

УДК 681.142. 62

МАШИННОЕ ПОСТРОЕНИЕ ПРОЕКЦИЙ ТРЕХМЕРНЫХ ОБЪЕКТОВ С УДАЛЕНИЕМ НЕВИДИМЫХ ЛИНИЙ

В.М. Грин, В.А. Львов

Рассматриваются машинные методы построения изображений трехмерных объектов в виде параллельных (аксонометрических) и центральных (перспективных) проекций на плоскость, а также построение широкоугольной центральной проекции на поверхность кругового цилиндра (панорамная проекция). Преобразование координат, связанное с выбором типа проекции, т.е. изменение расстояния и ориентации объекта относительно центра проекции, и перспективное масштабирование выполняются в матричной форме с использованием однородных координат. Составлены компактные алгоритмы преобразования, реализованные в виде процедур на языке АЛГОЛ.

Описываются два алгоритма удаления невидимых линий: для совокупности произвольных многогранников и для графиков однозначных функций двух переменных. Приводятся тексты процедур и примеры работы программы.

1. Построение центральной проекции

Для получения центральной проекции преобразование координат точек удобно осуществлять в матричной форме [1-5]. При

этом используются матрица преобразования T порядка 4×4 и вместо обычных координат $(x; y; z)$ однородные координаты $(x; y; z; 1)$:

$$(x; y; z; 1) \cdot T = (u; v; w; t). \quad (1)$$

Переход от однородных координат $(u; v; w; t)$ к обычным декартовым $(x'; y'; z')$ производится по формулам [6]:

$$x' = \frac{u}{t}, \quad y' = \frac{v}{t}, \quad z' = \frac{w}{t}. \quad (2)$$

Особое значение имеет матрица, полученная из единичной матрицы заменой первого элемента четвертого столбца на число κ :

$$T_{\kappa} = \begin{pmatrix} 1 & 0 & 0 & \kappa \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

При умножении 4-мерного вектора точки на матрицу T_{κ} получим:

$$(x; y; z; 1) \cdot T_{\kappa} = (x; y; z; 1 + \kappa x). \quad (3)$$

Учитывая (2), устанавливаем, что преобразование (3) переводит бесконечно удаленную точку на оси OX $(1; 0; 0; 0)$ в точку $(1; 0; 0; \kappa)$, то есть в точку, лежащую на оси OX на расстоянии $1/\kappa$ от начала координат. После такого преобразования все линии, параллельные оси OX , пересекутся в точке $(1/\kappa; 0; 0; 1)$. Эту точку будем называть точкой схода перспективного преобразования.

Примем, что плоскостью проекции является плоскость $x = 0$. Тогда при исключении координаты X координаты Y и Z могут быть использованы как координаты ξ и η перспективной проекции точки на эту плоскость. Центром проекции является точка $S(-1/\kappa; 0; 0)$ на оси OX , расположенная симметрично точке схода относительно плоскости $x = 0$ (рис.1).

Если все элементы четвертого столбца матрицы отличны от нуля

$$T = \begin{pmatrix} 1 & 0 & 0 & \kappa \\ 0 & 1 & 0 & \ell \\ 0 & 0 & 1 & m \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

то координаты точки схода определяются как $(1/k; 1/l; 1/m)$, а центр проекции расположен симметрично точке схода относительно начала координат в точке $(-1/k; -1/l; -1/m)$. Такая матрица по-

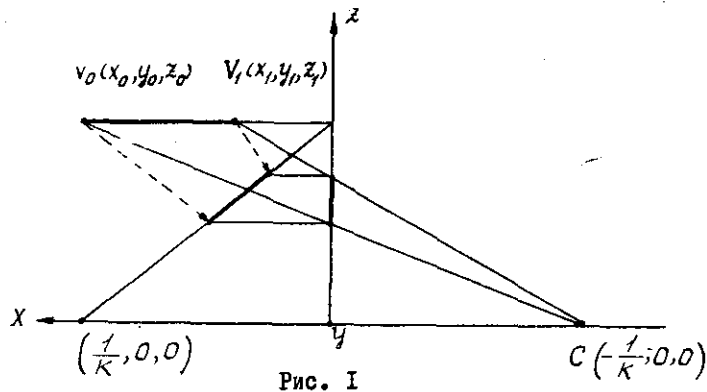


Рис. 1

зволяет строить перспективные проекции на плоскость, перпендикулярную направлению зрения.

Кроме перспективных преобразований, матрица T используется для масштабирования, вращения вокруг координатных осей, параллельного переноса. По своему влиянию элементы матрицы группируются следующим образом:

масштабирование, сдвиг, вращение	$\begin{pmatrix} a & b & c & k \\ d & e & f & l \\ g & h & i & m \\ dx & dy & dz & t \end{pmatrix}$	перспективное преобразование
перенос	$\begin{pmatrix} dx & dy & dz & t \end{pmatrix}$	общее масштабирование

Процесс формирования матрицы T состоит из последовательности шагов, соответствующих последовательности элементарных преобразований (масштабирование, вращение, параллельный перенос и т.д.). Порядок выполнения элементарных преобразований не безразличен, так как умножение матриц в общем случае некоммутативно.

В частном случае ($k=l=m=0$) преобразование с помощью матрицы T позволяет получить любую ортогональную аксонометрию

ческую проекцию объекта. Для этого необходимо предварительно выполнить элементарные преобразования вращения вокруг координатных осей на углы α , β и γ (для изометрии, например, $\gamma_u = -45^\circ$, $\beta_u = 35^\circ$, $\alpha_u = 0$).

По изложенной методике разработан комплекс программ (оформленных в виде процедур на языке АЛГОЛ - см. табл. I) для различных элементарных преобразований. Каждое элементарное преобразование изменяет только матрицу T [1:4, 1:4], не затрагивая массива исходных данных - координат точек трехмерного объекта, записанного в виде массива A [1: n , 1:3], где n - количество

точек, $A[i, 1]$, $A[i, 2]$ и $A[i, 3]$ - координаты x , y и z соответственно для i -й точки.

Рассмотрим пример программы построения центральной проекции единичного куба. Центр куба находится в начале координат, а наблюдатель смотрит на куб сверху под углом 30° к горизонту и сбоку под углом 30° к профильной плоскости (XOZ). Расстояние наблюдателя от центра куба равно 4.

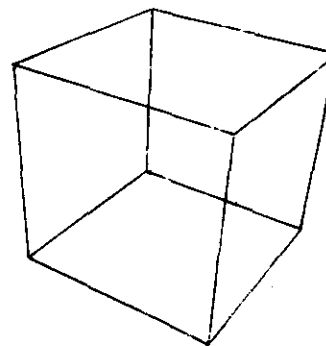


Рис. 2

из Приложения описания процедур `init`, `rotate`, `project`
`trans`.

```
array T [1:4, 1:4], A[1:16, 1:3];
input (1, A);
init (T);
rotate (T, 3, 30);
rotate (T, 2, 30);
project (T, 4);
trans (A, 16, T);
end;
```

Результат работы программы показан на рис. 2.

Т а б л и ц а I

Процедуры для построения центральных проекций

№ пп	Название	Назначение
1.	init (T)	Формирование единичной матрицы T.
2.	scale (T, sx, sy, sz)	Масштабирование объекта.
3.	rotate (T, i, teta)	Выполнение вращения вокруг одной из координатных осей ($i=1$ для оси OX , $i=2$ для оси OY , $i=3$ для оси OZ).
4.	translate (T, dx, dy, dz)	Параллельный перенос вдоль координатных осей.
5.	project (T, r)	Получение центральной проекции из точки оси OX с координатами $(z; 0; 0)$ на плоскость YOZ .
6.	from xyz (T, x, y, z)	Получение центральной проекции с центром проекции в точке с координатами $(x; y; z)$ на плоскость, перпендикулярную прямой, проходящей через эту точку и начало координат.
7.	ortiso (T)	Получение ортогональной изометрической проекции.
8.	dimetr (T)	Получение диметрической проекции.
9.	return (T, xo, zo)	"Возвращение" начала координат после ряда преобразований в заданную точку.
10.	trans (A, n, T)	Преобразование n координат объекта, заданного массивом A, с помощью матрицы преобразования T.
II.	matmul (m, n, k, A, B, C)	Умножение матрицы A порядка $m \times n$ на матрицу B порядка $n \times k$, результатом является матрица C порядка $m \times k$.

ПРИМЕЧАНИЕ. Полные тексты всех процедур помещены в приложении, стр. 82.

2. Построение панорамной проекции

Установим соотношение между координатами точки в пространстве $(x; y; z)$ и координатами ее проекции на панораме $(\xi; \eta)$.

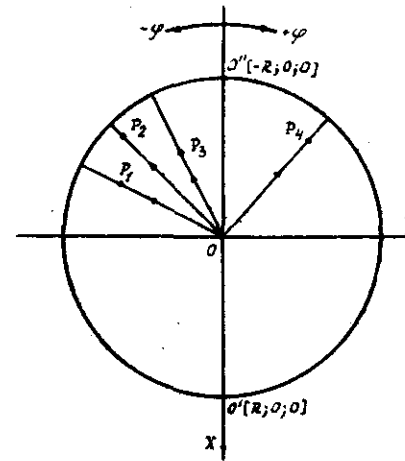


Рис. 3

Ось η будет служить образующая цилиндра, проходящая через точку $(-R; 0; 0)$ и направленная в ту же сторону, что и ось OZ . Координата ξ отсчитывается вдоль окружности цилиндра и пропорциональна углу. Вращение вправо от точки O'' до O' считается положительным (рис. 3), влево от точки O'' до O' — отрицательным.

Из построений (рис. 4) получим формулы для ξ и η :

$$\left. \begin{aligned} \xi &= R \cdot \varphi, \\ \eta &= \frac{R \cdot z}{\sqrt{x^2 + y^2}} \end{aligned} \right\} \quad (4)$$

где

$$\varphi = \begin{cases} -\arctg(y/x) & \text{при } x < 0, \\ -\arctg(y/x) + \pi & \text{при } x > 0 \text{ и } y > 0, \\ -\arctg(y/x) - \pi & \text{при } x > 0 \text{ и } y < 0 \end{cases} \quad (5)$$

Формулы (4) и (5) позволяют построить панорамную проекцию любой точки, заданной своими координатами, кроме точек на оси цилиндра.

Прямая на панорамной проекции всегда изображается отрезком синусоиды. Уравнение панорамной проекции прямой, проходящей через точки $V_0(x_0, y_0, z_0)$ и $V_1(x_1, y_1, z_1)$, может быть записано в виде:

(6)



φ, φ_0 - углы проециций векторов точек V_i и V_0 , полученные по формуле (5);

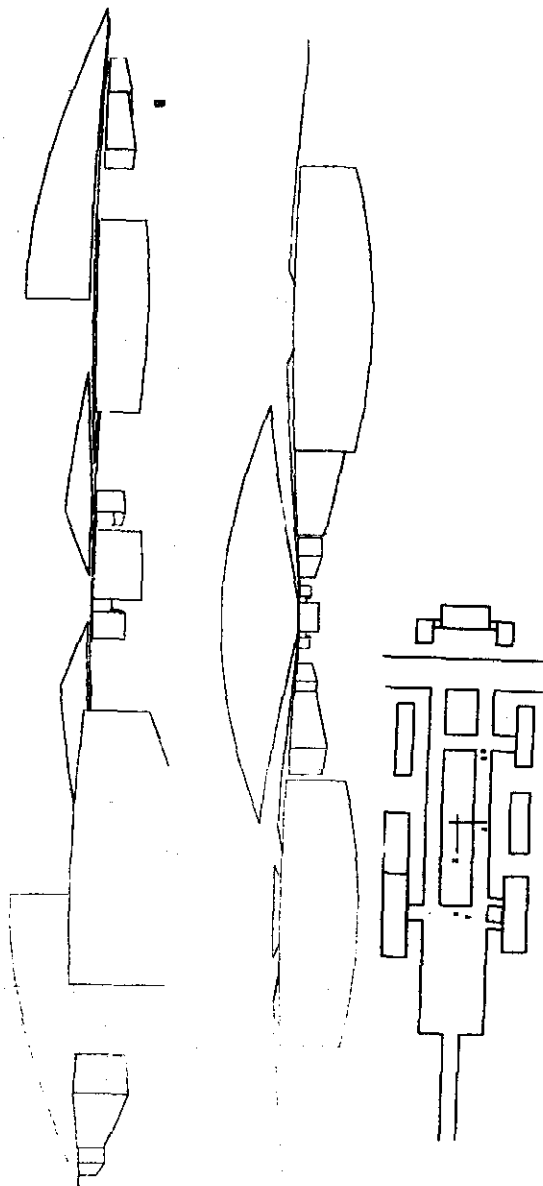
$$\text{sign}(\varphi_i - \varphi_0) = \begin{cases} 1 & \text{при } \varphi_i \geq \varphi_0, \\ -1 & \text{при } \varphi_i < \varphi_0; \end{cases}$$

$$q = a \cdot \sin \alpha + b \cdot \cos \varphi_0 ;$$

$$w = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2}$$

Рис. 5

Панорамная проекция может применяться при проектировании архитектурных комплексов, позволяя получить точное изображение вида из любой точки зрения. На рис. 6 изображены две панорамные проекции группы трехмерных объектов. Центры проекций отмечены на плане точками А и В.



3. Удаление невидимых линий при построении проекций

Наглядность изображений проекции трехмерных объектов на плоскости повышается при удалении так называемых "невидимых линий", то есть закрытых для наблюдателя другими частями объекта. Известно несколько алгоритмов [7-12] для ряда частных случаев с существенными ограничениями на форму изображаемого объекта, структуру данных, используемых для его задания, скорость выполнения операций и объем памяти ЭВМ. Анализ работ [7-12] позволил составить обобщенный алгоритм определения невидимости линий для широкого класса трехмерных объектов, представляющих собой замкнутые многогранники. Объекты считаются непрозрачными и могут быть как выпуклыми, так и вогнутыми. Линии пересечения каждой грани с соседними гранями (ребра) образуют замкнутый контур, ограничивающий в общем случае многосвязную область. Ребра пересекаются в вершинах.

Отдельная точка считается видимой только в том случае, если проектирующий луч из центра проекции в данную точку не пересекает ни одной грани. Отрезок прямой считается видимым, если все его точки видны. Изображение строится из проекций видимых отрезков ребер с точным определением границ перехода от видимой части к невидимой.

Для работы алгоритма удаления невидимых линий каждый объект должен быть задан:

1) списком вершин $V_i(x_i; y_i; z_i)$, где индекс i является номером вершины;

2) списком всех граней P_j , где каждая грань задается списком номеров вершин $i(i, u)$, встречающихся при обходе контура, ограничиваемого гранью с номером j . Обход производится по внешней по отношению к объекту стороне грани; направление обхода выбирается таким образом, чтобы ограниченная контуром область всегда оставалась слева. Индекс $u = 1, 2, \dots, n_j$ является порядковым номером вершины при обходе контура грани P_j .

Алгоритм позволяет строить проекции и определять невидимые линии как для отдельных объектов, так и для совокупности объектов. В результате работы алгоритма определяются видимые отрезки проекций ребер всех рассматриваемых объектов.

Количественной характеристикой видимости θ точки будем называть число граней, заслоняющих данную точку от воображаемого наблюдателя, то есть пересекающих отрезок, проведенный из центра проекции в данную точку. Для определения θ необходимо просмотреть полный список граней совокупности объектов и увеличивать θ на единицу всякий раз при выполнении следующих условий:

а) прямая, соединяющая данную точку с центром проекции, пересекает исследуемую грань;

б) точка пересечения этой прямой с гранью находится ближе к центру проекции, чем данная точка.

Назовем грань условно видимой (JV), если внешняя по отношению к объекту нормаль к плоскости грани образует с вектором, проведенным из центра проекции в основание этой нормали, угол больше 90° , и условно невидимой (JH) — в противном случае. Ребро, соединяющее вершины z и s , будем обозначать (z, s) . Условимся называть ребро выпуклым, если образующие его грани составляют выпуклый угол, и вогнутым, если грани составляют вогнутый двугранный угол.

Если $\vec{V}_{i(j,u)}$ — векторы вершин грани с номером j , то направление внешней нормали к плоскости грани, совпадает с направлением вектора площади:

$$\vec{A}_j = \frac{1}{2} \sum_{u=2}^{n_j} (\vec{V}_{i(j,u)} - \vec{V}_{i(j,1)}) \times (\vec{V}_{i(j,u+1)} - \vec{V}_{i(j,1)}). \quad (7)$$

Для определения выпуклости или вогнутости ребра, соединяющего вершины z и s и образованного при пересечении граней P_m и P_n , достаточно вычислить скалярное произведение

$$C = (\vec{V}_s - \vec{V}_z) \cdot (\vec{A}_m \times \vec{A}_n) \quad (8)$$

Тогда, если $C > 0$, то ребро выпукло, в противном случае — вогнуто. Последнее справедливо, если плоскость P_m находится "слева" при движении из вершины z в вершину s .

При построении нескольких различных проекций объекта выпуклость или вогнутость ребер определяется только один раз, так как эта характеристика не зависит от расположения и ориентации объекта в пространстве.

Из множества ребер выделим подмножество контурных линий, то есть ребер, образованных пересечением одной условно видимой и одной условно невидимой плоскости. Эти ребра представляют специфический интерес, так как при движении вдоль некоторой линии количественная характеристика видимости изменяется только в точках пересечения проекции исследуемой линии с проекциями контурных линий, причем только в тех случаях, когда исследуемая линия проходит за данной контурной линией (рис. 7). Как видно из рисунка, количественная характеристика

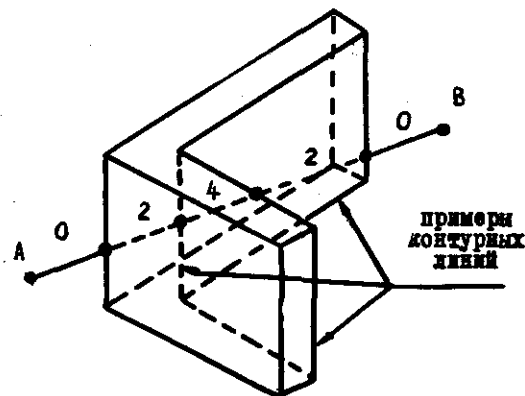


Рис. 7

видимости увеличивается на 2, когда исследуемая линия заходит за двугранный угол, образующий контурную линию, и уменьшается на 2, когда линия выходит из — за двугранного угла.

Для определения изменения характеристики видимости необходимо знать условную видимость грани, расположенной слева при движении вдоль контурной линии, и знак момента контурной линии относительно начальной вершины исследуемой линии. При этом считается, что вектор контурной линии направлен от вершин с меньшим номером к вершине с большим номером (рис. 8). Возможные случаи изменения характеристики видимости сведены в табл. 2.

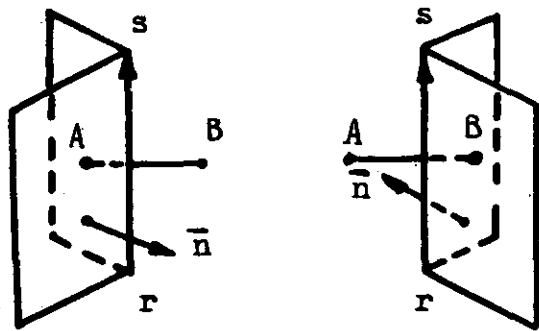


Рис. 8

Т а б л и ц а 2

Знак момента	Условная видимость грани	Изменение характеристики видимости
+	УВ	+2
-	УВ	-2
+	УН	-2
-	УН	+2

Перечислим основные шаги алгоритма:

Шаг 1. Определяем выпуклость или вогнутость всех ребер.

Шаг 2. Удаляем проекции полностью невидимых ребер, а именно любые ребра, принадлежащие двум условно невидимым граням, либо вогнутые ребра, принадлежащие одной условно видимой и одной условно невидимой граням. Если рассматривается отдельный выпуклый объект, на этом работа алгоритма заканчивается.

Шаг 3. Исследуем ребра, не удаленные на втором шаге. Определяем количественную характеристику видимости (θ) начальной вершины каждого ребра, затем вычисляем ее изменение при пересечении проекции ребра с проекциями контурных линий; начальную и конечную точки проекции той части исследуемого ребра, вдоль которой $\theta = 0$, заносим в специальный "массив видимых

отрезков". Последний может быть использован для получения изображения на любом устройстве вывода графической информации. В результате анализа пересечений проекции ребра с проекциями контурных линий будет известна характеристика видимости конечной точки ребра; следовательно, непосредственное определение θ необходимо лишь для небольшого числа вершин.

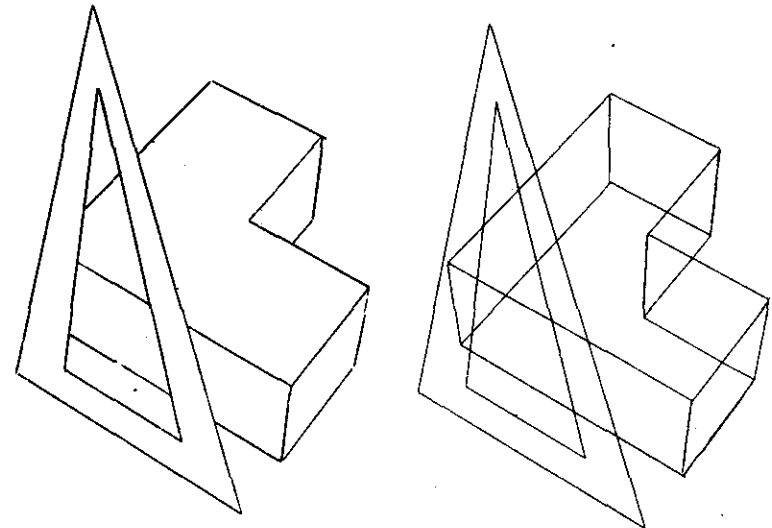


Рис. 9

Алгоритм записан на АЛГОЛе, рабочая программа получена для ЭВМ "Минск-22". На рис. 9-10 приведены примеры работы программы.

Описания трехмерных объектов подготавливаются и вводятся либо как отдельные массивы данных (микроописание), либо с помощью специальных программ-генераторов (макроописание). Возможны три режима работы программы для вычерчивания проекций:

- 1) без удаления невидимых линий;
- 2) с полным удалением невидимых линий;
- 3) с проведением невидимых линий штрихами.

По быстродействию алгоритм вполне пригоден для практического использования.

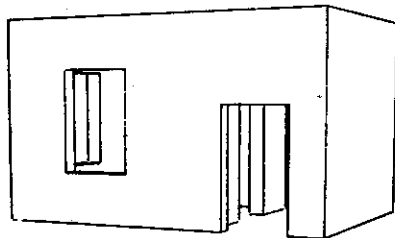


Рис. 10

Несколько проще может быть решен вопрос удаления невидимых линий при вычерчивании графика функции двух переменных $z = f(x, y)$ в виде проекций сечений плоскостями $x = \text{const}$ или $y = \text{const}$. Для примера на рис. 11 приведены два изображения одной и той же функции: α - без удаления "невидимых" линий, δ - с удалением.

В [11] рассматривается такой алгоритм удаления невидимых линий для однозначной непрерывной функции $f(x, y)$, значения которой заданы в узлах прямоугольной сетки, покрывающей область $R = \{(x, y): x_1 \leq x \leq x_2, y_1 \leq y \leq y_2\}$ с постоянным шагом по координатам X и Y . При работе программы, составленной по этому алгоритму, весь массив значений функции f должен храниться в оперативной памяти ЭВМ. Это не позволяет реализовать алгоритм на вычислительных машинах с малым объемом памяти (до 8 К).

Для поверхностей того же класса предлагается алгоритм, не использующий существенно меньший объем оперативной памяти за счет последовательного про-

смотра сечений. Сечения поверхности $z = f(x, y)$ могут быть заданы с неравномерным шагом.

Пусть заданы не пространственные координаты точек сечения (x_i, y_i, z_i) , а их проекции на плоскость рисунка (ξ_i, η_i) , полученные в результате выполнения некоторого преобразования. Тогда часть линии κ -го сечения, расположенная "выше" или "ниже" (по координате η) предыдущих $(\kappa - 1)$ сечений, будет видима. Все остальные участки линии будут невидимы (рис. 12).

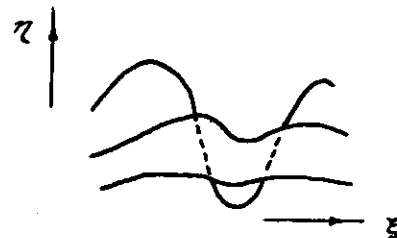


Рис. 12

При реализации этого метода были введены понятия верхней и нижней границ видимости, которые корректируются после анализа каждого нового сечения, начиная с самых близких сечений к предполагаемой точке зрения и далее вглубь. Границы видимости представляют собой массивы чисел, задающих координаты верхней и нижней границ на отрезке $[\xi_{min}, \xi_{max}]$ с шагом h вдоль оси абсцисс. Величина шага h определяется требованиями к точности выполнения чертежа в точках переходов к невидимой области и разрешающей способностью вычерчивающего устройства, например, конкретное значение h может иметь порядок толщины линии. Кроме того, ввиду достаточно малой величины шага h анализ относительного расположения линий сечения и границ видимости производится не для всех значений координат ξ , а только для точек, кратных по величине шагу h ; это вносит некоторую погрешность, но увеличивает скорость работы программы.

Правило определения видимости для любой точки $P(x_m, y_m)$ формулируется следующим образом:
точка M не видима, если

где g'_κ и g''_κ - значения нижней и верхней границ соответственно в точке, ближайшей слева от точки M с абсциссой $(\xi_{min} +$

+ $\kappa \cdot h$);

точка M видима, если

$$y_m < g'_k \quad \text{или} \quad y_m > g''_k$$

После вычерчивания кривой рассматриваемого сечения участками кривой, находящимися выше верхней и ниже нижней границ, заменяются соответствующие участки старых границ для следующего сечения. Перед просмотром первого сечения в качестве верхней границы выбирается значение $\eta_{\min}$, а нижней — $\eta_{\max}$. Алгоритм просмотра иллюстрируется рис. 13.

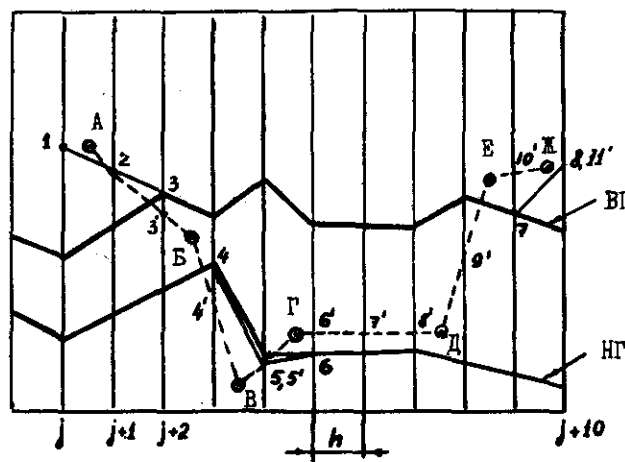


Рис. 13

Пусть ломаная ВГ задает часть верхней границы видимости, а ломаная НГ — часть нижней границы видимости, полученных после просмотра ($i-1$) сечений. Ломаная АБВГДЕЖ задает i -е сечение.

Вместо точки А возьмем точку 1 и будем двигаться вправо с шагом h . Точка 2 видима, поэтому соединим ее отрезком с точкой 1. На следующем шаге точка 3' невидима, проведем отрезок из точки 2 в точку 3. Точка 4' видима, однако начнем линию на границе в точке 4. Далее получим отрезки 4-5 и 5-6. Точки 7', 8', 9' невидимы, поэтому продолжение линии начнем в точке 7,

расположенной на границе. Поступая аналогично, можно получить изображение с удаленными невидимыми линиями с достаточно высокой точностью, определяемой лишь величиной шага h . Алгоритм оформлен в виде процедуры на языке АЛГОЛ. Программа использована для построения изображений на рис. 11.

Л и т е р а т у р а

1. AHUJA D.V., COONS S.A. Geometry for construction and display. IBM Systems Journal, 1968, vol.7, N 3-4.
2. PAUL R., FAIK G., FELDMAN J.D. The Computer Representation of Simply Described Scenes. Pertinent Concepts in Computer Graphics, ed. M. Faiman, J. Nievergelt, University of Illinois, 1969, p. 87-103.
3. BARRY C.D., ELLIS R.A., GRAESSER S.M., MARSHALL G.R. Display and Manipulation in Three Dimensions, ibid, p. 104-153.
4. FORREST A.R. Coordinates, Transformations and Visualization Techniques. Cambridge CAD Group, Document N 23, June 1969.
5. GREATORREX F.S., DAN COHEN. Producing Dynamic Perspective Views for Vehicle Simulation. Data Processing, April, 1968.
6. МОДЕНОВ П.С., ПАРХОМЕНКО А.С. Геометрические преобразования, изд. Московского университета, 1961.
7. WARNOCK I.E. The Hidden Line Problem and the Use of Halftone Displays. Pertinent Concepts in Computer Graphics, ed. Faiman M., Nievergelt I., Univ. of Illinois, 1969.
8. BOUKNIGHT W.T. A procedure for Generation of Three - Dimensional Half-Toned Computer Graphics Presentations. Comm. of ACM, 1970, vol.13, N 9.
9. GALIMBERTI R., MONTANARI V. An Algorithm for Hidden Line Elimination. Comm. of ACM, 1969, vol.12, N 4.
10. APPEL A. The Notion of Quantitative Invisibility and the Machine Rendering of Solids. Proceedings of the ACM 22nd National Conference, 1967, Thompson Book Co., Washington.
11. KUBERT B., SZABO T., GIULIERI S. The Perspective Representation of Functions of two Variables. J. of ACM, vol.15, N 2, April, 1968.
12. SPROULL F.F., SUTHERLAND I.E. A Clipping Divider. Proc. of the AFIPS FJCC, 1968, Thompson Book Co.

Поступила в ред.-изд. отд.

11. УП. 1971 г.

Приложение

1. Процедура init (T)

В результате работы процедуры в массиве T [1:4, 1:4] будет записана единичная матрица.

```
procedure init (T);
array T;
begin integer i, j;
  for i:= 1 step 1 until 4 do
    for j:= 1 step 1 until 4 do T[i, j]:= 0;
  for i:= 1 step 1 until 4 do T[i, i]:= 1.0;
end init;
```

2. Процедура scale (T, SX, SY, SZ).

SX, SY, SZ - масштабные коэффициенты по осям OX, OY и OZ, соответственно.

```
procedure scale (T, SX, SY, SZ);
value SX, SY, SZ; real SX, SY, SZ; array T;
begin array R[1:4, 1:4];
  init (R);
  R[1, 1] := SX; R[2, 2] := SY; R[3, 3] := SZ;
  matmul (4, 4, 4, T, R, T);
```

end scale;

3. Процедура rotate (T, i, teta).

Вращение вокруг оси с номером i (i = 1 - ось OX, i = 2 - ось OY, i = 3 - ось OZ) на угол teta (в градусах).

```
procedure rotate (T, i, teta);
value i, teta; real teta; integer i; array T;
begin real a, b, d;
  integer j, k, h; array R[1:4, 1:4];
  init (R); teta:= teta * 3.14159 / 180.0;
  a:= cos (teta); b:= sin (teta);
  if i = 1 then
    begin j:= 2; k:= 2; h:= 0; d:= 1.0;
  end else if i = 2 then
    begin j:= 1; k:= 1; h:= 1; d:= -1;
  end else
    begin j:= 1; k:= 1; h:= 0; d:= 1.0;
  end ;
```

```
R[j, k] := a; R[j+h+1, k+h+1] := a;
R[j, k+h+1] := d * b; R[j+h+1, k] := -d * b;
matmul (4, 4, 4, T, R, T);
```

end rotate;

4. Процедура translate (T, dx, dy, dz);

Параллельный перенос вдоль осей OX, OY, OZ на расстояния dx, dy и dz, соответственно.

```
procedure translate (T, dx, dy, dz);
value dx, dy, dz; real dx, dy, dz; array T;
begin array R[1:4, 1:4];
  init (R);
  R[4, 1] := dx; R[4, 2] := dy; R[4, 3] := dz;
  matmul (4, 4, 4, T, R, T);
```

end translate;

5. Процедура project (T, r)

Центральное проектирование из точки на оси OX с координатами (r, 0, 0).

```
procedure project (T, r);
value r; real r; array T;
begin array P[1:4, 1:4];
  init (P);
  P[1, 4] := -1/r;
  matmul (4, 4, 4, T, P, T);
```

end project;

6. Процедура fromxyz (T, x, y, z).

Центральное проектирование из точки с координатами (x, y, z) на плоскость, перпендикулярную прямой, соединяющей точки (0, 0, 0) и (x, y, z). Плоскость проекции проходит через точку (0, 0, 0).

```
procedure fromxyz (T, x, y, z);
value x, y, z; real x, y, z; array T;
```

```

begin real a,b,c;
  if x<0 then c:=-1.0 else c:=1.0;
  a:=-arctg (c * y / (abs(x) + 0.0001));
  if x<0 then a:= a + 3.14159;
  b:= arctg (x/(sqrt(x2 + y2) + 0.0001));
  rotate ( T, 3, a * 180.0/3.14159);
  rotate ( T, 2, b * 180.0/3.14159);
  project ( T, sqrt(x2 + y2 + z2 ));

```

end fromxyz;

7. Процедура ortiso (T).

Формируется матрица для получения ортогональной изометрической проекции объекта.

```

procedure ortiso (T);
array T;
begin array R[1:4, 1:4];
  init (R);
  R[1,1]:=0.577; R[1,2]:=-0.707;
  R[1,3]:=-0.408; R[2,1]:=0.577;
  R[2,2]:=0.707; R[2,3]:=-0.408;
  R[3,1]:=-0.577; R[3,3]:=0.817;
  matmul (4, 4, 4, T, R, T);

```

end ortiso;

8. Процедура dimetr (T).

Формируется матрица для получения диметрической проекции объекта.

```

procedure dimetr (T);
array T;
begin array R[1:4, 1:4];
  init (R);
  R[1,1]:=0.354; R[1,2]:=-0.926;
  R[1,3]:=-0.134; R[2,1]:=0.866;
  R[2,2]:=0.378; R[2,3]:=-0.327;
  R[3,1]:=-0.354; R[3,3]:=0.935;
  matmul (4, 4, 4, T, R, T);

```

end dimetr;

9. Процедура return (T, y0, z0).

После ряда преобразований производится "обратный" параллельный перенос объекта вдоль осей OY и OZ, так что точка с координатами (x, 0, 0) переходит в точку (x, y0, z0). Введение этой процедуры связано с тем, что положение области индикации на плоскости YOZ фиксировано.

```

procedure return (T, y0, z0);
value y0, z0; real y0, z0; array T;
begin array R[1:4, 1:4];
  init (R);
  R[4,2]:=(y0-T[4,2])/T[4,4];
  R[4,3]:=(z0-T[4,3])/T[4,4];
  matmul (4, 4, 4, T, R, T);

```

end return;

10. Процедура trans (A, N, T).

Производит преобразование объекта, заданного массивом A, с помощью матрицы преобразования T.

```

procedure trans (A, N, T);
value N; integer N; array A, T;
begin integer i, j;
  array R[1:1, 1:4];
  for i:=1 step 1 until N do
    begin for j:=1 step 1 until 3 do
      R[i,j]:=A[i,j]; R[i,4]:=1.0;
      matmul (1, 4, 4, R, T, R);
      for j:=1 step 1 until 3 do
        A[i,j]:=R[i,j]/R[i,4];

```

end;

end trans;