СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Компьютерная графика является неотъемлемой частью жизни человека. Она обладает высоким спросом: её используют сфере развлечений, например, эффекты в кинофильмах и сериалах, анимации в играх; в научных и инженерных дисциплинах, каких как, медицина, геофизика, ядерная физика, картография, для визуализации и лучшего восприятия информации.

Прогресс компьютерной графики не стоит на месте: со временем развиваются и оптимизируются алгоритмы, которые позволяют получить реалистичное трехмерное изображение. Вместе с прогрессом растут и требования к реалистичности, что приводит к росту как сложности алгоритмов, так и к затратам ресурсов компьютера, таких как оперативная память и время работы алгоритма.

Целью курсовой работы является разработка программного обеспечения для моделирования изображения трёхмерного фрактала "оболочка Мандельброта".

Для достижения поставленной цели необходимо решить следующие **задачи**:

1) выбрать и описать алгоритмы реализации, необходимые для визуализации поставленной задачи;

2) составить требования к программному продукту;

3) реализовать выбранные алгоритмы;

1 Аналитическая часть

В данном разделе проводится анализ существующих алгоритмов для решения поставленной задачи. В результате анализа выбирается алгоритм для дальнейшей реализации.

1.1 Анализ предметной области

Понятия фрактал и фрактальная геометрия появились в конце 70-х годов XX века, и их основоположником принято считать Бенуа Мандельброта, который в 1977 году выпустил книгу "Фрактальная геометрия природы в которой Мандельброт даёт такое определение фрактала:

Фрактал — структура, состоящая из частей, которые в каком-то смысле подобны целому. Слово фрактал образовано от латинского *fractus* - состоящий из фрагментов. [1]

По общепринятой классификации фракталы делят на геометрические, алгебраические и стохастические. [2]

Геометрические фракталы (в двумерном случае) получают на основе некоторой исходной фигуры путём ее дробления и выполнения различных преобразований полученных фрагментов.

Одним из наиболее известных примеров геометрического фрактала является снежинка Коха. В качестве исходного объекта в ней выбран равносторонний треугольник со сторонами единичной длины. На каждой итерации стороны фигуры делятся на 3 равные части, центральная часть отбрасывается, а на её месте строятся две боковые стороны равностороннего треугольника.

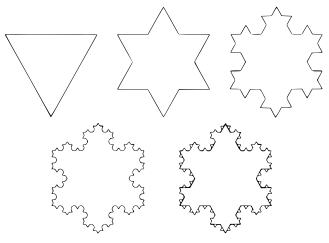

Рисунок 1 – Снежинка Коха на разных итерациях

Алгебраические фракталы получают с помощью итерационных процессов по выражениям типа

$$Z_{n+1} = f(Z_n),$$

где z - комплексное число, а f - некая функция.

Наиболее ярким представителем квадратичных алгебраических фракталов является множество Мандельброта. Его получают на комплексной плоскости при итерировании по формуле

$$Z_{n+1} = Z_n^2 + c, \tag{1}$$

где c - комплексная переменная.

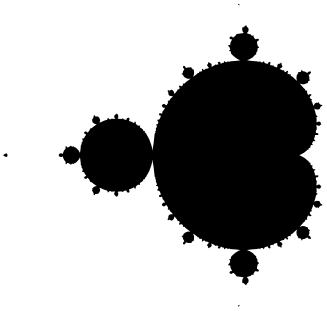

Рисунок 2 – Множество Мандельброта

Стохастические фракталы получают при хаотическом изменении каких-либо параметров в итерационном процессе. В отличие от алгебраических и геометрических фракталов, они не являются детермнеированными.

Ярким примером стохастического фрактала является траектория броуновского движения на плоскости.

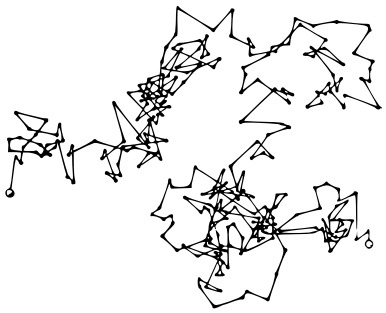

Рисунок 3 – Траектория броуновского движения

Новой ступенью развития фрактальной геометрии являются трёхмерные фракталы.

Оболочка Мандельброта — трёхмерный фрактал, аналог множества Мандельброта, созданный Д. Уайтом и П. Ниландером с использованием гиперкомплексной алгебры, основанной на сферических координатах.

Формула для n-ой степени трехмерного гиперкомплексного числа x, y, z следующая:

$$\langle x, y, z\rangle^n = r^n\langle cos(n\theta)cos(n\phi), sin(n\theta)cos(n\phi), sin(n\phi)\rangle, \qquad (2)$$

где

$$r = \sqrt{x^2 + y^2 + z^2};$$
$$\theta = arctan(y/x);$$
$$\phi = arctan(z/\sqrt{x^2 + y^2}) = arcsin(z/r);$$

r - модуль, θ и ϕ - аргументы гиперкомплексного числа.

В модели Уайта и Ниландера используется итерация $z \mapsto z^n + c$, где z и c — трехмерные гиперкомплексные числа, на которых операция возведения в натуральную степень выполняется аналогично 2. Для $n > 3$, результатом является трёхмерный фрактал. Чаще всего используется восьмая степень.

1.2 Формализация объектов синтезируемой сцены

Сцена состоит из следующего набора объектов:

1) точечный источник света — задается положением в пространстве, интенсивностью и цветом. В зависимости от характеристик источника определяется освещенность объектов сцены. Характеристики имеют значения по умолчанию, но имеется возможность их изменить;

2) камера — задается положением в пространстве, углом поворота вокруг каждой из трех координатных осей;

3) изображаемый фрактал — задаётся координатами центра объекта, степенью уравнения фрактала, количеством итераций для построения объекта.

1.3 Анализ методов рендера модели

Рендеринг (*англ. "rendering— "визуализация"*) — термин, обозначающий процесс получения изображения по модели с помощью компьютерной программы.

Фрактал "оболочка Мандельброта"является алгебраическим, т. е. описывается математическими уравнениями и не имеет явной геометрии (например, треугольников или вокселей).

Рассмотрим основные методы рендера моделей:

1.3.1 Растеризация

Растровое изображение — это изображение, представляющее собой сетку пикселей – цветных точек на мониторе, бумаге и других отображающих устройствах.

Растеризация — это процесс получения растрового изображения.

Технология основана на обходе лучем вершин треугольного полигона, который сохраняет свою форму даже после попадания из трехмерного пространства в двухмерное. Каждая точка объекта в трехмерном пространстве переводится в точку на экране, а затем точки соединяются и получается изображение исходного объекта.

Преимущества:

— современные компьютеры оптимизированы для рендеринга растровых изображений, из-за чего процесс растеризации достаточно быстрый.

Недостатки:

— изображение на выходе получается ступенчатым, требуется дополнительное сглаживание;

— метод работает с примитивами, следовательно требуется ресурсоемкая полигонализация сложных объектов;

— для моделей сцены могут использоваться миллионы полигонов, каждый из которых должен быть подвержен растеризации;

— каждый пиксель может обрабатываться множество раз (например, при наложении полигонов);

1.3.2 Трассировка лучей

Трассировка лучей (*англ. "ray tracing"*) — это технология отрисовки трехмерной графики, симулирующая физическое поведение света.

Суть алгоритма заключается в моделировании пути света для создания изображения. Лучи света, исходящие из камеры, проверяются на пересечение с полигонами и примитивами, принадлежащими объектам сцены. Для каждого пересечения вычисляются параметры освещения, и испускаются вторичные лучи, которые используются для определения отражений и теней.

Преимущества:

— вычислительная сложность метода слабо зависит от сложности сцены;

— отсечение невидимых поверхностей, перспектива и корректное изменение поля зрения являются следствием алгоритма;

Недостатки:

— алгоритм может работать только с примитивами, следовательно для сложных объектов необходимо проводить полигонализацию;

— метод имеет крайне высокую вычислительную сложность и является крайне ресурсоёмким.

1.3.3 Марширование лучей

Марширование лучей (*англ. "ray marching"*) — разновидность алгоритма трассировки лучей.

Как и трассировка лучей, этот метод испускает луч в сцену для каждого пикселя экрана. Однако, в отличии от трассировщика лучей, данный метод не пытается вычислить пересечение луча и объекта аналитически. В

нём происодит смещение текущего положения вдоль луча, пока не найдется точка, пересекающая объект. Такая операция переноса является простой и нересурсозатратной, в отличие от аналитического вычисления пересечения, однако данный способ является менее точным, чем обычная трассировка.

Для сцен, состоящих из непрозрачных объектов, мжно ввести ещё одну оптимизацию в данный алгоритм. Она заключается в использовании полей расстояний со знаком.

Поле расстояний со знаком (*англ. "signed distance field"*) — это функция, получающая на вход точку пространства и возвращающая кратчайшее расстояние от этой точки до поверхности каждого объекта в сцене. Если точка находится внутри объекта, то функция возвращает отрицательное число.

Эта оптимизация позволяет заметно ограничить количество шагов при движении вдоль луча, что увеличивает эффективность метода.

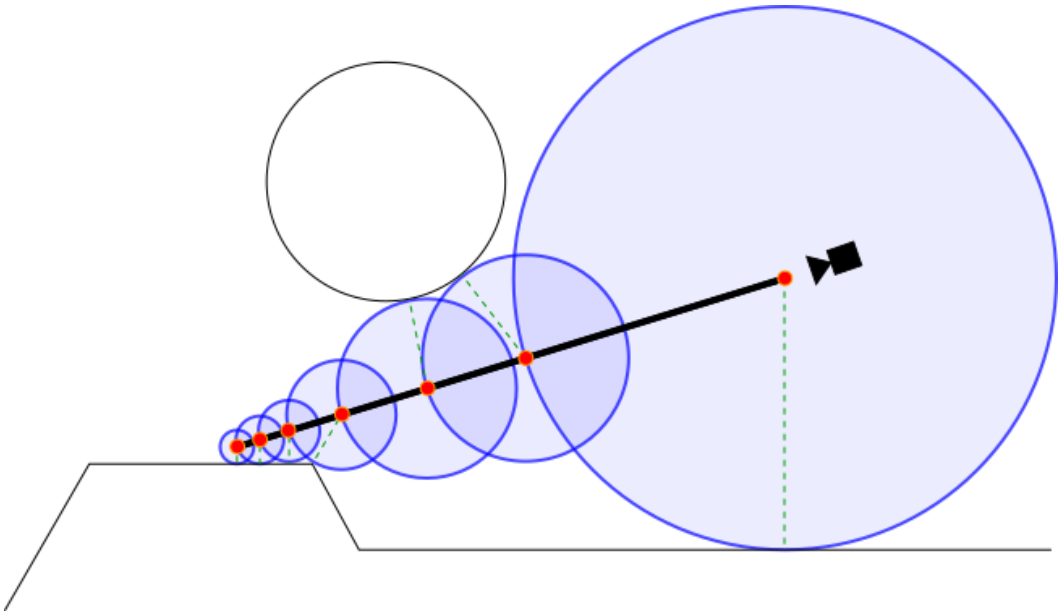

Рисунок 4 – Визуализация маршрирования лучей с использованием SDF. Красным отмечены все проверяемые точки

Преимущества:

— решает проблему производительности трассировки лучей за счёт оптимизаций;

— не требует разбиения поверхностей на примитивы, подходит для рендера сложных объектов;

— качество получаемого изображения сопоставимо с получаемым при трассировке лучей.

Недостатки:

— пересечение вычисляется менее точно, чем при трассировке лучей, имеется некоторая погрешность.

1.3.4 Сравнение алгоритмов

Сравнение методов рендеринга будет проводиться по следующим критериям:

1) качество изображения;

2) эффективность по времени выполнения рендера;

3) возможность рендерить сложные объекты без разбиения на примитивы

В таблице 1 приведено сравнение рассмотренных методов по указанным выше критериям.

Таблица 1 – Сравнение алгоритмов

Метод	Качество	Быстродействие	Сложные объекты
Растеризация	-	+	-
Ray Tracing	+	-	-
Ray Marching	+	+	+

Алгоритм маршрирования лучей подходит для рендера фракталов лучше прочих из рассмотренных, так как он не требует разбиения модели на примитивы для работы, не требует хранения информации о вершинах и гранях объекта, а также обладает высоким быстродействием.

1.4 Анализ алгоритмов закраски

1.4.1 Простая закраска

Алгоритм подразумевает однотонную закраску кажого многоугольника, принадлежащего объекту, в зависимости от интенсивности в одной из его точек. При этом предполагается, что:

— источник света расположен в бесконечности;

— наблюдатель находится в бесконечности;

— многоугольник представляет собой реальную моделируемую поверхность, а не является аппроксимацией криволинейной поверхности.

Ключевым недостатком данного алгоритма является то, что все точки грани будут иметь одинаковую интенсивность.

1.4.2 Метод закраски Гуро

Метод Гуро позволяет устранить дискретность изменения интенсивности и создать иллюзию гладкой криволинейной поверхности. Он основан на интерполяции интенсивности. Сам алгоритм можно разбить на четыре этапа:

1) Вычисление нормалей ко всем полигонам объекта;

2) Определение нормали в вершинах путём усреднения нормалей по всем граням, принадлежащим рассматриваемым вершинам;

3) Вычисление значения интенсивности в вершинах на основе выбранной модели освещения;

4) Закраска каждого многоугольника путём линейной интерполяции в вершинах сначала вдоль ребер, а затем между ними.

Достоинства:

— изображение получается более реалистичным, чем при простой закраске.

Недостатки:

— алгоритм обеспечивает непрерывность значений интенсивности вдоль границ многоугольников, но не обеспечивает непрерывность изменения интенсивности.

1.4.3 Метод закраски Фонга

При такой закраске, в отличие от метода закраски Гуро, интерполируется значение вектора нормали, а не интенсивности. Алгоритм также имеет четыре основных этапа:

1) Вычисление векторов нормалей к каждой грани и к каждой вершине грани;
2) Интерполяция векторов нормалей вдоль ребер грани;
3) Линейная интерполяция векторов нормалей вдоль сканирующей строки;
4) Вычисление интенсивности в очередной точке сканирующей строки.

Достоинства:

— можно достичь лучшей аппроксимации кривизны поверхности;

— более реалистичные блики.

Недостатки:

— высокая ресурсоёмкость;

— высокая вычислительная сложность.

1.4.4 Выбор алгоритма

В качестве метода закраски был выбран метод Гуро, так как он является менее ресурсоёмким, чем метод Фонга, что позволяет отрисовывать изображение в реальном времени.

Вывод

В данном разделе было проведено описание объектов сцены, рассмотрены алгоритмы визуализации сцены, отвечающие заданным требованиям. В качестве алгоритма для рендера изображения был выбран алгоритм маршрирования лучей как быстродейственный алгоритм, имеющий небольшую погрешность и позволяющий визуализировать сложные объекты сцены. В качестве алгоритма закраски был выбран метод закраски Гуро, так как он является быстродейственным и позволяет достаточно качественно закрашивать объекты.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Mandelbrot B. B. The Fractal Geometry of Nature. — San Francisco, 1982. — P. 462.

2. Морозов А. Д. Введение в теорию фракталов. — М. Ижевск: Институт компьютерных исследований, 2006. — С. 162.