|  |  |
| --- | --- |
| 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 «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К КУРСОВОМУ ПРОЕКТУ***

***НА ТЕМУ:***

***«»***

Студент \_\_\_\_\_\_ИУ7-55Б\_\_\_\_\_\_\_\_\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_**Д.Р.Жигалкин**\_\_\_\_\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсового проекта **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_**О.В.Кузнецова**\_\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

*2020 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 \_\_ИУ7\_\_\_\_

(Индекс)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_И.В.\_Рудаков\_\_\_\_

(И.О.Фамилия)

« \_\_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_ 20 \_\_\_\_ г.

**ЗАДАНИЕ**

**на выполнение курсового проекта**

по дисциплине \_\_\_\_\_\_Компьютерная графика\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент группы \_\_\_\_ИУ7-55Б\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Жигалкин Дмитрий Романович\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия, имя, отчество)

Тема курсового проекта \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

Направленность КП (учебный, исследовательский, практический, производственный, др.)

\_\_\_\_\_\_\_\_учебный\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

График выполнения проекта: 25% к \_4\_ нед., 50% к \_7\_ нед., 75% к \_11нед., 100% к \_14нед.

***Задание\_\_\_\_\_***

***Оформление курсового проекта:***

Расчетно-пояснительная записка на \_25-30\_ листах формата А4.

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

На защиту проекта должна быть предоставлена презентация, состоящая из 15-20 слайдов.\_\_\_

На слайдах должны быть отражены: постановка задачи, использованные методы и\_\_\_\_\_\_\_\_\_

алгоритмы, расчетные соотношения, структура комплекса программ, интерфейс,\_\_\_\_\_\_\_\_\_\_\_

результаты проведенных исследований.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания « » \_\_\_\_\_\_\_\_ 2020г.

**Руководитель курсового проекта**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_О.В.Кузнецова\_\_\_\_

(Подпись, дата) (И.О.Фамилия)

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_**Д.Р.Жигалкин**\_\_\_\_**

(Подпись, дата) (И.О.Фамилия)

Оглавление

[Введение 4](#_Toc26537825)

[1. Аналитическая часть 5](#_Toc26537826)

[1.1 Обзор и анализ существующих программных систем и обоснование необходимости разработки 5](#_Toc26537827)

[1.2 Формализация объектов синтезируемой сцены 7](#_Toc26537827)

[1.3Обоснование использованных алгоритмов 8](#_Toc26537828)

[1.4 Описание трехмерных преобразований сцены 9](#_Toc26537837)

[1.5 Выводы из аналитического раздела 10](#_Toc26537838)

[2. Конструкторская часть 11](#_Toc26537839)

[2.1 Описание алгоритма трассировки лучей 11](#_Toc26537827)

[2.2 Уравнение лучей 12](#_Toc26537827)

[2.3 Пересечение луча и сферы 13](#_Toc26537827)

[3. Технологическая часть 16](#_Toc26537847)

[3.1 Выбор и обоснование языка программирования и среды разработки 16](#_Toc26537848)

[3.2 Структура и состав классов 16](#_Toc26537849)

[3.3 Сведения о модулях программы 19](#_Toc26537850)

[3.4 Интерфейс программы 19](#_Toc26537851)

[4. Экспериментальная часть 21](#_Toc26537852)

[4.1 Цель эксперимента 21](#_Toc26537853)

[4.2 Апробация 21](#_Toc26537854)

[4.3 Описание эксперимента 24](#_Toc26537855)

[Заключение 25](#_Toc26537856)

[Список использованной литературы 26](#_Toc26537857)

# Введение

В компьютерной графике на сегодняшний день большое внимание уделяется алгоритмам получения реалистических изображений. Эти алгоритмы являются самыми затратными по времени. Обусловлено это тем, что они должны предусматривать множество физических явлений, таких как преломление, отражение, рассеивание света. Профессиональные программы для кинематографа учитывают еще больше явлений (дифракцию, интерференцию, зависимость коэффициентов преломления, отражения, поглощения от длины волны падающего света, вторичное, третичное отражение света).

В моей курсовой работе для рендеринга применяется алгоритм обратной трассировки. На сегодняшний день он считается одним из лучших для формирования реалистических изображений. Его используют большинство трехмерных графических редакторов. Применяется так же алгоритм z-буфера. Подобным методом пользуются в программах, где крайне важна скорость.

Цель данной работы - …

Для увеличения скорости работы алгоритма я применилтехнологиюпараллельных вычислений.

Чтобы достигнуть поставленной цели, требуется решить следующие задачи:

1) описать структуру трехмерной сцены, включая объекты, из которых состоит сцена, и дать описание выбранных свойств;

2) выбор и/или модифицирование существующих алгоритмов трехмерной графики, которые позволят визуализировать трехмерную сцену;

3) реализация данных алгоритмов для создания трехмерной сцены;

4) разработать программное обеспечение, которое позволит отобразить трехмерную сцену и визуализировать оптические эффекты.

# Аналитическая часть

* 1. **Обзор и анализ существующих программных систем и обоснование необходимости разработки**

На данный момент существует множество программных систем с открытым исходным кодом для работы над трассировкой лучей, но я выделил для себя основные:

* BRL-CAD
* Blender

BRL-CAD–представляет собой мощную 3Dсистему автоматизированного проектирования составных объемных тел методом конструктивной блочной геометрии. Включает в себя интерактивный геометрический редактор, параллельную трассировку лучей, рендеринг и геометрический анализ.

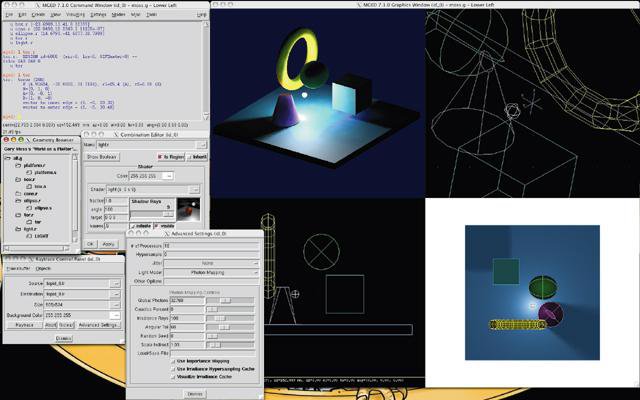


Рисунок 1.1 Интерфейс программы BRL-CAD.

Недостатки: большой объем программы, непростой интерфейс для неподготовленного пользователя.

Преимущества: кроссплатформенность, регулярные обновления, большие возможности для подготовленного пользователя.

Blender – профессиональное свободное программное обеспечение для создания трехмерной графики, включающее в себя множество возможностей, в том числе и рендеринг.

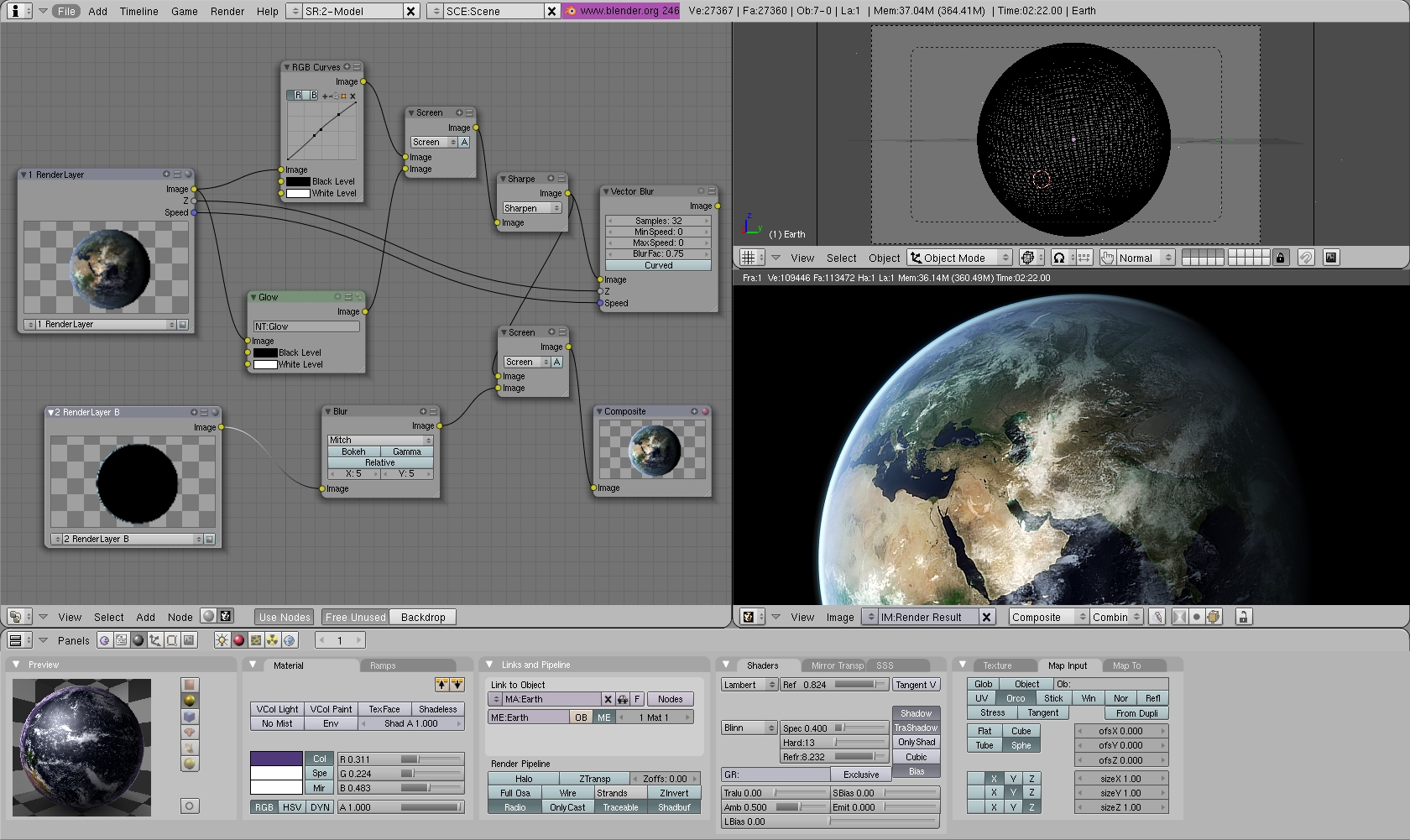


Рисунок 1.2 Интерфейс программы Blender.

Недостатки:Blenderимеет репутацию программы сложной для изучения. Практически каждая функция имеет соответствующее ей сочетание клавиш. Высокие системные требования для комфортной работы с приложением.

Преимущества: характерной особенностью является небольшой размер. Поддержка разнообразных геометрических примитивов, включая полигональные модели, универсальные встроенные механизмы рендеринга.

Вывод из проведенного анализа: не смотря на то, что существует немало многофункционального программного обеспечения для работы с трассировкой лучей, характерным для него недостатком является сложный и не совсем понятный для неподготовленного пользователя интерфейс. В своем программном продукте я хочу устранить данный недостаток.

## 1.2 Формализация объектов синтезируемой сцены

Сцена состоит из следующих объектов.

* Источников света – представляют собой 3 подобъекта, которые расположены на бесконечности:

1. Окружающее освещение. Описывается коэффициентом освещенности.
2. Точечный источник света. Описывается тремя координатами положения и коэффициентом освещенности.
3. Направленный источник света. Описывается вектором направления и коэффициентом освещенности.

* Плоскость земли – ограничивающая плоскость. Изначально расположенавнизу экрана, параллельна oXZ. Размеры задаются шириной, длиной и цветом RGB.
* Объемные тела – каждое из которых характеризуется следующими основными параметрами: координаты центра, цвет, коэффициент диффузного отражения, коэффициент зеркального отражения, прозрачность.

Они представляют собой 2 подобъекта:

1. Тела вращения:

Сфера. Помимо основных параметров задается еще и радиус.

1. Тела, заданные гранями:

Параллелепипед.Помимо основных параметров задается длина, ширина, высота.

Плоскость. Задается ширина и длина.

* Луч – невидимый объект, который задается параметрическим уравнением P= O + t \* direction. Где O– координаты точки начала луча, tпараметр, изменяя который можно получить любую точку на луче, direction–вектор направление луча.

## 1.3 Обоснование использованных алгоритмов

Главным алгоритмом, используемым в программе для удаления невидимых граней, а так же моделирования эффектов преломления и отражения, является алгоритм обратной трассировки лучей. Выбор был сделан в его пользу, так как в сравнении с его аналогом – алгоритмом трассировки лучей он оказался быстрее за счет того, что обрабатываются только те пиксели, на которые направлена камера.

Этот алгоритм позволил мне добиться максимальной реалистичности изображения. Он позволяет смоделировать распространение света в пространстве, учитывая законы геометрической оптики. Алгоритм можно быстро модернизировать, добавив в него обработку новых световых явлений. Алгоритм позволяет строить качественные тени с учетом большого числа источников света. Другие алгоритмы не позволяют так просто и качественно строить тени.

Достоинством алгоритма является то, что он не требователен к памяти, в отличие от алгоритма z-буфера. А недостатком является то, что работает он сравнительно долго и не позволяет строить изображения в реальном времени.

Для ускорения алгоритма применен метод иерархических оболочек. Он позволяет сократить время работы алгоритма трассировки в десятки, а на некоторых сценах в сотни раз. Среди всех алгоритмов оптимизации данный оказался самым эффективным. Метод BSP деревьев не дал значительного ускорения, а метод порталов в случае произвольной геометрии сцены вообще не применим. Алгоритм дает наилучшие результаты в сценах, где треугольники имеют примерно одинаковые размеры, а так же в разреженных сценах (т.е. в сценах, где объекты расположены на расстоянии, гораздо большем их линейных размеров).

Для визуализации диффузного отражения был использован закон Ламберта, который понятен и удобен при трассировке лучей.

Моделируя зеркальное отражение, я сделал выбор в пользу модели распределения Фонга. Хотя она и не основана на физических явлениях и является произвольной, она проста в вычислениях и хорошо выглядит.

## 1.4 Описание трехмерных преобразований сцены

Сдвиг точки:



Масштабирование относительно начала координат:



Любое вращение в трёхмерном пространстве может быть представлено как композиция поворотов вокруг трёх ортогональных осей (например, вокруг осей декартовых координат). Этой композиции соответствует матрица, равная произведению соответствующих трёх матриц поворота.

Матрицами вращения вокруг оси декартовой системы координат на угол {\displaystyle \alpha }α в трёхмерном пространстве с неподвижной системой координат являются:

* Вращение вокруг оси х:
* Вращение вокруг оси у:
* Вращение вокруг оси z:

## 1.5 Выводы из аналитического раздела

В данном разделе был рассмотрен алгоритмобратной трассировки лучей для удаления невидимых граней. Были изучены его преимущества и недостатки, проведено сравнение с алгоритмом Z-буфера. Также были определены и формализованы объекты сцены, преобразования сцены с объектами.

# Конструкторская часть

* 1. **Описание алгоритма трассировки лучей.**

Методы трассировки лучей на сегодняшний день считаются наиболее мощными методами создания реалистических изображений. Универсальность методов трассировки в значительной степени обусловлена тем, что в их основе лежат простые и ясные понятия, отражающие наш опыт восприятия окружающего мира.

Рассмотрим, как формируется изображение. Изображение получается из-за того, что свет попадает в камеру. Выпустим из источников света множество лучей. Назовем их первичными лучами. Часть этих лучей улетит в свободное пространство, а часть попадет на объекты. На них лучи могут преломиться, отразится. При этом часть энергии луча поглотится. Преломленные и отраженные лучи образуют множество вторичных лучей. Далее эти лучи опять же преломятся и отразятся и образуют новое поколение лучей. В конечном итоге часть лучей попадет в камеру и сформирует изображение.

Существуют алгоритмы, работающие по такому алгоритму. Но они крайне неэффективны, так как большинство лучей, исходящих из источника, не попадают в камеру. А приемлемая картинка получается, если трассировать большое число лучей, что займет очень много времени. Данный алгоритм называется прямой трассировкой лучей.

Метод обратной трассировки лучей позволяет значительно сократить перебор световых лучей. Этот метод разработали в 80-х годах Уиттед и Кэй. В этом методе отслеживаются лучи не от источников, а из камеры. Таким образом, трассируется определенное число лучей, равное разрешению картинки.

Предположим, что у нас есть камера и экран, находящийся на расстоянии d от нее. Разобьем экран на квадратики. Дальше будем по очереди проводить лучи из камеры в центр каждого квадратика (первичные лучи). Найдем пересечение каждого такого луча с объектами сцены и выберем среди всех пересечений самое близкое к камере. Далее, применив нужную модель освещения, можно получить изображение сцены. Это самый простой метод трассировки лучей. Он позволяет лишь отсечь невидимые грани.

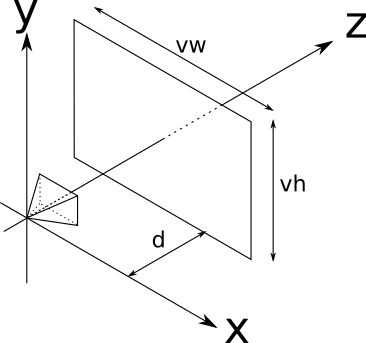


Рисунок 2.1 Расположение камеры и сцены

Но можно пойти дальше. Если мы хотим смоделировать такое явление, как отражение, нам необходимо из самого близкого пересечения пустить вторичные лучи. Например, если поверхность отражает свет и она идеально ровная, то необходимо отразить первичный луч от поверхности и пустить по этому направлению вторичный луч.   
 О прозрачности. Когда луч падает на прозрачную поверхность, мы пускаем вторичный луч и вычисляем дополнительный цвет — цвет света, проходящего сквозь объект.

* 1. **Уравнение лучей.**

Наилучшим способом представления лучей для нашей цели будет использование параметрического уравнения. Мы знаем, что луч проходит через O (точка, в которой находится камера), и мы знаем его направление (из O в V), поэтому мы можем выразить любую точку P луча как

где t — произвольное действительное число.  
Давайте обозначим , то есть направление луча, как  тогда уравнение примет простой вид

* 1. **Пересечение луча и сферы.**

Что такое сфера? Сфера — это множество точек, лежащих на постоянном расстоянии (называемом радиусом сферы) от фиксированной точки (называемой центром сферы):

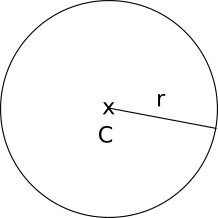


Рисунок 2.3 Изображение сферы.

Если C — центр сферы, а r — радиус сферы, то точки P на поверхности сферы удовлетворяют следующему уравнению:

Расстояние между P и C- это длина вектора из P в C:

Длина вектора — это квадратный корень его скалярного произведения на себя:

Или

Поскольку P — это одна и та же точка в обоих уравнениях, мы можем заменить P в первом на выражение для P во втором. Это даёт нам: