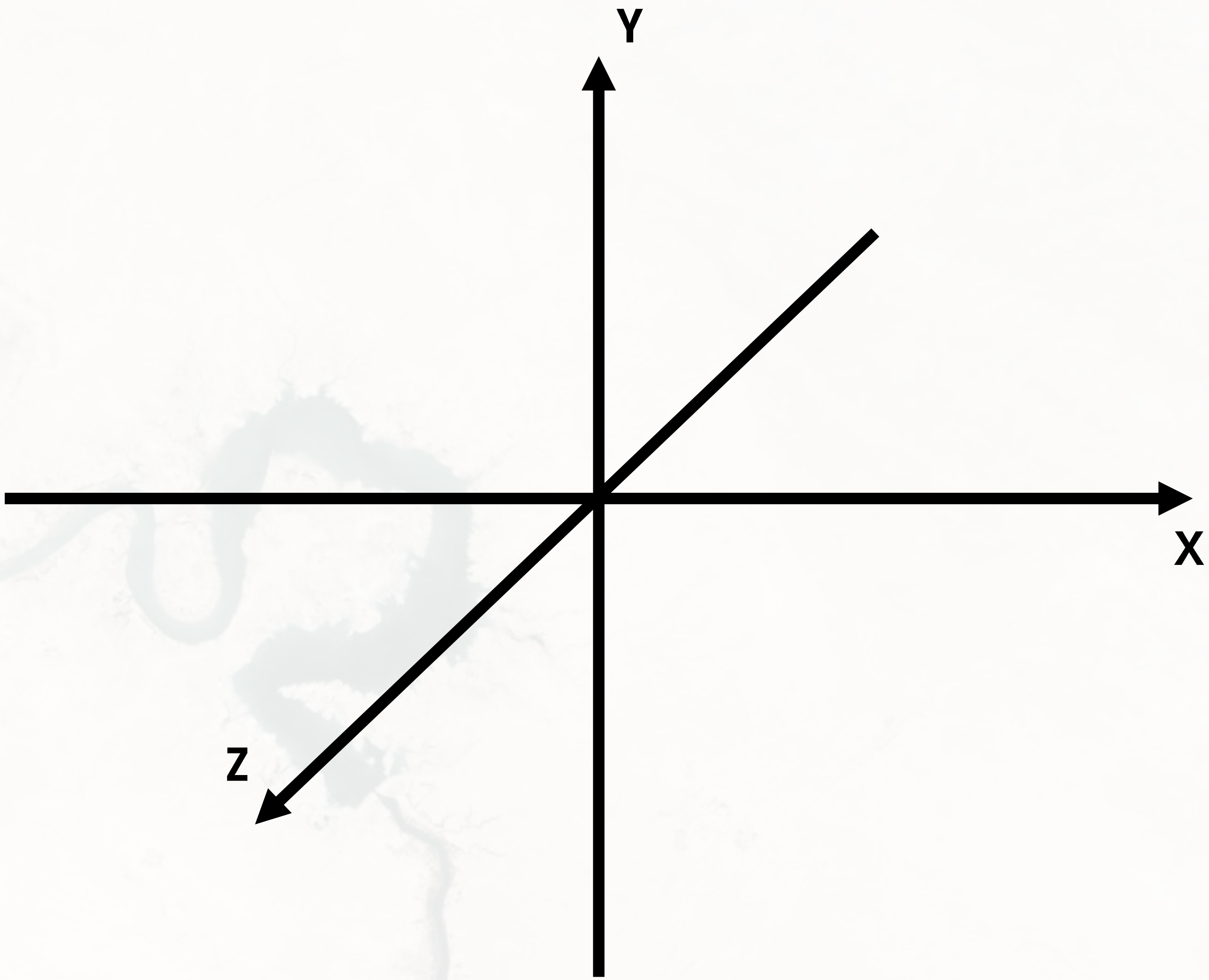
Имитация камеры в WebGL

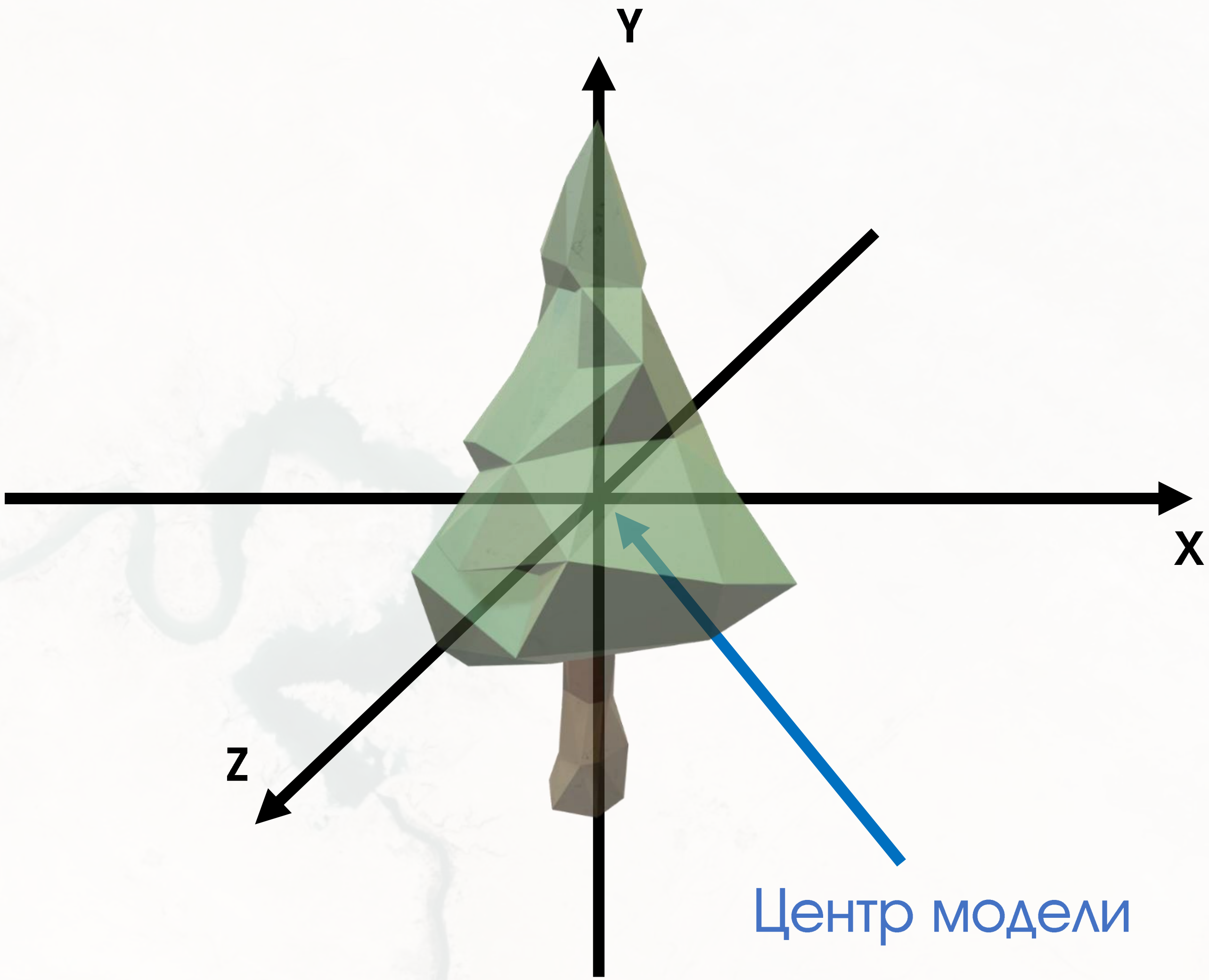
Выполнил студент группы P3417 Дмитрий Плюхин

Введение

Правая система координат



объектные координаты



Преобразования координат в WebGL

$$\begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

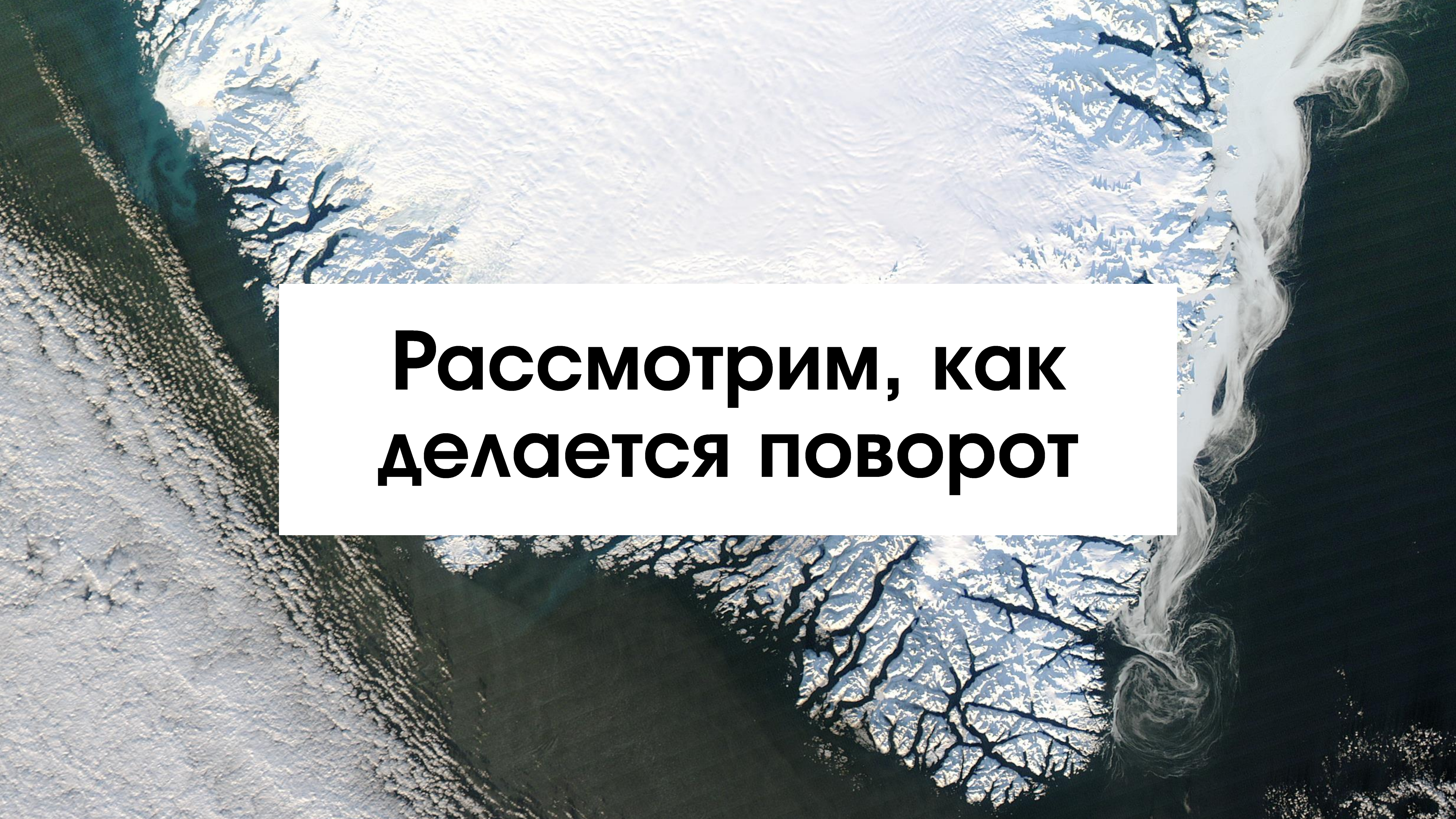
матрица масштабирования

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ tx & ty & tz & 1 \end{bmatrix}$$

матрица смещения

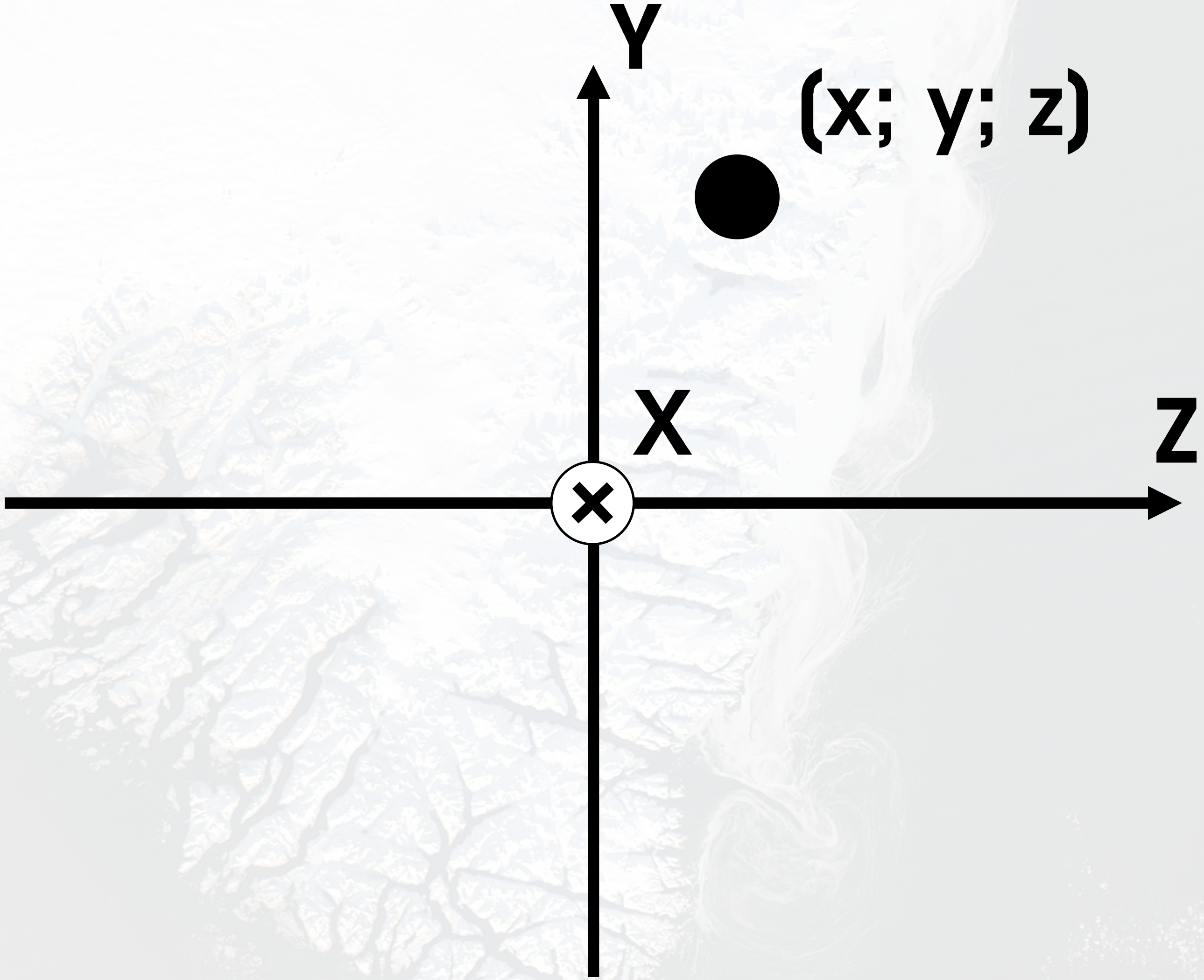
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c & s & 0 \\ 0 & -s & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

матрица **поворота** вокруг оси OX

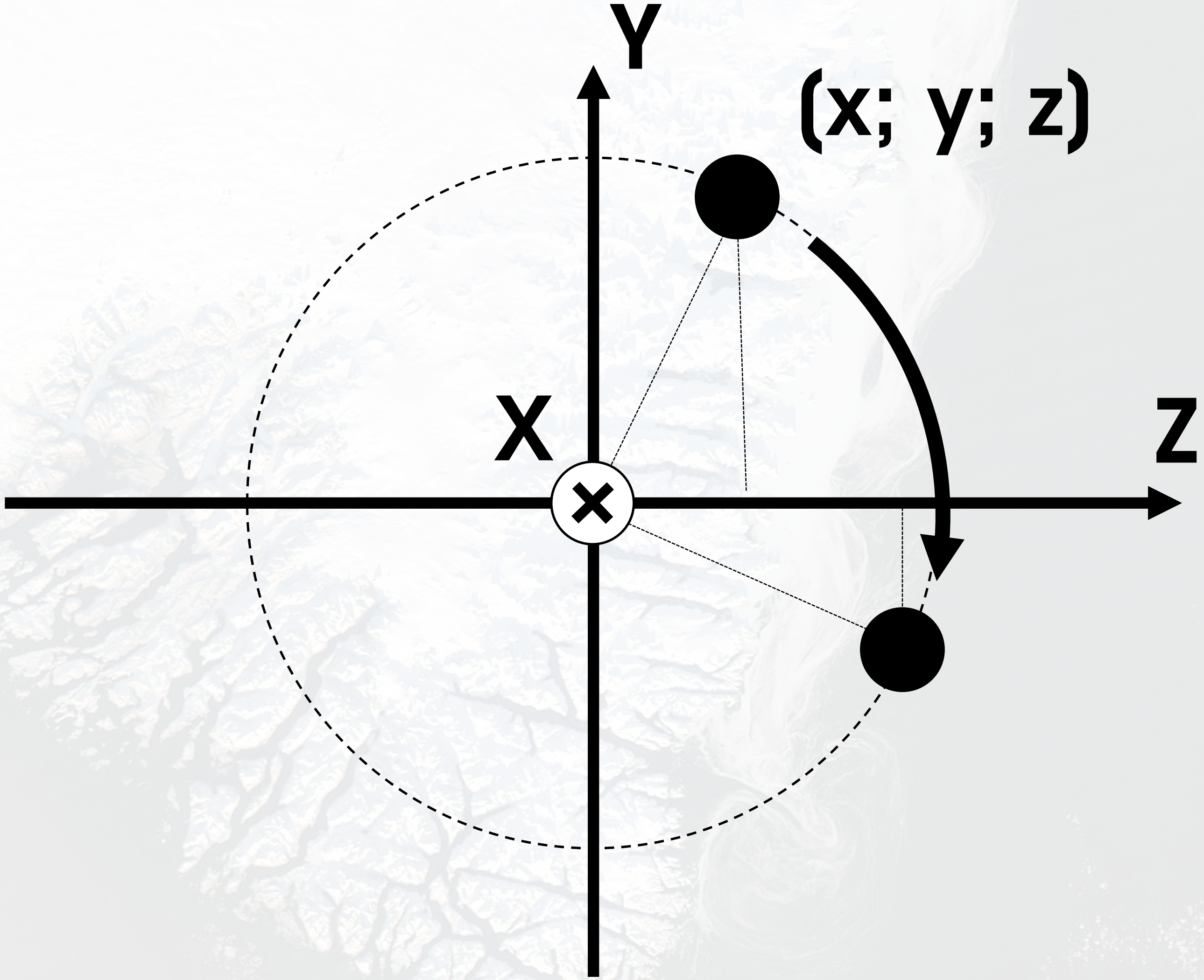


Рассмотрим, как делается поворот

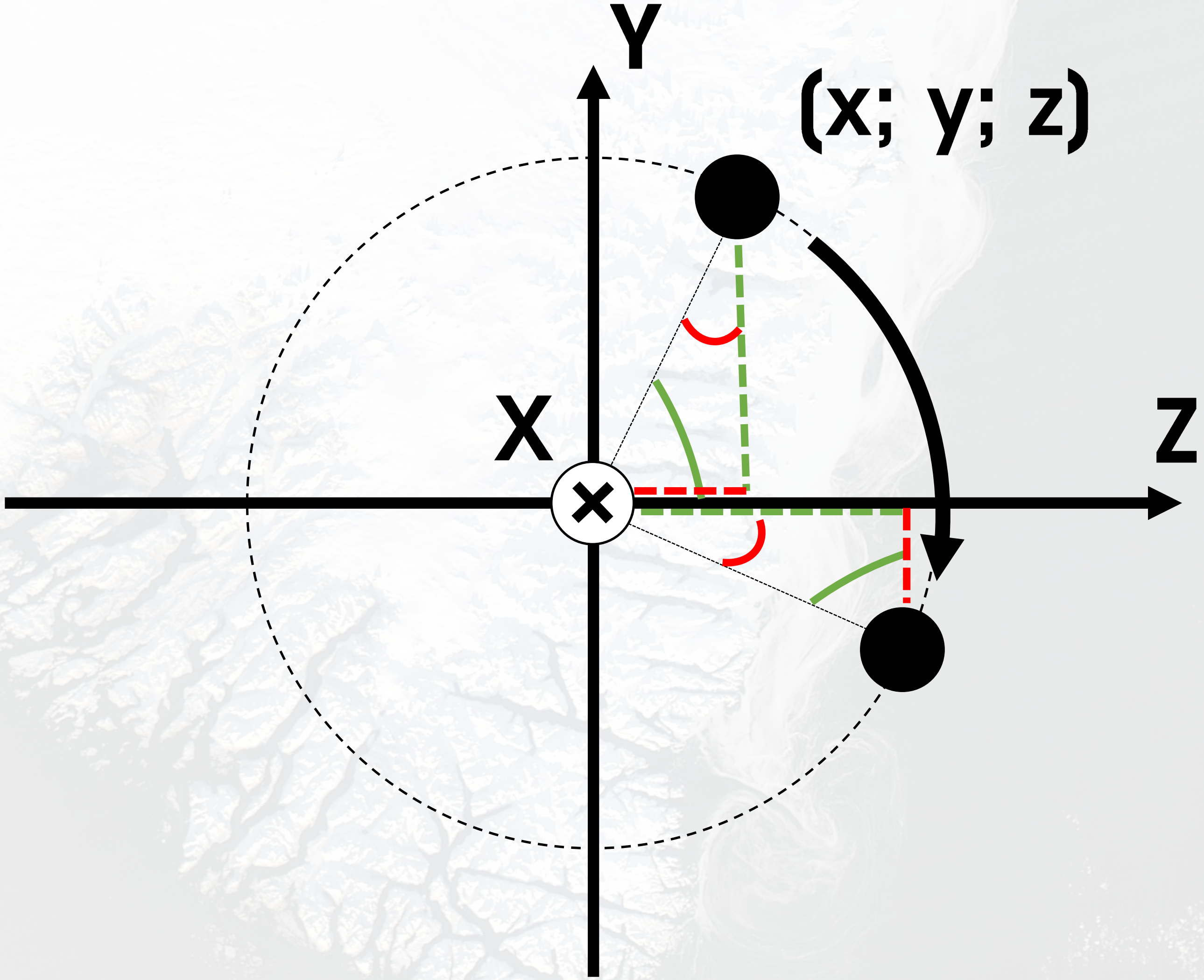
поворот
на 90 градусов



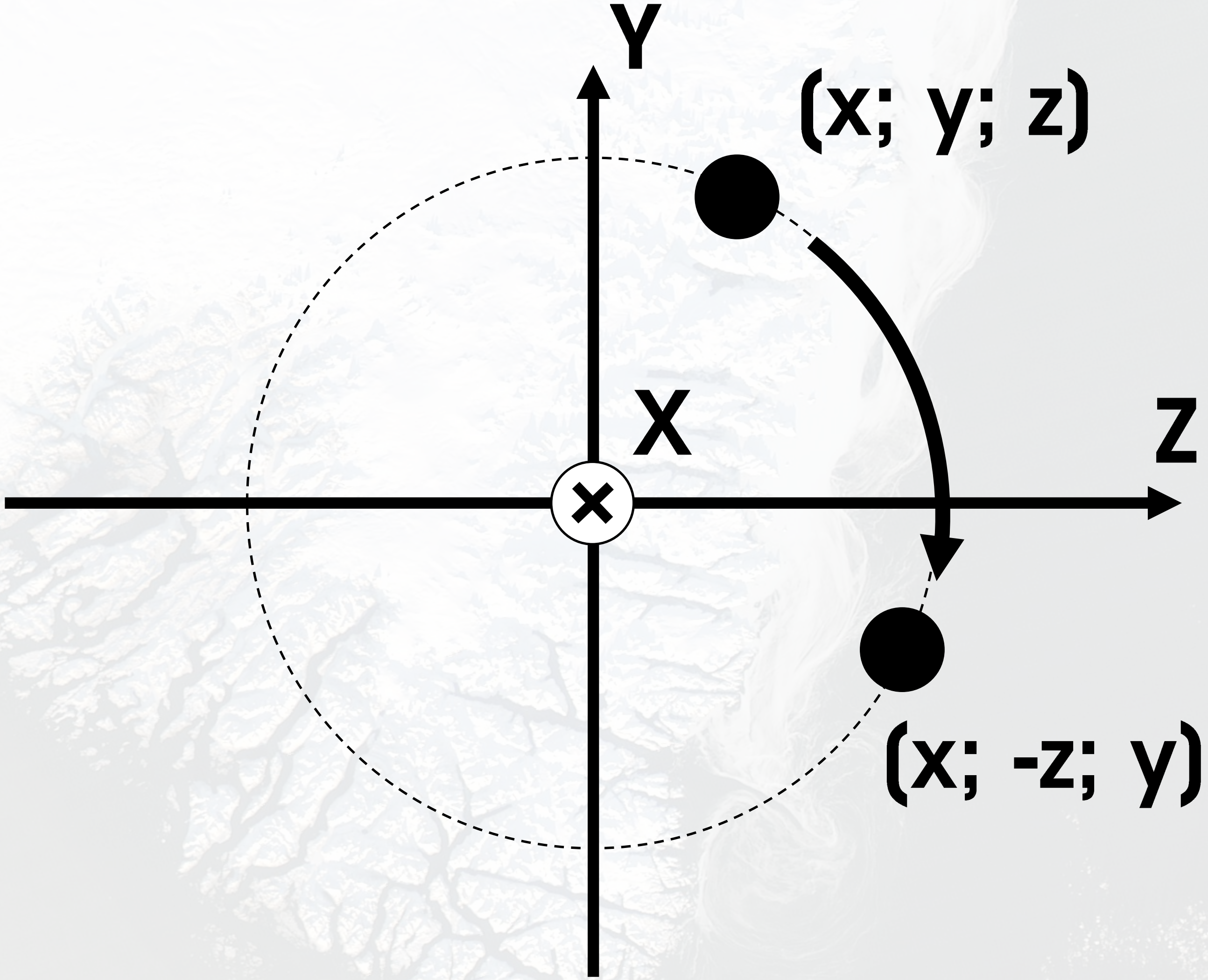
поворот
на 90 градусов



поворот на 90 градусов



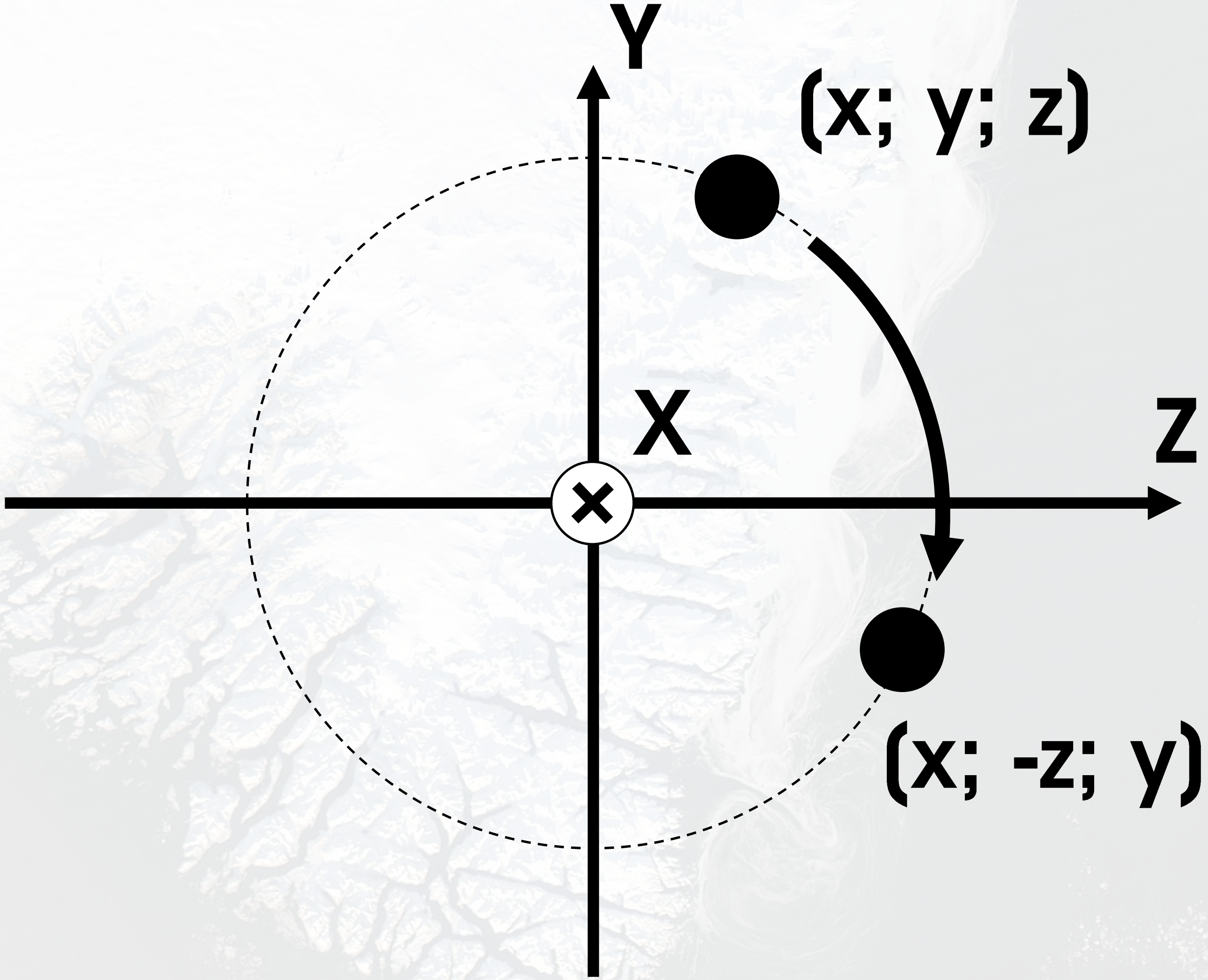
поворот
на 90 градусов



поворот на 90 градусов

y_new = a*y + b*z = -z

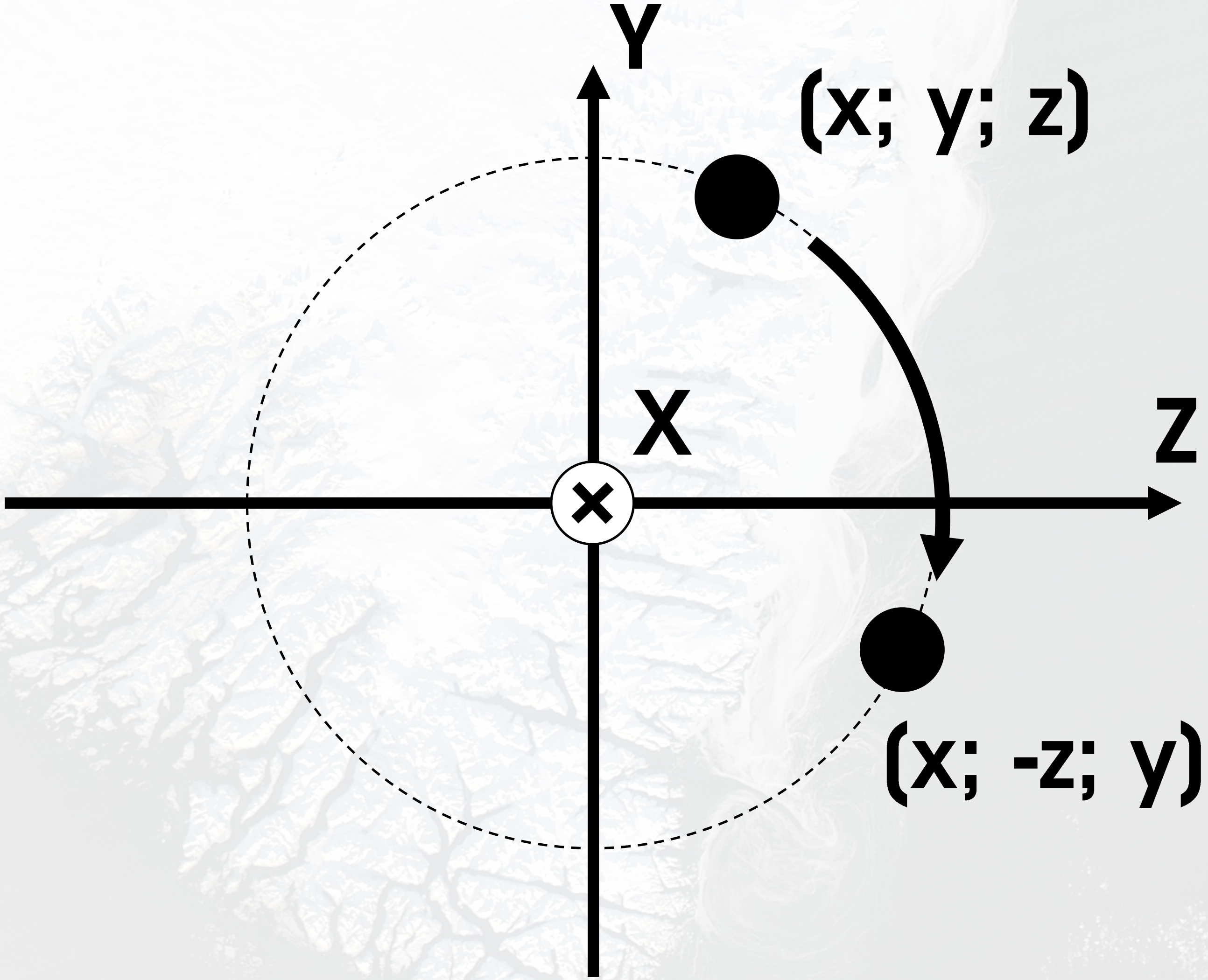
z_new = c*y + d*z = y



поворот на 90 градусов

y_new = a*y + b*z = -z

z_new = c*y + d*z = y



a = 0 b = -1

c = 1 d = 0

поворот на 90 градусов

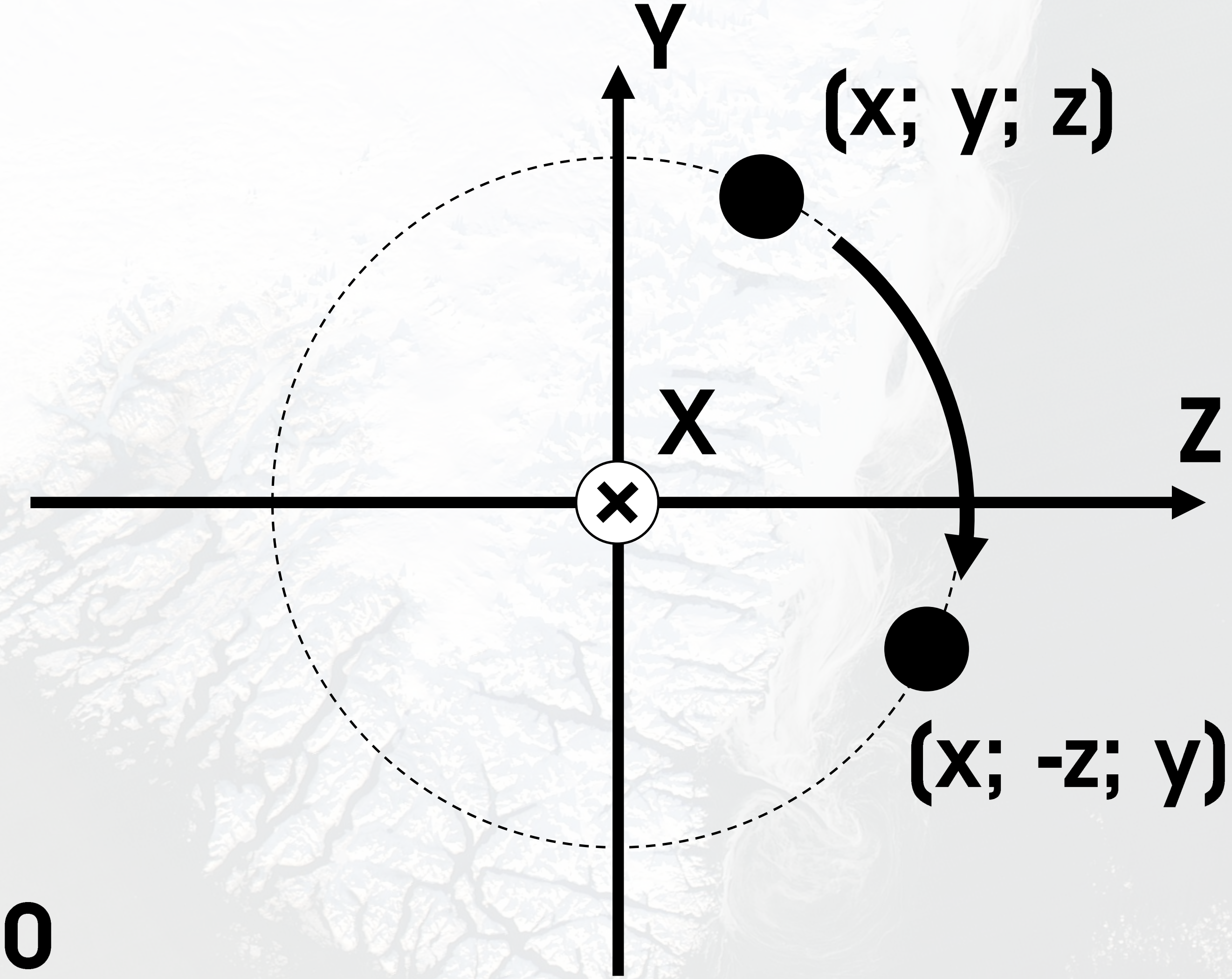
y_new = a*y + b*z = -z

z_new = c*y + d*z = y

a = 0 b = -1

c = 1 d = 0

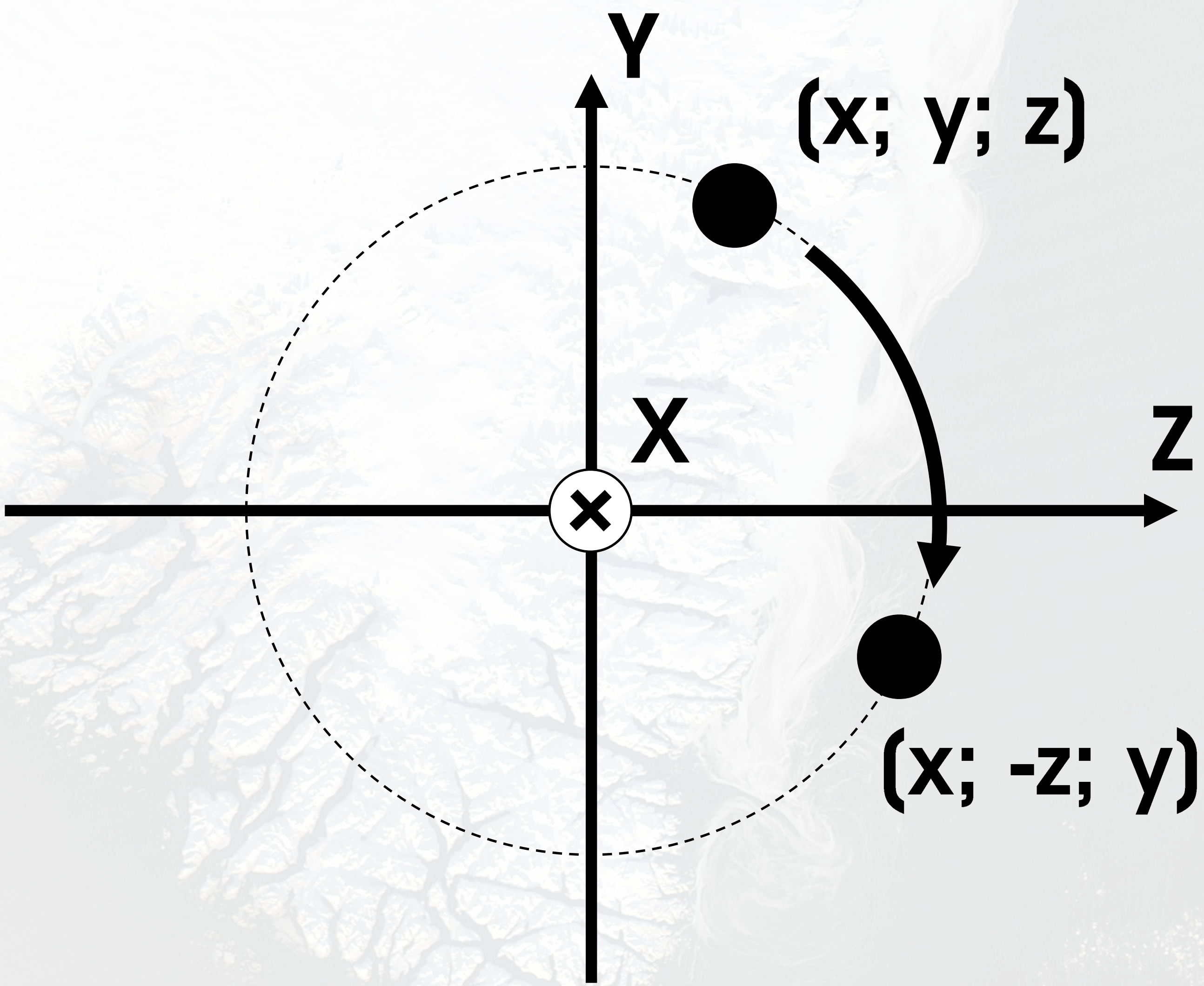
sin(90) = 1   cos(90) = 0



поворот
на 90 градусов

y_new = a*y + b*z = -z

z_new = c*y + d*z = y



a = 0 b = -1

c = 1 d = 0

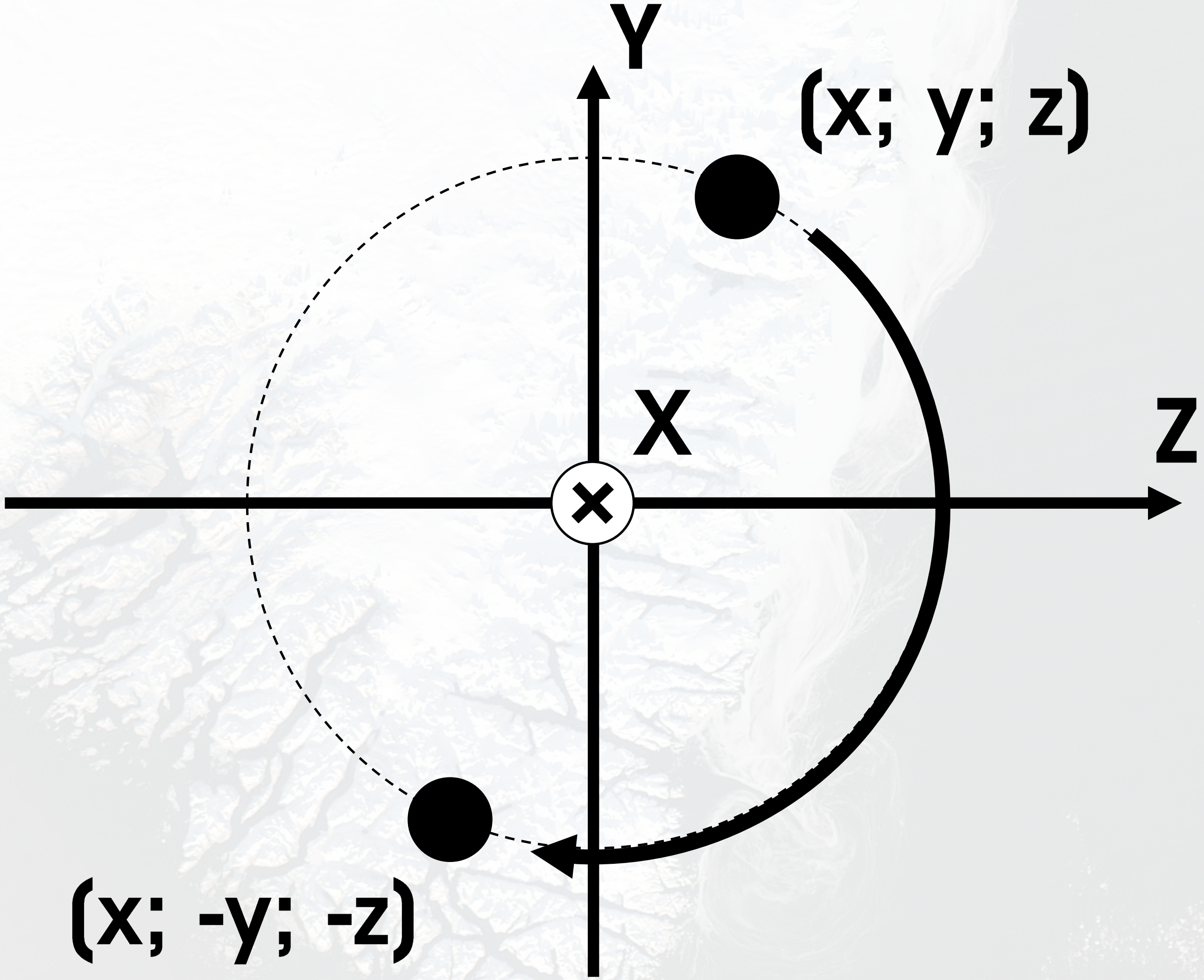
sin(90) = 1

cos(90) = 0

a = ±cos(α) b = -sin(α)

c = sin(α) d = ±cos(α)

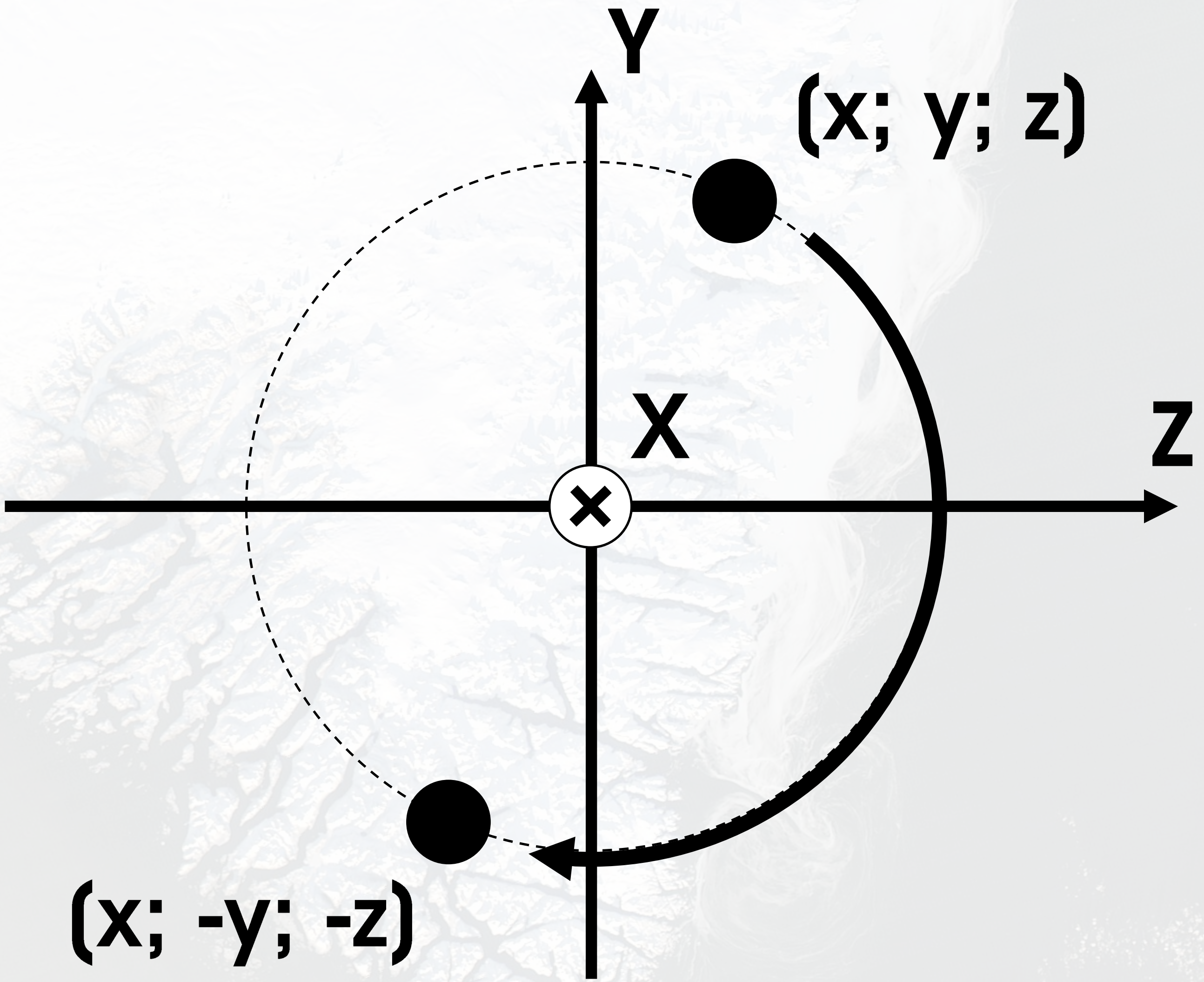
поворот
на 180 градусов



поворот на 180 градусов

y_new = a*y - s*z = -y

z_new = s*y + d*z = -z

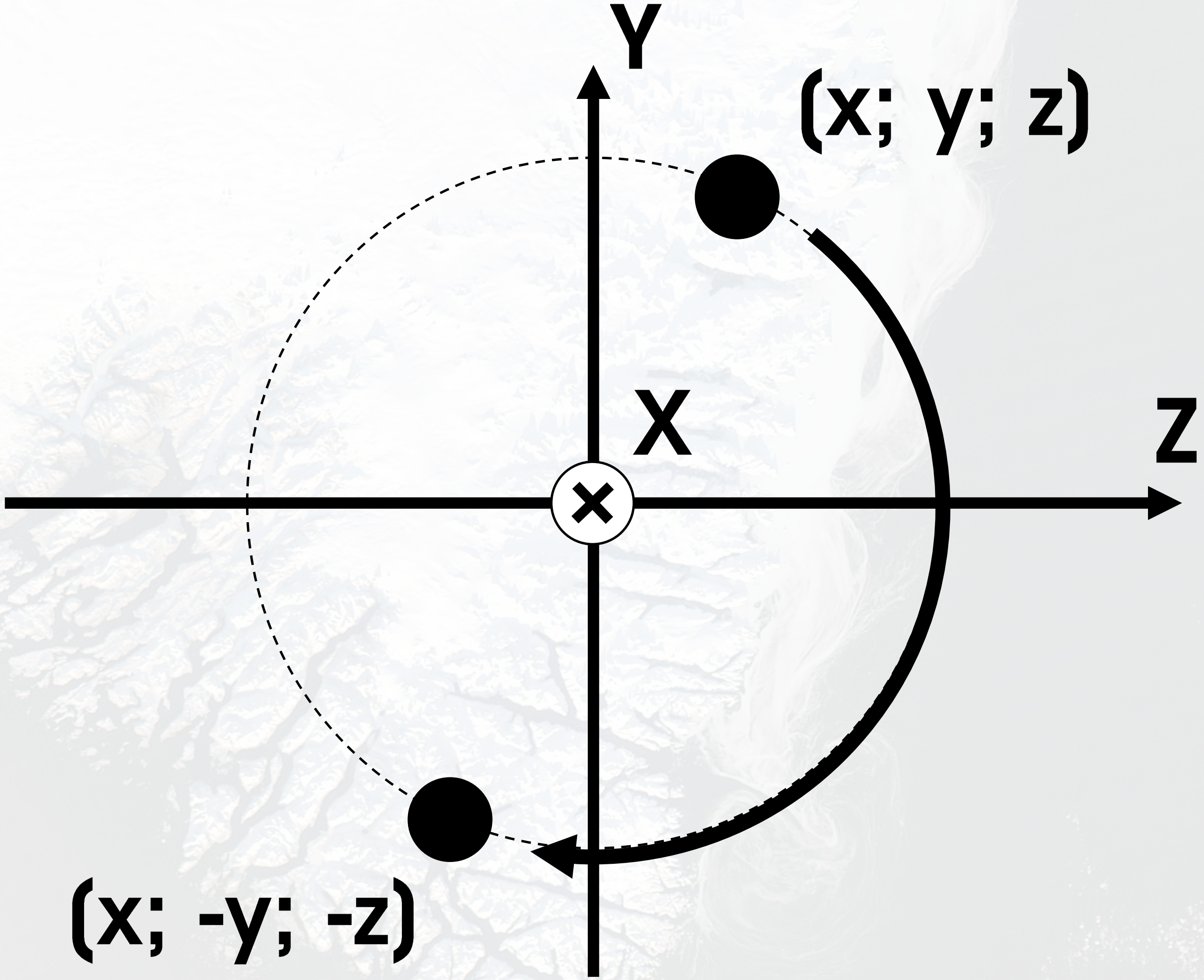


поворот на 180 градусов

y_new = a*y - s*z = -y

z_new = s*y + d*z = -z

d = -1 a = -1



поворот на 180 градусов

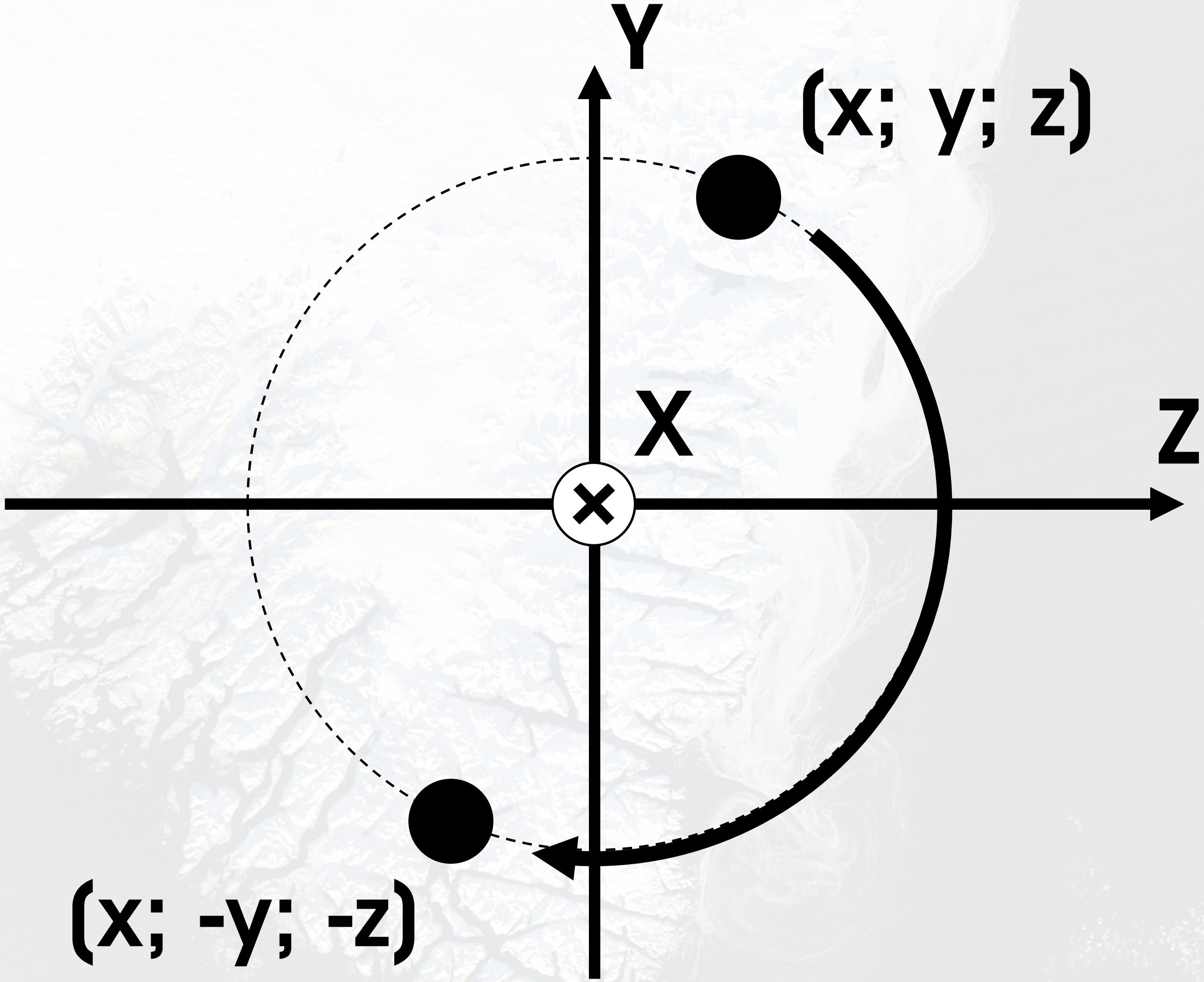
y_new = a*y - s*z = -y

z_new = s*y + d*z = -z

a = -1

d = -1

cos(180) = -1



поворот на 180 градусов

y_new = a*y - s*z = -y

z_new = s*y + d*z = -z

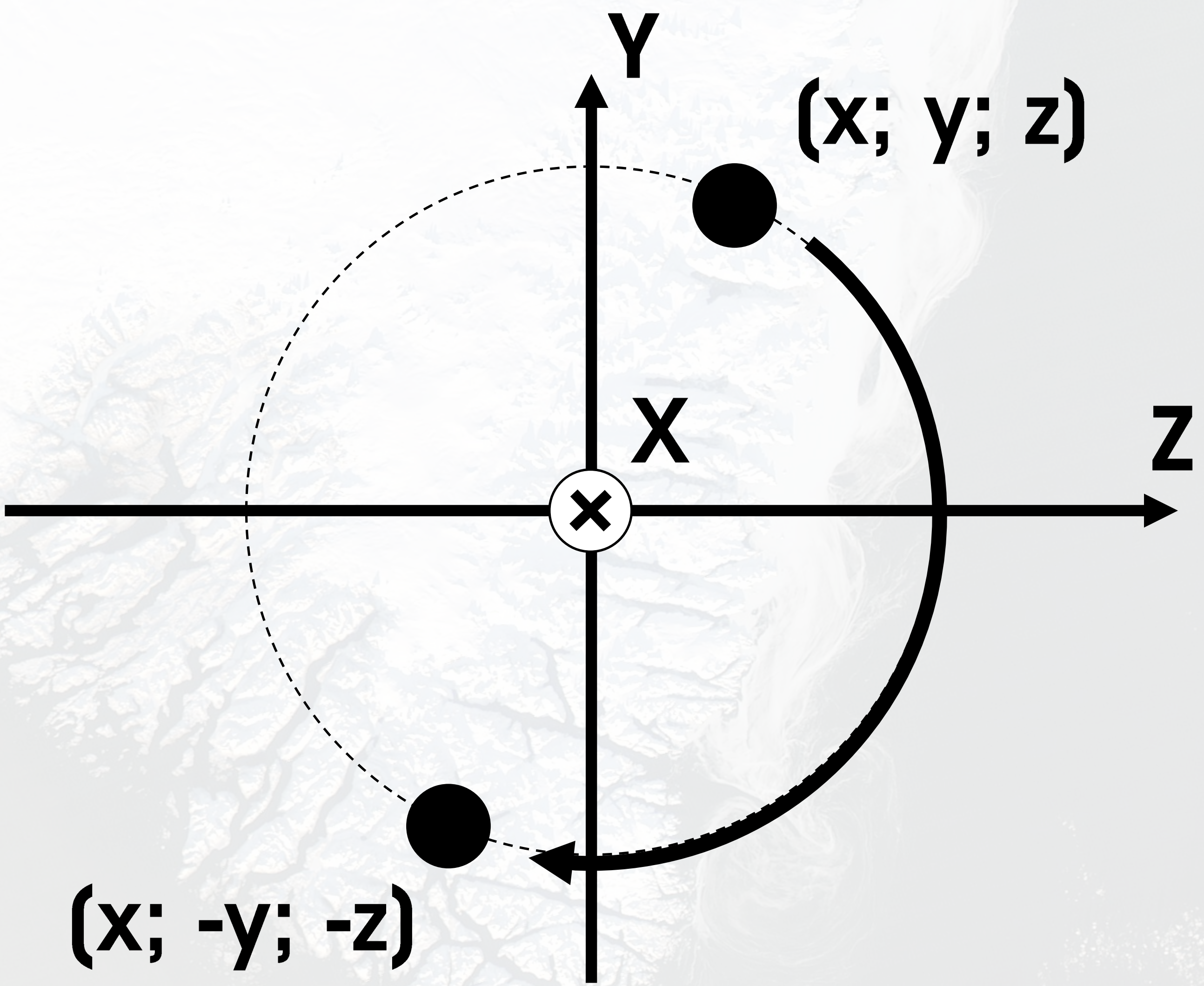
a = -1

d = -1

cos(180) = -1

a = cos(α)

d = cos(α)



y_new = c*y - s*z
z_new = s*y + c*z

поворот вокруг оси OX

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c & s & 0 \\ 0 & -s & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Аналогично делается поворот вокруг других осей

$$\begin{bmatrix} c & 0 & -s & 0 \\ 0 & 1 & 0 & 0 \\ s & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

матрица поворота вокруг оси OY

$c = \cos(\alpha)$ $s = \sin(\alpha)$

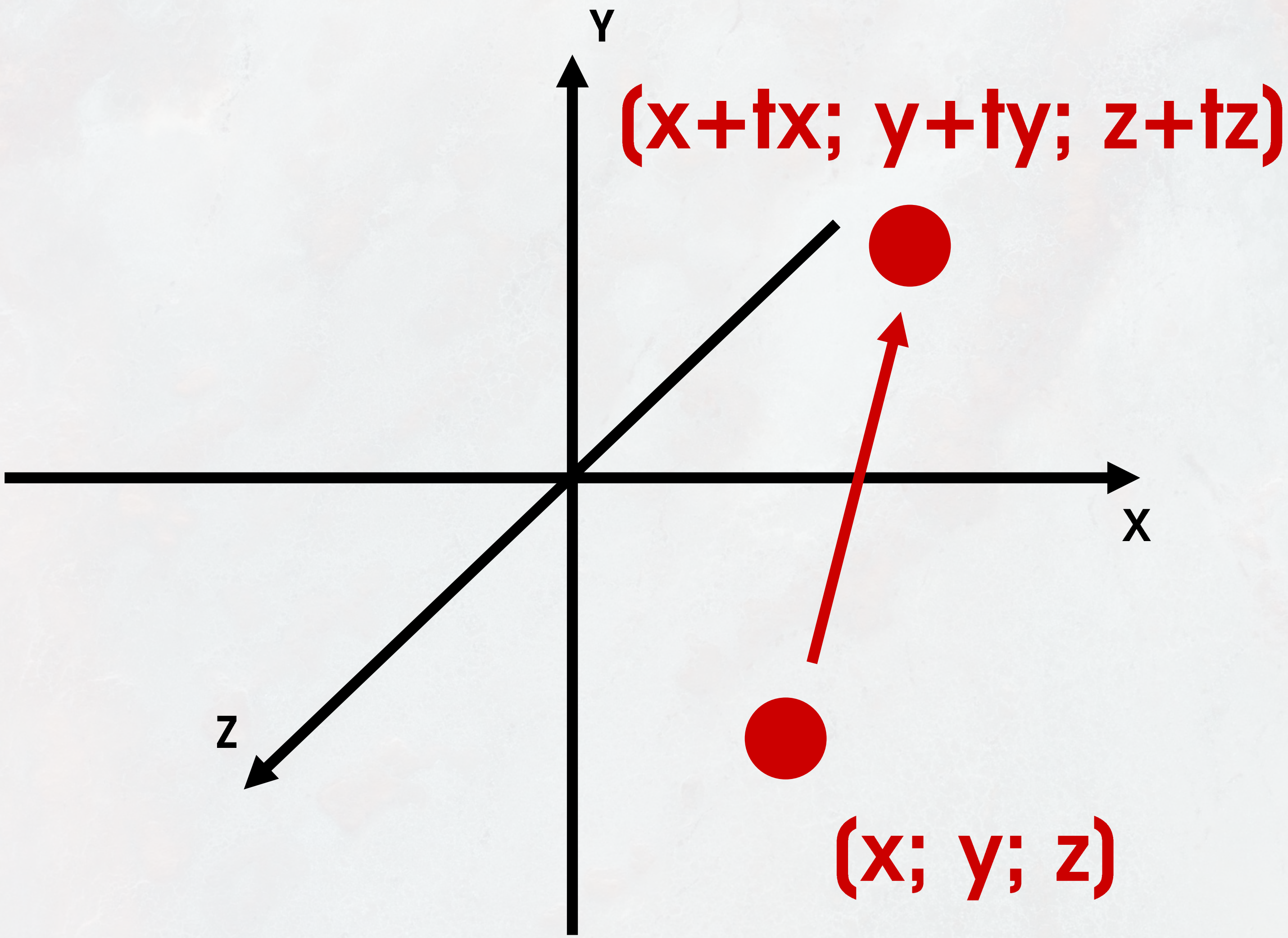
$$\begin{bmatrix} c & s & 0 & 0 \\ -s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

c = cos(α) s = sin(α)

матрица поворота вокруг оси OZ

Примеры преобразований

смещение точки

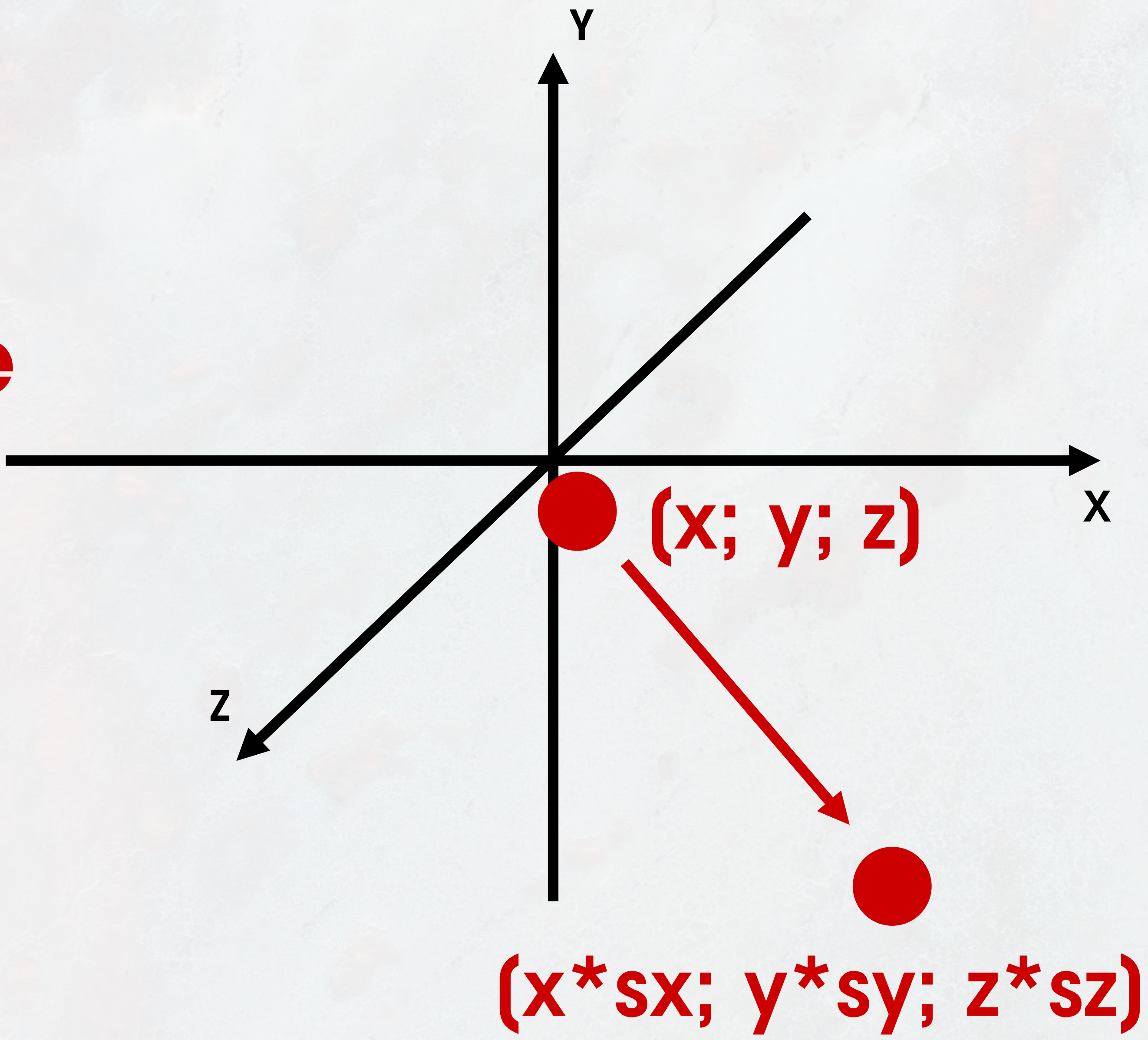


смещение
точки

$$[x \quad y \quad z \quad 1] * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ tx & ty & tz & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} x*1+y*0+z*0+1*tx \\ x*0+y*1+z*0+1*ty \\ x*0+y*0+z*1+1*tz \\ x*0+y*0+z*0+1*1 \end{bmatrix}^T = [x+tx \quad y+ty \quad z+tz \quad 1]$$

масштабирование
координат
точки



масштабирование координат точки

$$[x \quad y \quad z \quad 1] * \begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} x * sx + y * 0 + z * 0 + 1 * 0 \\ x * 0 + y * sy + z * 0 + 1 * 0 \\ x * 0 + y * 0 + z * sz + 1 * 0 \\ x * 0 + y * 0 + z * 0 + 1 * 1 \end{bmatrix}^T = [x * sx \quad y * sy \quad z * sz \quad 1]$$

$$\begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ tx & ty & tz & 1 \end{bmatrix} =$$

$$\begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ tx & ty & tz & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} \boldsymbol{sx} * 1 + 0 * 0 + 0 * 0 + 0 * tx & sx * 0 + 0 * 1 + 0 * 0 + 0 * ty & sx * 0 + 0 * 0 + 0 * 1 + 0 * tz & sx * 0 + 0 * 0 + 0 * 0 + 0 * 1 \\ 0 * 1 + sy * 0 + 0 * 0 + 0 * tx & 0 * 0 + \boldsymbol{sy} * 1 + 0 * 0 + 0 * ty & 0 * 0 + sy * 0 + 0 * 1 + 0 * tz & 0 * 0 + sy * 0 + 0 * 0 + 0 * 1 \\ 0 * 1 + 0 * 0 + sz * 0 + 0 * tx & 0 * 0 + 0 * 1 + sz * 0 + 0 * ty & 0 * 0 + 0 * 0 + \boldsymbol{sz} * 1 + 0 * tz & 0 * 0 + 0 * 0 + sz * 0 + 0 * 1 \\ 0 * 1 + 0 * 0 + 0 * 0 + 1 * \boldsymbol{tx} & 0 * 0 + 0 * 1 + 0 * 0 + 1 * \boldsymbol{ty} & 0 * 0 + 0 * 0 + 0 * 1 + 1 * \boldsymbol{tz} & 0 * 0 + 0 * 0 + 0 * 0 + 1 * 1 \end{bmatrix} =$$

$$\begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ tx & ty & tz & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} \boldsymbol{sx} * 1 + 0 * 0 + 0 * 0 + 0 * tx & sx * 0 + 0 * 1 + 0 * 0 + 0 * ty & sx * 0 + 0 * 0 + 0 * 1 + 0 * tz & sx * 0 + 0 * 0 + 0 * 0 + 0 * 1 \\ 0 * 1 + sy * 0 + 0 * 0 + 0 * tx & 0 * 0 + \boldsymbol{sy} * 1 + 0 * 0 + 0 * ty & 0 * 0 + sy * 0 + 0 * 1 + 0 * tz & 0 * 0 + sy * 0 + 0 * 0 + 0 * 1 \\ 0 * 1 + 0 * 0 + sz * 0 + 0 * tx & 0 * 0 + 0 * 1 + sz * 0 + 0 * ty & 0 * 0 + 0 * 0 + \boldsymbol{sz} * 1 + 0 * tz & 0 * 0 + 0 * 0 + sz * 0 + 0 * 1 \\ 0 * 1 + 0 * 0 + 0 * 0 + 1 * \boldsymbol{tx} & 0 * 0 + 0 * 1 + 0 * 0 + 1 * \boldsymbol{ty} & 0 * 0 + 0 * 0 + 0 * 1 + 1 * \boldsymbol{tz} & 0 * 0 + 0 * 0 + 0 * 0 + 1 * 1 \end{bmatrix} =$$

$$= \begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ tx & ty & tz & 1 \end{bmatrix}$$

$$\begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ tx & ty & tz & 1 \end{bmatrix}$$

матрица
смещения и масштабирования

смещение и масштабирование координат точки

$$[x \quad y \quad z \quad 1] * \begin{bmatrix} sx & 0 & 0 & 0 \\ 0 & sy & 0 & 0 \\ 0 & 0 & sz & 0 \\ tx & ty & tz & 1 \end{bmatrix} =$$

$$= \begin{bmatrix} x * sx + y * 0 + z * 0 + 1 * tx \\ x * 0 + y * sy + z * 0 + 1 * ty \\ x * 0 + y * 0 + z * sz + 1 * tz \\ x * 0 + y * 0 + z * 0 + 1 * 1 \end{bmatrix}^T =$$

$$=[x * sx + tx \quad y * sy + ty \quad z * sz + tz \quad 1]$$

сначала масштабирование, потом - смещение

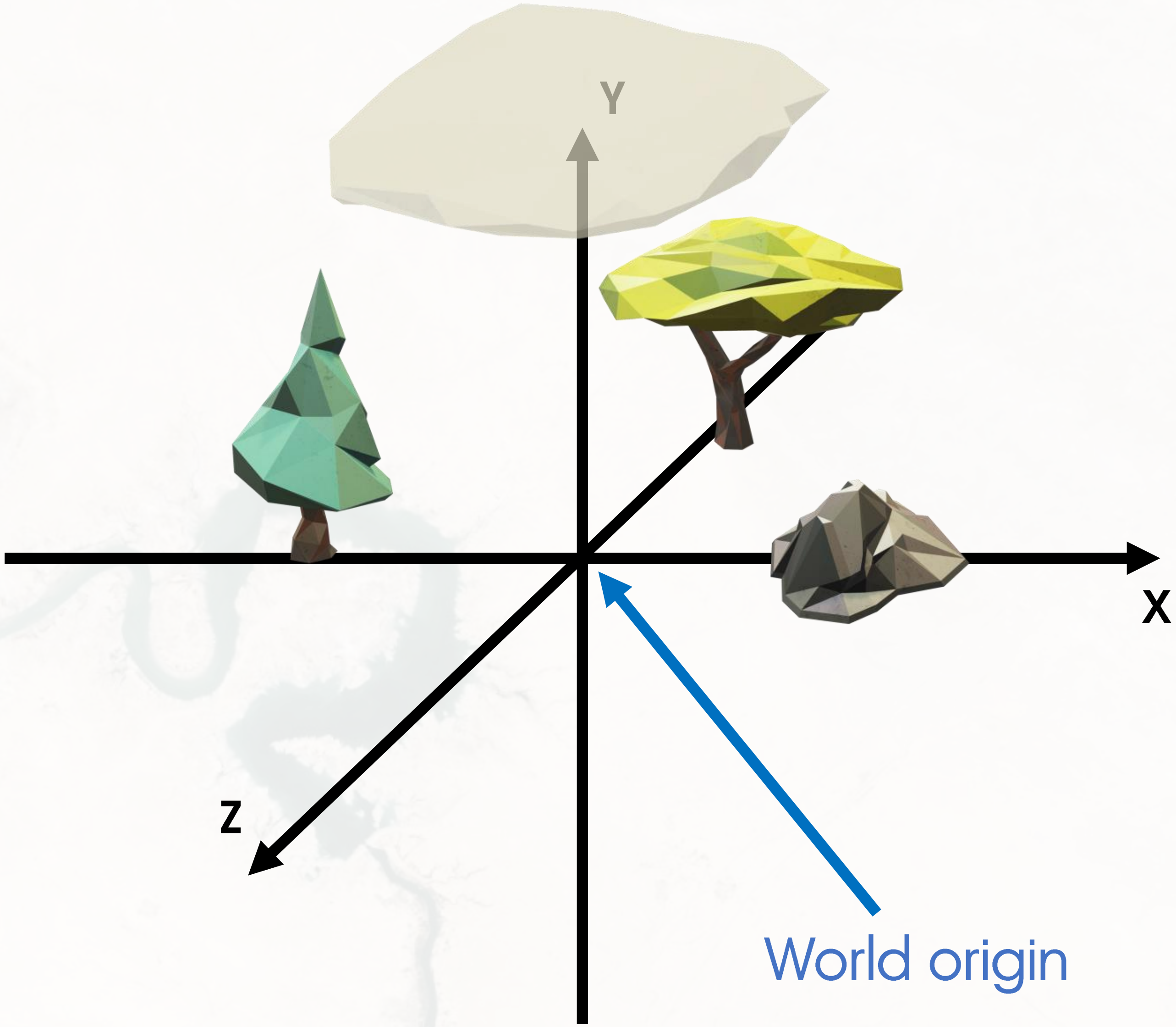
$$10 * 2 + 200 = 220$$

$$(10 + 200) * 2 = 420$$

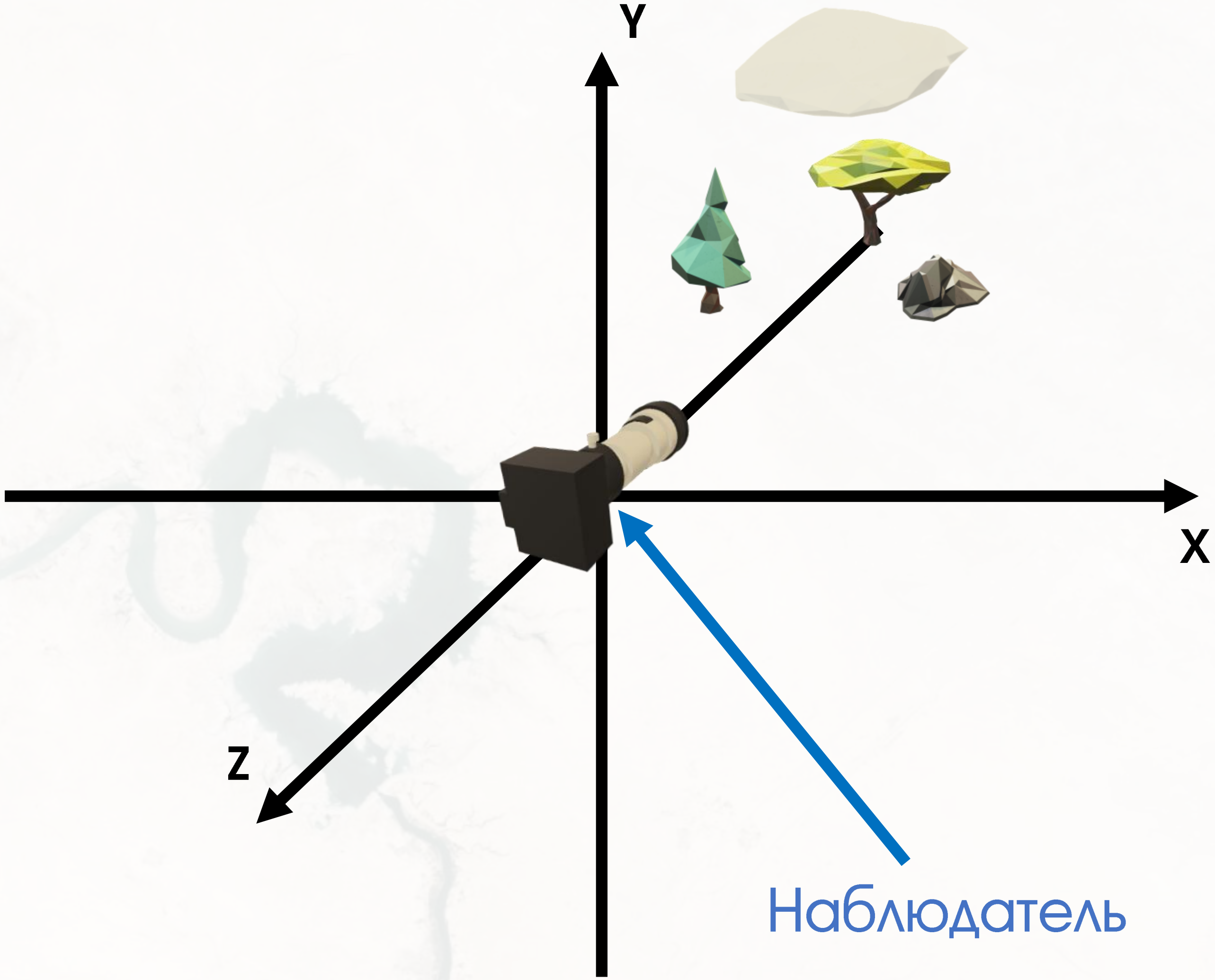
сначала смещение, потом - масштабирование

Для всех преобразований используются матрицы

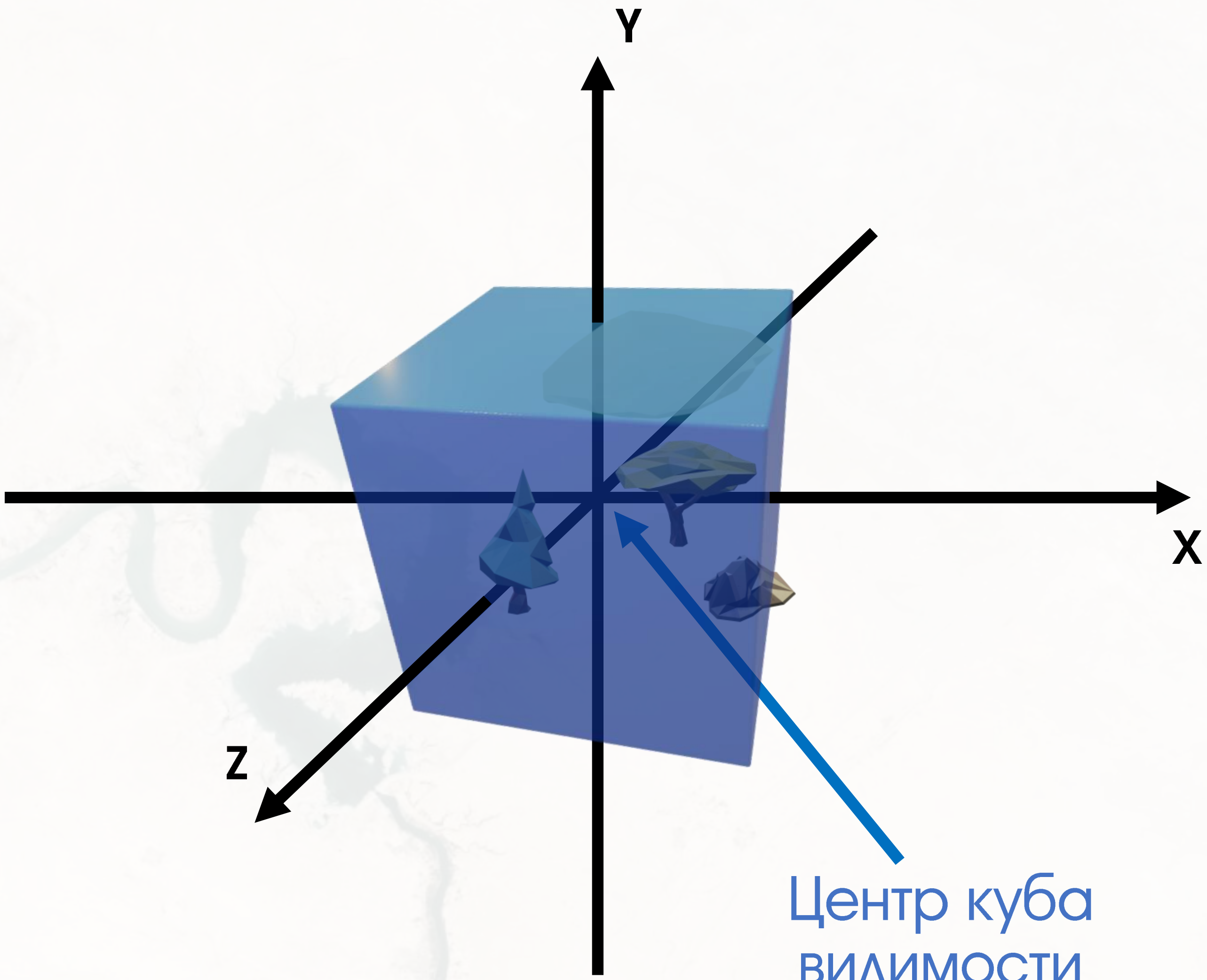
мировые координаты



координаты **относительно камеры**



нормализованные координаты



Точка отсчета
экранных координат

экранные координаты

Object coordinates
Координаты объекта относительно центра модели

↓ Трансформация объектов по-отдельности

World coordinates
Координаты относительно единого центра (world origin)

↓ Трансформация объектов вместе

Eye Coordinates
Координаты относительно камеры (наблюдателя)

↓ Частичная нормализация

Clip Coordinates
Частично нормализованные координаты

↓ Завершение нормализации (деление на w)

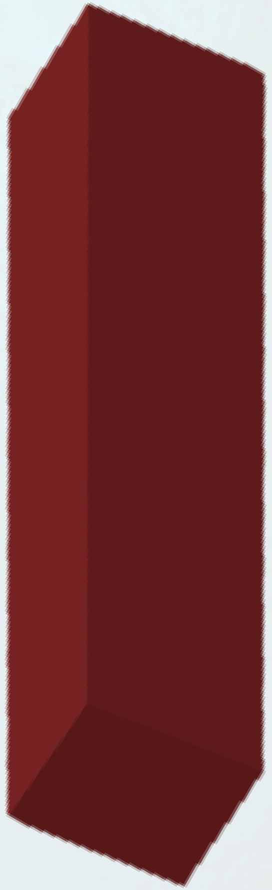
Viewport (Normalized) Coordinates
Координаты готовые для отображения

↓ Отображение точек

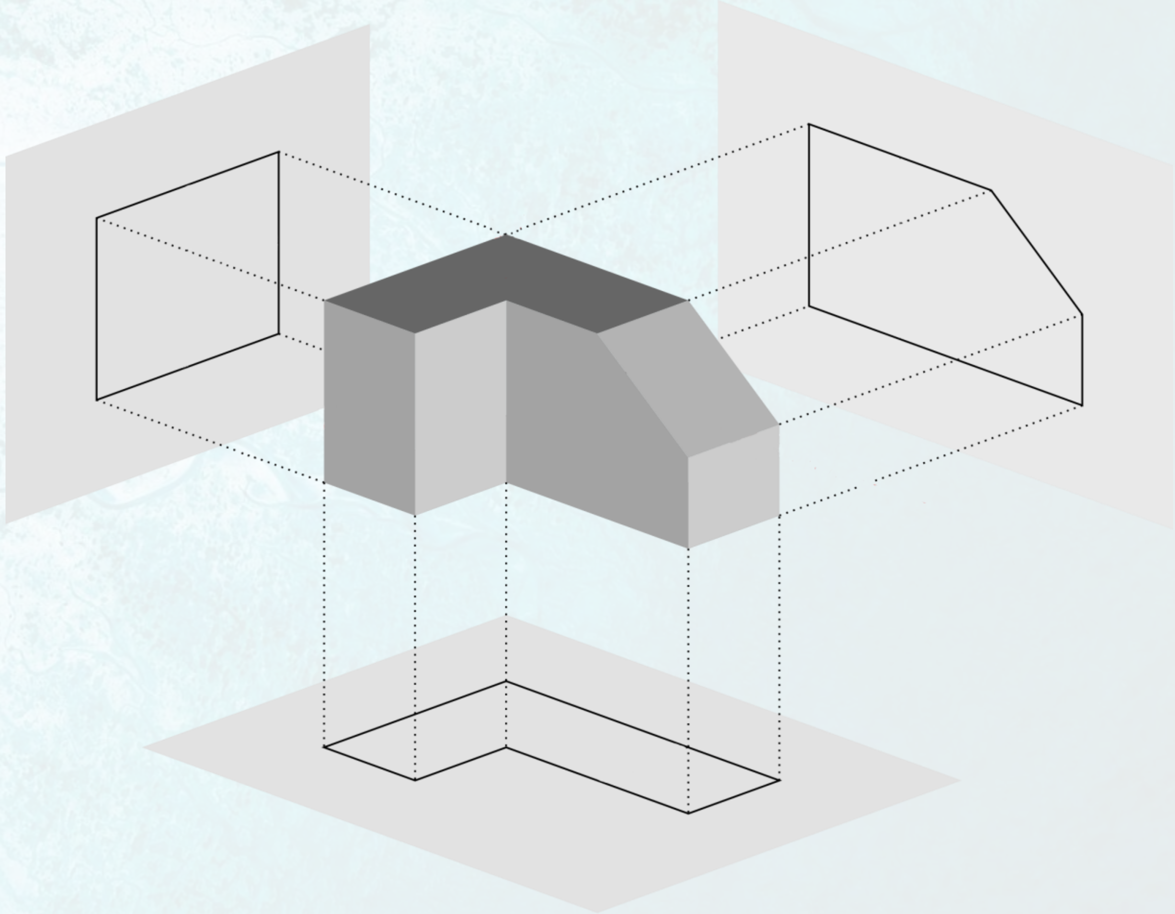
Screen Coordinates
Двумерные координаты отображенной точки

Перспектива

Перспектива
и
ортография



применение ортографии



$$\begin{bmatrix} \frac{2}{right-left} & 0 & 0 & 0 \\ 0 & \frac{2}{top-bottom} & 0 & 0 \\ 0 & 0 & \frac{2}{near-far} & 0 \\ \frac{left+right}{left-right} & \frac{bottom+top}{bottom-top} & \frac{near+far}{near-far} & 1 \end{bmatrix}$$

$$x \in [left; right] \to [-1; 1]$$

$$y \in [bottom; top] \to [-1; 1]$$

$$-z \in [-near; -far] \to [-1; 1]$$

матрица нормализации координат при ортографии

Обоснование матрицы нормализации координат при ортографии

$$x \in [left; right] \to [-1; 1]$$

По двум известным точкам и линейному характеру отображения составим систему уравнений

$$x \in [left; right] \to [-1; 1]$$

$$\begin{cases} k * left + c = -1 \\ k * right + c = 1 \end{cases}$$

Составили систему уравнений, теперь выразим c через k

$$x \in [left; right] \rightarrow [-1; 1]$$

$$\begin{cases} k * left + c = -1 \\ k * right + c = 1 \end{cases}$$

$$c = -1 - k * left$$

Выразили c через k, теперь выполним подстановку

$$x \in [left; right] \to [-1; 1]$$

$$\begin{cases} k * left + c = -1 \\ k * right + c = 1 \end{cases}$$

$$c = -1 - k * left$$

$$k * right + c = k * right - 1 - k * left = 1$$

Нашли, чему равно k

$$x \in [left; right] \to [-1; 1]$$

$$\boldsymbol{k = \frac{2}{right - left}}$$

$$\begin{cases} k * left + c = -1 \\ k * right + c = 1 \end{cases}$$

Нашли, чему равно c

$$c = -1 - k * left$$

$$k * right + c = k * right - 1 - k * left = 1$$

$$c = -1 - k * left = -1 - \frac{2 * left}{right - left} = \frac{-right - left}{right - left} = \frac{right + left}{left - right}$$

$$x \in [left; right] \to [-1; 1]$$

$$\mathbf{k = \frac{2}{right - left}}$$

$$\begin{cases} k * left + c = -1 \\ k * right + c = 1 \end{cases}$$

$$\mathbf{c = \frac{right + left}{left - right}}$$

$$c = -1 - k * left$$

$$k * right + c = k * right - 1 - k * left = 1$$

$$c = -1 - k * left = -1 - \frac{2 * left}{right - left} = \frac{-right - left}{right - left} = \frac{right + left}{left - right}$$

$$k_x = \frac{2}{right - left}$$

$$c_x = \frac{right + left}{left - right}$$

$$x \epsilon [left; right] \rightarrow [-1; 1]$$

$$y \epsilon [bottom; top] \rightarrow [-1; 1]$$

То же самое – для y

$$k_x = \frac{2}{right - left}$$

$$c_x = \frac{left + right}{left - right}$$

$$k_y = \frac{2}{top - bottom}$$

$$c_y = \frac{bottom + top}{bottom - top}$$

$$x \in [left; right] \to [-1; 1]$$

$$y \in [bottom; top] \to [-1; 1]$$

Почти то же самое – для z

$$\boldsymbol{k_x = \frac{2}{right - left}}$$

$$x \in [left; right] \to [-1; 1]$$

$$y \in [bottom; top] \to [-1; 1]$$

$$-z \in [-near; -far] \to [-1; 1]$$

$$\boldsymbol{c_x = \frac{left + right}{left - right}}$$

$$\boldsymbol{k_z} = \frac{2}{(-\boldsymbol{far}) - (-\boldsymbol{near})} = \boldsymbol{\frac{2}{near - far}}$$

$$\boldsymbol{c_z} = \frac{(-\boldsymbol{far}) + (-\boldsymbol{near})}{(-\boldsymbol{near}) - (-\boldsymbol{far})} = \boldsymbol{\frac{near + far}{near - far}}$$

$$\begin{bmatrix} \frac{2}{right-left} & 0 & 0 & 0 \\ 0 & \frac{2}{top-bottom} & 0 & 0 \\ 0 & 0 & \frac{2}{near-far} & 0 \\ \frac{left+right}{left-right} & \frac{bottom+top}{bottom-top} & \frac{near+far}{near-far} & 1 \end{bmatrix}$$

$$\boldsymbol{k_x = \frac{2}{right-left}}$$

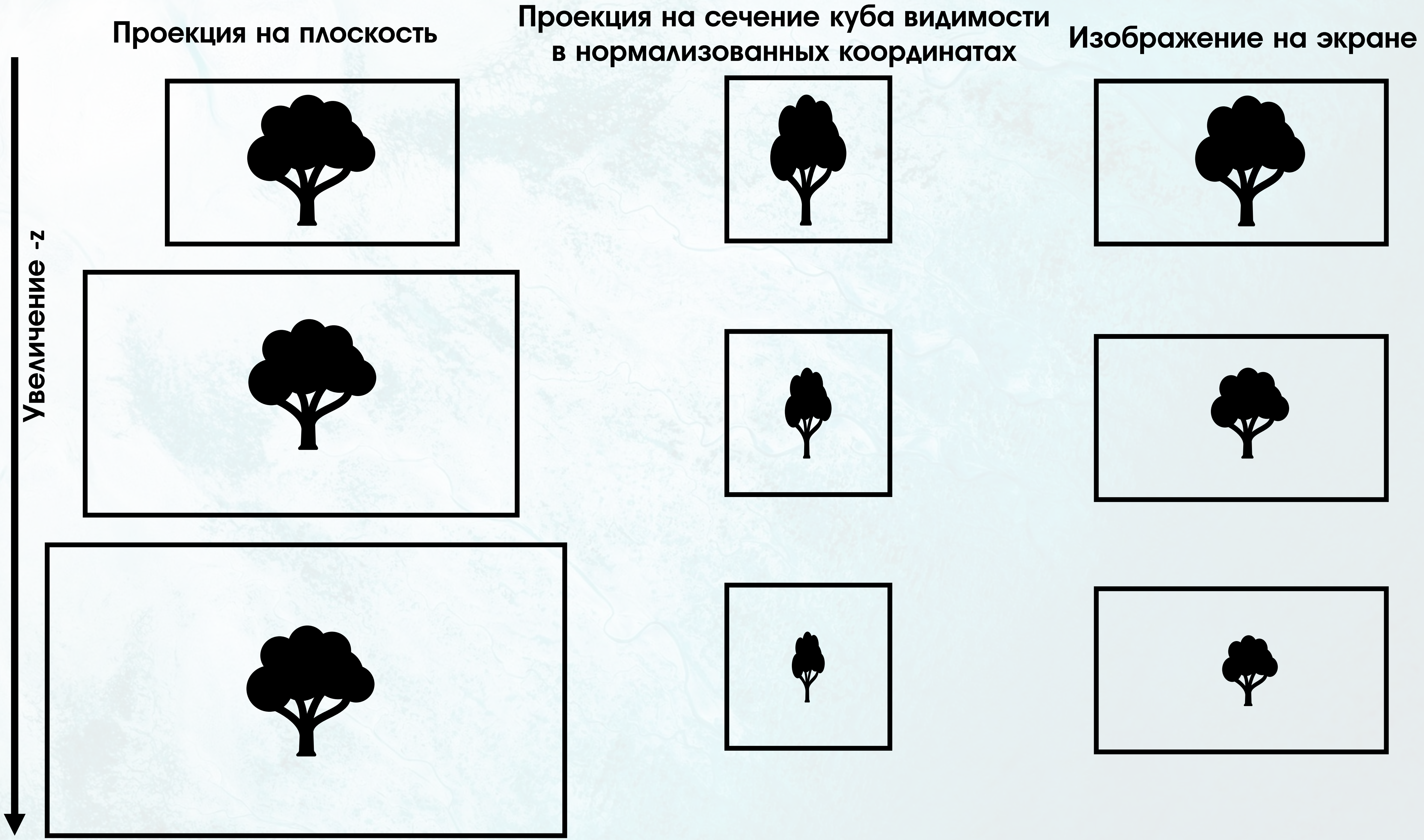
$$\boldsymbol{c_x = \frac{left+right}{left-right}}$$

$$\boldsymbol{k_y = \frac{2}{top-bottom}}$$

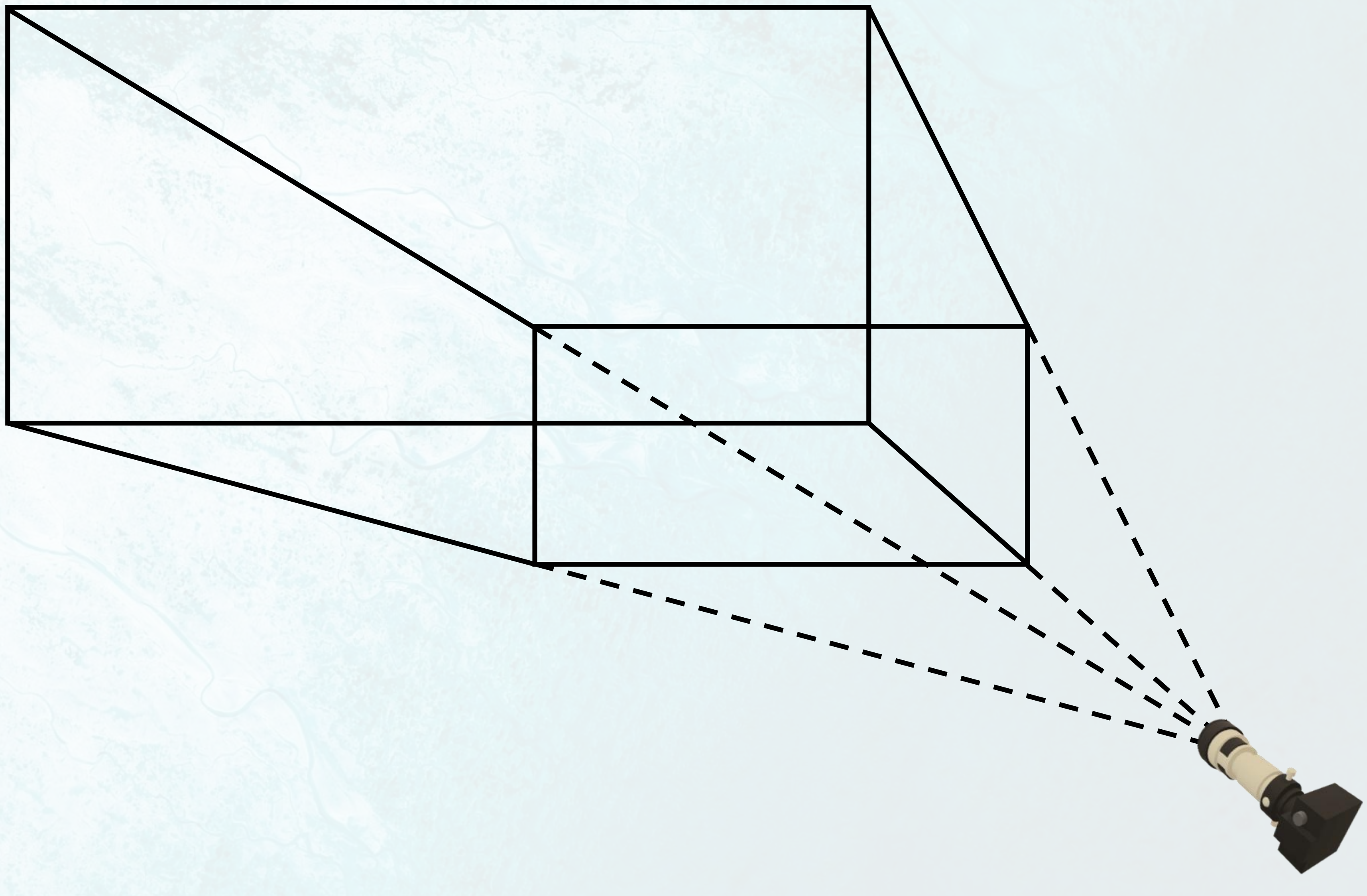
$$\boldsymbol{c_y = \frac{bottom+top}{bottom-top}}$$

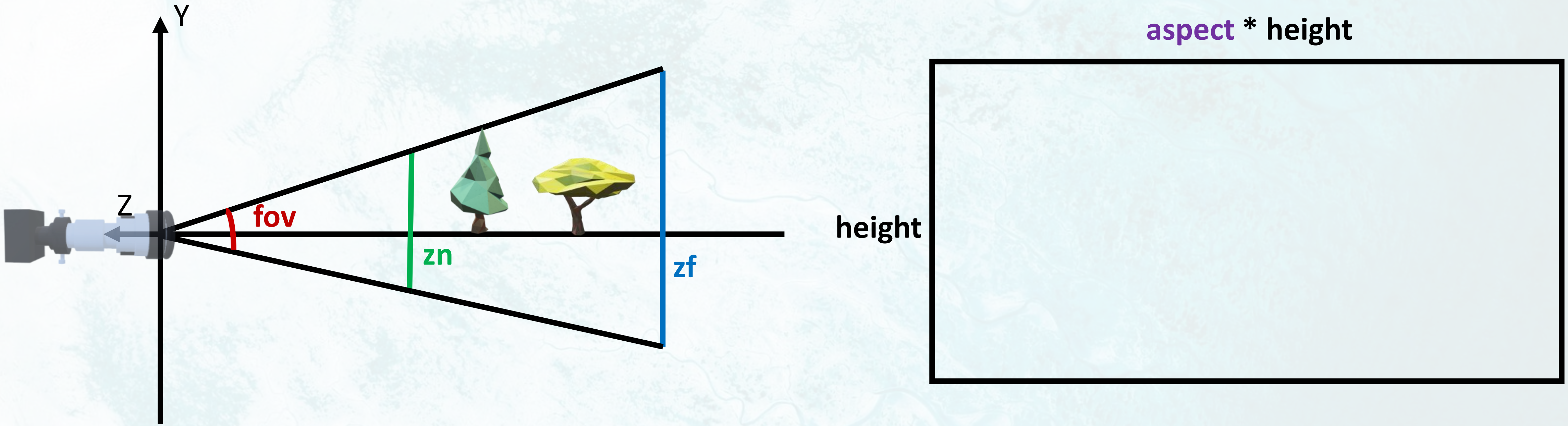
$$\boldsymbol{k_z = \frac{2}{near-far}}$$

$$\boldsymbol{c_z = \frac{near+far}{near-far}}$$

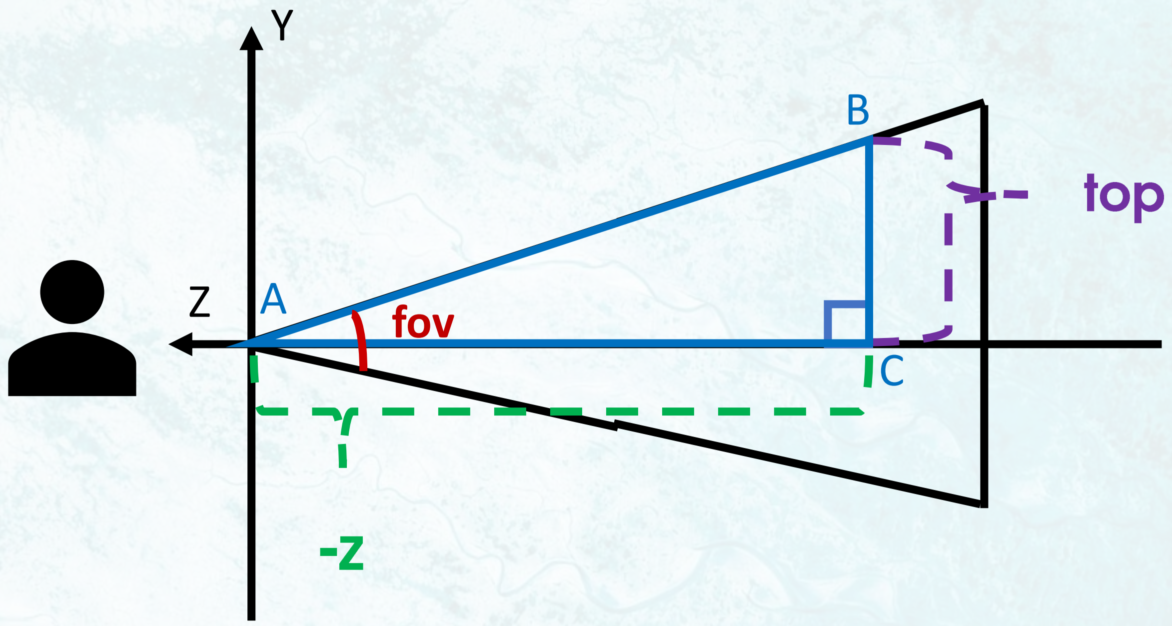

Проекция на плоскость
Проекция на сечение куба видимости в нормализованных координатах
Изображение на экране
Увеличение -z

фрустум





fov (field_of_view) – телесный угол в вершине фрустума, ближайшей к точке наблюдения
zn (z_near) – минимальная координата по оси z, которую должна иметь точка, чтобы попасть в поле зрения
zf (z_far) – максимальная координата по оси z, которую может иметь точка, чтобы попасть в поле зрения
aspect – отношение ширины сечения фрустума к высоте



fov (field of view) - телесный угол в вершине фруструма, ближайшей к точке наблюдения

top – наибольшая координата по оси y, которую может иметь точка, чтобы попасть внутрь фруструма

$$\frac{top}{-z} = tg\left(\frac{fov}{2}\right)$$

$$\frac{top}{-z} = tg\left(\frac{fov}{2}\right)$$

$$top = -z \ * \ tg\left(\frac{fov}{2}\right)$$

определим зависимость top от –z и fov

$$\frac{top}{-z} = tg\left(\frac{fov}{2}\right)$$

$$top = -z * tg\left(\frac{fov}{2}\right)$$

$$y_{normalized} = \frac{y}{top}$$

определим зависимость нормализованной координаты y от top

$$\frac{top}{-z} = tg\left(\frac{fov}{2}\right)$$

$$top = -z \ * \ tg\left(\frac{fov}{2}\right)$$

$$y_{normalized} = \frac{y}{top}$$

$$y_{normalized} = \frac{y}{-z * tg\left(\frac{fov}{2}\right)}$$

определим зависимость нормализованной координаты y от –z и fov

введем обозначение

$$y_{normalized} = \frac{y * f}{-z} \text{ где } f = \frac{1}{tg\left(\frac{fov}{2}\right)} = tg\left(\frac{\pi}{2} + \frac{fov}{2}\right)$$

$$x_{normalized} = \frac{x}{right}$$

определим зависимость нормализованной координаты x от right*

*right по аналогии с top - наибольшая координата по оси x, которую может иметь точка, чтобы попасть внутрь фрустума

$$x_{normalized} = \frac{x}{right}$$

$$x_{normalized} = \frac{x}{top * aspect}$$

определим зависимость нормализованной координаты x от top и aspect

*right по аналогии с top - наибольшая координата по оси x, которую может иметь точка, чтобы попасть внутрь фрустума

$$top = \frac{-z}{f}$$

$$x_{normalized} = \frac{x}{right}$$

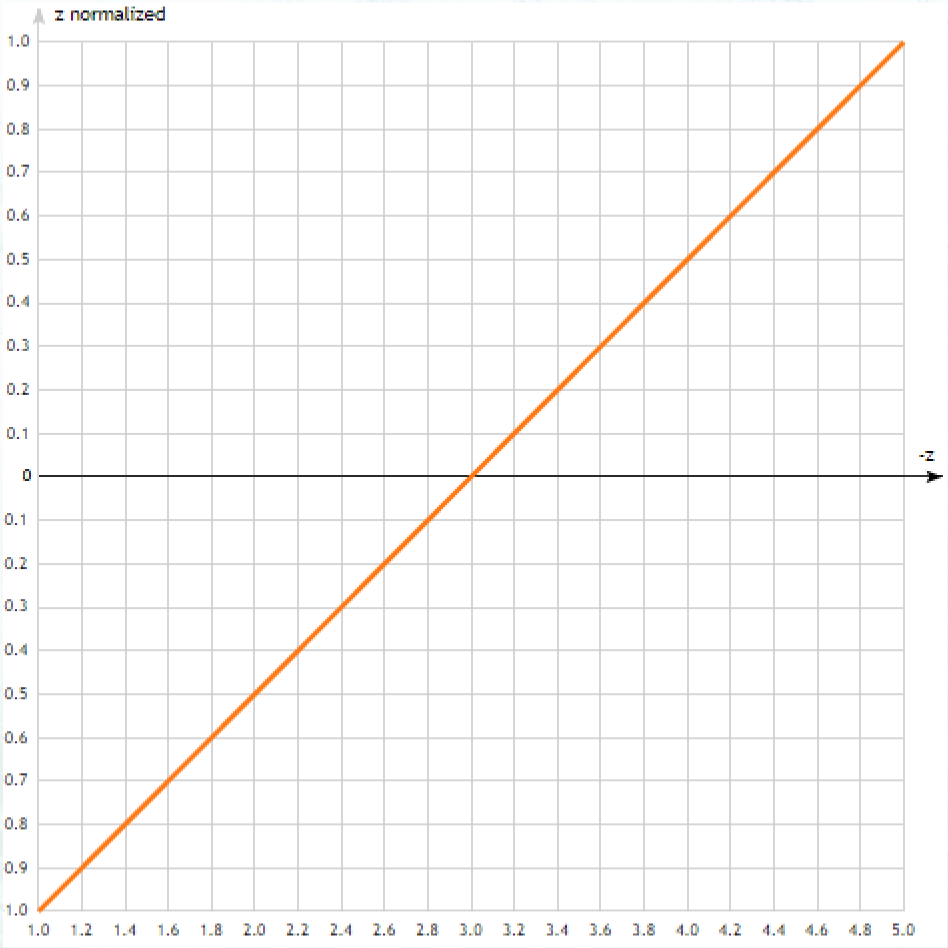
$$x_{normalized} = \frac{x}{top * aspect}$$

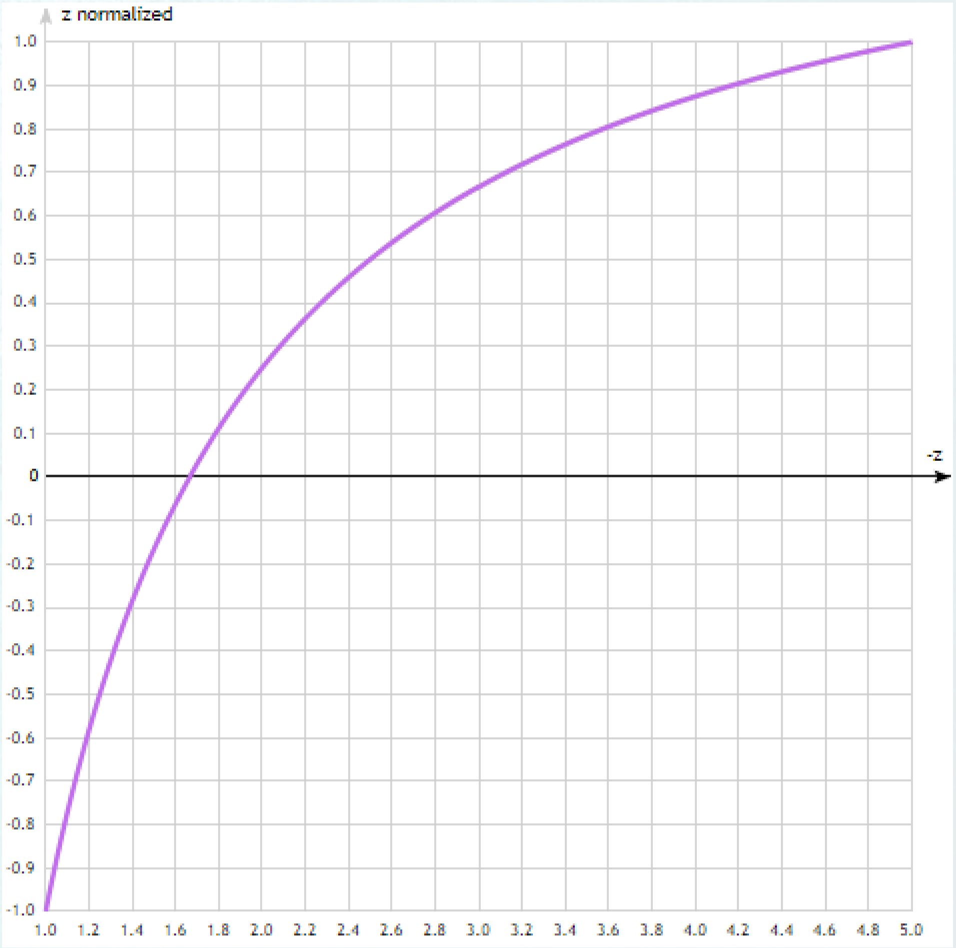
$$x_{normalized} = \frac{x * f}{-z * aspect}$$

определим зависимость нормализованной координаты x от aspect, f и -z

*right по аналогии с top - наибольшая координата по оси x, которую может иметь точка, чтобы попасть внутрь фрустума

линейное отображение
и
отображение по закону обратной пропорциональности





$$f(x) = \frac{m}{x} + c$$

Функция обратной пропорциональности

$$-z \in [-near; -far] \rightarrow [-1; 1]$$

$$f(x) = \frac{m}{x} + c$$

$$\begin{cases} \dfrac{m}{-near} + c = -1 \\ \dfrac{m}{-far} + c = 1 \end{cases}$$

По двум известным точкам и характеру отображения в виде функции обратной пропорциональности составим систему уравнений

$$\begin{cases} \dfrac{m}{-near} + c = -1 \\ \dfrac{m}{-far} + c = 1 \end{cases}$$

$$c = -1 - \frac{m}{-near}$$

Выразили c через m, теперь выполним подстановку

$$\begin{cases} \dfrac{m}{-near} + c = -1 \\ \dfrac{m}{-far} + c = 1 \end{cases}$$

Нашли, чему равно m

$$c = -1 - \frac{m}{-near}$$

$$\frac{m}{-far} + c = \frac{m}{-far} - 1 - \frac{m}{-near} = 1$$

$$\frac{-near + far}{near * far} = \frac{2}{m}$$

$$\begin{cases} \dfrac{m}{-near} + c = -1 \\ \dfrac{m}{-far} + c = 1 \end{cases}$$

$$\boldsymbol{m = -\frac{2 * near * far}{near - far}}$$

Нашли, чему равно c

$$c = -1 - \frac{m}{-near}$$

$$\frac{m}{-far} + c = \frac{m}{-far} - 1 - \frac{m}{-near} = 1 \qquad \frac{-near + far}{near * far} = \frac{2}{m}$$

$$\boldsymbol{c} = -1 - \frac{m}{-near} = -1 - \frac{2 * far}{near - far} = \boldsymbol{\frac{-near - far}{near - far}}$$

$$\begin{cases} \frac{m}{-near} + c = -1 \\ \frac{m}{-far} + c = 1 \end{cases}$$

$$\boldsymbol{m = -\frac{2 * near * far}{near - far}}$$

$$\boldsymbol{c = \frac{-near - far}{near - far}}$$

$$c = -1 - \frac{m}{-near}$$

$$\frac{m}{-far} + c = \frac{m}{-far} - 1 - \frac{m}{-near} = 1 \qquad \frac{-near + far}{near * far} = \frac{2}{m}$$

$$\boldsymbol{c} = -1 - \frac{m}{-near} = -1 - \frac{2 * far}{near - far} = \boldsymbol{\frac{-near - far}{near - far}}$$

$$m = -\frac{2 * near * far}{near - far}$$

$$c = \frac{-near - far}{near - far}$$

Таким образом

$$z_{normalized} = \frac{m}{-z} + c = \frac{2 * near * far}{-z * (near - far)} - \frac{near + far}{near - far}$$

Clip coordinates

$$x_{clip} = \frac{x * f}{aspect}$$

$$y_{clip} = y * f$$

$$z_{clip} = z * \frac{near + far}{near - far} + \frac{2 * near * far}{near - far}$$

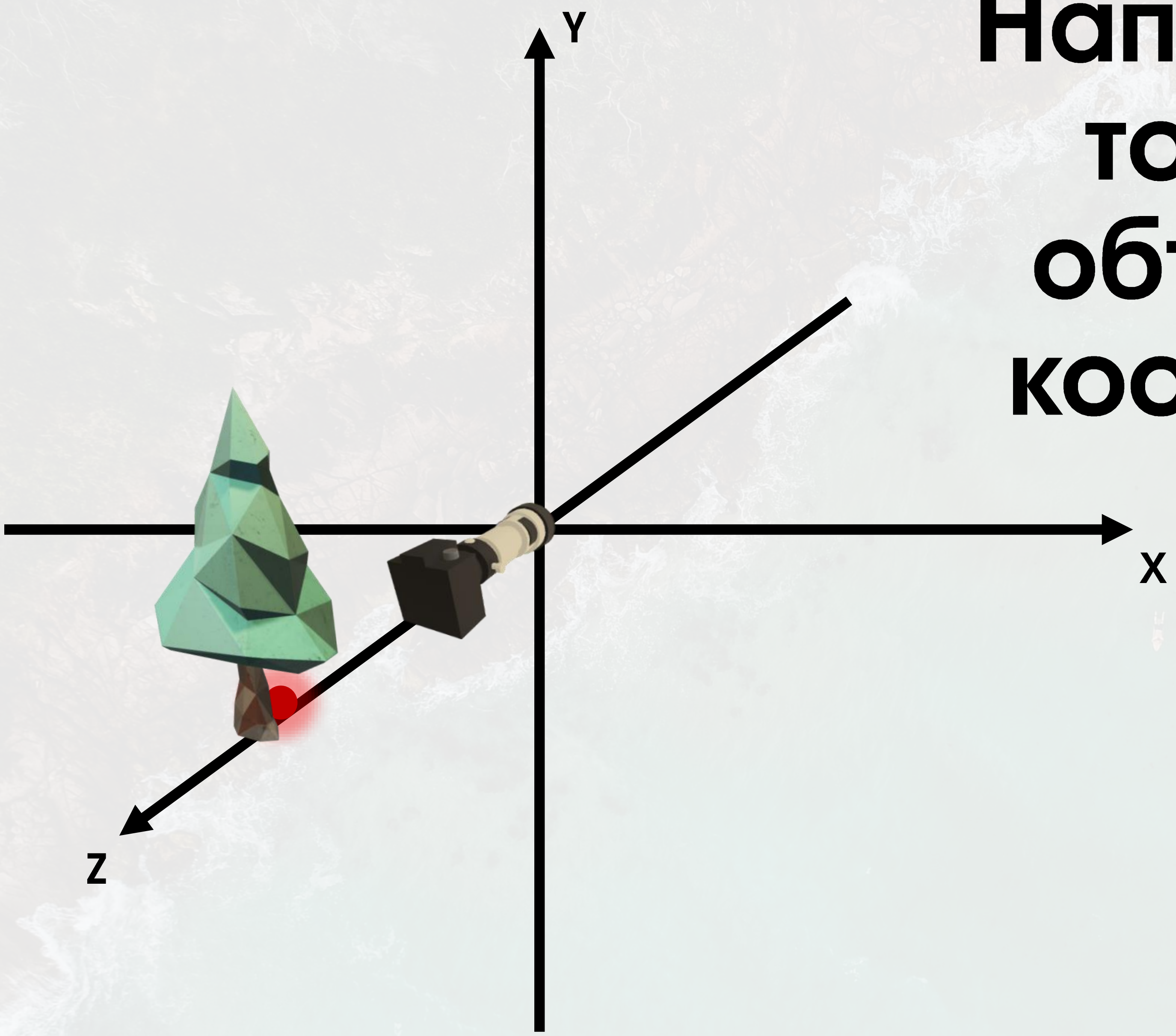
$$w_{clip} = -z$$

Наконец, составим матрицу нормализации для создания эффекта перспективы

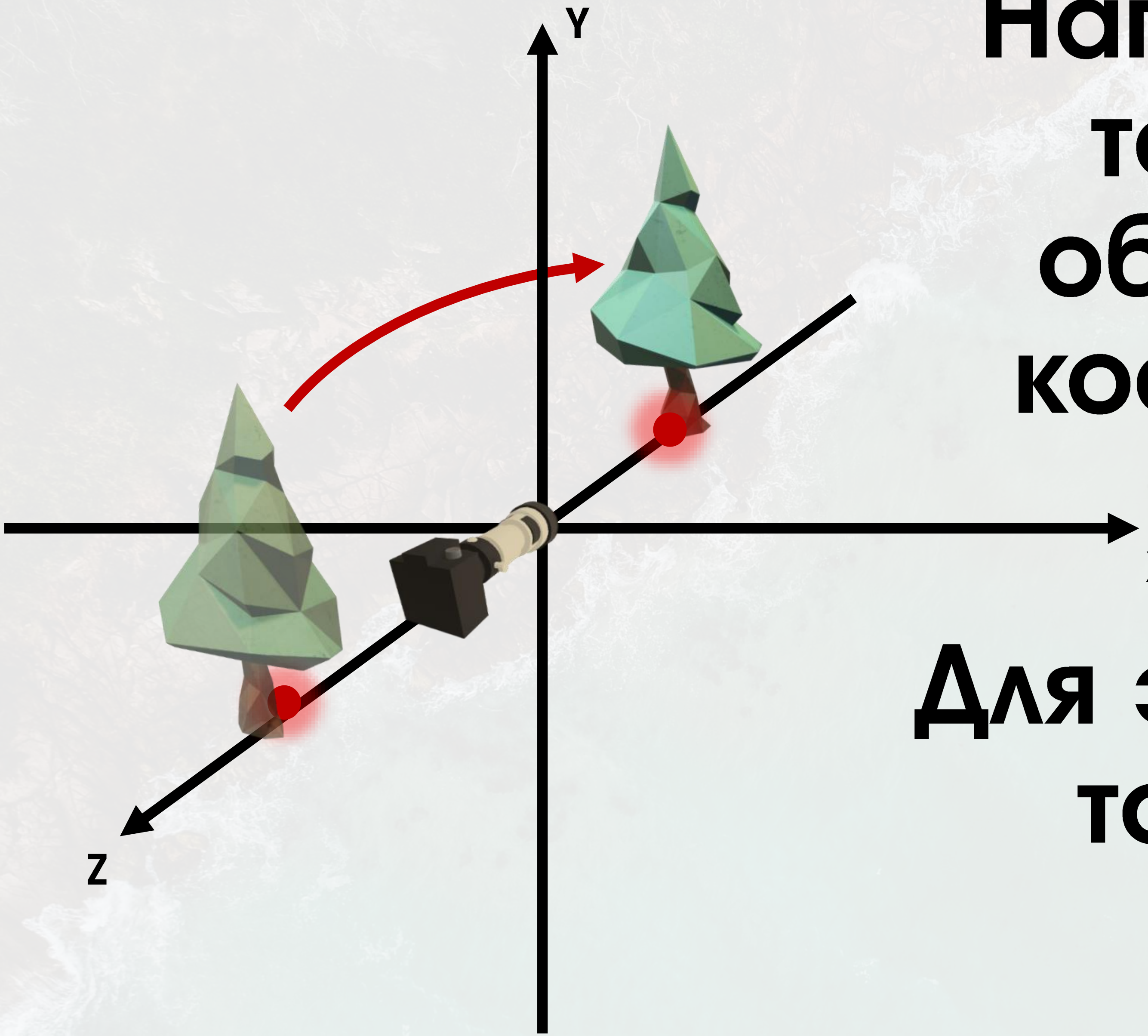
$$\begin{bmatrix} \dfrac{f}{aspect} & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & \dfrac{near + far}{near - far} & -1 \\ 0 & 0 & \dfrac{2 * near * far}{near - far} & 0 \end{bmatrix}$$

где $f = \dfrac{1}{tg\left(\dfrac{fov}{2}\right)} = tg\left(\dfrac{\pi}{2} + \dfrac{fov}{2}\right)$

Слежение за объектом



Направим камеру на точку некоторого объекта, имеющую координаты (0; 0; 4)

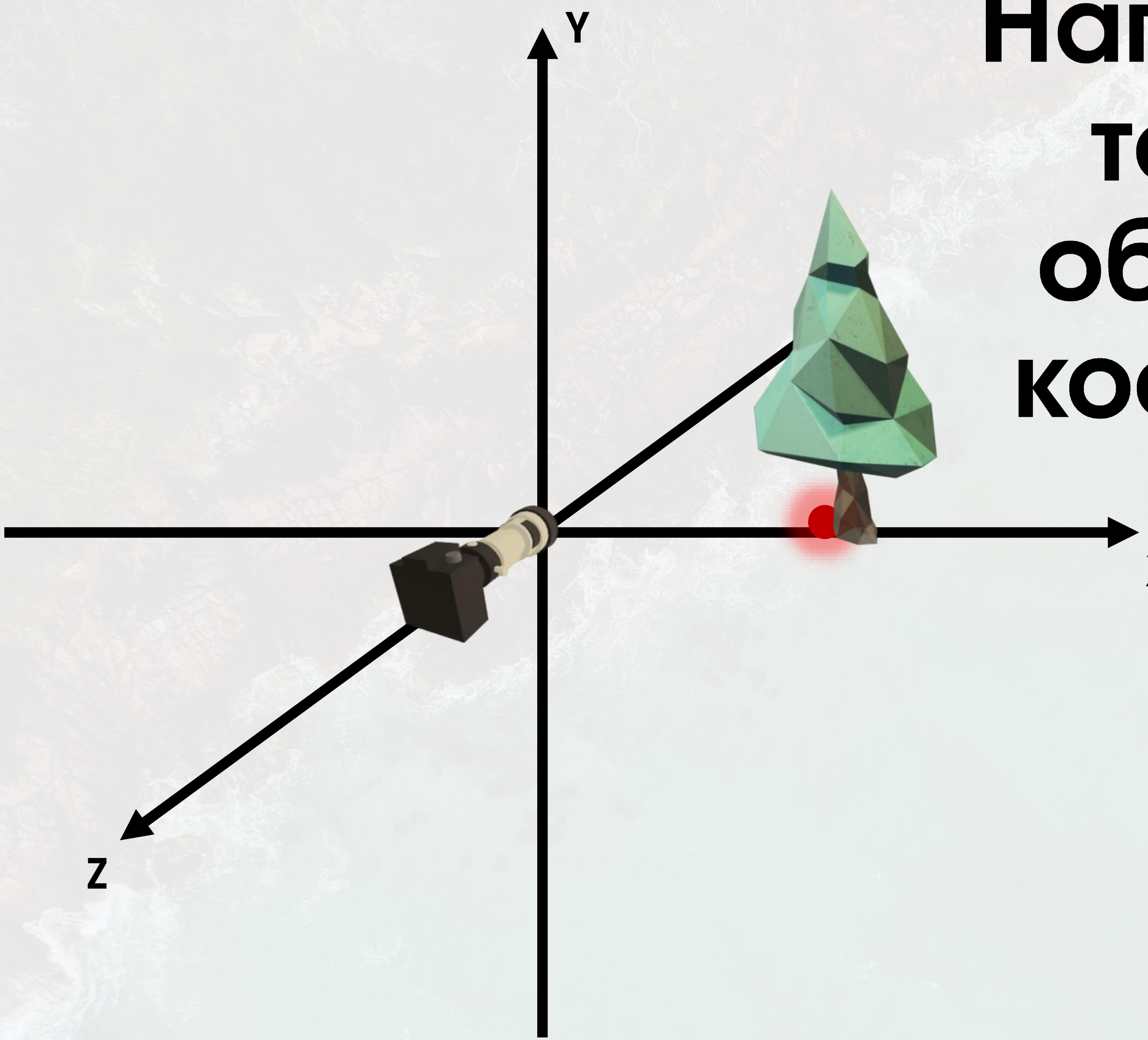


Направим камеру на точку некоторого объекта, имеющую координаты (0; 0; 4)

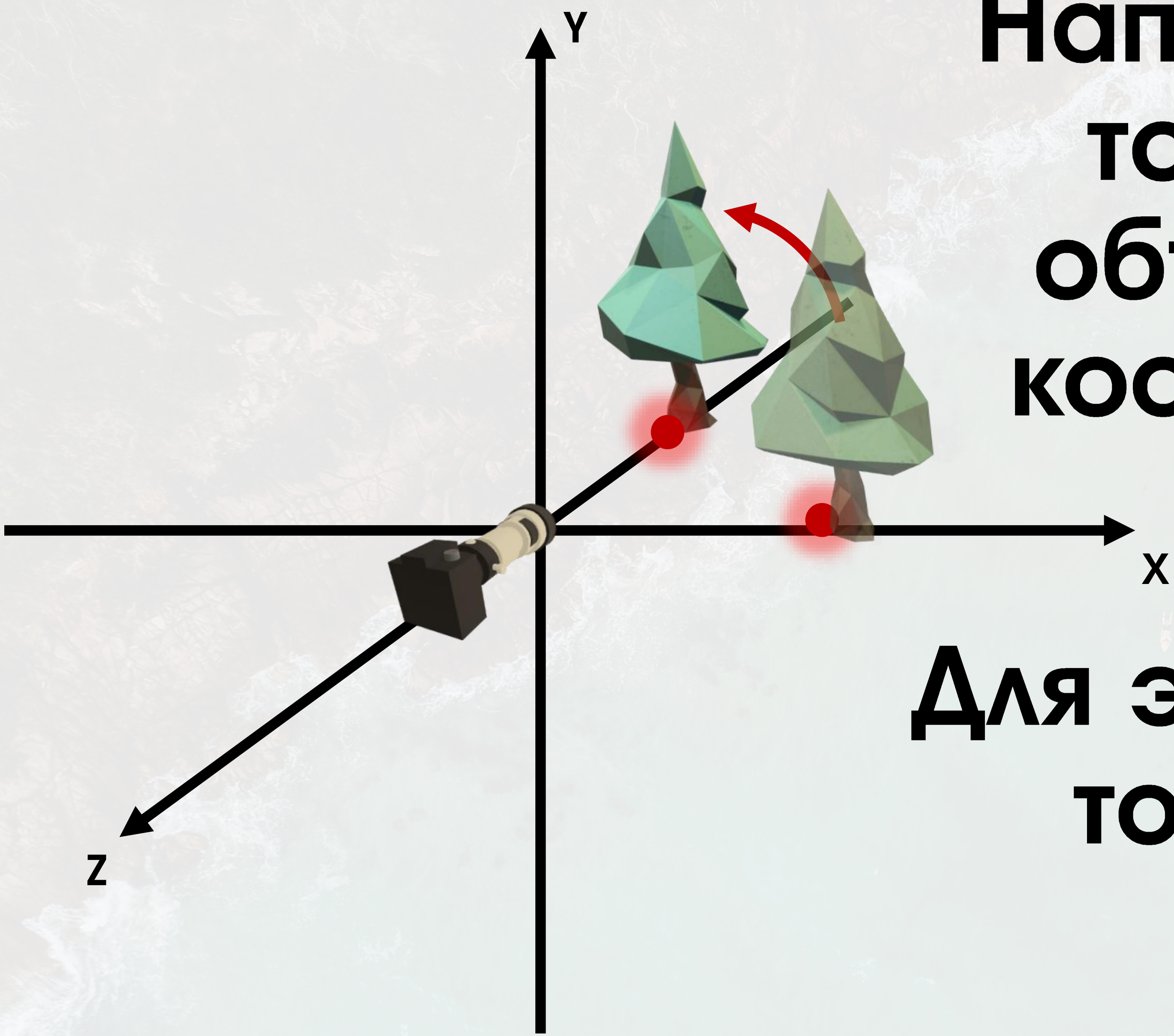
Для этого переместим точку в позицию (0; 0; -4)

Используем матричное умножение

$$\begin{bmatrix} 0 & 0 & 4 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & -4 & 1 \end{bmatrix}$$



Направим камеру на точку некоторого объекта, имеющую координаты (4; 0; 0)

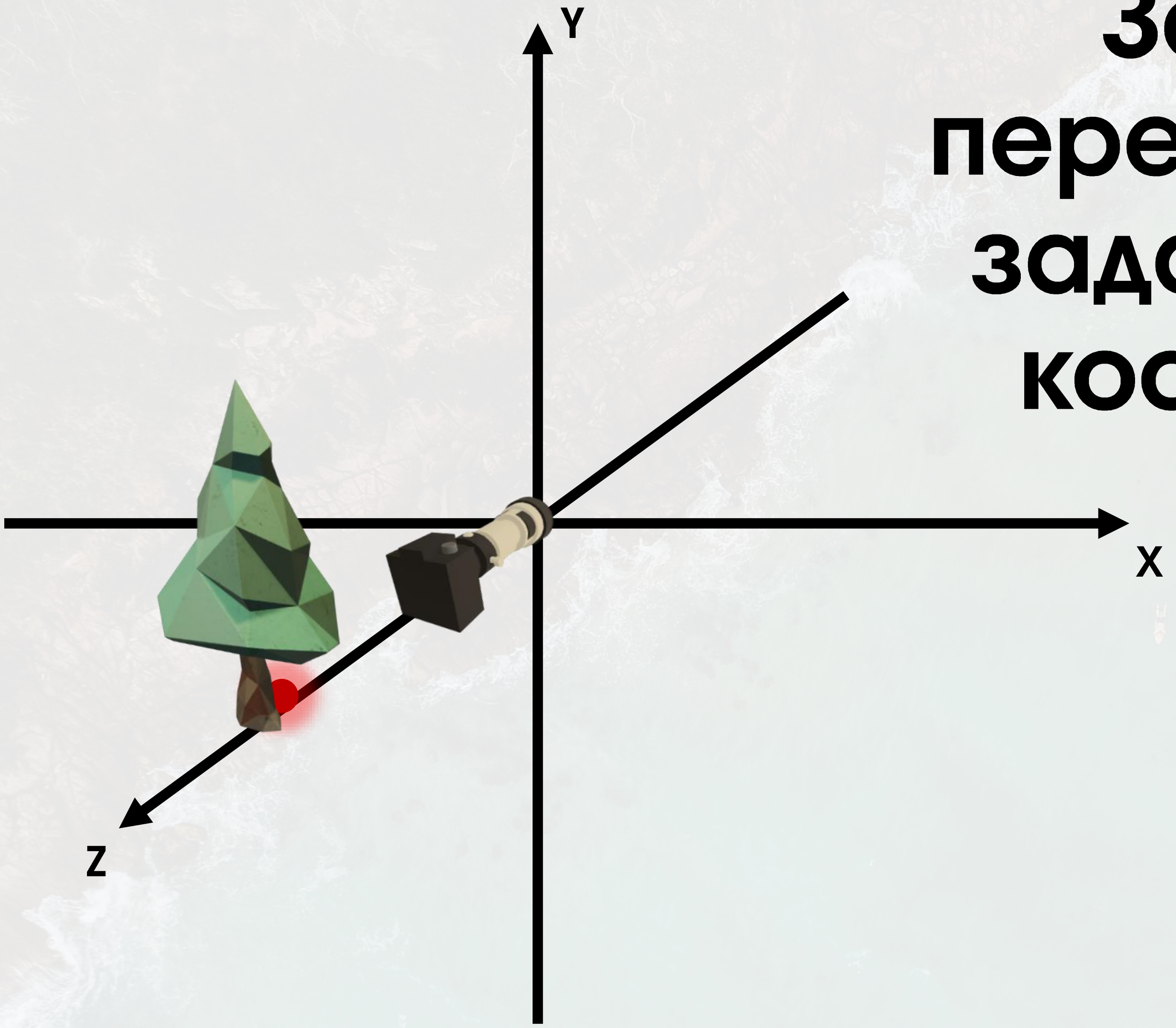


Направим камеру на точку некоторого объекта, имеющую координаты (4; 0; 0)

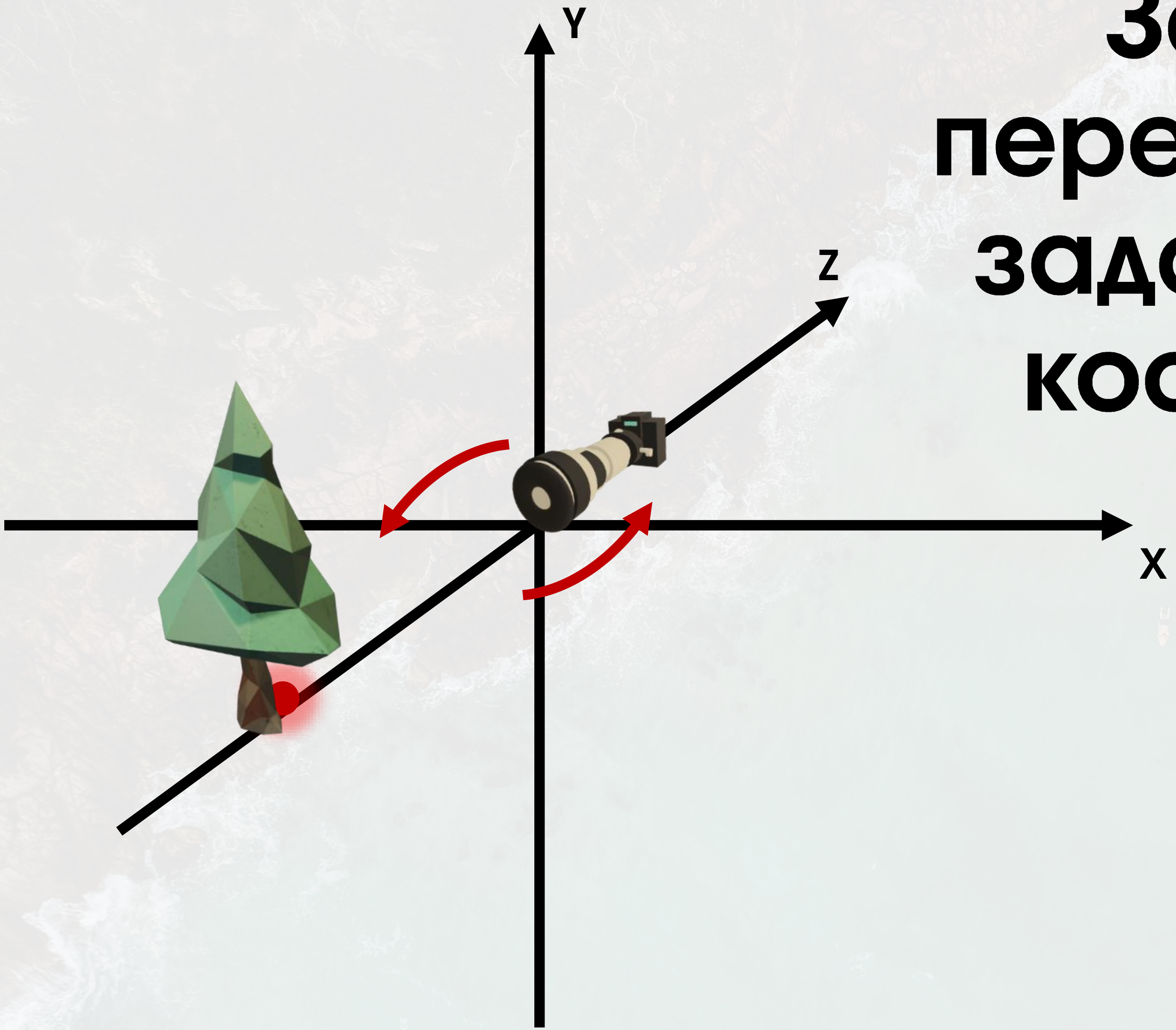
Для этого переместим точку в позицию (0; 0; -4)

Используем матричное умножение

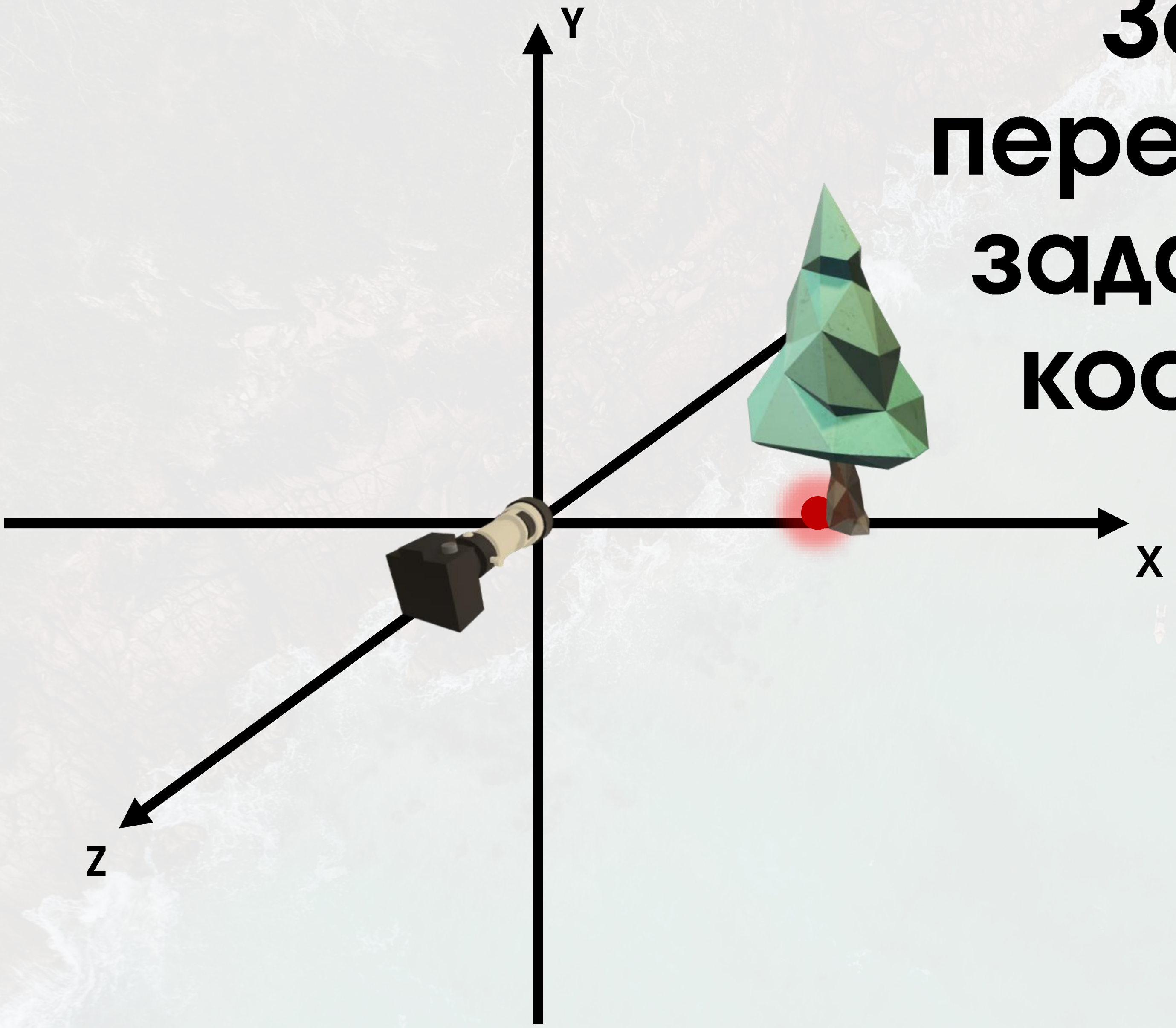
$$\begin{bmatrix} 4 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & -4 & 1 \end{bmatrix}$$



Заменим задачу перемещения точки на задачу перемещения координатных осей в первом случае



Заменим задачу перемещения точки на задачу перемещения координатных осей в первом случае



Заменим задачу перемещения точки на задачу перемещения координатных осей во втором случае

Заменим задачу перемещения точки на задачу перемещения координатных осей во втором случае

Направим камеру на точку некоторого объекта, имеющую координаты **(-3; 0; -4)**

Направим камеру на точку некоторого объекта, имеющую координаты (-3; 0; -4)

Для этого переместим точку в позицию (0; 0; -5)

Найдем необходимую матрицу

Из координат x (-3) и z (-4) должна получиться новая координата z = -5, то есть

$$-3a-4b=-5$$

Найдем необходимую матрицу

Из координат x (-3) и z (-4) должна получиться новая координата z = -5, то есть

$$-3a - 4b = -5$$

$$b = \frac{-3a + 5}{4}$$

Найдем необходимую матрицу

Из координат x (-3) и z (-4) должна получиться новая координата x = 0, то есть

$$-3c - 4d = 0$$

$$d = -\frac{3}{4}c$$

Необходимая матрица имеет вид

$$\begin{bmatrix} c & 0 & a & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{3}{4}c & 0 & \frac{5-3a}{4} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Введем ограничение – первые 3 строки представляют собой единичные вектора, тогда

$$\begin{cases} \sqrt{a^2 + c^2} = 1 \\ \sqrt{\left(\frac{3}{4}c\right)^2 + \left(\frac{5 - 3a}{4}\right)^2} = 1 \end{cases}$$

Из

$$\sqrt{a^2 + c^2} = 1$$

следует, что

$$c^2 = 1 - a^2$$

Тогда, если подставить это во второе уравнение системы, то

$$9 - 9a^2 + 25 - 30a + 9a^2 = 16$$

$$34 - 30a = 16$$

$$\boldsymbol{a = \frac{18}{30} = 0,6} \qquad \boldsymbol{c = \sqrt{1 - 0,36} = 0,8}$$

Значит искомая матрица имеет вид

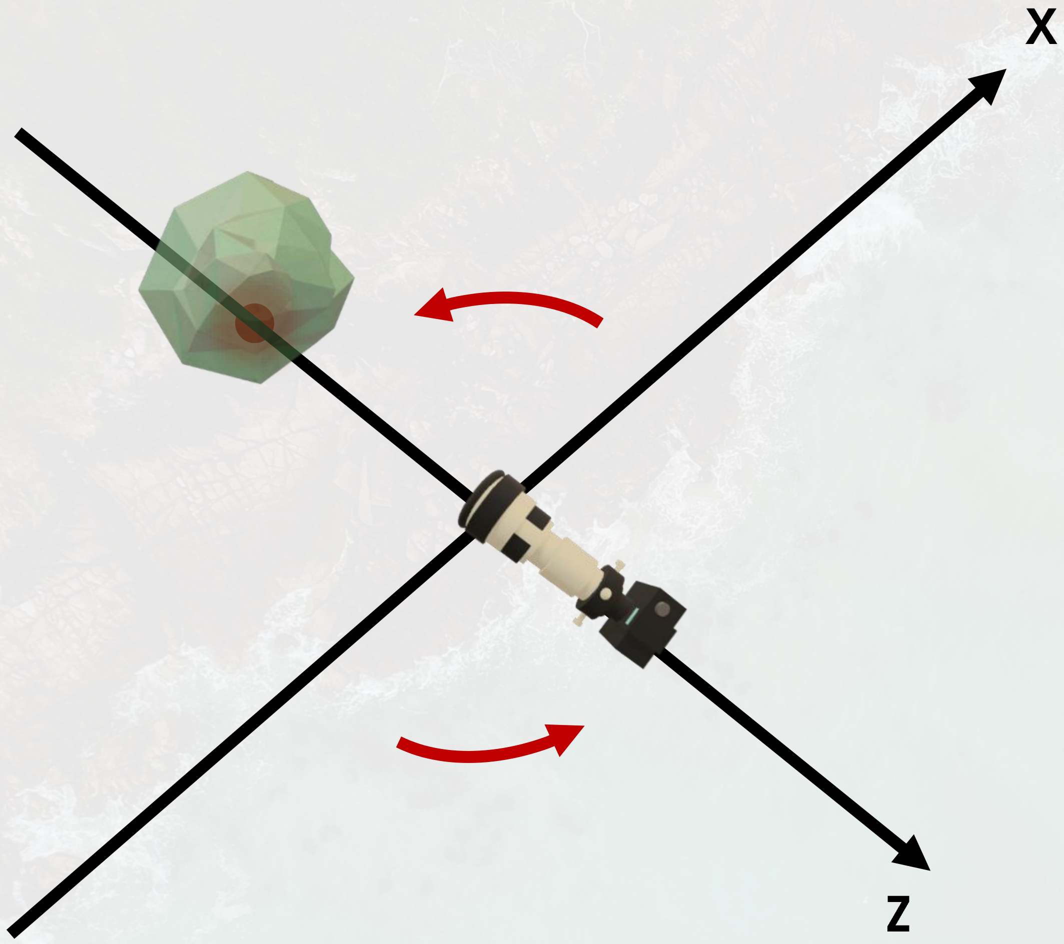
$$\begin{bmatrix} 0{,}8 & 0 & 0{,}6 & 0 \\ 0 & 1 & 0 & 0 \\ -0{,}6 & 0 & 0{,}8 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Проверим полученный результат

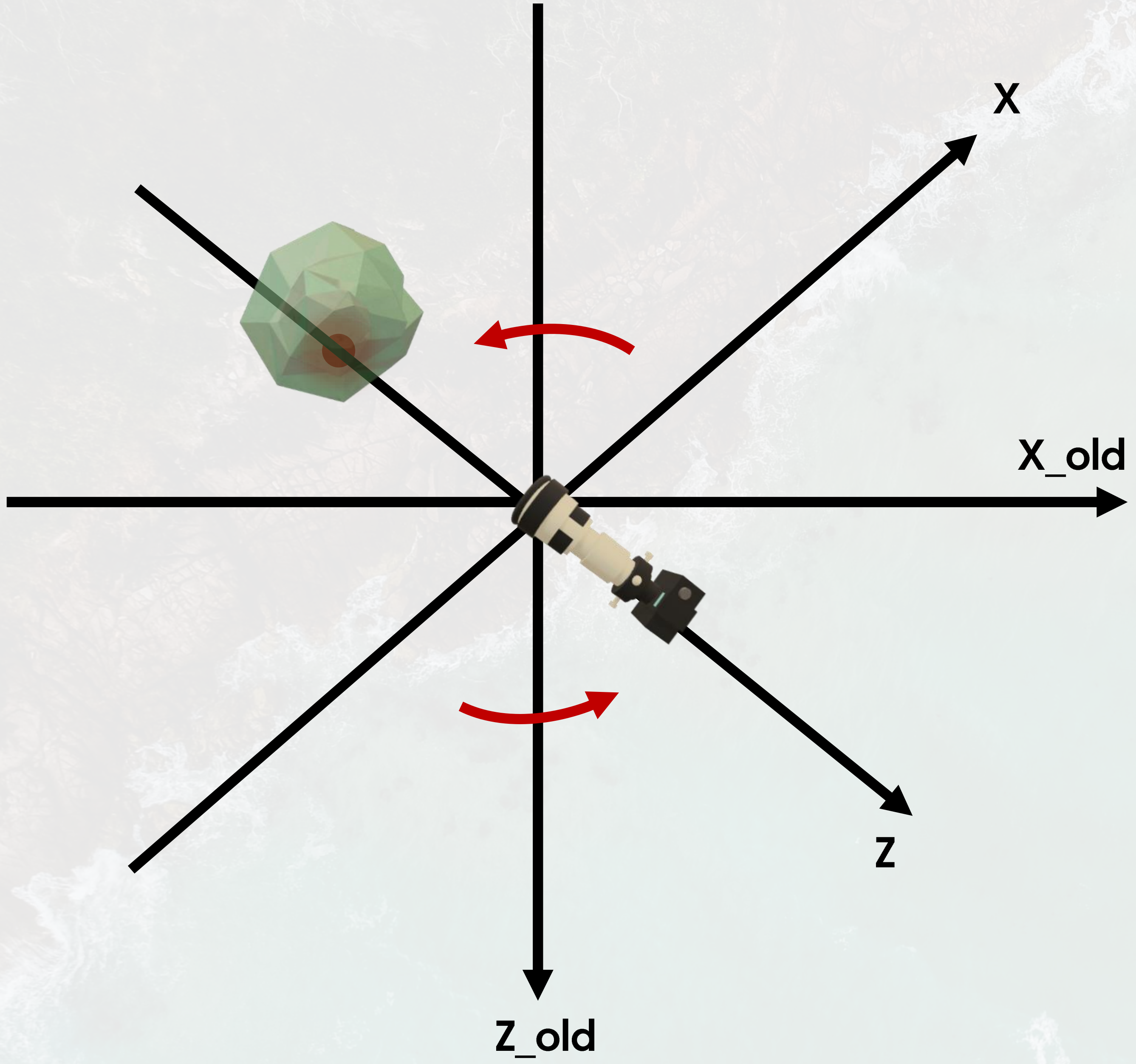
$$[\mathbf{-3} \quad \mathbf{0} \quad \mathbf{-4} \quad \mathbf{1}] * \begin{bmatrix} 0,8 & 0 & 0,6 & 0 \\ 0 & 1 & 0 & 0 \\ -0,6 & 0 & 0,8 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

$$= [-2,4 + 2,4 \quad 0 \quad -1,8 - 3,2 \quad 1] = [\mathbf{0} \quad \mathbf{0} \quad \mathbf{-5} \quad \mathbf{1}]$$

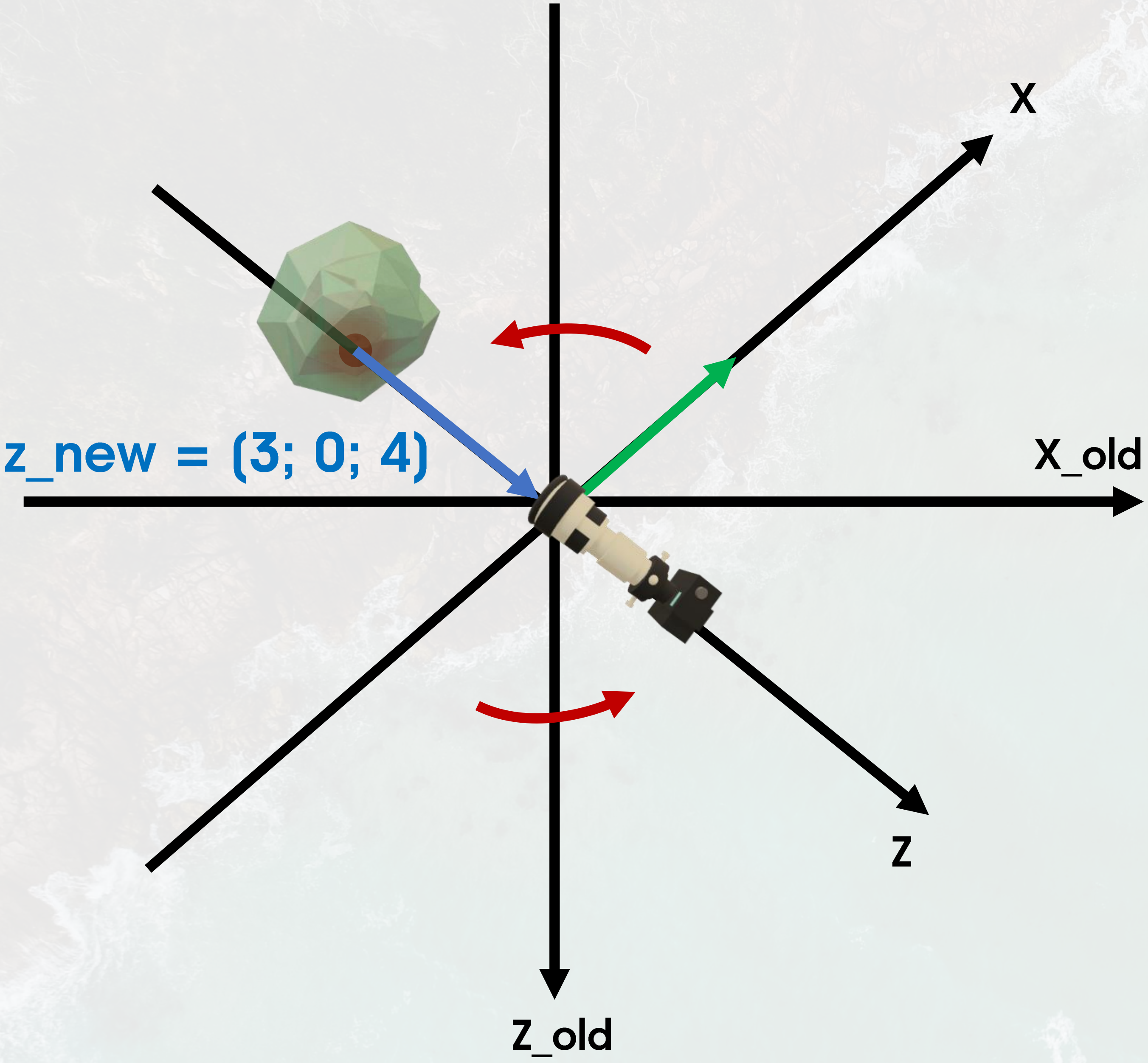
Заменим задачу перемещения точки на задачу перемещения координатных осей



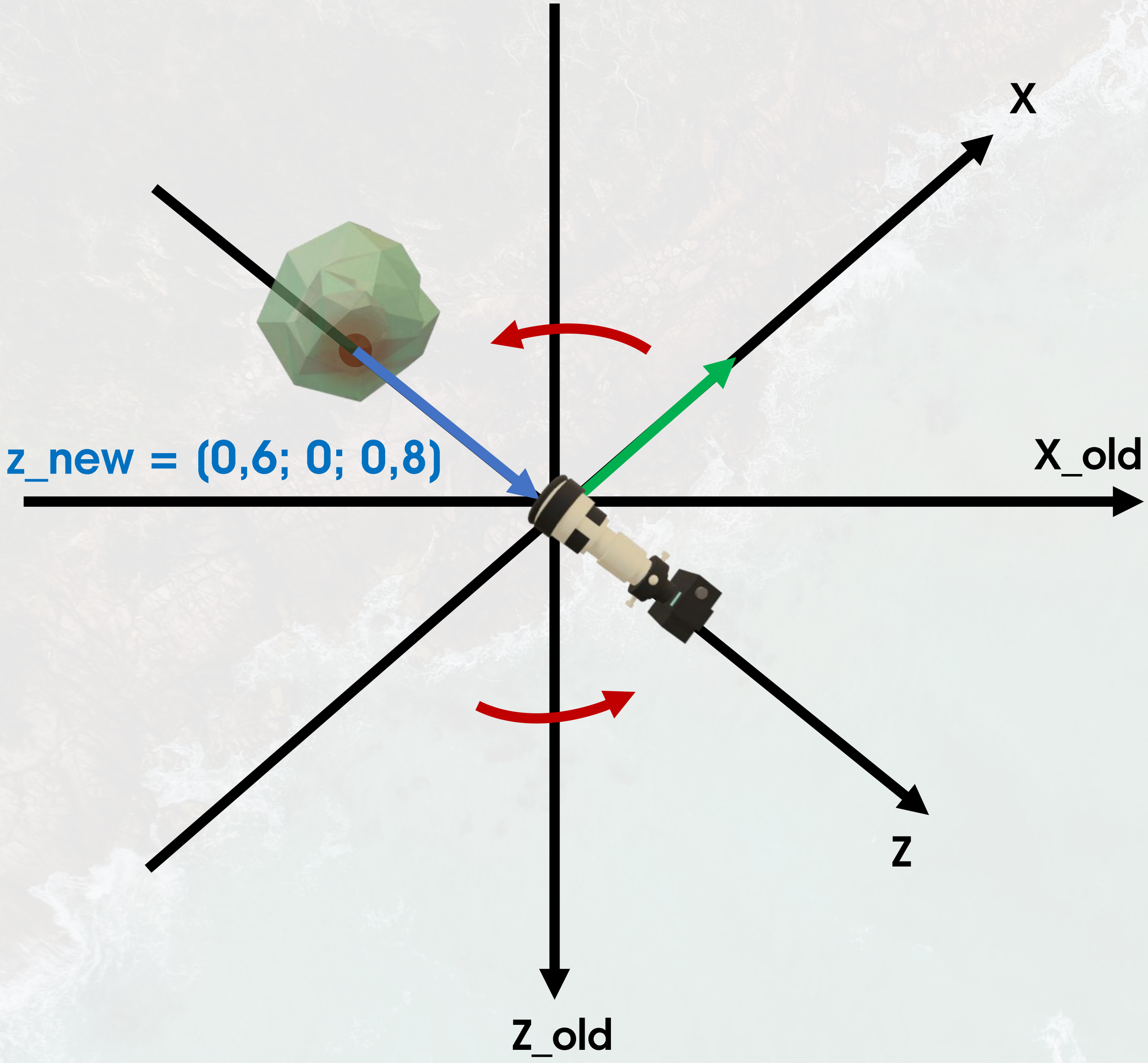
Заменим задачу перемещения точки на задачу перемещения координатных осей



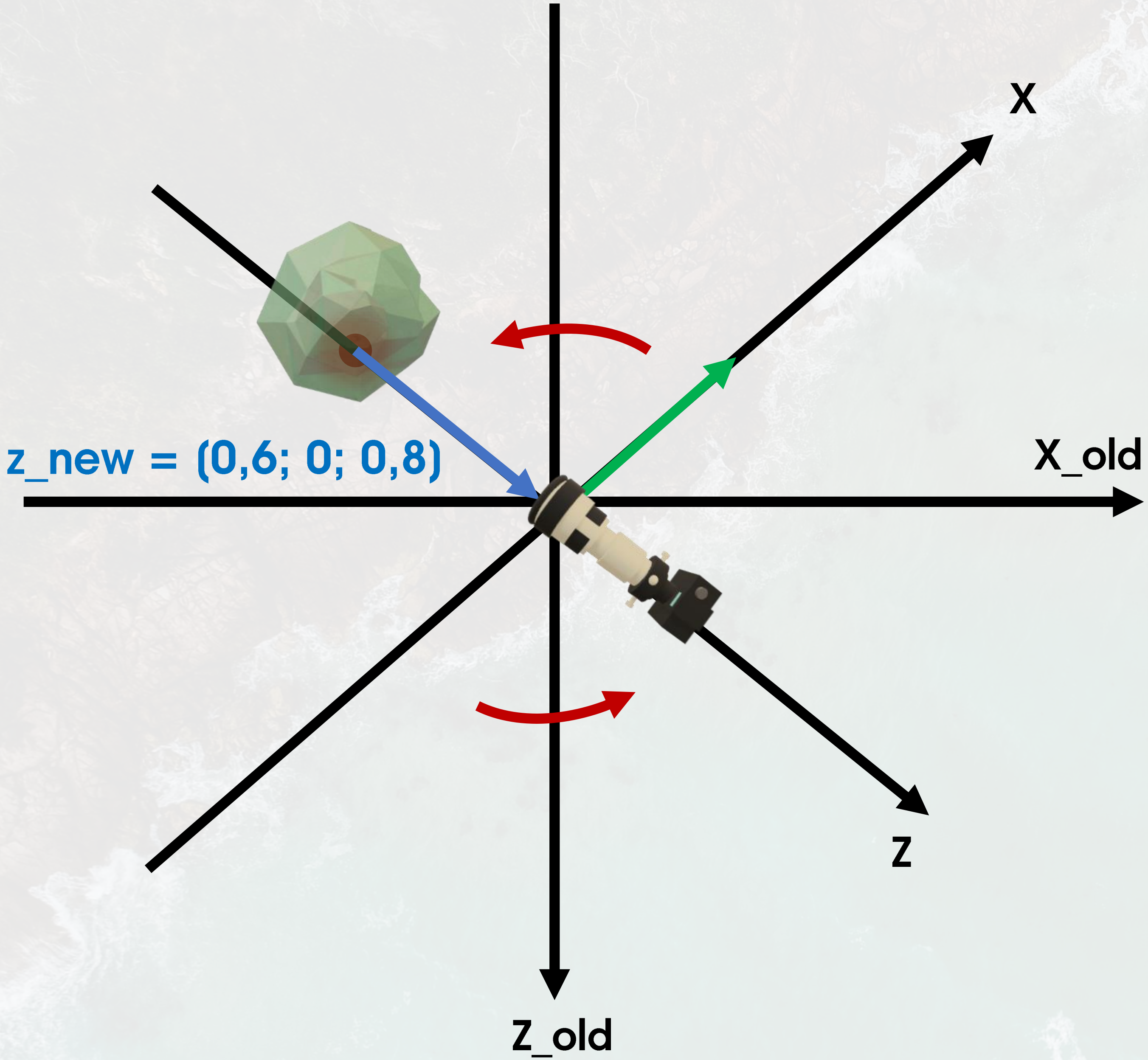
Заменим задачу перемещения точки на задачу перемещения координатных осей



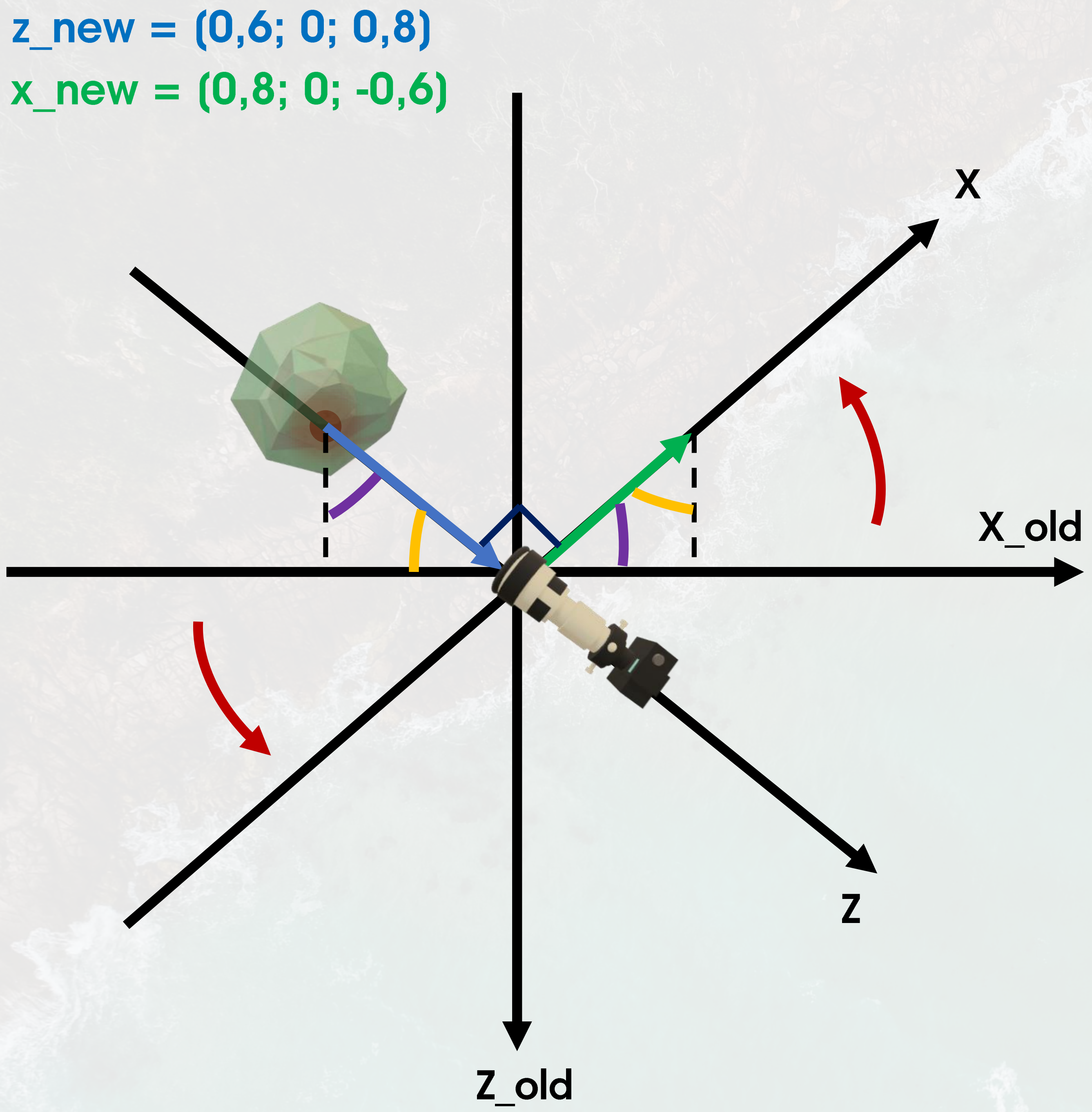
Определим координаты вектора, задающего новое направление оси Z, в старой системе координат



Выполним нормализацию



Определим координаты вектора, задающего новое направление оси X, в старой системе координат



Определим координаты вектора, задающего новое направление оси X, в старой системе координат

Интерпретация вектора z_new – определение того, как распределится старое значение z на координатные оси, если ось Z будет ориентирована в направлении вектора z_new

Интерпретация вектора z_new в контексте матрицы поворота вокруг оси OY – определение угла поворота оси OZ относительно исходного положения

$$\begin{bmatrix} 0{,}8 & 0 & -0{,}6 & 0 \\ 0 & 1 & 0 & 0 \\ 0{,}6 & 0 & 0{,}8 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Матрица, состоящая из координат новых векторов

$$\begin{bmatrix} 0{,}8 & 0 & -0{,}6 & 0 \\ 0 & 1 & 0 & 0 \\ 0{,}6 & 0 & 0{,}8 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \mathrm{c} & 0 & -s & 0 \\ 0 & 1 & 0 & 0 \\ s & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Сравним с матрицей поворота вокруг оси OY

$$\cos(\alpha) = 0{,}8 \ \sin(\alpha) = 0{,}6$$

$$\alpha \approx 36{,}89$$

$$\begin{bmatrix} 0{,}8 & 0 & -0{,}6 & 0 \\ 0 & 1 & 0 & 0 \\ 0{,}6 & 0 & 0{,}8 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

С помощью матрицы можно сделать поворот объекта на 36,89 градусов, но нам нужно повернуть не объект, а камеру

Для перехода от поворота координатных осей обратно к повороту объекта возьмем такую матрицу, которая бы выполняла такой же поворот, но в обратном направлении

$$\begin{bmatrix} 0{,}8 & 0 & -0{,}6 & 0 \\ 0 & 1 & 0 & 0 \\ 0{,}6 & 0 & 0{,}8 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \ ? = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Обратная матрица совпадает с той, которую мы получили ранее другим способом

$$\begin{bmatrix} 0{,}8 & 0 & -0{,}6 & 0 \\ 0 & 1 & 0 & 0 \\ 0{,}6 & 0 & 0{,}8 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} \mathbf{0,8} & \mathbf{0} & \mathbf{0,6} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathbf{-0,6} & \mathbf{0} & \mathbf{0,8} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 0,8 & 0 & 0,6 & 0 \\ 0 & 1 & 0 & 0 \\ -0,6 & 0 & 0,8 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \mathrm{c} & 0 & -s & 0 \\ 0 & 1 & 0 & 0 \\ s & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Сравним с матрицей поворота вокруг оси OY

cos(α) = 0,8 sin(α) = -0,6

α ≈ -36,89

Обобщение результатов

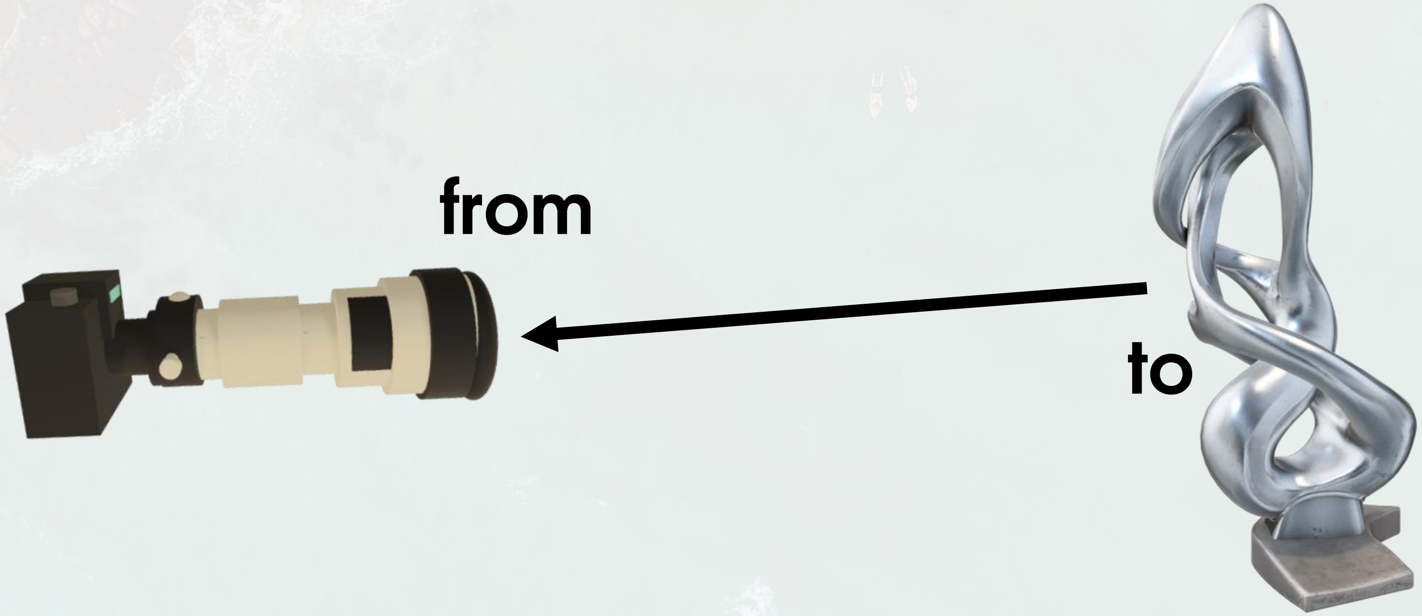
Матрица слежения за объектом

$$\begin{bmatrix} x_{new_x} & x_{new_y} & x_{new_z} & 0 \\ y_{new_x} & y_{new_y} & y_{new_z} & 0 \\ z_{new_x} & z_{new_y} & z_{new_z} & 0 \\ from_x & from_y & from_z & 1 \end{bmatrix}$$

новые координаты камеры

$$z_{new} = \left(\frac{from_x - to_x}{length}, \frac{from_y - to_y}{length}, \frac{from_z - to_z}{length} \right)$$

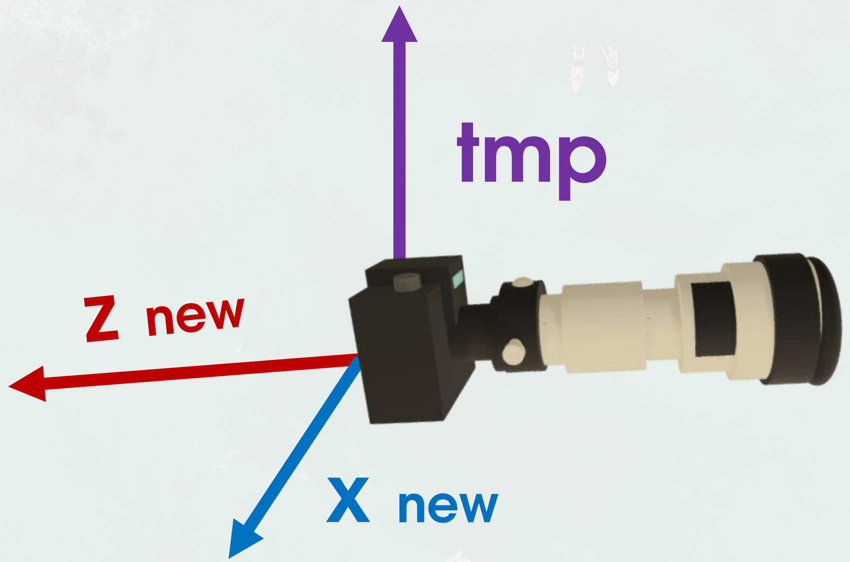
$$length = \sqrt{(from_x - to_x)^2 + \left(from_y - to_y\right)^2 + (from_z - to_z)^2}$$



$$tmp = (0,1,0)$$

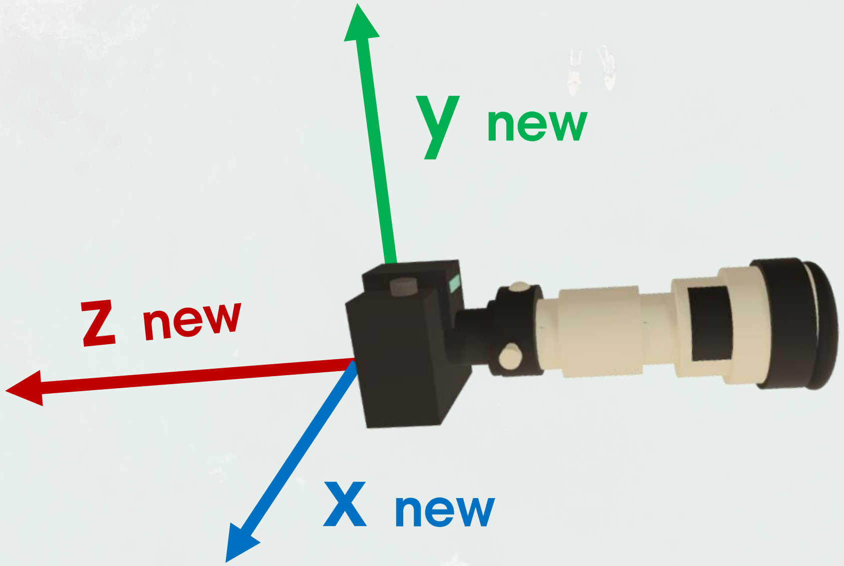
$$x_{new}(z_{new}, tmp) =$$

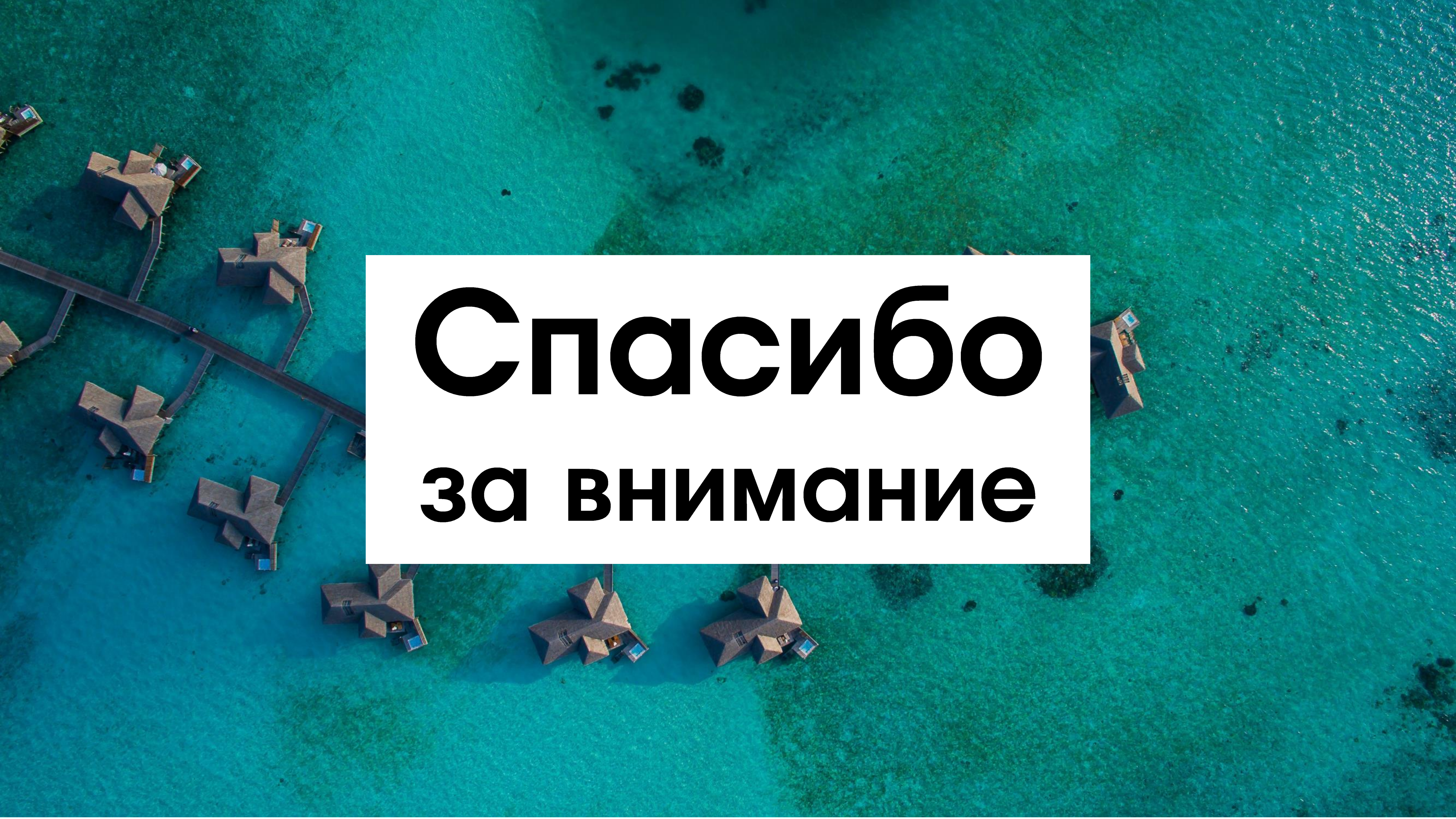
$$= (z_{new_y} * tmp_z - z_{new_z} * tmp_y, z_{new_z} * tmp_x -$$
$$-z_{new_x} * tmp_z, z_{new_x} * tmp_y - z_{new_y} * tmp_x)$$



$$y_{new}(\mathrm{z}_{\mathrm{new}}, x_{new}) =$$

$$= (z_{new_y} * x_{new_z} - z_{new_z} * x_{new_y}, z_{new_z} * x\_new_x -$$
$$-z_{new_x} * x_{new_z}, z_{new_x} * x_{new_y} - z_{new_y} * x_{new_x})$$





Спасибо за внимание

https://github.com/zeionara/webgl_camera_imitation