МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное образовательное учреждение высшего образования
САНКТ – ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ

КАФЕДРА № 42

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

доцент, канд. техн. наук		А.В. Аграновский
должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1

Исследование фрактальной графики

по курсу: Компьютерная графика

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ гр. № 4321		К.А. Лебедев
	подпись, дата	инициалы, фамилия

Санкт-Петербург 2024

СОДЕРЖАНИЕ

1 Цель работы

Целью работы является изучение теоретических основ фрактальной графики, включая ее математическое описание и визуализацию, а также разработка программных алгоритмов для построения фракталов. В процессе выполнения работы исследуются особенности различных видов фракталов (геометрических, алгебраических, стохастических), их основные свойства и методы генерации.

Особое внимание уделяется разработке и программной реализации первого варианта фрактала (например, стохастического фрактала Плазма), с использованием алгебраических методов. Задачами работы являются: освоение принципов итеративного построения фрактальных структур, изучение параметров, влияющих на детализацию и визуализацию фракталов, и создание программы для генерации фрактальных изображений с возможностью изменения параметров отображения.

2 Задание

В рамках данной работы необходимо реализовать программное построение фрактала с использованием алгебраических методов. В качестве задачи выбран стохастический фрактал Плазма.

Основными задачами является: разработать программу, способную генерировать изображение стохастического фрактала Плазма, используя алгебраические методы. Предусмотреть возможность масштабирования изображение, что позволит приближать и детализировать отдельные участки фрактала. Программа должна отображать результат в виде генерации фрактала исходя из случайно заданных угловых цветов.

3 Теоретические сведения

Фрактальная графика — это раздел компьютерной графики, связанный с визуализацией математических объектов, называемых фракталами. Фракталы — это множества, обладающие свойством самоподобия, то есть они выглядят одинаково или схоже при увеличении масштаба. Основное свойство фракталов — это бесконечная сложность, возникающая при простом процессе построения.

Под фракталом понимается множество, которое имеет дробную размерность и создается итеративным способом, повторяя одно и то же преобразование над исходным объектом. Существует несколько основных типов фракталов:

– геометрические фракталы: создаются путем итеративных геометрических фигур,

– алгебраические фракталы: формируются на основе алгебраических уравнений и включают такие фракталы, как множество Мандельброта и множество Жюлиа,

– стохастические фракталы: создаются с использованием случайных процессов, например, фрактал "плазма".

Стохастические фракталы получаются, если какие-то параметры итерационного процесса, обеспечивающего построение как геометрического, так и алгебраического фрактала, являются случайными. Классическим представителем этого вида фракталов является фрактал «плазма». Для его генерации выбираются цвета четырех угловых пикселей некоторого квадрата с размерностью 2n 1, а затем определяется цвет его центрального пиксела как среднее арифметическое цветов четырех угловых пикселей с добавлением случайного числа. Берутся средние пикселы ребер исходного квадрата, для которых устанавливается среднее значение цветов двух соседних с ним пикселей плюс случайное значение. Тем самым получаем 4 новых квадрата с известными цветами угловых пикселей. Многократное повторение указанного алгоритма позволяет определить цвета всех пикселей исходного прямоугольника.
Случайное число обычно выбирается в промежутке [-ri, ri], где r ∈ [0; 1], а i - номер итерации (указанные выше два шага являются одной итерацией). Таким образом, случайное число уменьшается с увеличением номера итерации, что позволяет обеспечить плавное изменение цветов. В заключение отметим, что фрактальная графика является вычисляемой. В памяти сохраняются только формулы, а изображение получается путем расчета по этим формулам. Таким образом, удается не только получить оригинальное легкомасштабируемое изображение, но и уменьшить объем необходимой для его хранениянамяти. С этой точки зрения фрактальная графика напоминает векторную.

4 Алгоритм построения Стохастического фрактала Плазма

Построение стохастического фрактала Плазма осуществляется итеративным методом, с использованием случайных значений. Ниже приведен пошаговый алгоритм, описывающий процесс генерации фрактала.

4.1 Инициализация

Определить начальных параметров: установить начальные случайные значения для четырёх цветов, которые будут использоваться в процессе генерации фрактала. Эти значения определяют основную цветовую палитру фрактала.

Настройка области построения: определить размер области (канваса), на которой будет отображаться фрактал. Обычно это делается путём установки ширины и высоты канваса, соответствующих размеру экрана.

4.2 Расчёт цвета центрального пикселя

Цвет центрального пикселя: рассчитать цвет центрального пикселя исходного квадрата как среднее арифметическое цветов четырёх угловых пикселей, добавляя случайное число. Случайное число выбирается в интервале [-r_i, r_i][- r_i, r_i][- r_i, r_i], где r — коэффициент, варьирующийся от 0 до 1, и i — номер текущей итерации.

4.3 Определение цвета средних пикселей

Цвета пикселей на рёбрах: для пикселей, расположенных на рёбрах квадрата, вычислить цвет как среднее арифметическое цветов двух соседних пикселей плюс случайное число.

4.4 Рекурсивное деление и обработка. Итерации

Разделение на меньшие квадраты: разделить исходный квадрат на четыре меньших квадрата. Для каждого из этих новых квадратов повторить расчёт цветов угловых пикселей и средних пикселей, применяя шаги 2 и 3. Рекурсивно продолжить процесс деления и расчёта цветов, уменьшив случайное число r_i с увеличением номера итерации.

4.5 Окрашивание пикселей

Цветовое представление: определить цвет каждого пикселя на основе вычисленных значений после применения случайного значения и преобразования цветов. Отобразить каждый вычисленный цвет на канвасе.

4.6 Завершение работы

По завершении процесса генерации и отображения фрактала на экране программа завершает выполнение, предоставляя пользователю готовое изображение фрактала.

5 Язык программирования и используемые библиотеки

Для реализации программы построения стохастического фрактала Плазма был использован язык программирования JavaScript, а также следующие API

5.1 Язык программирования JavaScript

JavaScript был выбран в качестве основного языка программирования для реализации задачи ввиду его расширяемости, читабельности и мультипарадигменности его предметной области. Браузерный Javascript поддерживает однопоточность и предоставляет API для работы с графикой в браузере (Canvas API) , что делает его удобным инструментом для построения сложных математических структур, таких как фракталы

5.2 Используемые API

Canvas API — это API для двумерного рисования. Он позволяет рисовать линии, фигуры, изображения и текст прямо в браузере без использования плагинов, таких как Flash или Java. Canvas изначально был создан Apple для своих виджетов, но с тех пор был принят всеми разработчиками основных браузеров и теперь является частью спецификации HTML5.

Browser API — это интерфейс прикладного программирования для веб-сервера или веб-браузера. Это концепция веб-разработки, обычно ограниченная клиентской стороной веб-приложения (включая любые используемые веб-фреймворки), и поэтому обычно не включает детали реализации веб-сервера или браузера, такие как SAPI или API, если они не доступны для общего доступа через удаленное веб-приложение.

5.3 Используемые интерфейсы

document: используется для манипуляциями с DOM (древовидная структура элементов веб страницы).

Math используется для реализации математических операций и вычислений.

6 Описание программы построения фрактала

Программа для построения стохастического фрактала Плазма реализована на языке программирования JavaScript и состоит из нескольких функций и обработчиков событий, которые обеспечивают выполнение алгоритма построения и визуализацию фрактала.

6.1 Инициализация

Программа начинается с установки начальных значений цвета и размеров холста:

Цвета: color_0, color_1, color_2, color_3 - случайные значения, которые определяют цвета углов изображения.

Холст: создаётся элемент Canvas с размерами, соответствующими размеру экрана пользователя.

6.2 Функция createPlasmaGenerator

Эта функция инициирует процесс генерации фрактала плазма. В её составе:

noise: используется для добавления случайных изменений в цвет и создаёт более разнообразный эффект фрактала.

Функция hslToRgb: преобразует цвет из формата HSL (оттенок, насыщенность, светлота) в RGB, который используется для установки цветов пикселей.

Функция calcPlasma: рекурсивно рассчитывает значения пикселей для фрактала плазма, используя алгоритм разбиения на квадраты. Делит область на четыре квадрата и вычисляет цвета для каждого квадрата на основе цветов углов и случайного шума.

Функция drawPlasma: создает изображение, вызывая calcPlasma, и затем отображает его на Canvas.

6.3 Основной цикл

При загрузке страницы (событие DOMContentLoaded) вызывается функция createPlasmaGenerator, которая создаёт фрактал плазма и отображает его на Canvas.

При нажатии на Canvas (событие click) размеры холста увеличиваются на 20%, и фрактал пересчитывается и перерисовывается с новыми размерами.

6.4 Функция calcPlasma

Базовый случай: если область содержит только один пиксель, вычисляется цвет на основе color_0.

Рекурсия: если область больше одного пикселя, делится на четыре части, вычисляются средние цвета и случайный шум, а затем функция рекурсивно вызывается для каждой из четырёх частей.

6.5 Генерация и визуализация

Создаётся ImageData для холста, куда записываются вычисленные пиксели.

ctx.putImageData(imageData, 0, 0) отображает финальное изображение на Canvas.

6.6 Завершение работы программы

После завершения всех вычислений и отрисовки фрактала, пользователь видит изображение фрактала плазма на экране.

7. Завершение работы

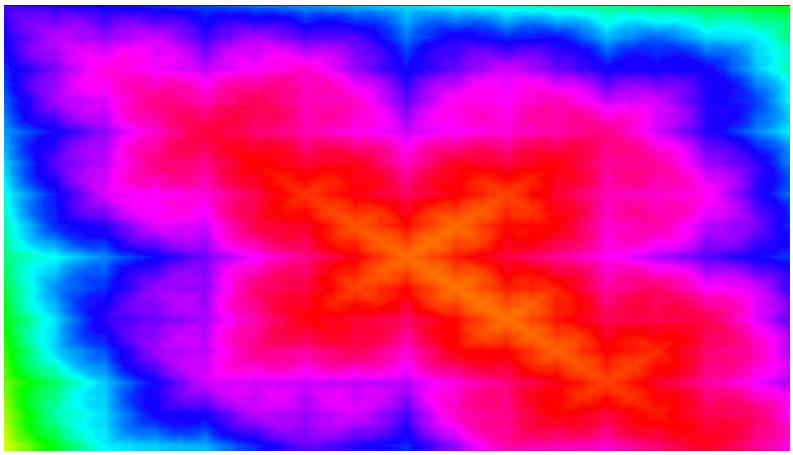

Рисунок 1 – Результат работы программы

8 Вывод

В ходе лабораторной работы были изучены теоретические основы фрактальной графики и разработан алгоритм построения стохастического фрактала плазма. Работа позволила не только разобраться в ключевых математических аспектах фракталов, но и реализовать алгоритм генерации фрактала с помощью современных вычислительных методов.

1) Фракталы и их классификация. В ходе работы были изучены основные виды фракталов, включая геометрические, алгебраические и стохастические. Особое внимание было уделено стохастическим фракталам, таким как фрактал плазма. Также был проанализирован принцип самоподобия и методы итеративного построения фракталов.

2.) Алгоритм построения фрактала плазма: был разработан алгоритм для генерации фрактала плазма, основанный на рекурсивном разбиении области и генерации случайного значения. Этот алгоритм позволил создать детализированное изображение, используя начальные цвета углов области и вычисления среднего цвета для внутренних точек.

3) Цветовая палитра и визуализация: важным аспектом работы стала разработка и применение цветовой палитры для фрактала плазма. Были рассмотрены различные подходы к выбору и применению цветов, что позволило создать эстетически привлекательное изображение фрактала. Применение цветовой палитры помогло визуально различать различные участки фрактала и улучшило его восприятие.

Лабораторная работа позволила не только освоить теоретические основы фрактальной графики, но и приобрести практические навыки в разработке программ для генерации фракталов. Реализация алгоритма генерации фрактала плазма показала важность понимания математического анализа и эффективного использования вычислительных методов. Полученные результаты подтверждают практическую значимость фрактальной графики и открывают новые возможности для её применения в различных областях, таких как компьютерная графика и моделирование природных структур.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1) ГУАП, документация для учебного процесса. – URL: https://guap.ru/regdocs/docs/uch (дата обращения 10.05.2024)

2) Руководство по JavaScript – URL: https://developer.mozilla.org/ru/docs/Web/JavaScript

3) Мартин Роберт, Чистый код: создание, анализ и рефакторинг / Мартин Роберт. — СПб.: Питер, 2022. — 464 с.

4) Фракталы: что это такое и какие они бывают– URL: https://skillbox.ru/media/code/fraktaly-chto-eto-takoe-i-kakie-oni-byvayut

5) HTML5: Canvas. Основы– URL: https://webmaster.alexanderklimov.ru/html/canvas_basic.php

6) Чекко Рафаэло, Чистый код: создание, анализ и рефакторинг / Мартин Роберт. — СПб.: Питер, 2013. — 272 с.

ПРИЛОЖЕНИЕ

Исходный код программы

```
const [color_0, color_1, color_2, color_3] = [
    Math.random(),
    Math.random(),
    Math.random(),
    Math.random(),
];

const canvas = document.getElementById("canvas");
const ctx = canvas.getContext("2d");
canvas.width = screen.width;
canvas.height = screen.height;

function createPlasmaGenerator() {
    const noise = 0.00025;

    function hslToRgb(hue, saturation, lightness) {
        lightness /= 100;
        const a = (saturation * Math.min(lightness, 1 - lightness)) / 100;
        const f = n => {
            const k = (n + hue / 30) % 12;
            const color =
                lightness - a * Math.max(Math.min(k - 3, 9 - k, 1), -1);
            return Math.round(Math.min(255, Math.max(0, 255 * color)));
        };
        return [f(0), f(8), f(4)];
    }

    function calcPlasma(
        imageData,
        top,
        left,
        bottom,
```

```
    right,
    color_0,
    color_1,
    color_2,
    color_3
) {
    const width = canvas.width;

    if (left > right || top > bottom) return;

    if (left === right && top === bottom) {
        const idx = (top * width + left) * 4;
        const [right, g, bottom] = hslToRgb(
            Math.floor(color_0 * 360),
            100,
            50
        );
        imageData.data[idx] = right;
        imageData.data[idx + 1] = g;
        imageData.data[idx + 2] = bottom;
        imageData.data[idx + 3] = 255;
        return;
    }

    const midX = Math.floor(left + (right - left) / 2);
    const midY = Math.floor(top + (bottom - top) / 2);
    const topColor = (color_0 + color_1) / 2;
    const leftColor = (color_0 + color_2) / 2;
    const bottomColor = (color_2 + color_3) / 2;
    const rightColor = (color_1 + color_3) / 2;
    const centerColor =
        (color_0 + color_1 + color_2 + color_3) / 4 +
        Math.sqrt((right - left) ** 2 + (bottom - top) ** 2) * noise;

    calcPlasma(
```

```
    imageData,
    top,
    left,
    midY,
    midX,
    color_0,
    topColor,
    leftColor,
    centerColor
);
calcPlasma(
    imageData,
    top,
    midX + 1,
    midY,
    right,
    topColor,
    color_1,
    centerColor,
    rightColor
);
calcPlasma(
    imageData,
    midY + 1,
    left,
    bottom,
    midX,
    leftColor,
    centerColor,
    color_2,
    bottomColor
);
calcPlasma(
    imageData,
    midY + 1,
```

```
            midX + 1,
            bottom,
            right,
            centerColor,
            rightColor,
            bottomColor,
            color_3
        );
    }

    function drawPlasma() {
        const width = canvas.width;
        const height = canvas.height;

        const imageData = ctx.getImageData(0, 0, width, height);

        calcPlasma(
            imageData,
            0,
            0,
            height - 1,
            width - 1,
            color_0,
            color_1,
            color_2,
            color_3
        );
        ctx.putImageData(imageData, 0, 0);
    }

    drawPlasma();
}

document.addEventListener("DOMContentLoaded", () => {
    createPlasmaGenerator();
```

```
});

document.addEventListener("click", () => {
    canvas.width = canvas.width * 1.2;
    canvas.height = canvas.height * 1.2;

    createPlasmaGenerator();
});
```