Трансляция презентации (во время очных лекций).

При просмотре презентации в PDF для отображения анимаций на слайдах необходимо использовать Acrobat Reader, KDE Okular, PDF-XChange или Foxit Reader.

Компьютерная графика

Лекция 5
Операции с вершинами в графическом конвейере OpenGL

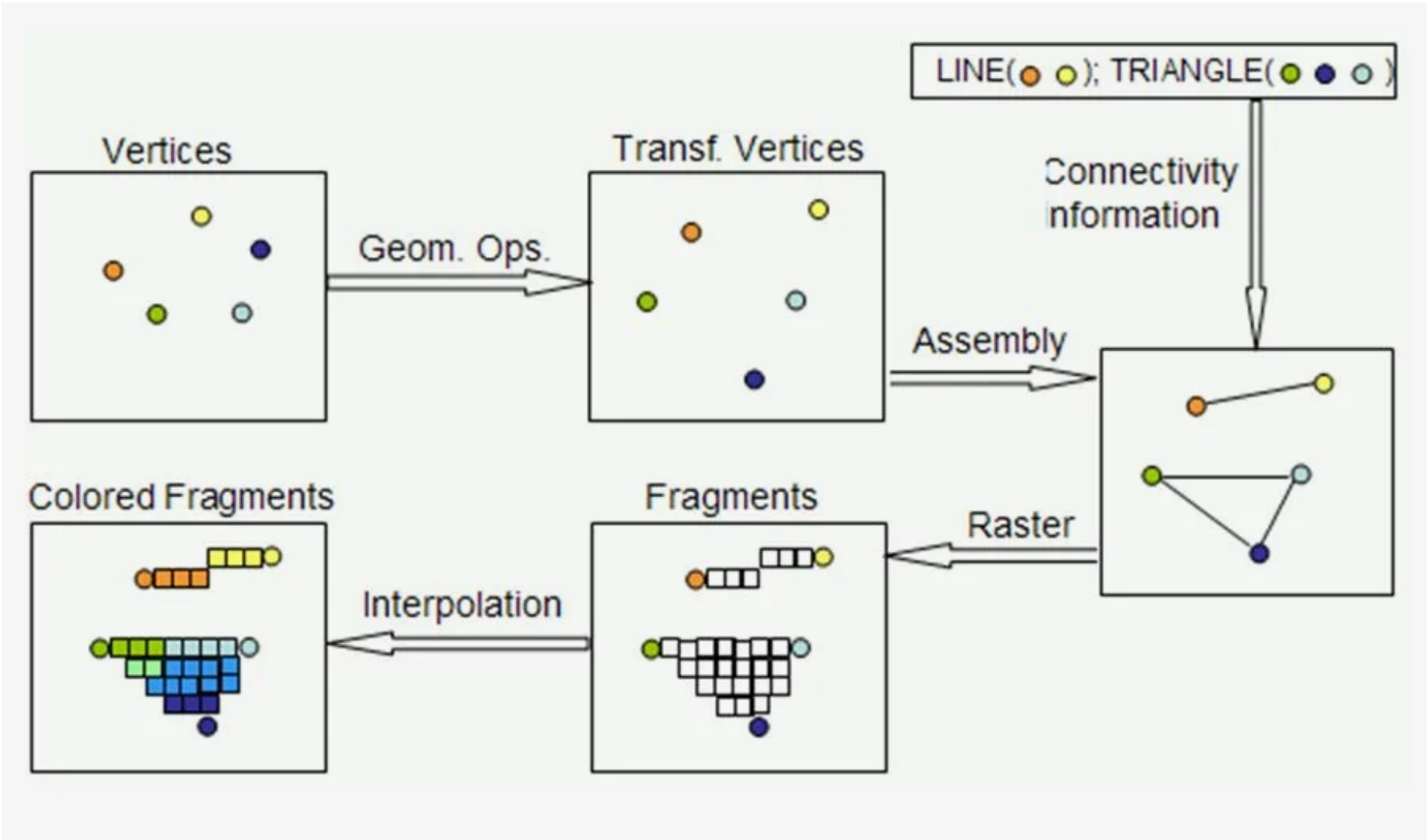
Гаврилов Андрей Геннадьевич

Кафедра Информационных технологий и вычислительных систем
МГТУ «СТАНКИН»

9 апреля 2024 г.

План лекции

Класический конвейер OpenGL

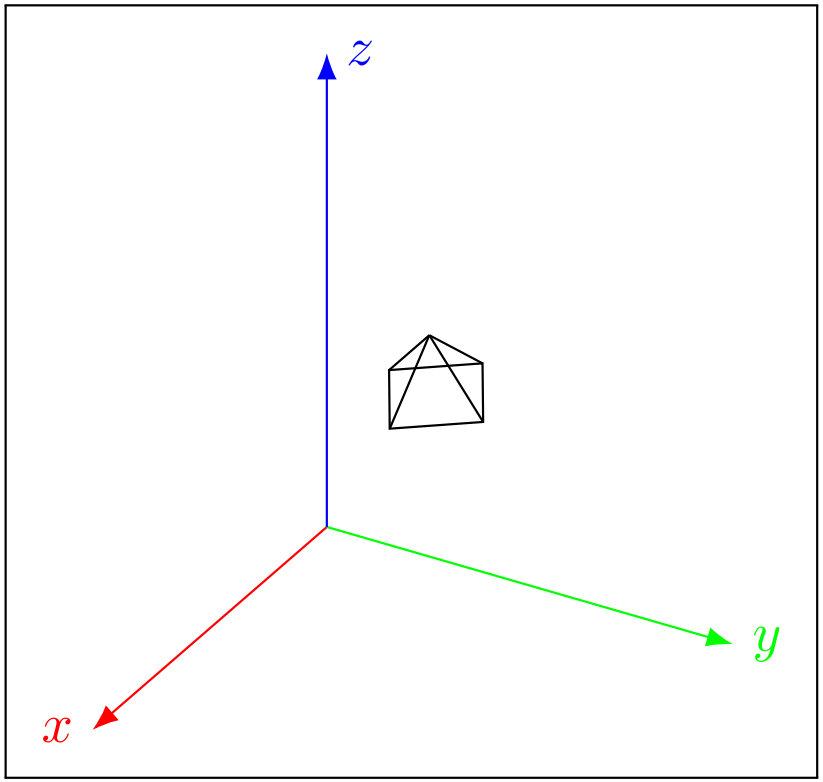


Раздел 1

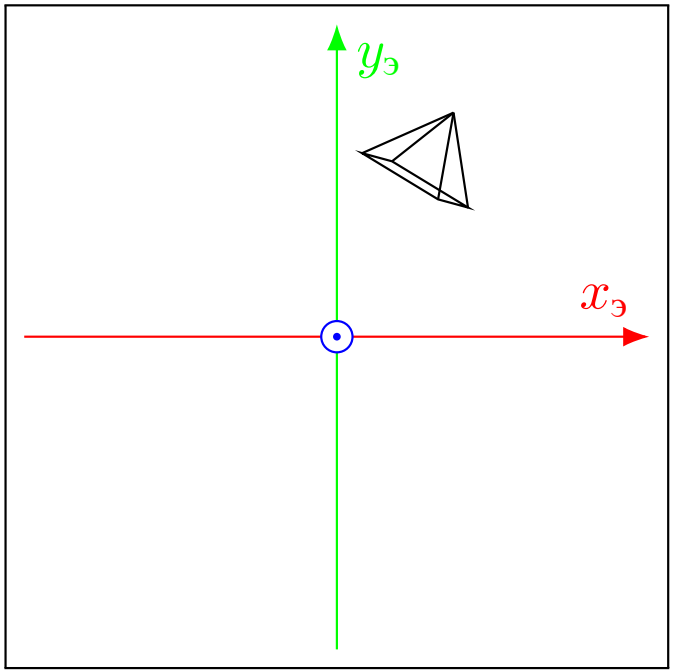
Вершинные операции. Преобразование координат

Цель вершинных операций

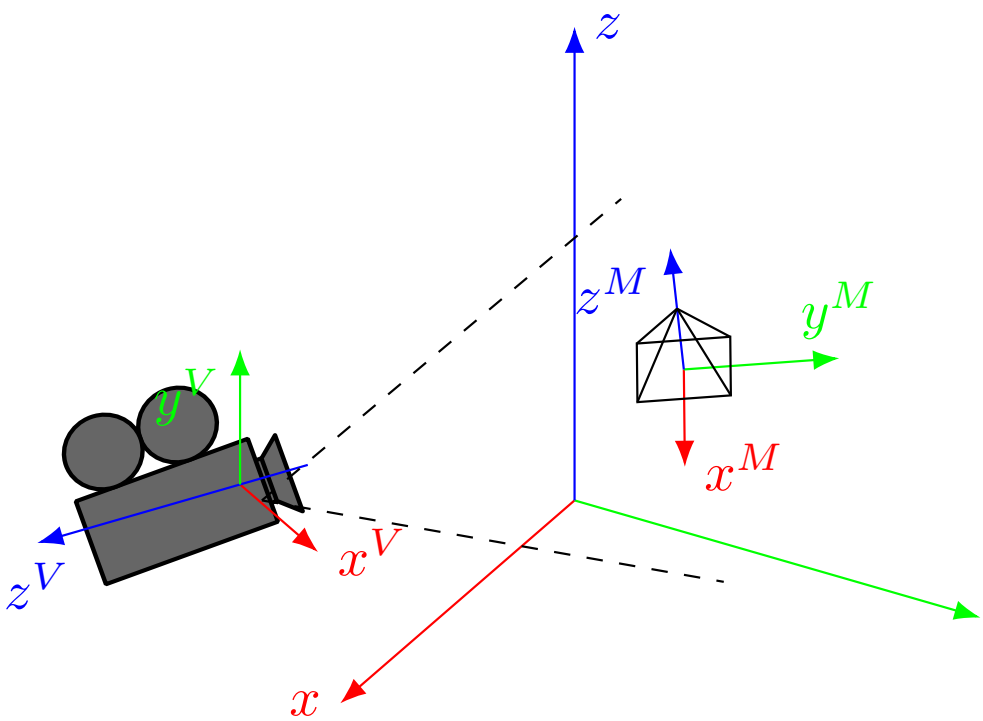
У нас есть объект в пространстве



А нам надо объект на экране



Системы координат сцены



$x^M y^M z^M$ — С.К. модели
xyz — мировая С.К.
$x^V y^V z^V$ — С.К. наблюдателя (видовая)

$$\mathbf{V}^M = \begin{pmatrix} 0.5 & -0.5 & -0.5 & 0.5 & 0 \\ 0.5 & 0.5 & 0.5 & -0.5 & 0 \\ 0 & 0 & 0.5 & 0 & 0.4 \end{pmatrix}$$

Задача:
Получить координаты объекта V^M в видовой СК:
$V^M \to V \to V^V$

Расчёт мировых координат

Система координат модели V определяется *модельной матрицей* $\mathbf{M}^M$.

$\mathbf{M}^M$ — совокупность трансформаций объекта, выраженных матрицами $\mathbf{T}(\vec{p})$, $\mathbf{R}(\vec{v}, \theta)$, $\mathbf{S}(s_x, s_y, s_z)$, т.е. — совокупность переносов, поворотов и масштабирований.

Пример:

$$\mathbf{M}^M = \mathbf{T}(1,3,5)\mathbf{R}_x(32^\circ)\mathbf{R}_z(-45^\circ)$$

Порядок действий (обратный записи):
1. Поворот вокруг z на -45°.
2. Поворот вокруг x на 32°.
3. Перенос на $(1,3,5)$.

Переход в мировую С.К.

$$V^M \to V: \ \mathbf{V} = \mathbf{M}^M \mathbf{V}^M$$

Расчёт видовых координат

Видовая система координат определяется видовой матрицей $\mathbf{M}^V$. Центр видовой С.К. можно интерпретировать как положение наблюдателя сцены (камеры):

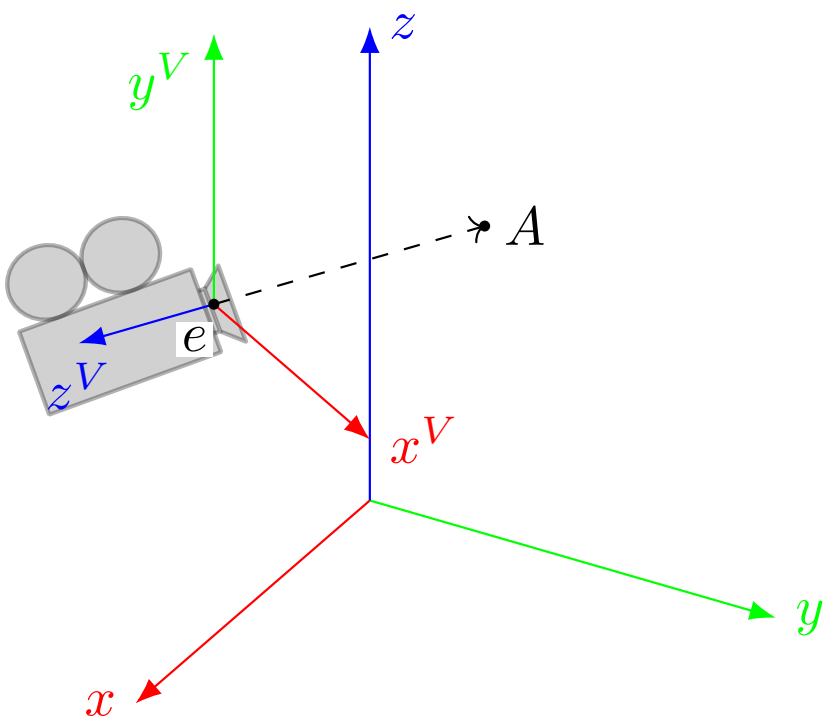
- Наблюдатель смотрит в направлении $-z^V$.
- Ось y^V — направление наверх для наблюдателя.

Положение камеры определяется комбинаций перемещений и поворотов, которые составляют $\mathbf{M}^V$.

Переход в видовую С.К.

$$V \to V^V : \mathbf{V}^V = \mathbf{M}^V \mathbf{V}$$

Видовая матрица. Способ получения 1.



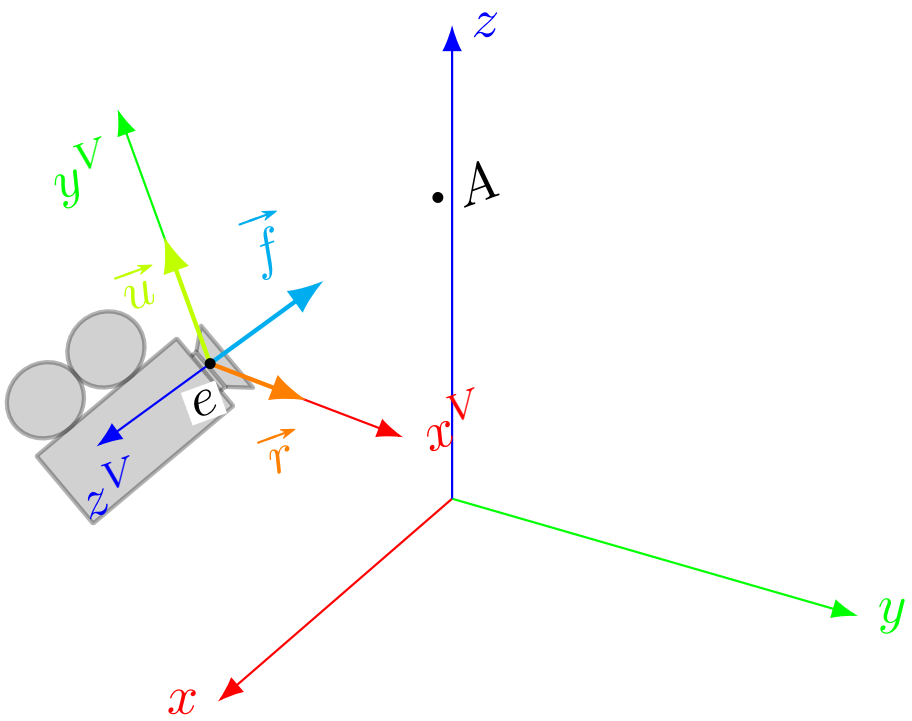
Положение камеры определяется совокупностью перемещений и поворотов, которые поместили её туда, где она располагается:
$\mathbf{M} = \mathbf{T}(\dots)\mathbf{R}(\dots)\mathbf{T}(\dots)\dots$

Задача сводится с переходу $xyz \to x^V y^V z^V$:
$\mathbf{A}^V = \mathbf{M}^{-1}\mathbf{A}$

$\mathbf{M}^{-1} = \mathbf{M}^V$ — видовая матрица.

или
$\mathbf{M}^V = \mathbf{T}^{-1}(\dots)\mathbf{R}^{-1}(\dots)\mathbf{T}^{-1}(\dots)\dots$

Видовая матрица. Способ получения 2.



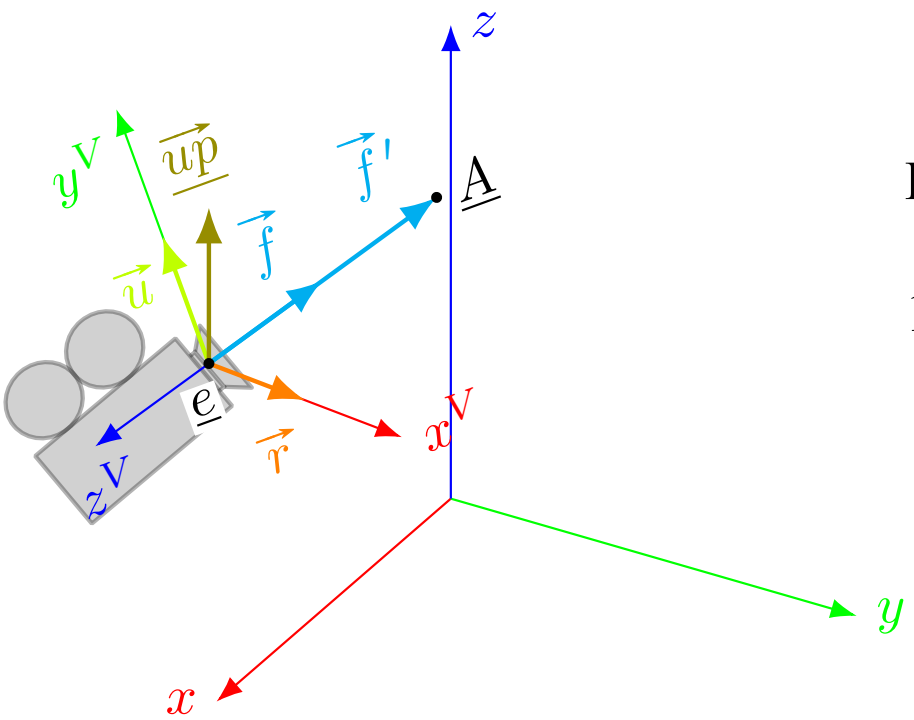
Задача сводится к переходу от базиса $[\vec{i}, \vec{j}, \vec{k}]$ к $[\vec{r}, \vec{u}, -\vec{f}]$ и сдвигу на $(-\vec{e})$

Камера располагается в координате $\vec{e}$.
Направлена на т. A.
Можно определить следующие единичные вектора:

- $\vec{f}$ — направление на объект.
- $\vec{u}$ —направление «наверх» для наблюдателя.
- $\vec{r}$ —направление «направо» для наблюдателя.

$$\mathbf{M}^V = \begin{pmatrix} r_x & u_x & -f_x & -e_x \\ r_y & u_y & -f_y & -e_y \\ r_z & u_z & -f_z & -e_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Настройка камеры



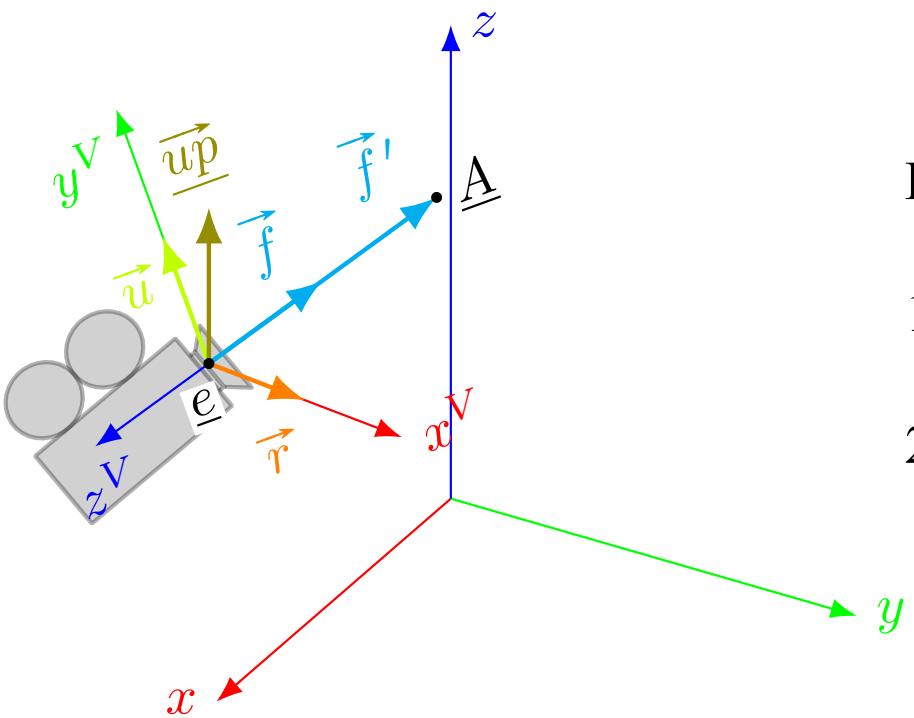
Исходные данные (подчёркнуты на рис.):

- Куда смотрим — A.
- Откуда смотрим — e
- Где небо — $\overrightarrow{up}$.

Найти: $\overrightarrow{r}$, $\overrightarrow{u}$, $\overrightarrow{f}$.

1. $\overrightarrow{f}' = \overrightarrow{A} - \overrightarrow{e}$, $\overrightarrow{f} = \overrightarrow{f}'/|\overrightarrow{f}'|$

Настройка камеры



Исходные данные (подчёркнуты на рис.):

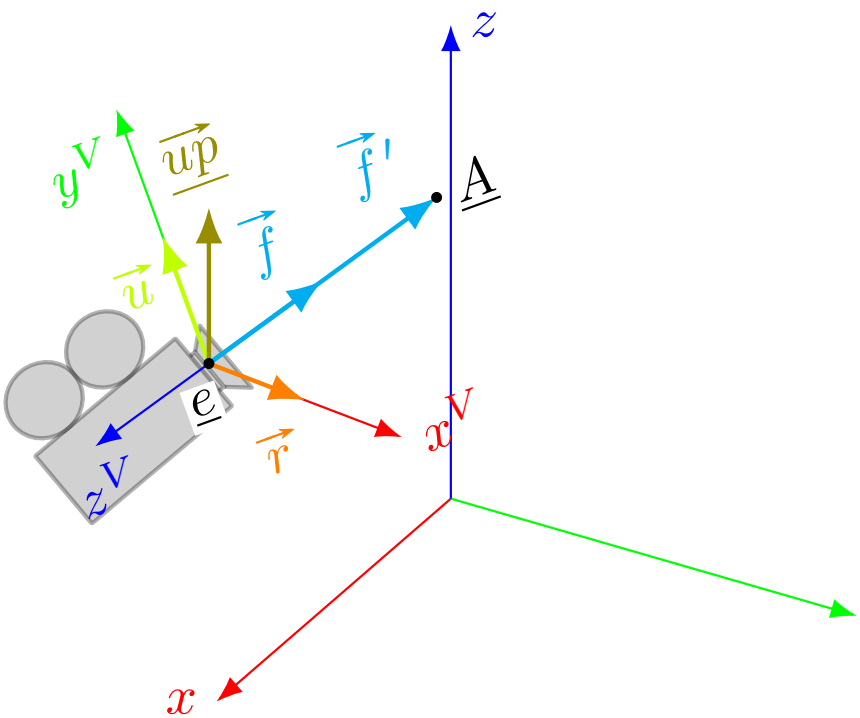
- Куда смотрим — A.
- Откуда смотрим — e
- Где небо — $\overrightarrow{up}$.

Найти: $\overrightarrow{r}$, $\overrightarrow{u}$, $\overrightarrow{f}$.

1. $\overrightarrow{f}' = \overrightarrow{A} - \overrightarrow{e}$, $\overrightarrow{f} = \overrightarrow{f}'/|\overrightarrow{f}'|$

2. $\overrightarrow{up}' = \overrightarrow{up}/|\overrightarrow{up}|$, $\overrightarrow{r} = \overrightarrow{f} \times \overrightarrow{up}'$

Настройка камеры



Исходные данные (подчёркнуты на рис.):

- Куда смотрим — A.
- Откуда смотрим — e
- Где небо — $\overrightarrow{up}$.

Найти: $\overrightarrow{r}$, $\overrightarrow{u}$, $\overrightarrow{f}$.

1. $\overrightarrow{f}' = \overrightarrow{A} - \overrightarrow{e}$, $\overrightarrow{f} = \overrightarrow{f}'/|\overrightarrow{f}'|$

2. $\overrightarrow{up}' = \overrightarrow{up}/|\overrightarrow{up}|$, $\overrightarrow{r} = \overrightarrow{f} \times \overrightarrow{up}'$

3. $\overrightarrow{u} = \overrightarrow{r} \times \overrightarrow{f}$

$$\mathbf{M}^V = \begin{pmatrix} r_x & u_x & -f_x & -e_x \\ r_y & u_y & -f_y & -e_y \\ r_z & u_z & -f_z & -e_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

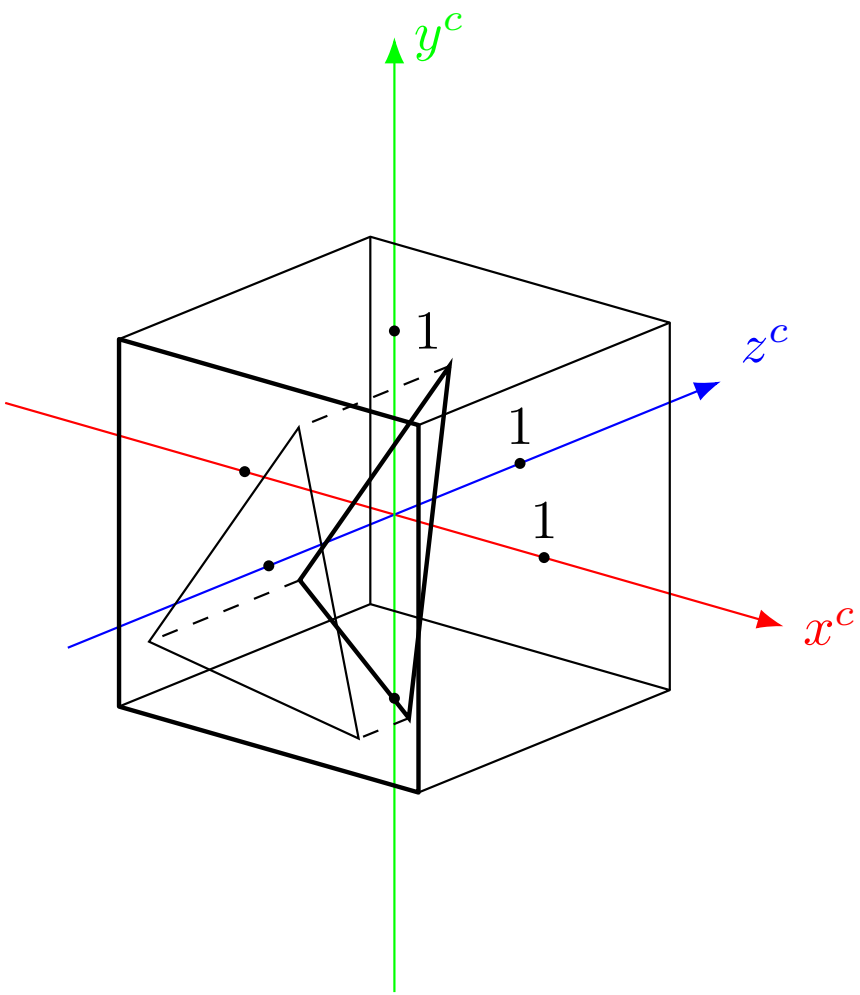
Преобразования в классическом OpenGL

GL_MODELVIEWMATRIX — константа обозначающая $\mathbf{M}^V\mathbf{M}^M$ (MV матрицу).
glGetFloatv(GL_MODELVIEW_MATRIX, ptr) — получить VM-матрицу в указатель ptr.
glMatrixMode(GL_MODELVIEWMATRIX) — режим с VM-матрией. Команда говорит о том, что все дальнейшие матричные операции будут проводится с этой матрицей.
glLoadIdentity() — установить единичную матрицу, сброс всех преобразований.
glMultMatrixd(ptr) — умножить текущую матрицу на матрицу в указателе ptr.
glLoadMatrixd(ptr) — установить текущую матрицу из указателя ptr.
glRotated(…), glTranslated(…), glScaled(…) — домножить матрицу на матрицы поворота, перемещения, масштабирования.
glPushMatrix(), glPopMatrix() — записать/считать матрицу из стека.
gluLookAt(ex,ey,ez,cx,cy,cz,ux,uy,uz) — настроить камеру с центром в e, смотрящую на c, небо по направлению $\vec{u}$.

Раздел 2

Вершинные операции. Проецирование

Координатная система OpenGL

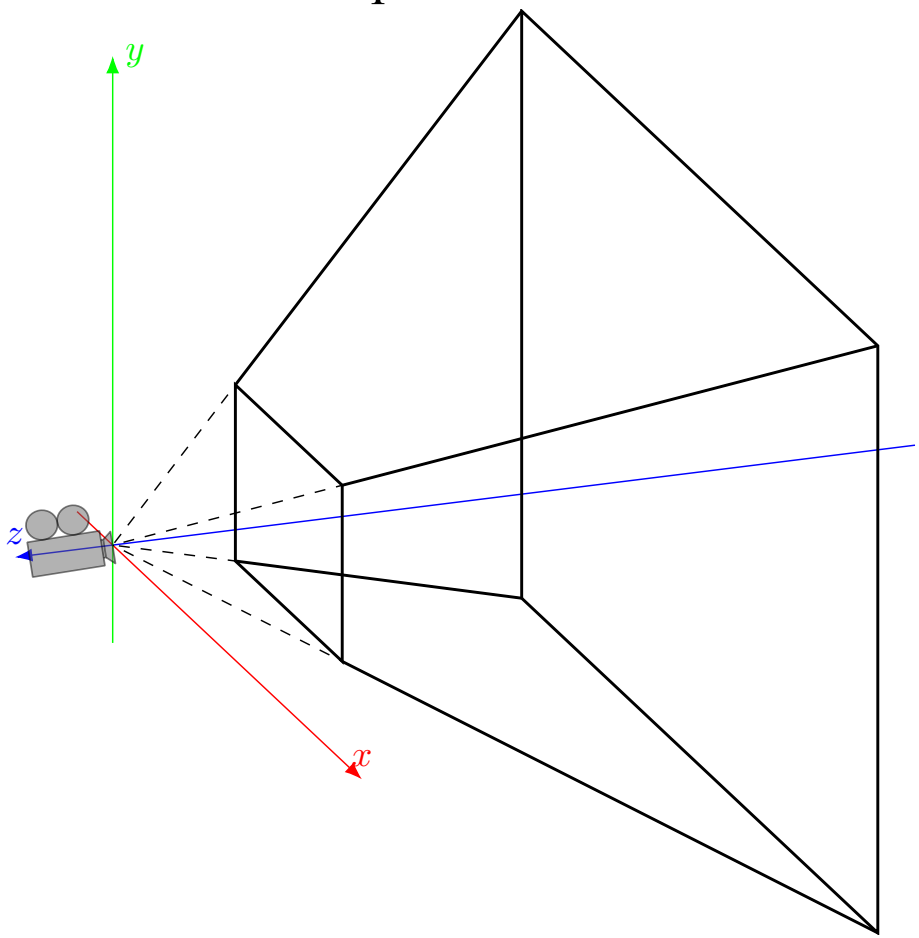


Канонический объём отсечения или Canonic view volume (CVV) — куб с ребром = 2 и центром в начале координат.

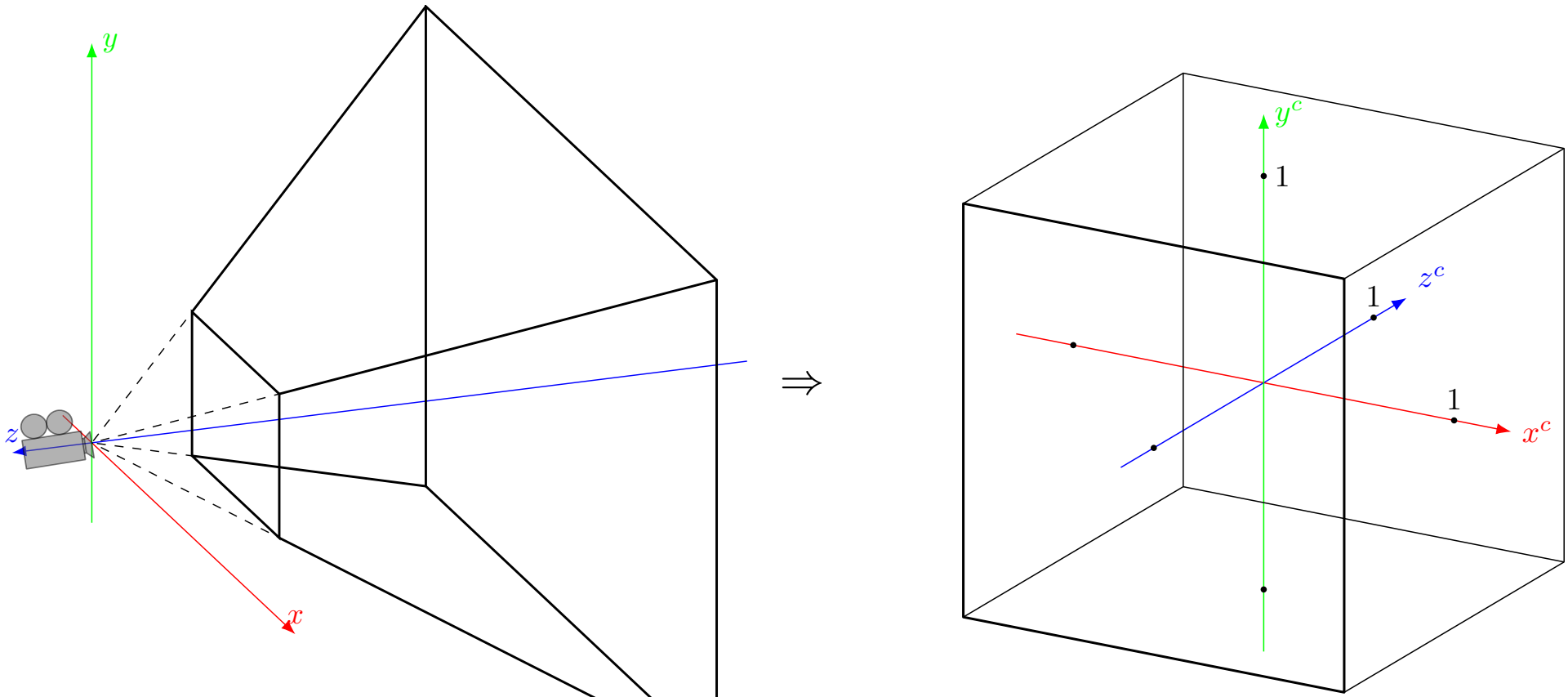
Грань куба $z^c = -1$ выступает в роли «экрана». В кадр попадёт только то, что находится внутри куба, т.е. все точки у которых $|x^c| < 1$, $|y^c| < 1$, $|z^c| < 1$.

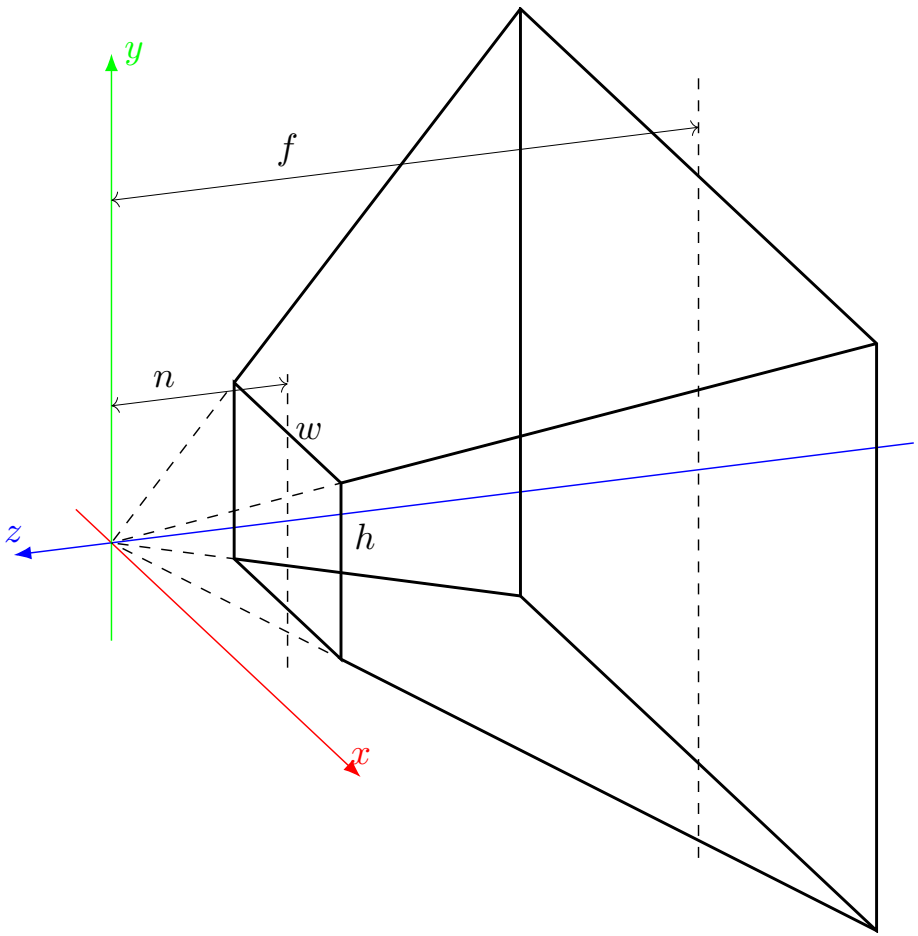
А что мы видим?

Видовая система координат и пространство отсечения при перспективной проекции



Проекционное преобразование





$z = -n$ — ближняя плоскость отсечения

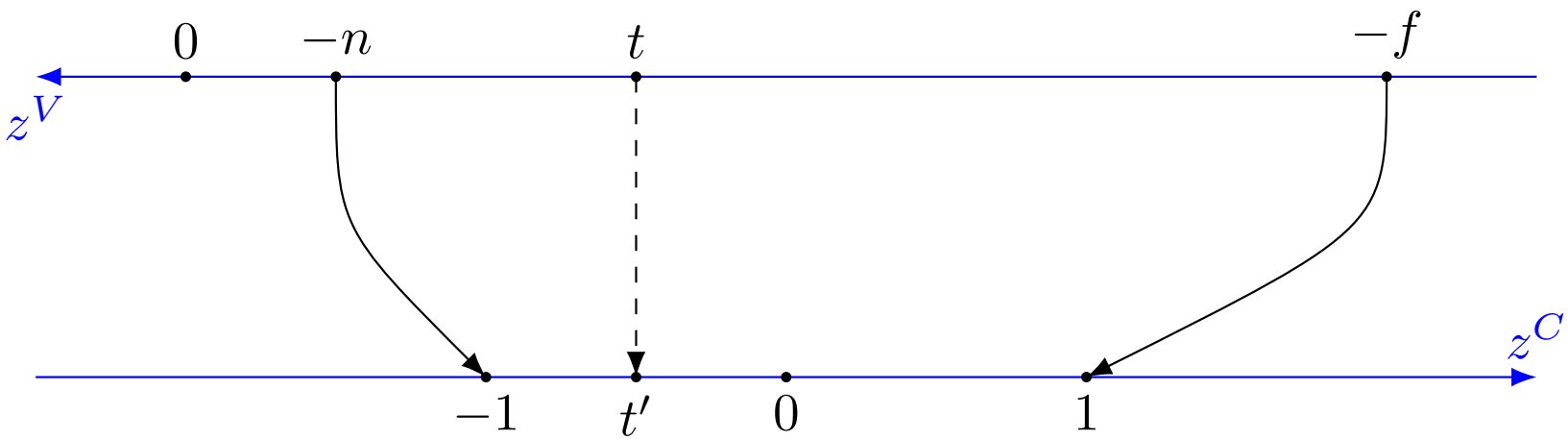
$z = -f$ — дальняя плоскость отсечения

$w \times h$ — ширина и высота «экрана»

Исходя из этих параметров необходимо сконструировать матрицу преобразования вида:

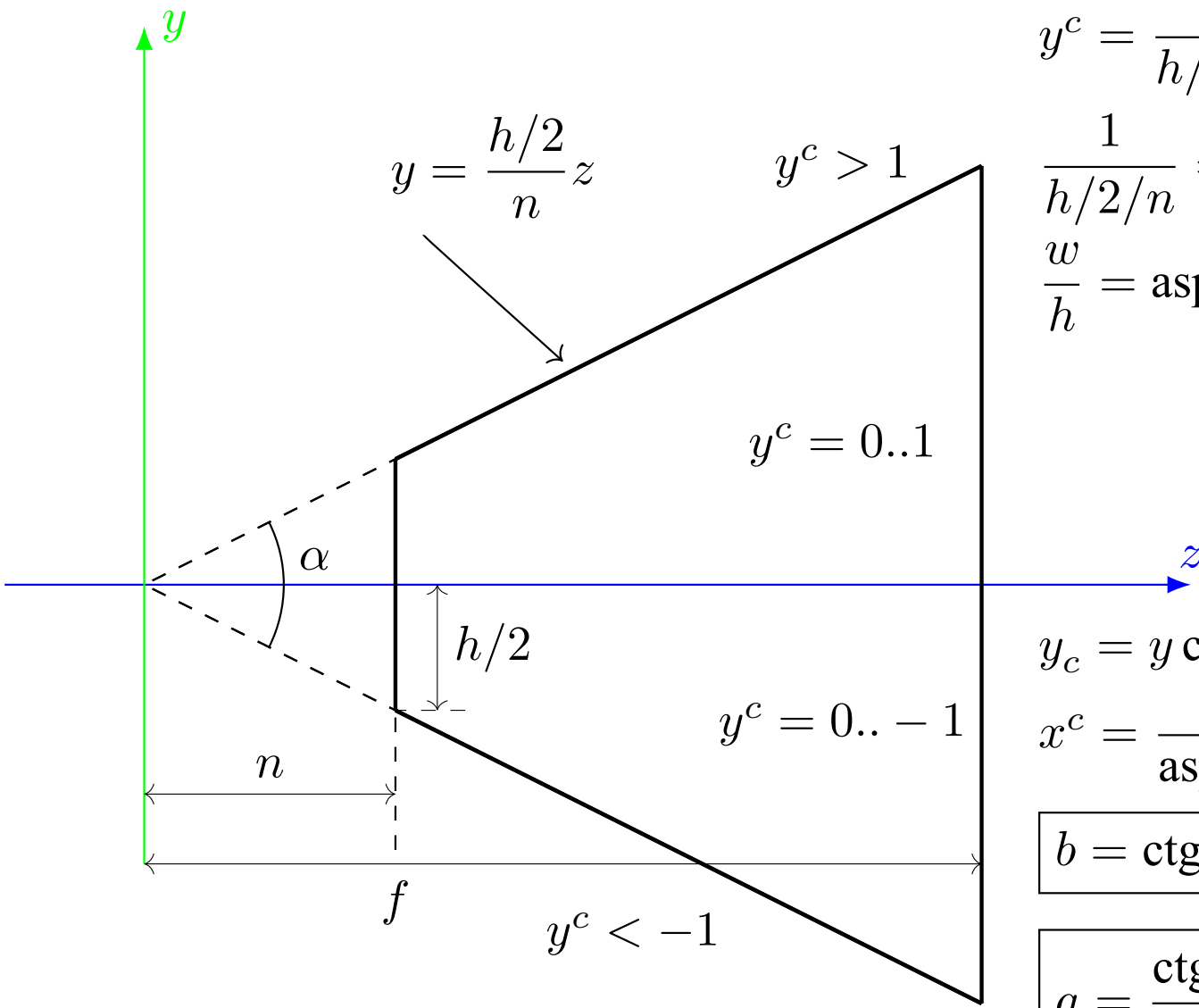
$$\begin{pmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & d \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} =$$

$$= \begin{pmatrix} ax \\ by \\ cz + d \\ -z \end{pmatrix} \Rightarrow \begin{pmatrix} -ax/z \\ -by/z \\ -(cz + d)/z \\ 1 \end{pmatrix}$$



$$z^c = -(cz + d)/z$$

$$\begin{cases} -1 = & (-cn + d)/n \\ 1 = & (-cf + d)/f \end{cases} \Leftrightarrow \begin{cases} \boxed{c = (n + f)/(n - f)} \\ \boxed{d = 2fn/(n - f)} \end{cases}$$



$$y^c = \frac{y}{h/2/n},\ x^c = \frac{x}{w/2/n}$$

$$\frac{1}{h/2/n} = \operatorname{ctg} \alpha/2$$

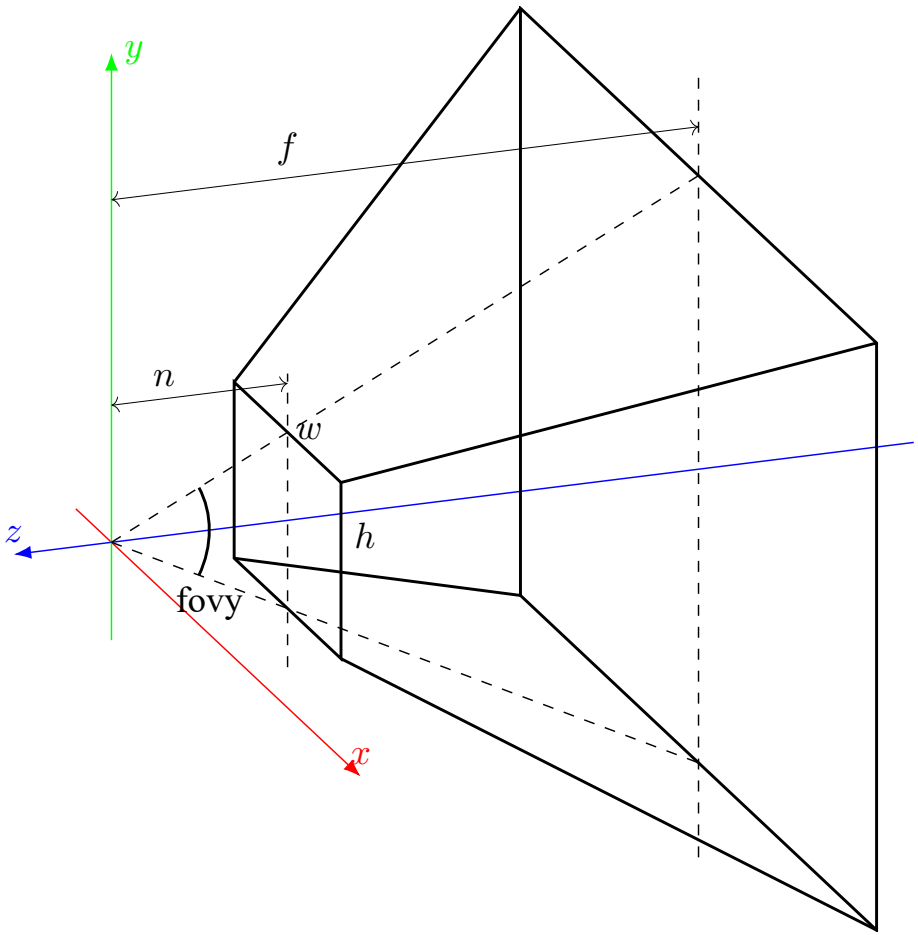
$$\frac{w}{h} = \text{asp}$$

$$y_c = y \operatorname{ctg}(\alpha/2)$$

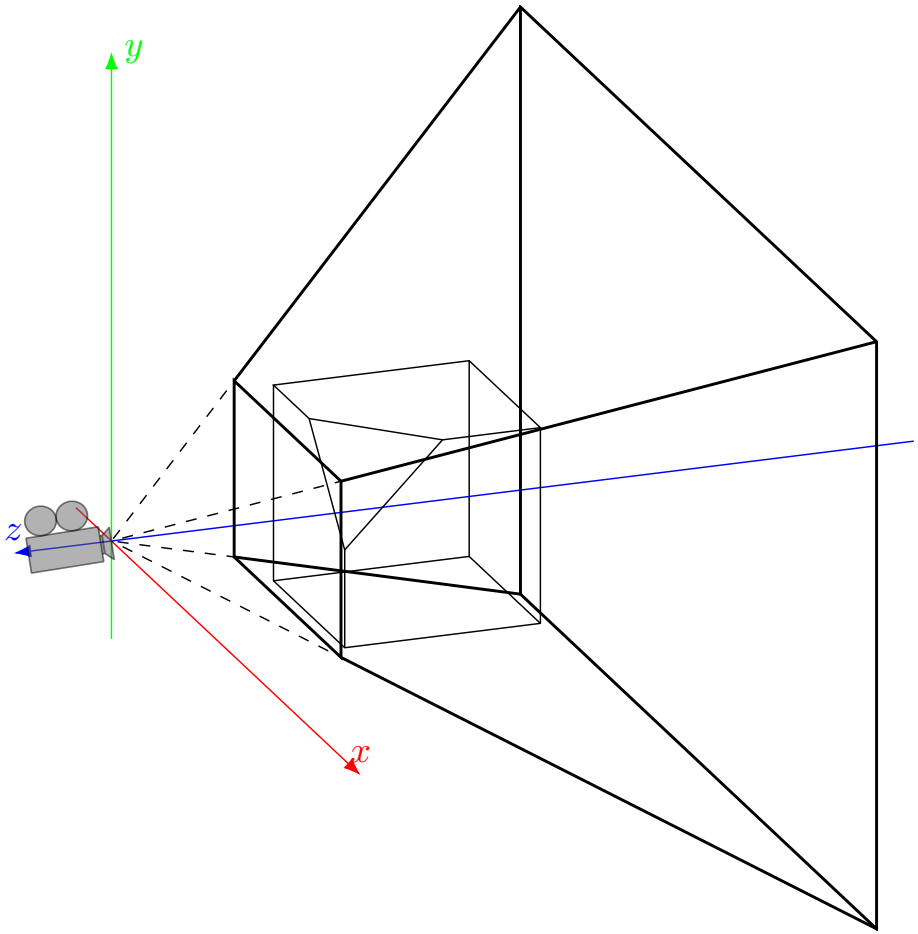
$$x^c = \frac{x}{\text{asp}\, h/2/n} = x \frac{\operatorname{ctg}(\alpha/2)}{\text{asp}}$$

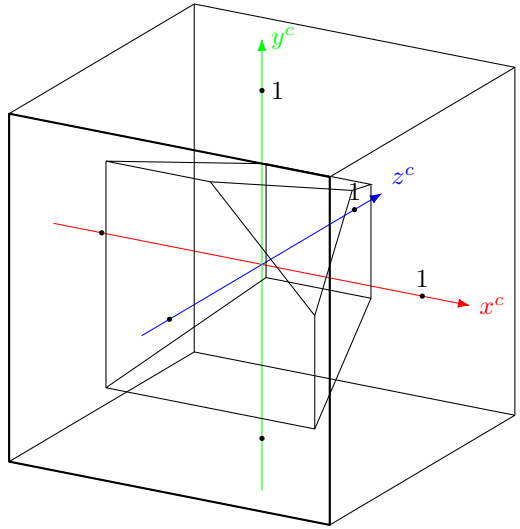
$$\boxed{b = \operatorname{ctg}(\alpha/2)}$$

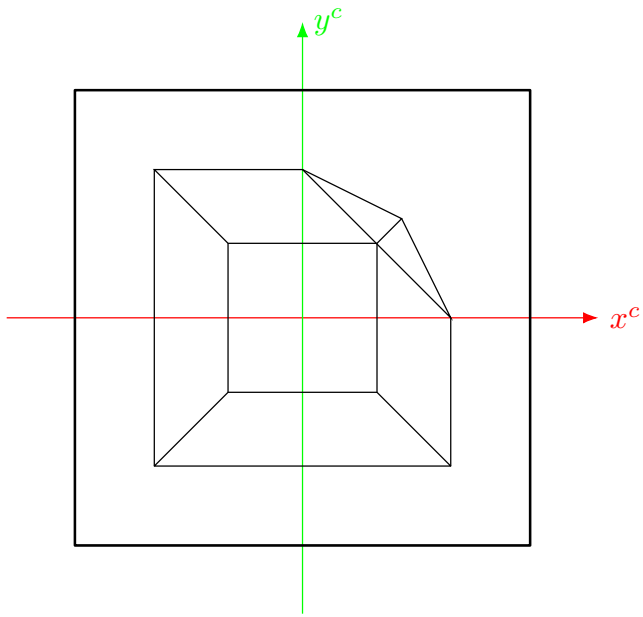
$$\boxed{a = \frac{\operatorname{ctg}(\alpha/2)}{\text{asp}}}$$



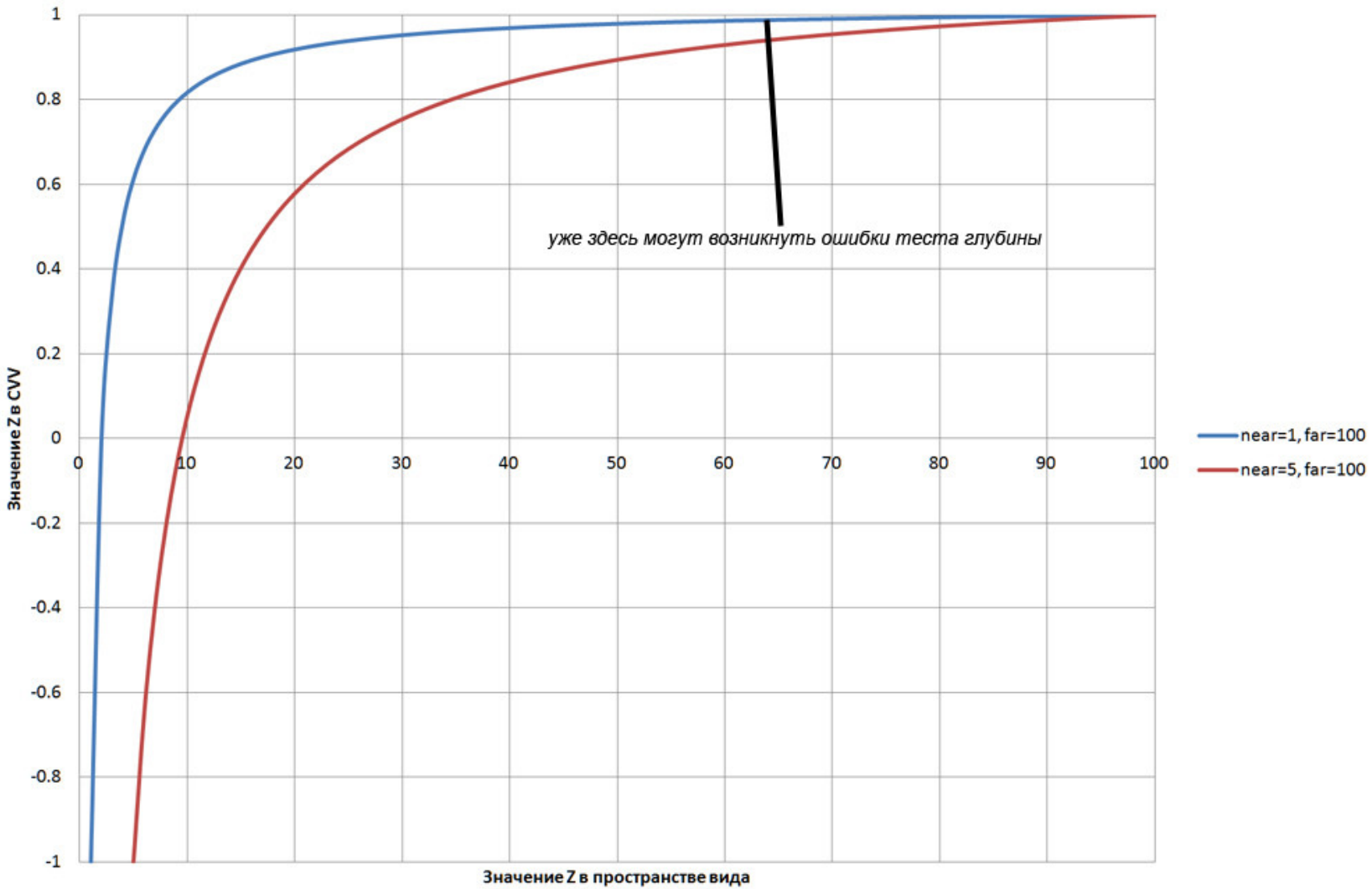
$$\mathbf{P} = \begin{pmatrix} \dfrac{\operatorname{ctg}\left(\dfrac{\text{fovy}}{2}\right)}{\text{asp}} & 0 & 0 & 0 \\ 0 & \operatorname{ctg}\left(\dfrac{\text{fovy}}{2}\right) & 0 & 0 \\ 0 & 0 & \dfrac{n+f}{n-f} & \dfrac{2fn}{n-f} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$


y
z
x

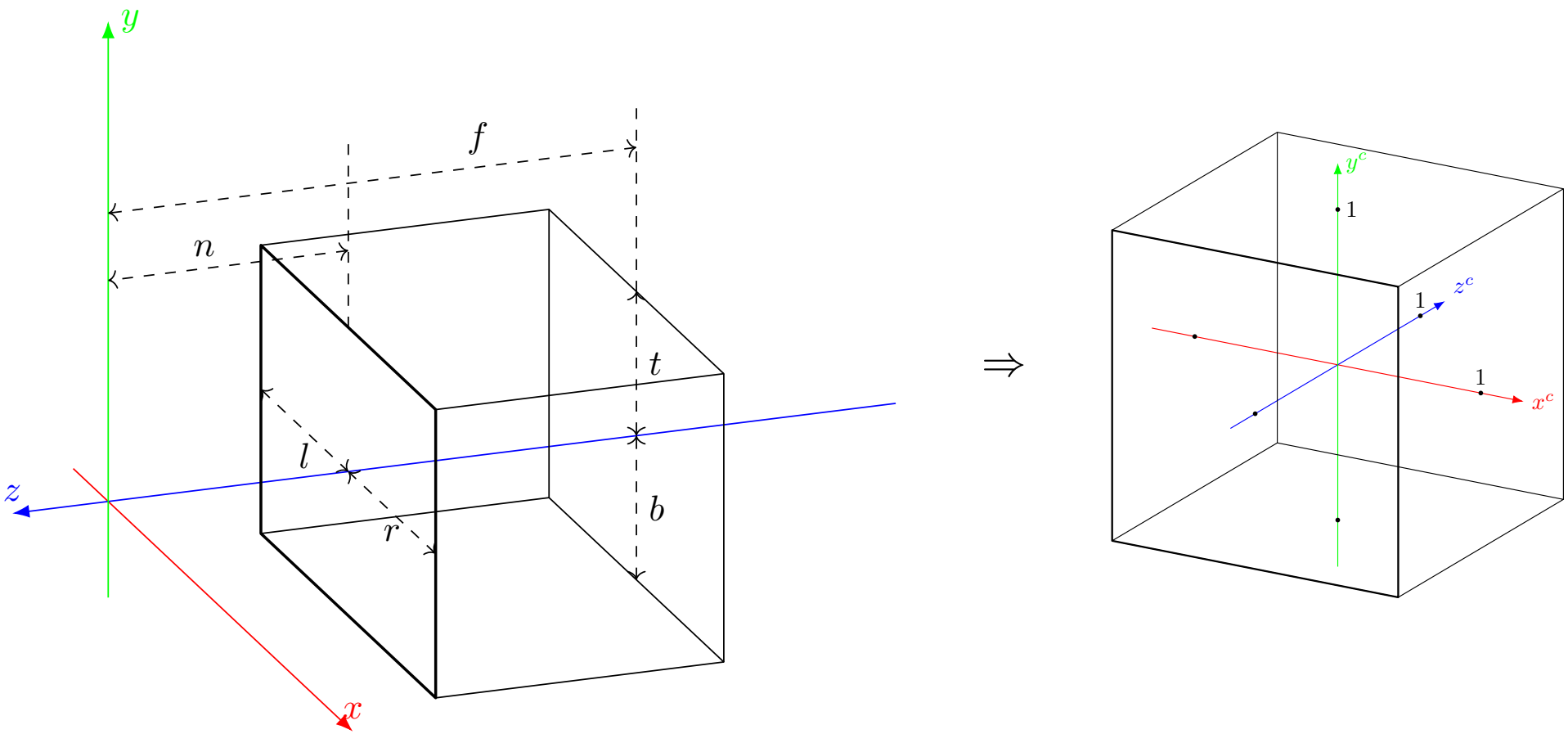

y^c
1
z^c
1
1
x^c


y^c
x^c

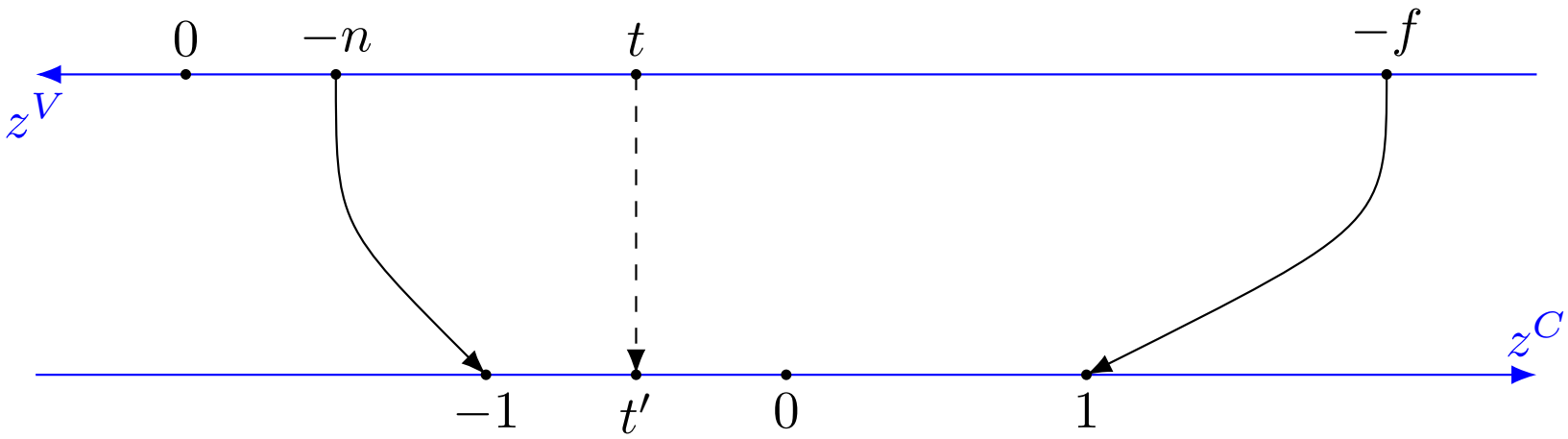
Ошибки теста глубины



Параллельное проекционное преобразование



Матрица параллельной проекции



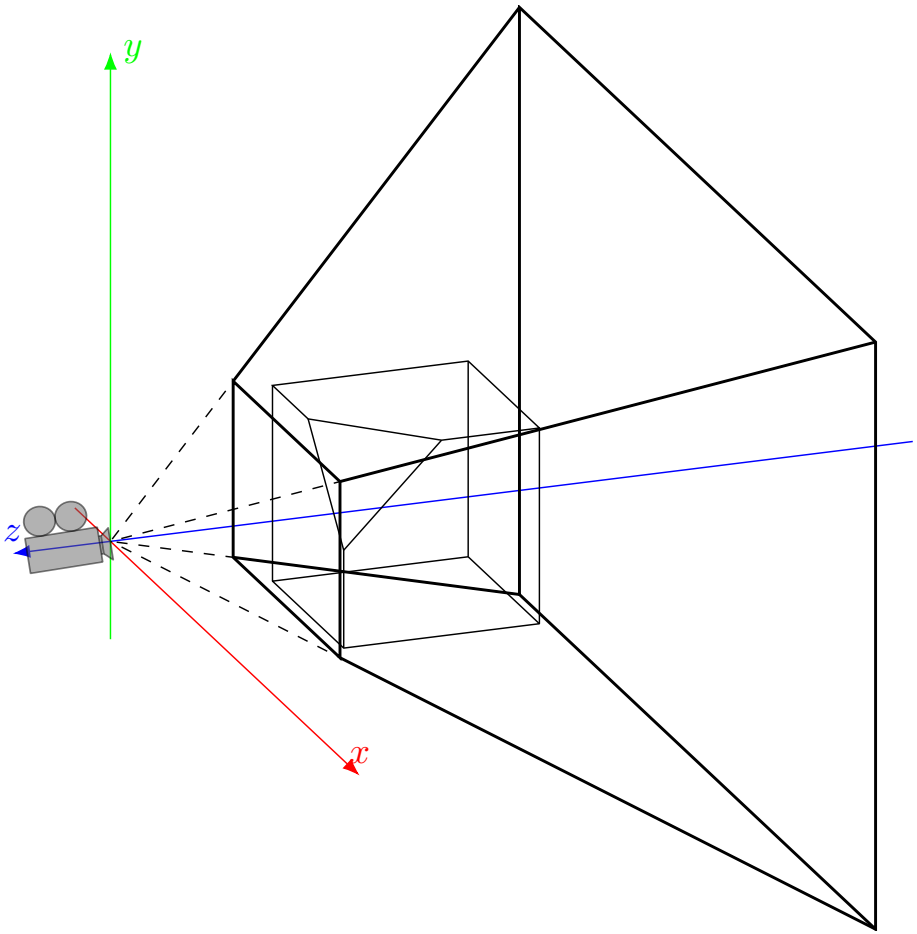
$$\mathbf{P} = \begin{pmatrix} a & 0 & 0 & t_x \\ 0 & b & 0 & t_y \\ 0 & 0 & c & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$z^c = cz + t_z$$

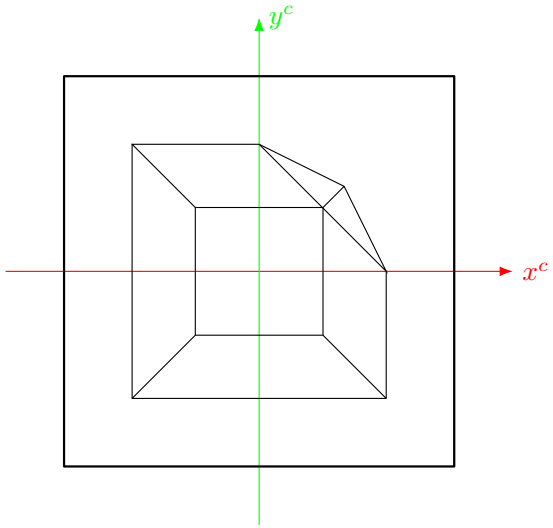
$$\begin{cases} -1 = -cn + t_z \\ 1 = -cf + t_z \end{cases} \Leftrightarrow$$

$$\Leftrightarrow \begin{cases} c = -2/(f-n) \\ t_z = (f+n)/(f-n) \end{cases}$$

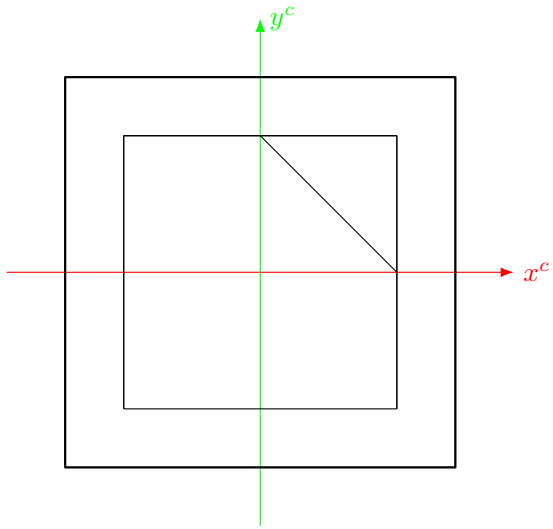
$$\mathbf{P} = \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{-2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



Перспективная проекция



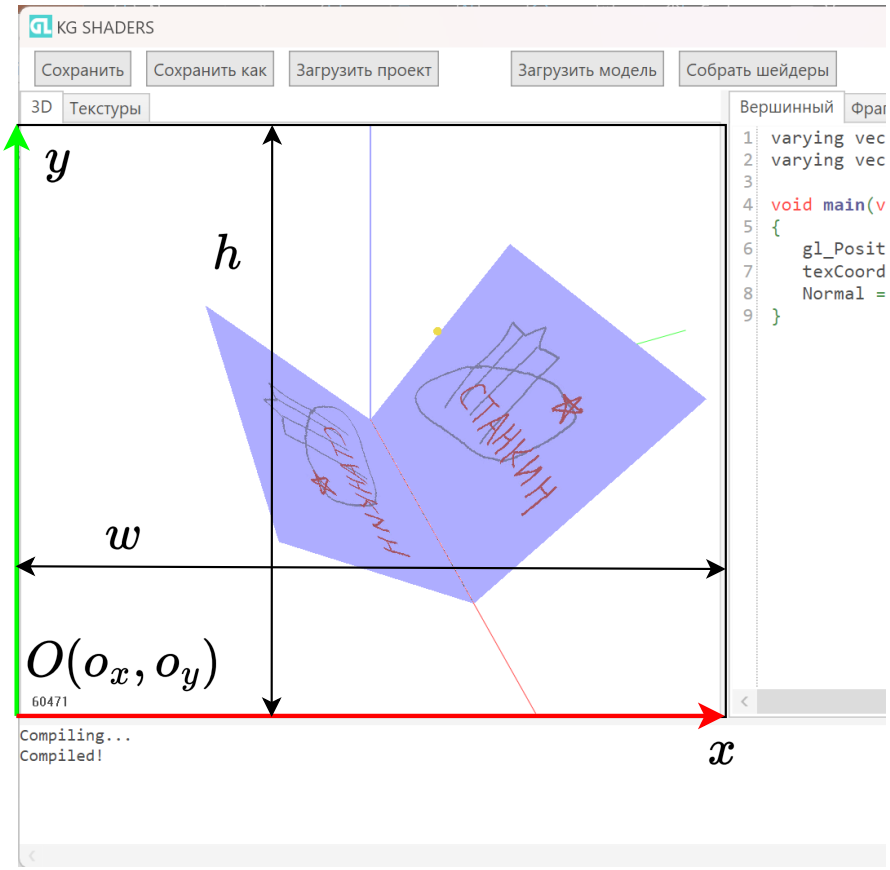
Параллельная проекция



Проекционные преобразования в классическом OpenGL

glMatrixMode(GL_PROJECTIONMATRIX) — устанавливаем режим работы с матрицей проекции. После этого функции для работы с матрицами будут затрагивать именно её.
gluPerspective(fovy,asp,n,f) — установка матрицы перспективной проекции.
glOrtho(l,r,t,b,n,f) — установка матрицы параллельной проекции.
gluOrtho2D(l,r,t,b) — установка матрицы параллельной проекции с $n = -1$, $f = 1$.

Порт просмотра (viewport)



$$(x^c, y^c) \to (x^{vp}, y^{vp})$$

$$x^{vp} = (x^c + 1)\frac{w}{2} + o_x$$

$$y^{vp} = (y^c + 1)\frac{h}{2} + o_y$$

glViewport(ox,oy,w,h) — установка вьюпорта.

Подведём итоги