26.02.14-----------------------------------------------------------------------

1. *Нахождение двумерных перспективных проекций объектов сцены на картинную плоскость.*
2. *Исключение невидимых объектов сцены при заданном положении наблюдателя. Закрашивание и затенение видимых участков сцены.*
3. *Вывод полученного полутонового изображения на экран растрового дисплея.*

Алгоритмы нижнего уровня -> Алгоритмы среднего уровня (построение плоских изображений) -> Алгоритмы верхнего уровня (удаление невидимого, построение реалистичного изображения)

**Растровая графика**

**Растровая развертка отрезка (разложение в растр)**

Процесс определения пикселей, наилучшим образом аппроксимирующих заданный отрезок. Простейшие случаи – горизонтальный, вертикальный и под 45°. В большинстве случаев проявляется лестничный (ступенчатый) эффект.

*Общие требования:*

1. Отрезок должен выглядеть как отрезок прямой, начинаться и заканчиваться в заданных точках
2. Интенсивность (яркость) вдоль отрезка должна быть постоянной. Отрезки, имеющие разные углы наклона, должны быть одной интенсивности. Восприятие человека зависит не только от интенсивности свечения объекта, но и от расстояния между светящимися объектами //чтобы удовлетворить этому требованию, надо высвечивать точки с переменной интенсивностью от расстояния – потребует дополнительных вычислений, без особой нужды не используется
3. Алгоритмы (особенно нижнего уровня) должны работать быстро

Все алгоритмы имеют пошаговый характер – на очередном шаге высвечиваем пиксель, и производим вычисления используемые в следующем шаге.

**Алгоритм цифрового дифференциалього анализатора**

Если прямая задана каноническим уравнением , то производная . Если заданы начальная и конечная точки х1у1 и х2у2, то . В данном случае дельта-х=1, а дельта-у – высчитывается и округляется. Однако дельта-х=1 следует выбирать не всегда – вначале нужно ответить на вопрос, какую. Если угол наклона <45, то |sX|=1, если > то |sY|=1

1. Начало
2. Ввод хн, ун, хк, ук
3. Проверка отрезка на вырожденность: если вырожденный то высветить точку и переход на конец
4. Если |dx|>|dy|, то l=|dx| иначе l=|dy|
5. Построение отрезка (цикл по i=1 до l+1)
   1. Высвечивание точки E(xt),E(yt)
   2. Вычисление координат следующей точки
6. Конец

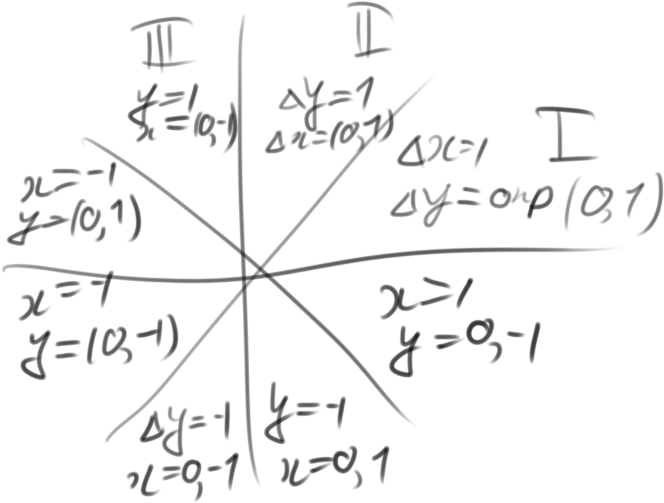
Недостатки – работает с целочисленной арифметикой (координаты текущей точки). Работает медленнее за счет операции округления.

**Алгоритмы Брезенхема**

Ошибка еi – величина изменяющаяся на каждом шаге , расстояние между точкой самого отрезка и точкой, аппроксимирующей его на очередном шаге. Если <0.5, то ордината пикселя не меняется, Если >0.5, то выбирается верхний пиксель.. Начальное значение ошибки e=m, где м – тангенс угла наклона, .

На i+1 шаге . В таком случае вычисляем предварительное . Если на очередном шаге мы корректируем значение у, то мы должны также скорректировать и ошибку.

Допускается начальное e=m-0.5 и дальше сравнивать ошибку с нулем, а не с 0.5



Алгоритм Брезенхема:

1. Начало
2. Ввод хн, ун, хк, ук
3. Проверка отрезка на вырожденность: если вырожденный то высветить точку хн,ун и переход на конец
4. Sx=sign(dx), sy=sign(dy)
5. Dx=|dx|, dy=|dy|
6. Если dx>dy то обмен=0, иначе обмен=1 {t=dy; dy=dx; dx=t}
7. .
8. Цикл построения отрезка (по i=1 to dx+1)
9. Конец