие, НИР) \_кафедра \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

График выполнения НИР: 25% к 5 нед., 50% к 11 нед., 75% к 14 нед., 100% к 16 нед.

Техническое задание:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Оформление научно-исследовательской работы:***

Расчетно-пояснительная записка на 14 листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.):

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания «8» февраля 2023 г.

|  |  |  |
| --- | --- | --- |
| **Руководитель НИР** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **Витюков Ф.А.** |
|  | (Подпись, дата) | И.О. Фамилия |
| **Студент** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **Губанов Д. А.** |
|  | (Подпись, дата) | И.О. Фамилия |

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 4](#_Toc133414239)

[1. OpenGL 4](#_Toc133414240)

[1. Назначение 4](#_Toc133414241)

[2. Архитектура 4](#_Toc133414242)

[3. Базовый функционал и возможности 5](#_Toc133414243)

[2. Immediate mode 7](#_Toc133414244)

[1. Инициализация 7](#_Toc133414245)

[2. Отображение полигонов 8](#_Toc133414246)

[3. Отображение цветов 8](#_Toc133414247)

[4. Трансформация 8](#_Toc133414248)

[5. Создание 3D моделей 8](#_Toc133414249)

[6. Наложение текстур 8](#_Toc133414250)

[3. Сore profile 8](#_Toc133414251)

[1. Инициализация 8](#_Toc133414252)

[2. Отображение полигонов 8](#_Toc133414253)

[3. Создание шейдеров 8](#_Toc133414254)

[4. Наложение текстур 8](#_Toc133414255)

[5. Трансформация 8](#_Toc133414256)

[6. Создание 3D моделей 8](#_Toc133414257)

[7. Преобразования пространств 8](#_Toc133414258)

[8. Трансформация камеры 8](#_Toc133414259)

[ЗАКЛЮЧЕНИЕ 8](#_Toc133414260)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 8](#_Toc133414261)

ВВЕДЕНИЕ

Сейчас трёхмерные изображения можно увидеть везде, начиная от компьютерных игр и заканчивая различными системами моделирования и симулирования. Раньше, когда трёхмерная графика существовала только у единиц, не существовало единого стандарта в области графики. Все программы писались с "нуля" или с использованием накопленного опыта, но в каждой программе реализовывались свои методы для отображения графической информации. С приходом мощных процессоров и графических ускорителей трёхмерная графика стала реальностью для персональных компьютеров. Но в тоже время производители программного обеспечения столкнулись с серьёзной проблемой - это отсутствие каких-либо стандартов, которые позволяли писать программы, независимые от оборудования и операционной системы. Одним из первых таких стандартов, существующий и по сей день является OpenGL.

OpenGL - это графический стандарт в области компьютерной графики. На данный момент он является одним из самых популярных графических стандартов во всём мире. Ещё в 1982 г. в Стенфордском университете была разработана концепция графической машины, на основе которой фирма Silicon Graphics в своей рабочей станции Silicon IRIS реализовала конвейер рендеринга. Таким образом была разработана графическая библиотека IRIS GL. На основе библиотеки IRIS GL, в 1992 году был разработан и утверждён графический стандарт OpenGL. Разработчики OpenGL - это крупнейшие фирмы разработчики как оборудования так и программного обеспечения: Silicon Graphics, Inc., Microsoft, IBM Corporation, Sun Microsystems, Inc., Digital Equipment Corporation (DEC), Evans & Sutherland, Hewlett-Packard Corporation, Intel Corporation и Intergraph Corporation.

OpenGL переводится как Открытая Графическая Библиотека (Open Graphics Library), это означает, что OpenGL - это открытый и мобильный стандарт. Программы, написанные с помощью OpenGL можно переносить практически на любые платформы, получая при этом одинаковый результат, будь это графическая станция или суперкомпьютер. OpenGL освобождает программиста от написания программ для конкретного оборудования. Если устройство поддерживает какую-то функцию, то эта функция выполняется аппаратно, если нет, то библиотека выполняет её программно.

Что же представляет из себя OpenGL? С точки зрения программиста OpenGL - это программный интерфейс для графических устройств, таких как графические ускорители. Он включает в себя около 150 различных команд, с помощью которых программист может определять различные объекты и производить рендеринг. Говоря более простым языком, вы определяете объекты, задаёте их местоположение в трёхмерном пространстве, определяете другие параметры (поворот, масштаб, ...), задаёте свойства объектов (цвет, текстура, материал, ...), положение наблюдателя, а библиотека OpenGL позаботится о том чтобы отобразить всё это на экране. Поэтому можно сказать, что библиотека OpenGL является только воспроизводящей (Rendering), и занимается только отображением 3D обьектов, она не работает с устройствами ввода (клавиатуры, мыши). Также она не поддерживает менеджер окон.

OpenGL имеет хорошо продуманную внутреннюю структуру и довольно простой процедурный интерфейс. Несмотря на это с помощью OpenGL можно создавать сложные и мощные программные комплексы, затрачивая при этом минимальное время по сравнению с другими графическими библиотеками.

1. OpenGL
   1. Назначение

Основная идея OpenGL: графическая библиотека должна быть аппаратно независимой, но в то же время использовать аппаратные ускорители, если они доступны. Кроме того, этот язык с самого начала предусматривал механизм расширения и гибкости - по мере того как расширения становились "общепринятыми", они становились частью следующего релиза.

Для любого протокола, будь то сетевой протокол или язык описания сцен, важным является вопрос уровня абстракции - то есть того, на каком уровне работает данная система или протокол, что является входными данными и что выходными, какие компоненты будут взаимодействовать в качестве поставщиков и приемников данных. Говоря попросту, нужно определиться по вопросу "что мы делаем и чего не делаем".

Создатели OpenGL планировали свой язык с явным намерением создать "виртуальный графический акселератор", так чтобы примитивы OpenGL максимально соответствовали примитивам современных графических карт и требовали минимум кода для трансляции из одной системы команд в другую. Фактически большинство современных графических процессоров (обычно называемых видеокартами, хотя к видео они имеют лишь касательное отношение) напрямую воспринимают OpenGL как язык входного уровня без какой-либо (или с минимумом) трансляции.

OpenGL оперирует графическими примитивами "начального уровня", такими как точки трехмерного пространства (вертексы, вершины), отрезки прямых, выпуклые полигоны и растровые изображения. Поддерживаются аффинные и проективные преобразования, вычисление освещения. К "продвинутым" функциям можно отнести мэппинг текстур (натягивание битовых карт на трехмерные поверхности) и антиалиасинг (сглаживание цветовых переходов - как локальное, в рамках отдельного объекта, так и глобальное, по всей сцене). Предполагается, что приложение более высокого уровня будет выполнять операции, которых недостает в OpenGL,- например, декомпозицию невыпуклых полигонов.

* 1. Архитектура

OpenGL имеет два конвейера для пиксельных данных и вершинных операций, то есть для операций с векторными данными. Каждый из потоков обрабатывается отдельно, до тех пор пока это возможно - то есть до стадии мэппинга, когда пиксельные растры как фактуры "натягиваются" на плоскости и более сложные поверхности.

Первый этап - аппроксимация кривых и поверхностей вычислением полиномов от входных значений. Второй проход оперирует с примитивами типа точек, отрезков и полигонов - они преобразуются по правилам аффинных преобразований, совмещаются и сцена отсекается в подготовке к растрированию. Растрирование в качестве результата создает список объектов (точек, отрезков и треугольников) в двумерной плоскости. Над отдельными объектами может быть выполнена операция раскрашивания, градиентной заливки или применения мэппинга, то есть наложения фактуры.

Готовые фрагменты окончательно обрабатываются перед тем, как они реально будут внесены в frame buffer. В частности, фрагменты сортируются зависимо от значений "глубины" - и эти значения сравниваются с известными на предмет рекомпозиции. Применение блендинга приводит к тому, что прозрачные фрагменты принимают цвет, состоящий из их собственного и цветов "ниже лежащих" фрагментов. Дополнительно может быть реализовано маскирование и другие эффекты.

Пиксельный процессор в ходе растрирования встраивает двумерные битовые фрагменты прямо в кадр. Часть готового кадра также может быть прочитана для повторного использования как массив пикселей - так что данные, отображаемые в буфере, могут стать частью других сцен.

* 1. Базовый функционал и возможности

Ниже перечислены основные функции OpenGL. По ним вы можете судить об уровне языка и используемых примитивах.

Альфа-канал. Позволяет делать предметы прозрачными, уровень прозрачности от 0 до 100%.

Антиалиасинг. Сглаживание цветовых переходов, более реалистическое изображение.

Дополнительный буфер для 2,5-мерных эффектов, спецэффектов и глобального сглаживания по всей сцене.

Градиентная заливка (gouraud shading). Линейно-градиентная заливка полигонов и отрезков.

Графические примитивы. В пространстве: точка, отрезок, полигон, битовое изображение или изображение в другом формате.

Двойная буферизация. Применяется для сглаживания эффектов анимации, когда новое изображение строится на заднем плане и потом отображается целиком. При этом пользователь не видит самого процесса создания изображения в несогласованном состоянии, например различных "изнанок объектов", "дыр в пространстве", "граней мира" и подобных нежелательных деталей.

Заливка и освещенность фактур. К фактурам применяются эффекты освещенности и затенения в зависимости от характеристик "материала".

Маскирование. Можно маскировать некоторые цвета по трафарету.

Массивы вершин. Новая возможность версии 1.5 - для повышения производительности задавать вершины массивами, а не отдельно.

Обратная связь. Данные после растрирования могут быть возвращены в приложение вместо передачи из/в frame buffer или параллельно с ней.

Пересечения. Автоматическое определение того, пересекает ли тот или иной объект заданный пространственный регион.

Пиксельные операции. Масштабирование и другие аффинные преобразования битовых образов.

Поддиапазоны. Возможность работать с частью матрицы вершин; применяется как метод оптимизации.

Полиноминальные операции. Поддержка неравномерных рациональных би-сплайнов для описания кривых поверхностей.

Полноцветное отображение. Представление в режиме RGBA, то есть тремя цветами и значением альфа-канала. Начиная с версии 1.2, поддерживаются также схемы BGRA и схемы с упакованными цветами для быстрой обработки популярных типов графических файлов.

Пространственные преобразования. Масштабирование, вращение и перемещение объектов в пространстве.

Режим индексированных цветов. Представление цветов не RGB-триплетами, а индексами в таблице цветов. Применяется для сжатия размена изображений "по глубине цвета" и эффектов быстрой замены одного цвета другим.

Режим прямой отрисовки. Рисование по мере поступления команд, без использования списков отображения.

Смешивание цветов. Позволяет устанавливать несколько режимов наложения одного изображения на другое. С помощью этой операции, в частности, реализуется альфа-канал и другие эффекты.

Список отображения (displa list). Пакет описания объектов сцены для предварительного разбора и оптимизации кэширования.

Текстуры. Наложение двухмерных изображений на объемные поверхности для придания сцене реализма. Начиная с версии 1.2, поддерживаются трехмерные текстуры. Кроме того, начиная с этой же версии, поддерживается уровень текстур, позволяющий загружать их только до определенной степени детализации - в целях экономии памяти текстур.

Z-буффер. Понятие об удалении объектов и их частей от наблюдателя, часть алгоритма удаления скрытых поверхностей.

1. Immediate mode
2. Инициализация

Для упрощения работы с OpenGL подключим вспомогательные библиотеки:

GLU- библиотека с функциями для работы со сплайнами, реализованы дополнительные операции над матрицами и дополнительные виды проекций),

GLAUX – библиотека с функциями для работы с окнами, клавиатурой и мышкой.

Создаем простое окно, используя функционал библиотек: GLU, GLAUX, и также саму сцену OpenGL: glViewport(0, 0, width, height).

Задаем задний фон нашей сцены: glClearColor(0.0f, 0.0f, 0.0f, 1.0f);

Чистую глубину буфера: glClearDepth(1.0f);

Устанавливаем флаг, что будет происходить проверка глубины буфера: glEnable(GL\_DEPTH\_TEST);

Чтобы задать угол обзора можно воспользоваться функцией: gluPerspective(45.0f, (GLfloat)width / (GLfloat)height, 0.1f, 100.0f);

И при каждом фрейме будем очищать буфер цвета и глубины: glClear(GL\_COLOR\_BUFFER\_BIT | GL\_DEPTH\_BUFFER\_BIT);

Результат исследования на данном этапе: [step-1](https://github.com/DanielDaVinci/OpenGL_Education/tree/step-1).

1. Отображение полигонов
2. Отображение цветов
3. Трансформация
4. Создание 3D моделей
5. Наложение текстур
6. Сore profile
7. Инициализация
8. Отображение полигонов
9. Создание шейдеров
10. Наложение текстур
11. Трансформация
12. Создание 3D моделей
13. Преобразования пространств
14. Трансформация камеры

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Обучающие лекции и материалы для OpenGL immediate mode - <http://pmg.org.ru/nehe/index.html>

Обучающие лекции и материалы для OpenGL profile core-

<https://learnopengl.com/>

<https://habr.com/ru/articles/310790/>