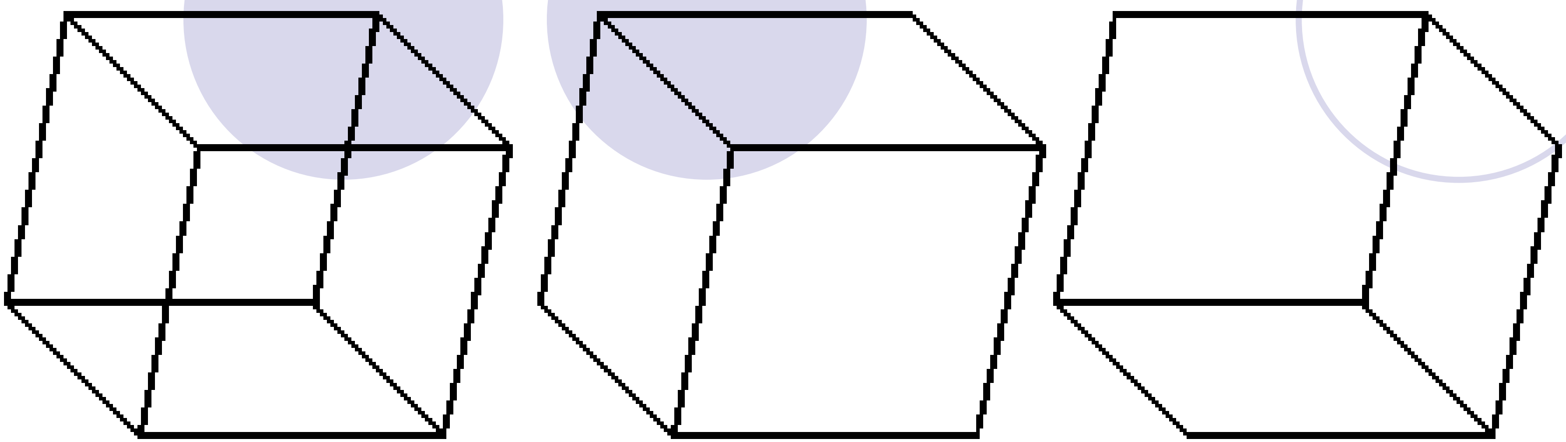
УДАЛЕНИЕ НЕВИДИМЫХ ЛИНИЙ И ПОВЕРХНОСТЕЙ

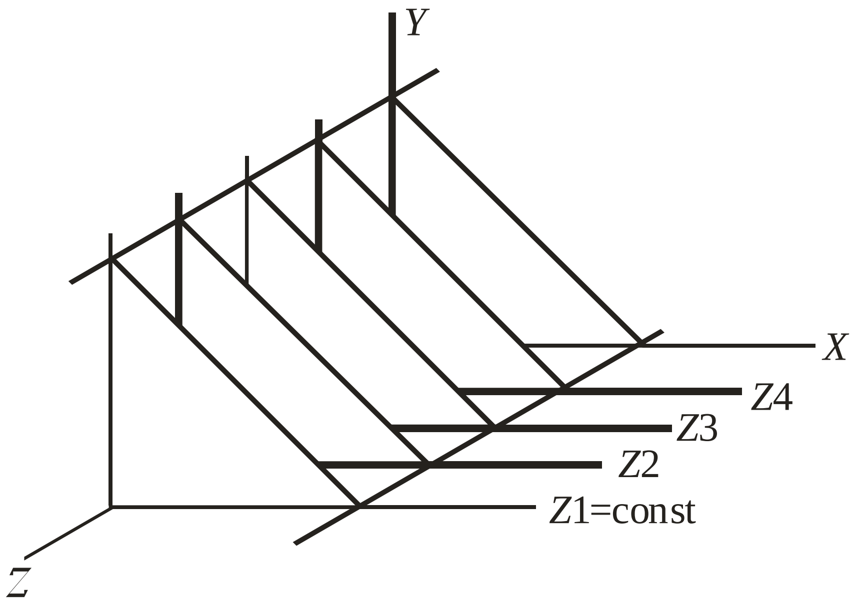


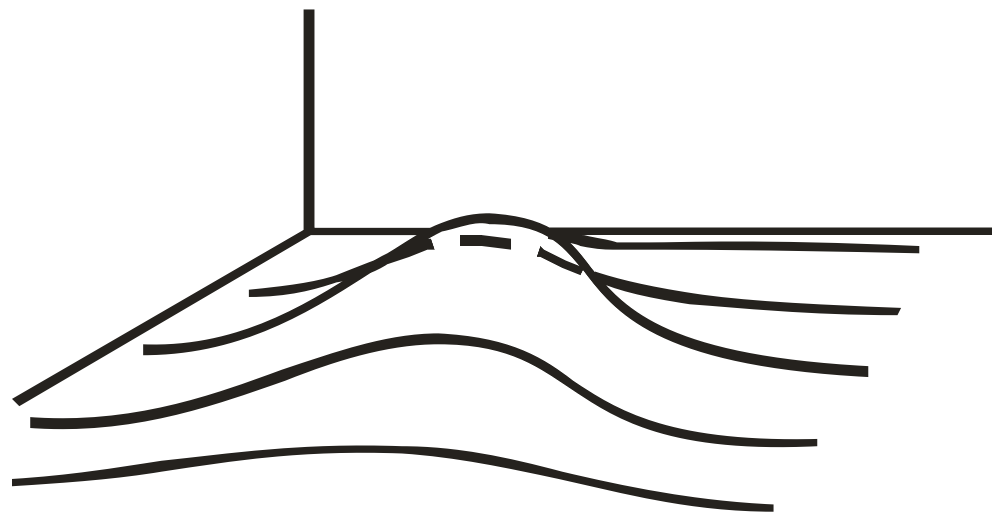
Классификация алгоритмов удаления

- Алгоритмы, работающие в объектном пространстве. **(Сложность n^2)** *Пример: алгоритм Робертса.*
- Алгоритмы, работающие в пространстве изображения или экрана **(Сложность nN).** *Пример: Z буффер.*
- Алгоритмы, формирующие список приоритетов. *Пример: Алгоритм Художника*

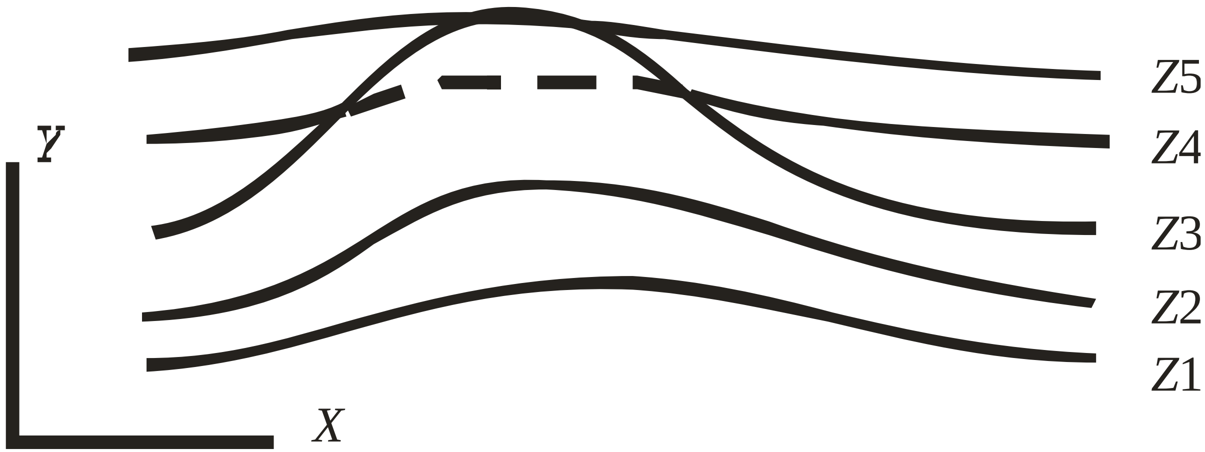
Алгоритм плавающего горизонта

- Алгоритм используется для построений поверхностей:
- *F(x, y, z)* = 0
- Главная идея данного метода заключается в сведении трехмерной задачи к двумерной путем пересечения исходной поверхности последовательностью параллельных секущих плоскостей, имеющих постоянные значения координат *x*, *y* или *z*. В результате получается набор кривых в пространстве.
- На следующем этапе кривые проецирует на КП





Проекция кривых на плоскость z = 0



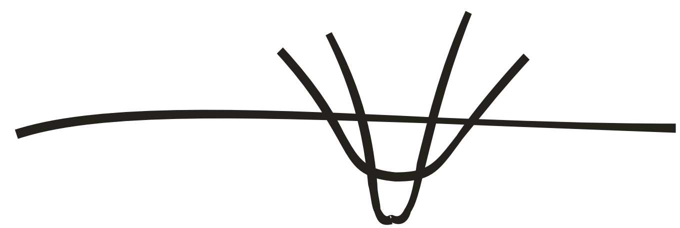
- Проекции кривых начинают с самой ближней к наблюдателю.
- При обработке i-й кривой руководствуются следующим правилом:
- Если на текущей плоскости при некотором заданном значении *x* соответствующее значение *y* на кривой больше значения *y* для всех предыдущих кривых при этом значении *x*, то текущая кривая видима в этой точке; в противном случае она невидима.

Особенности реализации

- Для хранения максимальных значений ***y*** при каждом значении ***x*** используется массив, длина которого равна числу различимых точек (разрешению) по оси ***x*** в пространстве изображения.
- Значения, хранящиеся в этом массиве, представляют собой текущие значения "горизонта". Поэтому по мере рисования каждой очередной кривой этот горизонт "всплывает". Фактически этот алгоритм удаления невидимых линий работает каждый раз с одной линией

Нижний горизонт

а



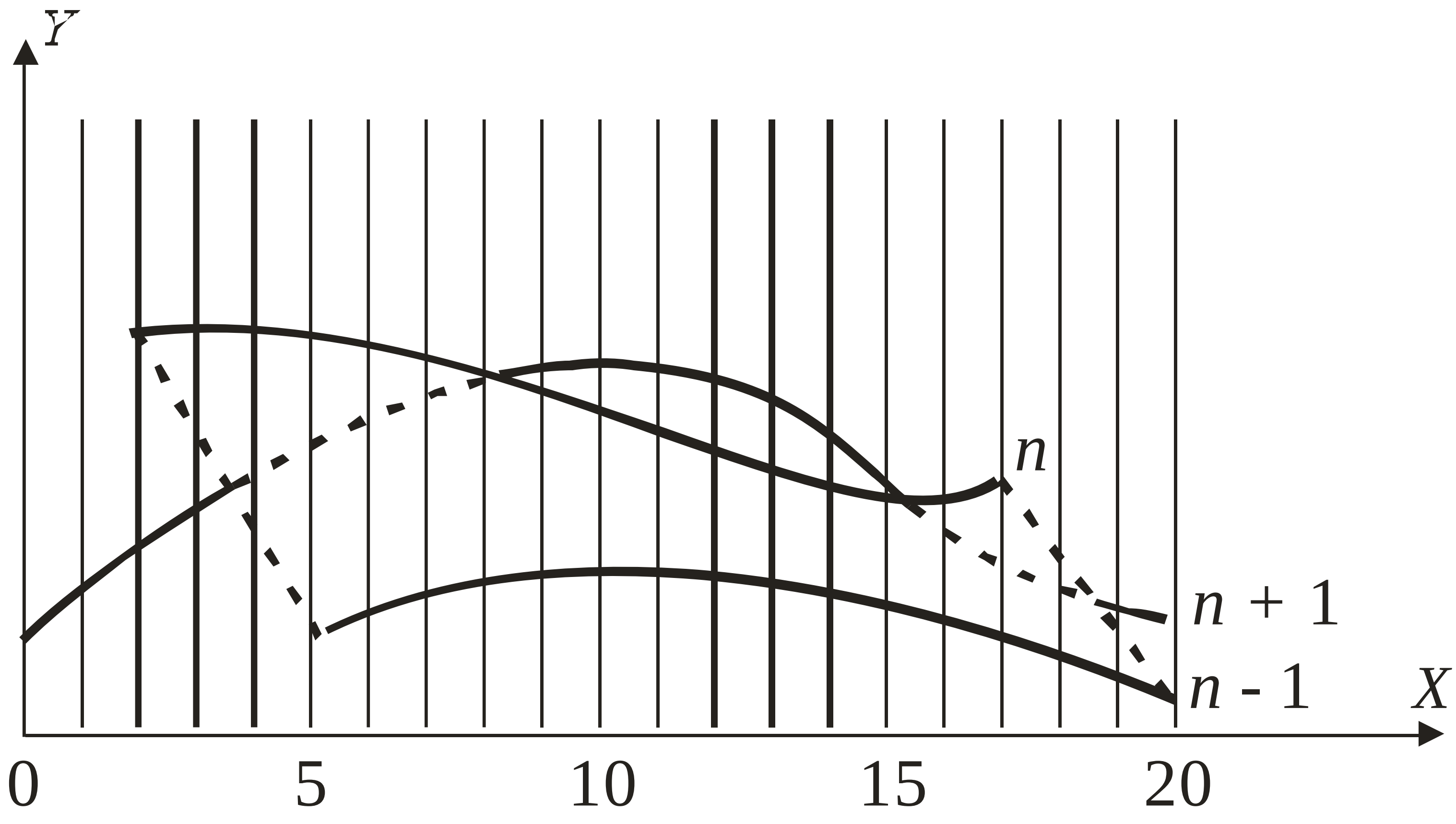
б

- реализуется при помощи второго массива, длина которого равна числу различимых точек по оси *x* в пространстве изображения. Этот массив содержит наименьшие значения *y* для каждого значения *x*.

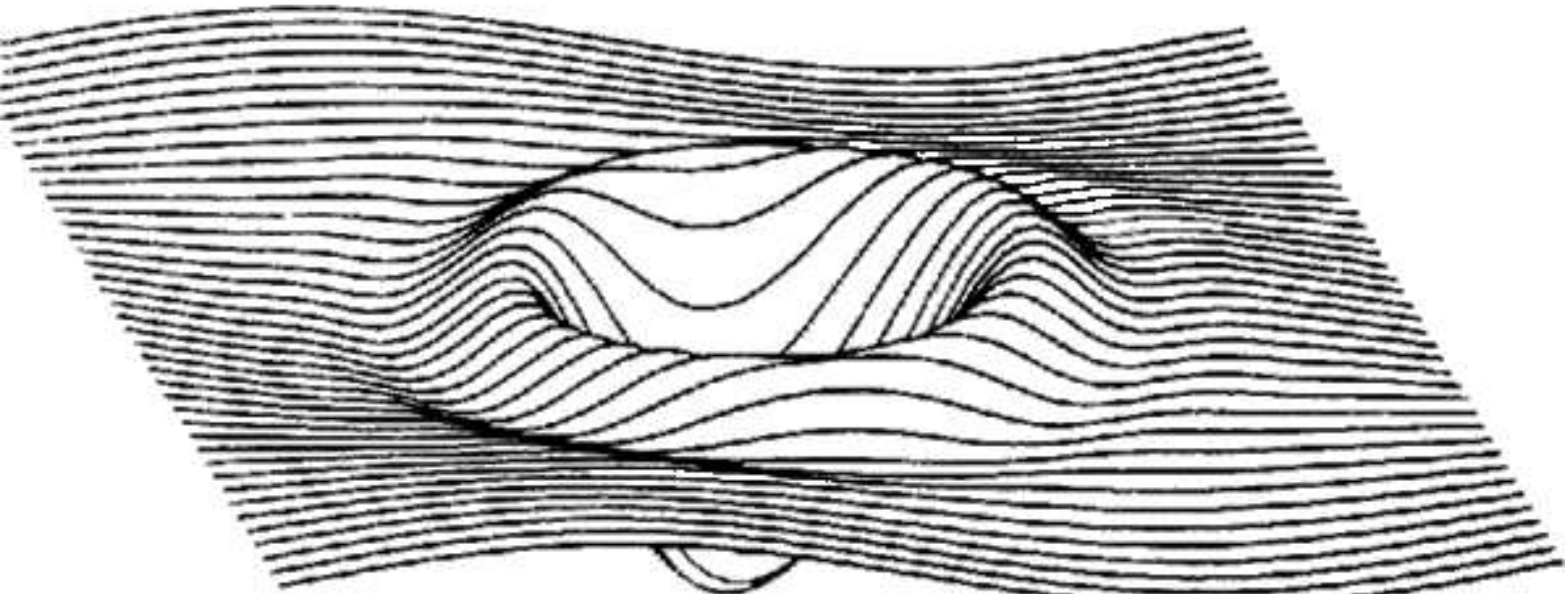
Программная реализация

- While z<5 do
- begin   x:=0;
- While x<=2*Pi do
- begin
- y := MyFunc(z,x); // Вычисление функции
- Preobr; //Вычисление проекции
- if yc<Up[xc] then //Обработка верхнего горизонта
- begin
- Pixels[xc,yc] := clBlack;   Up[xc]:=yc;
- end;
- if yc>Down[xc] then //Обработка нижнего горизонта
- begin
- Pixels[xc,yc] := clBlack;   Down[xc] := yc;
- end;
- x:=x+0.01;
- end;//while x
- z := z + 0.1;
- end;//while z

Эффект зазубренного ребра



Результат работы



Алгоритм Робертса

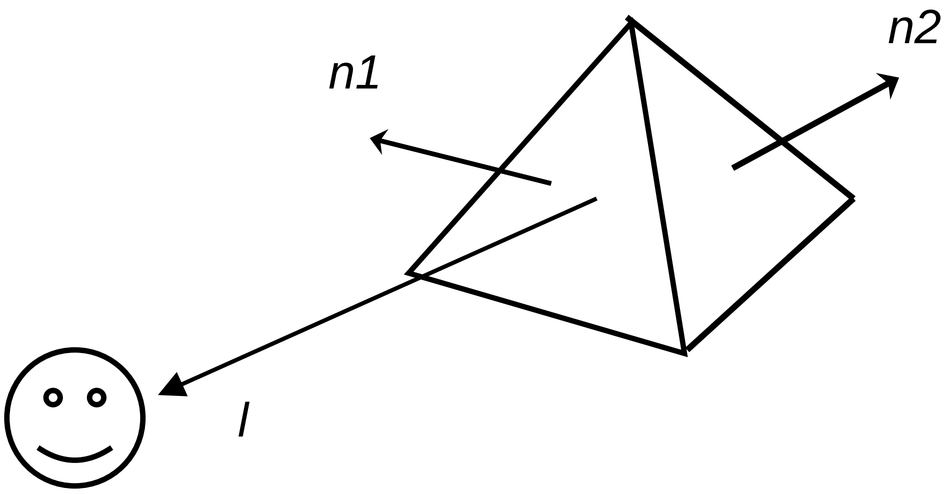
1. Определение нелицевых граней для каждого тела отдельно.
2. Определение и удаление невидимых ребер.

Лицевые и нелицевые грани

- *F* — некоторая грань многогранника
- Плоскость, несущая эту грань, разделяет пространство на два подпространства.
- Назовем положительным то из них, в которое смотрит внешняя нормаль к грани.
- Если точка наблюдения – в положительном подпространстве, то грань – **лицевая**, в противном случае – **нелицевая**.

Определение нелицевых граней

- IF (*l*,*n*) > 0 then грань лицевая
- else грань не лицевая
- где
- *l* – вектор, направленный к наблюдателю
- *n* – вектор внешней нормали грани

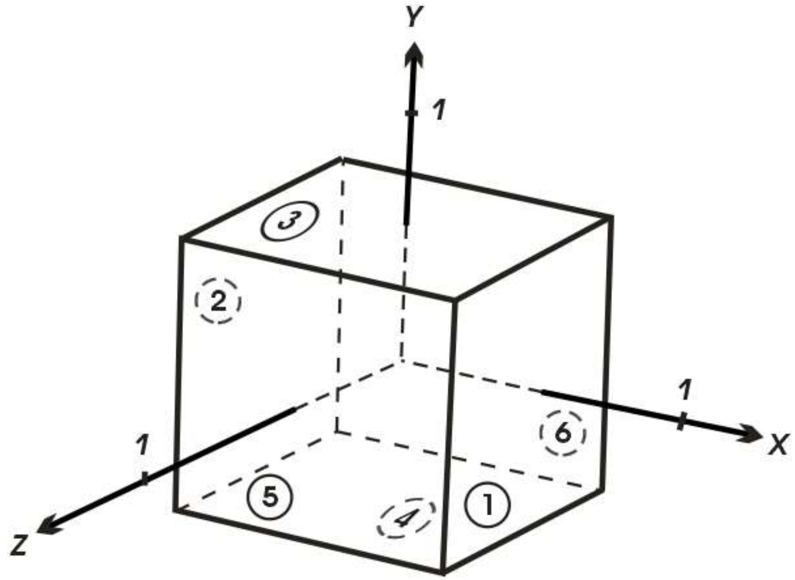


Матрица тела

- *Уравнение плоскости* $ax + by + cz + d = 0$
- В матричной форме $[x\ y\ z\ 1][P]^T = 0$
- Выпуклое твердое тело можно выразить матрицей тела, состоящей из коэффициентов уравнений плоскостей

$$[V] = \begin{bmatrix} a_1 & a_2 & \dots & a_n \\ b_1 & b_2 & \dots & b_n \\ c_1 & c_2 & \dots & c_n \\ d_1 & d_2 & \dots & d_n \end{bmatrix}$$

Пример задания матрицы тела



Шесть плоскостей, описывающих данный куб, таковы: $x1 = 1/2$, $x2 = -1/2$, $y3 = 1/2$, $y4 = -1/2$, $z5 = 1/2$, $z6 = -1/2$. Более подробно уравнение правой плоскости можно записать как
$x1 + 0 \cdot y1 + 0 \cdot z1 - (1/2) = 0$ или $2x1 - 1 = 0$

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ -1/2 & 1/2 & -1/2 & 1/2 & -1/2 & 1/2 \end{bmatrix} \qquad \begin{bmatrix} 2 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 2 \\ -1 & 1 & -1 & 1 & -1 & 1 \end{bmatrix}$$

Проверка матрицы тела

- Экспериментально проверим матрицу тела с помощью точки, о которой точно известно, что она лежит внутри тела. Если знак скалярного произведения для какой-нибудь плоскости больше нуля, то соответствующее уравнение плоскости следует умножить на -1. [S] = [1/4 1/4 1/4   1] = [1 1 1 4].
- [S]·[V] = [1 1 1 4] · $\begin{bmatrix} 1 & \boxed{1} & 0 & \boxed{0} & 0 & \boxed{0} \\ 0 & \boxed{0} & 1 & \boxed{1} & 0 & \boxed{0} \\ 0 & \boxed{0} & 0 & \boxed{0} & 1 & \boxed{1} \\ -1/2 & \boxed{1/2} & -1/2 & \boxed{1/2} & -1/2 & \boxed{1/2} \end{bmatrix}$
- =[-2 6 -2 6 -2 6].

Построение матрицы тела по трем точкам

- Подстановка координат трех неколлинеарных точек (x_1, y_1, z_1), (x_2, y_2, z_2), (x_3, y_3, z_3) в нормированное уравнение плоскости дает

$$ax_1 + by_1 + cz_1 = -1$$
$$ax_2 + by_2 + cz_3 = -1$$
$$ax_3 + by_2 + cz_3 = -1$$

- В матричной форме [X][C] = [D]

$$\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

- Решение уравнения $[C] = [X]^{-1}[D]$.

Второй вариант определения матрицы тела

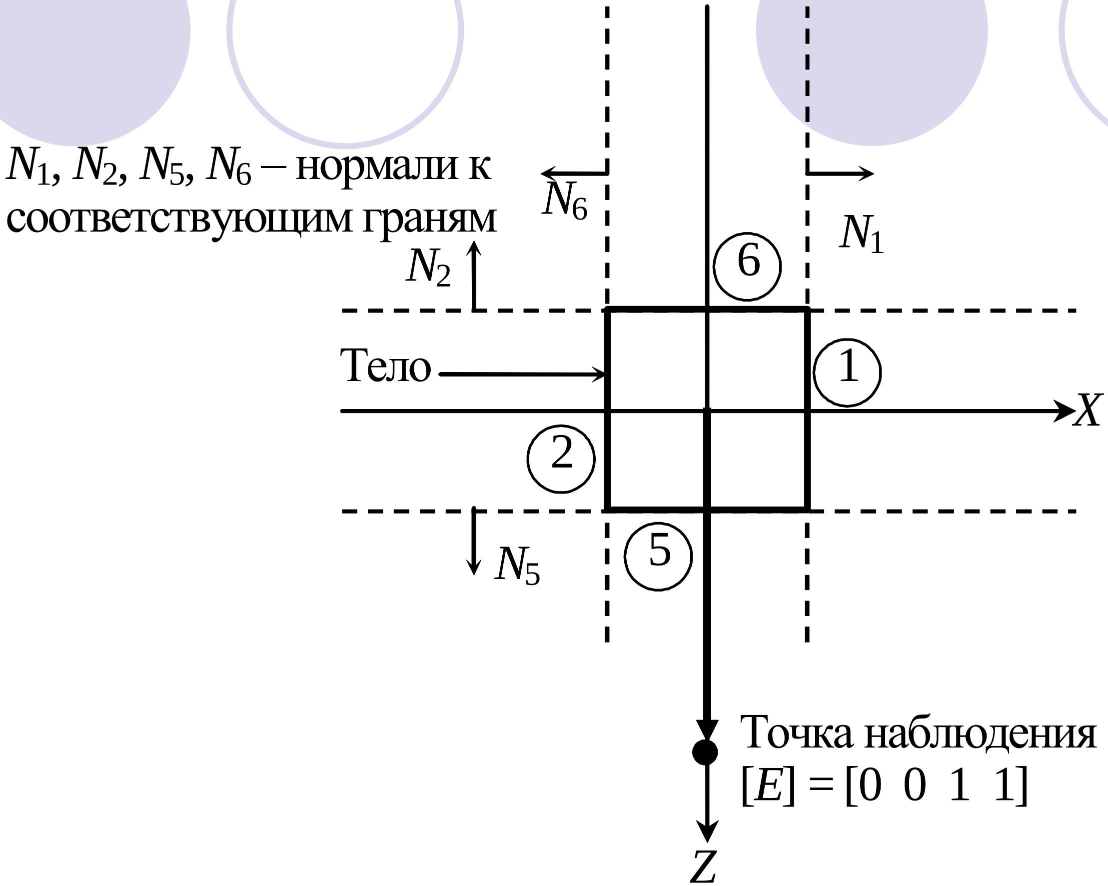
- Если известен вектор нормали к плоскости, т. е.
- n = $a\mathbf{i} + b\mathbf{j} + c\mathbf{k}$,
- где ***i, j, k*** – единичные векторы осей *x*, *y*, *z* соответственно. Тогда уравнение плоскости примет вид
- $ax + by + cz + d = 0$
- Величина *d* вычисляется с помощью произвольной точки на плоскости. В частности, если компоненты этой точки на плоскости
- (x_1, y_1, z_1), то
- $d = -(ax_1 + by_1 + cz_1)$

Получение матрицы тела при 3d преобразованиях

1. [B] – матрица однородных координат
2. [T] – матрица преобразования
3. Преобразованные вершины: [BT] = [B][T]
4. [V] – матрица тела
5. [D] нулевая матрица
6. Справедливо, что [B][V] = [D] и [BT][VT] = [D]
7. Приравниваем левые части в п. 6. и получаем [BT][VT] = [B][V]
8. Подставляем [BT] из п. 3 и получаем [B][T][VT] = [B][V]
9. Сокращаем на [B] и умножая слева на $[T]^{-1}$ и получаем

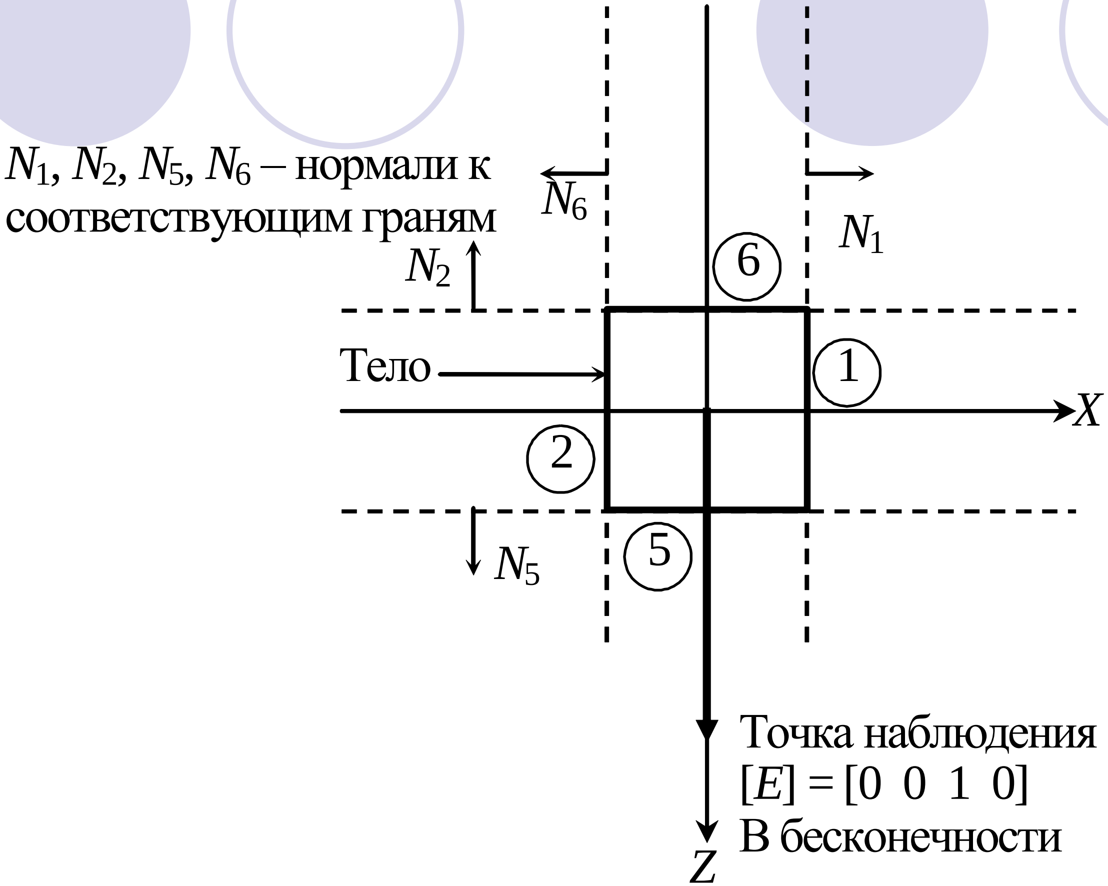
$$[VT] = [T]^{-1}[V]$$

Пример работы алгоритма



$$[E]\cdot[V] = [0\ \ 0\ \ 1\ \ 1] \cdot \begin{bmatrix} 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 \\ -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix} = [-1\ \ -1\ \ -1\ \ -1\ \ 1\ \ -3].$$

Пример для параллельных проекций



$$[E]\cdot[V] = [0\ \ 0\ \ 1\ \ 0]\cdot\begin{bmatrix} 2 & -2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & -2 \\ -1 & -1 & -1 & -1 & -1 & -1 \end{bmatrix} = [0\ \ 0\ \ 0\ \ 0\ \ 2\ \ -2].$$

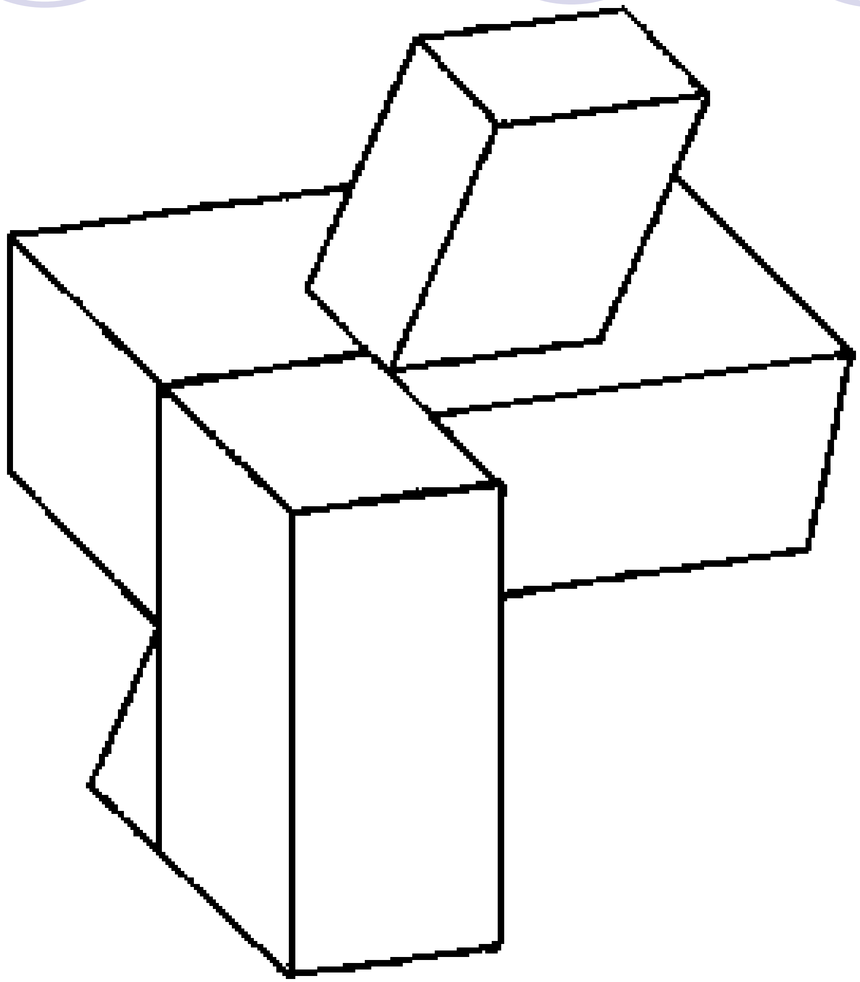
2 этап алгоритма Робретса (Удаление невидимых ребер)

- Каждый оставшийся отрезок или ребро нужно сравнить с другими телами

Возможны следующие случаи:

- Грань ребра не закрывает. Ребро остается в списке ребер.
- Грань полностью закрывает ребро. Ребро удаляется из списка рассматриваемых ребер.
- Грань частично закрывает ребро. В этом случае ребро разбивается на несколько частей, видимыми из которых являются не более двух. Само ребро удаляется из списка рассматриваемых ребер, но в список проверяемых ребер добавляются те его части, которые данной гранью не закрываются.

Результат



Алгоритм, использующий z–буфер