5. Алгоритмы отсечения

5.1. Двумерные алгоритмы отсечения невидимых линий

Рассмотрим проблему, незримо присутствующую в большинстве задач компьютерной графики. Эта проблема отсечения изображения по некоторой границе, например, по границе экрана, или, в общем случае, некоторого окна. Рассмотрим задачу отсечения применительно к отрезкам прямых. Некоторые из них полностью лежат внутри области экрана, другие целиком вне ее, а некоторые пересекают границу экрана. Правильное отображение отрезков означает нахождение точек пересечения их с границей экрана и рисование только тех их частей, которые попадают на экран.

Область, относительно границ которой проводится отсечение, будем называть областью видимости (ОВ). Будем считать, что областью видимости может являться некоторый выпуклый многоугольник (см. рис. 5.1). Продолжим каждую из границ

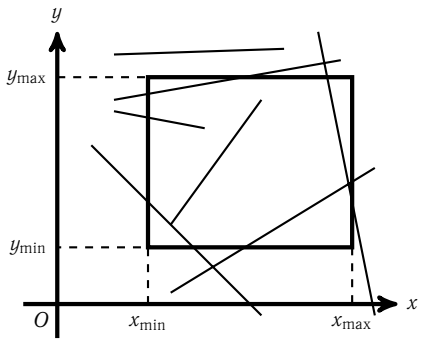


Рис. 5.1. Границы прямоугольной области видимости.

области видимости до бесконечных прямых. Каждая из таких прямых делит плоскость на 2 части. Назовем «стороной внутренности» ту полуплоскость, в которой находится область видимости, как это показано на рис. 5.2. Другую полуплоскость назовем «стороной внешности».

Таким образом, область видимости может быть определена как область, которая находится на стороне внутренности всех линий границ.

Рассматривая положение отрезка относительно линии i-ой границы ОВ будем представлять его в виде связанного вектора. В таком случае можно охарактеризовать

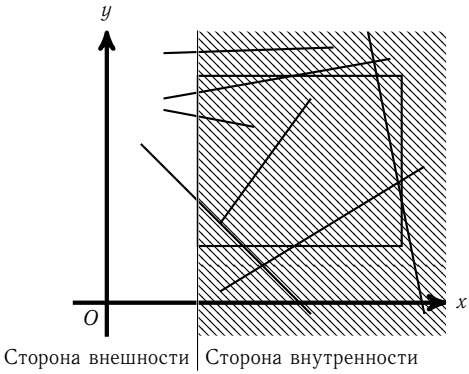


Рис. 5.2. Стороны внешности и внутренности.

направление этого отрезка. Продолжим отрезок до прямой линии. Будем говорить, что отрезок направлен в сторону внутренности относительно линии i-ой границы OB, если двигаясь по линии отрезка в направлении, заданном отрезком как связанным вектором, пересекая i-ю границу OB попадаем со стороны внешности на сторону внутренности.

Подобное определение можно сказать о направлении отрезка в сторону внешности относительно линии i-ой границы OB.

5.1.1. Алгоритм Коэна — Сазерленда

Алгоритм предполагает, что область видимости — прямоугольник, стороны которого параллельны осям координат.

Этот алгоритм позволяет быстро выявить отрезки, которые могут быть или приняты или отброшены целиком. Вычисление пересечений требуется когда отрезок не попадает ни в один из этих классов. Этот алгоритм особенно эффективен в двух крайних случаях:

- большинство примитивов содержится целиком в большом окне,
- большинство примитивов лежит целиком вне относительно маленького окна.

Идея алгоритма состоит в следующем: Прямые линии, ограничивающие область видимости делят всё двумерное пространство на 9 областей (см. рис. 5.3).

Каждой из областей присваивается 4-х-разрядный двоичный код. Пример такой кодировки приведен на рис. 5.4. Здесь:

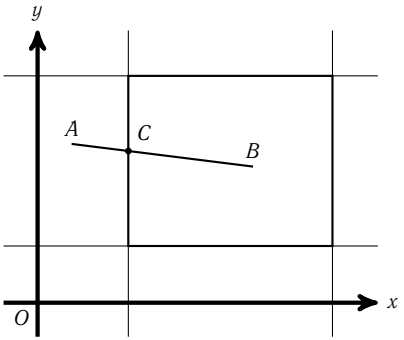


Рис. 5.3. Девять областей при прямоугольной области видимости

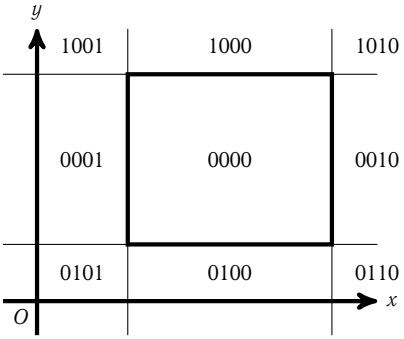


Рис. 5.4. Кодировка областей.

- единица в первом разряде — область слева от левой границы области видимости;
- единица во втором разряде — область справа от правой границы области видимости;
- единица в третьем разряде — область под нижней границей области видимости;
- единица в четвертом разряде — область над верхней границей области видимости;

Две конечные точки отрезка получают 4-х-разрядные двоичные коды, соответствующие областям, в которые они попали. В нижеприведенном алгоритме будем обозначать $C(A)$ — код области, в которую попала точка A.

Определение того является ли отрезок видимым выполняется следующим образом:

- если коды обоих концов отрезка равны 0, то отрезок целиком внутри окна, отсечение не нужно, отрезок принимается как тривиально видимый (отрезок AB на рис. 5.5);

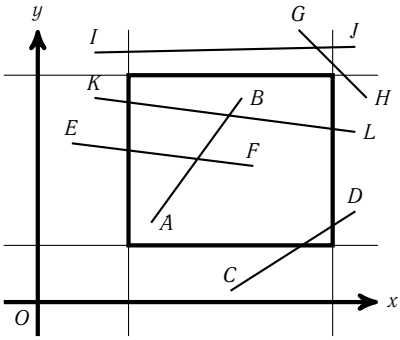


Рис. 5.5. Примеры расположения отрезков.

- если логическое поразрядное «И» кодов обоих концов отрезка не равно нулю, то отрезок целиком вне окна, отсечение не нужно, отрезок отбрасывается как тривиально невидимый (отрезок IJ на рис. 5.5);
- если логическое поразрядное «И» кодов обоих концов отрезка равно нулю, то отрезок подозрительный, он может быть частично видимым (отрезки CD, EF, KL) или целиком невидимым (отрезок GH); для него нужно определить координаты пересечений со сторонами окна и для каждой полученной части определить тривиальную видимость или невидимость. При этом для отрезков EF и GH потребуется вычисление одного пересечения, для остальных (CD и KL) - двух.

При расчете пересечения используется горизонтальность либо вертикальность сторон окна, что позволяет определить координату x или y точки пересечения без вычислений.

Несколько детальней метод Коэна—Сазерленда представлен в алгоритме 4.

Алгоритм 4: Алгоритм Коэна—Сазерленда

Вход: S — список всех отрезков сцены.
Выход: S_1 — список всех видимых частей отрезков сцены.
начало алгоритма
  $S_1 = \varnothing$;
  цикл пока $S \neq \varnothing$ **выполнять**
    Пусть AB очередной отрезок из S;
    Определить C_1 и C_2 — коды областей, в которые попали точки A и B соответственно;
    если $C_1 = C_2 = 0$ **то** *// отрезок полностью видим*
      Поместить AB в список S_1;
    иначе если $C_1 \& C_2 = 0$ **то** *// отрезок может быть частично видим*
      если $C_1 = 0$ **то** поменять местами значения A и B;
      // Теперь A точно за пределами области видимости.
      Определить одну из границ области видимости, за которой лежит точка A;
      Найти точку C — точку пересечения AB с этой границей;
      Поместить отрезок CB в список S;
    // Иначе, если $C_1 \& C_2 \neq 0$, то отрезок полностью невидим.
  Выдать S_1;
конец алгоритма

5.1.2. Алгоритм Лианга — Барски

В 1982 г. Лианг и Барски предложили алгоритмы отсечения прямоугольным окном с использованием параметрического представления для двух, трех и четырехмерного отсечения. По утверждению авторов, данный алгоритм в целом превосходит алгоритм Коэна — Сазерленда. Однако, это утверждение справедливо только для случая когда оба конца видимого отрезка вне окна и окно небольшое (до 50×50 при разрешении 1000×1000).

Пусть внутренняя часть области видимости может быть выражена с помощью следующих неравенств (рис. 5.1).

$$\begin{cases} x_{\min} \leqslant x \leqslant x_{\max}, \\ y_{\min} \leqslant y \leqslant y_{\max}. \end{cases} \tag{5.1}$$

Отсекаемый отрезок прямой может быть представлен в параметрическом представлении (как в п. 2.3).

$$\begin{cases} x(t) = x_A + (x_B - x_A)t, \\ y(t) = y_A + (y_B - y_A)t, \\ 0 \leqslant t \leqslant 1. \end{cases} \tag{5.2}$$

Подставляя параметрическое представление, заданное уравнениями (5.2), в неравенства (5.1), получим следующие соотношения для части отрезка, находящейся в окне отсечения:

$$\begin{cases} x_{\min} \leqslant x_A + (x_B - x_A)t \leqslant x_{\max}, \\ y_{\min} \leqslant y_A + (y_B - y_A)t \leqslant y_{\max}, \\ 0 \leqslant t \leqslant 1. \end{cases} \tag{5.3}$$

Рассматривая неравенства (5.3), видим, что они имеют одинаковую форму вида:

$$\begin{cases} P_i t \leqslant Q_i, \\ i = 1, 2, 3, 4, \\ 0 \leqslant t \leqslant 1. \end{cases}$$

Здесь использованы следующие обозначения:

$$\begin{aligned} P_1 &= x_A - x_B; & Q_1 &= x_A - x_{\min}; \\ P_2 &= x_B - x_A; & Q_2 &= x_{\max} - x_A; \\ P_3 &= y_A - y_B; & Q_3 &= y_A - y_{\min}; \\ P_4 &= y_B - y_A; & Q_4 &= y_{\max} - y_A. \end{aligned} \tag{5.4}$$

Вспоминая определения стороны внутренности и стороны внешности, замечаем, что каждое из неравенств (5.4) соответствует одной из граничных линий (левой, правой, нижней и верхней, соответственно) и описывает ее видимую сторону. Удлиним AB до бесконечной прямой. Тогда каждое неравенство задает диапазон значений параметра t, для которых эта удлиненная линия находится на видимой стороне соответствующей линии границы. Более того, конкретное значение параметра t для точки пересечения есть $t = Q_i/P_i$.

Знак Q_i показывает на какой стороне соответствующей линии границы находится точка A. А именно, если $Q_i \geqslant 0$, тогда A находится на стороне внутренности линии границы, включая и ее. Если же $Q_i < 0$, тогда A находится на внешней (невидимой) стороне.

Рассмотрим P_i в соотношениях 5.4. Ясно, что любое P_i может быть меньше 0, больше 0 и равно 0. Если $P_i < 0$, тогда отрезок направлен в сторону внутренности линии i-ой границы, а значит, при пересечении с линией i-ой границы начало отрезка остается на стороне внешности (см. рис. 5.6, *а*). Если $P_i > 0$, тогда отрезок направлен

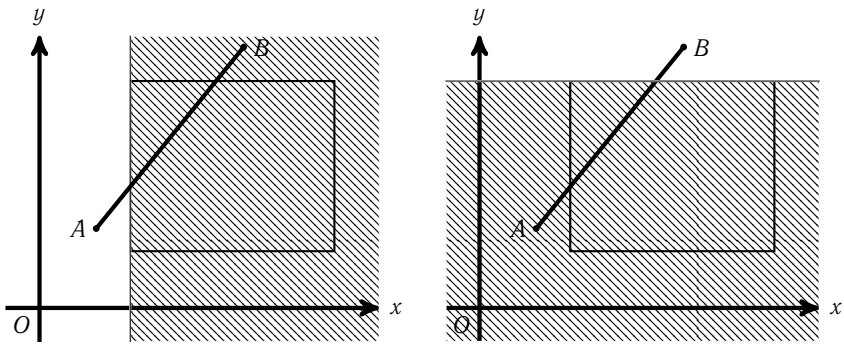


Рис. 5.6. Отрезок направлен: (*а*) в сторону внутренности относительно левой (первой) границы; (*б*) в сторону внешности относительно верхней (четвертой) границы

в сторону внешности линии i-ой границы, а значит, при пересечении с линией i-ой границы конец отрезка остается на стороне внешности (см. рис. 5.6, *б*). Если $P_i = 0$ — отрезок или вырожден в точку, или параллелен i-ой границе, но, в любом случае, он либо полностью находится на стороне внутренности или полностью на стороне внешности относительно линии i-ой границы.

Для всех i таких, что $P_i < 0$, условия видимости имеет вид: $t \geqslant Q_i/P_i$. Из условия принадлежности точек удлиненной линии отрезку AB имеем $t \geqslant 0$. Таким образом, нужно искать:

$$t \geqslant \max(\{Q_i/P_i \mid P_i < 0,\ i = 1,2,3,4\} \cup \{0\}). \tag{5.5}$$

Аналогично, для всех i таких что $P_i > 0$, условия видимости — $t \leqslant Q_i/P_i$ и $t \leqslant 1$. А следовательно

$$t \leqslant \min(\{Q_i/P_i \mid P_i > 0,\ i = 1,2,3,4\} \cup \{1\}). \tag{5.6}$$

Наконец, для всех i, таких что $P_i = 0$ следует проверить знак Q_i. Если $Q_i < 0$, то это случай тривиального отбрасывания, задача отсечения решена и дальнейшие вычисления не нужны. Если же $Q_i \geqslant 0$, то информации, даваемой неравенством, недостаточно и это неравенство игнорируется.

Правая часть неравенств (5.5) и (5.6) — значения параметра t, соответствующие началу и концу видимого сегмента, соответственно. Обозначим эти значения как $t_{\min}$

и $t_{\max}$:

$$\begin{cases} t_{\min} = \max(\{Q_i/P_i \mid P_i < 0,\ i = 1,2,3,4\} \cup \{0\}), \\ t_{\max} = \min(\{Q_i/P_i \mid P_i > 0,\ i = 1,2,3,4\} \cup \{1\}). \end{cases} \tag{5.7}$$

Если сегмент отрезка AB видим, то ему соответствует интервал параметра

$$t_{\min} \leqslant t \leqslant t_{\max}.$$

Следовательно, необходимое условие видимости сегмента:

$$t_{\min} \leqslant t_{\max}$$

Но это недостаточное условие, так как оно игнорирует случай тривиального отбрасывания при $P_i = 0$, если $Q_i < 0$. Тем не менее это достаточное условие для отбрасывания, т. е. если $t_{\min} > t_{\max}$, то отрезок должен быть отброшен. Алгоритм проверяет, если $P_i = 0$ с $Q_i < 0$, или $t_{\min} > t_{\max}$ и в этом случае отрезок немедленно отбрасывается без дальнейших вычислений.

В алгоритме $t_{\min}$ и $t_{\max}$ инициализируются в 0 и 1, соответственно. Затем последовательно рассматривается каждое отношение Qi/Pi.

Если $Pi < 0$, то отношение вначале сравнивается с $t_{\max}$ и, если оно больше $t_{\max}$, то это случай отбрасывания. В противном случае оно сравнивается с $t_{\min}$ и, если оно больше, то $t_{\min}$ должно быть заменено на новое значение.

Если $P_i > 0$, то отношение вначале сравнивается с $t_{\min}$ и, если оно меньше $t_{\min}$, то это случай отбрасывания. В противном случае оно сравнивается с $t_{\max}$ и, если оно меньше, то $t_{\max}$ должно быть заменено на новое значение.

Наконец, если $P_i = 0$ и $Q_i < 0$, то это случай отбрасывания.

На последнем этапе алгоритма, если отрезок еще не отброшен, то $t_{\min}$ и $t_{\max}$ используются для вычисления соответствующих точек.

Общую схему метода Лианга—Барски можно увидеть в алгоритме 5.

Алгоритм 5: Алгоритм Лианга—Барски отсечения отрезков

Вход: S — список всех отрезков сцены.
Выход: S_1 — список всех видимых частей отрезков сцены.
начало алгоритма
  $S_1 = \varnothing$;
  цикл пока $S \neq \varnothing$ **выполнять**
    Пусть AB очередной отрезок из S;
    $t_{min} = 0$; $t_{\max} = 1$; $visible = True$;
    цикл для i *от* 1 *до* 4 **выполнять**
      Вычисляем P_i и Q_i;
      если $P_i = 0$ **то**
        если $Q_i < 0$ **то** *// отрезок полностью невидим*
          $visible = False$; **закончить цикл**;
      иначе
        если $P_i > 0$ **то** $t_{\max} = \min\left\{t_{\max}, \dfrac{Q_i}{P_i}\right\}$;
        иначе $t_{\min} = \max\left\{t_{\min}, \dfrac{Q_i}{P_i}\right\}$;
        если $t_{\min} > t_{\max}$ **то** $visible = False$; **закончить цикл**;
    если $visible$ **то**
      $A' = A + (B - A)t_{\min}$; $B' = A + (B - A)t_{\max}$;
      Поместить отрезок $A'B'$ в S_1;
  Выдать S_1;
конец алгоритма

5.1.3. Алгоритм Кируса — Бека

Рассмотренные выше алгоритмы проводили отсечение по прямоугольному окну, стороны которого параллельны осям координат. Это, конечно, наиболее частый случай отсечения. Однако, во многих случаях требуется отсечение по произвольному многоугольнику, например, в алгоритмах удаления невидимых частей сцены. В этом случае наиболее удобно использование параметрического представления линий, не зависящего от выбора системы координат.

Из предыдущего пункта ясно, что для выполнения отсечения в параметрическом представлении необходимо иметь способ определения ориентации удлиненной линии, содержащей отсекаемый отрезок, относительно линии границы — с внешней стороны на внутреннюю или с внутренней на внешнюю, а также иметь способ определения расположения точки, принадлежащей отрезку, относительно окна — вне, на границе, внутри.

Для этих целей в алгоритме Кируса—Бека, реализующем отсечение произвольным выпуклым многоугольником, используется вектор внутренней нормали к ребру окна.

Пусть область видимости — n-угольник, с вершинами, перечисленными по часовой

стрелке, $F_1, F_2, \ldots, F_n$. i-м ребром этого многоугольника будем называть ребро

$$\begin{cases} F_iF_{i+1}, & \text{если } 1 \leqslant i < n; \\ F_nF_1, & \text{если } i = n. \end{cases}$$

Будем обозначать N_i — нормаль к i-му ребру области видимости, направленную в сторону внутренности относительно i-го ребра (см. рис. 5.7).

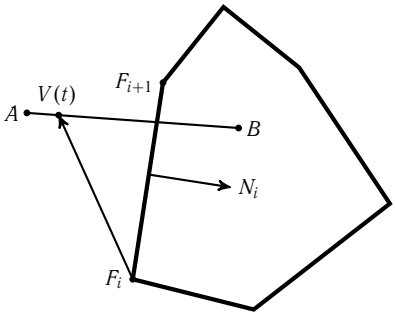


Рис. 5.7. Рассматриваемые векторы в алгоритме Кируса—Бека.

Рассмотрим основные идеи алгоритма Кируса — Бека.

Ориентация отрезка AB относительно i-й стороны области видимости определяется знаком скалярного произведения $P_i = (B - A)N_i$.

- При $P_i < 0$ отсекаемый отрезок направлен с внутренней на внешнюю стороны i-й граничной линии окна (так как, в таком случае, угол между отрезком и вектором нормали N_i больше $\pi/2$);
- при $P_i = 0$ либо точки A и B совпадают, либо отсекаемый отрезок параллелен i-й граничной линии окна;
- при $P_i > 0$ отсекаемый отрезок направлен с внешней на внутреннюю сторону i-й граничной линии окна (так как, в этом случае, угол между отрезком и вектором нормали N_i меньше $\pi/2$).

Для определения расположения точки относительно окна вспомним параметрическое представление отсекаемого отрезка:

$$\begin{cases} V(t) = A + (B - A)t, \\ 0 \leqslant t \leqslant 1. \end{cases} \tag{5.8}$$

Рассмотрим скалярное произведение внутренней нормали N_i к i-й границе на вектор $V(t) - F_i$ — вектор, начинающийся в начальной точке ребра окна и заканчивающийся в некоторой точке $V(t)$ удлиненной линии (см. рис. 5.7).

$$Q_i(t) = (V(t) - F_i)N_i. \tag{5.9}$$

Имеем:

- при $Q_i(t) < 0$ точка $V(t)$ лежит с внешней стороны i-ой границы;
- при $Q_i(t) = 0$ точка $V(t)$ лежит на несущей прямой i-ой границы;
- при $Q_i(t) > 0$ точка $V(t)$ лежит с внутренней стороны i-ой границы.

Таким образом, если необходимо найти точку пересечения линии отрезка с несущей прямой границы области видимости, то нужно найти такое t, для которого выполняется

$$Q_i(t) = 0$$

Распишем $Q_i(t)$ через (5.9) и подставим вместо $V(t)$ правую часть равенства из (5.8), получим

$$(A + (B - A)t - F_i)N_i = 0.$$

Отсюда получим

$$(B - A)N_i t - (F_i - A)N_i = 0. \tag{5.10}$$

Вспомним, что $A = V(0)$, откуда следует

$$P_i t + Q_i(0) = 0.$$

Разрешая равенство относительно t получим

$$t = -\frac{Q_i(0)}{P_i}. \tag{5.11}$$

Равенство (5.11) дает значение параметра t в параметрическом уравнении удлиненной прямой с границей области видимости.

Алгоритм построен следующим образом: Искомые значения параметров $t_{\min}$ и $t_{\max}$ точек концов видимой части отрезка инициализируются значениями 0 и 1, соответствующими началу и концу отсекаемого отрезка.

Затем в цикле для каждой i-й стороны окна отсечения вычисляются значения скалярных произведений, входящих в (5.11).

Если очередное P_i равно 0, то отсекаемый отрезок либо вырожден в точку, либо параллелен i-й стороне окна. В этом случае достаточно проанализировать знак Q_i. Если $Q_i < 0$, то отрезок вне окна и отсечение закончено иначе рассматривается следующая сторона окна.

Если же P_i не равно 0, то по (5.11) можно вычислить значение параметра t для точки пересечения отсекаемого отрезка с i-й границей. Так как отрезок AB соответствует диапазону $0 \leqslant t \leqslant 1$, то все решения, выходящие за данный диапазон следует отбросить. Выбор оставшихся решений определяется знаком P_i.

Если $P_i < 0$, т. е. удлиненная линия направлена с внутренней на внешнюю стороны граничной линии, то проводится поиск значения параметра для конечной точки видимой части отрезка. В этом случае определяется минимальное значение из всех получаемых решений. Оно даст значение параметра $t_{\max}$ для конечной точки отсеченного отрезка. Если текущее полученное значение $t_{\max}$ окажется меньше, чем $t_{\min}$, то отрезок отбрасывается, так как нарушено условие $t_{\min} \leqslant t_{\max}$.

Если же $P_i > 0$, т. е. удлиненная линия направлена с внешней на внутреннюю стороны граничной линии, то проводится поиск значения параметра для начальной точки видимой части отрезка. В этом случае определяется максимальное значение из всех получаемых решений. Оно даст значение параметра $t_{\min}$ для начальной точки отсеченного отрезка. Если текущее полученное значение $t_{\min}$ окажется больше, чем $t_{\max}$, то отрезок отбрасывается, так как нарушено условие $t_{\min} \leqslant t_{\max}$.

На заключительном этапе алгоритма значения $t_{\min}$ и $t_{\max}$ используются для вычисления координат точек пересечения отрезка с границами OB.

Общая схема метода изложена в алгоритме 6.

Алгоритм 6: Алгоритм Кируса—Бека отсечения отрезков

Вход: S — список всех отрезков сцены.
Выход: S_1 — список всех видимых частей отрезков сцены.
начало алгоритма
  $S_1 = \varnothing$;
  цикл пока $S \neq \varnothing$ **выполнять**
    Пусть AB очередной отрезок из S;
    $t_{min} = 0$; $t_{\max} = 1$; $visible = True$;
    цикл для i *от* 1 *до* n **выполнять** *// n — количество ребер OB*
      Вычисляем P_i и $Q_i(0)$;
      если $P_i = 0$ **то**
        если $Q_i(0) < 0$ **то** *// отрезок полностью невидим*
          $visible = False$; **закончить цикл**;
      иначе
        если $P_i > 0$ **то** $t_{\min} = \max\left\{t_{\min}, -\dfrac{Q_i(0)}{P_i}\right\}$;
        иначе $t_{\max} = \min\left\{t_{\max}, -\dfrac{Q_i(0)}{P_i}\right\}$;
        если $t_{\min} > t_{\max}$ **то** $visible = False$; **закончить цикл**;
    если $visible$ **то**
      $A' = A + (B - A)t_{\min}$; $B' = A + (B - A)t_{\max}$;
      Поместить отрезок $A'B'$ в S_1;
  Выдать S_1;
конец алгоритма

5.1.4. Алгоритм Скала

В 1993 году Вацлав Скала предложил еще один алгоритм отсечения отрезков относительно области видимости, заданной выпуклым многоугольником. Основная идея его алгоритма состоит в том, что так как область видимости — выпуклый многоугольник, то любая частично-видимая прямая пересекает границу этого многоугольника точно в двух точках (исключая вырожденные случаи, когда прямая касается границы области видимости). Чтобы определить пересекает ли прямая, на которой лежит отрезок AB, ребро F_iF_{i+1} области видимости Скала предлагает сравнивать знаки псевдоскалярных произведений $\xi = (B - A) \times (F_i - A)$ и $\eta = (B - A) \times (F_{i+1} - A)$, т. е. произведение вектора отрезка с векторами начинающимися в начальной точке отрезка и заканчивающимися в конечных точках ребра области видимости (см. рис. 5.8). Если знаки ξ и η одинаковые, то ребро F_iF_{i+1} лежит по одну сторону от прямой, на которой лежит отрезок AB. Если знаки разные, то прямая на которой лежит AB, пересекает F_iF_{i+1}.

В ходе алгоритма производится подсчет количества ребер области видимости, для

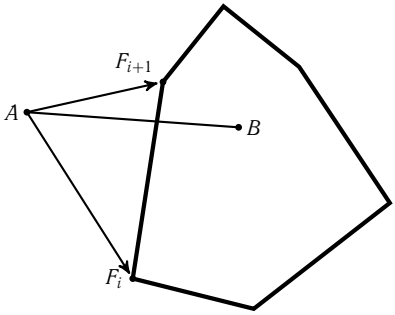


Рис. 5.8. Векторы для произведения в алгоритме Скала.

которых найдется точка пересечения с прямой отрезка AB.

Если найдутся два ребра области видимости, которые пересекаются с прямой отрезка AB, то предлагается проверить принадлежат ли точки пересечения прямой отрезку. Если так, то в соответствующие точки пересечения следует перенести концы отрезка. Например, на рисунке 5.9 прямая отрезка AB пересекает пересекает ребра области видимости в точках C_1 И C_2. Но только лишь точка C_1 принадлежит отрез-

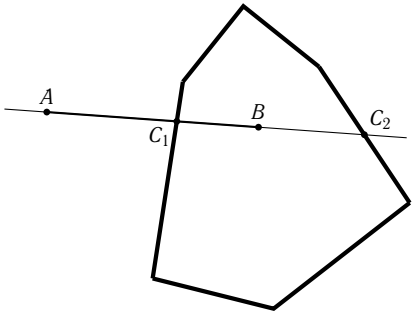


Рис. 5.9. Отсечение отрезка в алгоритме Скала.

ку AB. Только один конец отрезка будет изменен: точка A будет перенесена в точку пересечения C_1.

Когда показатели ξ или η обращается в ноль, то прямая отрезка AB проходит через точку F_i или F_{i+1} соответственно, и в этом случае необходимо проводить дополнительные проверки. Если ξ и η обратились в ноль одновременно, то прямая отрезка AB является несущей прямой данного ребра области видимости. В таком случае не стоит увеличивать счетчик точек пересечения (так как он будет увеличен при рас-

смотрении соседних ребер). Если же в ноль обратился только один из показателей, то нужно зачесть вершину, через которую проходит прямая, за точку пересечения. При этом стоит учесть, что при проверке второго ребра, инцидентного данной вершине, данная точка уже не будет рассматриваться как точка пересечения.

Как и в алгоритме Кируса—Бека, точки пересечения отрезка с ребрами многоугольника, ограничивающего область видимости, будем вычислять посредством параметрического уравнения прямой отрезка. Воспользуемся тем свойством, что если $V(t)$ в параметрическом уравнении (5.8) является точкой пересечения, то в ноль обращается следующее псевдоскалярное произведение

$$(V(t) - F_i) \times F_iF_{i+1} = 0.$$

Распишем в последнем равенстве $V(t)$

$$(A + (B - A)t - F_i) \times F_iF_{i+1} = 0.$$

Перепишем левую часть равенства через сумму двух произведений и вынесем t за пределы произведения

$$(A - F_i) \times F_iF_{i+1} + t((B - A) \times F_iF_{i+1}) = 0.$$

Отсюда, значение t для точки пересечения несущей прямой отрезка AB и ребра F_iF_{i+1}:

$$t = \frac{(F_i - A) \times F_iF_{i+1}}{(B - A) \times F_iF_{i+1}} = \frac{AF_i \times F_iF_{i+1}}{AB \times F_iF_{i+1}}.$$

В общем виде метод Скала представлен в алгоритме 7.

Алгоритм 7: Алгоритм Скала

Вход: список S отрезков сцены.
Выход: S_1 — список видимых частей отрезков из S.
начало алгоритма
  Пусть F_i — все вершины области видимости, $1 \leqslant i \leqslant n$;
  цикл для каждого $2 \leqslant i \leqslant n$ **выполнить** $\overline{w}_i = F_i - F_{i-1}$;
  $\overline{w}_1 = F_1 - F_n$;
  $S_1 = \varnothing$;
  цикл пока S *не пуст* **выполнять**
    Взять из S отрезок AB;
    Присвоить $i = 1$; $k = 0$; $specialCase = True$; $\bar{v}_0 = F_n - A$; $\xi = AB \times \bar{v}_0$;
    цикл пока $i \leqslant n$ *и* $k \neq 2$ **выполнять**
      $\bar{v}_1 = F_i - A$; $\eta = AB \times \bar{v}_1$; $pointFound = False$;
      если $\xi\eta = 0$ **то**
        если $\xi \neq 0$ **то** $pointFound = True$;
        иначе если $\eta \neq 0$ *и* $specialCase$ **то** $pointFound = True$;
      если $\xi\eta < 0$ *или* $pointFound$ **то** $k = k + 1$; $t_k = (\bar{v}_1 \times \overline{w}_i)/(AB \times \overline{w}_i)$;
      если $\xi = 0$ *и* $\eta = 0$ **то** $specialCase = True$;
      иначе $specialCase = False$;
      $\xi = \eta$; $i = i + 1$;
    если $k = 2$ **то**
      если $t_1 > t_2$ **то** поменять местами значения t_1 и t_2;
      если $0 \leqslant t_1 \leqslant 1$ **то** $A' = A + (B - A)t_1$ **иначе** $A' = A$;
      если $0 \leqslant t_2 \leqslant 1$ **то** $B' = A + (B - A)t_2$ **иначе** $B' = B$;
      Поместить $A'B'$ в S_1;
  Выдать S_1;
конец алгоритма

5.1.5. FC-алгоритм

В 1987 г. Собков, Поспишил и Янг предложили алгоритм, названный ими FC-алгоритмом (Fast Clipping), в котором в отличие от алгоритма Коэна—Сазерленда кодируются не конечные точки отрезка, а отрезки целиком. Схема кодирования повторяет используемую в алгоритме Коэна—Сазерленда. На рисунке 5.10 представлен вариант кодирования областей (повторяющий вариант на рисунке 5.4), где каждый код области представлен одной шестнадцатеричной цифрой ($0x$ — префикс, показывающий, что последующее число записано в шестнадцатеричной форме). Код отрезка — комбинация кодов начала и конца отрезка. Так, на рисунке 5.5, код отрезка KL — $0x12$, отрезка GH — $0x82$, а отрезка IJ — $0x9A$.

Если вычислены коды концов отрезка C_1 и C_2, то код отрезка вычисляется по формуле:

$$C_1 * 16 + C_2$$

Так как каждый код может принимать одно из девяти значений, то всего имеется

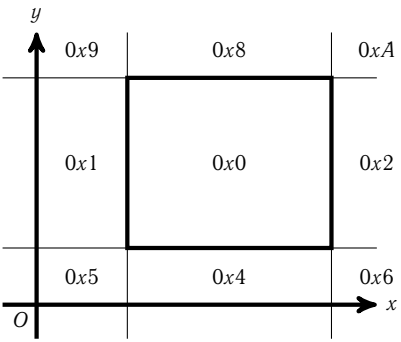


Рис. 5.10. Кодировка областей.

72 возможных варианта расположения отрезка (81 минус 9 случаев, при которых код начала отрезка равен коду конца отрезка).

Каждый код отрезка требует своего набора вычислений для определения отсечения отрезка за минимальное время. Всего имеется одиннадцать основных случаев отсечения, а остальные симметричны к ним. Рассмотрим эти случаи.

Будем использовать следующие обозначения.

- (x_1, y_1), (x_2, y_2) — координаты точки начала и конца отрезка соответственно;
- Clip1Left, Clip1Bottom обозначают алгоритмы переноса начальной точки отрезка в точку пересечения отрезка с левой или нижней границей соответственно;
- Clip2Right, Clip2Top обозначают алгоритмы переноса конечной точки отрезка в точку пересечения отрезка с правой или верхней границей соответственно.

Код отрезка $0x00$ (рис. 5.11, отрезок MN) Случай простого принятия отрезка;

Код отрезка $0x11$ (рис. 5.11, отрезок BL) Случай простого отбрасывания;

Код отрезка $0x19$ (рис. 5.11, отрезок AD) Случай простого отбрасывания;

Код отрезка $0x18$ (рис. 5.11, отрезки AE и BG) Отрезки точно пересекают левую границу области видимости, поэтому вначале надо выполнить Clip1Left, после чего проверить координату y_1. Для отрезка AE это дает $y_1 > y_{\max}$, так что отрезок должен быть отброшен без дальнейших вычислений. Для отрезка BG будет выполняться $y_1 < y_{\max}$, поэтому он входит в окно с левой стороны и,

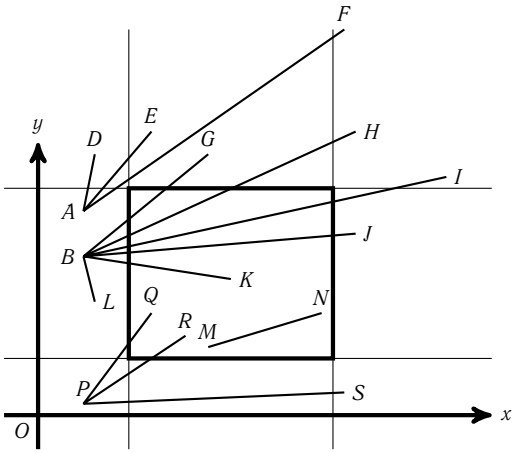


Рис. 5.11. Варианты расположения отрезков

следовательно, выходит через верхнюю границу. Следовательно, следующее отсечение должно быть Clip2Top, после которого отрезок принимается;

Код отрезка $0x1A$ (рис. 5.11, отрезки AF, BH и BI) Отрезки точно пересекают левую границу области видимости, поэтому вначале надо выполнить Clip1Left. После проверки координаты y_1 отрезок AF будет отброшен. Отрезки точно пересекают верхнюю границу области видимости. Поэтому следующее отсечение — Clip2Top, после которого проверяется координата x_2. Для отрезка BH это дает $x_2 < x_{\max}$, поэтому он принимается. Для отрезка BI будет выполняться $x_2 > x_{\max}$, поэтому он выходит из окна видимости с правой стороны и, следовательно, нужно выполнить Clip2Right, после чего отрезок принимается;

Код отрезка $0x10$ (рис. 5.11, отрезок BK) Отрезки точно пересекают левую границу области видимости, поэтому вначале надо выполнить Clip1Left, после чего отрезок принимается;

Код отрезка $0x12$ (рис. 5.11, отрезок BJ) Сначала надо выполнить Clip1Left, затем Clip2Right, после чего отрезок принимается;

Код отрезка $0x55$ (рис. 5.12, отрезок X_1Y) Случай простого отбрасывания;

Код отрезка $0x56$ (рис. 5.11, отрезок PS) Случай простого отбрасывания;

Код отрезка $0x50$ (рис. 5.11, отрезки PQ и PR) Вначале надо выполнить Clip1Left

и проверить значение y_1. Для отрезка PQ будет выполняться $y_1 > y_{\min}$, этот отрезок просто принимается. Для отрезка PR будет $y_1 < y_{\min}$. Для него нужно выполнить Clip1Bottom и принять отрезок;

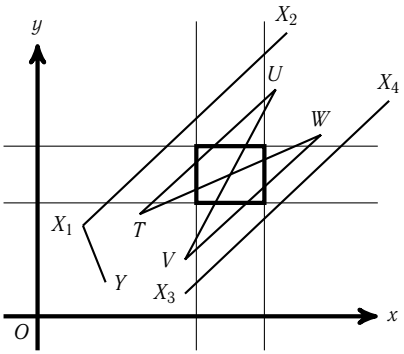


Рис. 5.12. Варианты расположения отрезков

Код отрезка $0x5A$ (рис. 5.12, все отрезки, кроме X_1Y) Вначале надо выполнить Clip1Left и проверить значение y_1. Для отрезка X_1X_2 будет выполняться $y_1 > y_{\max}$, этот отрезок отбрасывается. Для отрезков VU, V_W, X_3X_4 будет $y_1 < y_{\min}$. Для них нужно выполнить Clip1Bottom и проверить x_1. У отрезка X_3X_4 будет выполняться $x_1 > x_{\max}$, этот отрезок отбрасывается. Остальные отрезки должны быть видимы. Для них выполняем сначала Clip2Top и проверяем значение x_2. Для отрезков TU и VU будет выполняться $x_2 < X_{\max}$, они принимаются. Для остальных отрезков нужно выполнить Clip2Right, после чего принять.

Для остальных случаев достаточно либо продублировать условия обработки, либо провести предварительное преобразование поворота отрезка (вместе с окном отсечения) и/или его зеркального отражения для применения одного из вышеперечисленных случаев рассмотрения с последующим обратным преобразованием. Например для того, чтобы отсечь отрезок с кодом $0x4A$ можно повернуть систему координат на 90 градусов против часовой стрелки, после чего провести зеркальное отражение системы относительно оси Oy. В результате получим новые координаты концов отрезка

$$x'_1 = y_1; \qquad y'_1 = x_1;$$
$$x'_2 = y_2; \qquad y'_2 = x_2.$$

При этом область видимости изменит значения своих параметров:

$$x'_{\min} = y_{\min}; \qquad y'_{\min} = x_{\min};$$
$$x'_{\max} = y_{\min}; \qquad y'_{\max} = x_{\max}.$$

В новой системе координат отрезок будет иметь код $0x5A$ и его можно отсечь по уже предложенному алгоритму. После чего, если отрезок не будет отброшен, нужно сделать обратное преобразование для результата отсечения.

Подобную последовательность действий для сведения случая расположения отрезка к одному из перечисленных можно найти для каждого кода отрезка.

5.1.6. Алгоритм Николь—Ли—Николь

Алгоритм Николь—Ли—Николь подобен FC-алгоритму в том, что в нем перебираются всевозможные случаи расположения отрезков и области видимости, и для каждого случая определяется последовательность действий для отсечения. Причем для отсечения любого отрезка находятся максимум две точки пересечения.

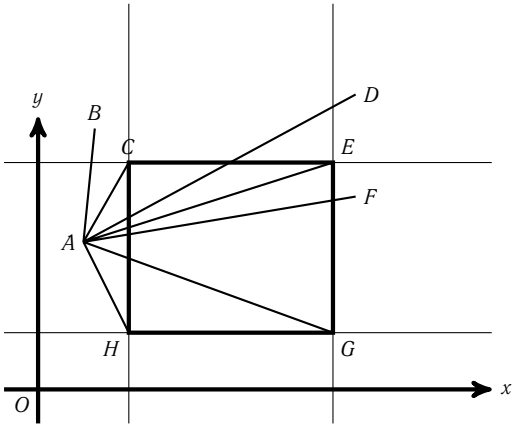


Рис. 5.13. Варианты расположения отрезков

Рассмотрим, например, точку A — начало отрезков, находящуюся левее области видимости (рис. 5.13). Отрезок, исходящий из точки A может быть частично видим, если угол его наклона находится в диапазоне от угла наклона отрезка AH до угла наклона отрезка AC. Точки H и C — соответственно точки левого нижнего и левого верхнего угла области видимости. Так, угол наклона отрезка AB больше, чем угол

наклона AC, то можно сделать вывод, что AB невидим и его отбросить. По аналогичным соображениям можем выявить, что так как угол наклона AD меньше чем угол наклона AC, но больше, чем угол наклона AE, и из-за того, что значение координаты x точки D больше $x_{\max}$, отрезок AD может выходить из области видимости только через верхнюю границу.

Вместо угла наклона целесообразнее сравнивать тангенс угла наклона $(y_2-y_1)/(x_2-x_1)$ отсекаемого отрезка и тангенс угла наклона отрезка, соединяющего точку начала отсекаемого отрезка с углом области видимости, $(y_a-y_1)/(x_a-x_1)$ (через (x_a, y_a) здесь обозначаем координаты некоторой угловой точки области видимости). Чтобы исключить необходимость деления, сравнение тангенсов угла наклона лучше привести к сравнению величин $(y_2-y_1)(x_a-x_1)$ и $(y_a-y_1)(x_2-x_1)$.

Так же как и в FC-алгоритме рассмотрим основные случаи работы алгоритма Николь—Ли—Николь.

Будем использовать следующие обозначения.

- (x_1, y_1), (x_2, y_2) — координаты точки начала и конца отрезка соответственно;
- $\Delta x = x_2 - x_1$, $\Delta y = y_2 - y_1$;
- $\Delta x_L = x_{\min} - x_1$, $\Delta x_R = x_{\max} - x_1$, $\Delta y_B = y_{\min} - y_1$, $\Delta y_T = y_{\max} - y_1$;
- Clip1Left, Clip1Bottom обозначают алгоритмы переноса начальной точки отрезка в точку пересечения отрезка с левой или нижней границей соответственно;
- Clip2Right, Clip2Top обозначают алгоритмы переноса конечной точки отрезка в точку пересечения отрезка с правой или верхней границей соответственно.

Пусть точка начала отрезка лежит не правее и не выше точки конца отрезка, т. е.

$$\Delta x \geqslant 0; \tag{5.12}$$
$$\Delta y \geqslant 0. \tag{5.13}$$

Тогда

если $x_2 < x_{\min}$, $x_1 > x_{\max}$ отрезок полностью невидим, случай простого отбрасывания.

если $y_2 < y_{\min}$, $y_1 > y_{\max}$ отрезок полностью невидим, случай простого отбрасывания.

если $x_1 \geqslant x_{\min}$, $x_2 \leqslant x_{\max}$, $y_1 \geqslant y_{\min}$, $y_2 \leqslant y_{\max}$ отрезок видим, случай простого принятия.

В остальных случаях отрезок полностью или частично лежит за пределами области видимости. Рассмотрим семейство случаев расположения отрезка, когда его начало лежит левее области видимости.

Пусть для точки начала отрезка выполняется $x_1 < x_{\min}$, $y_{\min} \leqslant y_1 \leqslant y_{\max}$ (см. рис. 5.14, точка A). Тогда, из (5.12) и (5.13) следует, что отрезок располагается в од-

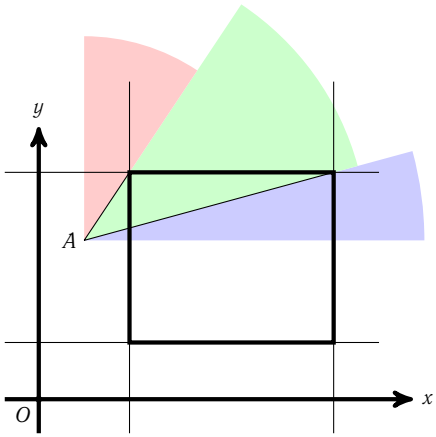


Рис. 5.14. Начальная точка и регионы расположения отсекаемого отрезка. Случай боковой области

ной из трех областей, отмеченных цветом на рисунке 5.14. Области разграничиваются прямыми, на которых лежат отрезки соединяющие точку (x_1, y_1) с точками $(x_{\min}, y_{\max})$ и $(x_{\max}, y_{\max})$ (см. рисунок).

На рисунке 5.15 изображены примеры расположения отрезков, начинающихся слева от области видимости.

Для того, чтобы определить область, в которую попадает отрезок, достаточно проверить:

если $\Delta y \Delta x_L > \Delta y_T \Delta x$ то отрезок попадает в розовую область: он полностью невидим (рис. 5.15, отрезок AB). Случай отбрасывания;

иначе, если $\Delta y \Delta x_R > \Delta y_T \Delta x$ то отрезок попадает в зеленую область: он частично видим (рис. 5.15, отрезки AD и AF). Он точно пересекает прямую, ограничивающую область видимости слева. Выполняем CLIP1LEFT. Если верно условие $y_2 \leqslant y_{\max}$, то полученный отрезок принимается (отрезок AF). В противном

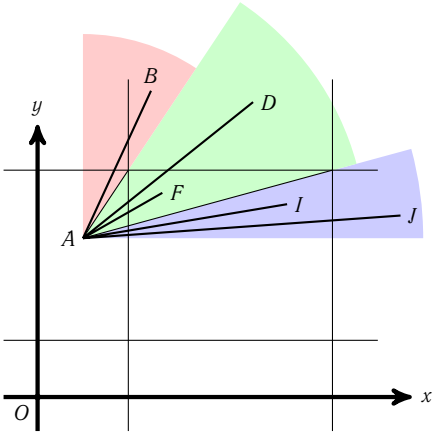


Рис. 5.15. Варианты расположения отсекаемых отрезков. Случай боковой области

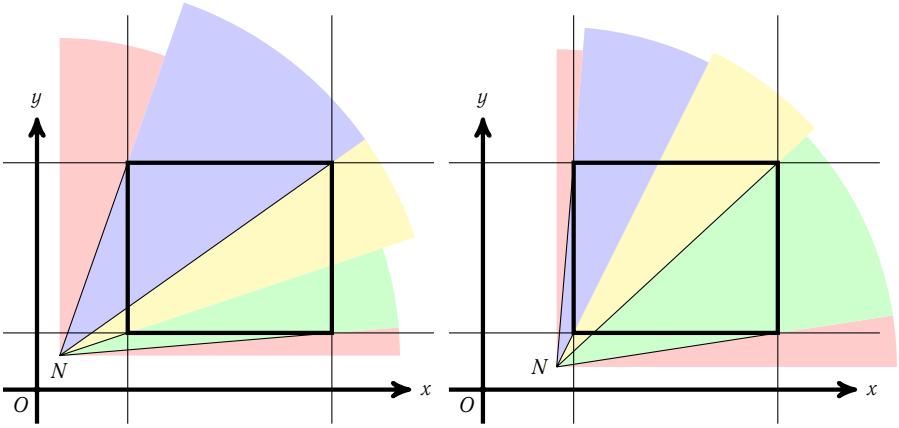
случае отрезок выходит из области видимости через верхнюю границу (отрезок AD). Выполняем Clip2Top и принимаем отрезок.

иначе отрезок попадает в голубую область: он частично видим (рис. 5.15, отрезки AI и AJ). Он точно пересекает прямую, ограничивающую область видимости слева. Выполняем Clip1Left. Если верно условие $x_2 \leqslant x_{\max}$, то полученный отрезок принимается (отрезок AI). В противном случае отрезок выходит из области видимости через правую границу (отрезок AJ). Выполняем Clip2Right и принимаем отрезок.

Если точка начала отрезка лежит в нижней левой угловой области, т. е. выполняется $x_1 < x_{\min}$, $y_1 < y_{\min}$ (см. рис. 5.16, точка N) то имеем пять регионов расположения отрезка, образованных, как и ранее, с помощью прямых соединяющих точку (x_1, y_1) с углами области видимости. Области, закрашенные на рисунке в розовый цвет, задают варианты расположения отрезка, при которых он отбрасывается. Если отрезок попадает в один из остальных регионов, то к нему нужно применить максимум два отсечения, чтобы он стал видим. Разграничение областей зависит от расположения начальной точки отрезка относительно прямой проходящей через точки $(x_{\min}, y_{\min})$ и $(x_{\max}, y_{\max})$. Это расположение можно определить, проверив неравенство

$$\Delta y_B \Delta x_R \leqslant \Delta y_T \Delta x_L.$$

Если неравенство выполняется, значит начальная точка отрезка лежит выше прямой и

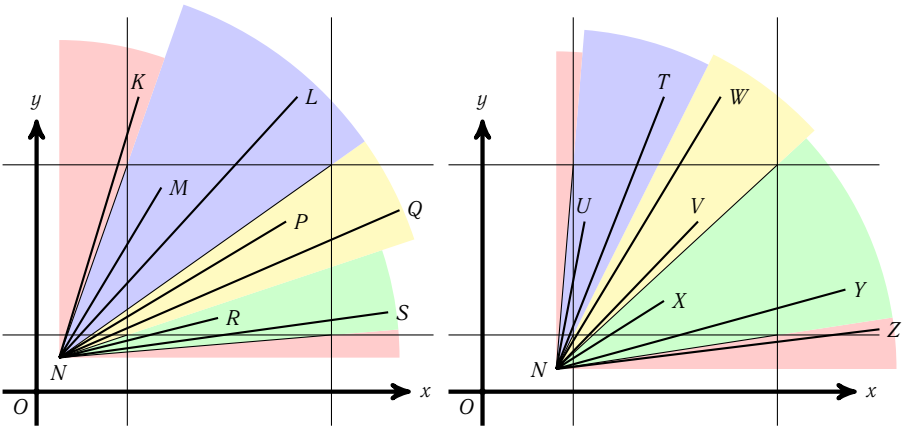


(*а*). $\Delta y_B \Delta x_R \leqslant \Delta y_T \Delta x_L$ (*б*). $\Delta y_B \Delta x_R > \Delta y_T \Delta x_L$

Рис. 5.16. Начальная точка и регионы расположения отсекаемого отрезка. Случай угловой области

имеет место случай, показанный на рисунке 5.16, *а*. В противном случае — начальная точка лежит ниже прямой и имеет место случай, показанный на рисунке 5.16, *б*.

На рисунке 5.17 приведены примеры расположения отрезков, начинающихся в нижней левой угловой области.



(*а*). $\Delta y_B \Delta x_R \leqslant \Delta y_T \Delta x_L$ (*б*). $\Delta y_B \Delta x_R > \Delta y_T \Delta x_L$

Рис. 5.17. Варианты расположения отсекаемых отрезков. Случай угловой области

Порядок отсечения отрезка может проводиться следующим образом:

если $\Delta y \Delta x_L > \Delta y_T \Delta x$ то отрезок попадает в левую розовую область: он полностью невидим (рис. 5.17, *а*, отрезок NK). Случай отбрасывания;

иначе, если $\Delta y \Delta x_R < \Delta y_B \Delta x$ то отрезок попадает в нижнюю розовую область: он полностью невидим (рис. 5.17, *б*, отрезок NZ). Случай отбрасывания;

иначе, если $\Delta y_B \Delta x_R \leqslant \Delta y_T \Delta x_L$

если $\Delta y \Delta x_R > \Delta y_T \Delta x$ то отрезок попадает в голубую область: он частично видим (рис. 5.17, *а*, отрезки NM и NL). Он точно пересекает прямую, ограничивающую область видимости слева. Выполняем Clip1Left. Если верно условие $y_2 \leqslant y_{\max}$, то полученный отрезок принимается (отрезок NM). В противном случае отрезок выходит из области видимости через верхнюю границу (отрезок NL). Выполняем Clip2Top и принимаем отрезок.

иначе, если $\Delta y \Delta x_L > \Delta y_B \Delta x$ отрезок попадает в желтую область: он частично видим (рис. 5.17, *а*, отрезки NP и NQ). Он точно пересекает прямую, ограничивающую область видимости слева. Выполняем Clip1Left. Если верно условие $x_2 \leqslant x_{\max}$, то полученный отрезок принимается (отрезок NP). В противном случае отрезок выходит из области видимости через правую границу (отрезок NQ). Выполняем Clip2Right и принимаем отрезок.

иначе отрезок попадает в зеленую область: он частично видим (рис. 5.17, *а*, отрезки NR и NS). Он точно пересекает прямую, ограничивающую область видимости снизу. Выполняем Clip1Bottom. Если верно условие $x_2 \leqslant x_{\max}$, то полученный отрезок принимается (отрезок NR). В противном случае отрезок выходит из области видимости через правую границу (отрезок NS). Выполняем Clip2Right и принимаем отрезок.

иначе (если $\Delta y_B \Delta x_R > \Delta y_T \Delta x_L$**)**

если $\Delta y \Delta x_L > \Delta y_B \Delta x$ то отрезок попадает в голубую область: он частично видим (рис. 5.17, *б*, отрезки NU и NT). Он точно пересекает прямую, ограничивающую область видимости слева. Выполняем Clip1Left. Если верно условие $y_2 \leqslant y_{\max}$, то полученный отрезок принимается (отрезок

NU). В противном случае отрезок выходит из области видимости через верхнюю границу (отрезок NL). Выполняем Clip2Top и принимаем отрезок.

иначе, если $\Delta y \Delta x_R > \Delta y_T \Delta x$ отрезок попадает в желтую область: он частично видим (рис. 5.17, *б*, отрезки NV и NW). Он точно пересекает прямую, ограничивающую область видимости снизу. Выполняем Clip1Bottom. Если верно условие $y_2 \leqslant y_{\max}$, то полученный отрезок принимается (отрезок NV). В противном случае отрезок выходит из области видимости через верхнюю границу (отрезок NW). Выполняем Clip2Top и принимаем отрезок.

иначе отрезок попадает в зеленую область: он частично видим (рис. 5.17, *б*, отрезки NX и NY). Он точно пересекает прямую, ограничивающую область видимости снизу. Выполняем Clip1Bottom. Если верно условие $x_2 \leqslant x_{\max}$, то полученный отрезок принимается (отрезок NX). В противном случае отрезок выходит из области видимости через правую границу (отрезок NY). Выполняем Clip2Right и принимаем отрезок.

Для остальных случаев расположение отсекаемого отрезка и области видимости можно составить подобные условия обработки, либо провести предварительный набор преобразований поворота и зеркального отображения отрезка (вместе с окном отсечения) для применения одного из вышеперечисленных случаев рассмотрения с последующим обратным преобразованием.