**Отчет по лабораторной работе: Сравнение алгоритмов растеризации отрезков и кривых**

**Введение:**

Целью данной лабораторной работы является закрепление теоретических знаний и практическое освоение основных алгоритмов растеризации отрезков и кривых. В ходе работы были изучены и реализованы следующие алгоритмы:

* Пошаговый алгоритм;
* Алгоритм цифрового дифференциального анализатора (ЦДА);
* Алгоритм Брезенхема для отрезков;
* Алгоритм Брезенхема для окружностей.

**Теоретическая часть:**

**Пошаговый алгоритм**

Пошаговый алгоритм является базовым методом растеризации отрезков, основанным на уравнении прямой линии. Для каждого значения x вычисляется соответствующее значение y, и пиксель с координатами

(x,y) отображается на экране. Этот метод прост в реализации, но неэффективен для компьютерной графики из-за высоких вычислительных затрат на операции с плавающей запятой.

**Алгоритм ЦДА**

Алгоритм цифрового дифференциального анализатора (ЦДА) улучшает пошаговый алгоритм путем использования приращений координат. Вместо вычисления каждого y заново, алгоритм добавляет фиксированное значение приращения на каждом шаге, что уменьшает количество вычислений. Однако алгоритм все еще использует операции с плавающей запятой, что влияет на его производительность.

**Алгоритм Брезенхема для отрезков**

Алгоритм Брезенхема является эффективным методом растеризации отрезков, который использует только целочисленные операции. Он основывается на вычислении ошибки отклонения от идеальной линии и позволяет быстро определять, какой пиксель следует закрасить на каждом шаге. Это делает его предпочтительным для многих графических приложений.

**Алгоритм Брезенхема для окружностей**

Алгоритм Брезенхема был расширен для растеризации окружностей. Подобно версии для отрезков, этот алгоритм использует целочисленные операции и симметрию окружности для эффективного вычисления точек, составляющих окружность на экране.

**Сравнение алгоритмов**

**Вычислительная сложность**

Пошаговый алгоритм: Использует операции с плавающей запятой для каждого пикселя. Вычислительная сложность пропорциональна длине отрезка (O(n)), но с высокой затратой ресурсов из-за операций с плавающей запятой.

Алгоритм ЦДА: Также имеет сложность O(n), но снижает количество операций с плавающей запятой за счет использования приращений.

Алгоритм Брезенхема (отрезок): Имеет сложность O(n), но использует только целочисленные операции, что значительно повышает производительность.

Алгоритм Брезенхема (окружность): Сохраняет эффективность версии для отрезков, с учетом симметрии окружности, что дополнительно уменьшает количество вычислений.

**Точность и качество отображения**

Пошаговый алгоритм: Могут возникать проблемы с равномерностью линии из-за округления значений координат.

Алгоритм ЦДА: Улучшает равномерность, но все еще подвержен ошибкам округления.

Алгоритм Брезенхема: Обеспечивает высокую точность и равномерность линий благодаря целочисленным расчетам и управлению ошибкой отклонения.

**Использование ресурсов**

Пошаговый алгоритм: Высокие затраты на вычисления из-за операций с плавающей запятой.

Алгоритм ЦДА: Меньшие затраты, но все еще требует операций с плавающей запятой.

Алгоритм Брезенхема: Минимальные затраты ресурсов благодаря использованию целочисленной арифметики.

**Практическая часть**

В ходе лабораторной работы были реализованы все четыре алгоритма на языке программирования Kotlin. Были проведены тесты на отрезках разной длины и наклона, а также на окружностях различного радиуса.

**Результаты**

Пошаговый алгоритм: Замечена низкая производительность и неравномерность линий при больших наклонах.

Алгоритм ЦДА: Производительность улучшилась, но при больших масштабах все еще наблюдались неточности.

Алгоритм Брезенхема (отрезок): Линии отображались равномерно и быстро, без заметных задержек.

Алгоритм Брезенхема (окружность): Окружности были построены корректно, с правильной симметрией и без искажений.

**Заключение**

В результате проведенной работы было установлено, что алгоритм Брезенхема является наиболее эффективным и точным методом для растеризации отрезков и окружностей. Он превосходит пошаговый алгоритм и алгоритм ЦДА по производительности и качеству отображения благодаря использованию целочисленных операций и оптимизации вычислений.