

Отчет для лабораторной работы № 3

Алгоритмы растеризации:

1. Пошаговый алгоритм

Описание: Алгоритм строит отрезок между двумя точками, вычисляя координаты последовательно вдоль одной из осей (X или Y), в зависимости от угла наклона отрезка. Если угол наклона меньше 45° , итерация выполняется по оси X, иначе — по оси Y. На каждом шаге вычисляется соответствующая координата и округляется до ближайшего целого

Реализация:

Вычисляются разности: $dx = x_2 - x_1$, $dy = y_2 - y_1$

Если $|dx| \geq |dy|$, итерация идет по X, иначе — по Y

Коэффициент наклона $k = dy/dx$ (или dx/dy для Y)

На каждом шаге координата округляется до целого

2. Алгоритм ЦДА (цифровой дифференциальный анализатор)

Описание: Алгоритм использует вещественные приращения для плавного построения отрезка. Количество шагов равно максимальной разности по осям. На каждом шаге координаты увеличиваются на фиксированные приращения и округляются

Реализация:

Вычисляются dx , dy

Число шагов: $steps = \max(|dx|, |dy|)$

Приращения: $xIncrement = dx/steps$, $yIncrement = dy/steps$ (из кода)

Начальная точка — (x_1, y_1) . На каждом шаге прибавляются приращения, результат округляется

3. Алгоритм Брезенхема для отрезка

Описание: Алгоритм использует целочисленные вычисления для минимизации ошибки. На каждом шаге выбирается следующая точка на основе значения ошибки, которая корректируется в зависимости от наклона отрезка

Реализация:

Вычисляются $dx = |x_2 - x_1|$, $dy = |y_2 - y_1|$

Определяются знаки шагов: sx , sy

Ошибка: $err = dx - dy$

На каждом шаге проверяется ошибка:

- Если $2*err > -dy$, то X изменяется на sx, ошибка уменьшается на dy
- Если $2*err < dx$, то Y изменяется на sy, ошибка увеличивается на dx

4. Алгоритм Брезенхема для окружности

Описание: Алгоритм строит окружность, используя симметрию и целочисленные вычисления. Начинается с точки (0, R), и на каждом шаге вычисляется следующая точка с учетом ошибки

Реализация:

Начальные значения: $x = 0$, $y = R$, $d = 3 - 2*R$

Пока $y \geq x$:

Если $d > 0$, то y уменьшается, d корректируется

Иначе d корректируется без изменения Y

Добавляются 8 симметричных точек относительно центра

Дополнительно для них всех: **измерение времени**

- Время выполнения алгоритма измеряется с помощью “System.nanoTime()” до и после вычисления точек
- Результат выводится в наносекундах