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)

ФАКТУЛЬЕТ ___Информатика и системы управления_____

КАФЕДРА ___Программное обеспечение ЭВМ и информационные технологии_____

ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ

Студент _____Саркисов Артём Сергеевич_____
фамилия, имя, отчество

Группа____ИУ7-43Б_____

Тип практики
___стационарная_____

Название предприятия____МГТУ им. Н.Э. Баумана, каф. ИУ7_____

Студент

_____ ___Саркисов А.С._
подпись, дата *фамилия,и.о.*

Руководитель практики

_____ ___Куров А.В.____
подпись, дата *фамилия,и.о.*

Оценка_____

Москва, 2020 г.

Содержание

Введение

В наши дни 3D моделирование играет важную роль в жизни современного общества. Сегодня оно широко используется в сфере маркетинга, архитектурного дизайна и кинематографии, не говоря уже о промышленности. 3D - моделирование позволяет создать прототип будущего сооружения, коммерческого продукта в объемном формате. 3D - моделирование не обходит стороной и нишу ювелирных изделий. В производстве ювелирных украшений каждое изделие проходит стадию моделирования, с последующей презентацией полученной модели заказчику, только после этапа моделирования наступает этап производства. Специалисты стремятся максимально реалистично изобразить свои идеи, используя построенные на компьютере изображения. Драгоценный камень является неотъемлемой частью почти любого ювелирного изделия, в частности бриллиант является самым часто используемым драгоценным камнем.

Целью проекта является разработка программы для формирования реалистичного изображения бриллианта, с видом огранки по выбору, при различном освещении с учетом оптических свойств материала.

Для достижения поставленной цели необходимо решить следующие задачи:

- выбрать и исследовать алгоритмы построения трехмерного изображения, генерации текстур и освещения;
- произвести основные математические расчеты для реализации выбранных алгоритмов и методов;
- выбрать подходящий язык программирования для реализации поставленной задачи; реализовать интерфейс программного модуля.

1. Аналитический раздел

В этом разделе проводится анализ существующих алгоритмов построения трехмерного изображения, генерации текстур и освещения и выбираются наиболее подходящие алгоритмы для решения поставленных задач.

1.1 Анализ предметной области

Бриллиант - алмаз, которому посредством обработки придана ограненная форма, максимально выявляющая его естественный блеск. Источник света является точечным объектом. Описывается расположением и цветом. Объект невидимый. Камера является точечным объектом, с помощью которой получается перспективное отображение объектов сцены.

1.2 Обзор и анализ существующих программных систем и обоснование необходимости разработки

Крупные компании, занимающиеся разработкой программного обеспечения, не обходят стороной и программы для 3D моделирования. Так компания Pixlogic, являющаяся лидером рынка программного обеспечения для 3D моделирования в 2010 году выпустила программу ZBrush. Отличительной особенностью данного ПО является имитация процесса «лепки» трёхмерной скульптуры, усиленного движком трёхмерного рендеринга в реальном времени, что существенно упрощает процедуру создания требуемого трёхмерного объекта. Каждая точка (называемая пиксель) содержит информацию не только о своих координатах XY и значениях цвета, но также и глубине Z, ориентации и материале. Это значит, что вы не только можете «лепить» трёхмерный объект, но и «раскрасить» его, рисуя штрихами с глубиной. Но вам не придётся рисовать тени и блики, чтобы они выглядели натурально — ZBrush это сделает автоматически. Также быстро работает со стандартными 3d объектами, используя кисти для модификации геометрии

материалов и текстур. Позволяет добиться интерактивности при большом количестве полигонов. Используя специальные методы, можно поднять детализацию до десятков миллионов полигонов. ZBrush активно применяется для моделирования ювелирных изделий, в частности драгоценных камней. Однако подобные программные решения остаются недоступными для небольших рекламных или игровых студий в силу высоких требований к технике и колоссальных вычислительных затрат.

1.3 Выделение ключевых моментов разработки модели

Объекты сцены:

Бриллиант - алмаз, которому посредством обработки придана ограненная форма, максимально выявляющая его естественный блеск.
При классической бриллиантовой огранке на лицевую сторону (корону), на которой расположена площадка, наносятся три пояса граней таким образом, что на лицевой стороне, вместе с площадкой, располагаются 33 грани. На тыльной стороне (павильоне) находятся 24 грани. Таким образом, полная бриллиантовая огранка состоит из 57 граней. Полученный бриллиант имеет в плане круглую форму. Максимальное сверкание округлого бриллианта достигается при соблюдении точно рассчитанных пропорций граней павильона, обеспечивающих полное внутреннее отражение света. Входящий луч света должен полностью дважды отразиться от тыловых граней на противоположных сторонах камня и выйти из коронки, создавая максимальное сверкание:

Подставка для бриллианта – …;

Источник света – источник света является точечным объектом, описывается расположением и цветом, объект невидимый;

Камера – точечный объект, с помощью которого получается перспективное отображение объектов сцены.

1.4 Анализ алгоритмов построения трёхмерного изображения

В этом разделе будут рассмотрены и проанализированы наиболее подходящие для решения поставленной задачи алгоритмы построения трёхмерного изображения.

1.4.1 Алгоритм Робертса

Алгоритм Робертса - алгоритм получения изображений одиночных выпуклых объектов, составленных из плоских граней, работающий в объектном пространстве. Алгоритм Робертса может быть применен для изображения множества выпуклых многогранников на одной сцене в виде проволочной модели с удаленными невидимыми линиями. Метод не пригоден непосредственно для передачи падающих теней и других сложных визуальных эффектов.

Достоинствами алгоритма Робертса являются скорость работы и простота реализации. Основным недостатком метода, определившим ограниченность его распространения, являются неспособность без привлечения других подходов реализовать падающие тени, невозможность передачи зеркальных эффектов и преломления света и, наконец, строгая ориентация метода только на выпуклые многогранники.

1.4.2 Алгоритм Варнока

Непосредственно в алгоритме Варнока и его вариациях предполагается, что большие области изображения когерентны. В пространстве изображения выделяется окно, затем определяется достаточно ли простое содержимое внутри этого окна для визуализации или, если внутри пусто происходит переход к следующему окну. Если содержимое окна не удовлетворяет заданным мерам сложности, окно разбивается на фрагменты до тех пор, пока содержимое фрагмента не станет достаточно простым для визуализации или его размер не достигнет требуемого предела разрешения. В

последнем случае информация, содержащаяся в окне, усредняется, и результат изображается с одинаковой интенсивностью или цветом.

Достоинством алгоритма Варнока является учет когерентности, из-за чего скорость его работы повышается при увеличении размеров однородных областей изображения. К недостаткам алгоритма можно отнести невозможность передачи зеркальных эффектов и преломления света, а также несовершенство способа разбиения изображения – в сложных сценах число разбиений может стать очень большим, что приведет к потере скорости.

1.4.3 Алгоритм Вейлера-Азертона

Алгоритм Вейлера-Азертона по сути является оптимизированной версией алгоритма Варнока. В этом алгоритме разбиением окна вдоль границ многоугольника добиваются минимизации количества шагов.

Эффективность алгоритма Вейлера-Азертона, как и алгоритма Варнока, зависит от эффективности разбиений. В дальнейшем этот алгоритм был распространен на сплайновые поверхности. К достоинствам алгоритма можно отнести скорость работы, учет когерентности изображения. Недостатками алгоритма является сложность реализации, а также невозможность передачи зеркальных эффектов и преломления света.

1.4.4 Алгоритм трассировки лучей

Алгоритм трассировки лучей заключался в отслеживании путей, как первичных лучей из точки наблюдения через точки экранной плоскости, так и дальнейших отражённых и преломлённых лучей. Процесс продолжается до тех пор, пока очередные лучи не останутся без пересечений. Отражение и преломление рассчитываются по законам геометрической оптики.

Техника рендеринга с трассировкой лучей отличается высоким реализмом получаемого изображения. Она позволяет изобразить гладкие объекты без аппроксимации их полигональными поверхностями, а также

воссоздать тени, отражения и преломления света. Алгоритм трассировки позволяет параллельно и независимо трассировать два и более лучей, разделять участки для трассирования на разных узлах кластера и т.д. Также присутствует отсечение невидимых поверхностей, перспектива.

Недостатком метода трассирования является производительность. Трассировка лучей каждый раз начинает процесс определения цвета пикселя заново, рассматривая каждый луч наблюдения в отдельности.

1.4.5 Алгоритм обратной трассировки лучей

В прямом алгоритме трассировки значительная часть лучей, исходящая от источников света и других поверхностей, не попадает в поле зрения наблюдателя, то отслеживать их все не имеет смысла. Поэтому для формирования изображения используется обратная трассировка, то есть лучи отслеживаются в обратном порядке: от положения наблюдателя через все точки картинной плоскости к объектам и далее – по отражённым и преломлённым лучам.

Такая техника рендеринга отличается высоким реализмом получаемого изображения. Она позволяет изобразить гладкие объекты без аппроксимации их полигональными поверхностями, а также воссоздать тени, отражения и преломления света. Алгоритм трассировки позволяет параллельно и независимо трассировать два и более лучей, разделять участки для трассирования на разных узлах кластера и т.д. Также присутствует отсечение невидимых поверхностей, перспектива.

Недостатком метода обратной трассирования является производительность. Трассировка лучей каждый раз начинает процесс определения цвета пикселя заново, рассматривая каждый луч наблюдения в отдельности.

1.5 Выбор и обоснование выбора алгоритма построения трёхмерного изображения

Для решения поставленной задачи, поскольку поверхности бриллиантов очень гладкие, не смотря на высокие требования к производительности, в качестве алгоритма построения трёхмерного изображения был выбран алгоритм обратной трассировки лучей, позволяющий изобразить гладкие объекты без аппроксимации их полигональными поверхностями, также выбранный алгоритм отличается высоким реализмом получаемого изображения, однако для создания цветов, вызванных отражением Френеля, объемным поглощением и рассеиванием света, алгоритм необходимо расширить.

Для отражения Френеля будет необходимо вычислить коэффициент Френеля R, при пересечении лучом света поверхности, а затем определить интенсивность отражённого света. Для объемного поглощения в конце расчёта затенения света (перед возвратом значения оттенка) необходимо будет применить затухание в соответствии с законом Бугера-Ламберта. Для рассеивания света луч, помимо содержания геометрической информации, то есть начальной точки и направления, должен содержать информацию о монохроматичности и значении длины волны, в случае если он монохроматический, кроме того, расширенный трассировщик лучей должен порождать серию рассеянных лучей, когда световой луч входит в рассеивающий объект.

1.6 Анализ моделей освещения

В этом разделе будут рассмотрены и проанализированы наиболее подходящие для решения поставленной задачи модели освещения.

1.6.1 Модель освещения Кука-Торренса

Модель освещения Кука-Торранса представляет собой аналитическую модель ДФОС (двулучевая функция отражательной способности),

которая описывает зависящее от длины волны свойства отражения поверхности на основе принципов теории “микрограней”. Точное описание того, как свет отражается от поверхности, является фундаментальной предпосылкой для компьютерного зрения и графических приложений. Материалы реального мира демонстрируют характерную отражательную способность поверхности, такую как глянцевые или зеркальные блики, анизотропия или зеркальное отражение, которые необходимо моделировать для таких приложений. Поверхностная отражательная способность материала формализована понятием двулучевая функция отражательной способности (ДФОС), которая описывает отраженный отклик поверхности в определенном исходящем направлении освещения, определенного направления падающего света в полусфере направлений.

1.6.2 Модель освещения Ламберта

Модель Ламберта - простейшая модель освещения, моделирующая идеально диффузное освещение. Считается, что свет, падающий в точку, одинаково рассеивается по всем направлениям полупространства. Таким образом, освещенность в точке определяется только плотностью света в точке поверхности, которая в свою очередь линейно зависит от косинуса угла падения.

Модель освещения Ламберта хорошо работает только для сравнительно гладких поверхностей. В отличии от нее модель Орен-Найара основана на предположении, что поверхность состоит из множества микрограней, освещение каждой из которых описывается моделью Ламберта. Модель учитывает взаимное закрывание и затенение микрограней и также учитывает взаимное отражение света между микрогранями.

1.6.3 Модель освещения Фонга

Модель Фонга – модель освещения, состоящая из диффузной компоненты (модель Ламберта) и зеркальной компоненты. Помимо равномерного

освещения, модель реализует появляющиеся на модели блики. Отраженная составляющая освещенности в точке зависит от того, насколько близки направления вектора, направленного на наблюдателя, и отраженного в сторону наблюдателя луча.

В модели учитываются интенсивности фоновой, рассеянной компонент освещения, а также глянцевые блики.

1.6.4 Модель освещения Уиттеда

Модель освещения Уиттеда предназначена для того, чтобы рассчитать интенсивность отраженного к наблюдателю света в каждом пикселе изображения, которая в свою очередь может быть локальной или глобальной. В первом случае во внимание принимается только свет, падающий от источника (источников), и ориентация поверхности. Во втором учитывается свет, отраженный от других объектов сцены или пропущенный сквозь них.

Уиттед пользуется моделью с такими же членами рассеянного и ламбертовского диффузного отражения, а также зеркального отражения Фонга, как и в локальной модели освещения. Модель Уиттеда учитывает эффекты преломления и отражения, зеркальности.

1.7 Выбор и обоснование выбора модели освещения

При отрисовке с помощью обратной трассировки с учетом поставленной задачи необходимо реализовать эффекты преломления, отражения, прозрачности, затенения, поэтому, чтобы добиться реалистичного изображения решено было, несмотря на сложность реализации, использовать глобальную модель освещения Уиттеда. В ней будут учитываются только точечные источники различного цвета и источники рассеянного освещения.

2. Конструкторский раздел

2.1 Реализация расширенного алгоритма обратной трассировки лучей

В этом разделе будет представлена реализации расширенного алгоритма обратной трассировки лучей, выбранного для решения поставленной задачи.

2.1.1 Общая реализация алгоритма обратной трассировки лучей

Алгоритм обратной трассировки лучей, схема алгоритма которого представлена на рисунке 1.2, выглядит следующим образом: из камеры через каждый пиксел изображения испускается луч и находится точка его пересечения с поверхностью сцены. Лучи, выпущенные из камеры, называют первичными. Пусть, первичный луч пересекает некий объект 1 в точке H1, как показано на Рисунке 1.1.

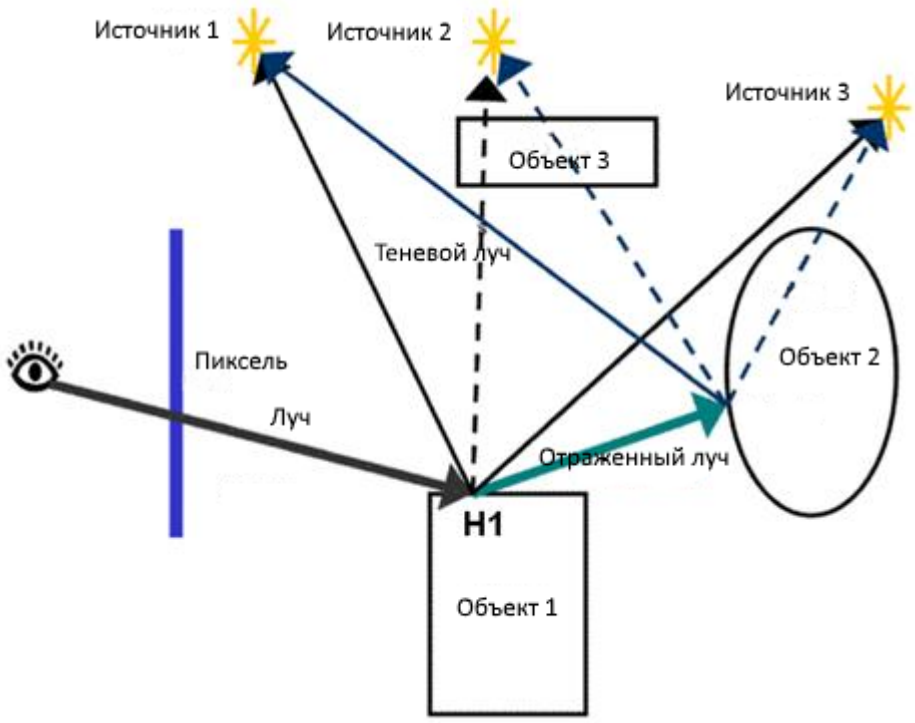

Рисунок 1.1. Алгоритм обратной трассировки лучей

Далее необходимо определить для каждого источника освещения, видна ли из него эта точка. Тогда в направлении каждого точечного источника света испускается теневой луч из точки H1. Это позволяет определить, освещается ли данная точка конкретным источником. Если теневой луч находит пересечение с другими объектами, расположенными ближе к точке H1, чем источник света, значит, точка H1 находится в тени от этого источника и освещать ее не надо. Иначе считаем освещение по некоторой локальной модели. Освещение со всех видимых (из точки H1) источников света складывается. Далее, если материал объекта 1 имеет отражающие свойства, из точки H1 испускается отраженный луч и для него вся процедура трассировки рекурсивно повторяется. Аналогичные действия должны быть выполнены, если материал имеет преломляющие свойства.

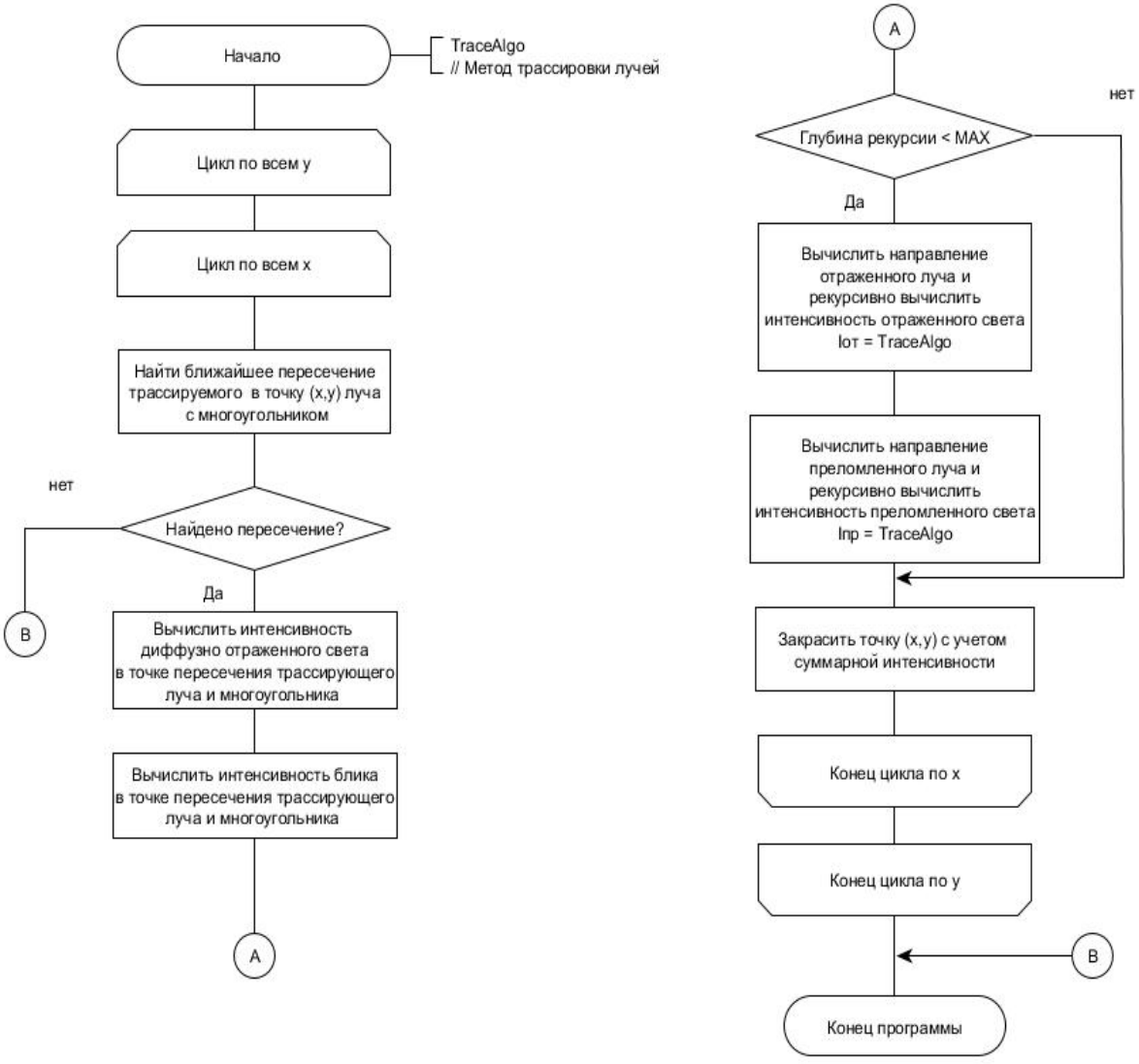

Рисунок 1.2. Схема алгоритма обратной трассировки лучей

2.1.2 Отражение Френеля

Когда световая волна отражается от границы, коэффициент отражения (отношение отражающей энергии к падающей энергии) зависит не только от угла падения и электрических свойств отражающего материала, но и от поляризации падающей световой волны. Согласно формулам Френеля:

$$\begin{cases} R_{\parallel} = \left(\dfrac{n_2 \cos\theta_i - n_1 \cos\theta_t}{n_2 \cos\theta_i + n_1 \cos\theta_t} \right)^2 = \dfrac{\tan^2(\theta_i - \theta_t)}{\tan^2(\theta_i + \theta_t)} \\ R_{\perp} = \left(\dfrac{n_1 \cos\theta_i - n_2 \cos\theta_t}{n_1 \cos\theta_i + n_2 \cos\theta_t} \right)^2 = \dfrac{\sin^2(\theta_i - \theta_t)}{\sin^2(\theta_i + \theta_t)} \end{cases}$$

где R || и R⊥ (называемые коэффициентами Френеля) соответствуют поляризациям, которые параллельны и перпендикулярны плоскости падения, n1 и n2 - показатели преломления сред, а θi и θt - углы падения и отражения. Если поляризацией пренебречь, отражение можно описать одним коэффициентом R (2):

$$R = \frac{1}{2}(R_{\parallel} + R_{\perp})$$

Это допущение широко применимо, поскольку на практике поляризация часто бывает не важна. Это допущение будет использоваться при решении поставленной задачи.

2.1.3 Объёмное поглощение

Объемное поглощение формирует внешний вид цветных бриллиантов. Согласно закону Бугера-Ламбертиана, внутреннее пропускание света, движущегося от одной точки к другой внутри прозрачного материала можно описать формулой (3):

$$T_{\text{internal}}(\lambda) = 10^{-a(\lambda)l}$$

где a (λ) - поглощающая способность материала, а l – длина светового пути. Согласно этому закону, увеличение l не только снижает интенсивность и углубляет цветовую насыщенность света, но также может изменять его оттенок. Это соотношение можно увидеть из спектральных кривых на рисунке 2.1, где l равно 1, 2, 4, 6, 8, 10 и 12 мм сверху вниз. Вот почему объемное поглощение может вызывать различные, но связанные цвета на цветных бриллиантов.

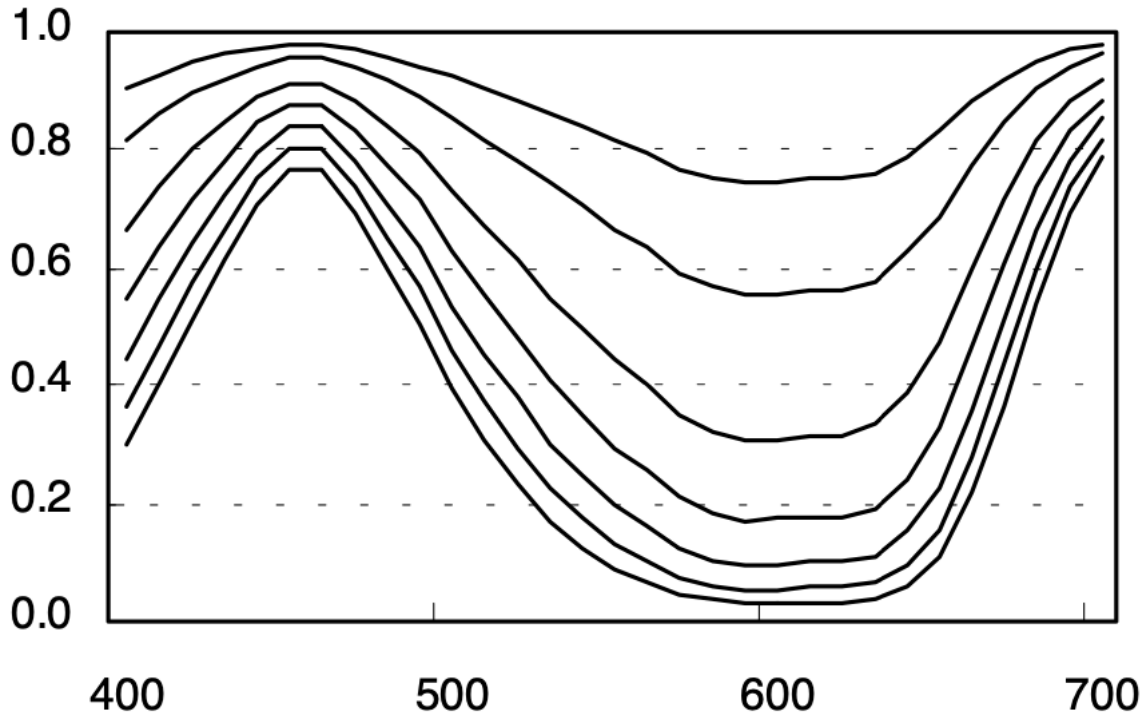

Рисунок 2.1. Спектральные кривые

2.1.4 Рассеивание света

Рассеивание света - это оптическое явление, при котором свет распадается на монохроматические компоненты и окрашивается в рассеивающих объектах, такич как бриллианты. Это вызвано зависимостью показателя преломления материала от длины волны.

Рассмотрим луч света, попадающий из воздуха в алмаз с показателем преломления n (λ) с углом падения θi. Угол преломления θr (λ) зависит от длины волны в соответствии с законом Снеллиуса:

$$\sin[\theta_r(\lambda)] = \frac{\sin(\theta_i)}{n(\lambda)}$$

При визуализации световой дисперсии наиболее важной частью является то, как представить монохроматические огни, созданные дисперсией.

Основная идея составной спектральной модели состоит в том, чтобы разложить спектральную функцию на гладкую составляющую и набор пиков. Гладкий компонент можно представить в виде выборки, а шип - в виде его местоположения и веса (или высоты). Аналитически выражая эту идею, любую спектральную функцию S (λ) можно записать как:

$$\begin{aligned} S(\lambda) &= S_{\text{smooth}}(\lambda) + S_{\text{spiky}}(\lambda) \\ &= S_{\text{smooth}}(\lambda) + \sum_{m=1}^{M} w_m \delta(\lambda - l_m) \end{aligned}$$

где l_m и w_m расположение пика и его высота. Численные, а также аналитические исследования показывают, что достаточная точность

достигается, если мы берем восемь точек выборки для гладких компонентов.

Одним из преимуществ этой модели является ее линейность в вычислительной мощности. При рендеринге изображений на основе спектра общая эффективность в значительной степени определяется эффективностью спектрального умножения, поскольку такое вычисление связано с каждым отражением и передачей. В рамках составной спектральной модели мы можем получить продукт двух сглаженных спектральных компонентов, умножив их соответствующие функциональные значения в одних и тех же точках выборки, с линейным весом, в выбранных точках. Произведение между гладким компонентом и пиком может быть вычислено с помощью линейной интерполяции.

$$S_{\text{smooth}}(l_m)\delta(\lambda - l_m) = \frac{(l_m - \lambda_i)}{\Delta\lambda} S_{\text{smooth}}(\lambda_i) + \frac{(\lambda_{i+1} - l_m)}{\Delta\lambda} S_{\text{smooth}}(\lambda_{i+1})$$

где значение расположения пика l_m находится в интервале [λi, λi+1]. Очевидно, что эти вычисление тоже линейны для рассматриваемых точек.

Еще одно преимущество, важное в данном контексте, состоит в том, что составная спектральная модель предлагает особые возможности для представления монохроматического света. Спектр монохроматического света можно описать как пик без гладких составляющих. Поскольку рассеянный свет является монохроматическим, эта модель прекрасно обеспечивает точную теоретическую основу для визуализации рассеивания света.

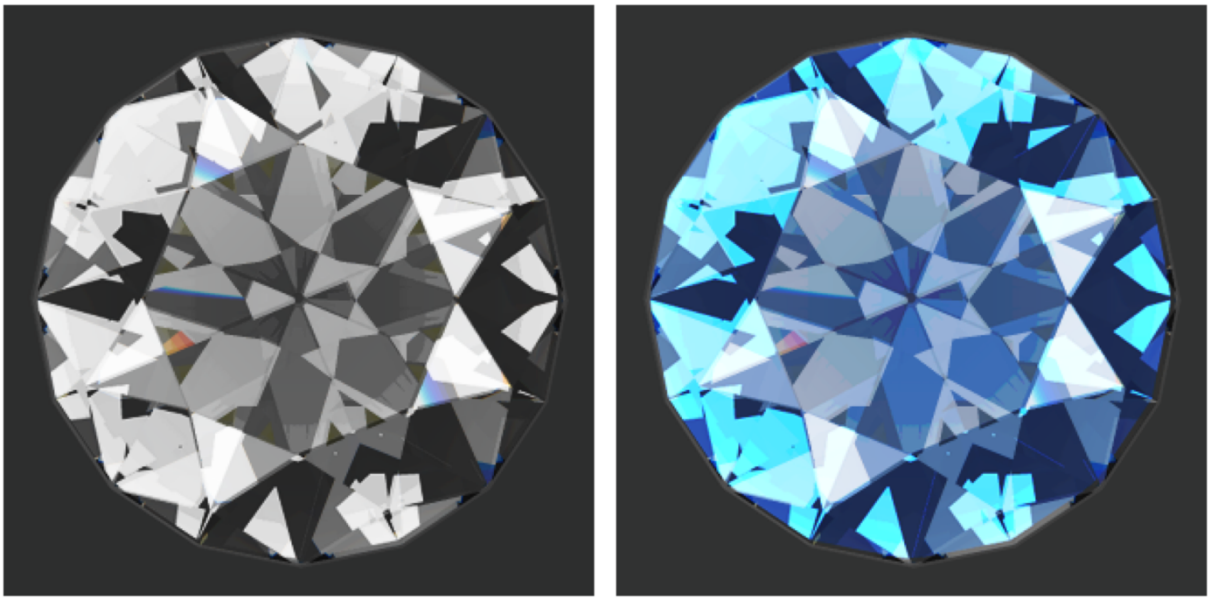

Рисунок 2.2. Сформированное изображение бесцветного бриллианта (слева) и цветного бриллианта (справа). Оба примера демонстрируют появление цветов радуги, вызванного рассеиванием света.

2.2 Реализация модели освещения Уиттеда

В этом разделе будет представлена реализации расширенного алгоритма обратной трассировки лучей, выбранного для решения поставленной задачи.

2.2.1 Общая реализация модели освещения Уиттеда

Модель освещения предназначена для того, чтобы рассчитать интенсивность отраженного к наблюдателю света в каждом пикселе изображения. Она может быть локальной или глобальной [10]. В первом случае во внимание принимается только свет, падающий от источника (источников), и ориентация поверхности. Во втором учитывается также свет, отраженный от других объектов сцены или пропущенный сквозь них, как показано на Рисунке 3.1, где $\vec{V}$ – направление взгляда, $\vec{R}$ – отраженный

луч направления взгляда, $\vec{P}$ – преломленный луч направления взгляда, L – направление на источник освещения, n1, n2 – показатели преломления сред.

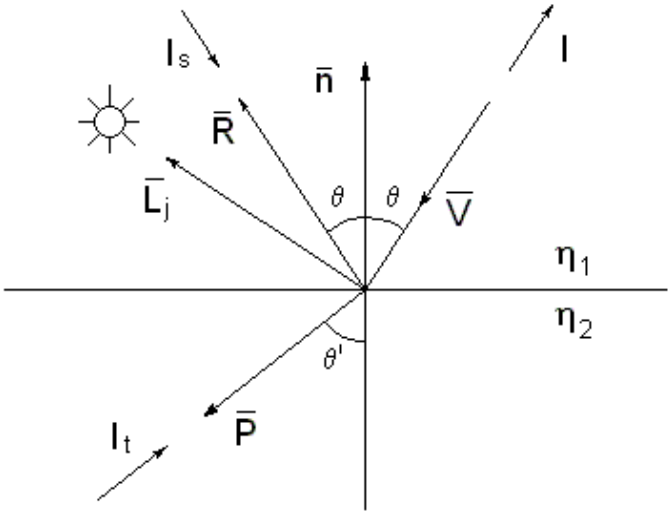

Рисунок 3.1. Зеркальное отражение и преломление

Согласно модели Уиттеда наблюдаемая интенсивность определяется суммарной интенсивностью:

$$I = k_a I_a C + k_d I_d C + k_s I_S + k_r I_r + k_t I_t,$$

где

k_a - коэффициент рассеянного отражения;

k_d - коэффициент диффузного отражения;

k_s- коэффициент зеркальности;

k_r - коэффициент отражения;

k_t - коэффициент преломления;

I_a- интенсивность фонового освещения;

I_d - интенсивность, учитываемая для диффузного рассеивания;

I_S - интенсивность, учитываемая для зеркальности;

I_r - интенсивность излучения, приходящего по отраженному лучу;

I_t - интенсивность излучения, приходящего по преломленному лучу;

C - цвет поверхности.

Модель Уиттеда учитывает эффекты преломления и отражения, зеркальности.

2.2.2 Нахождение отражённого и преломлённого лучей

Для нахождения направлений преломлённого и отражённого лучей достаточно знать направление падающего луча $\vec{L}$ и нормали к поверхности $\overrightarrow{N}$ в точке падения луча. Направление падающего луча и нормали к поверхности на данном этапе известны. Разлож. $\vec{L}$ раскладывается на два вектора $\overrightarrow{Lp}$ и $\overrightarrow{Ln}$, таких, что $\vec{L} = \overrightarrow{Lp} + \overrightarrow{Ln}$, где $\overrightarrow{Ln}$ параллелен $\vec{N}$, а $\overrightarrow{Lp}$ перпендикулярен, как изображено на Рисунке 3.2.

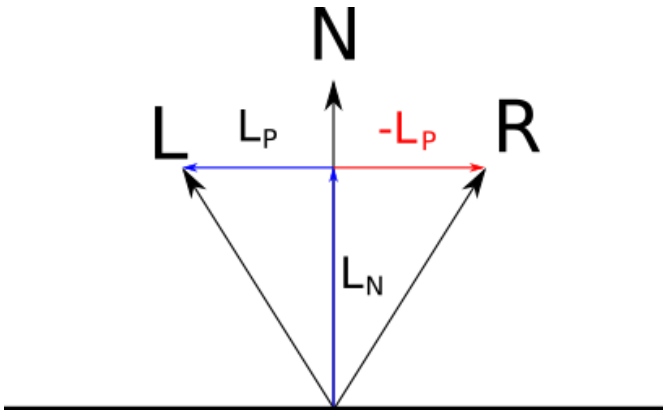

Рисунок 3.2. Разложение вектора падающего луча

$\overrightarrow{Ln}$ – проекция $\vec{L}$ на $\vec{N}$; по свойствам скалярного произведения и исходя из того, что $|\vec{N}| = 1$, длина этой проекции равна $(\vec{N}, \vec{L})$, поэтому $\overrightarrow{Ln} = \vec{N}(\vec{N}, \vec{L})$.

Отсюда $\overrightarrow{Lp} = \vec{L} - \overrightarrow{Ln} = \vec{L} - \vec{N}(\vec{N}, \vec{L})$.

Очевидно, что $\vec{R} = \overrightarrow{Ln} - \overrightarrow{Lp}$.

Подставим полученные ранее выражения и упростим, получим:

$\vec{R} = 2\vec{N}(\vec{N}, \vec{L}) - \vec{L}$. – Направление отраженного луча.

Преломленный луч $\vec{P}$ можно рассчитать из закона преломления. Он звучит следующим образом - луч падающий, луч преломленный и нормаль к поверхности в точке преломления лежат в одной плоскости. Углы падения и преломления при этом связаны следующим соотношением, называемым законом Снеллиуса:

$$\eta_i \sin\theta_i = \eta_t \sin\theta_t$$

Здесь:

η_i — показатель преломления среды, из которой свет падает на границу раздела двух сред;

Θ_i — угол падения света — угол между падающим на поверхность лучом и нормалью к поверхности;

η_t — показатель преломления среды, в которую свет попадает, пройдя границу раздела двух сред;

Θ_t — угол преломления света — угол между прошедшим через поверхность лучом и нормалью к поверхности.

Можно получить:

$$\vec{P} = \frac{n_1}{n_2} * \vec{L} + \left(\frac{n_1}{n_2} * \cos(\theta_i) - \sqrt{1 - sin^2(\theta_t)}\right) * \vec{N}$$

При этом $sin^2(\theta_t) = \left(\frac{n_1}{n_2}\right)^2 * \sin^2(\theta_i)$

Где θ_i – угол падения, θ_t – угол преломления, n_2 и n_1 – абсолютные показатели преломления двух сред.

2.2.3 Расчёт интенсивностей

Интенсивность света, диффузно отражающегося в точке поверхности, можно вычислить следующим образом (не зависит от положения наблюдателя).

$$I_d = k_d * \sum I_i * (\vec{n}, \overrightarrow{L_i}),$$

где

k_d – коэффициент диффузного отражения;

I_i – интенсивность света, попадающего в точку от i-го источника освещения;

$\vec{n}$ - нормаль к поверхности в данной точке;

$\overrightarrow{L_i}$ - единичный вектор, совпадающий по направлению с вектором, проведенным из i-го источника в рассматриваемую точку.

Интенсивность света, отраженного зеркально, может быть вычислена следующим образом (зависит от положения наблюдателя):

$$I_s = k_s \sum I_i * (\vec{S}, \overrightarrow{R_i})^N,$$

где

k_s – коэффициент зеркального отражения;

I_i – интенсивность света, попадающего в точку от i-го источника освещения;

$\vec{S}$ - единичный вектор, совпадающий по направлению с вектором из рассматриваемой точки в точку наблюдения;

$\overrightarrow{R_i}$- единичный вектор, задающий направление отраженного луча от i-го источника;

N - степень, аппроксимирующая пространственное распределение зеркально отраженного света.

Общую интенсивность можно определить по формуле:

$$I_r = k_a \sum I_{ia} + k_d \sum I_i * (\vec{n}, \overrightarrow{L_i}) + k_s \sum I_i * (\vec{S}, \overrightarrow{R_i})^N + k_r I_r + k_t I_t,$$

где

I_r и I_t – интенсивности, принесенные составляющими оттрассированных отраженного и преломленного лучей, I_{ia} - составляющие рассеянного освещения от i-го источника,

k_a - коэффициент рассеянного отражения,

k_d - коэффициент диффузного отражения,

k_s - коэффициент зеркальности,

k_r - коэффициент отражения,

k_t - коэффициент преломления.

3. Технологический раздел

В этом разделе будут обоснованы выборы языка, среды и платформы программирования, представлено описание структуры программы, что в свою очередь состоит из описания основных модулей и интерфейса программы.

3.1 Выбор и обоснование выбора языка, среды и платформы программирования

Для реализации поставленной задачи был выбран язык C++. Язык имеет статическую типизацию, поддерживает полиморфизм, перегрузку операторов (в том чисел операторов явного и неявного приведения типа), делегаты, атрибуты, события, переменные, свойства, обобщённые типы и методы, итераторы, анонимные функции с поддержкой замыканий, исключения. Это мощный объектно-ориентированный язык, который широко распространен, что подразумевает существование большого числа библиотек, что в свою очередь существенно упрощает разработку.

Для реализации проекта была выбрана среда программирования Xcode IDE 11.4. Пакет Xcode включает в себя изменённую версию свободного набора компиляторов GNU Compiler Collection и поддерживает языки C, C++, Objective-C, Objective-C++, Swift, Java, AppleScript, Python и Ruby с различными моделями программирования, включая, но не ограничиваясь, Cocoa, Carbon. Сторонними разработчиками реализована поддержка GNU Pascal, Free Pascal, Ada, C#, Perl, Haskell и D. Пакет Xcode использует GDB в качестве бэк-энда для своего отладчика.

3.2 Описание структуры программы

В этом разделе будут предоставлено описание основных модулей и интерфейса программы.

3.2.1 Описание основных модулей программы

Модуль	Функционал
main	Точка входа в программу и сборка всей сцены
gem_models	Описание моделей бриллианта разной огранки, как объектов сцены (координаты объекта)
gem_material	Описание модели бриллианта с точки зрения оптических свойств материала
ray_tracer	Описание алгоритма обратной трассировки лучей
light	Описание модели освещения
rotation	Описание модуля поворота объекта на сцене
gui	Описание модуля интерфейса

Таблица 4.1. Основные модули программы

3.2.2 Структура и классы программы

struct Color - описывает цвет по трём составляющим (RGB).

struct Cord - описывает координату в трёхмерном пространстве (x, y, z).

class Vector3 - описывает вектор трёхмерного пространства и основные методы для работы с ним.

class Material - описывает материал объекта, его основные свойства и цвет.

class Light - используется для описания источника освещения. Содержит

позицию источника и цвет излучения.

class Vertex - описывает вершину в трёхмерном пространстве.

class Primitive - базовый класс, хранит информацию о материале объекта и прототипы методов проверки на пересечения и поиски нормали.

class Triangle - описывает полигон (треугольник). Потомок класса Primitive. Хранит три вершины и координаты нормали. Обладает собственными методами проверки на пересечение и поиска нормали.

class Scene - описывает сцену. Хранит количество примитивов, количество источников освещения, указатели на массивы вершин, примитивов и источников освещения.

class Engine - основной класс программы. Содержит методы трассировки лучей, поиска пересечения.

class Ray - описывает луч. Содержит текущее положение, направление. Методы доступа и изменения положения и направления.

3.2.3 Описание пользовательского интерфейса

Position — параметр, отвечающий за позицию камеры;

Light Mode — параметр, отвечающий за присутствие света на сцене;

Auto Rotate — параметре, отвечающий за цикличный поворот объекта вокруг своей оси;

Gem Color — параметр, отвечающий за цвет бриллианта на сцене;

Gem Cut — параметр, отвечающий за огранку бриллианта на сцене;

Close Controls — кнопка для того, чтобы скрыть панель управления.

Рисунок 4.2. Пользовательский интерфейс

4. Экспериментально – исследовательский раздел

В данном разделе будет поставлен эксперимент по определению оптимальных параметров работы программы и приведены примеры использования.

4.1 Технические характеристики

Экспериментальная часть курсовой работы проводилась на компьютере со следующими характеристиками:

- Операционная система: macOS Catalina 10.15.7;
- Память: 16 GB 3773 MHz LPDDR4X;
- Процессор: 2 GHz Quad-Core Intel Core i5.

4.2 Цель работы

Основной целью курсовой работы было написание законченного программного продукта и нахождения некоторых закономерностей в алгоритмах, их анализ и проведение некоторых экспериментов связанных с этим анализом. Так как алгоритм обратной трассировки лучей является далеко не самым быстрым, то естественно напрашивалось исследовать время генерации изображения в зависимости от различных факторов. Была исследована зависимость скорости генерации изображения от свойств материала. Свойства бриллианта менялись в каждом опыте, но его размер и положение оставалось неизменным. В итоге было проведено четыре опыта с замерами по времени, замер для каждой конфигурации был проведён по 10 раз, конечно время – среднее для этих 5 замеров.

Набор свойств материала для каждого замера:

1. Без эффектов
2. Эффект отражения
3. Эффект преломления
4. Эффект преломления и отражения

Замер / Набор	1	2	3	4	5	Среднее
1	4.4 с	4.6 с	4.3 с	4.4 с	4.5 с	4.4 с
2	5.0 с	4.7 с	5.4 с	5.1 с	4.9 с	5.0 с
3	5.5 с	5.2 с	5.1 с	5.7 с	5.5 с	5.4 с
4	11.3 с	11.1 с	11.2 с	11.0 с	11.1 с	11.1 с

Таблица 4.3. Замеры по времени разных наборов свойств материалов

4.3 Вывод из результата исследований

Из результатов экспериментов следует, что использование обоих эффектов (преломления и отражения), замедляет генерацию кадра примерно в 2,5 раза.

Тем самым, опыт показывает, что наложение немногих дополнительных условий и небольших добавок сказывается на времени выполнения задач, что является логичным, так как производится больше расчетов. Но нужно учесть, что основное время алгоритм все-таки тратит именно на просчет пересечений лучей с объектами, поэтому именно здесь нужна модернизация.

4.4 Примеры работы программы

В данном разделе будет продемонстрирована примеры работы программы.

Пример работы программы для бриллианта классической огранки, белого цвета, при активном освещении и виде спереди (Рисунок 4.4), те же самые параметры, но вид сверху (Рисунок 4.5). Пример работы программы для бриллианта квадратной огранки, белого цвета, при не активном освещении, вид спереди и сверху (Рисунок 4.6). Пример работы программы для бриллианта огранки в виде сердца, синего цвета, при активном освещении, вид спереди (Рисунок 4.7)

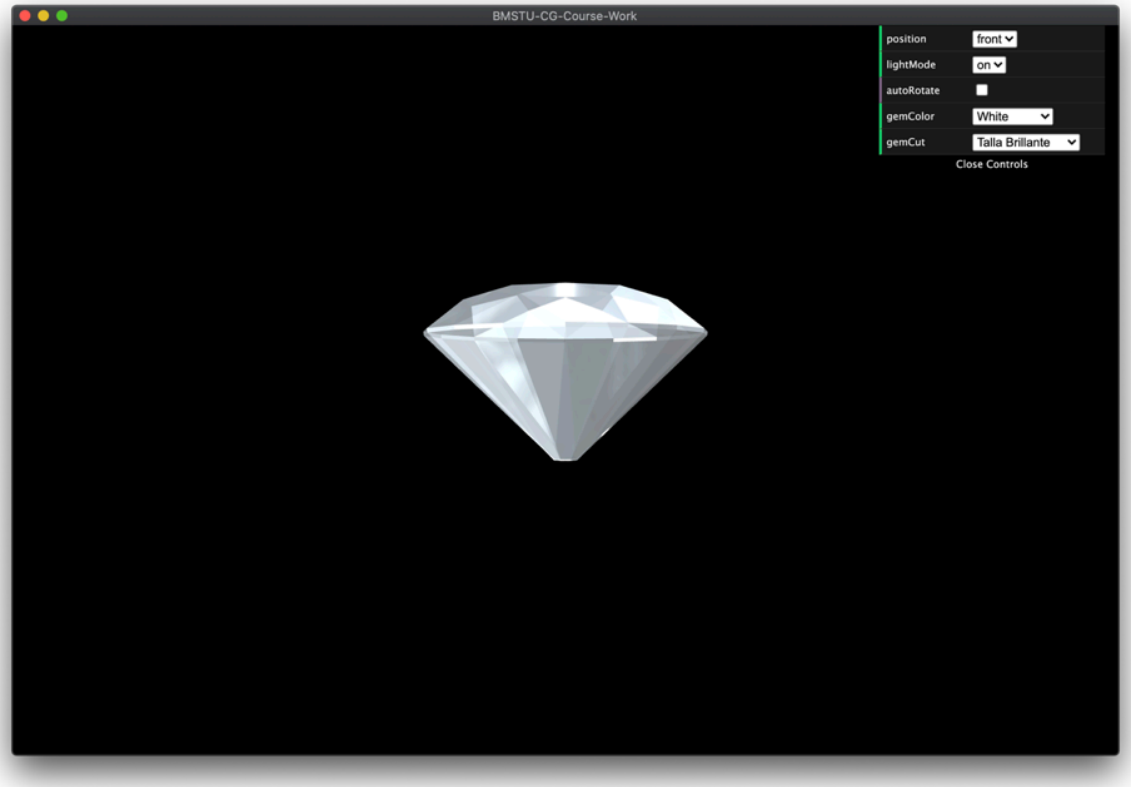

Рисунок 4.4. Пример работы программы для бриллианта классической огранки, белого цвета, в активном освещении, вид спереди.

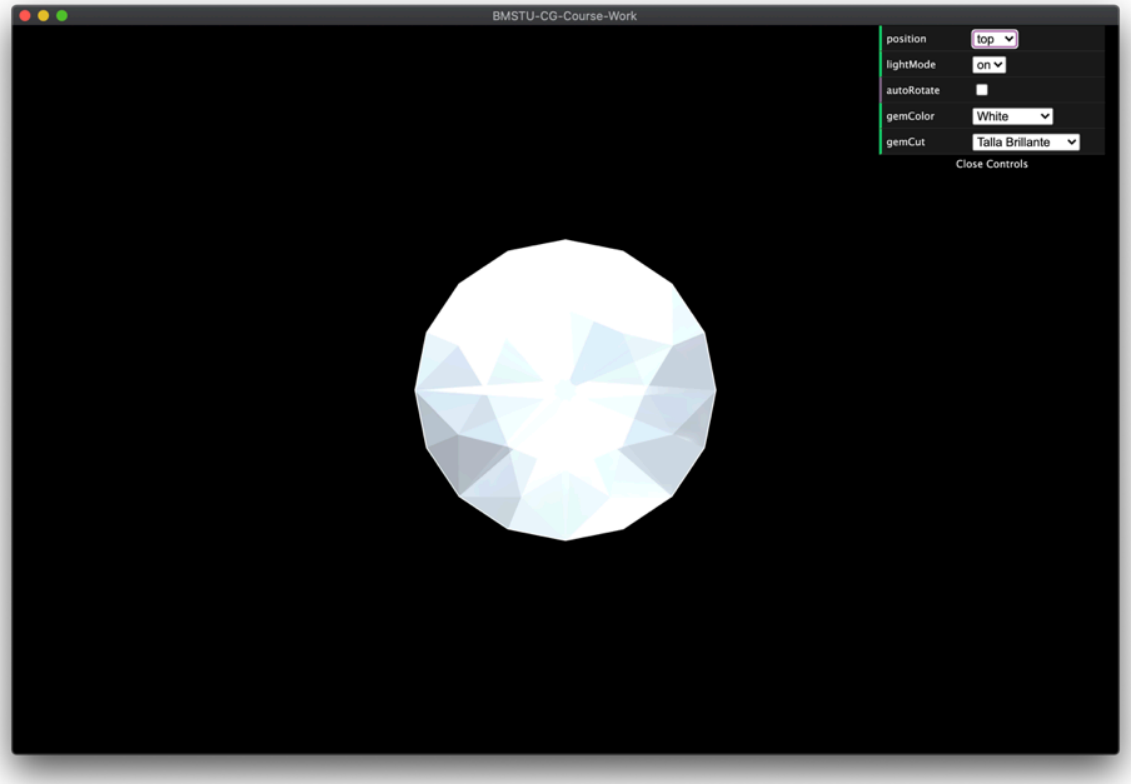

Рисунок 4.5. Пример работы программы для бриллианта классической огранки, белого цвета, в активном освещении, вид сверху.

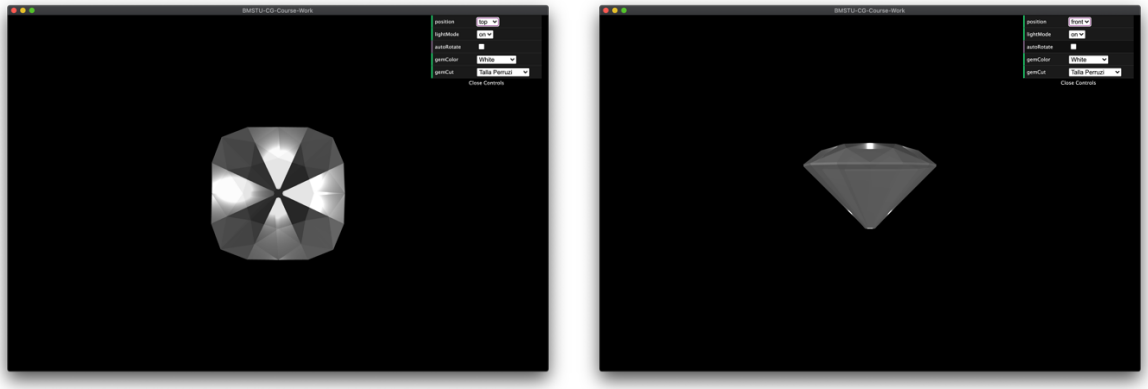

Рисунок 4.6. Пример работы программы для бриллианта огранки квадратом, белого цвета, в не активном освещении, вид спереди и сверху.

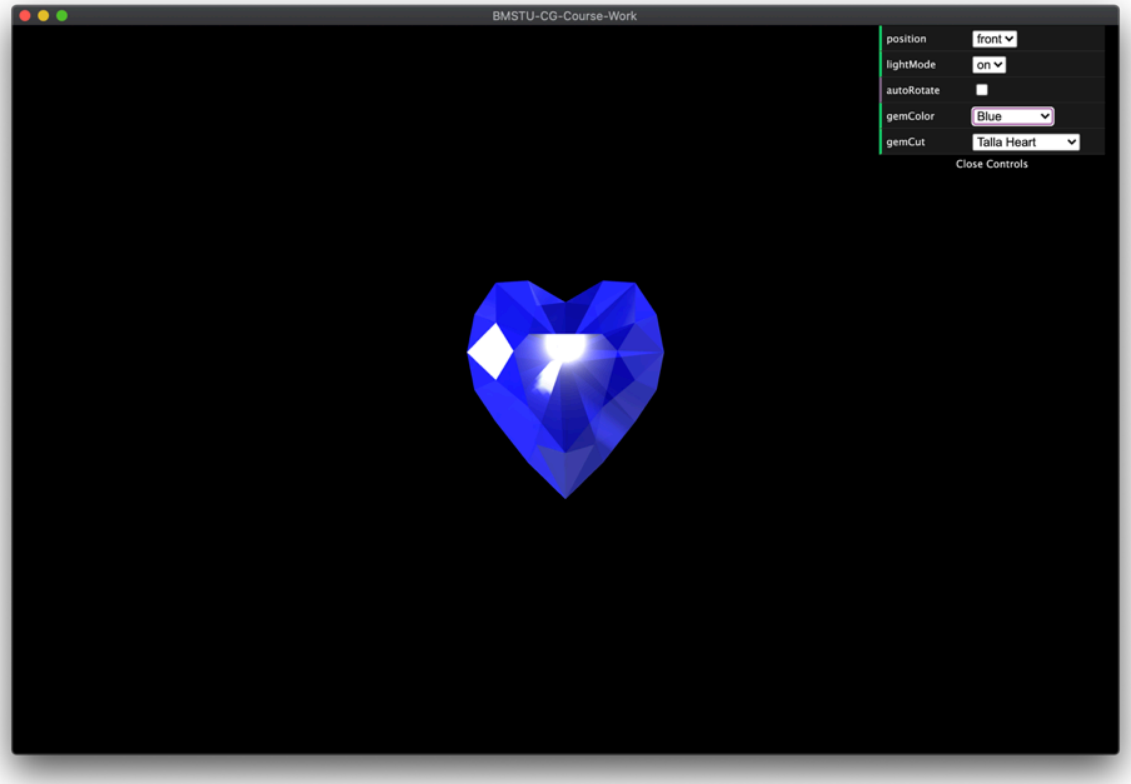

Рисунок 4.7. Пример работы программы для бриллианта огранки в виде сердца, синего цвета, в активном освещении, вид спереди.

5. Заключение

В этой программе были продемонстрированы результаты роботы алгоритма генерации реалистичных изображений бриллианта с помощью обратной трассировки лучей, отражения Френеля, объемного поглощения и рассеивания света. Данный алгоритм со всеми дополнительными модулями позволил выполнить поставленную задачу. А именно: были сгенерированы бриллианты разной огранки, разного цвета, при разной степени активности света со свойствами преломления и отражения, была создана камера, позволяющая поворачивать объект вокруг своей оси.

На сегодняшний день в реальном времени алгоритм обратной трассировки лучей используют лишь в исследовательских целях на сверхмощных компьютерах. Также считается, что алгоритм трассировки

лучей идеален для изображений искусственных объектов с геометрически простыми формами, например, автомобили, самолеты, здания и пр. Генерация таких объектов, как человеческое лицо, шерсть животных или лесной массив – это крайне трудная для алгоритма задача, которая повышает итак немалые требования к аппаратной части компьютера.

Однако алгоритм является интуитивно понятным, так как работает по законам реального мира - по законам физики. Трассировка лучей является универсальным методом и позволяет строить изображение высокого качества. Именно из-за этих качеств алгоритм достаточно популярен и методы его оптимизации совершенствуются постоянно.

Алгоритм обратной трассировки лучей действительно обладает большими вычислительными затратами. Использовать его для генерации изображения в реальном времени невозможно. Однако этот алгоритм в сочетании с дополнительными модулями, такими как отражение Френеля, объемное поглощение и рассеивание света хорошо справляется с задачей генерации реалистичного изображения бриллианта.

Литература

1. Макс Борн и Эмиль Вольф, Принципы оптики: электромагнитная теория распространения, интерференции и дифракции света.
2. Роберт Л. Кук и Кеннет Э. Торранс, Модель отражения для компьютерной графики.
3. Ральф М. Эванс, Введение в цвет.
4. Джеймс Д. Фоули, Андрис Ван Дам, Стивен К. Фейнер и Джон Ф. Хьюз, Принципы и практика компьютерной графики, второе издание.
5. Хуберт Нгуен, GPU Gems 3.
6. Редактор кода VS Code. Режим доступа: https://code. visualstudio.com/. Дата обращения: 13.09.2020.
7. OpenGL - The Industry’s Foundation for High Performance Graphics. Режим доступа: https://www.opengl.org/. Дата обращения: 13.09.2020.
8. GPU Gems. Chapter 5. Implementing Improved Perlin Noise. Режим доступа: https://developer.nvidia. com/gpugems/gpugems/part-i-natural-effects/ chapter-5-implementing-improved-perlin-noise. Дата обращения: 12.09.2020.