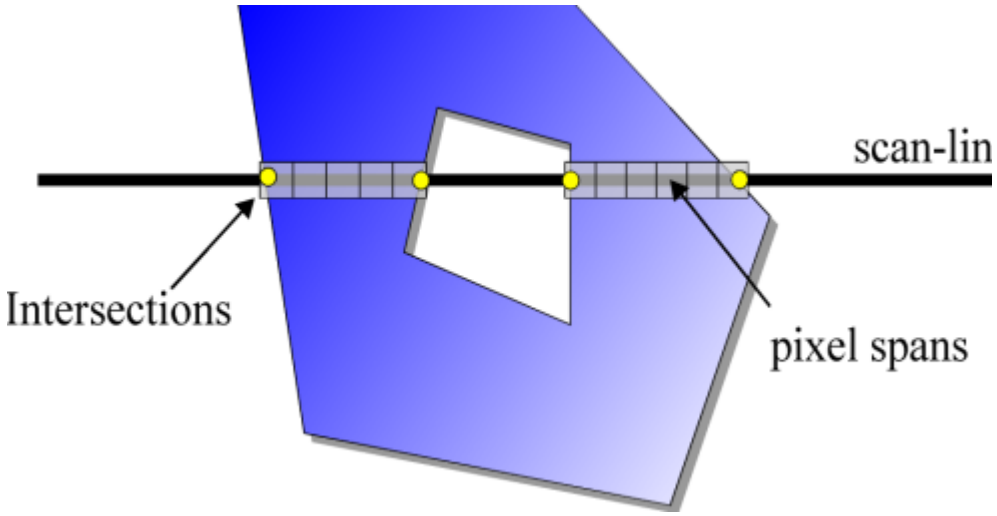
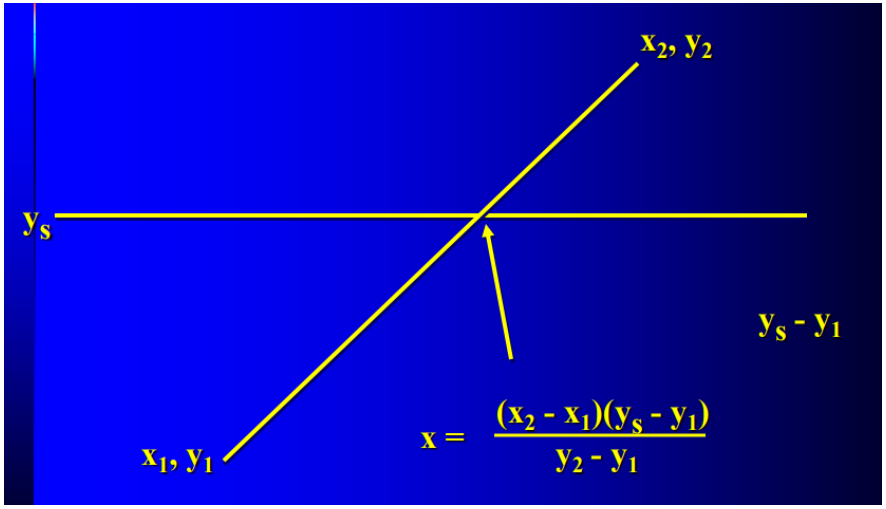
Растровая развёртка многоугольника – классический алгоритм, заполнения основан на предположении, что любое горизонтальное сечение контура многоугольника состоит из четного числа точек.



Последовательность действий

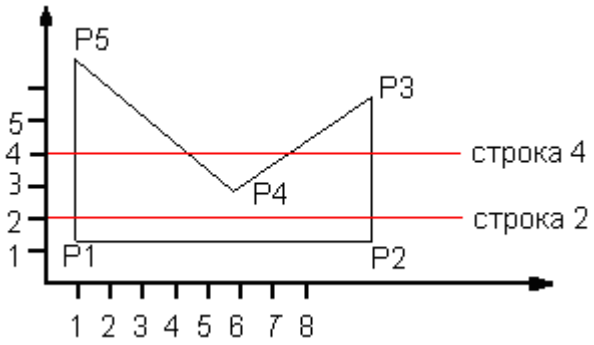
- Для каждой сканирующей строки определяются точки пересечения со сторонами полигона
- Вычисляются промежутки строк, соответствующих внутренним частям многоугольника
- Промежутки закрашиваются требуемым цветом.



Вычисление точек пересечения

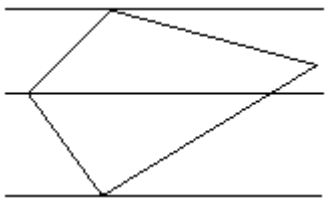
Алгоритмы построчного сканирования

Можно разработать более эффективный способ заполнения, если воспользоваться тем фактом, что соседние пикселы имеют одинаковые характеристики (кроме пикселов граничных ребер). Это свойство называется пространственной когерентностью. Характеристики пикселов на данной строке изменяются только там, где ребро многоугольника пересекает строку. Эти пересечения делят строку на части.



Строка 2 пересекает многоугольник в двух точках. Получаем три области: две вне, одна внутри. Строка 4 делится на пять областей.

Точки пересечения необходимо отсортировать в возрастающей последовательности.



Дополнительная трудность возникает при пересечении сканирующей строки и многоугольника точно по вершине. Правильный результат можно получить, учитывая точку пересечения в вершине два раза, если она является точкой локального минимума или максимума, или учитывая ее один раз в противном случае.

Простой алгоритм с упорядоченным списком ребер

- Определить для каждого ребра многоугольника точки пересечения со сканирующими строками, проведенными через середину интервалов (можно использовать алгоритм Брезенхема или ЦДА). Горизонтальные ребра игнорируются. Занести каждое значение (X,Y+1/2) в список.

- Отсортировать список по строкам и по возрастанию в строке. Преобразовать эти данные в растровую форму:

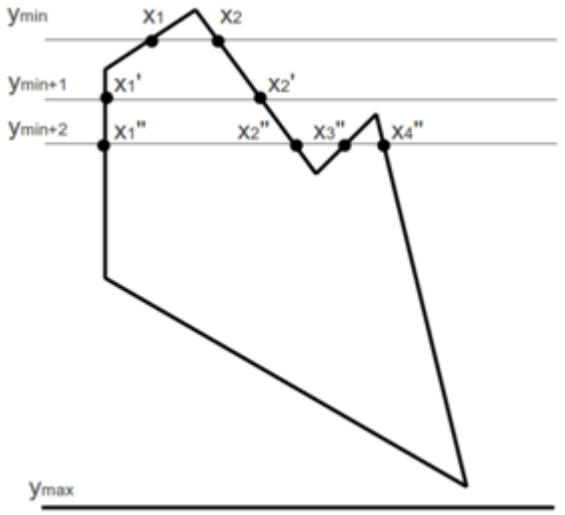
  - Выделить из списка пары элементов (X_1,Y_1) и (X_2,Y_2). Структура списка гарантирует, что $Y=Y_1=Y_2$ и $X_1<=X_2$.
  - Активировать на сканирующей строке пикселы для целых значений X таких, что $X_1<=X+1/2<=X_2$.

Алгоритмы со списком рёберных точек

Будем предполагать, что каждое ребро многоугольника задаётся координатами его концов x1,y1 и x2,y2

Будем обрабатывать только те рёбра, которые пересекаются с текущей линией развёртки.

Найдем y_min и y_max



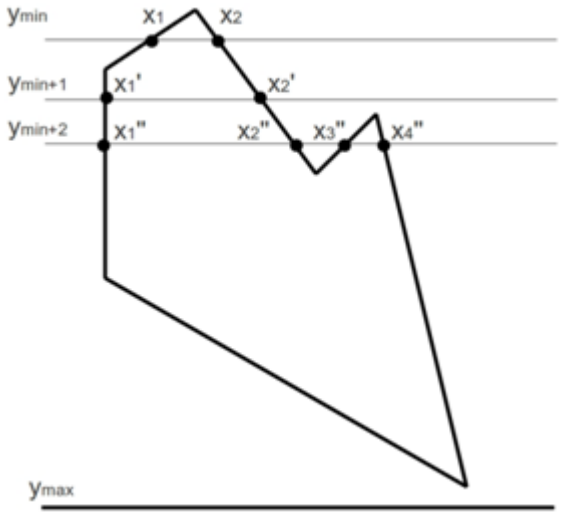
Далее от y_min до y_max находим точки пересечения с ребрами x1 и x2.

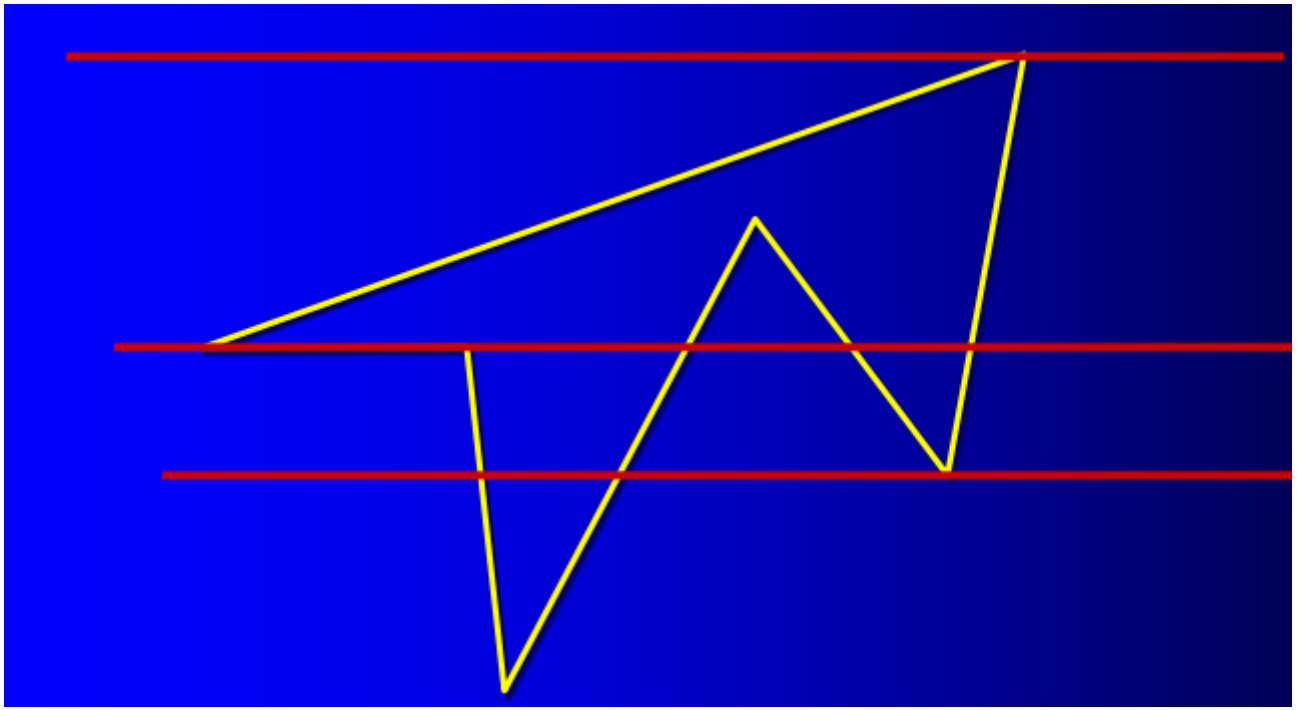
Если точек более 2 или менее 2 рассматриваем этот случай

Упорядочиваем по возрастанию.

Заполняем полученный отрезок.

Переходим к следующей строке.





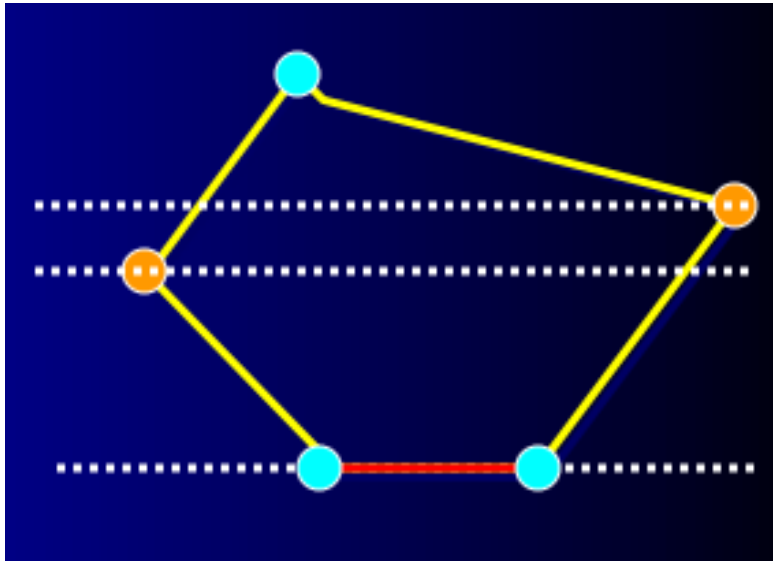
Исключительные случаи:

- отдельно стоящая вершина,
- горизонтальное ребро,
- двойное пересечение

Обработка исключительных случаев

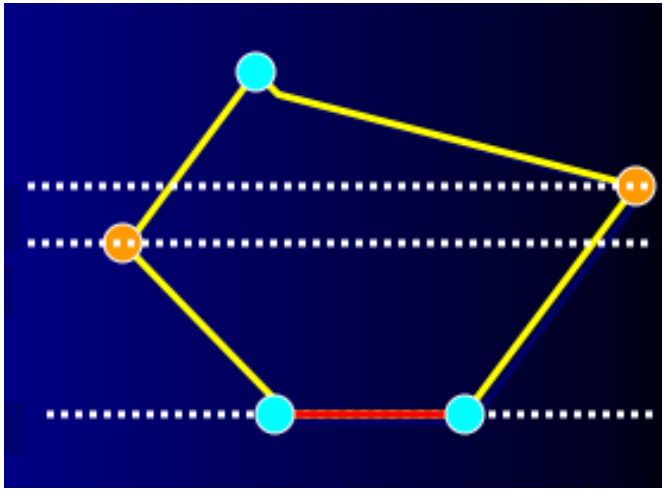
Исключительные случаи:

– горизонтальные рёбра игнорируются,
– отдельно стоящая вершина учитывается два раза при условии, что она
– локальный экстремум, и один раз если она им не является.



Проверка вершины на локальный экстремум

Проверяем оставшиеся концы рёбер, смыкающихся в данной вершине, если у обоих рёбер у координаты концов больше, чем у проверяемой вершины – то вершина лок. минимум, если меньше - то лок. максимум. В оставшемся случае – вершина не является локальным экстремумом.



Алгоритм с упорядоченным списком рёбер

Базируясь на рассмотренных выше принципах были разработаны алгоритмы, называемые алгоритмами с упорядоченными списками рёбер. Они все используют сортировку и их эффективность, как правило, зависит от эффективности метода сортировки.

Простейший вариант алгоритма:

- Для каждого ребра определяются точки пересечения
- Горизонтальные рёбра игнорируются
- Каждое пересечение заносится в список. Список сортируется по строкам (т.е. по y координате), затем в пределах каждой строки – сортировка по x координате.
- Из списка выбираются пары элементов, образующих закрашиваемый интервал.

Недостаток: большой список точек для сортировки.

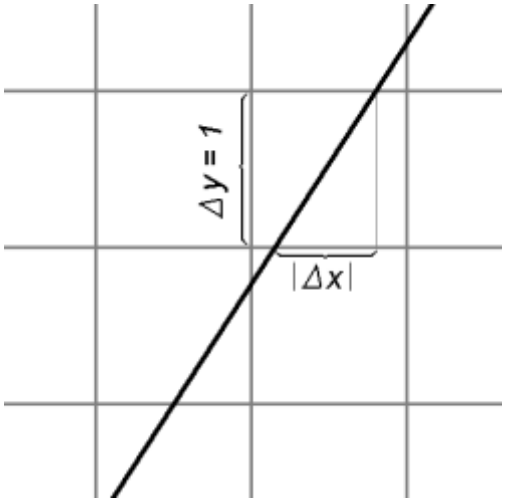
Разделим один список на несколько – для каждой строки будет свой отдельный. Тогда алгоритм будет состоять из трёх шагов.

Шаг I.

Растеризуем все негоризонтальные ребра многоугольника, помещая полученные точки в списки: каждому значению y = ymin, ymin+1, …, ymax сопоставляется список x-координат всех пикселей из растеризации, находящихся на горизонтали y (здесь ymin и ymax – минимальная и максимальная y-координаты пикселей в растровом изображении многоугольника).

Формально эту процедуру можно записать следующим образом:

Для каждого ребра (x1, y1) – (x2, y2)




```
{
 y=int(y1+1);         //Округление до большего
 dx = (x2 - X1)/(y2 - y1);
 x = X1+dx*(y2 - y1);
while(y <= y2)
{
 PutToList(x, y);   //поместить x в список, соответствующий данному y
 y++;
 x += Δx;
}
}
```

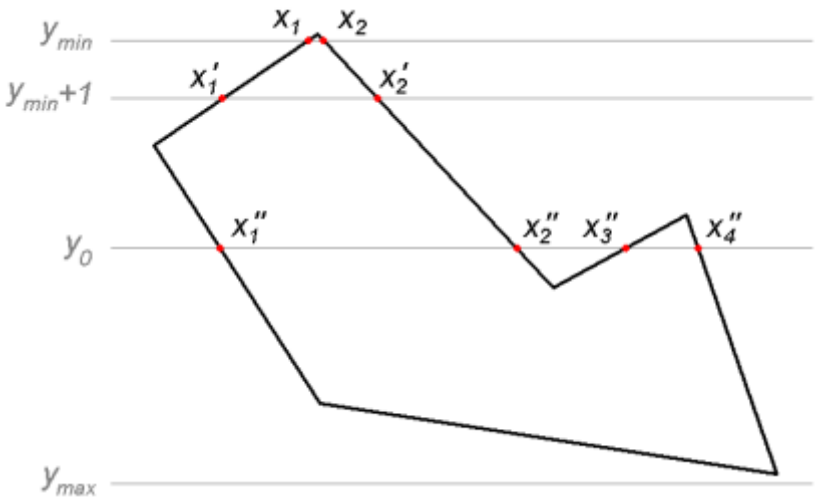
где dx – смещение по x при движении вдоль ребра, соответствующее увеличению y-координаты на 1

Шаг II

Для каждого y упорядочим список по возрастанию x

Шаг III

Для каждого y закрашиваются все промежутки вида



ymin: x1 , x2
ymin+1: x1 ’, x2 ’
… …
y0 : x1 ”, x2 ”, x3 ”, x4 ”
…
ymax: –

Следует отметить, что в каждом списке будет содержаться чётное количество элементов.