|  |  |
| --- | --- |
| 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-ПОЯСТНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОМУ ПРОЕКТУ**

**НА ТЕМУ:**

Студент **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Барсуков Н.М**\_\_\_\_\_\_\_\_\_\_**

*подпись, дата фамилия, и.о.*

Руководитель курсового проекта **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**Силаньтьевна А.В**\_\_\_\_\_**

*подпись, дата фамилия, и.о.*

*2019 г.*

Оглавление

[Аналитическая часть 6](#_Toc32072918)

[1.3 Критерии для алгоритмов 6](#_Toc32072919)

[1.2 Анализ существующих алгоритмов 7](#_Toc32072920)

[***1.4.1 Рей кастинг*** 8](#_Toc32072921)

[***1.4.2 Трассировка пути*** 8](#_Toc32072922)

[***1.4.3 Трассировка лучей*** 8](#_Toc32072923)

[***1.4.4 Растеризация*** 11](#_Toc32072924)

[1.5 Выбор алгоритма 13](#_Toc32072925)

[Заключение 14](#_Toc32072926)

[Список литературы 14](#_Toc32072927)

# 

Введение

В современном мире компьютерная графика является неотъемлемой частью человеческой жизни. Она используется повсеместно: для наглядного отображения данных, в компьютерных играх, в кино для создания эффектов. Вследствие этого перед людьми, создающими трехмерные сцены, встает задача создания реалистичных изображений, которые будут учитывать оптические явления преломления, отражения и рассеивания света, а также выбранную текстуру или цвет. Для создания еще более реалистичного изображения учитывается дифракция, интерференция, вторичные отражения света.

Существует множество алгоритмов компьютерной графики, которые решают эту задачу. Зачастую эти алгоритмы ресурсозатратны: чем более качественное изображение требуется получить, тем больше времени и памяти тратится на его синтез. Это становится проблемой при создании сцены в реальном времени.

Данная работа может быть использована в следующих областях:

* В сфере дизайна для наглядного отображения дизайна проектов в реальном времени
* В игровой сфере в виде рендера

# 

Постановка задачи

Задача: Реализовать рендеринг зеркального лабиринта в реальном времени.

Для реализации поставленной задачи необходимо выполнить следующие подзадачи:

1. Изучить предметную область;
2. Формализовать поставленную задачу;
3. Проанализировать возможные методы решения;
4. Составить критерии для оценки эффективности методов под нашу задачу;
5. Выбрать метод по выдвинутым критериям;
6. Выбрать среду и язык программирования
7. Разработка структур и интерфейса программы
8. Провести тестирование
9. Сделать вывод

Аналитическая часть

Описание предметной области

Компьютерная графика – одно из многих значимых направлений в IT сфере. Методы и алгоритмы представленные в данной области, массово и повсеместно используются на сегодняшний день. От пользовательского интерфейса, упрощающего наше обращение с программой, до компьютерных игр и фильмов. Одним из важных направлений «КГ» является «Построение реалистичного изображения»

«Построение реалистичного изображения» - это реализация рендера изображения, включающая в себя учет физических и психологических процессов. Свет или электромагнитная волна после взаимодействия (преломления, столкновения, отражения) с окружающей средой попадает в глаз, где в результате физических и химических реакций, вырабатываются импульсы, воспринимаемые нашим мозгом.

Формализация объектов синтезируемой сцены

Сцена состоит из следующих объектов:

* Источник света – вектор направления света, предполагается что источник света расположен на бесконечности и является точечным. Цвет свечения описывается через RGB параметры;
* Пол – представляет из себя простой объект ограничивающий нашу видимость по оси Y
* Зеркала – представляют из себя прямоугольники.
* Модель пользователя – объект, от лица которого пользователь двигается по лабиринту, представляет из себя множество маленьких элементарных элементов

Критерии для алгоритмов

Для реализации поставленной задачи необходимо ввести следующие основные критерии:

1. Качество и приближённость изображения к реалистичному
2. Производительность
3. Удобство реализации.

Первостепенным критерием для поставленной задачи является качество. Поскольку по результату курсового проекта необходимо получить генерацию 3х мерного, фотореалистичного изображения с множеством эффектов таких как: тень; отражение в зеркалах и на объектах. Все перечисленные эффекты должны выглядеть максимально качественно и естественно.

Вторым по значимости критерием является производительность, по сколько, учитывая указанное качества, важна требовательность метода к вычислительным мощностям компьютера. Для решения поставленной задачи нет необходимости в использовании современных запредельных мощностей. В нашем случае технология должна быть наименьше требовательна к компьютеру.

Третьим по значимости, но также немаловажным является учет удобства данного метода. Поскольку от выбора зависит результат. Если один алгоритм предусматривает специфические особенности, которые не предусматривает второй, то это дает приоритет первому.

Анализ существующих алгоритмов

На сегодняшний момент времени программное обеспечение может использовать несколько алгоритмов для получения конечного изображения. Разработано 4 группы методов, более эффективный чем просчет всех лучей света, освещающих сцену:

* Трассировка лучей;
* Трассировка путей;
* Растеризация и метод сканирования строк;
* Рей кастинг;

Проведем краткий анализ с целью первичного отбора методов.

Рей кастинг

***Рей кастинг*** – один из методов рендера в компьютерной графике. Использует геометрические вычисления лучей. Работает по принципу «выстреливания лучами» при котором изображение строится на основе замеров пересечения лучей с визуализируемой поверхностью в пространстве. Работает в разы быстрее чем рейтрейсинг, по причине отсутствия вычисления новых тангенсов лучей света (возникающих после пересечения луча, идущего от глаза к источнику света с поверхностью), что приводит к невозможности точно отображения преломления и проекции теней. ***По этой причине данный алгоритм не подходит для выполнения поставленной задачи.***

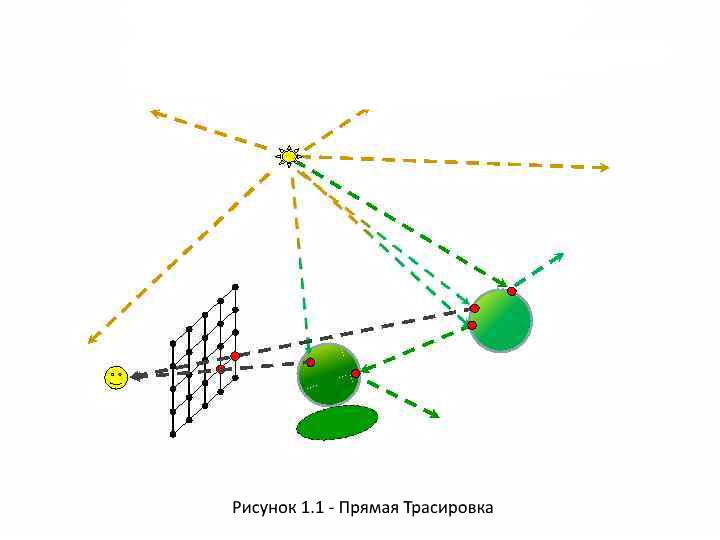
Трассировка пути

***Трассировка пути*** – методика рендера в компьютерной графике, нацеленная на реалистичную симуляцию поведения света (на сколько это возможно). Данный алгоритм рендера является изменённой версией трассировки лучей работает по принципу Монте Карла в котором новый луч света генерируется случайным образом внутри полусферы объекта до тех пор, пока не пересечется с источником света. (что может не произойти). Данный метод дает наибольшую приближенность к физическим законам распространения света, что делает его одним из ресурс требовательных подходов, ***в следствии чего данный подход не по падает под критерии производительности и является избыточным.***

Трассировка лучей

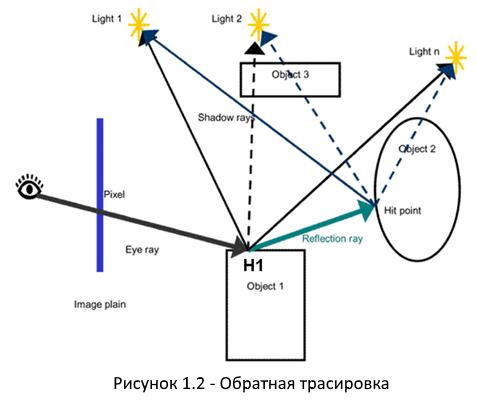
Трассировка лучей – технология, предназначенная для построения 3д изображений с помощью компьютера.

Главная идея данного метода, заключается в том, что наблюдатель видит любой объект, посредством испускаемого пучка лучей неким источником света, который падает на этот объект и затем каким – то путем доходит до наблюдателя. Свет может достичь наблюдателя, отразившись от поверхности, преломившись или пройдя через нее. Данный способ называется «прямая трассировка» (forward tracing).

К сожалению, данный подход является вычислительно неэффективным, по сколько если проследить за лучами, то можно убедиться, что весьма немногие из них дойдут до наблюдателя (см. рисунок 1.1).

Поэтому целесообразно рассмотреть другой способ трассировки, предложенный Аппелем, называемый «обратной трассировкой» (backward tracing) (см. рисунок 1.2)

Данный подход отслеживает те лучи которые в итоге попали на камеру. То есть мы генерируем лучи не от источника света, а от камеры, в следствии этого количество лучей ограничено, что уменьшает объем вычислительных ресурсов необходимых для расчета и моделирования сцены. В результате мы получаем изображение из «засвеченных» пикселей. Но это еще не все, чтобы получить наиболее реалистичный рендеринг, необходимо учитывать характеристики отражения и преломления материала, то есть количество отраженного и преломлённого света. Данный алгоритм является рекурсивным.



В начале мы рассчитываем направление отражения. Для этого нам необходимо знать исходное направление луча (от камеры к середине пикселя) и нормаль к поверхности в точке пересечения. Узнав направление отражения, мы отправляем новый луч в этом направлении. Далее делаем тоже самое с преломлением, но для этого нам нужно знать индекс преломления материала.

Недостатками данного алгоритма являются:

* Для остановки рекурсии необходимо вводить лимит интенсивности, чтобы избежать бесконечного просчета луча.
* Большая часть вычислений приходится на просчет пересечения лучшей с поверхностью. Что делает данный неэффективным для просчета сложных сцен с большим количеством объектов.

Плюсы данного подхода:

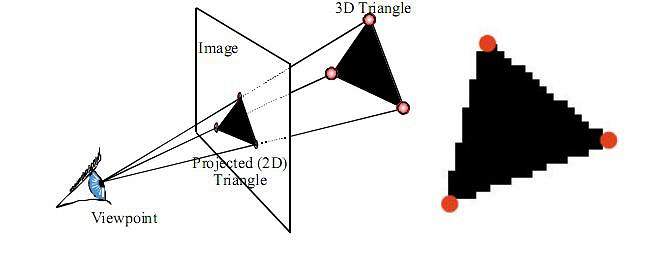
* Высокое качество изображения
* Возможность корректного создания большинства оптических эффектов, таких как тень, преломления, диффузные отражения.

Данный алгоритм достаточно прост в реализации, но к сожалению данная простота является обманчивой, по сколько от метода реализации определения пересечения с объектами, будет зависеть производительность данного метода.

Растеризация

Растеризация – метод рендеринга. Визуализация производится путем проецирования объектов сцены на экран без рассмотрения эффекта перспективы относительно наблюдателя. Эта технология является наиболее распространенной и популярной, так этот данный метод стар и проверен временем. Кроме того, он является более производительным, чем трассировка лучей. Именно это обеспечило ему такую популярность, по сколько современные вычислительные мощности домашних систем не могут обеспечить трассировку лучей в реальном времени в сценах, нагруженных объектами, обеспечив при этом большое количество кадров в секунду.

В сущности, алгоритмы растеризации являются способом решения видимости – то есть определения какая часть 3д объекта видна камере. Некоторые части этих объектов могут лежать за пределами видимости камеры, или могут быть закрыты другими объектами. Как правило объекты представляют в виде многоугольников или в частности треугольниками, которые необходимо проецировать на экран.



Не будем забывать, что основная задача растеризации – это решения проблемы с видимостью объектов. Поэтому наиболее часто встречающим алгоритмом растеризации является z – буфер, который позволяет нам понять какой из объектов находится ближе к камере.

Проблемой данного подхода является работа с хорошо детализированными объектами, так как они многополигональны, но это не порождает больших проблем по сколько удаленные объекты не нагружают систему.

В итоге получается, преимуществом растеризации является высокая производительность, которая и на сегодняшний день позволяет ему оставаться наиболее часто используемым методом рендеринга.

Выбор алгоритма

Учитывая ранее введённые критерии для способов рендеринга и учитывая особенности поставленной задачи. Алгоритмом для реализации проекта был выбран метод ***Трассировки лучей.***

С помощью данного метода мы можем генерировать фотореалистичное изображение с множеством различных оптических эффектов, такие как тени и отражения, прозрачные и непрозрачные объекты. Производительность данного подхода не критична по причине достаточной простоты сцены. Учитывая все выше перечисленное и достаточную простую реализацию, то Трассировка лучей является лучшим решением.

А если сравнивать с растеризацией, то благодаря имитации распространения лучей света, итоговое изображение, при трассировке получается выше качеством, при этом оба подхода отсекают невидимые поверхности.

А оптические эффекты необходимые для поставленной задачи: отражение, преломление, тени – достаточно сложны для реализации в растеризации, но достаточно тривиальны в алгоритме трассировки лучей.

Для выполнения поставленной задачи была выбрана модель Блинна – Фонга. Данная модель является достаточно реалистичной и позволяет изображать блики на объектах. Так же данная модель требует значительно меньше вычислений в случае бесконечно удаленного источника света, который будет использоваться в данной работе.

## **Заключение**

По результатам практики была разработана аналитическая часть курсового проекта по теме «Моделирования прогулки по зеркальному лабиринту». Сделан анализ алгоритмов, подходящих для решения поставленной задачи и выбран метод с помощью которого будет реализоваться курсовой проект.

# **Список литературы**

**Книга одного автора**

Роджерс Д.Ф. Алгоритмические основы машиной графики: Пер. с англ. – М.: Мир, 1989. – 512с., ил.

1. Алексей Берилло, DirectX Raytracing: трассировка лучей в реальном времени [электронный ресурс]. – URL <https://www.ixbt.com/3dv/directx-raytracing.html> (Дата обращения 12.06.2019)
2. Brian Caulfield, What’s the Difference Between Ray Tracing and Rasterization? [электронный ресурс], - URL <https://blogs.nvidia.com/blog/2018/03/19/> (Дата обращения 12.06.2019)
3. Dr. Jon Peddie, What’s the Difference Between Ray Tracing, Ray Casting, and Ray Charles? [электронный ресурс], - URL <https://www.electronicdesign.com/displays/what-s-difference-between-ray-tracing-ray-casting-and-ray-charles> (Дата обращения 12.06.2019)
4. Raycsting [электронный ресурс], - URL https://people.cs.clemson.edu/~dhouse/courses/405/notes/raycast.pdf (Дата обращения 13.06.2019)
5. Monte Carlo Path Tracing [электронный ресурс], - URL <http://www.graphics.stanford.edu/courses/cs348b-01/course29.hanrahan.pdf> (Дата обращения: 13.06.2019)
6. Метод трассировки лучей против растеризации: новое поколение качества графики? [электронный ресурс], - URL <http://www.thg.ru/graphic/ray_tracing_rasterization/print.html> (Дата обращения: 14.06.2019)
7. Rasterization: a Practical Implementation [электронный ресурс], - URL <https://www.scratchapixel.com/lessons/3d-basic-rendering/rasterization-practical-implementation> (Дата обращения: 14.06.2019)