*Федеральное государственное бюджетное образовательное учреждение*

*высшего профессионального образования*

|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | ***«Московский государственный технический университет  имени Н.Э. Баумана»***  ***(МГТУ им. Н.Э. Баумана)*** |

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ЭВМ И ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

**РАСЧЁТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**к курсовому проекту на тему:**

ПОСТРОЕНИЕ СУПЕРПОЗИЦИИ ПРИЗМ

Студент группы ИУ7-53 **\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_**Гусев М.А.

(Подпись, дата) (И.О.Фамилия)

Руководитель курсового проекта \_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_**Майков К.А.

(Подпись, дата) (И.О.Фамилия)

Москва, 2017

# Задание на курсовой проект

Во время работы над курсовым проектом нужно создать программное обеспечение для построения реалистичной сцены, являющейся суперпозицией объектов, построенных на основе призм.

Для этого необходимо решить множество задач:

1. Обеспечить возможность редактирования пользователем сцены (реализовать добавление и удаление объектов, источников освещения, изменение их характеристик)
2. Выбрать наиболее подходящее для построения реалистического изображения решение
3. Эффективно реализовать его для достижения максимально возможной производительности
4. Протестировать данное решение и выявить границы его применимости при различных параметрах сцены.

Оглавление

[Задание на курсовой проект 2](#_Toc500010260)

[Реферат 5](#_Toc500010261)

[Введение 6](#_Toc500010262)

[1. Аналитический раздел 7](#_Toc500010263)

[1.1 Математическая модель объектов сцены 7](#_Toc500010264)

[1.2 Обоснование выбора алгоритмов 8](#_Toc500010265)

[1.2.1 Выбор алгоритма удаления невидимых линий 9](#_Toc500010266)

[1.2.2 Выбор модели освещения 11](#_Toc500010267)

[1.2.3 Вывод 13](#_Toc500010268)

[2. Конструкторский раздел 14](#_Toc500010269)

[2.1 Вспомогательные математические соотношения 14](#_Toc500010270)

[2.1.1 Нахождение уравнения плоскости по трем точкам 14](#_Toc500010271)

[2.1.2 Определение принадлежности точки грани 14](#_Toc500010272)

[2.1.3 Нахождение отраженного от грани луча 15](#_Toc500010273)

[2.1.4 Нахождение преломленного гранью луча 16](#_Toc500010274)

[2.1.5 Расчет освещенности точки грани от первичных источников света 17](#_Toc500010275)

[2.2 Алгоритм трассировки лучей с учетом глобальной модели освещения 18](#_Toc500010276)

[3. Технологический раздел 19](#_Toc500010277)

[3.1 Выбор языка программирования и среды разработки 19](#_Toc500010278)

[3.2 Структура приложения 20](#_Toc500010279)

[3.3 Интерфейс приложения 25](#_Toc500010280)

[3.3.1 Меню верхней панели 25](#_Toc500010281)

[3.3.2 Поле управления призмами 26](#_Toc500010282)

[3.3.3 Поле управления источниками света 27](#_Toc500010283)

[3.3.4 Поле выбора способа отображения сцены 28](#_Toc500010284)

[3.3.5 Схема управления с помощью клавиатуры 28](#_Toc500010285)

[4. Исследовательский раздел 29](#_Toc500010286)

[4.1 Технические характеристики ЭВМ, на которой произведены тесты 29](#_Toc500010287)

[4.2 Зависимость времени отображения от количества объектов на сцене 29](#_Toc500010288)

[4.3 Зависимость времени отображения от количества граней в одном объекте. 30](#_Toc500010289)

[Заключение 32](#_Toc500010290)

[Список литературы 33](#_Toc500010291)

# Реферат

Во время проведения данной курсовой работы исследовался алгоритм трассировки ручей с использованием глобальной модели освещения. Целью работы являлось определение возможностей алгоритма, его характеристик и границ применимости.

Для этого был разработан программный продукт, реализующий данный алгоритм для отображения суперпозиций объектов на основе призм. В результате были получены и описаны зависимости времени работы по отображению сцены от количества граней в объектах и от количества объектов и сделаны выводы о возможностях данного алгоритма и его границах применимости.

# Введение

В современном мире задача построения реалистических изображений находит широкое применение. Техническое развитие позволяет просматривать и создавать высококачественные изображения, а также видео, на домашних компьютерах и мобильных устройствах, скорость передачи данных в интернете, а также его распространение, растут. Компьютерную графику используют в первую очередь огромные рынки развлечений: кино и видеоигры. Также графика используется в промышленности, исследовательской деятельности и транспортной индустрии: например, для создания реалистичных тренажеров для обучения пилотов гражданской и военной авиации.

Для различных областей применения перед компьютерной графикой ставятся различные задачи. При создании анимационных фильмов крупными западными студиями в приоритет ставится качество изображения, так как особенностью фильмов является то, что все их кадры рассчитываются при создании, а не непосредственно при проигрывании в кинотеатре, а у студии есть технические и материальные средства, позволяющие обрабатывать каждый отдельный кадр большое количество времени, но при этом укладываться в установленные сроки производства. Совершенно по-другому дело обстоит в индустрии видеоигр и обучающих симуляторов: на первое место выходят требования производительности, поскольку каждый кадр обрабатывается в реальном времени, и для комфортного восприятия человеческим глазом требуется рендер как минимум 15 кадров в секунду (граница восприятия глазом изображений как части непрерывной анимации, а не отдельных кадров, а лучше 24 (европейский телевизионный стандарт) или более. Это налагает ограничения на качество изображения и используемые алгоритмы расчета.

# Аналитический раздел

## Математическая модель объектов сцены

Для удобства построения сцены в качестве объектов сцены выбраны призмы с осевой симметрией и тела, построенные на их основе. Каждый объект имеет следующие параметры:

1. Количество боковых граней (не менее трех).
2. Координаты.
3. Высоту
4. Радиусы описанных вокруг каждого из двух оснований окружностей.
5. Коэффициент диффузного отражения.
6. Коэффициент зеркального отражения.
7. Коэффициент преломления.

Источники света являются точечными и имеют следующие параметры:

1. Координаты.
2. Цвет.

Глобальные координаты призмы соответствуют положению центра нижнего основания. Также у каждого объекта имеется вектор направления, по умолчанию направленный обратно оси OZ глобальной системы координат. Он нужен для описания ориентации объектов в пространстве. Этих параметров достаточно для того, чтобы однозначно задать все объекты, находящиеся на сцене.

Во время работы программы для уменьшения количества вычислений призмы задаются координатами вершин их граней.

Ниже представлен алгоритм нахождения координат вершин оснований призматических объектов. Количество точек равно количеству граней.

1. Точки нижнего основания имеют координаты
2. Точки верхнего основания имеют координаты
3. Первая точка основания имеет координаты конца вектора направления призмы, отложенного от середины нижнего основания призмы (опорной точки). Координаты i-й точки находятся с помощью сложения данного вектора, повернутого на вокруг оси OY, с координатами середины нижнего основания.

## Обоснование выбора алгоритмов

Целью данной работы является построение наиболее качественного и реалистического изображения и исследование границы применимости использованных решений, в связи с этим следует выбрать наиболее подходящие алгоритмы для достижения наилучшего, с визуальной точки зрения, результата.

Задача построения реалистического изображения разбивается на следующие подзадачи (А.В., 2017):

1. Построение математической модели сцены.
2. Задание позиций и характеристик тел и источников освещения, присутствующих на сцене.
3. Задание положения наблюдателя, картинной плоскости, размеров окна вывода.
4. Преобразование глобальных координат в координаты наблюдателя.
5. Отсечение объектов по границам пирамиды видимости.
6. Вычисление двумерных перспективных проекций на картинную плоскость видимых объектов сцены.
7. Удаление невидимых линий и поверхностей при заданном положении наблюдателя.
8. Освещение и затенение объектов.
9. Вывод полученного изображения на экран растрового дисплея.

На качество изображения влияют алгоритмы удаления невидимых линий

и поверхностей, а также алгоритмы освещения и затенения.

Так как объектами сцены в нашей задаче являются многогранники, не следует рассматривать алгоритмы освещения, визуально сглаживающие поверхности, к примеру алгоритмы Гуро и Фонга.

### 1.2.1 Выбор алгоритма удаления невидимых линий

Рассмотрим основные решения:

1. Алгоритм Робертса (Д., 1989).
2. Алгоритм, использующий z-буфер (Д., 1989).
3. Алгоритм, использующий трассировку лучей (www17).

**Алгоритм Робертса** является алгоритмом отсечения невидимых граней и ребер. В нем используются уравнения плоскостей, пересечением которых образовано тело. применим к данной задаче из-за свойства выпуклости объектов сцены, однако не является оптимальным из-за невозможности учесть прозрачность объектов сцены, а также затенение, и, следовательно, построить реалистичную модель освещения.

**Алгоритм, использующий z-буфер**, в простейшей реализации описывается следующим образом:

1. Создание буфера, хранящего цвет каждого пикселя экрана.
2. Переводе координат объектов сцены в экранные координаты.
3. Нахождение для каждой точки экрана наиболее близкой к ней точки одного из тел.
4. Занесении в буфер цвета данной точки.

Он используется в графических библиотеках, таких как **OpenGL**, поскольку обладает следующими преимуществами:

1. Время его работы пропорционально размерам экрана и, в отличии от других распространенных алгоритмов, меньше зависит от сложности изображаемой сцены.
2. С помощью модификаций возможно учесть прозрачность объектов и затенение, что позволяет построить реалистичное изображение.
3. Возможность добиться стабильной работы в реальном времени.

Недостатками являются:

1. Сложность модификации для учета прозрачности и затенения объектов, кратное увеличение затрат памяти при реализации данных модификаций.
2. Невозможность учета глобальной модели освещения

**Алгоритм, использующий трассировку лучей**, используется в основном для создания анимационных фильмов, видеоигр и компьютерной графики в художественных и обучающих фильмах.

В простейшем виде суть алгоритма заключается в следующем:

1. Экран представляется как физический объект, находящийся на некотором удалении от наблюдателя
2. В каждую точку экрана пускается луч
3. Для каждого луча ищется пересечение с охватывающими оболочками объектов сцены
4. В случае нахождения пересечения с охватывающей оболочкой, ищется пересечение луча с самим объектом
5. Если пересечение есть, для данной точки рассчитывается суммарная интенсивность света от каждого из источников.

Данный алгоритм обладает следующими преимуществами:

1. Возможно учесть глобальную модель освещения сцены
2. Возможно построение освещения и затенения в полном соответствии с законами геометрической оптики

Главным недостатком является огромное количество вычислений, и, следовательно, невозможность полноценной работы в реальном времени (с частотой обновления изображения более 15 кадров в секунду).

**Вывод:**

**Целесообразно выбрать алгоритм трассировки лучей, как наиболее физически достоверный, и дающий наиболее качественные результаты при построении изображений.**

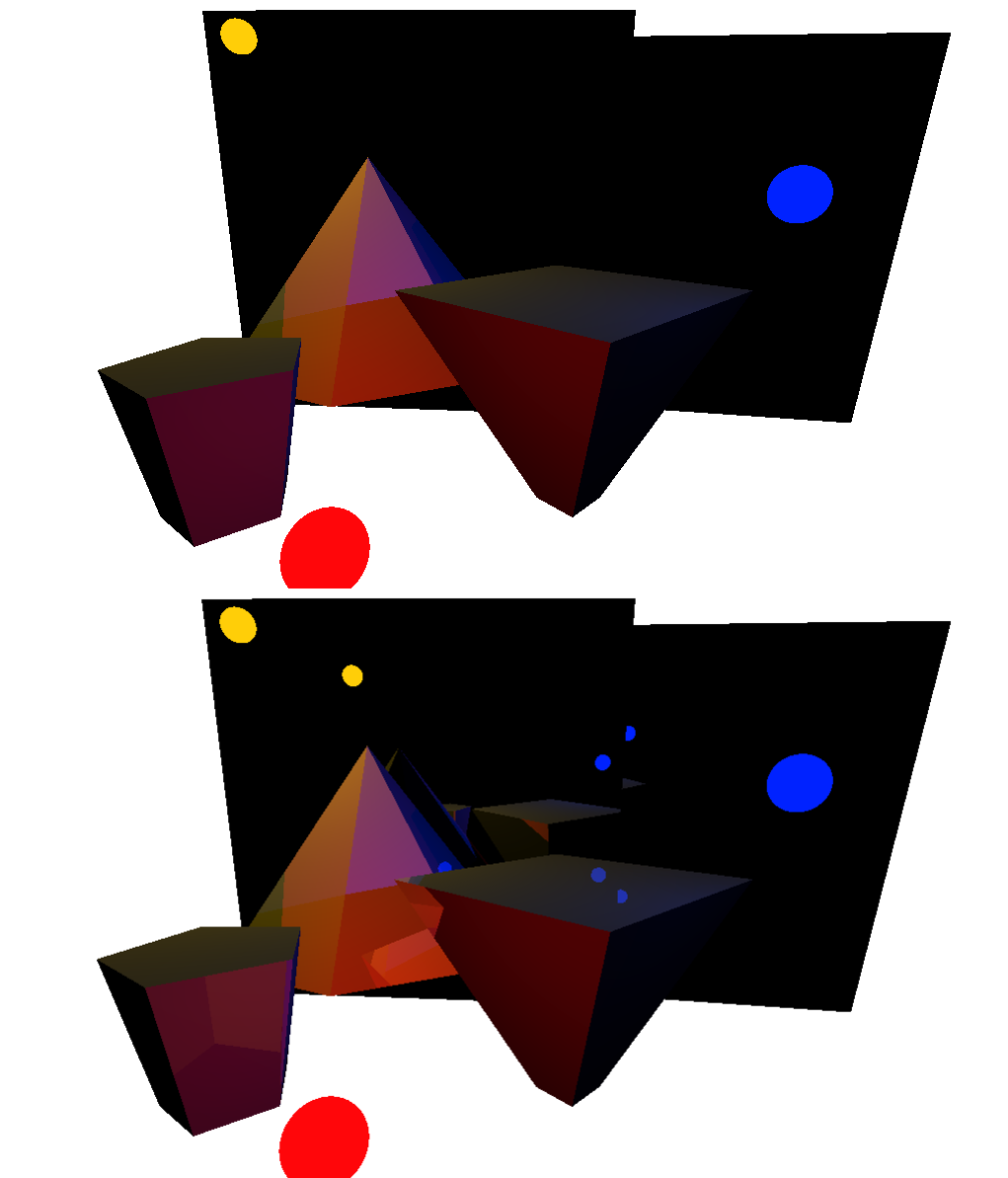
**Проблема невозможности работы приложения в реальном времени была решена добавлением возможности отображения сцены в виде каркасных моделей, позволило быстро изменять суперпозицию и положение камеры, не затрачивая ресурсы компьютера на отображение промежуточных состояний.**

### ****1.2.2 Выбор модели освещения****

Существует локальная и глобальная модели освещения. Локальная учитывает лишь первичные источники света, а глобальная еще и вторичные (отражающие и преломляющие поверхности). Выбранный алгоритм позволяет с легкостью это сделать:

1. Для каждого луча, попавшего в объект сцены, строится два дочерних: преломленный луч и отраженный луч, полученные с помощью законов геометрической оптики
2. Для каждого из дочерних лучей операция повторяется до тех пор, пока все дочерние лучи нижнего уровня не уйдут в бесконечность или не попадут в первичные источники света, или пока не будет достигнута максимальная глубина расчета (данная мера введена для предотвращения бесконечного зацикливания программы)
3. После построения полного «дерева лучей» производится расчет интенсивности с самого нижнего уровня до самого верхнего.

*Рис 1. Сравнение сцены с локальной и глобальной моделью освещения*



**Вывод:**

Для достижения наиболее физически достоверного и реалистичного изображения целесообразно выбрать глобальную модель освещения.

### ****1.2.3 Вывод****

После рассмотрения существующих алгоритмов были выл выбран алгоритм, использующий трассировку лучей с глобальной моделью освещения для итогового отображения суперпозиции и алгоритм, отображающий объекты сцены в каркасном представлении для промежуточных состояний, то есть для состояний, которые сцена принимает во время построения требуемой суперпозиции.

# Конструкторский раздел

В данном разделе описывается реализация алгоритмов, примененных в данной задаче. Вначале описаны вспомогательные математические формулы, используемые для расчетов.

## Вспомогательные математические соотношения

### Нахождение уравнения плоскости по трем точкам

Уравнение плоскости необходимо для определения угла падения луча на грань, нахождения пересечения луча с гранью.

Входные данные: координаты трех точек p1, p2, p3 (первые три точки, задающие грань).

Выходные данные: коэффициенты для уравнения Ax + By + Cz + D = 0.

1. Получаем координаты векторов v1 и v2 путем вычитания координат первой точки из координат второй и третьей точки соответственно.
2. Найдем коэффициент А по формуле: А = v1.y \* v2.z – v1.z \* v2.y.
3. Найдем коэффициент B по формуле: B = v1.z \* v2.x – v1.x \* v2.z.
4. Найдем коэффициент C по формуле: C = v1.x \* v2.y – v1.y \* v2.x.
5. Найдем коэффициент D по формуле: D = –(p1.x \* A + p1.y \* B + p1.z \* C).
6. Конец.

### Определение принадлежности точки грани

Замечание: в условиях данной задачи все грани являются выпуклыми, что позволяет использовать данный метод.

Входные данные: координаты точки, координаты вершин грани.

Выходные данные: подтверждение или опровержение принадлежности точки грани.

1. Подставить координаты точки в уравнение плоскости, которой принадлежит грань. В случае невыполнения данного уравнения точка не принадлежит грани.
2. Для каждого ребра вычислить координаты двух векторов: вектора из начальной точки ребра в конечную и вектора из начальной точки ребра в исследуемую точку.
3. Вычислить координаты проверочного вектора путем векторного произведения этих векторов.
4. Если рассматриваемое ребро не первое, произвести скалярное произведение полученного вектора с вектором, полученным при рассмотрении предыдущего ребра. Если оно отрицательное – точка лежит вне грани.
5. В случае, если все ребра проверены – точка внутри грани.
6. Конец.

### Нахождение отраженного от грани луча

Входные данные: начальная и конечная точки луча, уравнение плоскости

Выходные данные: начальная и конечная точки отраженного луча.

1. Найти точку пересечения луча и плоскости. Эта точка является начальной точкой отраженного луча.
2. Найти вектор normal нормали к плоскости (коэффициенты A, B, C уравнения данной плоскости являются, соответственно, его координатами X, Y, Z).
3. Вычислить координаты входного луча beam.
4. Вычислить угол между данными векторами α.
5. Если угол больше 90 градусов – умножить координаты вектора нормали на -1.
6. Привести normal к размеру, при котором начальная точка луча находится на том же удалении от плоскости, как и конечная точка нормали: .
7. Вычислить координаты вектора из начальной точки первого луча в конечную точку отраженного луча: .
8. Конечная точка отраженного луча – это начальная точка луча + .
9. Конец.

### Нахождение преломленного гранью луча

Входные данные: начальная и конечная точки луча, уравнение плоскости, коэффициент преломления k.

Выходные данные: начальная и конечная точки преломленного луча.

1. Найти точку пересечения луча и плоскости. Эта точка является начальной точкой отраженного луча.
2. Найти вектор normal нормали к плоскости (коэффициенты A, B, C уравнения данной плоскости являются, соответственно, его координатами X, Y, Z).
3. Вычислить координаты входного луча beam.
4. Вычислить угол между данными векторами α.
5. Если угол больше 90 градусов – умножить координаты вектора нормали на -1.
6. Вычислить синус преломленного угла β: . (И.В., 1970) Если он больше 1 - произошло явление полного внутреннего отражения, прекратить расчет преломленного луча.
7. Привести normal к размеру, при котором начальная точка луча находится на том же удалении от плоскости, как и конечная точка нормали: .
8. Вычислить координаты вектора из начальной точки первого луча в конечную точку преломленного луча: .
9. Конечная точка преломленного луча - это конечная точка луча .
10. Конец.

### Расчет освещенности точки грани от первичных источников света

Входные данные: координаты и интенсивности источников света, координаты объектов сцены

Выходные данные: интенсивность света в рассматриваемой точке.

1. Для каждого источника света:

а) Проверить прямую видимость из источника света данной точки (проверить, что луч из источника к данной точке на своем протяжении не пересекает другие тела). Если видимости нет – данный источник игнорируется.

б) Вычислить угол α между плоскостью и падающим лучом.

в) Вычислить итоговую интенсивность как , где R – расстояние от источника до точки, k – коэффициент диффузионного отражения.

2) Произвести сложение полученных значений.

3) Конец.

## 2.2 Алгоритм трассировки лучей с учетом глобальной модели освещения

Входные данные: объекты сцены, размеры и положение экранной плоскости.

Выходные данные: изображение.

Для каждой точки экранной плоскости:

1. Получить луч из положения наблюдателя в данную точку (координатами начальной точки луча будут координаты наблюдателя, конечной – координаты рассматриваемой точки экранной плоскости).
2. Для каждой призмы проверить луч на пересечение со сферической оболочкой объекта сцены, если оно имеется, проверить пересечение с самим объектом. Если пересечение найдено, и точка пересечения ближе ранее найденной, запомнить данную точку пересечения, грань, с которой имеется пересечение и оптические характеристики грани, либо, в случае пересечения источника света – оптические характеристики источника.
3. Если пересечения нет – закрасить точку цветом фона. Если пересечен источник света – закрасить точку цветом источника.
4. Если пересечена грань – построить дерево отраженных и преломленных лучей.
5. Произвести расчет дерева. Для каждой вершины:

5.1) Если луч пересекает источник, интенсивностью вершины является интенсивность источника.

5.2) Если луч пересекает грань, то цвет грани складывается из интенсивности полученного расчетом освещения от первичных источников света сложенного с интенсивностями преломленного и отраженного лучей, умноженных соответственно на коэффициенты преломления и отражения.

6) Закрасить точку рассчитанным цветом.

7) Конец.

# Технологический раздел

В данном разделе описан выбор языка программирования и среды разработки, структуры данных и интерфейс приложения.

## Выбор языка программирования и среды разработки

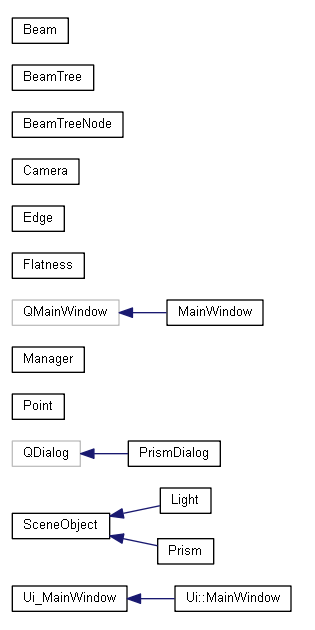
В качестве языка программирования выбран С++. Его достоинствами являются:

1. «Прозрачная» работа с памятью и отсутствие «сборщика мусора». Программист всегда достоверно знает, на что тратятся память и вычислительные ресурсы и в каком размере.
2. Более высокий уровень удобства программирования по сравнению с C, возможность программирования с использованием ООП подхода.

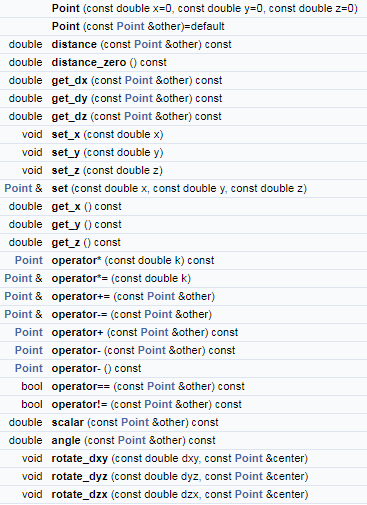
В качестве среды разработки использована Qt, т.к. она имеет широкие возможности для проектирования графического интерфейса, удобную встроенную систему сборки проекта, а также из-за того, что код, написанный с использованием Qt, не зависит от используемой операционной системы, и, к примеру, не требуется реализации различных обработчиков нажатия на клавиши клавиатуры для Linux и Windows.

## Структура приложения

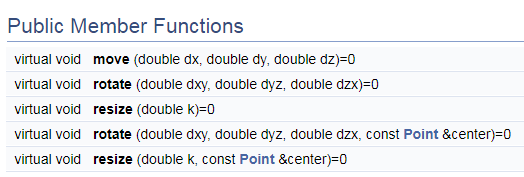
*Рис 2. Общий список классов приложения. Отношения наследования заданы стрелками от класса потомка к классу-родителю.:*



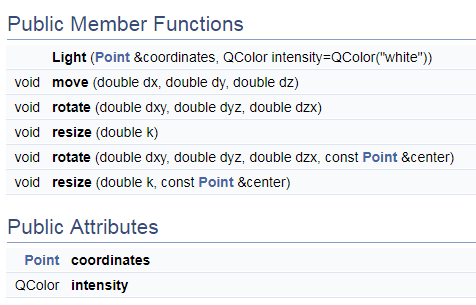
*Рис. 3 Класс Point*



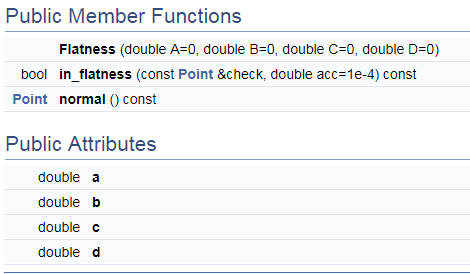
*Рис.4 Класс SceneObject*



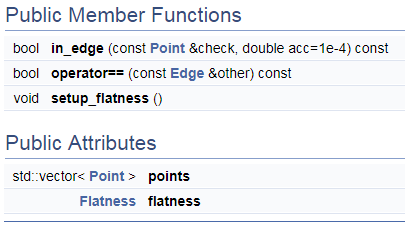
*Рис 5. Класс Prism*



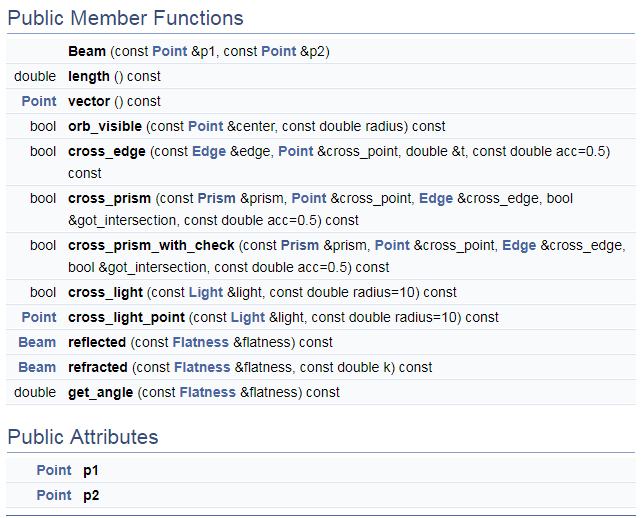
*Рис. 6 Класс Flatness*



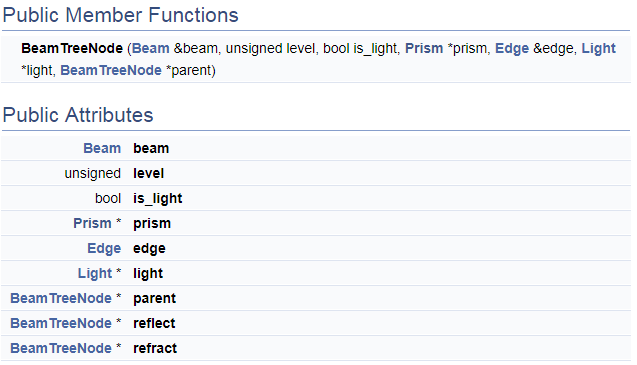
*Рис. 7 Класс Edge*



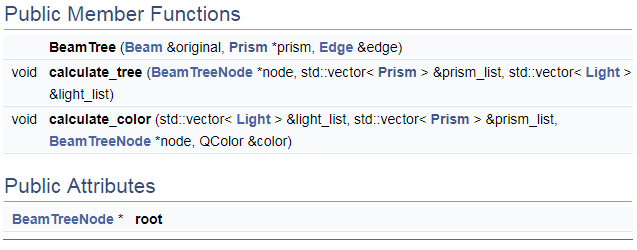
*Рис. 8 Класс Beam*



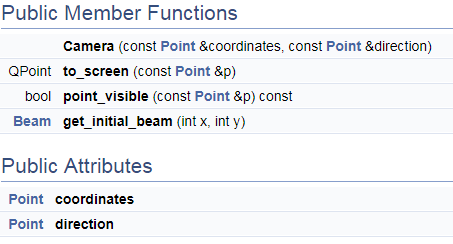
*Рис. 9 Класс BeamTreeNode*



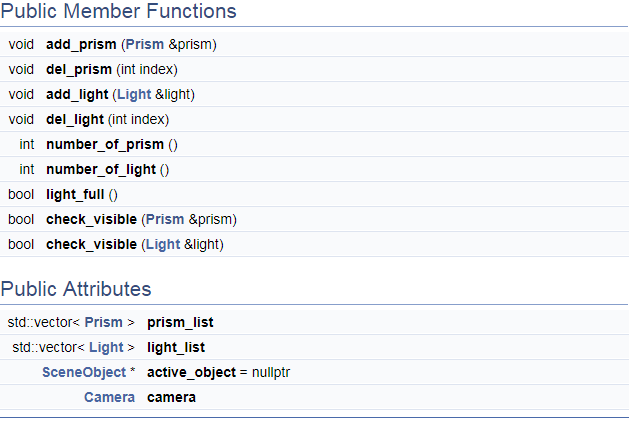
*Рис. 10 Класс BeamTree*



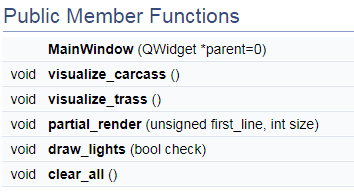
*Рис. 11 Класс Camera*



*Рис. 12 Класс Manager*

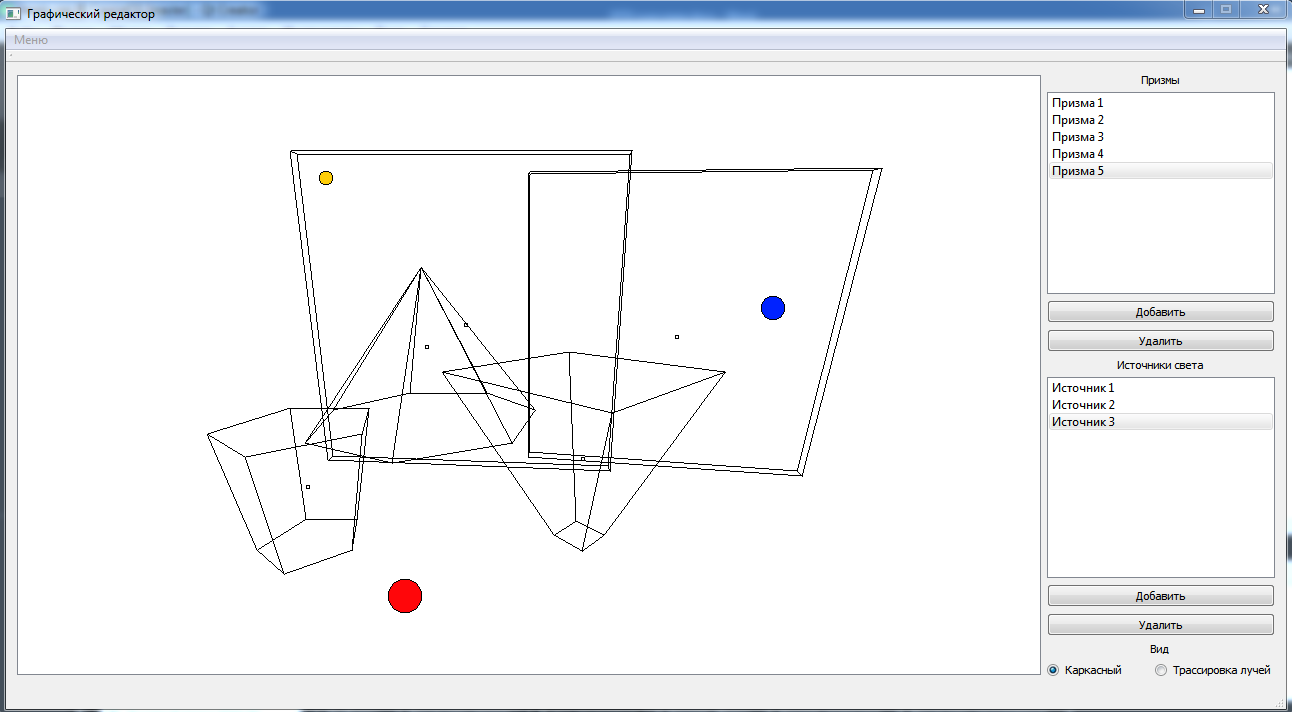


*Рис. 13 Класс MainWindow*



## Интерфейс приложения

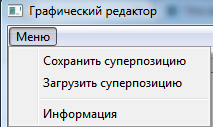
*Рис. 14 Общий вид графического интерфейса программы*



Программа состоит из экрана, на который выводится изображение, поля управления призмами, состоящего из списка призм и кнопок для их добавления и удаления, аналогичного поля управления источниками света, полем выбора способа отображения сцены, а также выпадающим меню на верхней панели с возможностью сохранения и загрузки суперпозиции и кнопкой получения общей информации о приложении.

### Меню верхней панели

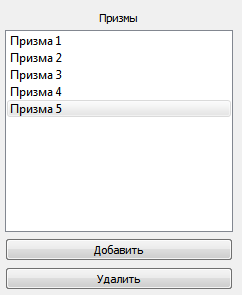
*Рис. 15 Меню верхней панели*



Кнопки «Сохранить суперпозицию» и «Загрузить суперпозицию» вызывают файловый менеджер, который позволяет создать или выбрать исходный файл в формате pzm. При нажатии на кнопку «Информация» будет выведено окно с кратким описанием предназначения приложения.

### Поле управления призмами

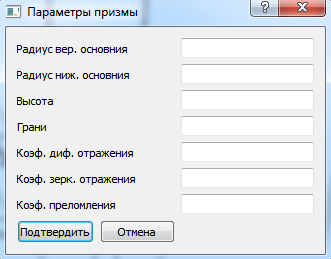
*Рис. 16 Поле управления призмами*



Данное поле состоит из интерактивного списка призм и кнопок добавления и удаления призм. При нажатии на одно из полей списка призм объект, ассоциированный с данным полем списка, становится активным (управляемым при одиночных преобразованиях с помощью устройства управления.

При нажатии на кнопку «Добавить», вызывается диалоговое окно, в котором нужно задать параметры создаваемой призмы.

*Рис. 17 Диалоговое окно создания призмы*

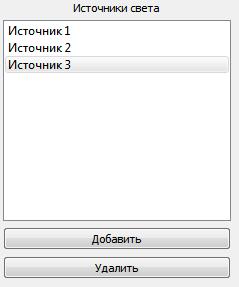


Радиус одного из оснований может быть равен нулю, высота должна быть положительна, количество боковых граней не может быть менее 3, а сумма коэффициентов не должна превышать 1.

При нажатии на кнопку «Удалить» производится удаление со сцены выделенной в списке призмы.

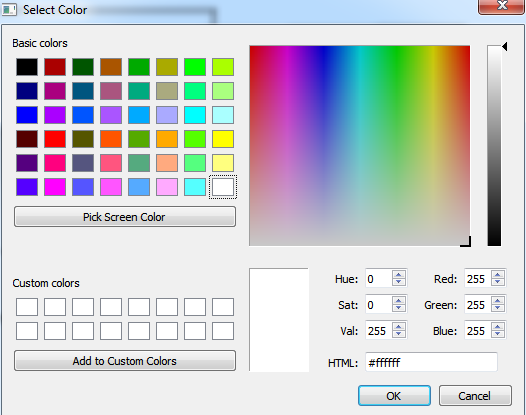
### Поле управления источниками света

*Рис 18. Поле управления источниками света*



Данное поле аналогично предыдущему, за исключением ограничения количества источников света на 3 (при попытке добавления нового источника при наличии трех источников освещения на сцене будет выведено сообщение об ошибке), а также типа диалогового окна при нажатии кнопки «Добавить».

*Рис 7. Диалоговое окно добавления нового источника света*



### Поле выбора способа отображения сцены

*Рис. 19 Поле выбора способа отображения сцены*

https://i.gyazo.com/083c73a016b123991f1ffe5cc8200e6f.png

В данном поле можно выбрать один из видов отображения сцены. При изменении типа отображения сцены происходит ее немедленное отображение в выбранном типе.

### Схема управления с помощью клавиатуры

Примечание: для команд переноса и поворота при удержании клавиши **SHIFT** активным объектом становится камера. Для команд изменения масштаба удержание **SHIFT** позволяет масштабировать все объекты сцены относительно центра сцены.

Клавиши **W/S** предназначены для перемещения активного объекта по оси OY.

Клавиши **A/D** предназначены для перемещения активного объекта по оси OX.

Клавиши **Q/E** предназначены для перемещения активного объекта по оси OZ.

Клавиши **I/K** предназначены для вращения объекта вокруг оси, параллельной оси OX и проходящей через центр объекта

Клавиши **J/L** предназначены для вращения объекта вокруг оси, параллельной оси OY и проходящей через центр объекта

Клавиши **O/U** предназначены для вращения объекта вокруг оси, параллельной оси OZ и проходящей через центр объекта

Клавиши **T/U** предназначены для масштабирования объекта относительно его центра.

# Исследовательский раздел

В данном разделе проводится исследование зависимости времени отображения сцены от количества объектов на сцене и количества граней у единственного объекта. Вычисления были произведены параллельно на 60 потоков.

## Технические характеристики ЭВМ, на которой произведены тесты

1. Процессор Intel Core i5-2435M 2.40 GHz.
2. 4Gb RAM.
3. Операционная система Windows 7.

## Зависимость времени отображения от количества объектов на сцене

В данном исследовании каждый объект являлся пятигранной призмой одинаковых размеров и имеющих коэффициенты диффузионного, зеркального отражений и преломления по 0.3. На сцене присутствовало три источника освещения.

|  |  |  |
| --- | --- | --- |
| Количество объектов на сцене | Время отображения если объекты в одной точке, с | Время отображения, если объекты в стоят рядом, с |
| 1 | 3.245 | 3.245 |
| 2 | 4.039 | 4.542 |
| 3 | 4.532 | 5.022 |
| 4 | 4.877 | 5.647 |
| 5 | 5.032 | 6.041 |
| 6 | 5.123 | 8.485 |
| 7 | 5.256 | 9.285 |
| 10 | 5.502 | 11.699 |
| 15 | 6.87 | 17.50 |

**Вывод:** время отображения сцены линейно зависит от площади занимаемого объектами пространства и линейно зависит от количества объектов на сцене.

## Зависимость времени отображения от количества граней в одном объекте.

|  |  |
| --- | --- |
| Количество граней | Время отображения, с |
| 5 | 3.163 |
| 6 | 3.276 |
| 7 | 3.527 |
| 8 | 3.506 |
| 10 | 4.163 |
| 50 | 6.917 |
| 100 | 10.644 |
| 200 | 18.699 |
| 500 | 46.221 |

**Вывод:** время отображения линейно зависит от количества граней в объектах.

# Заключение

В результате проделанной работы был исследован алгоритм трассировки лучей с глобальной моделью освещения. Были установлены следующие зависимости:

1. Время визуализации прямо пропорционально площади занимаемой суперпозицией части экрана.
2. Время визуализации прямо пропорционально количеству граней объектов, участвующих в суперпозиции.

Первая зависимость следует из того, что с ростом занимаемой суперпозицией площади экрана кратно растет количество лучей от положения наблюдателя в точки экранной плоскости, для которых следует проводить проверку пересечения ими объектов сцены.

Вторая зависимость следует из того, что при расчете ближайшей точки пересечения лучом объекта сцены происходит проверка пересечения данным лучом каждой грани объекта, то есть количество вычислений прямо пропорционально количеству граней данного объекта.

Так как данный алгоритм позволяет добиться максимальной физической достоверности отображаемой сцены, его имеет смысл применять для отображения сложных сцен. Границы применимости зависят от решаемых задач. Для задач обучения (например, для демонстрации работы законов геометрической оптики) изображение должно быть построено в течении одной минуты, что, для используемой в данной работе ЭВМ накладывает ограничение в 660 граней на сцену. Для задач создания реалистичных анимационных фильмом нет существенных ограничений по временным затратам и границы применимости определяются экономическими возможностями компании.

# Список литературы

1. Алгоритмические основы машинной графики [Книга] / авт. Д. Роджерс. - Москва : Мир, 1989.
2. Курс лекций по компьютерной графике в МГТУ им. Н.Э. Баумана / авт. А.В. Куров. - Москва : [б.н.], 2017 г..
3. Курс общей физики [Книга] / авт. И.В. Савельев. - Москва : Наука, 1970. - Т. 3.
4. An Overview of the Ray-Tracing Rendering Technique [В Интернете] // www.scratchapixel.com/. - 25 Ноябрь 2017 г.. - www.scratchapixel.com/lessons/3d-basic-rendering/ray-tracing-overview.