Государственное учреждение образования

“БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ”

Кафедра: Интеллектуальных информационных технологий

Дисциплина: Графический интерфейс интеллектуальных систем

Отчет по лабораторной работе №1

**“Алгоритмы построения отрезков”**

Выполнили:

студент гр.121702

Яхья-заде Айлин

Витковская С. И.

Проверил:

Сальников Д.А.

Минск 2024

**1. Цель работы**

Разработать элементарный графический редактор, реализующий построение отрезков с помощью алгоритма ЦДА, целочисленного алгоритма Брезенхема и алгоритма Ву. Вызов способа генерации отрезка задается из пункта меню и доступно через панель инструментов «Отрезки». В редакторе кроме режима генерации отрезков в пользовательском окне должен быть предусмотрен отладочный режим, где отображается пошаговое решение на дискретной сетке.

**2. Ход работы**

1. Разработка графического интерфейса, позволяющего производить элементарное закрашивание пикселей.
2. Реализация варианта соединения двух закрашенных пикселей алгоритмом построения отрезков ЦДА.
3. Реализация варианта соединения двух закрашенных пикселей алгоритмом Брезенхема.
4. Реализация варианта соединения двух закрашенных пикселей алгоритмом Ву.

**3. Реализация**

**3.1. Алгоритм ЦДА**

1. На вход берутся координаты начальной точки (x1, y1) и конечной точки (x2, y2) отрезка. Вычисляются длина и высота отрезка ∆x и ∆y:

let dx = x2 - x1;

let dy = y2 - y1;

1. Выбирая максимум из длины и высоты линии, находится минимальное количество итераций, необходимое для отрисовки отрезка.

let steps = Math.max(Math.abs(dx), Math.abs(dy));

1. Вычисляются приращения на каждом шаге по осям абсцисс и ординат:

let x\_increment = dx / steps;

let y\_increment = dy / steps;

1. Задаются начальные значения x и y, опустошается массив connected\_dots:

let connected\_dots = [];

var x = x1;

var y = y1;

1. Запускается основной цикл генерации отрезка:

for (var i = 0; i <= steps; i++){

connected\_dots.push([Math.round(x), Math.round(y)]);

x = x + x\_increment;

y = y + y\_increment;

}

1. После заполнения массива connected\_dots, для каждой точки вызывается функция drawPixel(x, y), закрашивающая пиксель в выбранный цвет.

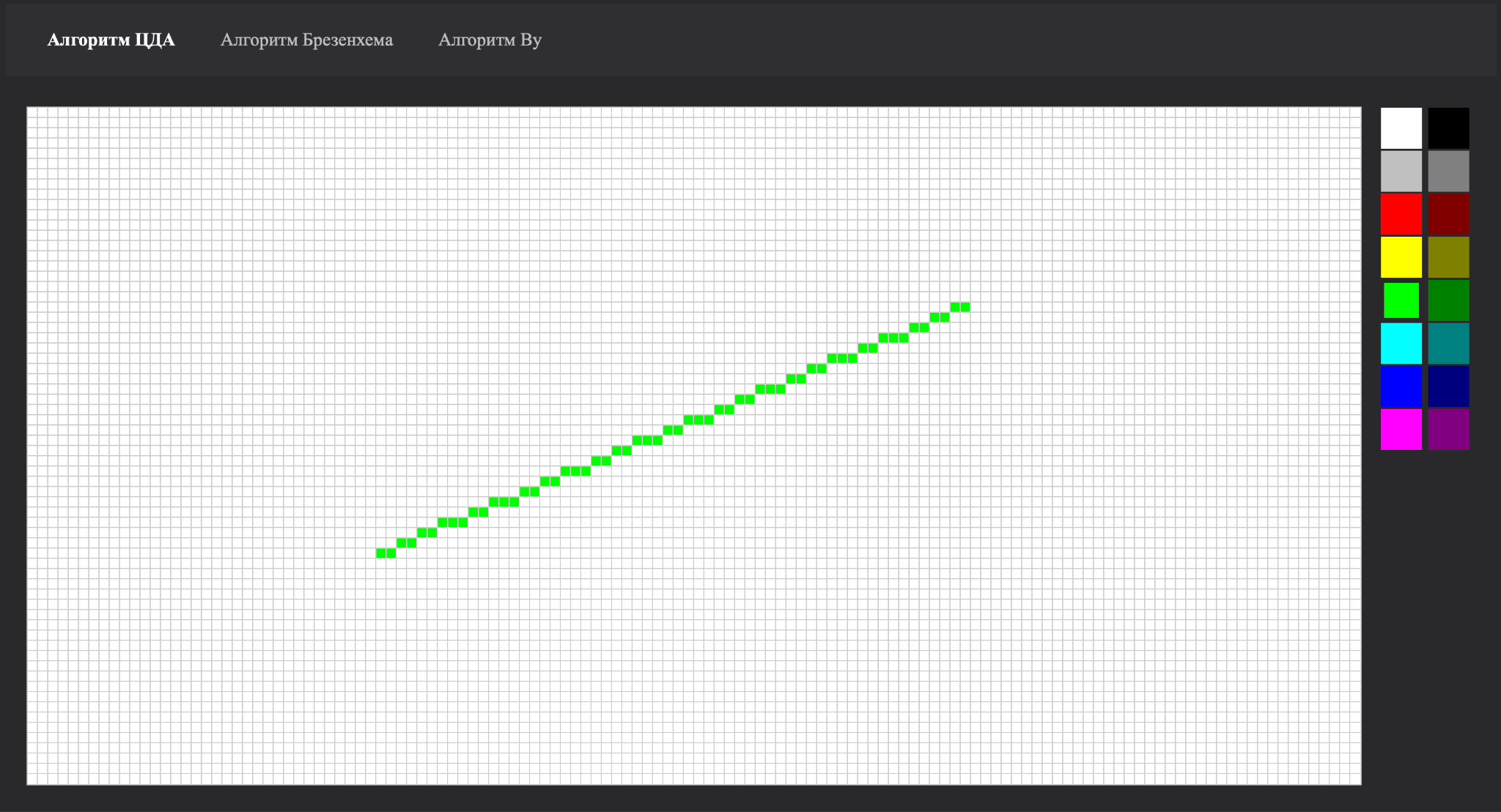
****

Рисунок 3.1 - Пример работы программы с использованиям алгоритма ЦДА.

**3.2. Алгоритм Брезенхема**

1. Определяется направление движения путем сравнения координат начальной и конечной точек (вправо/влево, вверх/вниз). От него зависят знаки коэффициентов, использующихся в цикле перемещения;
2. Определяется ось движения путем сравнения dx и dy(если отрезок более вертикален, счет будет вестись по оси Y, горизонтален – оси X);
3. Итерация по пикселям(в зависимости от определенного ранее направления, значения координат увеличиваются или уменьшаются). На каждой итерации вычисляется *ошибка.* Ошибка является расстоянием между текущим положением и точным значением функции, задающей прямую, фрагментом которой является рассматриваемый отрезок.

Если на предыдущем этапе была выбрана ось X, тот на каждой итерации будет изменяться значение y, значение х будет изменяться, если ошибка достигнет некоего порогового значения.

1. Координаты подходящих пикселей заносятся в массив connected\_dots, который в качестве аргумента передается в метод drawPixel.

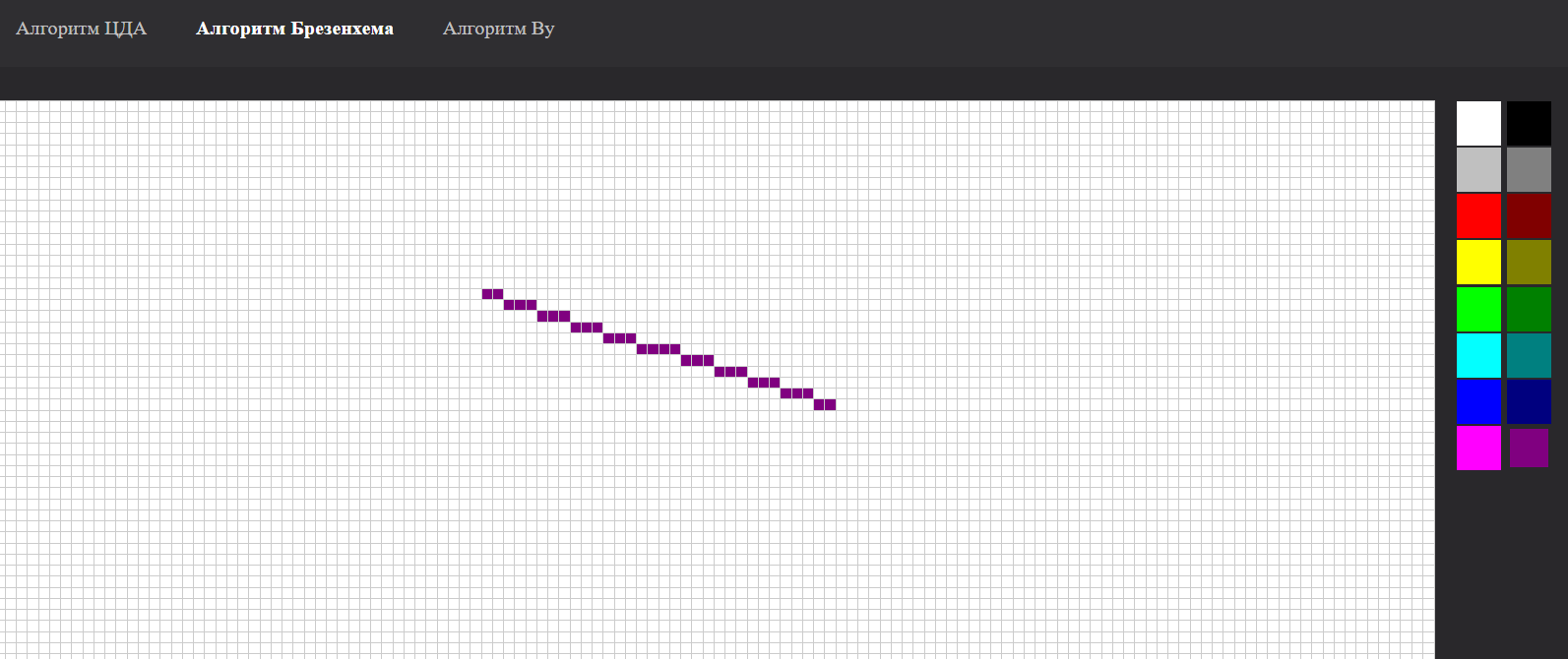


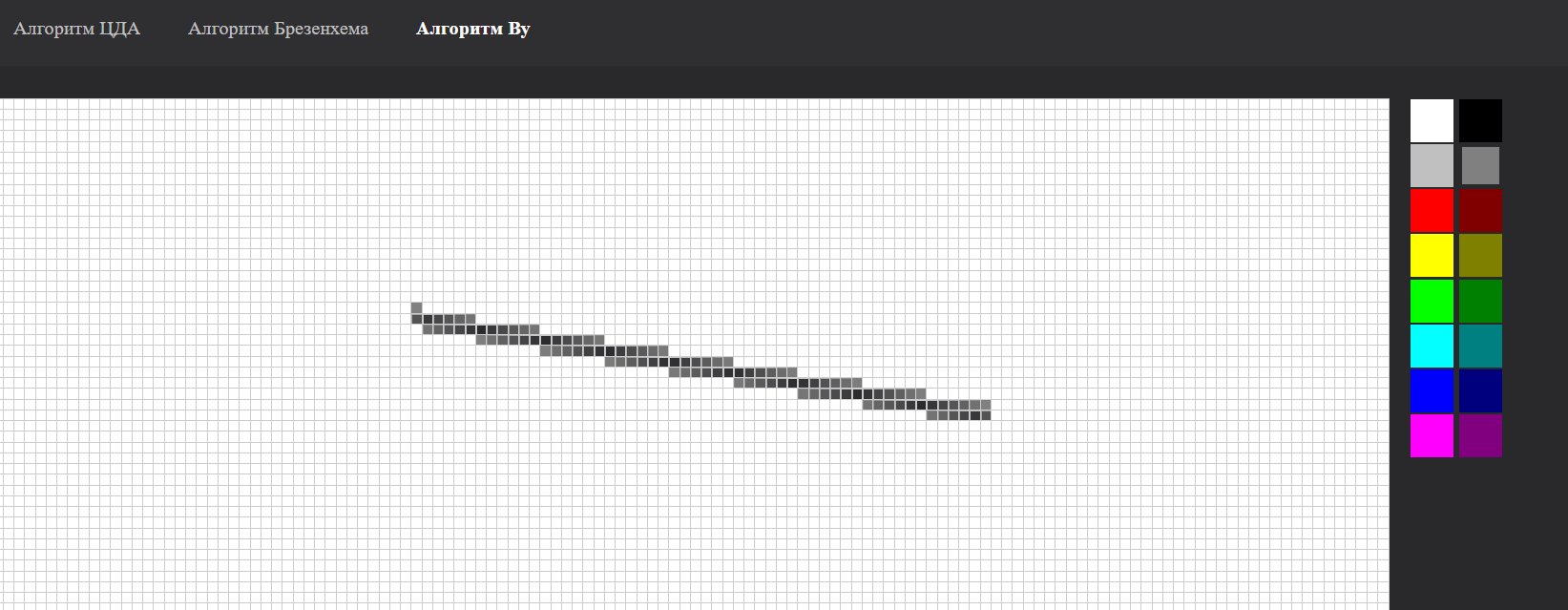
Рисунок 3.2 - Пример работы программы с использованиям алгоритма Брезенхэма

**3.3. Алгоритм Ву**

Алгоритм Ву — алгоритм разложения отрезка в растр со сглаживанием.

Алгоритм сочетает высококачественное устранение ступенчатости и скорость, близкую к скорости алгоритма Брезенхема без сглаживания.

1. Изначально определяется, является ли отрезок горизонталью, вертикалью или диагональю, т.к. для этих видов отрезков сглаживание не применяется. Координаты пикселей заносятся в массив connected\_dots и отрисовываются методом drawPixel.
2. Аналогично алгоритму Брезенхема определяется направление движения. Однако, если dx < dy, значения x и y в координатах меняются местами. Если перемещение происходит по Х влево, координаты меняются местами. Это не сказывается на результате, так как при отрисовке они меняются местами обратно.
3. Для каждой пары соседних пикселей при перемещении определяются интенсивности цвета. Для каждой пары интенсивность в сумме равна 1.
4. Каждый пиксель отрисовывается вложенной функцией plot с учетом вычисленной интенсивности с помощью функции rgba.



**Вывод:**

В ходе лабораторной работы был реализован графический интерфейс для построения отрезков используя алгоритмы ЦДА, Брезенхема и Ву. Все алгоритмыпостроения отрезков соответствуют общим требованиям к алгоритмам рисования отрезков:

1. Отрезки выглядят прямыми, начинаются и заканчиваются в заданных точках;
2. Яркость постоянная и не зависит от длины и наклона;
3. Алгоритмы соответствуют требованиям о быстродействии.