

Проецирование трехмерных объектов

Лекция №6
«компьютерная графика»

КЛАССИФИКАЦИЯ ПРОЕКЦИЙ

Определение

- В общем случае проекции преобразуют точки, заданные в системе координат размерностью n в точки системы координат размерностью меньшей, чем n
- точки трехмерного пространства преобразуются в точки двумерного пространства

Основные элементы проекции:

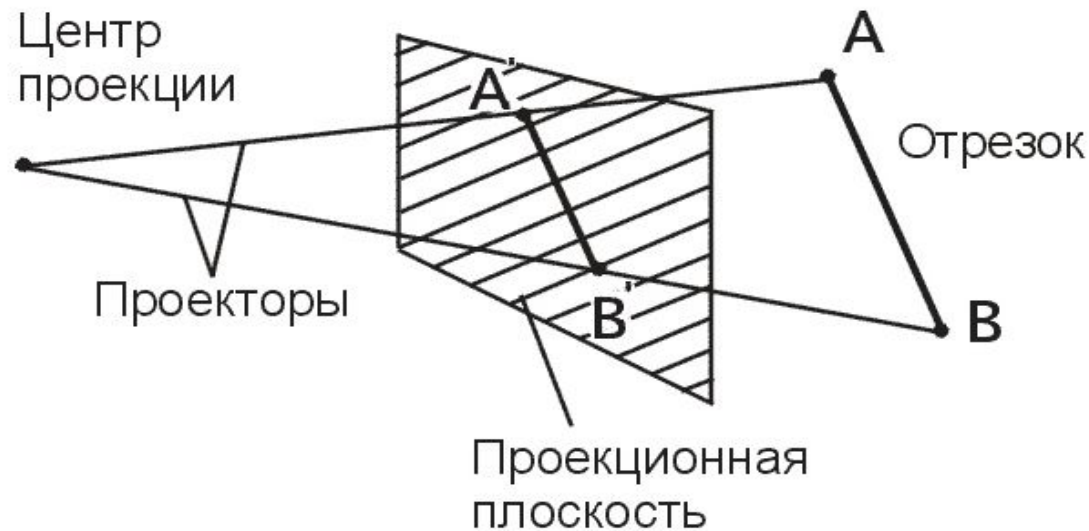
- Центр проекции
- Проецирующие лучи (проекторы)-
прямые
- Проекционная (картинная)
плоскость



плоские геометрические
проекции

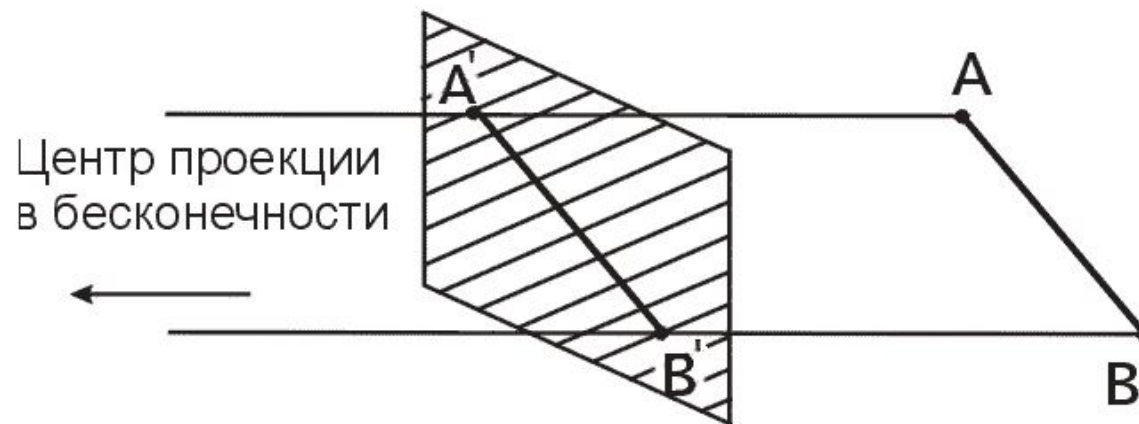
- **Плоские геометрические проекции**
- *центральные*
- *параллельные*

Центральная проекция



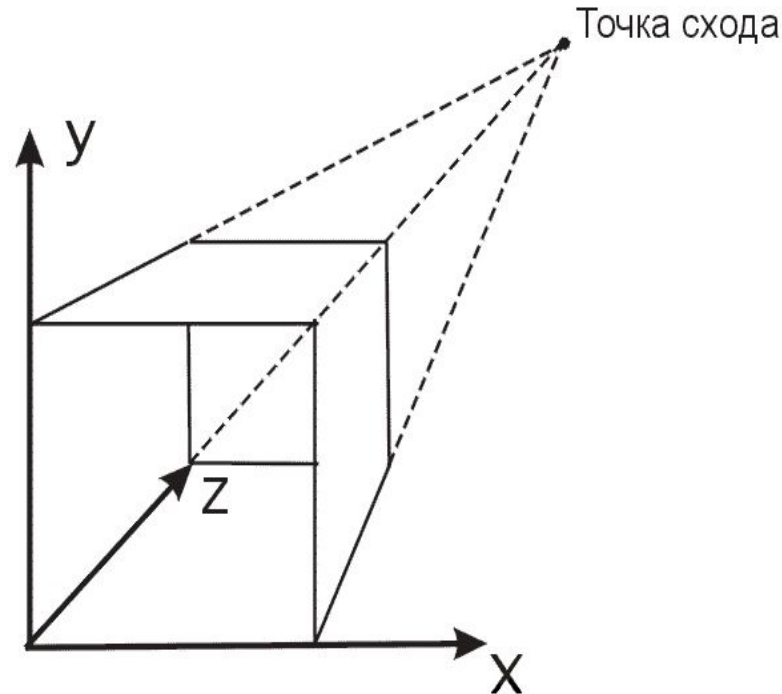
центр проекции находится на конечном расстоянии от проекционной плоскости

Параллельная проекция



центр проекции удален на
бесконечность

Одноточечная проекция

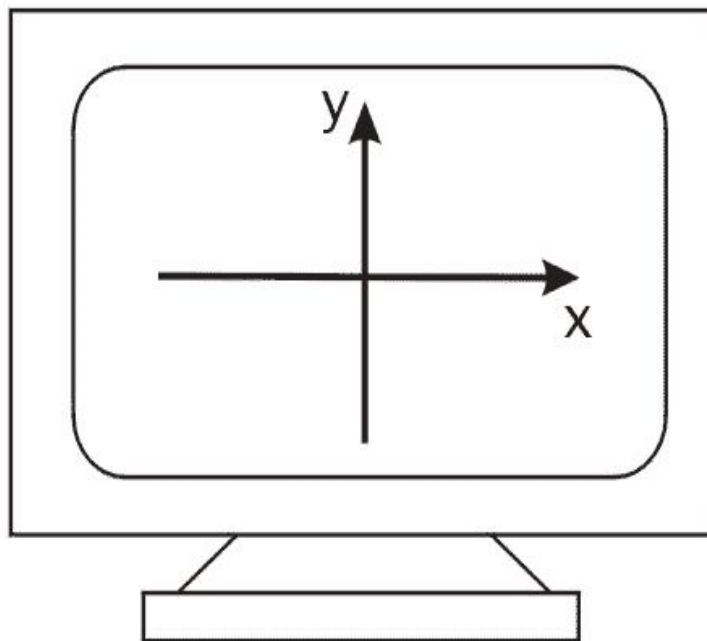


- Точка схода называется **главной**, если совокупность прямых параллельна одной из координатных осей

Основные виды проекций

- Плоские геометрические проекции
 - Параллельные
 - Ортографические
 - Аксонометрические
 - Изометрические
 - Косоугольные
- Центральные

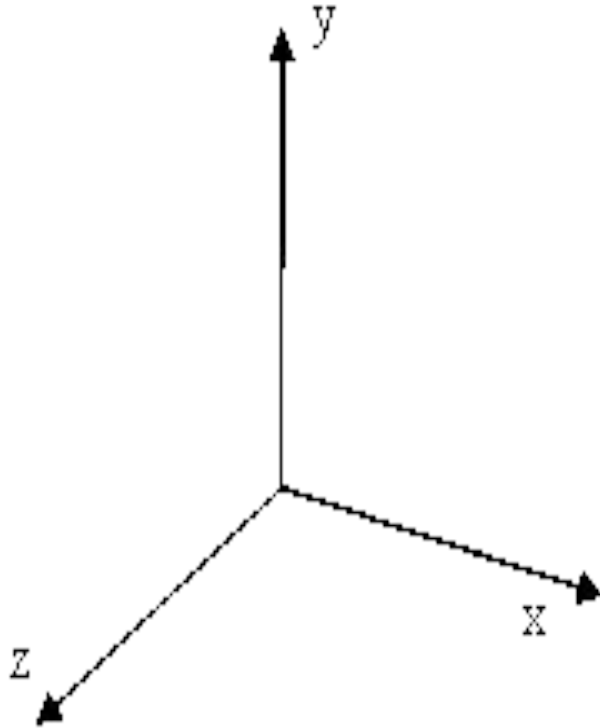
Вывод формул центральной перспективной проекции



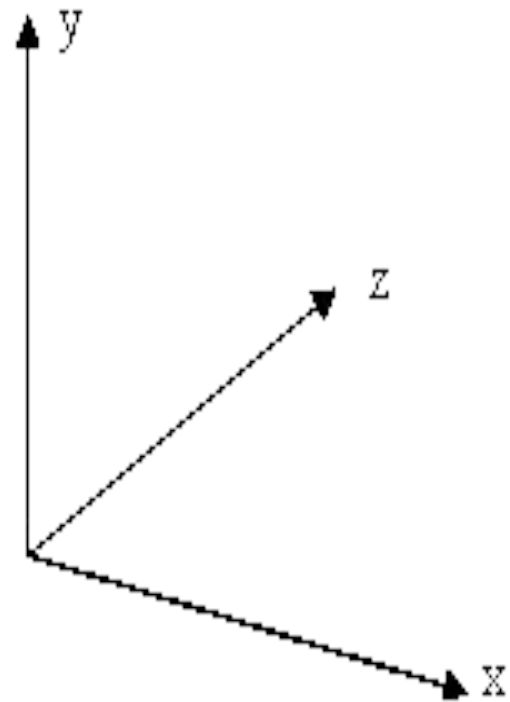
Расположение осей координат на экране

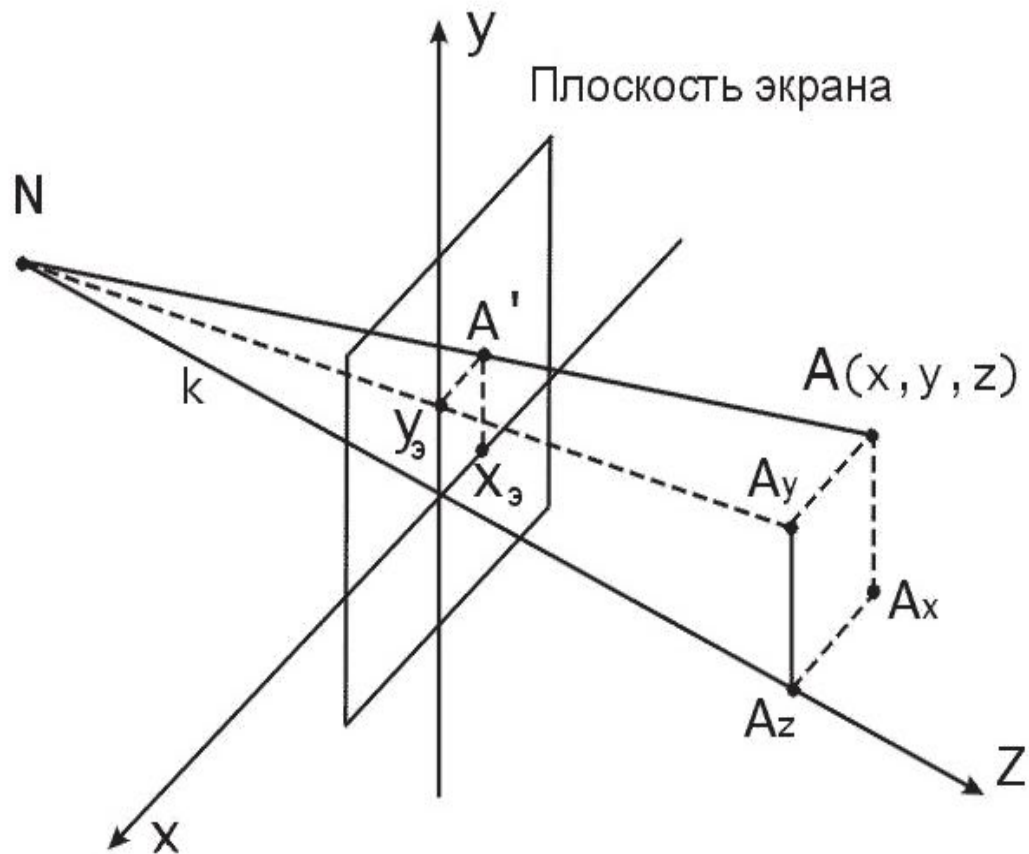
Системы координат

Правосторонняя система координат
(мировая)



Левосторонняя система координат
(видовая)





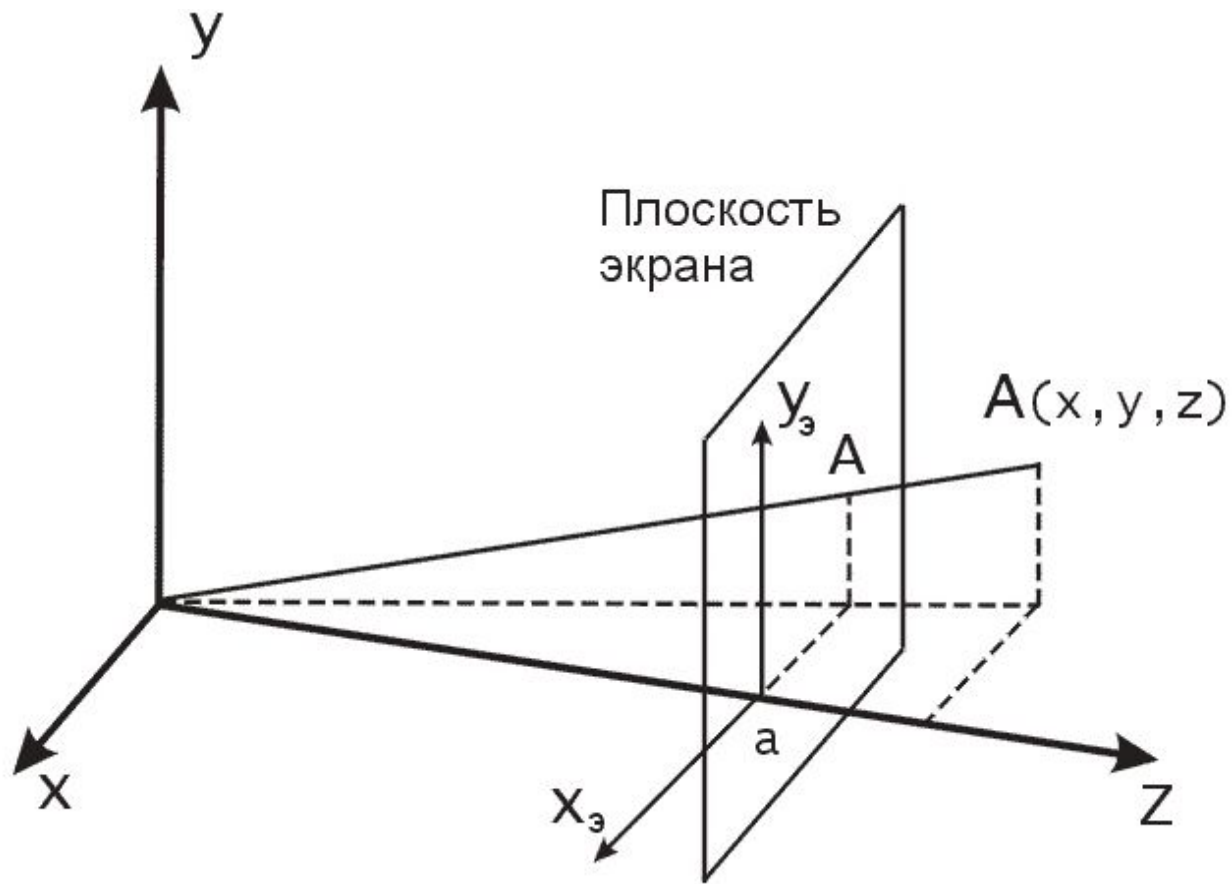
- Плоскость экрана совпадает с проекционной
- k – расстояние от наблюдателя до проекционной плоскости
- A проецируется на экран как A'

Определим координаты A'

Из подобия треугольников $A_y A_z N$ и $y_{\text{э}} ON$

$$\frac{y}{z+k} = \frac{y_{\text{э}}}{k}, \Rightarrow y_{\text{э}} = \frac{ky}{z+k}$$

$$x_{\text{э}} = \frac{kx}{z+k} \quad N = (0, 0, -k)$$



- точку наблюдения поместить в начало координат,
- проекционную плоскость на расстояние a

$$x_{\Theta} = \frac{kx}{Z} \quad y_{\Theta} = \frac{ky}{Z}$$

Буфер кадра (**Frame buffer**)

- **буфер глубины** или Z-буфер (Depth buffer),
- **буфер цвета** (Color buffer),
- **накопительный буфер** (Accumulation buffer)
- **буфер шаблона** (Stencil buffer).

Проверка глубины - это эффективная технология удаления скрытых поверхностей

- активизировать проверку глубины

```
glEnable(GL_DEPTH_TEST);
```

- В программе

```
glClear(GL_COLOR_BUFFER_BIT |
```

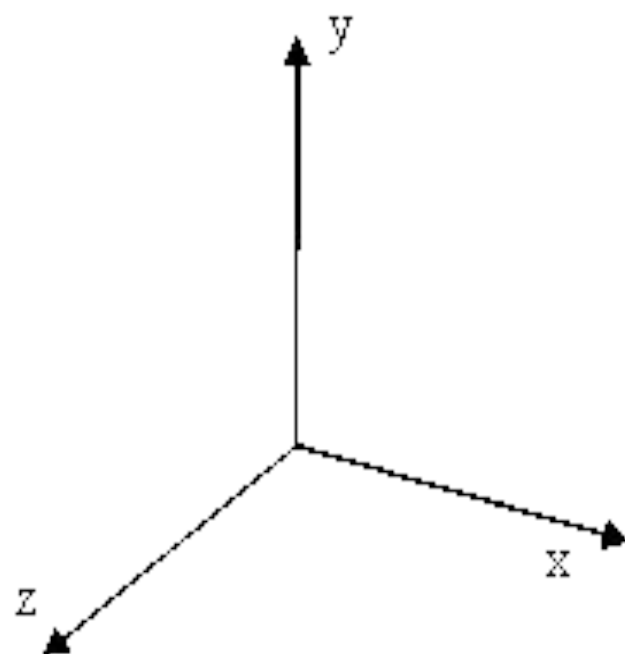
```
GL_DEPTH_BUFFER_BIT);
```

- Отключить

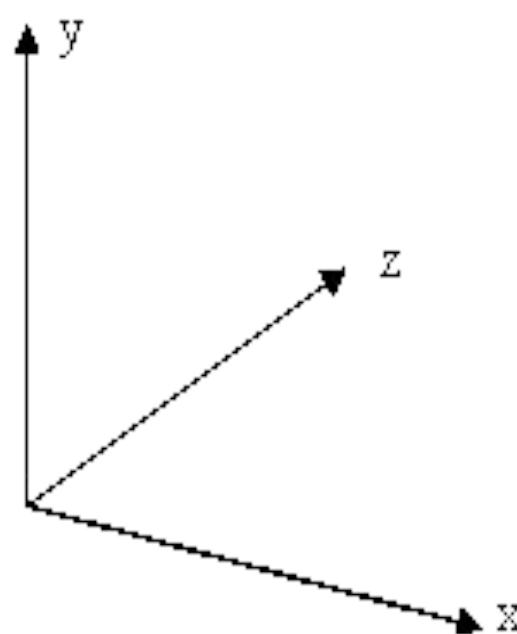
```
glDisable(GL_DEPTH_TEST);
```

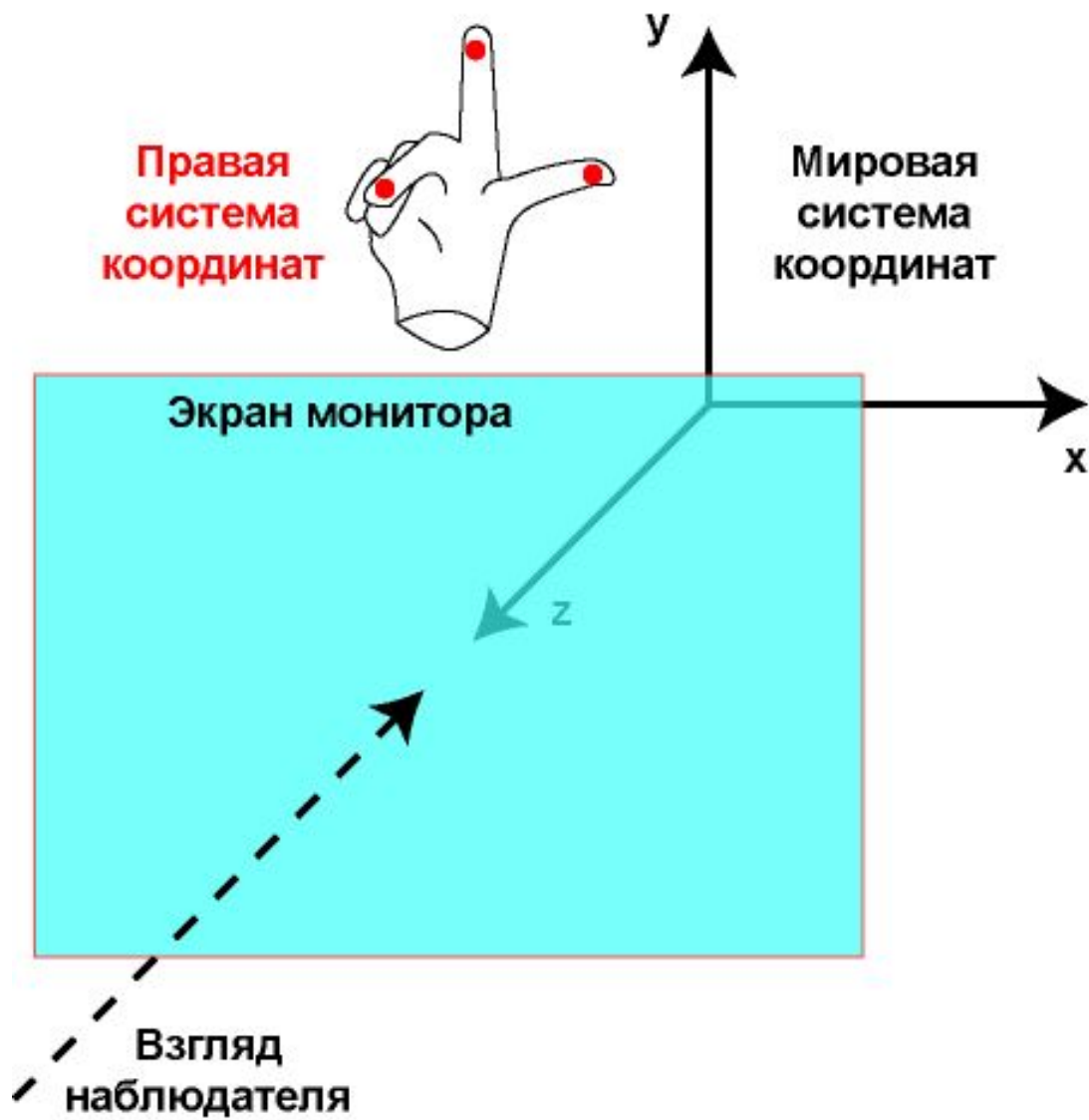
***ПРЕОБРАЗОВАНИЯ ТОЧЕК В
РАЗНЫХ СИСТЕМАХ КООРДИНАТ***

Правосторонняя система координат
(мировая)



Левосторонняя система координат
(видовая)





Масштабирование

- **glScale (arg1, arg2, arg3)**

аргументы - коэффициенты

масштабирования по каждой из осей

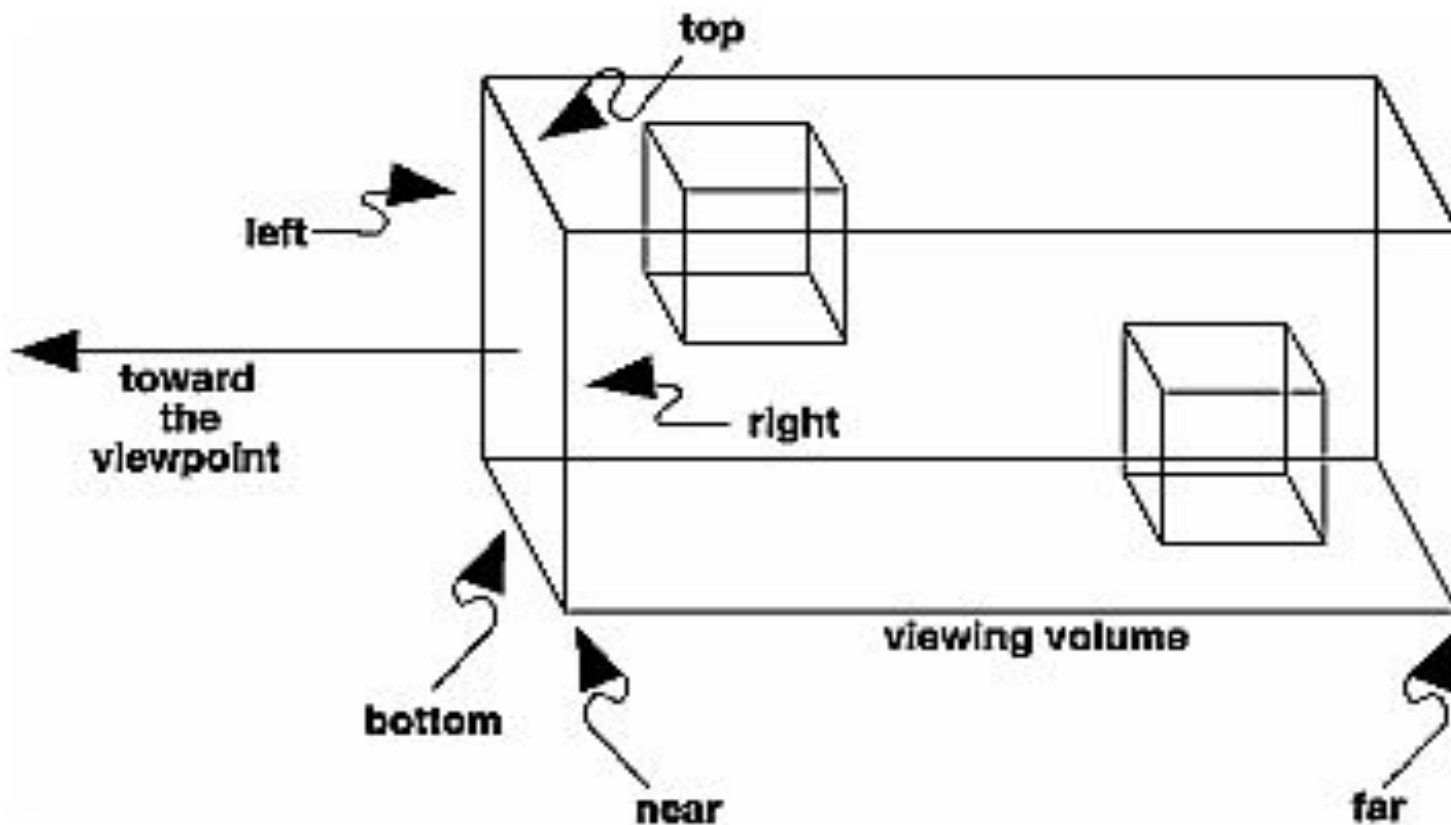
изменить положение точки наблюдения

- void **gluLookAt** (GLdouble eyex, GLdouble eyeey, GLdouble eyez, GLdouble centerx, GLdouble centery, GLdouble centerz, GLdouble upx, GLdouble upy, GLdouble upz)

Проекционные преобразования

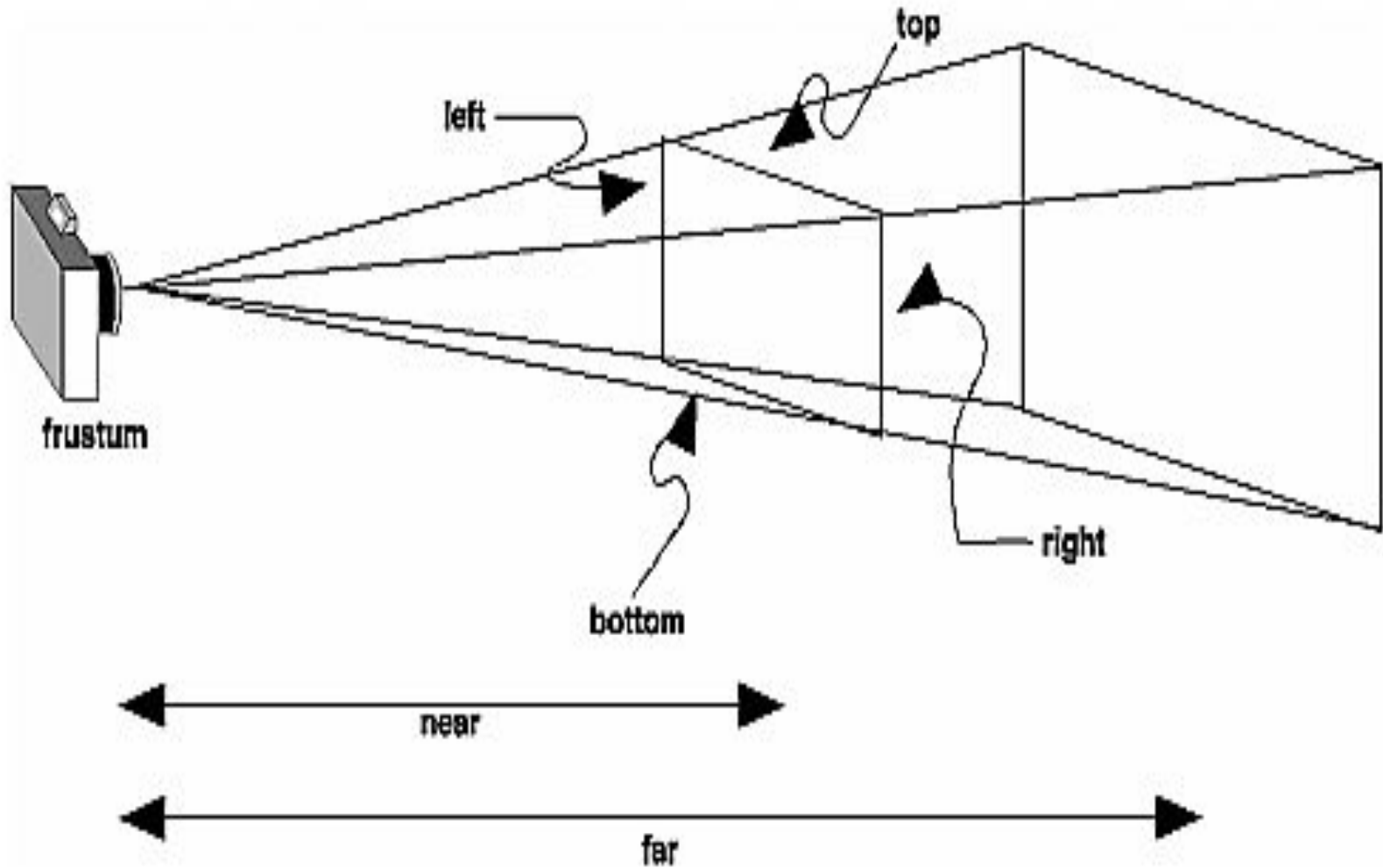
- определяем **отсекающий объем видимости**
 - Как сцена будет отображаться на экране монитора
 - Какие объекты или части объектов войдут в окончательное изображение

Параллельное проецирование в OpenGL



- void **glOrtho**(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far)
-
- void **gluOrtho2D**(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top)

Перспективные преобразования в OpenGL

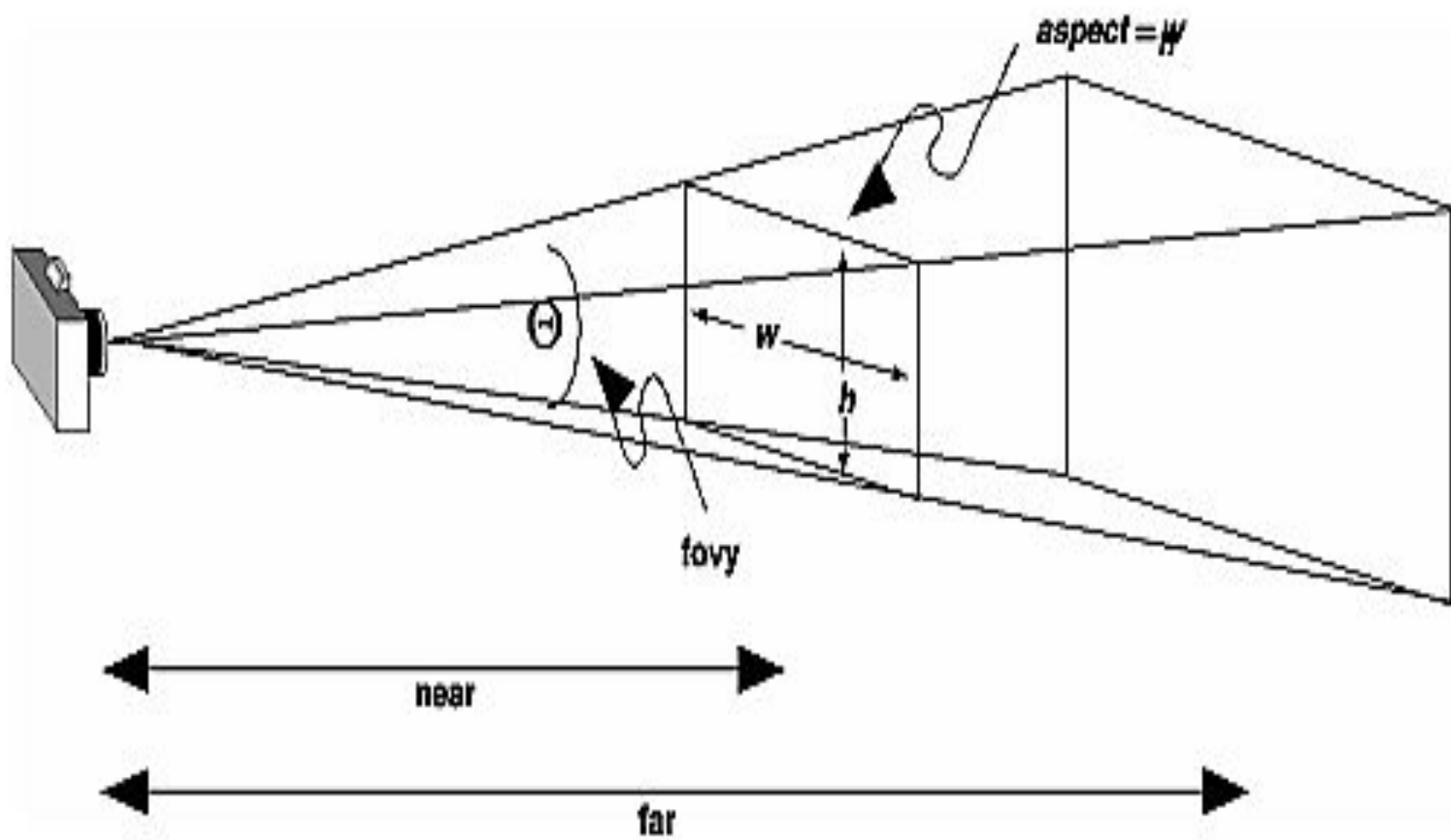


- *void **glFrustum**(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far);*

```
glMatrixMode(GL_PROJECTION);
```

```
glLoadIdentity();
```

```
glFrustum(xmin, xmax, ymin, ymax, near, far);
```



- void **gluPerspective**(GLdouble angley, GLdouble aspect, GLdouble znear, GLdouble zfar)

Область вывода

- void **glViewport**(GLint x, GLint y, GLint width, GLint height)

- $o_x = x + \text{width}/2$, $o_y = y + \text{height}/2$

- Пусть $p_x = \text{width}$, $p_y = \text{height}$,

$$(x_w, y_w, z_w)^T = \left(\begin{array}{l} (p_x/2) x_n + o_x, \\ (p_y/2) y_n + o_y, \\ [(f-n)/2] z_n + (n+f)/2 \end{array} \right)^T$$

- n и f задают минимальную и максимальную глубину точки в окне
- void **glDepthRange**(GLclampd n, GLclampd f)