Лабораторная работа №4

Цель работы:

Закрепить теоретический материал и практически освоить основные возможности по использованию базовых алгоритмов растеризации отрезков и кривых:

- пошаговый алгоритмов
- алгоритм ЦДА
- алгоритм Брезенхема
- алгоритм Брезенхема(окружность)
- алгоритм Ву

Задачи работы:

- Создать класс для отображения растеризованного отрезка на экране
- Создать класс для отображения пояснительной информации по ходу алгоритма на экране
- Создать удобный и понятный пользовательский интерфейс
- Реализовать пошаговый алгоритм
- Реализовать алгоритм ЦДА
- Реализовать алгоритм Брезенхема
- Реализовать алгоритм Брезенхема для окружности
- Реализовать алгоритм Ву для сглаженных линий

Использованные средства разработки:

- Фреймворк Qt и язык C++

Ход работы:

1. Создание класса PlotArea для отображения растеризованного отрезка на экране с поддержкой координатной сетки и изменения масштаба. Были реализованы основные методы DrawGrid, DrawAxis, DrawTicks, DrawPixels, AddPixel

2. Создание класса LogWidget для отображения поясняющей информации в ходе алгоритма. Реализованы основные методы AppendMessage и AppendSeparator
3. Проектировка и создание удобного пользовательского интерфейса с возможностью выбора алгоритма, изменением масштаба, введением координат исходного отрезка
4. Реализация пошагового алгоритма в виде метода NaiveLine
5. Реализация алгоритма ЦДА в виде метода DDALine
6. Реализация алгоритма Брезенхема в виде метода BresenhamLine
7. Реализация алгоритма Брезенхема для окружности в виде метода BresenhamCircle
8. Реализация алгоритма By для сглаженных линий в виде метода WuLine
9. Добавление поясняющих сообщений в ходе каждого алгоритма
10. Добавление поддержки измерения прошедшего времени для каждого алгоритма

Временные характеристики:

Были введены наибольшие поддерживаемые входные данные для отрезка:

x1 = -100 y2 = 100

y1 = -75 x2 = 75

Для окружности :

x0 = 0

y0 = 0

R = 75

	С поясняющей информацией	Без поясняющей информации
Пошаговый алгоритм	889 мс	690 мс
Алгоритм ЦДА	839 мс	680 мс
Алгоритм Брезенхема	882 мс	699 мс
Алгоритм By	1554 мс	1392 мс
Алгоритм Брезенхема для окружности	1526 мс	1494 мс

Можно заметить, что вывод поясняющей информации занимает 5 — 20 % работы программы.

Разницы между алгоритмом Брезенхема и пошаговым алгоритмом не заметно вообще, хотя алгоритм Брезенхема не использует дробную арифметику. Можно предположить, что это из-за эмуляции отрисовки пикселей на экране, а их число в этих алгоритмах сопоставимое. То есть отрисовка пикселей занимает большую часть времени, так как реализована не аппаратно.

Вывод:

В ходе выполнения данной работы я:

- создал приложение, позволяющее проводить растеризацию отрезков и кривых базовыми алгоритмами
- закрепил полученные лекционные знания по различным алгоритмам растеризации
- получил дополнительный опыт по проектировке приложений
- углубил знания фреймворка Qt, а также языка C++
- получил дополнительный опыт работы с системой контроля версий Git