Лекция 8. Атрибуты графических примитивов

Определение, примеры,
атрибуты точек, линий, закрашенных фигур,
алгоритмы закраски, методы сглаживания ступенчатости

Определение, примеры

- Атрибут - это параметр, который влияет на способ изображения примитива.
- Фундаментальные характеристики примитива определяют атрибуты цвета и размера.
- Линии могут быть сплошными или пунктирными, широкими или тонкими, синими или оранжевыми. Отдельные фигуры могут закрашиваться одним цветом или заполняться многоцветным узором.
- Текст может читаться слева направо, изображаться под наклоном по диагонали экрана или в вертикальных колонках. Отдельные символы могут писаться с помощью различных шрифтов, быть разного цвета и размера, а на краях объектов могут применяться эффекты изменения интенсивности, чтобы сгладить растровый «эффект лесенки».
- Во многих графических системах значения атрибутов сохраняются как переменные состояния, и примитивы создаются с учетов текущих значений атрибутов. Когда меняется значение переменной состояния, это отражается только на тех примитивах, которые задаются после этого изменения.

Цвет и шкала яркости

- Цвет – это основной атрибут всех примитивов.
- Цветовые опции могут задаваться численно, выбираться из меню или с помощью ползунков.
- На мониторе эти коды цвета преобразуются в настройки уровня интенсивности свечения пикселей, а для цветного графопостроителя позволяют управлять выделением краски или выбором пера.

Атрибуты точек

- В основном задаются два атрибута точек: цвет и размер.
- В растровых системах размер точки – это целое число, кратное размеру пикселя, так что большие точки изображаются как квадратные блоки пикселей.

Атрибуты прямых линий

- Атрибуты прямолинейного отрезка:
  - цвет;
  - ширина;
  - стиль.
- Как правило, цвет прямой линии задается с помощью той же самой функции, что и для всех графических примитивов.
- Ширина и стиль линии выбираются с помощью отдельных функций для прямых линий. Их настройка может проводиться с использованием таких дополнительных эффектов, как мазки кисти или пера.

Ширина линии

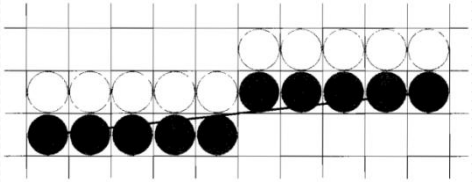


Рис. 1. Растровая линия двойной ширины с тангенсом угла наклона k<1, построенная с помощью вертикальных полос пикселей

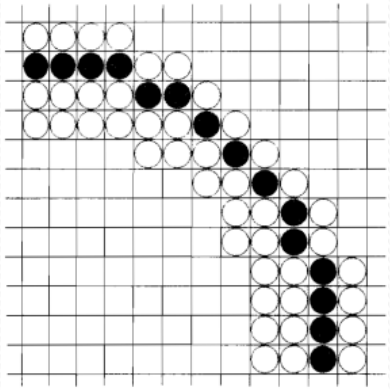


Рис. 2. Дуга окружности шириной 4, построенная с помощью вертикальных либо горизонтальных полос пикселей в зависимости от угла наклона кривой

- Выбор ширины линии зависит от возможностей устройства вывода. Широкие линии на мониторе могут изображаться как расположенные рядом друг и другом параллельные линии, тогда как для перьевого графопостроителя изображение широкой линии требует смены пера.
- В растровых системах линии стандартной ширины строятся путем использования одного пикселя в каждой точке выборки, как в алгоритме Брезенхема. Более широкие линии изображаются кратными положительному целому числу стандартных линий, для чего добавляются дополнительные пиксели на соседних параллельных линиях (рис. 1). Количество добавляемых пикселей устанавливается равным целому значению ширины линии. На рис. 1 в каждой точке выборки x находится соответствующее значение координаты y и изображаются пиксели с экранными координатами (x,y) и (x,y+1). Можно построить прямую линию с шириной 3 или больше, попеременно откладывая пиксели сверху и снизу от траектории прямой с единичной шириной.
- Ширина изображаемой линии (измеренная перпендикулярно к направлению прямой) зависит от тангенса угла ее наклона. Линия с углом наклона 45 градусов будет в 1/корень(2) раз тоньше по сравнению с горизонтальной или вертикальной прямой, построенной с помощью полос пикселей такой же ширины (рис. 2).

Ширина линии

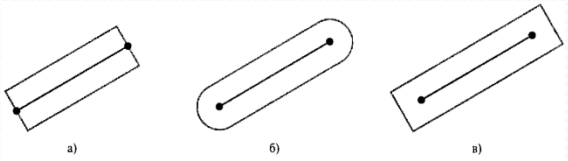


Рис. 3. Широкие линии, построенные с помощью прямоугольного (а), полукруглого (б) и расширенно-прямоугольного (в) окончаний

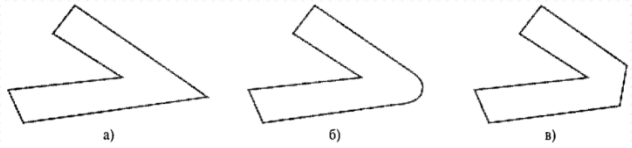


Рис. 4. Отрезки широких линий, соединенные с помощью углового (а), скругленного (б) и косого (в) соединения

- Еще одна проблема, связанная с выбором ширины с помощью горизонтальных и вертикальных полос пикселей, состоит в том, что этот метод дает линии, концы которых вертикальны или горизонтальны, независимо от угла наклона самой прямой. Это особенно заметно при использовании очень широких линий. Форму концов линий всегда можно подправить, чтобы они выглядели лучше, прибавив к ним окончания (рис. 3). Один из видов окончаний для прямой линии – это прямоугольное, для которого характерны прямоугольные края, перпендикулярные к направлению прямой. Если тангенс угла наклона заданной прямой равен k, то тангенс угла наклона прямоугольных концов широкой линии будет 1/k. Тогда на каждом конце заданной прямой каждая из составляющих ее параллельных линий будет изображаться между двумя перпендикулярными линиями. Еще один окончания прямой линии – полукруглое, которое получается путем прибавления закрашенного полукруга к каждому прямоугольному окончанию. Центры этих полукругов находятся в середине широкой линии, а их диаметр равен ширине линии. Третий вид окончаний – расширенно-прямоугольное. В этом случае прямая просто продолжается, и к ней прибавляются прямоугольные перекрытия, которые расположены на расстоянии половины ширины линии от заданных концов отрезка.
- Для непрерывного соединения звеньев широких (более 1 пикселя) ломаных линий необходимы дополнительные действия. На рис. 4 показаны три возможных способа непрерывного соединения двух прямолинейных отрезков. Угловое соединение выполняется путем продолжения внешних границ каждого из двух прямолинейных отрезков до тех пор пока они не пересекутся друг с другом. Скругленное соединение получается, если место соединения двух отрезков перекрывается полукругом, диаметр которого равен ширине линии. А косое соединение получается при изображении прямолинейных отрезков с прямоугольными окончаниями и закрашивании треугольного промежутка в том месте, где пересекаются эти отрезки. Если угол между двумя соединяющимися отрезками очень мал, угловое соединение может привести к появлению очень длинного острия, которое испортит внешний вид ломанной линии. В графических пакетах этого эффекта можно избежать путем перехода от углового соединения в косому в том случае, например, когда угол между двумя соседними отрезками очень мал.

Стиль линии

Рис. 5. Штрихи разной длины, которые изображаются с помощью одинакового количества пикселей

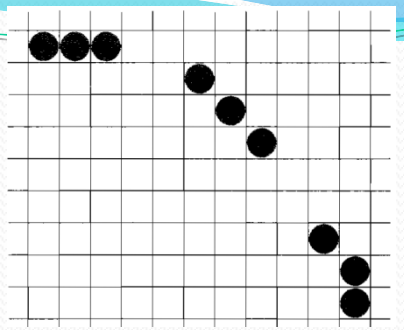


Рис. 6. Пунктирная дуга окружности, изображенная с помощью полос длиной 3 пикселя и расстоянием между штрихами 2 пикселя

- К возможным вариантам выбора атрибута стиля прямой линии можно отнести сплошные линии и различные виды пунктира. Во многих графических редакторах можно выбирать как длину самих штрихов пунктира, так и расстояние между ними.
- Количество пикселей, определяющее длину штриха и расстояние между отдельными штрихами задается с помощью пиксельной маски, которая представляет собой набор из двоичных чисел, сообщающий какие точки, лежащие на прямой, следует изображать. Линейная маска 11111000, например, может быть использована для изображения пунктирной прямой линии с длиной штриха в пять пикселей и расстоянием между штрихами в три пикселя. Пиксели, координатам которых соответствуют биты со значением 1, изображаются в текущем цвете, а пиксели, координатам которых соответствуют биты со значением 0, изображаются цветом фона.
- Построение штрихов с помощью фиксированного количества пикселей приводит к разной длине штрихов при различных ориентациях прямой линии (рис. 5). Оба штриха, показанные на этом рисунке, состоят из четырех пикселей, но диагональный пиксель длиннее в корень(2) раза.
- Для точных чертежей длина штрихов должна оставаться приблизительно постоянной для любой ориентации прямой. Чтобы это условие выполнялось, количество пикселей в сплошных полосах и в промежутках между полосами нужно подбирать в соответствии с углом наклона прямой. Можно сделать так, чтобы два штриха, изображенные на рис. 5, были приблизительно одинаковой длины, уменьшив длину диагонального штриха до трех пикселей.
- На рис. 6 показана часть пунктирной окружности с маской 11100. Используя свойства симметрии можно построить штрихи в различных октантах, но для сохранения правильной последовательности штрихов и промежутков между ними при переходе от одного октанта к следующему нужно будет смещать положения закрашенных и пустых пикселей. Если требуется изобразить штрихи одинаковой длины, нужно при перемещении по окружности подбирать количество пикселей в каждом штрихе. Чтобы получить штрихи одинаковой длины, не применяя пиксельную маску с равными полосами, можно наносить пиксели через одинаковые угловые расстояния.

Выбор перьев и кистей

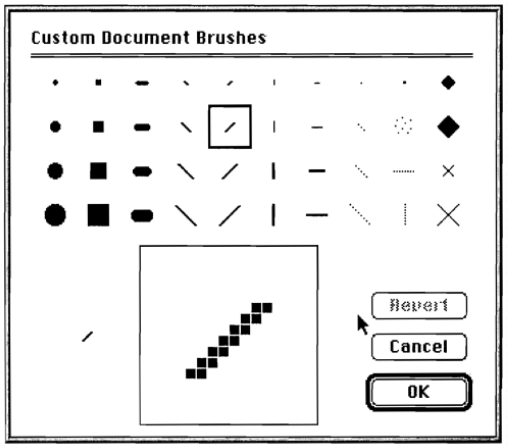
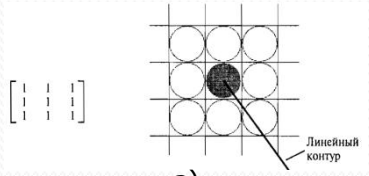
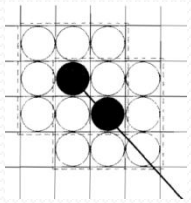


Рис. 7



а)



б)

Рис. 8

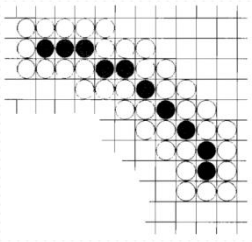


Рис. 9

Рис. 10

Рис. 11

- В некоторых пакетах, особенно в системах рисования и черчения, можно непосредственно выбирать различные виды перьев и кистей. Они различаются своей формой, размером и узором (рис. 7). Эти формы записываются в пиксельную маску, которая служит для определения массива координат пикселей и отображается на траектории прямой линии. Например, след от прямоугольного пера можно реализовать с помощью маски (рис. 8,а) по траектории прямой линии, как показано на рис. 8,б.
- На рис. 9 показана дуга окружности в первом квадранте. Здесь центр прямоугольного пера перемещается в последующие положения на кривой. Чтобы кривая имела однородную ширину, нужно поворачивать прямоугольное перо таким образом, чтобы оно совпадало с направлением кривой при перемещении по ней, или использовать перо круглой формы.
- Программы рисования и черчения позволяют изображать кривые, форма которых создается в интерактивном режиме с помощью таких устройств-указателей, как световое перо или графический планшет. Несколько примеров узоров из таких кривых, нарисованных с помощью кисти в форме квадрата, круга, диагональной линии, выцветшего аэрографа и точечного узора, показано на рис. 10. Дополнительной опцией, которая может предлагаться в пакетах рисования, является имитация мазков кисти. На рис. 11 показан рисунок, который получен с помощью различных видов мазков кисти.

Атрибуты закрашенных фигур

- В большинстве графических пакетов закрашенные фигуры ограничиваются многоугольниками, которые можно описать с помощью линейных уравнений. Возможно и более жесткое условие – все закрашенные фигуры должны быть выпуклыми многоугольниками (чтобы строки развертки не пересекались более чем с двумя поверхностями границ). Однако в общем случае закрашивать можно любые заданные участки, в том числе окружности, эллипсы и другие объекты с криволинейными границами. Кроме того, такие приложения, как программы рисования, предлагают опции для закрашивания областей произвольной формы.
- Существуют два метода закрашивания участков рисунка в растровых системах, если описание границы закрашенной области переведено в координаты пикселей:
  - В первом методе вначале определяются интервалы перекрытия для строк развертки, которые пересекают эту область. Затем пикселям, попадающим в эти интервалы перекрытия, присваиваются значения, соответствующие узору заполнения. Этот метод применяется для таких простых фигур, типа многоугольников или окружностей, также этот метод используется в универсальных графических пакетах.
  - Другой метод заполнения участков рисунка – начать с заданной точки, находящейся внутри этой области, и «зарисовывать пространство» наружу от этой точки, пока не дойдем до границы. Алгоритмы закрашивания, в которых применяется начальная внутренняя точка, применяются при закрашивании участков с более сложными границами, а также в интерактивных системах рисования.

Стили закрашивания

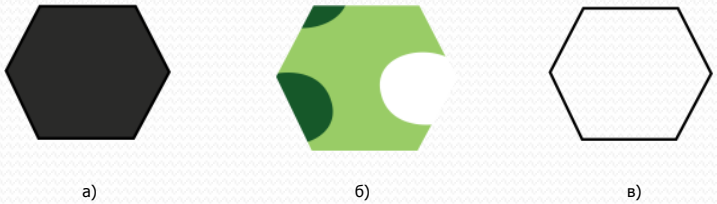


Рис. 12

- Основной атрибут закрашенных фигур, предлагаемый в универсальной графической библиотеке - это стиль изображения внутренней области. Данную область можно заполнить одним цветом (рис. 12,а) или заданным узором (рис. 12,б), или же оставить ее без заливки (рис. 12,в), показав только границы фигуры. Можно закрашивать выбранные участи сцены с помощью различных видов кистей, комбинаций смешивания цветов и текстур. Границы многоугольников можно показать другим цветом. Можно выбирать различные атрибуты изображения передней и задней сторон многоугольной области.
- Узоры заполнения могут задаваться в форме прямоугольных цветовых массивов, содержащих различные цвета для различных элементов массива, или битовых массивов, в которых указывается, какие относительные положения следует изображать одним выбранным цветом. Массив, с помощью которого описывается узор заполнения, называется маской, и он применяется к области изображения.

Смешивание цветов

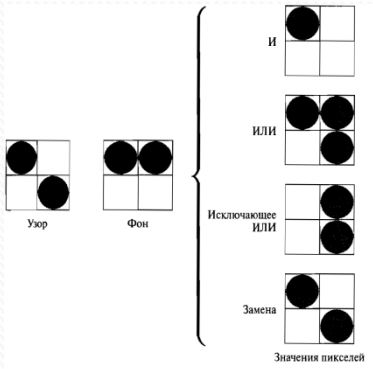


Рис. 13

- Существует несколько способов сочетания узора заполнения объектов с цветом фона.
- С помощью коэффициента прозрачности, показывающего, насколько фон должен смешиваться с цветом объекта. В качестве примера заполнения такого типа рассмотрим линейный алгоритмом. Обозначим F – цвет окрашивания, B – цвет фона. Текущий цвет P каждого пикселя в пределах закрашиваемой области является некоторой линейной комбинацией цветов F и B:

$$P=t*F + (1-t)*B,$$

  где коэффициент прозрачности (смешивания) t принимает значение от 0 до 1 для каждого пикселя. Для значений t, которые меньше 0.5, цвет фона дает больший вклад в результирующий цвет объекта, чем цвет заполнения.
- С помощью логических операций и операций замены. На рис. 13 показано объединение элемента узора заполнения размером 2 на 2 с узором фона для бинарной (черно-белой) системы с помощью логических операций.

Закраска многоугольников

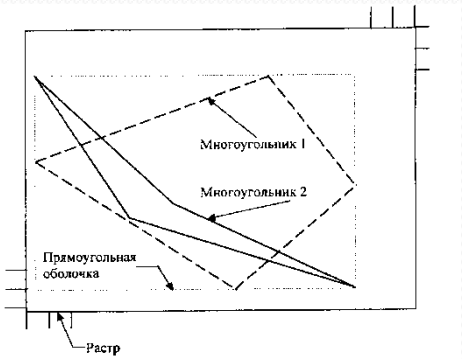


Рис. 14. Прямоугольная оболочка контура

- Простейший метод заполнения многоугольника состоит в проверке на принадлежность внутренности многоугольника каждого пикселя в растре в порядке сканирования строк. Поскольку обычно большинство пикселей лежит вне многоугольника, то данный метод слишком расточителен. Затраты можно уменьшить путем вычисления для многоугольника прямоугольной оболочки – наименьшего прямоугольника, содержащего внутри себя многоугольник. Проверяются только внутренние точки этой оболочки.

Алгоритм построчного заполнения многоугольника

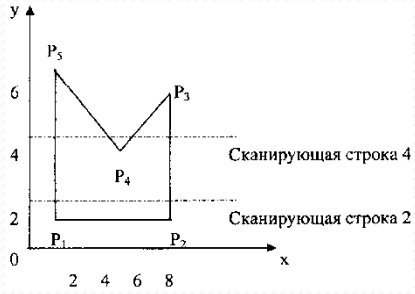
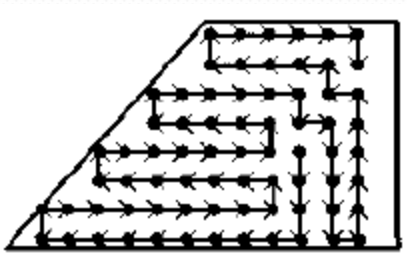


Рис. 15 Заполнение области внутри многоугольника

- Предположим, что область, подлежащая заполнению, представляет собой многоугольник, заданный совокупностью вершин $P_i=(X_i,Y_i)$, причем соседние точки в этом списке являются смежными вершинами сторон (рис. 15).
- Для каждой строки развертки пикселей находят пересечения этой строки с ребрами многоугольника и сортируют пересечения по возрастанию X. Затем серии пикселей между всеми парами закрашивают.
- На рис. 15 строка развертки 4 пересекает 4 ребра. Четыре абсциссы точек пересечения округляются до ближайшего целого и сортируются. В процессе движения вдоль строки развертки пикселей мы при каждом пересечении попадаем или внутрь многоугольника, или наружу от него, причем это состояние все время меняется.
- Если какие-либо ребра пересекались i-й строкой, то они скорее всего будут пересекаться также и строкой $i+1$. При переходе к новой строке легко вычислить новую координату X точки пересечения ребра, используя координату X старой точки пересечения и тангенс угла наклона ребра. Так продолжается до появления вершины P_i ребра.
- Для каждой строки сканирования рассматриваются только те ребра, которые пересекают строку. При появлении в строке сканирования вершин выполняется перестройка списка активных ребер. Ребра, которые перестали пересекаться, удаляются из списка, а все новые ребра, пересекаемые строкой, заносятся в него.

Алгоритм заполнения области с затравкой



```
flood_fill(int x, int y)
{
     if(read_pixel(x,y)==WHITE)
     {
          write_pixel(x,y,BLACK);
          flood_fill(x-1,y);
          flood_fill(x+1,y);
          flood_fill(x,y-1);
          flood_fill(x,y+1);
      }
}
```

Рис. 16

- В алгоритмах заполнения области с затравкой задается заполняемая область и начальная точка в области, начиная с которой начинается заполнение. Можно представить, что в начальной (затравочной) точке находится источник, заливающий всю область определенным цветом. Поэтому этот процесс иногда называют заливкой области.
- Для всех соседних пикселей проверяется, являются ли они граничными или уже перекрашенными. Если нет – то заносим этого соседа в стек и выполняем для него аналогичный алгоритм (рис. 16).
- Недостатком алгоритмов, непосредственно использующих связность закрашиваемой области, являются большие затраты памяти на стек, так как на каждый закрашенный пиксель в стеке по максимуму будет занесена информация еще о трех соседних. Кроме того, информация о некоторых пикселях может записываться в стек многократно.

Сглаживание ступенчатости

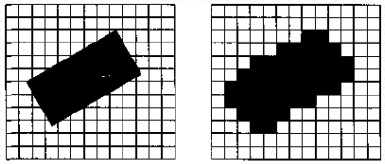


Рис. 17

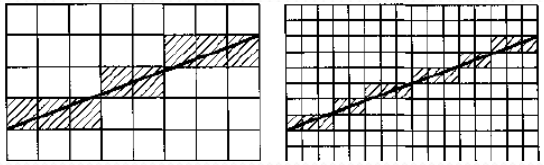
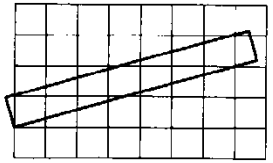


Рис. 18



0	0	0	0	0	0	0	0
0	0	0	3	7	8	12	8
3	7	10	13	10	7	3	0
12	7	4	2	0	0	0	0
0	0	0	0	0	0	0	0

Рис. 19

1/16	1/16	1/16
1/16	1/2	1/16
1/16	1/16	1/16

0	1/8	0
1/8	1/2	1/8
0	1/8	0

1/16	1/8	1/16
1/8	1/4	1/8
1/16	1/8	1/16

Рис. 20

- Появление ступенчатости, или лестничного эффекта (например, при растеризации линий с наклоном) связано с дискретной природой пикселей. На рис. 17 показан пример растеризации прямоугольника.
- Можно выделить два подхода к сглаживанию ступенчатости:
- Увеличение разрешения растрового изображения (рис. 18). Если разрешение выходного изображения ограничено возможностями графических устройств, то следует все вычисления проводить с высоким разрешением, а изображать с более низким, используя усреднения некоторого типа для получения атрибутов пикселей с более низким разрешением. Хотя результат будет выглядеть лучше, но квадратично увеличится расход памяти и время растеризации, поэтому этот подход дорогостоящий и полностью проблему сглаживания ступенчатости не решает.
- Размытие линий или границ областей при заданном разрешении. Возможно только в случае, если каждый пиксель будет иметь больше двух градаций яркости.
- В простейших алгоритмах сглаживания ступенчатости яркость пикселя вычисляется на основе площади пикселя, которая покрывается объектом. В этом случае линия, которая в идеале имеет нулевую толщину, представляется как линия шириной в один пиксель. На рис. 19 показана исходная линия и градации яркости растеризованной линии в случае 16 градаций яркости (0 – черный цвет, 15 – белый цвет).
- В более сложных алгоритмах сглаживания ступенчатости значение яркости каждого пикселя вычисляется как средневзвешенное значение соответствующего набора соседних пикселей (обычно из 8 ближайших соседей). На каждый ненулевой пиксель поочередно накладывается квадратная маска свертки некоторого фильтра. Затем вес каждой клетки умножается на соответствующее значение пикселя, девять произведений складываются и образуют значение яркости обрабатываемого пикселя. На рис. 20 приведены примеры масок, используемых в практических алгоритмах. Во всех масках вес центрального пикселя превышает вес соседей, а веса остальных пикселей линейно возрастают по мере движения от краев маски к ее центру. Иногда используют большие маски размером 5x5 и 7x7.