Машинная графика
Computer Graphics

Лекция 3.

«Алгоритмы Брезенхема»

План лекции

- Алгоритм Брезенхема для построения отрезка
- Целочисленный алгоритм Брезенхема
- Общий алгоритм Брезенхема
- Знакомство с OpenGL

Алгоритм Брезенхема для построения отрезка

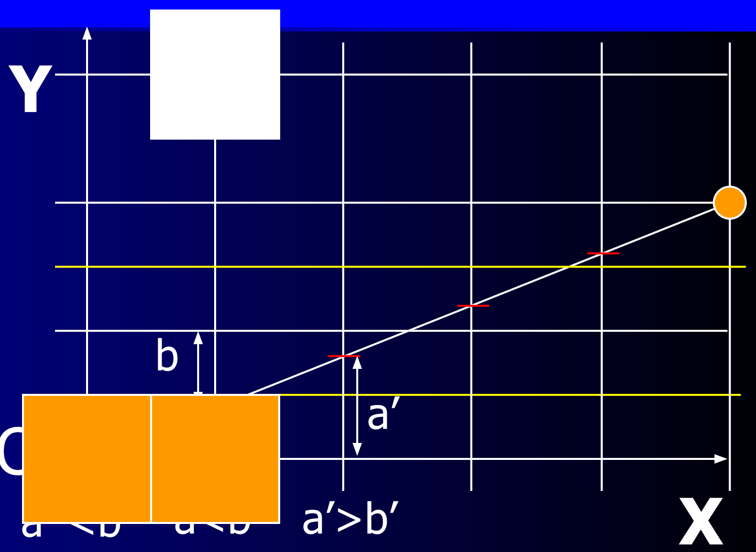
В 1965 г. Джек Брезенхем (Jack E. Bresenham) предложил алгоритм генерации отрезка, который был эффективнее алгоритма ЦДА.

Хотя алгоритм Брезенхема был первоначально разработан для цифровых графопостроителей, однако он в равной степени подходит и для использования растровыми устройствами с ЭЛТ.

Алгоритм выбирает оптимальные растровые координаты для представления отрезка. В процессе работы одна из координат — либо x, либо y (в зависимости от углового коэффициента) — изменяется на единицу, другая – либо остаётся в своём прежнем значении, либо так же изменяется на единицу. Выбор между этими двумя альтернативами зависит от того, какая из них обеспечивает меньшую погрешность.

Алгоритм Брезенхема для построения отрезка

Алгоритм построен таким образом, чтобы на каждом шаге оценивался лишь знак (+) или (-) определённой величины, называемой ошибкой. Под ошибкой подразумевается расстояние между действительной ординатой отрезка на текущем шаге и ближайшей точкой растра (т.е. центром!!! пикселя).



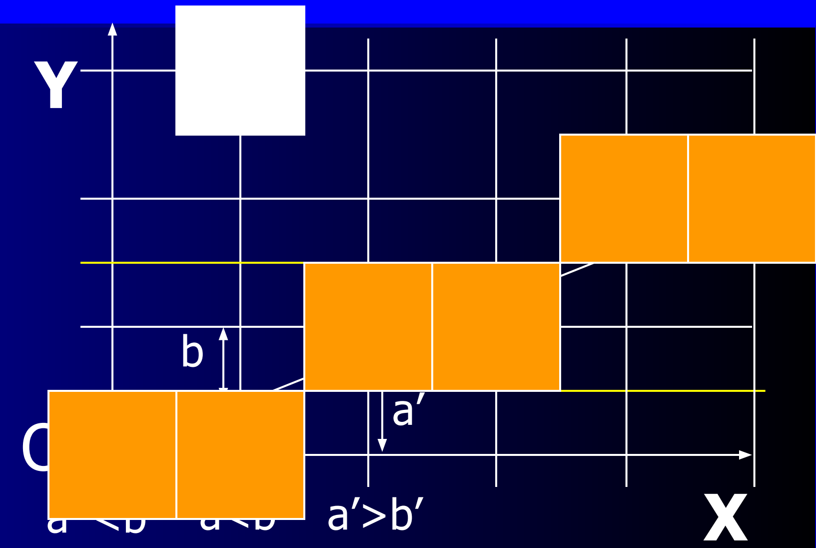
Рассмотрим канонический (1-й октант) случай генерации отрезка:

0-й шаг: Значение ошибки t_0 в начальной точке отрезка (0,0) принимается равным $-½$

1-й шаг: Для последующей точки вычисляется $t_1=t_0+k$, где k-угловой коэффициент $k= (y_к-y_0)/(x_к-x_0)$. Анализируется знак ошибки: если $t<0$, то лучшей аппроксимацией отрезка будет пиксель с коорд-ми (x_0+1, y_0), иначе если $t\geq0$, то - (x_0+1, y_0+1).

Алгоритм Брезенхема для построения отрезка

Прежде (!) чем приступить к след. шагу, значение ошибки следует скорректировать. Если $t \geq 0$, то из неё необходимо вычесть 1 : $t'_1 = t_1 - 1$. В противном случае ошибка не изменяется.



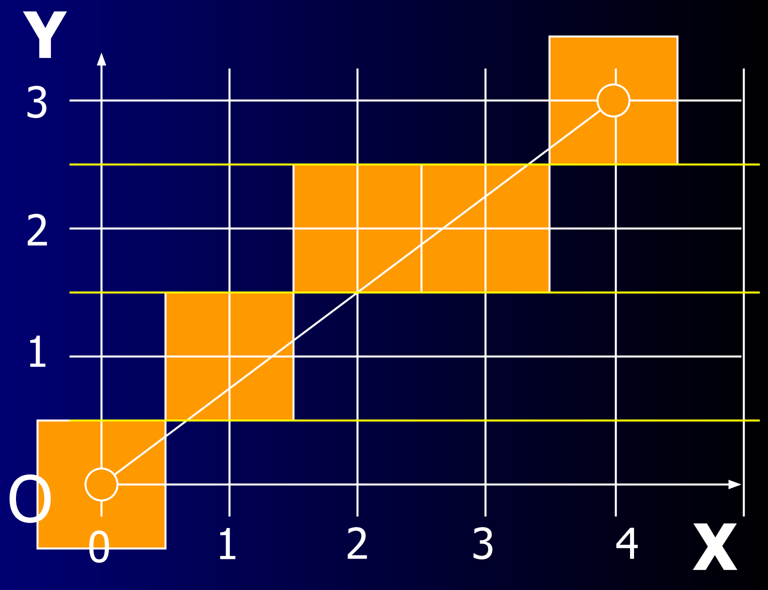
2-й и все последующие шаги: аналогичны шагу 1 - $t_i = t_{i-1} + k$,

если $t < 0$, то лучшей аппроксимацией отрезка будет пиксель с координатами $(x_i+1,\ y_0)$, иначе если $t \geq 0$, то - $(x_i+1,\ y_i+1)$.

Алгоритм Брезенхема для построения отрезка. Пример

Требуется провести линию из [0,0] в [4,3].

$x_0=0$, $y_0=0$, $k=(3-0)/(4-0)=\frac{3}{4}$, $t_0=-\frac{1}{2}$



1-й шаг: $t_1=t_0+k=-\frac{1}{2}+\frac{3}{4}=\frac{1}{4}$,
т.к. $t \geq 0$, то $x_1=x_0+1=1$, $y_1=y_0+1=1$,
Коррекция ошибки: т.к. $t \geq 0$, то
$t'_1=t_1-1=\frac{1}{4}-1=(-\frac{3}{4})$

2-й шаг: $t_2=t'_1+k=-\frac{3}{4}+\frac{3}{4}=0$,
т.к. $t \geq 0$, то $x_2=x_1+1=2$, $y_2=y_1+1=2$,
Коррекция ошибки: т.к. $t \geq 0$, то
$t'_2=t_2-1=0-1=-1$

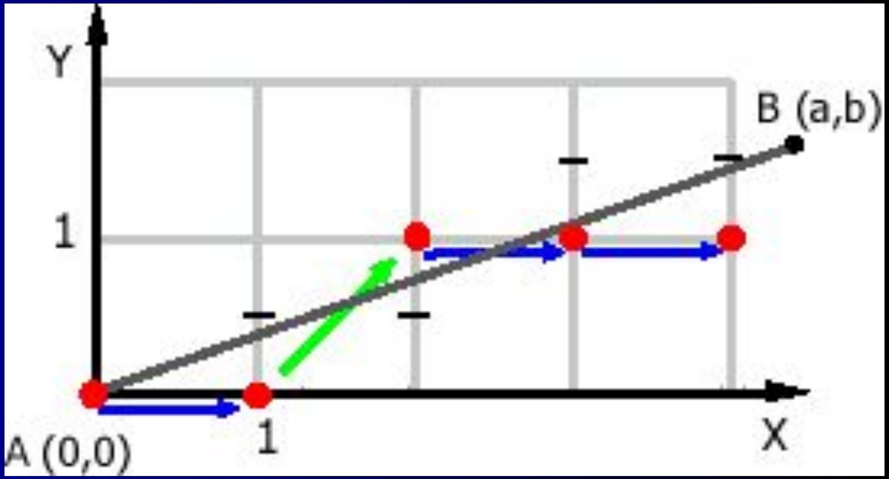
3-й шаг: $t_3=t'_2+k=-1+\frac{3}{4}=-\frac{1}{4}$,
т.к. $t < 0$, то $x_3=x_2+1=3$, $y_3=y_2=2$,
Коррекция ошибки: т.к. $t < 0$, то
коррекция не производится $t'_3=t_3$

4-й шаг: $t_4=t'_3+k=-\frac{1}{4}+\frac{3}{4}=\frac{1}{2}$
т.к. $t \geq 0$, то $x_4=x_3+1=4$,
$y_4=y_3+1=3$
Так как $x_4 \geq x_к$, то отрезок построен. Выход.

Сравнение с алгоритмом цифрового дифференциального анализатора

```
while(x ≤ a)
{
     plot(x,y);

     if (e>=1/2)
      {  // d : диагональное
         //      смещение
            x++; y++;
            e+=∆e-1 ;
         // коррекция ошибки т.к. произошло смещение по y на 1 вверх
       }
      else
       { // s : горизонтальное смещение
      x++;
            e+=∆e;
        }
}
```



<u>Недостаток алгоритма:</u>
Работает с числами с плавающей точкой.

Целочисленный алгоритм Брезенхема

Предыдущий вариант алгоритма требует использования вещественной арифметики, однако можно обойтись только целочисленной. Для этого Брезенхем умножил все переменные алгоритма на $2dx$, где $dx = x_к - x_0$. Т.е. алгоритм не изменился за исключением нескольких замен.

Модифицированное значение ошибки: $t\hat{} = t*2dx$,

соответственно нулевое значение ошибки: $t\hat{}_0 = -\frac{1}{2}*2dx = -dx$,

модифицированный угловой коэффициент:

$$k\hat{} = k*2dx = (dy/dx)*2dx = 2dy,$$

вычисление модифицированной ошибки:

$$t\hat{}_i = t\hat{}_{i-1} + k\hat{} = t\hat{}_{i-1} + 2dy,$$

коррекция модифицированной ошибки так же изменилась на 2dx:

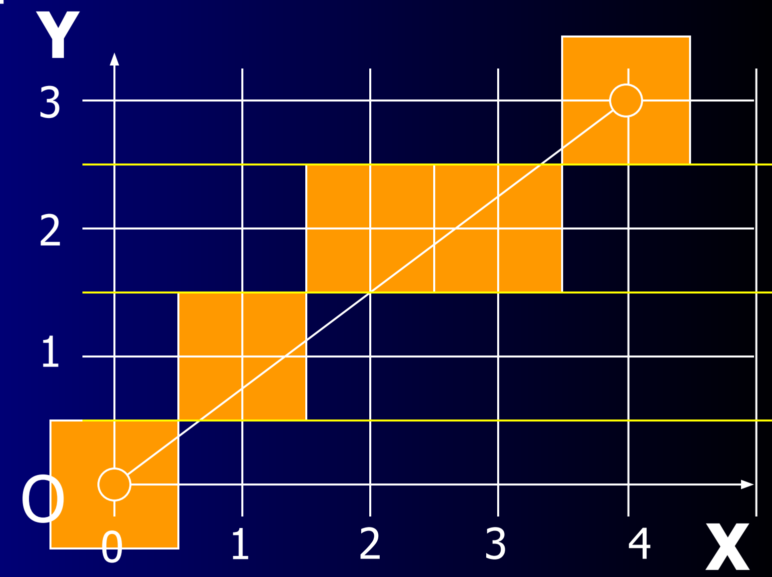
$$t\hat{}'_i = t\hat{}_i - 2dx$$

Целочисленный алгоритм Брезенхема. Пример

Требуется провести линию из [0,0] в [4,3].

$x_0=0$, $y_0=0$, $dx=4-0=4$, $dy=3-0=3$,

$\hat{t}_0 = -dx=-4$, $\hat{k}=2dy=6$



1-й шаг: $\hat{t}_1=\hat{t}_0+\hat{k} =- 4+6=2$,
т.к. $\hat{t}\geq0$, то $x_1=x_0+1=1$, $y_1=y_0+1=1$,
Коррекция ошибки: т.к. $\hat{t}\geq 0$, то
$\hat{t}'_1 = \hat{t}_1-2dx = 2 - 8 = - 6$

2-й шаг: $\hat{t}_2=\hat{t}'_1+\hat{k} = - 6+6 = 0$,
т.к. $\hat{t} \geq 0$, то $x_2=x_1+1=2$,
$y_2=y_1+1=2$,
Коррекция ошибки: т.к. $\hat{t} \geq 0$, то
$\hat{t}'_2 = \hat{t}_2-2dx = 0 - 8 = - 8$

3-й шаг: $\hat{t}_3=\hat{t}'_2+\hat{k} = - 8+6 = -2$,
т.к. $\hat{t} < 0$, то $x_3=x_2+1=3$, $y_3=y_2=2$,
Коррекция ошибки: т.к. $\hat{t} < 0$, то
коррекция не производится $\hat{t}'_3 = \hat{t}_3$

4-й шаг: $\hat{t}_4=\hat{t}'_3+\hat{k} =$
$= -2+6=4$, т.к. $\hat{t} \geq 0$, то
$x_4=x_3+1=4$, $y_4=y_3+1=3$
Так как $x_4 \geq x_к$, то отрезок построен. Выход.

Общий алгоритм Брезенхема

Чтобы реализация целочисленного алгоритма Брезенхема была полной, необходимо уметь отображать отрезки во всех октантах. Модификацию легко сделать, учитывая в алгоритме номер квадранта, в котором лежит отрезок, и его угловой коэффициент.

Когда абсолютная величина углового коэффициента больше 1, постоянно изменятся на единицу (+ или -) должна координата y, а критерий ошибки Брезенхема используется для принятия решения об изменении или не изменении на очередном шаге величины x. Знак суммируемой с x и y константой (+ 1 или - 1) зависит от квадранта, как показано на следующем рисунке.

Общий алгоритм Брезенхема

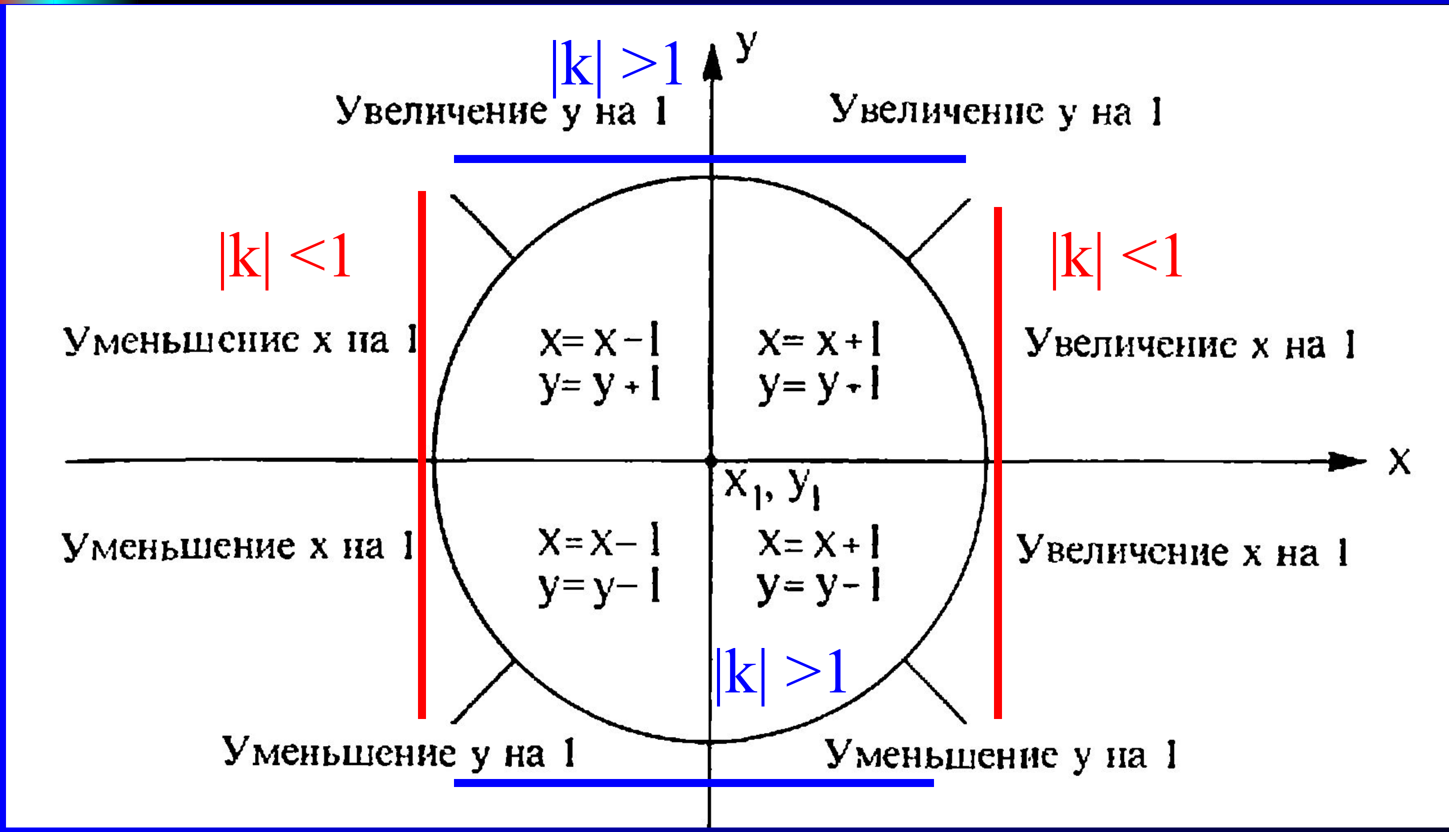


Рис. Разбор случаев для обобщённого алгоритма Брезенхема

Общий алгоритм Брезенхема

Общий алгоритм может быть оформлен в следующем виде:

Предполагается, что концы отрезка (x1, y1) и (x2, y2) не совпадают. Все переменные считаются целыми.

Функция Sign возвращает -1, 0, 1 для отрицательного, нулевого и положительного аргумента, соответственно.

Инициализация переменных:

x = x1
y = y1
dx = abs (x2 – x1)
dy = abs (y2 – y1)
s1 = Sign (x2 – x1)
s2 = Sign (y2 – y1)

Общий алгоритм Брезенхема

В зависимости от углового коэффициента наклона отрезка необходимо произвести обмен значений dx и dy:

```
if (dy > dx)
  { temp = dx;
    dx = dy;
    dy = temp;
    ChangeFlag = true;
  }
else
    ChangeFlag = false;
```

Инициализация t: $t = 2*dy - dx$

Общий алгоритм Брезенхема

Основной цикл алгоритма (Д.Роджерс с.61):

```
for (i=1;i++;i<dx)
{
    plot(x,y);
    while(t>=0)
    {
    if (ChangeFlag)
        x+=s1;     // увеличение либо уменьшение на 1
    else
        y+=s2;     // увеличение либо уменьшение на 1
    t  =  t – 2*dx;      // коррекция ошибки
     }
     if (ChangeFlag)
        y+=s2;
     else
        x+=s1;
      t  =  t + 2*dy;   // вычисление ошибки для следующего шага
}
```

Общий алгоритм Брезенхема. Пример

Рассмотрим отрезок из точки (0, 0) в точку (-8,- 4).

Инициализация:

x=0
y=0

dx=abs (y2 – y1) = 8
dy=abs (y2 – y1) = 4

S1= -1
S2= -1

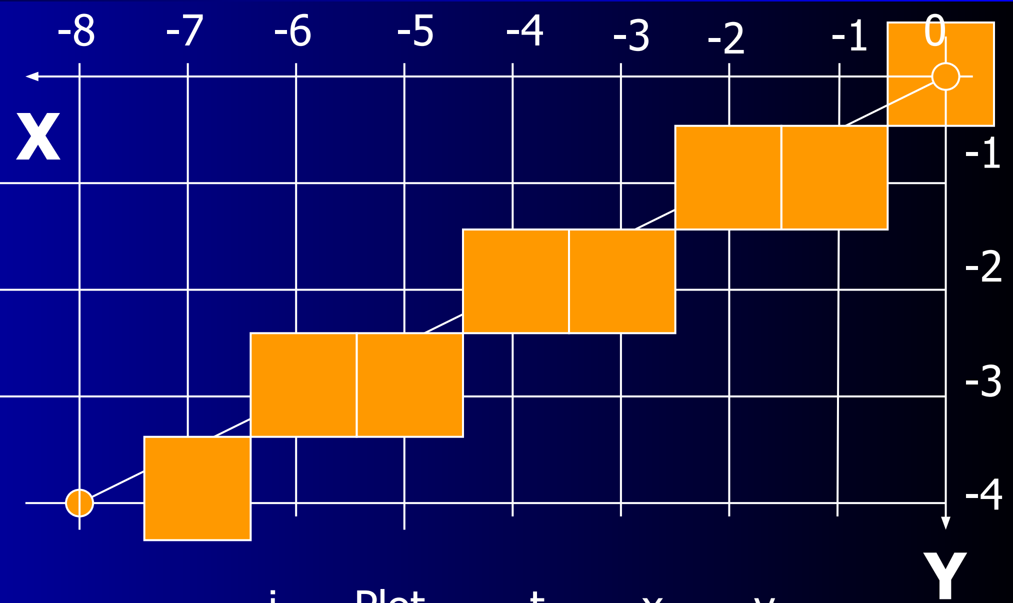
ChangeFlag = 0

t = 2*dy - dx = 0

Общий алгоритм Брезенхема. Пример

Пошаговое выполнение основного цикла:

i	Plot	t	x	y
		0	0	0
1	(0,0)			
		-16	0	-1
		- 8	-1	-1
2	(-1,-1)			
		0	-2	-1
3	(-2,-1)			
		-16	-2	-2
		- 8	-3	-2
4	(-3,-2)			
		-0	-4	-2
5	(-4,-2)			
		-16	-4	-3
		- 8	-5	-3
6	(-5,-3)			
		- 0	-6	-3



i	Plot	t	x	y
7	(-6,-3)			
		-16	-6	-4
		- 8	-7	-4
8	(-7,-4)			
		- 0	-8	-4

The End !!!