Алгоритмы вывода фигур

Существуют различные способы задания области. Одно из главных различий состоит в том, что область может быть определена «пиксельно» или «символически».

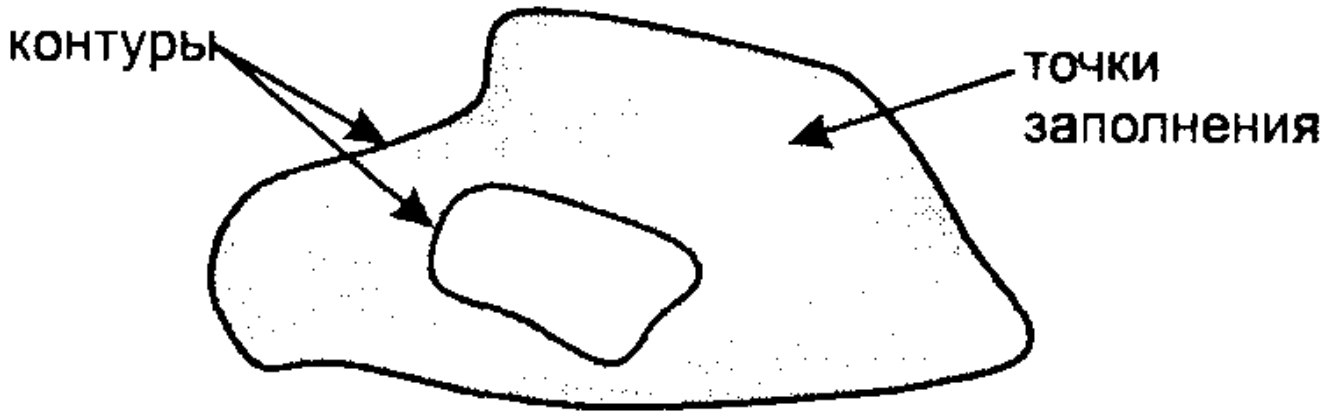
Пиксельно-определенная область характеризуется текущими цветами пикселов в пиксельной карте. Описанием области *R* может служить список всех пикселов, лежащих внутри этой области *R*: (34, 12), (34, 13), (34, 14) и т. д. Область *R* может быть также определена как совокупность всех пикселов, значение которых равно 77 и которые каким-либо образом «связаны» с пикселом (43,129). Понятие такой «связанности» должно быть определено очень тщательно. Для того чтобы увидеть, что представляет собой область *R:* следует изучить ее пиксельную карту и выяснить, какие именно пикселы находятся в *R* в соответствии с этим определением.

Алгоритмы вывода фигур

При символическом определении пикселы не перечисляются, однако описывается некоторое свойство, которым обладают все пикселы области R Такие описания считаются описаниями «более высокого уровня» или более абстрактными, чем прямое перечисление пикселов. Ниже приводятся некоторые возможные способы символического описания областей:

- все пиксели, лежащие внутри окружности радиуса 8 с центром в точке (5,23);
- все пиксели внутри полигона с вершинами (32, 56), (120, 546), (345,129), (80, 87).

Алгоритмы вывода фигур



Графический вывод фигур в этом случае может делится на две задачи: вывод контура и вывод точек заполнения.

Алгоритмы закрашивания областей, заданных цветом границы

Простейший алгоритм закрашивания. Для такого алгоритма закрашивания нужно задавать начальную точку внутри контура с координатами x_0, y_0, от которой будет выполнятся закраска каждого соседнего пикселя рекурсивно.

Алгоритм закрашивания линиями. Алгоритм также рекурсивный, на каждом шаге закрашивания рисуется горизонтальная линия, которая размещается между пикселями контура.

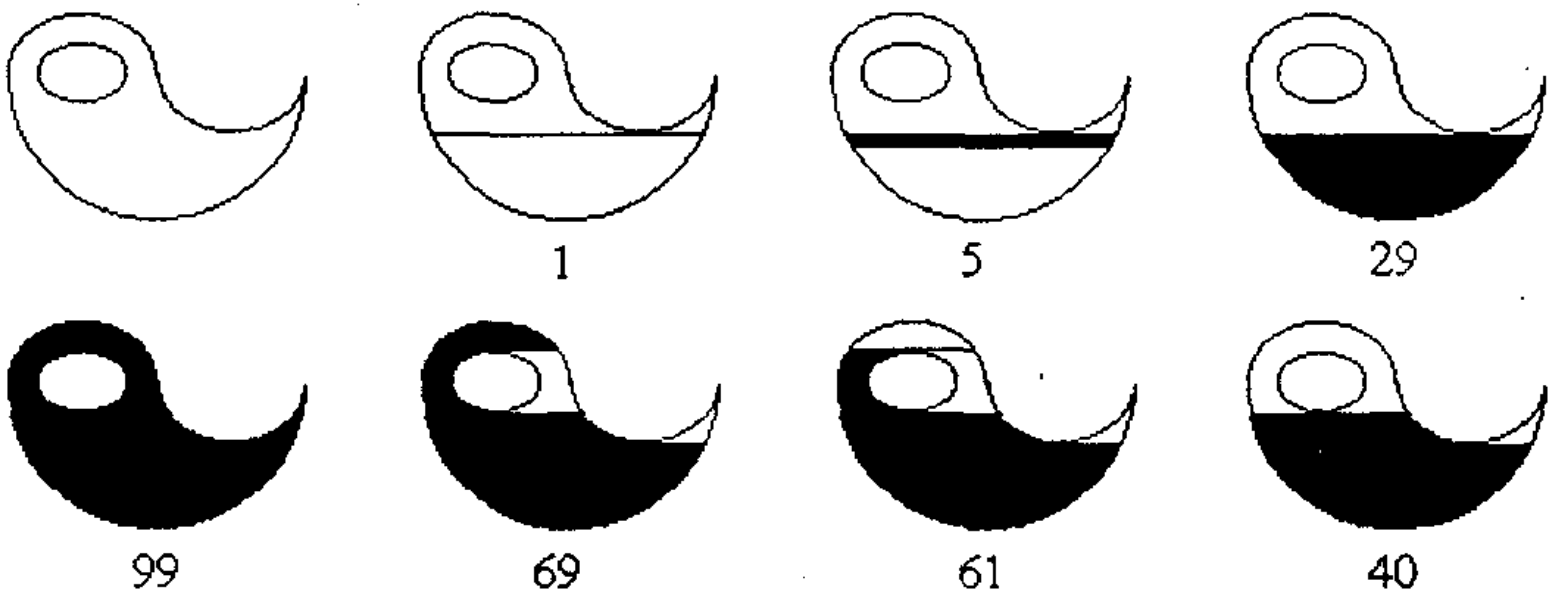
Алгоритмы заполнения, которые используют математическое описание контура. Для генерации точек заполнения не нужны предварительно сформированные границы контура фигуры. Контур может вообще не рисоваться.

Простейший алгоритм закрашивания

```
void PixelFill (int x, int y,unsigned char *border_color,
unsigned char *color) // рекурсивная закраска
{
unsigned char p[4];
int i,k=0,f=0;
glColor4ub(*(color+0),*(color+1),*(color+2),*(color+3));
glReadPixels(x, y, 1, 1, GL_RGBA, GL_UNSIGNED_BYTE, &p[0]);
```

```
for(i=0;i<4;i++)
{k++;
if(p[i]!=*(border_color+i)) break;}
     for(i=0;i<4;i++)
     {f++;
     if(p[i]!=*(color+i)) break;}
       if((k<4)&&(f<4))
          { drawDot(x,y);
            PixelFill (x, y+1 , border_color, color);
            PixelFill (x+1, y , border_color, color);
            PixelFill (x, y-1 , border_color, color);
            PixelFill (x-1, y , border_color, color);  } }
```

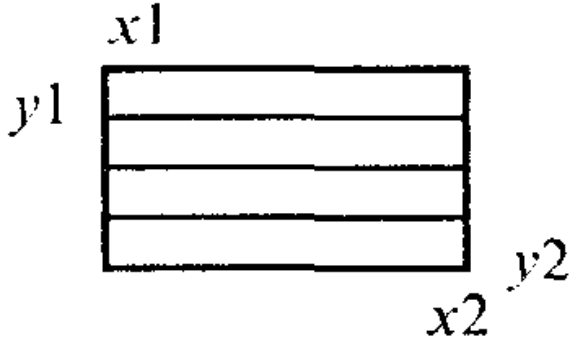
Алгоритм закрашивания линиями



Алгоритмы заполнения, которые используют математическое описание контура

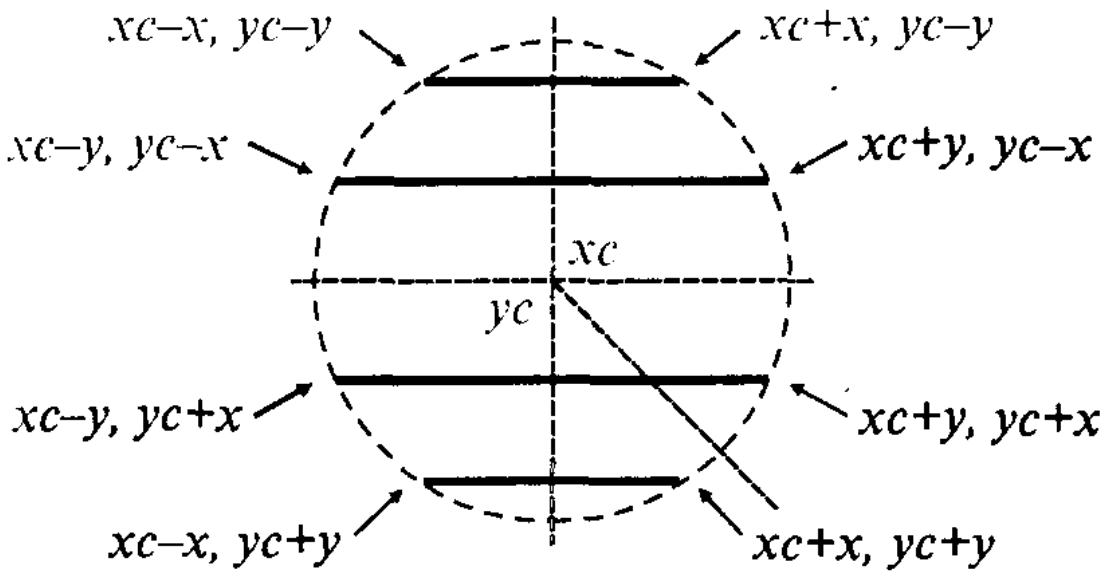
Математическим описанием контура фигуры может служить уравнение y = f(x) для контура чего-либо, например окружности, эллипса или другой кривой. Также выделяют заполнение прямоугольников, заполнение круга, заполнение полигонов.

Заполнение прямоугольников. Прямоугольник может быть задан координатами противоположных углов, например, левого верхнего (x_1, y_1) и правого нижнего (x_2, y_2), тогда алгоритм может состоять в последовательном рисовании горизонтальных линий заданного цвета.



Алгоритмы заполнения, которые используют математическое описание контура

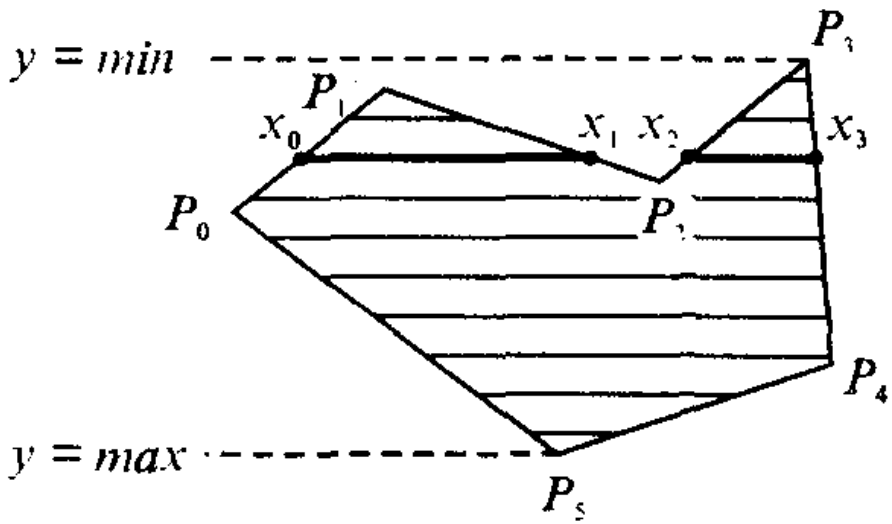
Заполнение круга. Для заполнения круга можно использовать алгоритм вывода контура. В процессе выполнения этого алгоритма последовательно вычисляются координаты пикселей контура в границах одного октанта. Для заполнения следует выводить горизонтали, которые соединяют пары точек на контуре, расположенные симметрично относительно оси y.



Алгоритмы заполнения, которые используют математическое описание контура

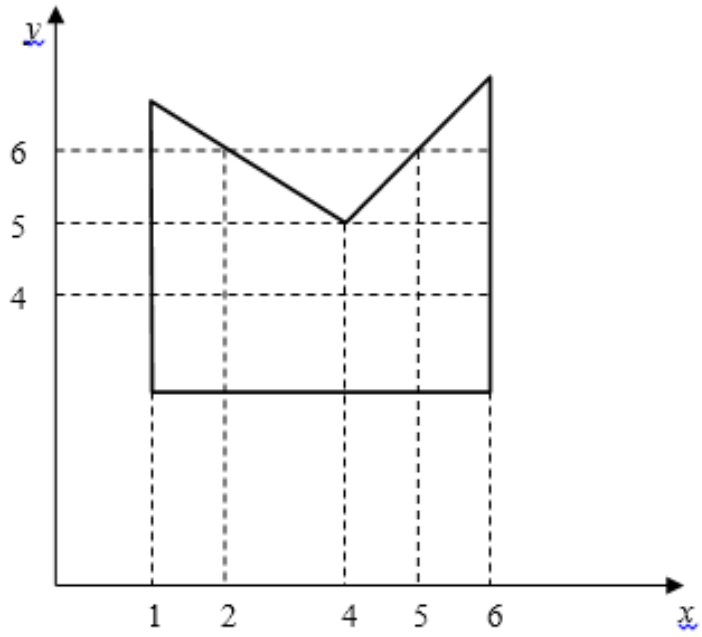
Заполнение полигонов. Контур полигона определяется вершинами, которые соединены отрезками прямых – ребрами.

Основная идея – закрашивание фигуры отрезками прямых линий. Удобней использовать горизонтали. Алгоритм представляет собою цикл вдоль оси y, в ходе этого цикла выполняется поиск точек пересечения линии контура с соответствующими горизонталями.



Метод построчного сканирования

Сканирующие строки обычно изменяются от «верха» многоугольника до его «низа». Находят пересечения сканирующей строки с ребрами многоугольника. Характеристики пикселей изменяются только там, где ребро многоугольника пересекает строку. Точки пересечения делят сканирующую строку на области закрашенных и не закрашенных пикселей.

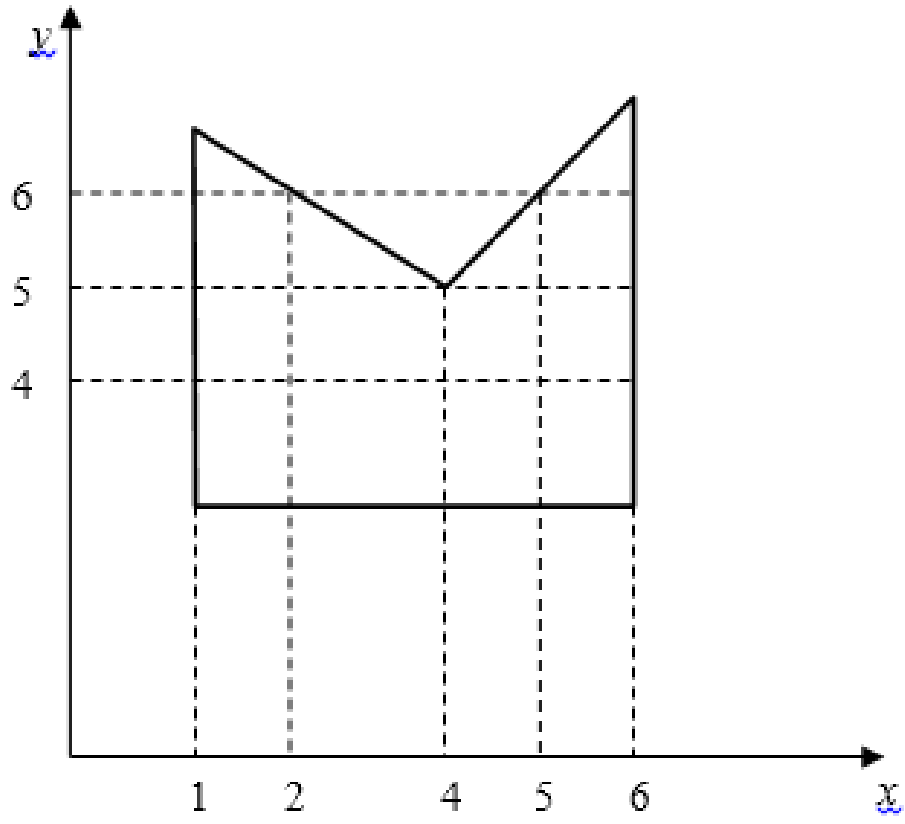


Метод построчного сканирования

1. Простой случай. Например, на рис. сканирующая строка y = 4 пересекает многоугольник при x = 1 и x = 6. Получается три области: x < 1; 1 ≤ x ≤ 6; x > 8. Сканирующая строка y = 6 пересекает многоугольник при x = 1; x = 2; x = 5; x = 6. Получается пять областей: x < 1; 1 ≤ x ≤ 2; 2 < x < 5; 5 ≤ x ≤ 6; x > 6.

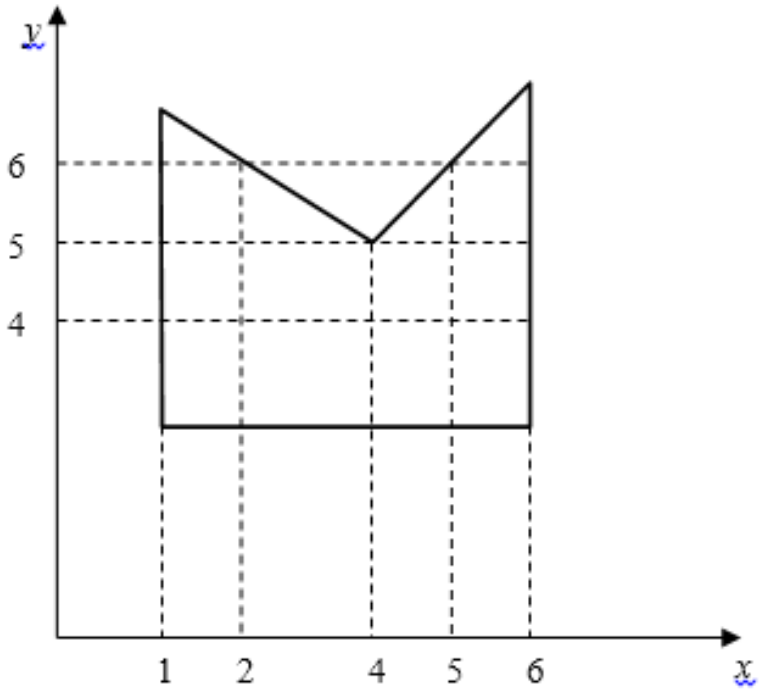
В этом случае x сортируется в порядке возрастания. Далее список иксов рассматривается попарно. Между парами точек пересечения закрашиваются все пикселы.

Для y = 4 закрашиваются пикселы в интервале (1, 6), для y = 6 закрашиваются пикселы в интервалах (1, 2) и (5, 6).



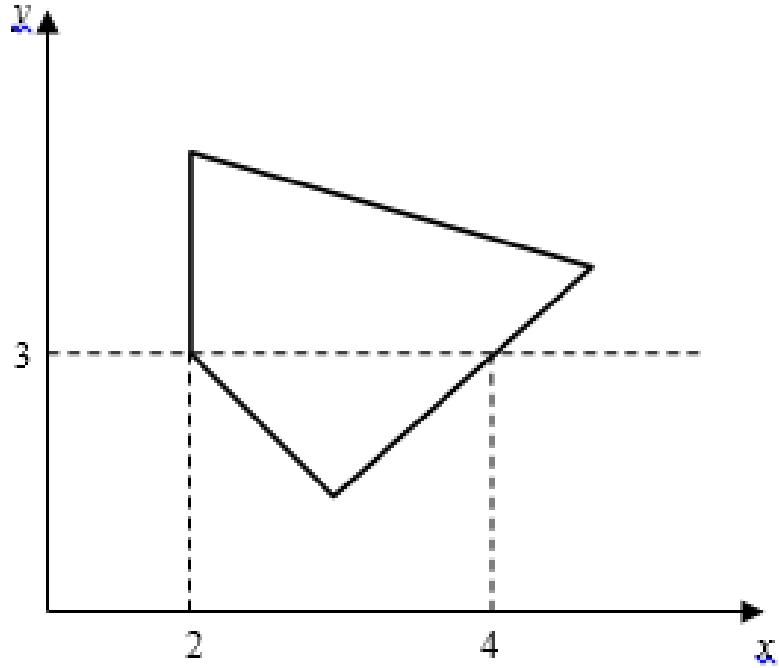
Метод построчного сканирования

2. Сканирующая строка проходит через локальный минимум или максимум (на рис при y = 5). В этом случае учитываются все пересечения вершин сканирующей строкой. На рис. при y = 5 формируется список x (1, 4, 4, 6). Закрашиваемые интервалы (1, 4) и (4, 6). Условие нахождения локального минимума или максимума определяется при рассмотрении концевых вершин для ребер, соединенных в вершине. Если у обоих концов координаты y больше, чем у вершины пересечения, то вершина – локальный минимум. Если меньше, то вершина пересечения – локальный максимум.



Метод построчного сканирования

3. Сканирующая строка проходит через вершину. Например, по сканирующей строке y = 3 упорядоченный список x получится как (2, 2, 4). Вершина многоугольника была учтена дважды, и поэтому закрашиваемый интервал получается неверным: (2, 2). Следовательно, при пересечении вершины сканирующей строкой она должна учитываться единожды. И список по x в приведенном примере будет (2, 4).



Алгоритмы закрашивания пиксельно-определенных областей

Простейший алгоритм закрашивания. Аналогично случаю закрашивания области, заданной цветом границы нужно задать начальную точку внутри интересующей области, затем необходимо менять цвет тех пикселей, которые принадлежат заданной области.

Использование связности: заполнение области на основе серий пикселов. С целью увеличения производительности вышеприведенных алгоритмов и для предотвращения переполнения стека рассмотрим более тонкий подход к заполнению области — не пиксель за пикселем, а целыми группами пикселей, объединенных в серии.
Серия — это группа соседних пикселей, лежащих на одной строке развертки. Если мы сможем обнаружить и закрасить за один раз целую серию внутри области, то мы радикально ускорим процесс заполнения.