|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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)** |

ФАКУЛЬТЕТ «Информатика и системы управления»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Лабораторная работа № 3**

|  |  |
| --- | --- |
| **Дисциплина Компьютерная графика**  **Тема Программная реализация основных алгоритмов построения отрезков и исследование их временных и визуальных характеристик**  **Студент Иванов В.А.**  **Группа ИУ7-42Б**  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель Куров А.В.** |  |

Москва.

2020 г.

**Цель работы**

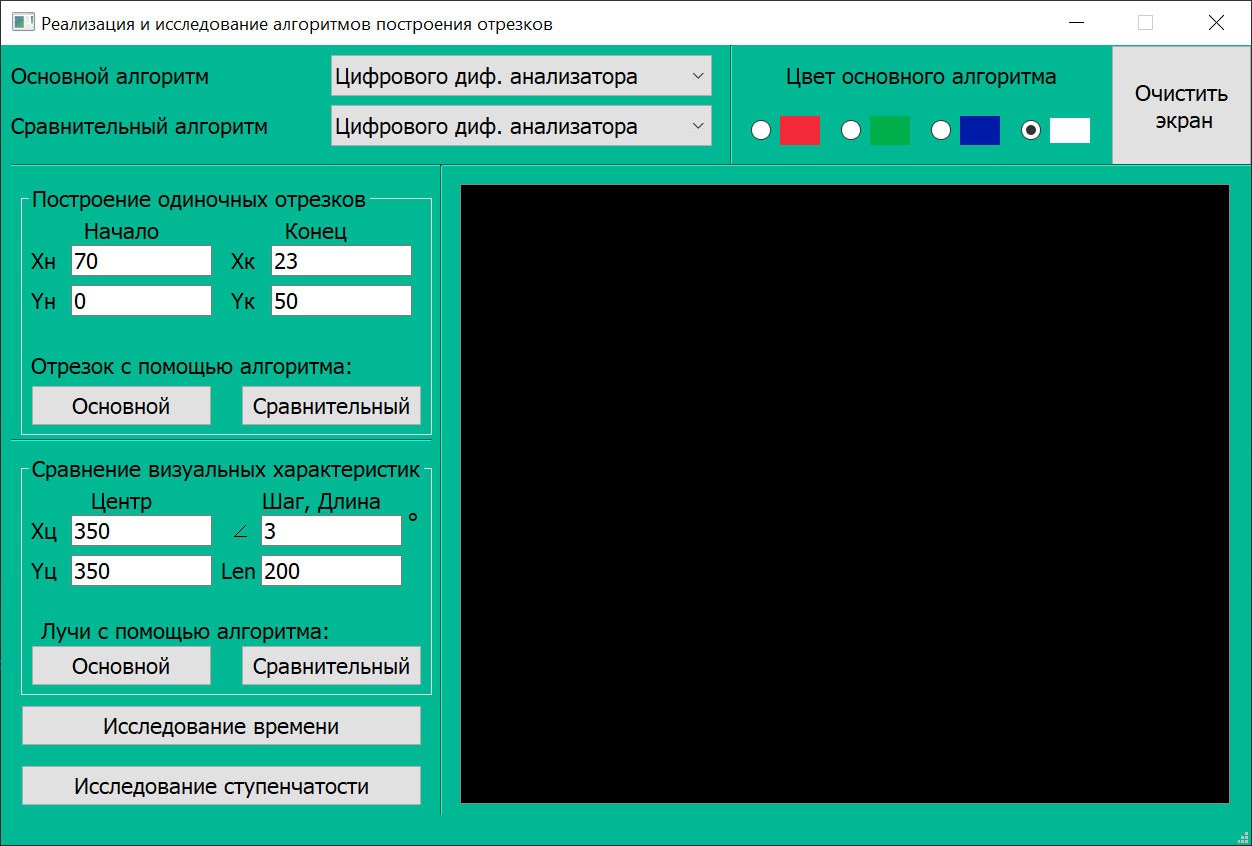
Реализация алгоритмов построения отрезков по методу цифрового дифференциального анализатора (ЦДА), алгоритмов Брезенхема (действительного, целочисленного и с устранением ступенчатости) и алгоритма Ву. Исследование их характеристик и сравнение полученных результатов.

**Описание задания**

1. Реализовать алгоритмы
2. Сравнить визуально отрезки, построенные в соответствии с каждым алгоритмом, а также с отрезком, построенным процедурой языка высокого уровня. Проверить попадание отрезка в заданную конечную точку.
3. Определить время, затрачиваемое на построение отрезка по каждому из алгоритмов.
4. Для заданного алгоритма получить зависимость длины максимальной ступеньки от угла наклона отрезка и отобразить ее в виде графика или гистограммы.

**Описание графического интерфейса**

Для использования функционала поставленных задач, был создан графический интерфейс

****

Интерфейс предоставляет возможность:

* Построения отрезка или пучка отрезков, в соответствии с выбранным алгоритмом, по заданным параметрам. Отрезок может быть нарисован одним из 4 цветов (Основной) или цветом фона (Сравнительный). Таким образом можно сравнить визуальные характеристики разных алгоритмов, нарисовав их по одинаковым параметрам, но разными цветами
* Исследование временных характеристик
* Исследование ступенчатости

**Описание алгоритмов**

Задача рассматриваемых отрезков обусловлена тем, что требуется изобразить непрерывный отрезок в дискретном (растровом) пространстве. Алгоритмы имеют следующие требования:

* Отрезок должен выглядеть прямым, начинаться и заканчиваться в заданной точке
* Яркость должна быть постоянна вдоль отрезка, и не должна зависеть от длины и наклона отрезка
* Алгоритм должен работать быстро

В данной лабораторной рассматриваются различные алгоритмы, которые, каждый в своей степени решает описанные выше требования. Схожей чертой у них является использование пошагового принципа.

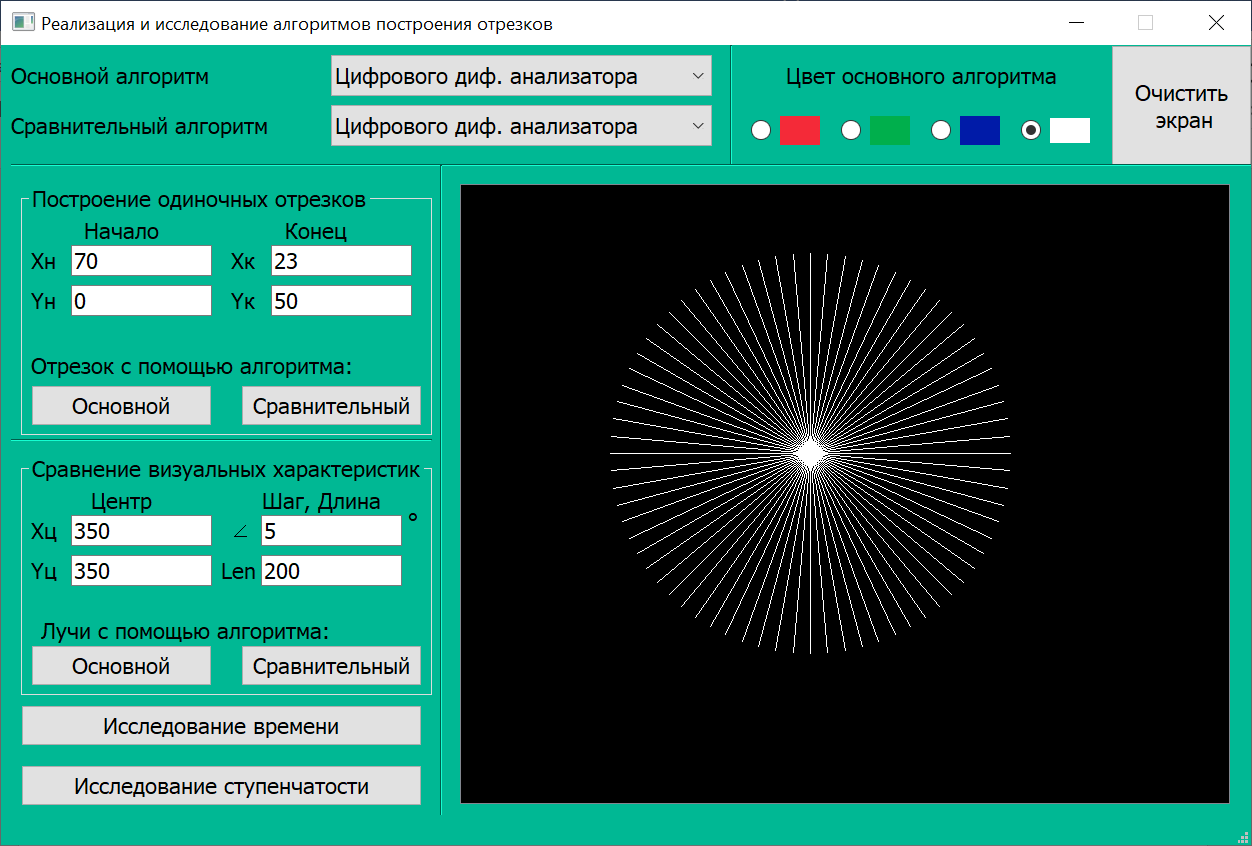
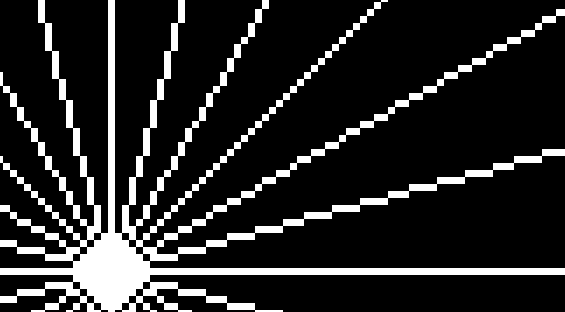
**Метод Цифрового Дифференциального Анализатора (ЦДА)**

Суть метода заключается в следующем. Перед началом выполнения цикла вычисляется DX и DY, равные разнице между координатами концов отрезка. Чтобы нарисовать все пиксели, затрагиваемые отрезком, достаточно выбрать приращение dx и dy, и пошагово высвечивать пиксели, высчитывая их координаты за счёт прибавления dx и dy к координатам предыдущего пикселя. Для растровой графики, все пиксели будут покрыты, если max(dx, dy) = 1. Поэтому их можно вычислить как   
dx = DX / max(|DX|, |DY|)

dy = DY / max(|DX|, |DY|)

Цикл проходит max(|DX|, |DY|) шагов, на каждом из них выводится пиксель (x, y) и производится увеличение x, y на dx, dy соотв. Изначально (x, y) задаётся как начало отрезка.

Результат работы алгоритма:

Плюсы алгоритма:

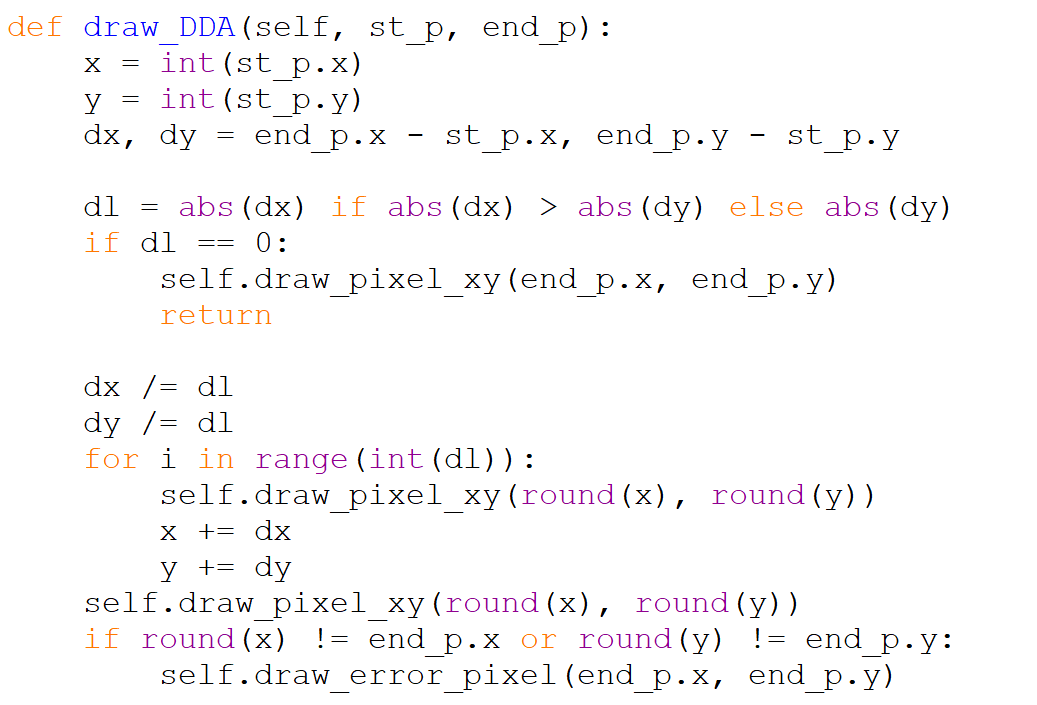
+ Простота реализации

Минусы алгоритма:

- Из-за того, что происходит оперирование вещественными числами, возрастает время построения отрезка, так как приходится производить округление к ближайшим целым

- Так же, из-за вещественных чисел возможно непопадание отрезка в заданную конечную точку

Код программы:

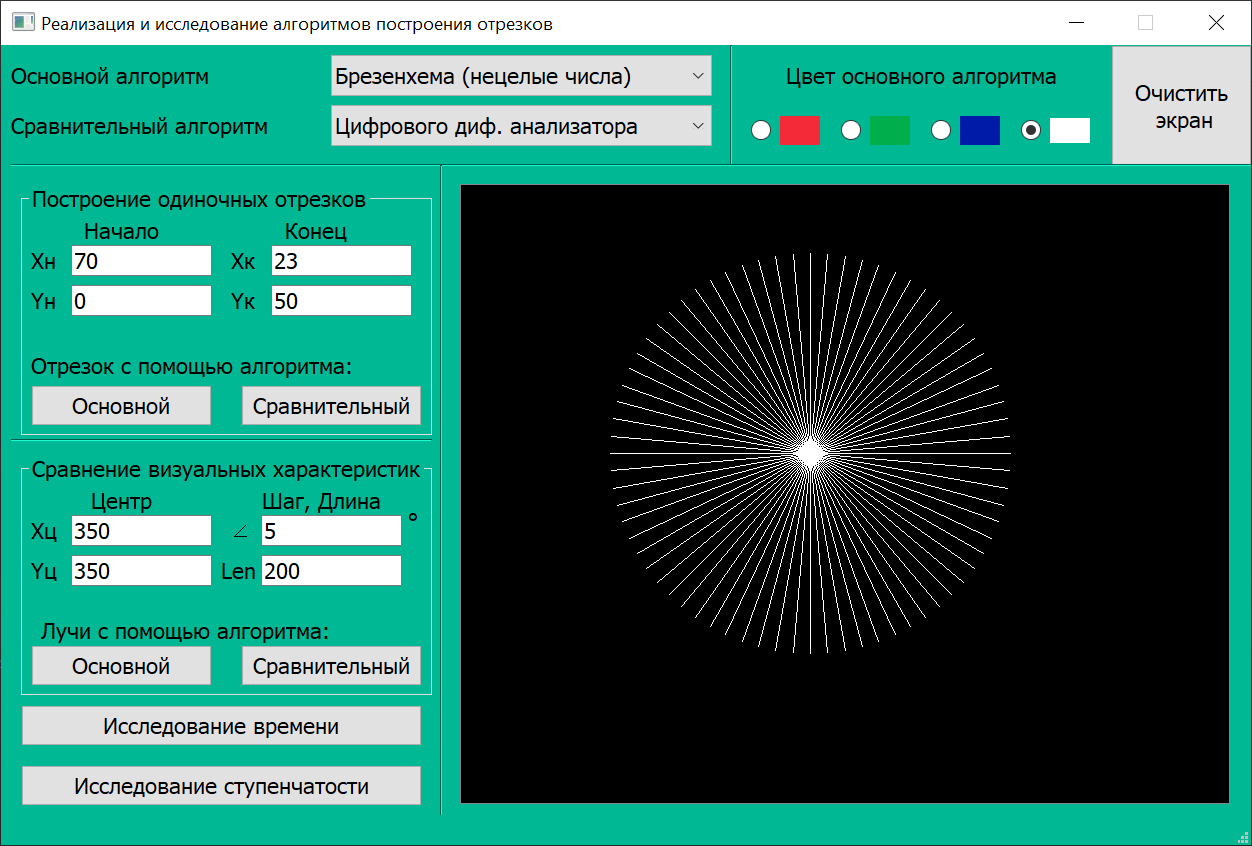


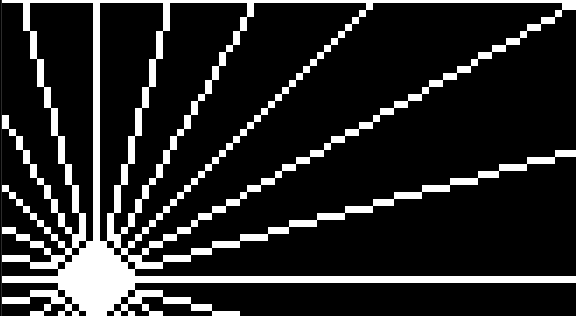
**Алгоритм Брезенхема (с вещественными числами)**

В процессе пошаговой отрисовки, алгоритм выбирает оптимальные пиксели для отображения отрезка. Выбор основывается на расстоянии между действительным положением отрезка и ближайшими координатами сетки пикселей. Такое расстояние называется ошибкой. В пошаговом выполнении производится анализ текущей ошибки, на основании которого одна из координат x, или y изменяется на 1 или 0. Другая координата прирастает на 1 на каждом шаге. После этого происходит высвечивание пикселя и изменение ошибки на величину, равную **dy/dx**.

Для сравнения с 0, изначально ошибка задаётся как **e = dy/dx – 0.5**

Результат работы алгоритма:





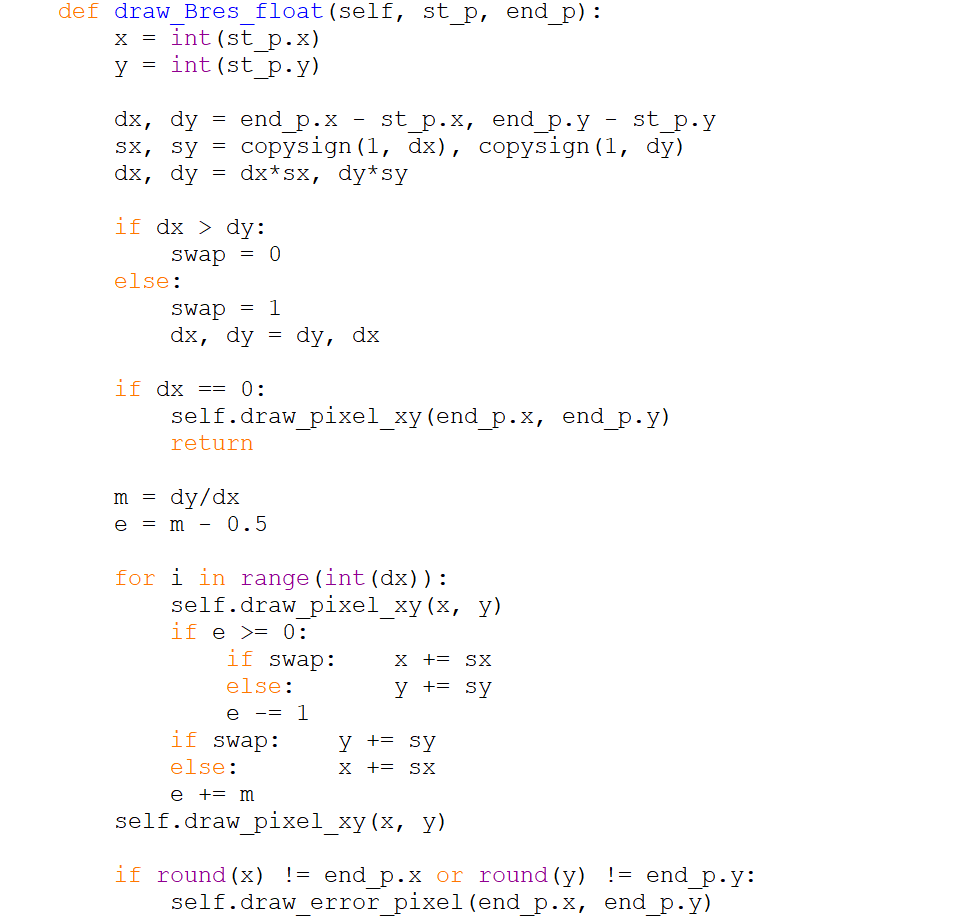
Плюсы алгоритма:

+ Не требуется округление координат пикселя, следовательно время лучше, чем в ЦДА

Минусы алгоритма:

- Присутствует оперирование вещественными числами в ошибке, значит возможно накопление погрешности.

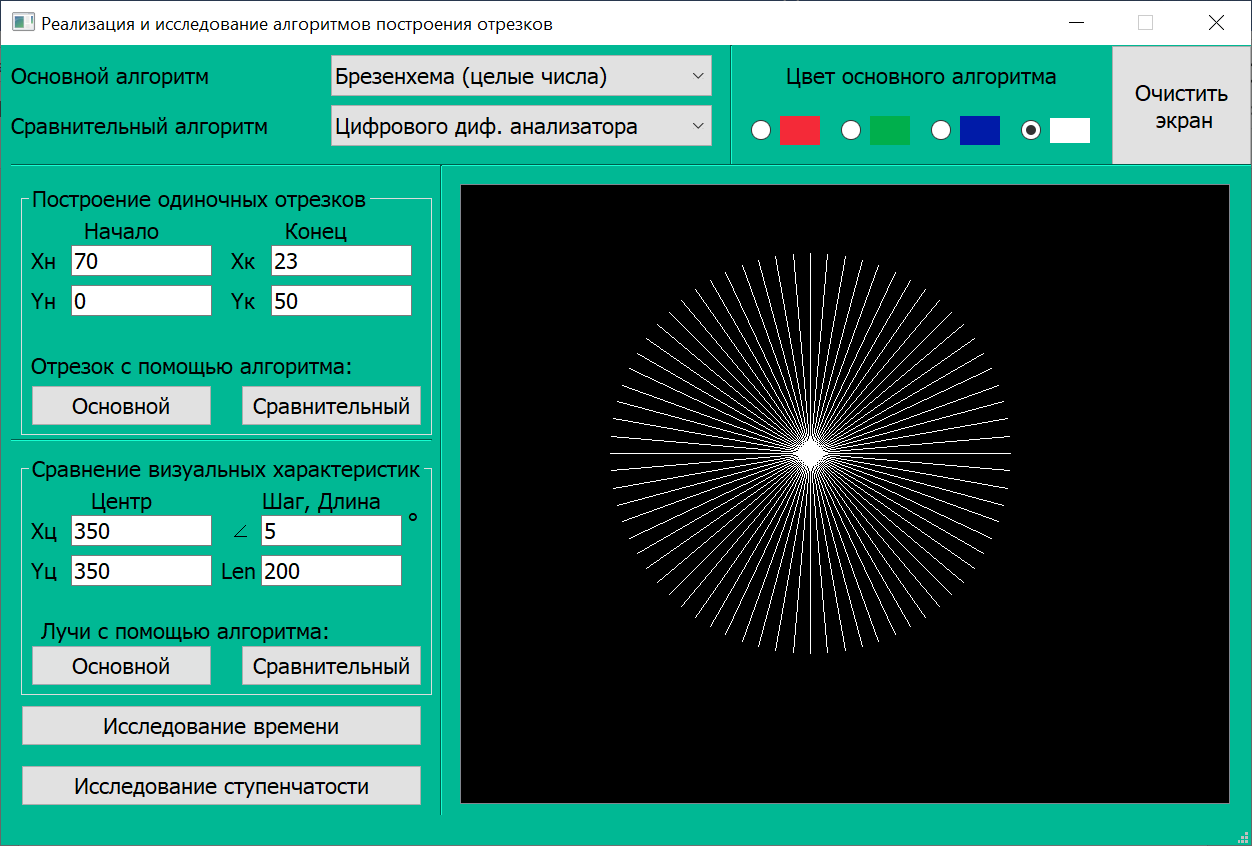
Код программы:

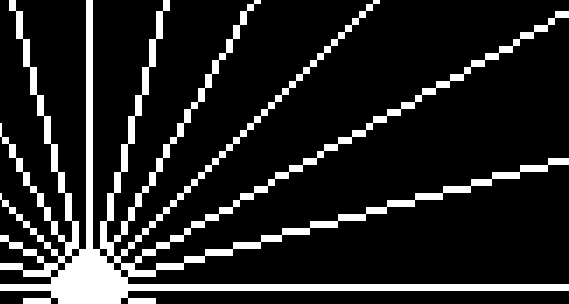


**Алгоритм Брезенхема (с целыми числами)**

Вышеописанный алгоритм Брезенхема можно улучшить, если исключить использование вещественных чисел. Этого можно добиться, если умножить все вычисления, связанные с ошибкой на 2dx. Таким образом изначально ошибка задаётся как **e = 2dy – 1**, а с каждым шагом увеличивается на 2dy.

Результат работы алгоритма:

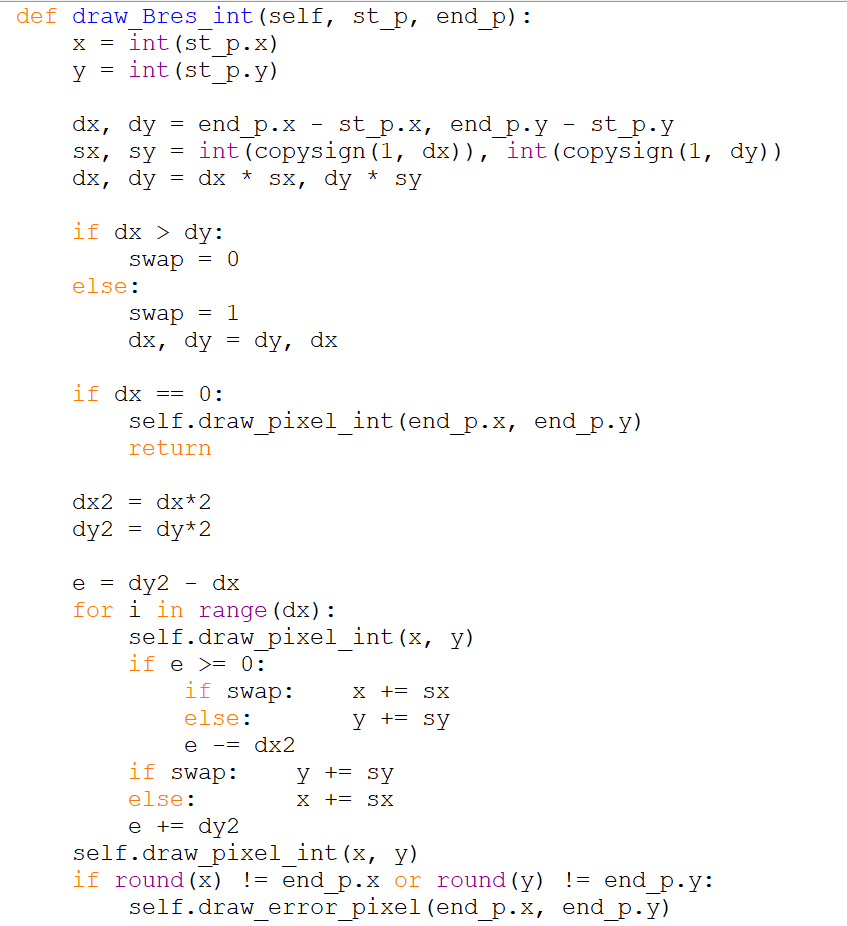




Плюсы алгоритма:

+ В процессе работы алгоритма не используются вещественные числа, следовательно исключена погрешность и сокращено время работы

Код программы:



**Алгоритм Брезенхема (со сглаживанием)**

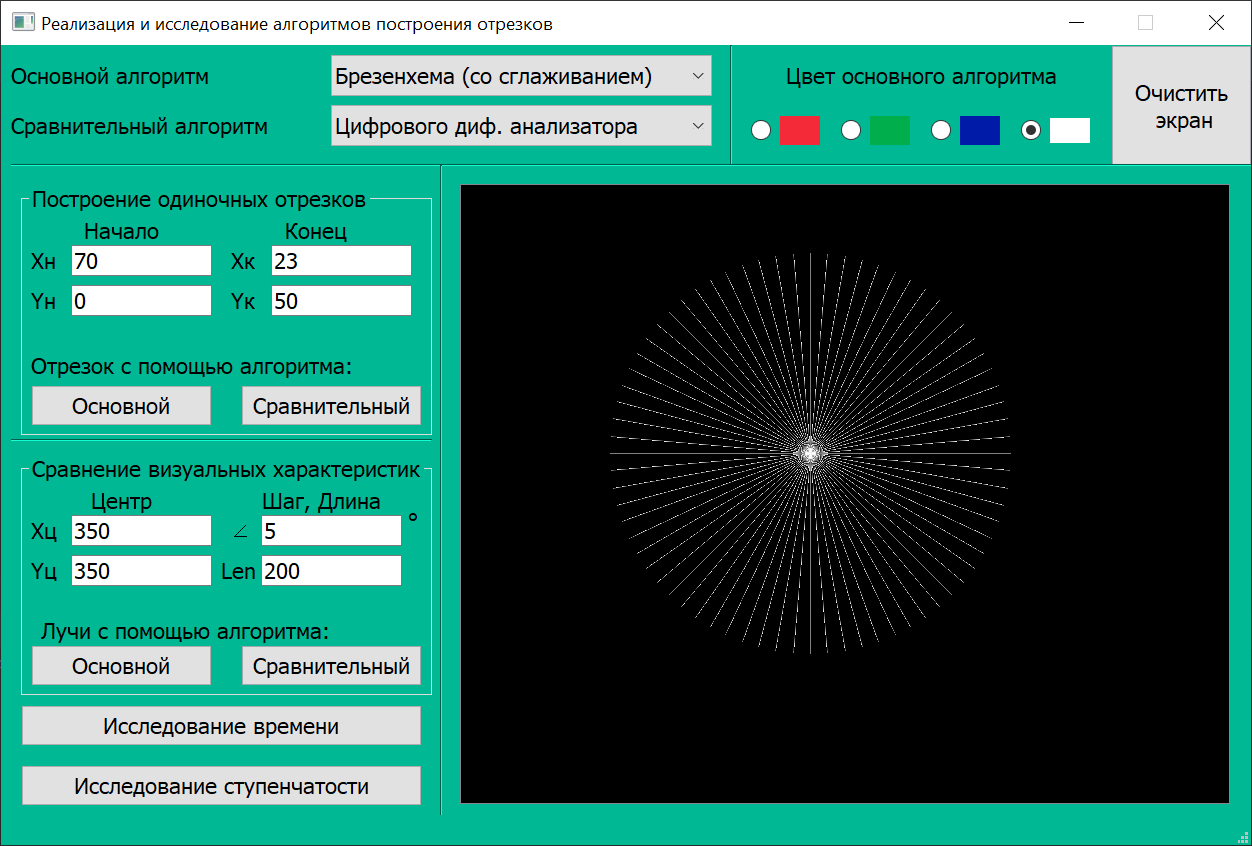
Алгоритм Брезенхема также может быть улучшен с помощью сглаживания ступенчатости. Ступенчатость устраняется за счёт регулирования интенсивности пикселя в зависимости от площади части пикселя, находящегося под отрезком. Рассмотрев случаи для соседних пикселей, получается, следующая зависимость площади пикселя от предыдущего:

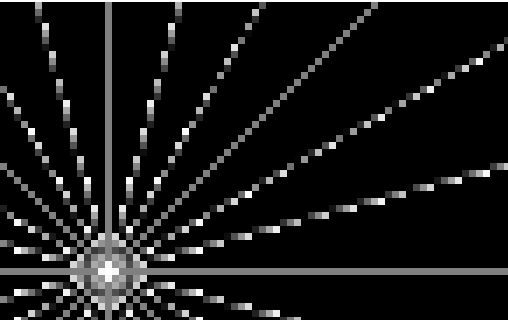
**Si = Si-1 + m**

В случаи перехода ступени, из S следует вычесть 1. Ошибка в таком случае ошибка сравнивается с w = 1-m

**е0 = m – 0.5, e = e + w**

Результат работы алгоритма:



****

Плюсы алгоритма:

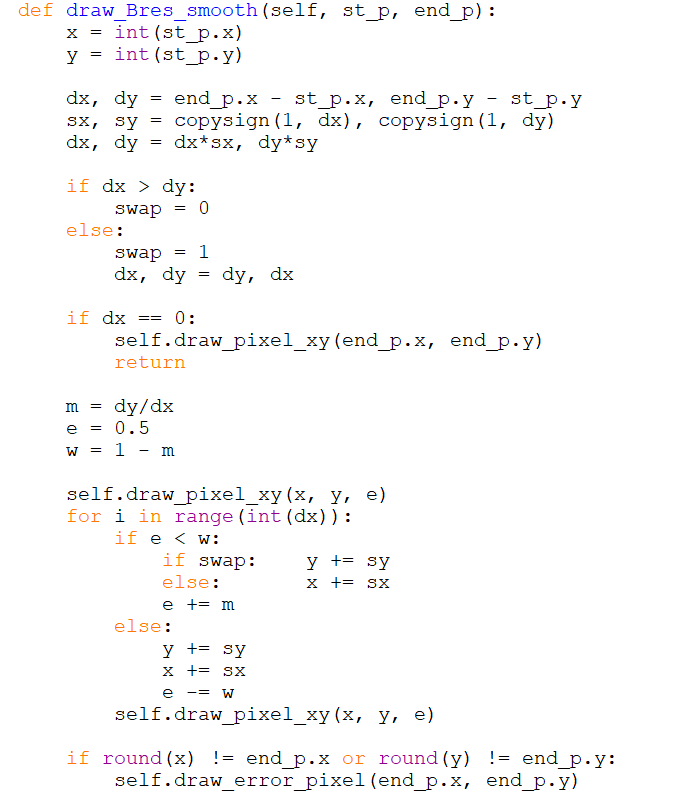
+ Создаёт устранение ступенчатости, которое может быть использовано при построении многоугольников, так как возможно закрашивание внутренней части ступени.

Минусы алгоритма:

- При рисовании отрезков, ступенчатость визуально не устранена

- Большие временные затраты, по сравнению с алгоритмом без сглаживания

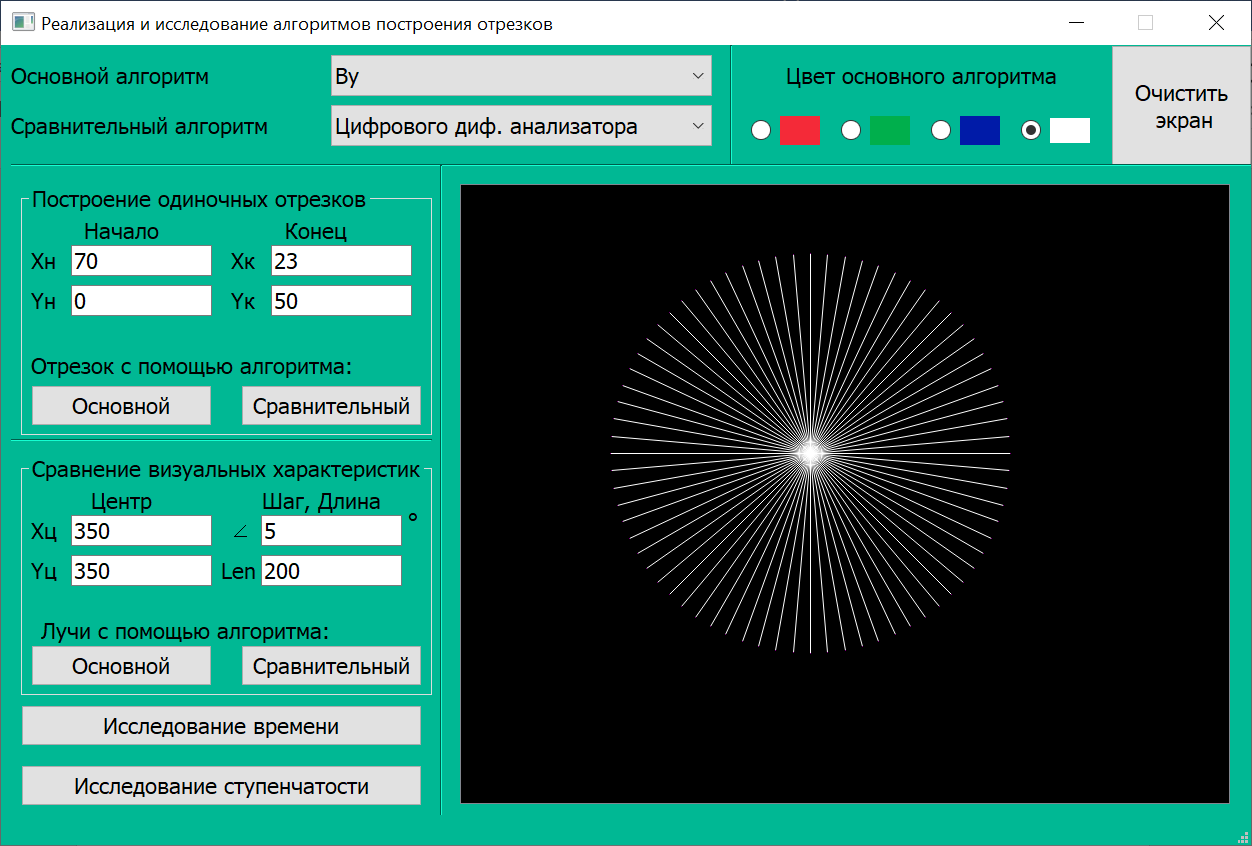
Код программы:

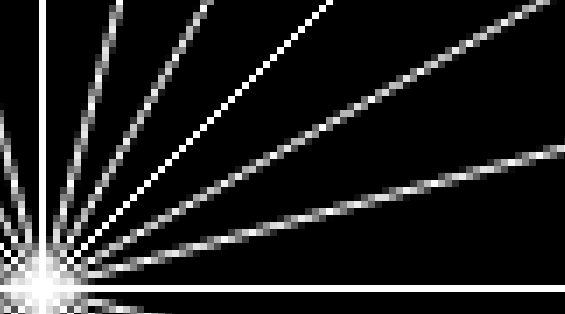


**Алгоритм Ву**

Алгоритм предназначен для создания изображения отрезка, наиболее приближённого к реальному. На каждом шаге высвечивается два соседних пикселя, с интенсивностями, вычисляемыми пропорционально их расстоянию до идеального расположения точки на отрезке. Таким образом, суммарная интенсивность на каждом шаге будет постоянна.

Результат работы программы:





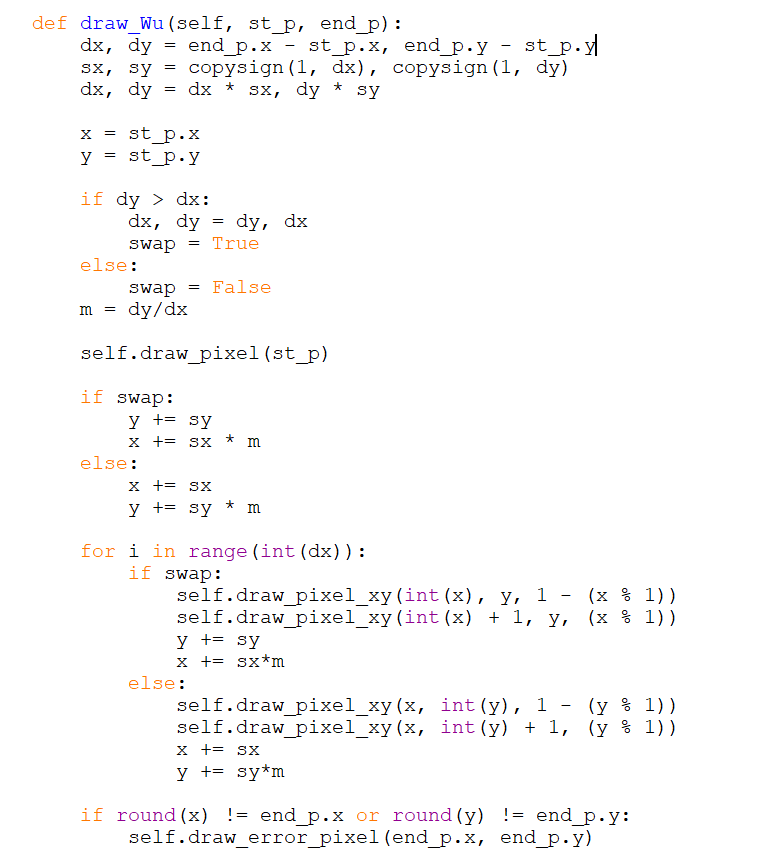
Плюсы алгоритма:

+ Наиболее приближенное к настоящему изображение отрезка

Минусы алгоритма:

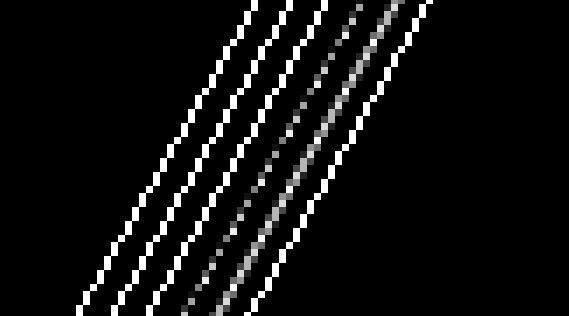
- Наибольшие временные затраты, ввиду использования вещественных чисел и высвечивания сразу двух пикселей на каждом шаге

Код программы:

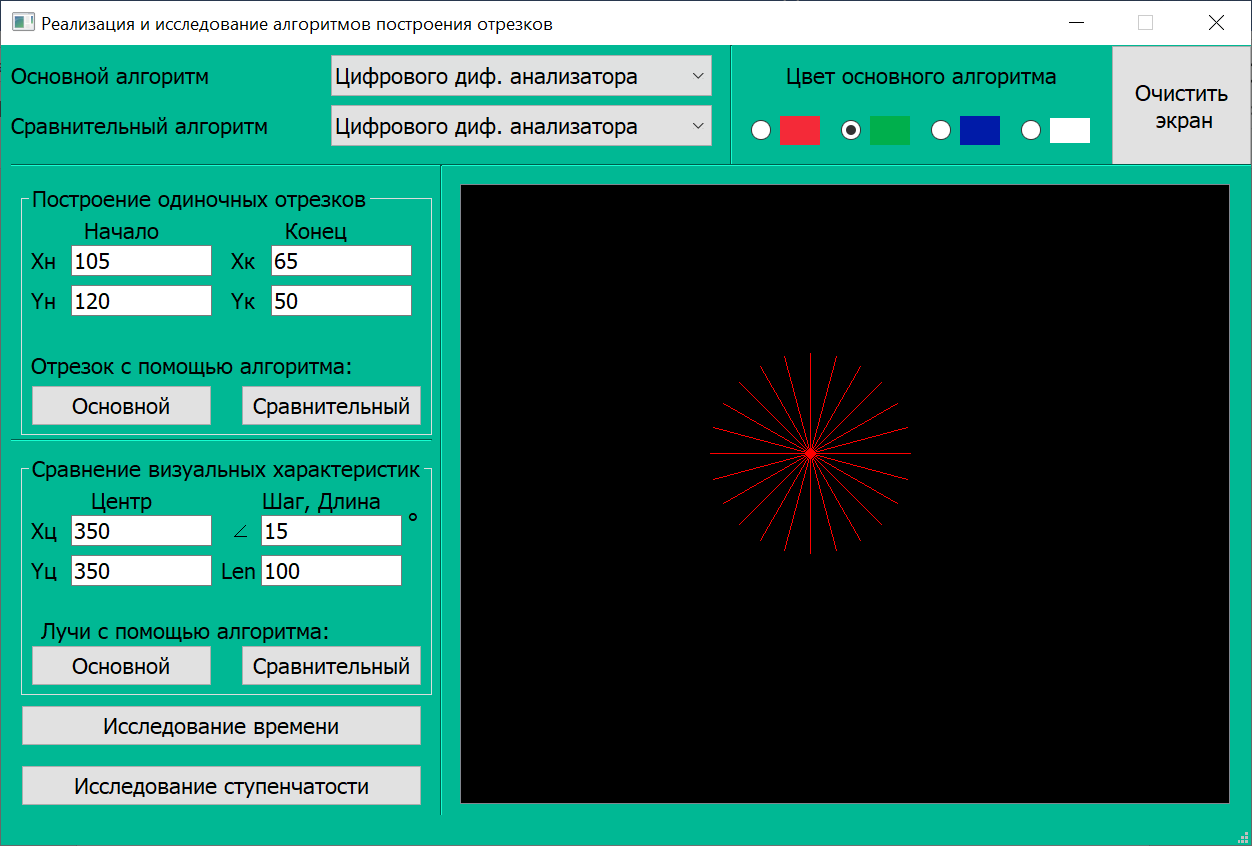


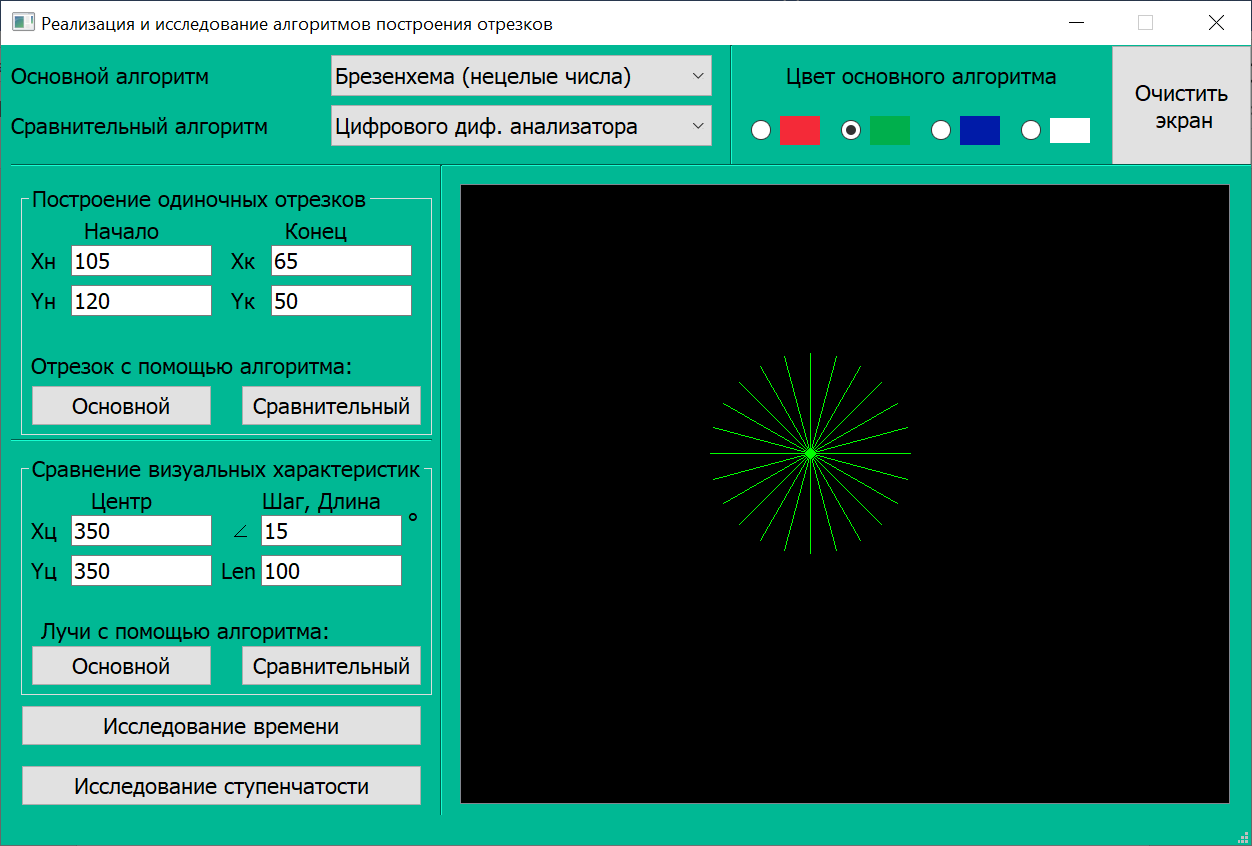
**Сравнение визуальных характеристик**

Слева-направо изображены алгоритмы ЦДА, Брезенхема (вещественные числа, целые и с устранением ступенчатости) и Ву и встроенный метод рисования отрезка графического интерфейса.



Из изображения видно, что ЦДА и Брезенхем без сглаживания не имеют визуальных различий, что подтверждается также при наложении одного отрезка на другой, с разными цветами.



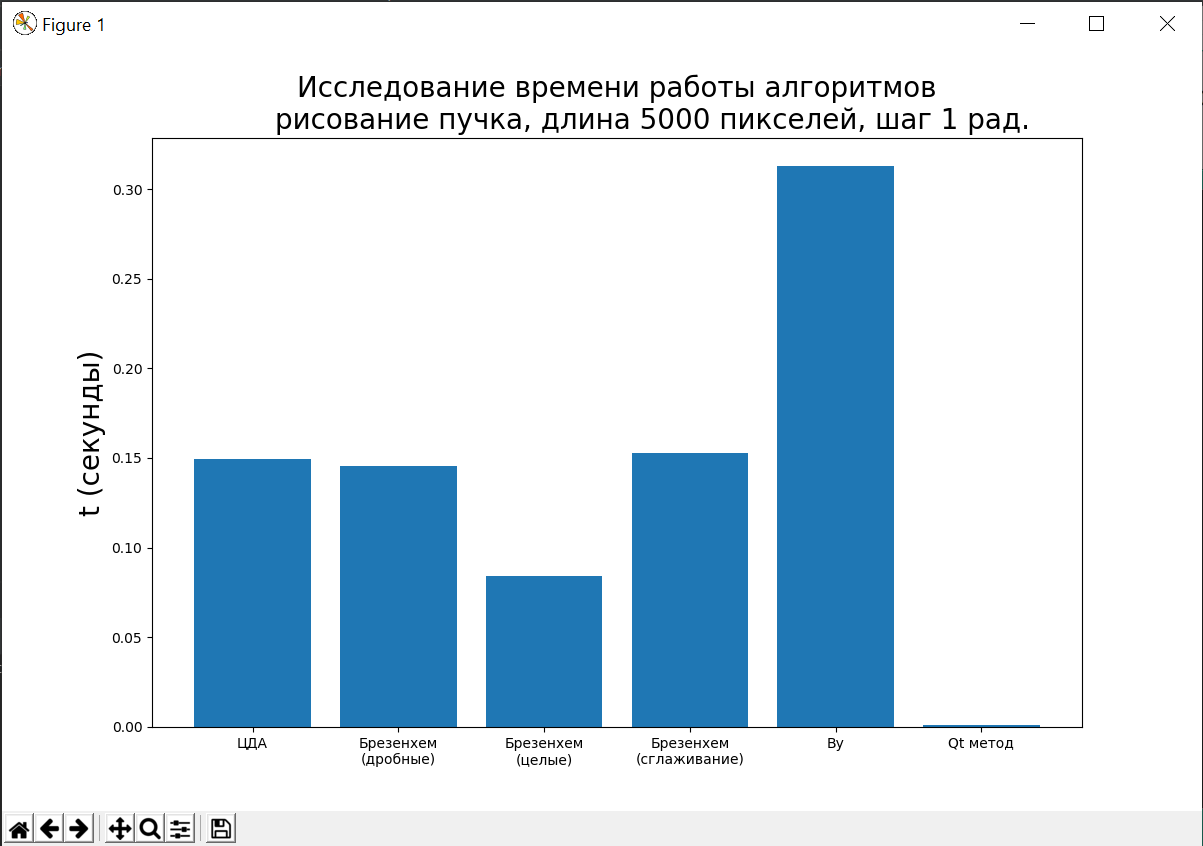


(ЦДА был нарисован красным цветом, и на той же точке был нарисован вещественный Брезенхем зелёным цветом)

Также, на изображении видно, что больше всего к реальному изображению отрезка приближен отрезок с помощью метода Ву.

**Анализ временных характеристик**

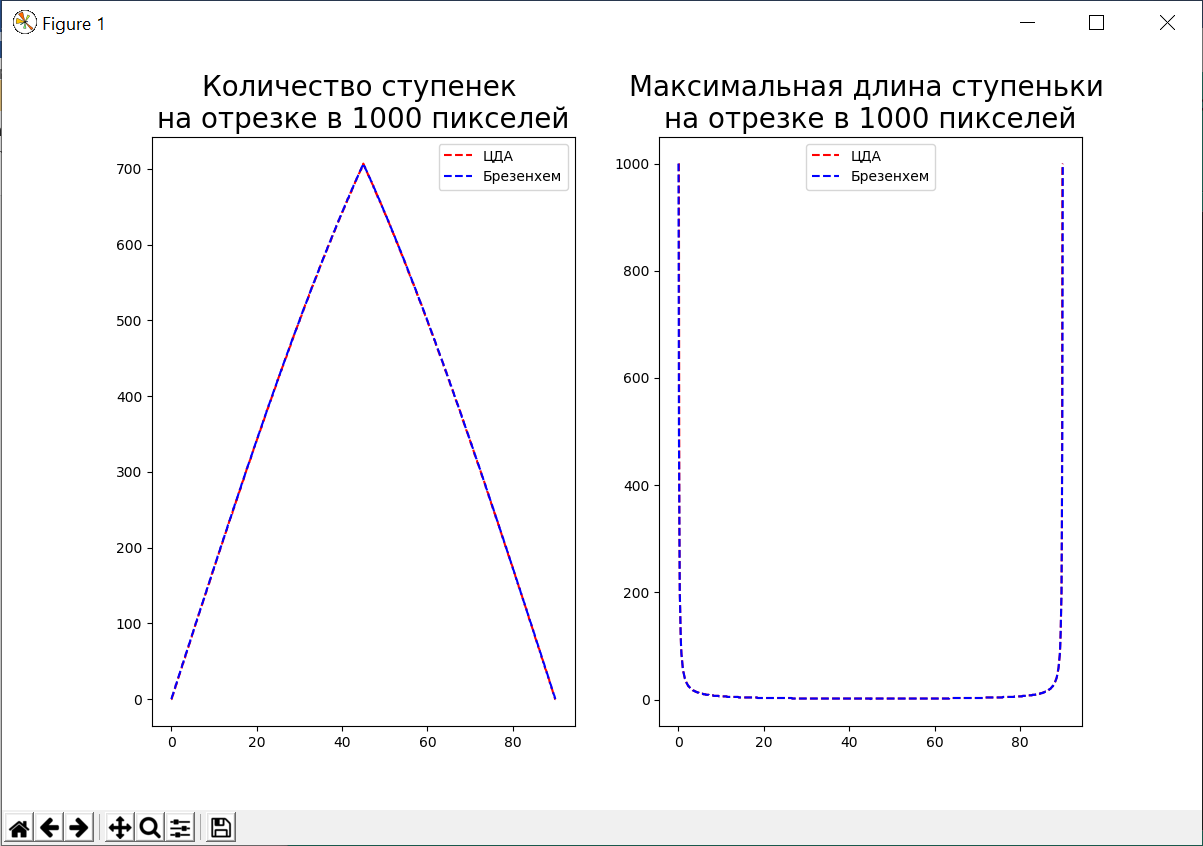
В программе реализовано средство замера времени работы различных алгоритмов



Из графика становится видно, что наиболее быстрым среди рассмотренных является алгоритм Брезенхема на целых числах, а наиболее медленным – алгоритм Ву. Причины этого были рассмотрены ранее, в анализе алгоритмов. Также стоит отметить, что встроенный метод графического интерфейса занимает гораздо меньше времени, что может быть вызвано тем, что при высвечивании пикселей он использует более быстро действенные функции. Также заметна и разница между целочисленным и вещественным Брезенхемом, достигнутая за счёт оптимизации.

**Анализ ступенчатости**

В программе реализовано средство замера количества ступенек и максимальной длины ступенек при отрисовке отрезков различными алгоритмами.



Из графика видно, что длина ступенек, убывает при отдалении отрезка от вертикали и горизонтали, в то время как количество ступенек возрастает. Также подтверждается и то, что ЦДА и Брезенхем не имеют визуальных отличий, так как их графики практически не имеют различий.

**Заключение**

В лабораторной работе были изучены различные алгоритмы построения растровых отрезков. Проведено сравнение их визуальных и временных характеристик, выявлены преимущества и недостатки.