|  |  |
| --- | --- |
| **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 (ИУ7)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.04 Программная инженерия**

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 3 |

**Название:**

Реализация и исследование алгоритмов построения отрезков

**Дисциплина:** Компьюетрная графика

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ7-43Б |  | 25.03.2020 | С.С. Кононенко |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | А.В. Куров |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2020

**1. Цель работы**

Научиться выполнять построение отрезков различными алгоритмами и проанализировать их.

**2. Техническое задание**

1. Рисование отдельных отрезков и сравнение их визуальных характеристик различными методами, среди которых:
   1. алгоритм цифрового дифференциального анализатора (ЦДА);
   2. алгоритмы Брезенхема (с дробными числами, с целыми числами, с устранением ступенчатости);
   3. алгоритм Ву;
   4. библиотечный алгоритм.
2. Исследование визуальных характеристик для отрезка, расположенного во всём спектре изменения углов;
3. Исследование временных характеристик (результат исследования представить в виде гистограммы).

**3. Теоретический материал**

Процесс определения пикселей, наилучшим образом аппроксимирующих положение отрезка, называется *разложение объекта в растр*.

Общие требования к построению отрезков:

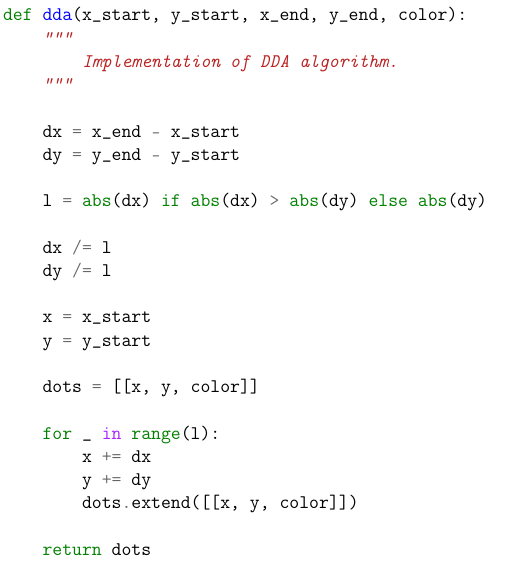
1. Отрезки выглядят как отрезки прямых;
2. Отрезки заканчиваются и начинаются в заданных точках;
3. Интенсивность вдоль отрезка должна быть постоянной. Отрезки, имеющие разные углы наклона, должны быть одинаковой интенсивности. Восприятие человека зависит не только от интенсивности свечения объекта, но и от расстония между светящимися объектами. Чтобы удовлетворить этому требованию, надо высвечивать точки с переменной интенсивностью от расстония, что потребует дополнительных вычислений, без особой нужды не используется;
4. Алгоритмы должны работать быстро.

Все алгоритмы имеют пошаговый характер: на очередном шаге высвечивается пиксель, затем производятся вычисления, используемые в следующем шаге.

**4. Реализация алгоритмов**

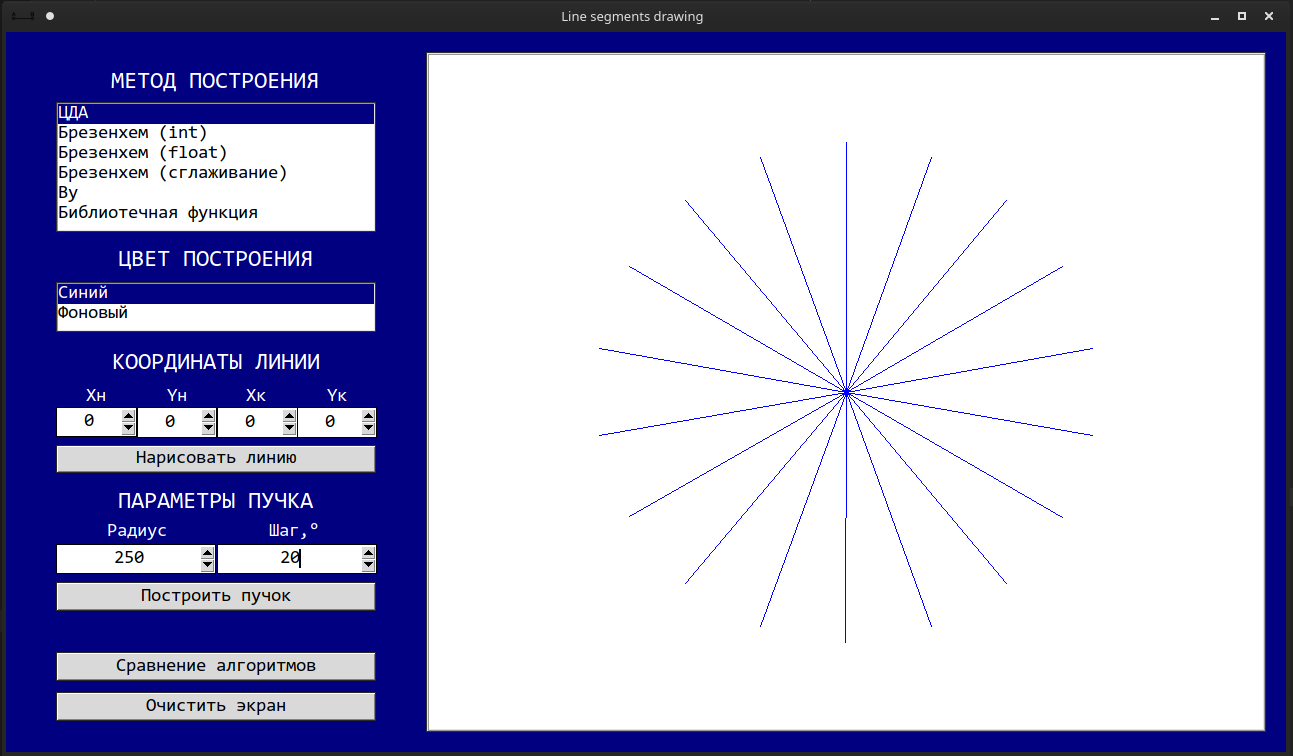
Работа алгоритмов будет демонстрироваться на *пучке отрезков*.

**4.1 Алгоритм цифрового дифференциального анализатора**

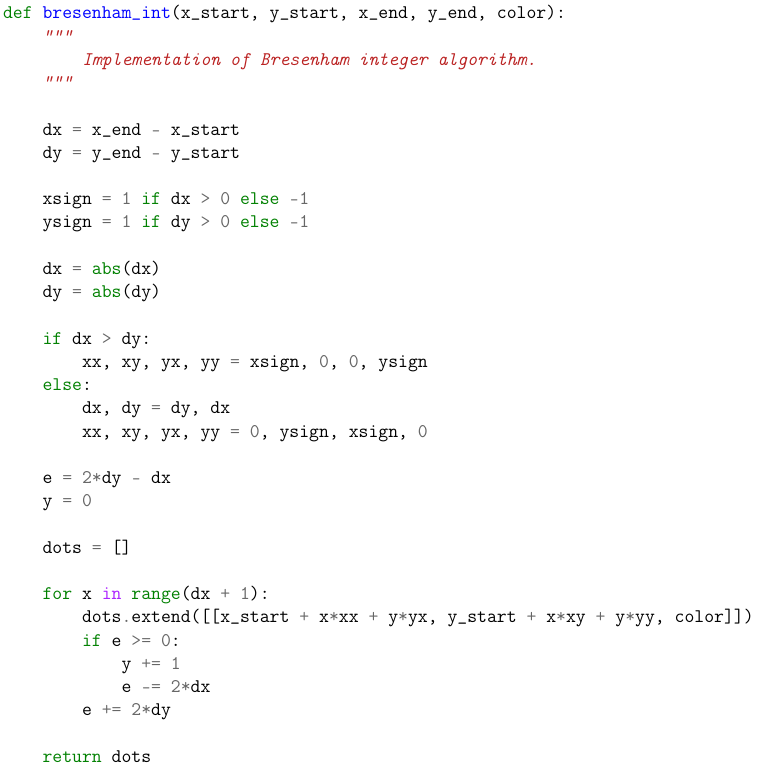
****

В алгоритме используется подход «в лоб»: высчитывается отношения приращения прямой к приращению ее аргумента и определяется, какое приращение имеет сама прямая относительно приращения аргумента (приращение аргумента берется равное 1, а приращение функции < 1). Прибавляя к аргументу единицу, а к значению прямой ее приращение (yi+1 = yi + δy/δx, где δy/δx — есть производная прямой) будут находиться последующие пиксели. Очевидно, что аргумент будет считаться в целых числах, а значение прямой — в дробных (и округляться при расчете координат пикселя).

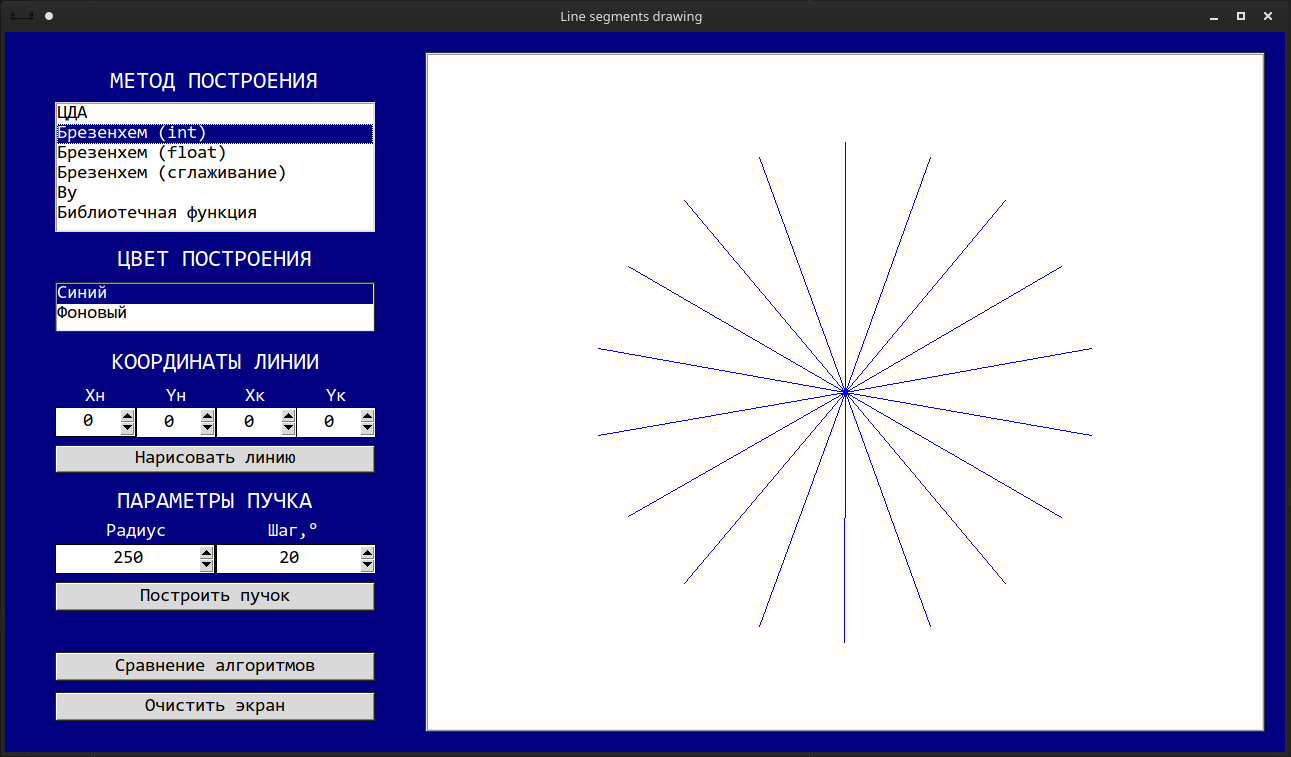
Основной недостаток данного алгоритма - математическое округление в цикле, из-за чего падает скорость алгоритма.



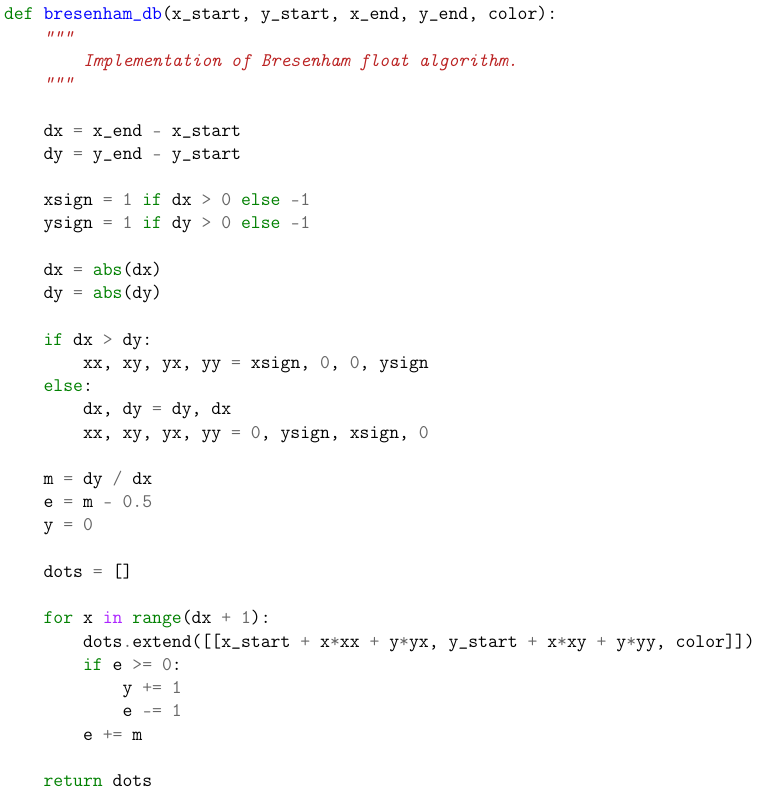
**4.2. Алгоритм Брезенхема с целочисленными коэффициентами**

****

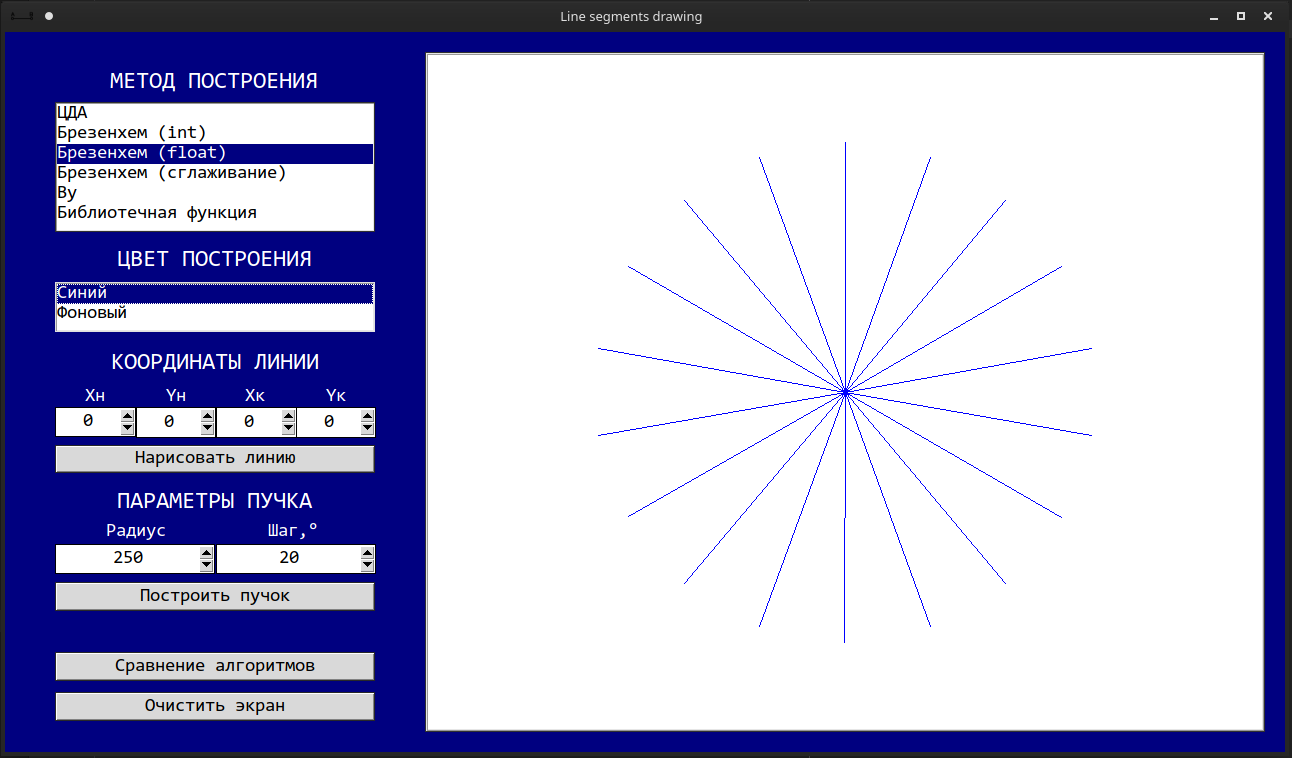
Данный алгоритм использует только сложение, вычитание и умножение для подсчета ошибки. Таким образом, происходит избавление от вещественных коэффициентов.



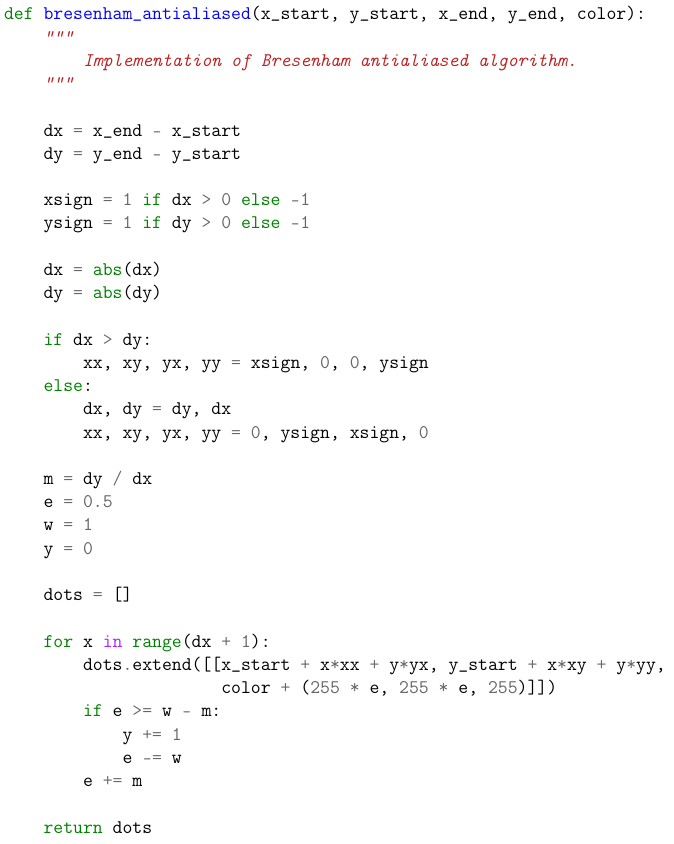
**4.3. Алгоритм Брезенхема с действительными коэффициентами**

****

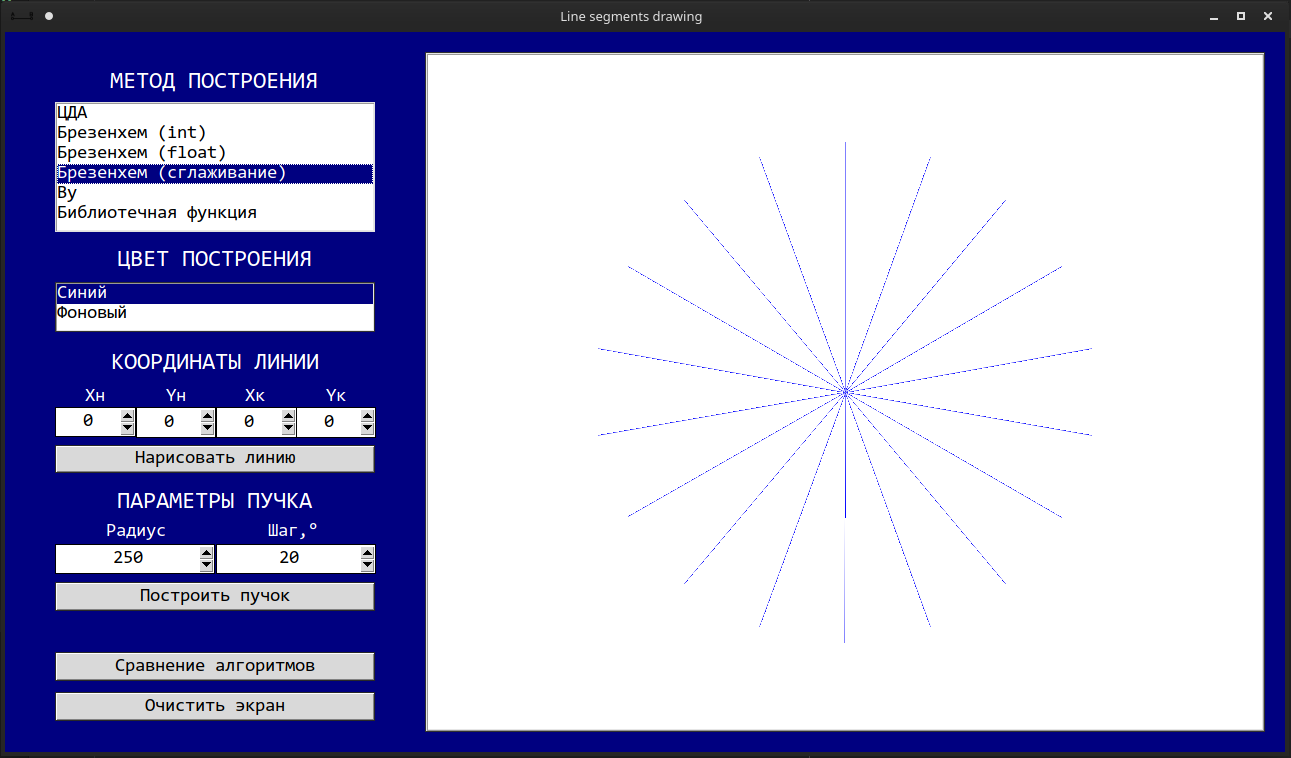
Ошибка ei - величина, изменяющаяся на каждом шаге, расстояние между точкой отрезка и точкой, аппроксимирующей его на очередном шаге. 0 ≤ ei ≤ 1. Если e < 0.5, то ордината пикселя не меняется, yi+1 = yi. Если e > 0.5, то выбирается верхний пиксель, yi+1 = yi + 1. Начальное значение ошибки e = m, где m - тангенс угла наклона отрезка, а ei+1 = ei + m = ybi − yi+1 = ybi+1 = ybi+m − yi, где yb - ордината идеального отрезка в вещественных координатах.



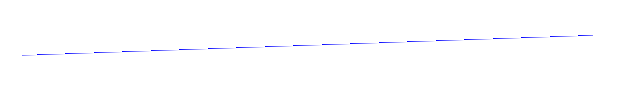
**4.4. Алгоритм Брезенхема с устранением ступенчатости**

****

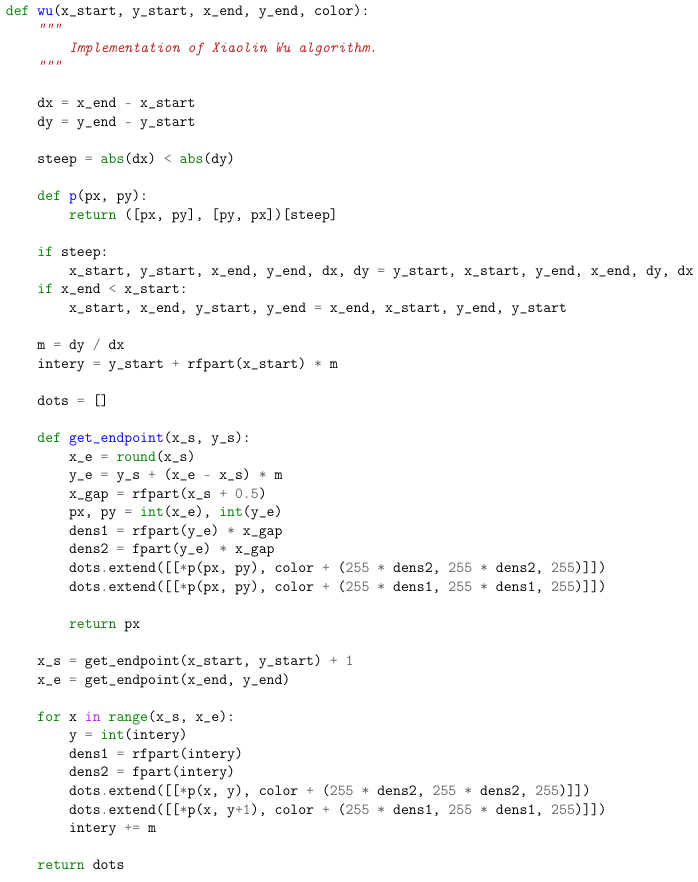
Данный алгоритм применяется при отображений ребер многоугольника, внутри которого происходит заливка. Идея заключается в сглаживании резких переходов от ступени к ступени. Сглаживание основывается на высвечивании пикселя. Интенсивность пикселя выбирается пропорционально площади части пикселя. Стоит отметить, что в данном случае один пиксель - это квадрат с единичной стороной, а не математическая точка.



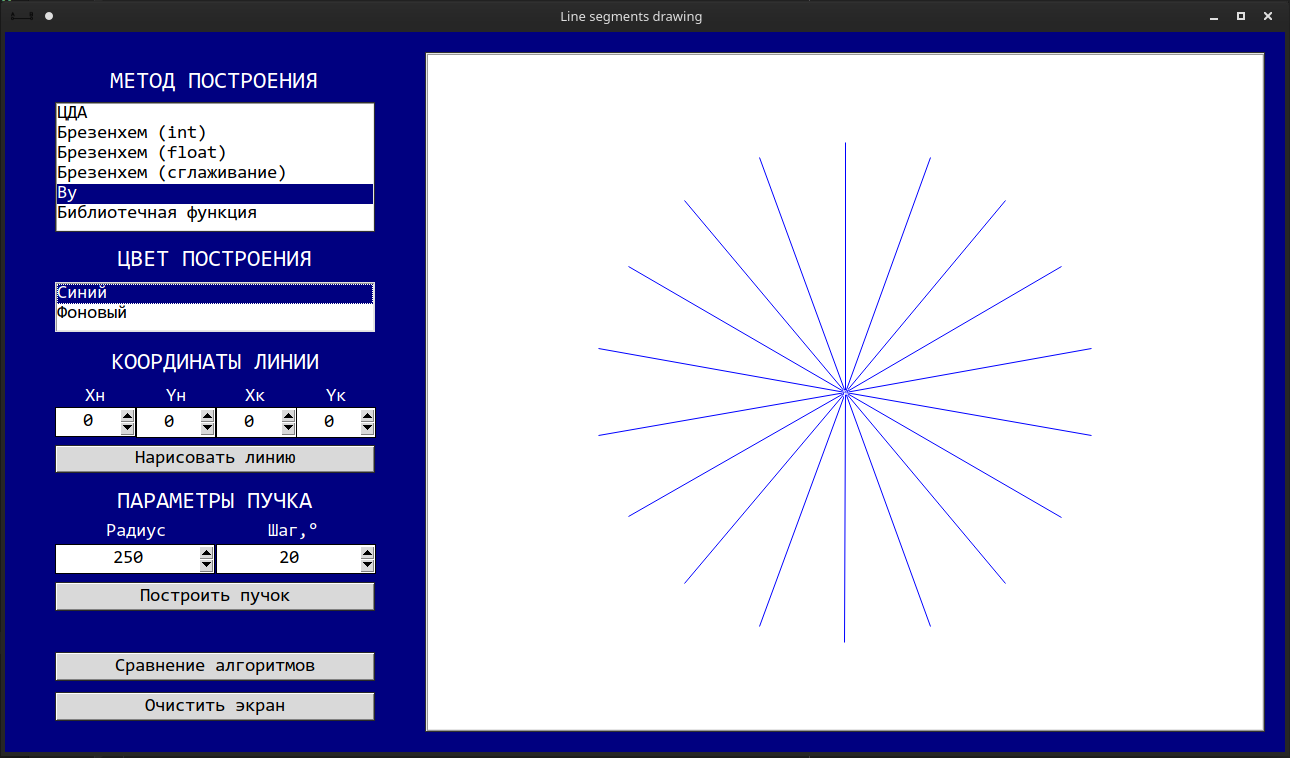
Данное изображение доходчиво показывает, что данный алгоритм будет хорошо применим для отображения многоугольников, внутренняя часть которых будет находиться над прямой, для дальнейшего закрашивания фигуры.



**4.5. Алгоритм Ву**

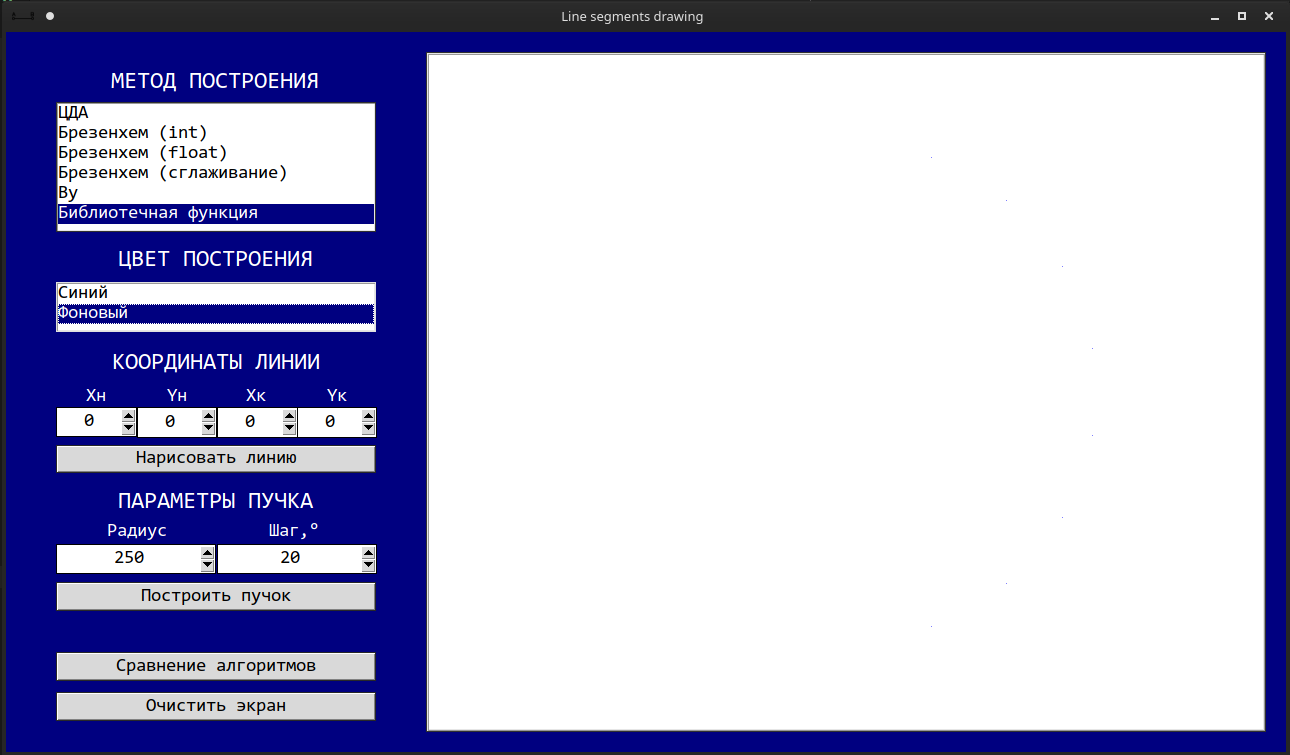
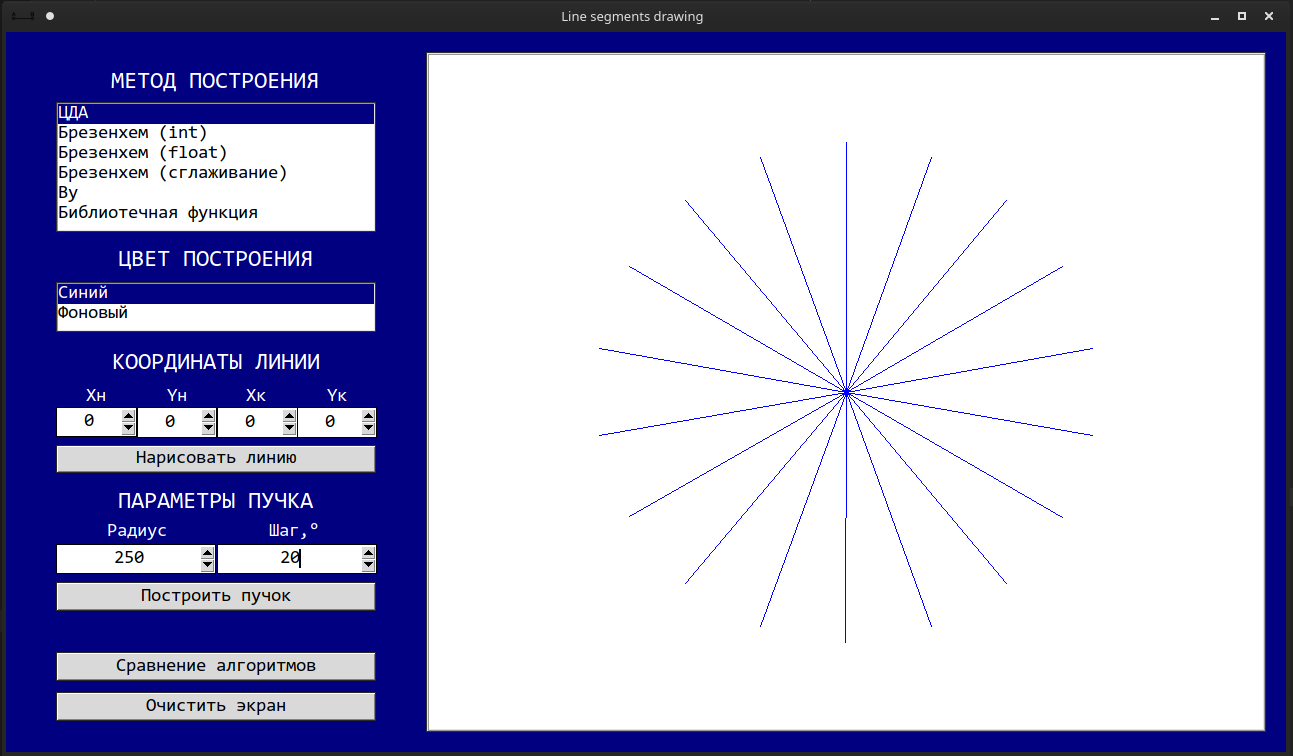
****

Данный алгоритм сглаживает отрезок толщиной 2 пикселя. Сглаживание осуществляется за счет перераспределения на каждом шаге интенсивности Iconst между двумя пикселями. Суммарная интенсивность двух пикселей, высвеченных на каждом шаге, постоянна (Ii1 + Ii2 = Iconst). Интенсивность пикселя задается в зависимости от расстояния между пикселем и точкой, расположенной на идеальном отрезке: чем ближе пиксель расположен к точке на идеальном отрезке, тем боольше его интенсивность.



**4.6. Библиотечная функция**

Лабораторная работа написана на языке программирования Python, используемая библиотека - Tkinter. При наложении любого алгоритма построения отрезка толщиной 1 пиксель без сглаживания (алгоритмы ЦДА, Брезенхема (как с целочисленными, так и с вещественными коэффициентами)), происходит полное совпадение, за исключением конечных точек в правой части пучка, что можно списать на странное поведение библиотечной функции. Предположение об использумемом методе в библиотечной функции будем строить на основе сравнения временных характеристик.



**5. Сравнение временных характеристик**

Проведенные исследование показали, что библиотечная функция использует, вероятнее всего, алгоритм Брезенхема. Во время измерений для каждого метода 10000 раз строился пучок с параметрами, использованными для демонстрации рабты алгоритмов, приведенных выше. Затем находилось среднее время, затраченное на выполнение повторений. Найденное время и было использовано в качестве демонстрационных данных.

