

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное  
образовательное учреждение высшего образования  
«Национальный исследовательский университет ИТМО»  
(Университет ИТМО)

Факультет систем управления и робототехники

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3  
по дисциплине  
*«Практическая линейная алгебра»*

по теме:  
МАТРИЦЫ В 3D-ГРАФИКЕ

Студент:  
*Группа № R3335*

*Зыкин Л. В.*

Предподаватель:  
*должность, уч. степень, уч. звание*

*Догадин Е. В.*

Санкт-Петербург  
2025

# **Содержание**

# Введение

В данной лабораторной работе изучаются матрицы преобразования пространства, которые широко используются в 3D-графике. Основная цель - познакомиться с различными типами матричных преобразований и их применением для создания трехмерных сцен.

## Однородные координаты

В 3D-графике используется система однородных координат, где точка  $(x, y, z)$  представляется как  $(x, y, z, w)$ , где  $w$  - однородная координата. Это позволяет объединить линейные преобразования (вращение, масштабирование) и аффинные преобразования (перемещение) в единую матричную форму.

Преобразование из однородных координат в декартовы:

$$(x, y, z) = \left( \frac{x}{w}, \frac{y}{w}, \frac{z}{w} \right) \quad (1)$$

## Задание 1: Создание кубика

### Теоретические основы

Кубик в 3D-пространстве определяется восемью вершинами и шестью гранями. В однородных координатах вершины кубика с ребром длины 2 и центром в начале координат имеют вид:

$$\text{Вершины} = \begin{bmatrix} -1 & 1 & 1 & -1 & -1 & 1 & 1 & -1 \\ -1 & -1 & 1 & 1 & -1 & -1 & 1 & 1 \\ -1 & -1 & -1 & -1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad (2)$$

Границы кубика определяются индексами вершин:

$$\text{Границы} = \begin{bmatrix} 0 & 1 & 5 & 4 \\ 1 & 2 & 6 & 5 \\ 2 & 3 & 7 & 6 \\ 3 & 0 & 4 & 7 \\ 0 & 1 & 2 & 3 \\ 4 & 5 & 6 & 7 \end{bmatrix} \quad (3)$$

### Практическая реализация

Базовый кубик был создан с использованием Python и библиотеки matplotlib. Код включает функции для отрисовки 3D объектов с поддержкой прозрачности и различных цветов.

## Почему используются 4-компонентные векторы?

Использование однородных координат  $(x, y, z, w)$  вместо декартовых координат  $(x, y, z)$  имеет несколько важных преимуществ:

1. **Единообразие преобразований:** Все аффинные преобразования (включая перемещение) можно представить в виде матричного умножения.
2. **Удобство композиции:** Композиция преобразований сводится к умножению матриц.

Тестовый кубик

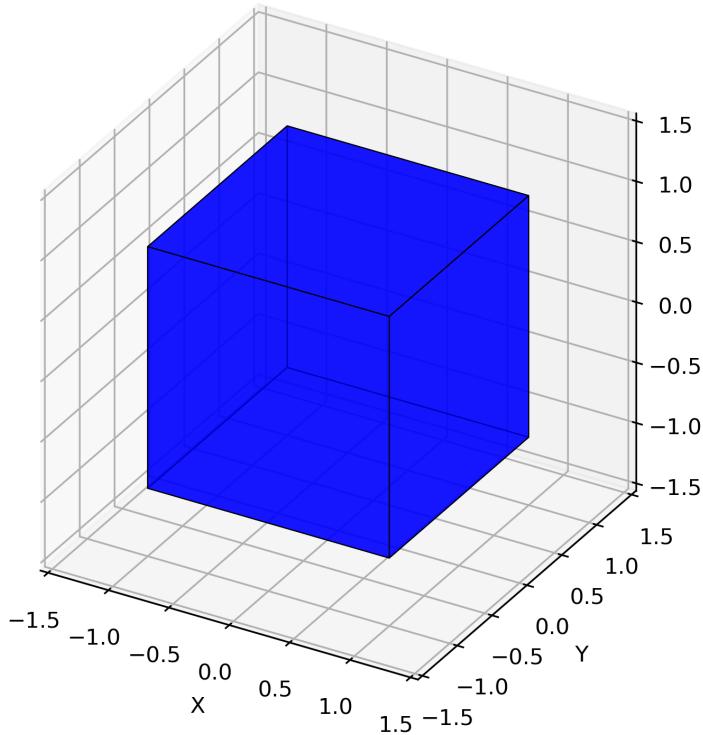


Рис. 1: Базовый кубик в 3D пространстве

3. **Поддержка перспективы:** Однородные координаты естественным образом поддерживают перспективные преобразования.
4. **Математическая элегантность:** Все преобразования становятся линейными в однородном пространстве.

## Как задать другие фигуры?

Для создания других 3D фигур необходимо:

1. **Определить вершины:** Задать координаты всех вершин фигуры в однородных координатах.
2. **Определить грани:** Указать индексы вершин, образующих каждую грань.
3. **Задать цвета и прозрачность:** Для визуализации.
4. **Использовать функции отрисовки:** Применить те же функции, что и для кубика.

Примеры других фигур:

- **Тетраэдр:** 4 вершины, 4 треугольные грани
- **Октаэдр:** 6 вершин, 8 треугольных граней
- **Призма:** 6 вершин, 5 граней (2 основания + 3 боковые)
- **Пирамида:** 5 вершин, 5 граней (1 основание + 4 боковые)

## Задание 2: Масштабирование кубика

### Матрица масштабирования

Матрица масштабирования  $S$  имеет вид:

$$S = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

где  $s_x, s_y, s_z$  - коэффициенты масштабирования по осям X, Y, Z соответственно.

### Виды масштабирования

1. **Равномерное масштабирование:**  $s_x = s_y = s_z = k$
2. **Неравномерное масштабирование:**  $s_x \neq s_y \neq s_z$
3. **Растяжение по одной оси:** например,  $s_x > 1, s_y = s_z = 1$
4. **Сжатие:**  $s_i < 1$  для некоторых осей

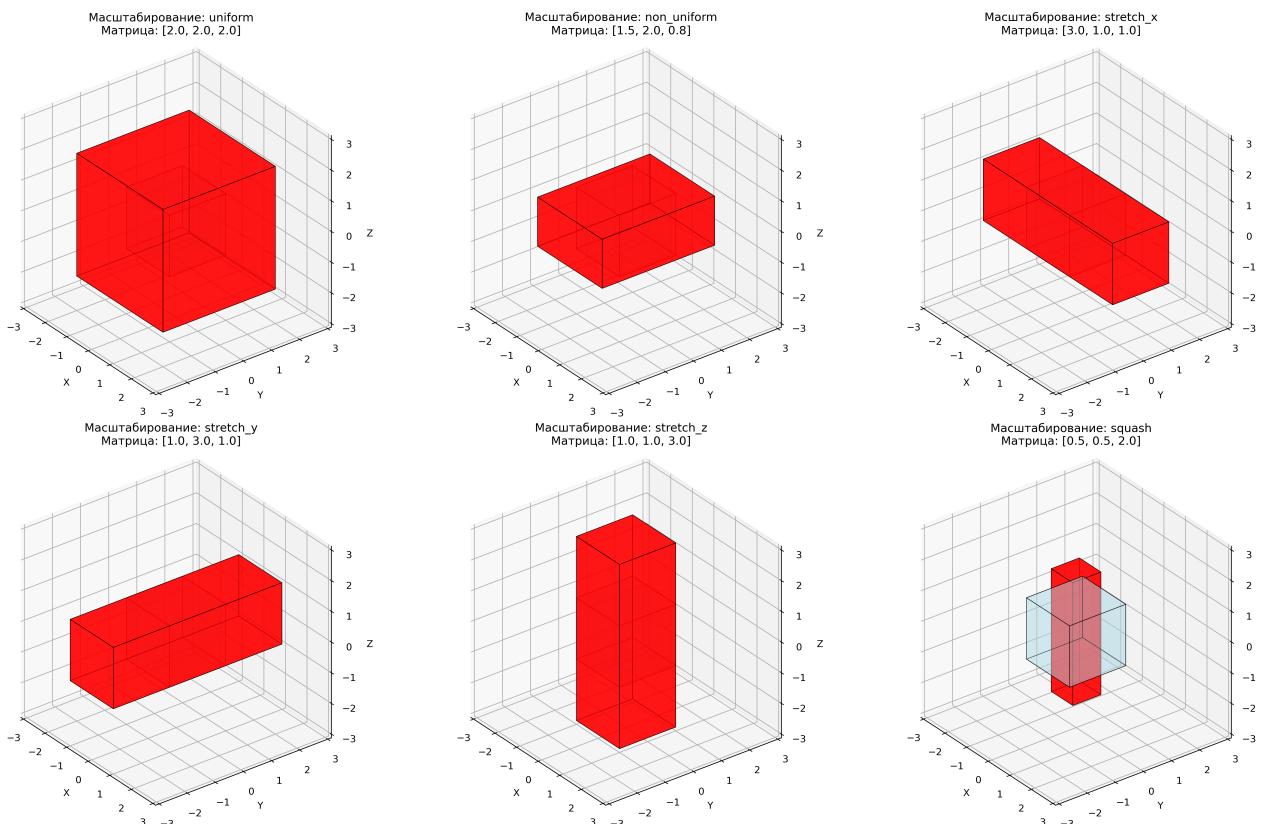


Рис. 2: Различные виды масштабирования кубика

## Сравнение TS и ST

Важное свойство матричных преобразований - некоммутативность масштабирования и перемещения:

$$TS \neq ST \quad (5)$$

где  $T$  - матрица перемещения,  $S$  - матрица масштабирования.

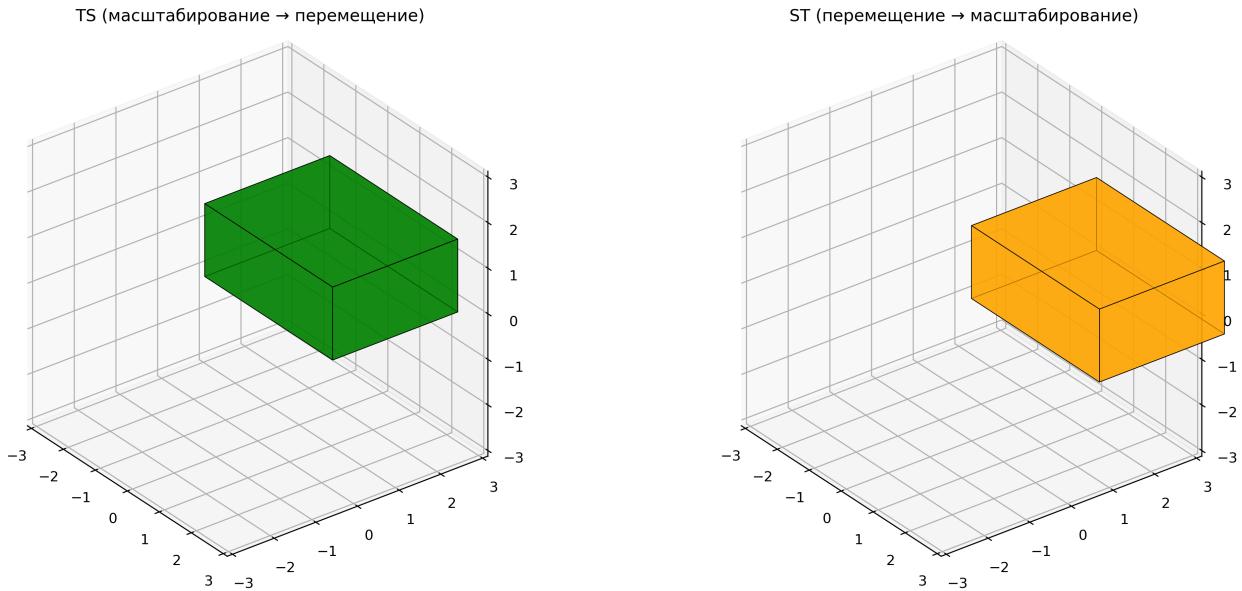


Рис. 3: Сравнение преобразований TS и ST

## Задание 3: Перемещение кубика

### Матрица перемещения

Матрица перемещения  $T$  имеет вид:

$$T = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (6)$$

где  $t_x, t_y, t_z$  - компоненты вектора перемещения.

### Свойства перемещения

- Перемещение коммутирует с самим собой:  $T_1 T_2 = T_2 T_1$
- Перемещение не коммутирует с масштабированием:  $TS \neq ST$
- Перемещение не коммутирует с вращением:  $TR \neq RT$

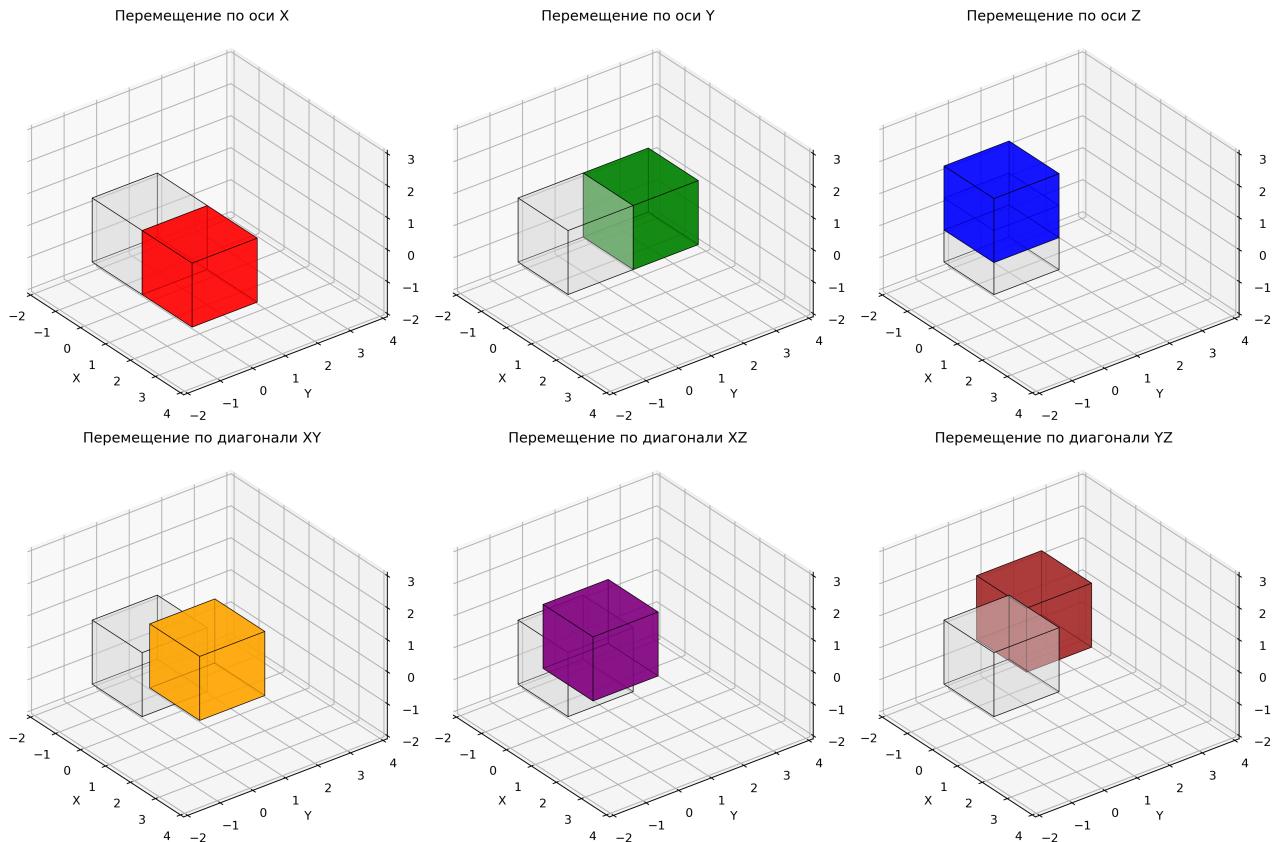


Рис. 4: Различные виды перемещения кубика

## Демонстрация перемещений

### Сравнение TS и ST для перемещения

### Задание 4: Вращение кубика

#### Вращение вокруг произвольной оси

Для вращения вокруг произвольной оси  $\mathbf{v} = [v_x, v_y, v_z]$  на угол  $\theta$  используется формула:

$$\mathbf{v} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}, \quad J = \frac{1}{\|\mathbf{v}\|} \begin{bmatrix} 0 & -v_z & v_y & 0 \\ v_z & 0 & -v_x & 0 \\ -v_y & v_x & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (7)$$

$$R_{\mathbf{v}}(\theta) = e^{J\theta} \quad (8)$$

#### Матрицы вращения вокруг координатных осей

При выборе вектора  $\mathbf{v}$  вдоль осей координат получаем стандартные матрицы вращения:

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta & 0 \\ 0 & \sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (9)$$

TS: Сначала масштабирование, потом перемещение

ST: Сначала перемещение, потом масштабирование

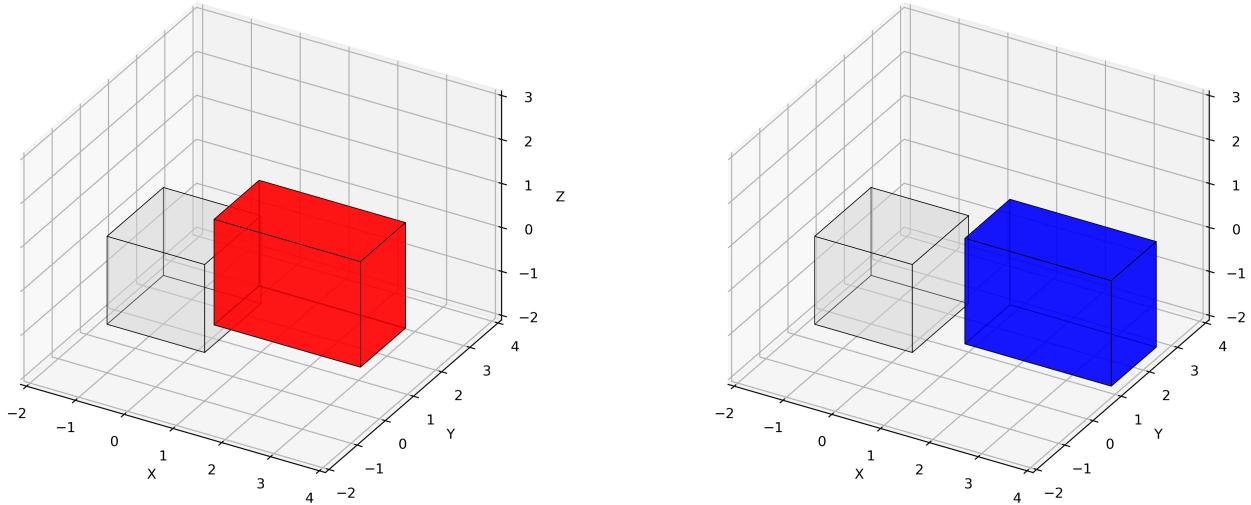


Рис. 5: Сравнение преобразований TS и ST для перемещения

$$R_y(\phi) = \begin{bmatrix} \cos \phi & 0 & \sin \phi & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \phi & 0 & \cos \phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (10)$$

$$R_z(\psi) = \begin{bmatrix} \cos \psi & -\sin \psi & 0 & 0 \\ \sin \psi & \cos \psi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (11)$$

## Теорема вращения Эйлера

Любое вращение в 3D-пространстве можно представить как композицию трех вращений вокруг координатных осей:

$$R = R_x(\theta)R_y(\phi)R_z(\psi) \quad (12)$$

Однако это представление не является единственным и может привести к проблеме "карданного подвеса"(gimbal lock).

## Достаточность композиции вращений

Композиция  $R_x(\theta)R_y(\phi)R_z(\psi)$  **не является достаточной** для описания всех возможных вращений в 3D-пространстве. Проблемы:

1. **Карданный подвес:** При  $\phi = \pm \frac{\pi}{2}$  теряется одна степень свободы
2. **Неединственность:** Одно и то же вращение может быть представлено разными наборами углов
3. **Сингулярности:** При определенных углах возникают проблемы с численной стабильностью

## Восстановление оси вращения

Ось вращения может быть восстановлена из матрицы вращения  $R$ :

1. Вычислить след матрицы:  $\text{tr}(R) = 1 + 2 \cos \theta$

2. Найти угол вращения:  $\theta = \arccos\left(\frac{\text{tr}(R)-1}{2}\right)$

3. Вычислить ось вращения:  $\mathbf{v} = \frac{1}{2 \sin \theta} \begin{bmatrix} R_{32} - R_{23} \\ R_{13} - R_{31} \\ R_{21} - R_{12} \end{bmatrix}$

Это возможно благодаря тому, что любое вращение в 3D имеет неподвижную ось (теорема Эйлера).

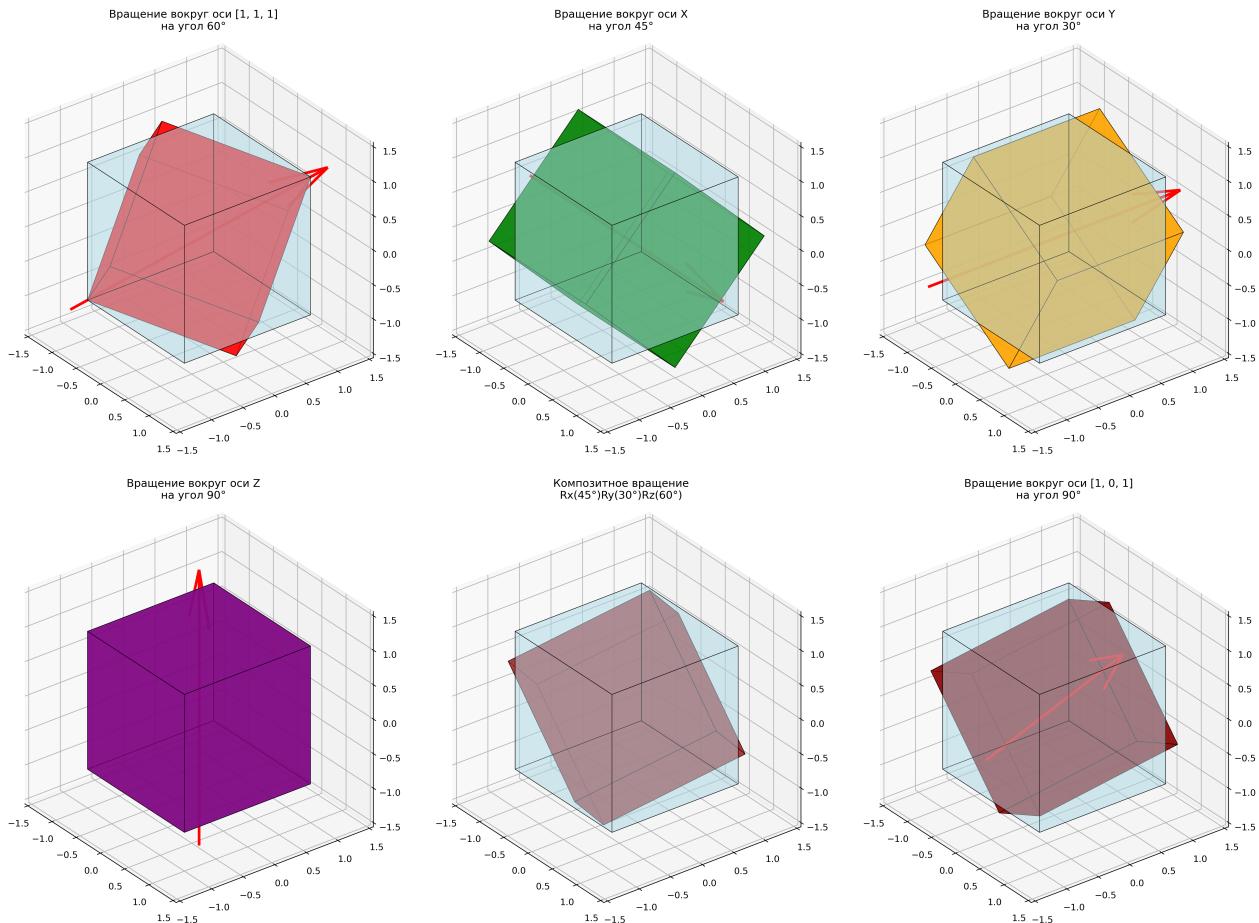


Рис. 6: Вращения кубика вокруг различных осей

## Сравнение формул вращения

### Задание 5: Вращение вокруг вершины

#### Алгоритм вращения вокруг точки

Для вращения объекта вокруг произвольной точки  $P = (p_x, p_y, p_z)$  используется композиция преобразований:

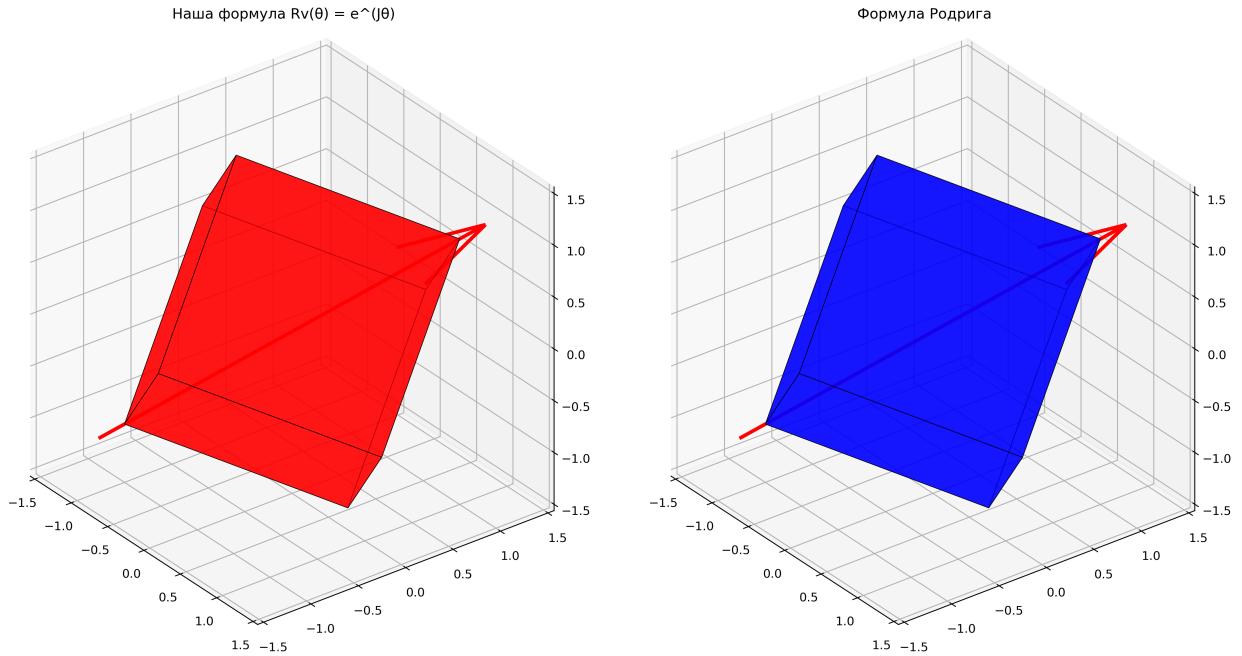


Рис. 7: Сравнение формулы  $R_v(\theta) = e^{J\theta}$  с формулой Родрига

1. Перенос точки  $P$  в начало координат:  $T_1 = T(-p_x, -p_y, -p_z)$
2. Вращение вокруг оси:  $R = R_v(\theta)$
3. Обратный перенос:  $T_2 = T(p_x, p_y, p_z)$

Итоговая матрица преобразования:

$$M = T_2 \cdot R \cdot T_1 \quad (13)$$

## Практическая реализация

### Задание 6: Реализация камеры

#### Матрица вида (View Matrix)

Матрица камеры  $C$  создается следующим образом:

1. Вычисление вектора направления камеры:  $\text{forward} = \frac{\text{target-position}}{\|\text{target-position}\|}$
2. Вычисление правого вектора:  $\text{right} = \frac{\text{forward} \times \text{up}}{\|\text{forward} \times \text{up}\|}$
3. Вычисление вектора "вверх":  $\text{up} = \text{right} \times \text{forward}$
4. Создание матрицы поворота  $R$  и перемещения  $T$
5. Матрица камеры:  $C = R \cdot T$

#### Эффект камеры

Применение матрицы камеры  $C$  к объектам сцены эквивалентно перемещению камеры в начало координат с стандартной ориентацией.

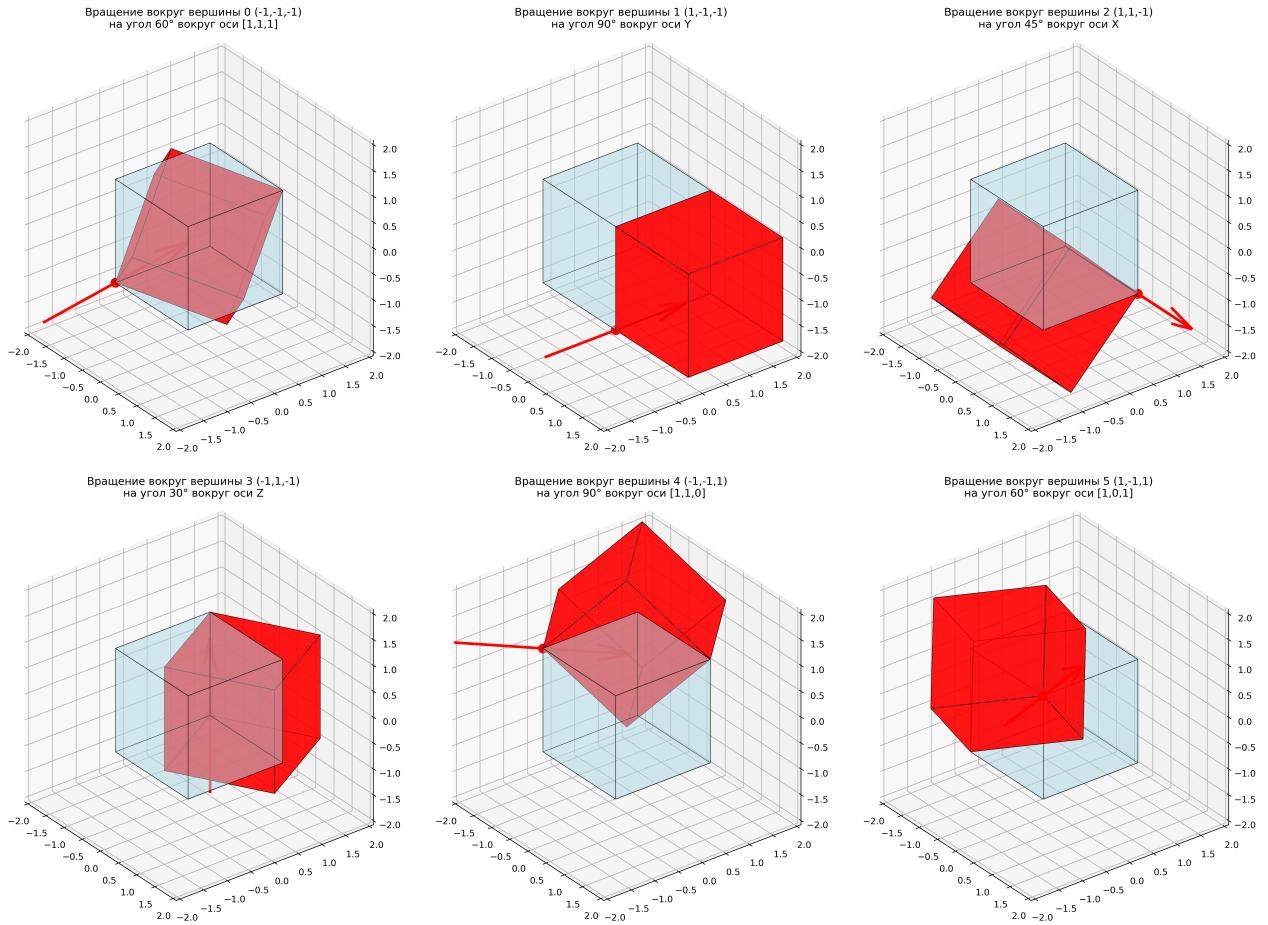


Рис. 8: Вращения кубика вокруг различных вершин

## Создание сцены

Для демонстрации работы камеры создается сцена из нескольких кубиков, расположенных в разных позициях. Используется команда `ax.view_init(azim=0, elev=-90)` для просмотра сцены "снизу".

## Выбор позиций камеры

Выбираются различные позиции камеры  $T_c$  и матрицы поворота  $R_c(\theta)$ :

- Камера в позиции  $[5, 5, 3]$  смотрит на центр сцены
- Камера в позиции  $[-4, -4, 2]$  смотрит на центр сцены
- Камера в позиции  $[0, 0, 8]$  (вид сверху)
- Камера в позиции  $[6, 6, 4]$  (вид под углом)

## Обратная матрица камеры

Находится матрица  $C^{-1}$ , которая перемещает камеру из выбранной позиции в начало координат с стандартной ориентацией. Применение этой матрицы ко всем объектам сцены создает эффект, как будто камера находится в начале координат.

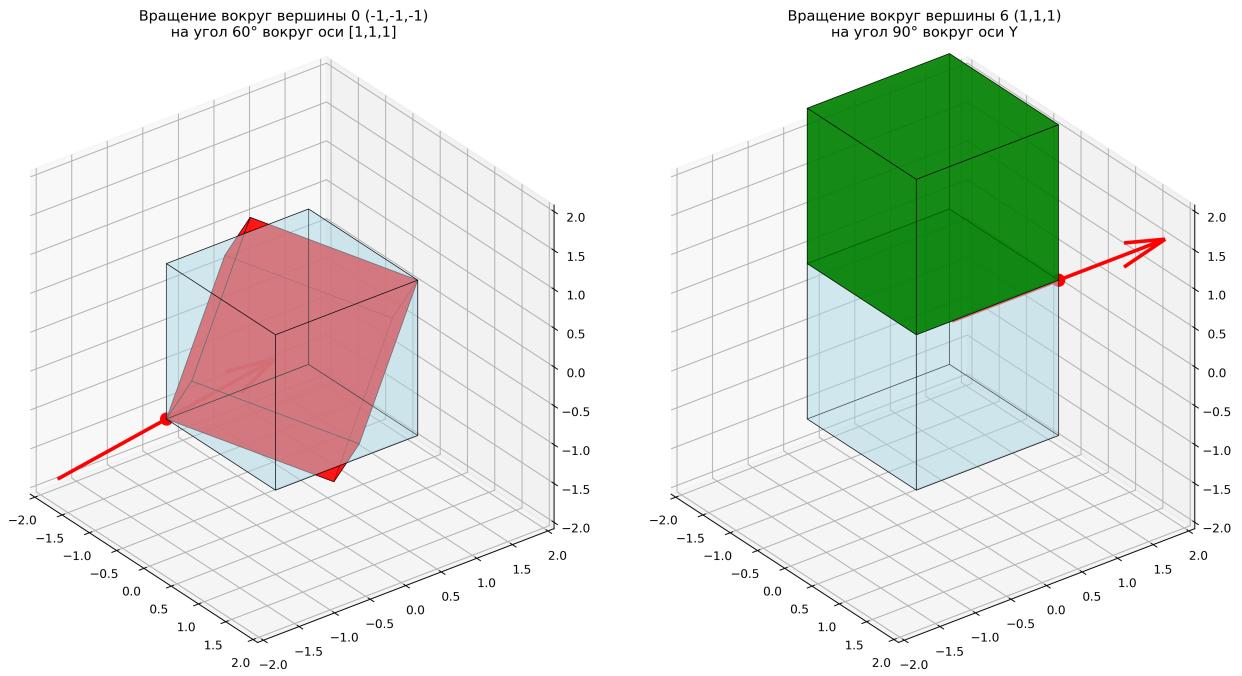


Рис. 9: Сравнение вращений вокруг разных вершин

## Сравнение эффектов

## Задание 7: Реализация перспективы

### Матрица перспективной проекции

Матрица перспективной проекции  $P$  имеет вид:

$$P = \begin{bmatrix} \frac{f}{\text{aspect}} & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & \frac{\text{far} + \text{near}}{\text{near} - \text{far}} & \frac{2 \cdot \text{far} \cdot \text{near}}{\text{near} - \text{far}} \\ 0 & 0 & -1 & 0 \end{bmatrix} \quad (14)$$

где  $f = \frac{1}{\tan(\text{FOV}/2)}$  - фокусное расстояние.

### Параметры перспективы

- **FOV (Field of View)**: угол поля зрения в градусах
- **Aspect Ratio**: соотношение сторон экрана
- **Near Plane**: ближняя плоскость отсечения
- **Far Plane**: дальняя плоскость отсечения

### Сравнение с ортографической проекцией

### Структура матрицы перспективы

Матрица перспективы  $P$  имеет следующую структуру:

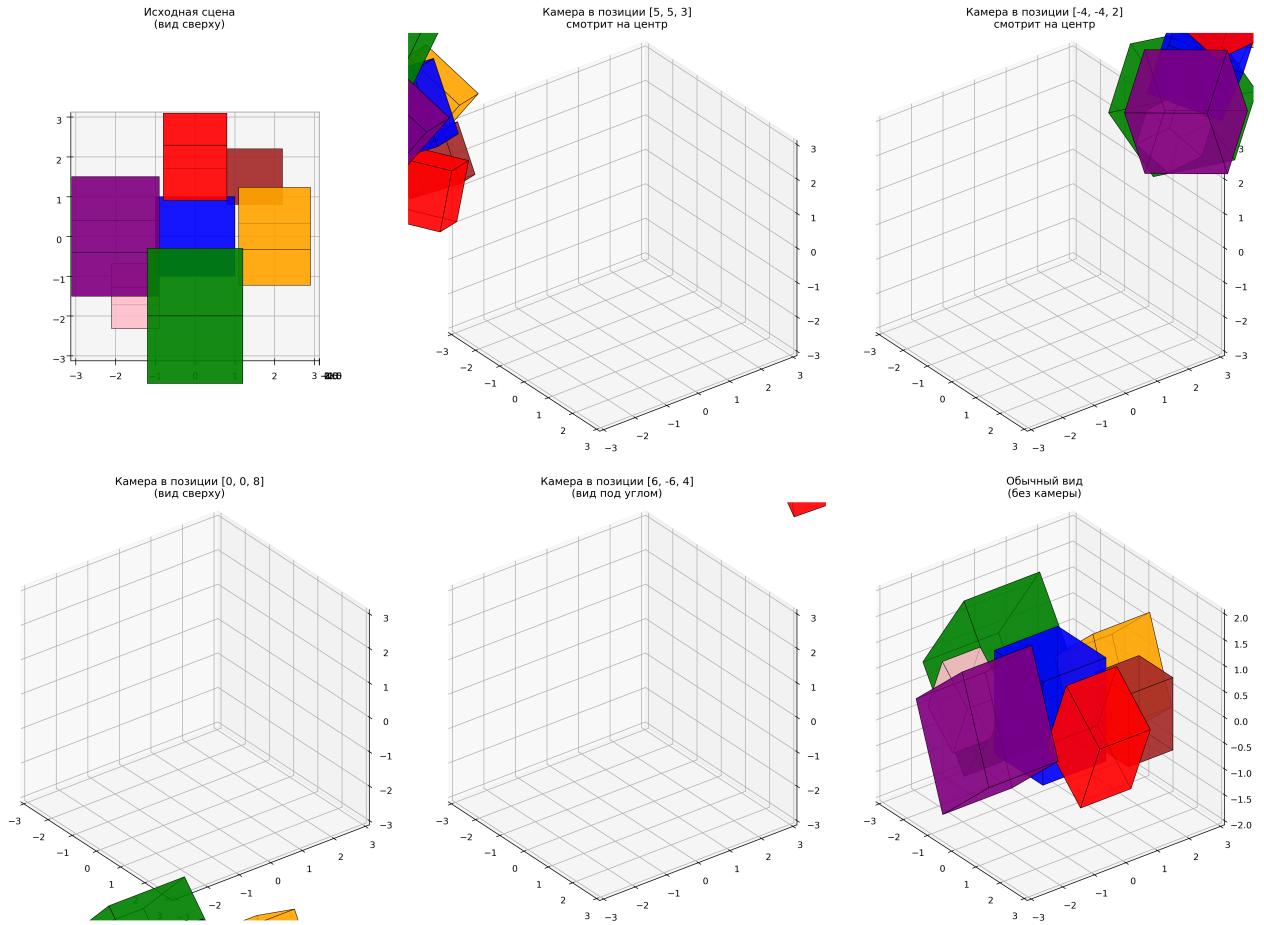


Рис. 10: Реализация камеры с различными позициями

- **Первые два элемента:** Масштабирование по осям X и Y в зависимости от FOV и aspect ratio
- **Третий элемент:** Преобразование координаты Z для создания эффекта глубины
- **Четвертый элемент:** Перенос в координату W для перспективного деления
- **Последняя строка:**  $[0, 0, -1, 0]$  - ключевой элемент для перспективного деления

## Параметры матрицы

- **FOV (Field of View):** Определяет угол обзора камеры (обычно 45-90°)
- **Aspect Ratio:** Соотношение ширины к высоте экрана
- **Near Plane:** Ближняя плоскость отсечения (обычно 0.1-1.0)
- **Far Plane:** Дальняя плоскость отсечения (обычно 100-1000)

## Исследование деформации пространства

Используется стандартное вращение графика для исследования того, как перспективная проекция деформирует пространство. Дальние объекты становятся меньше, создавая эффект глубины.

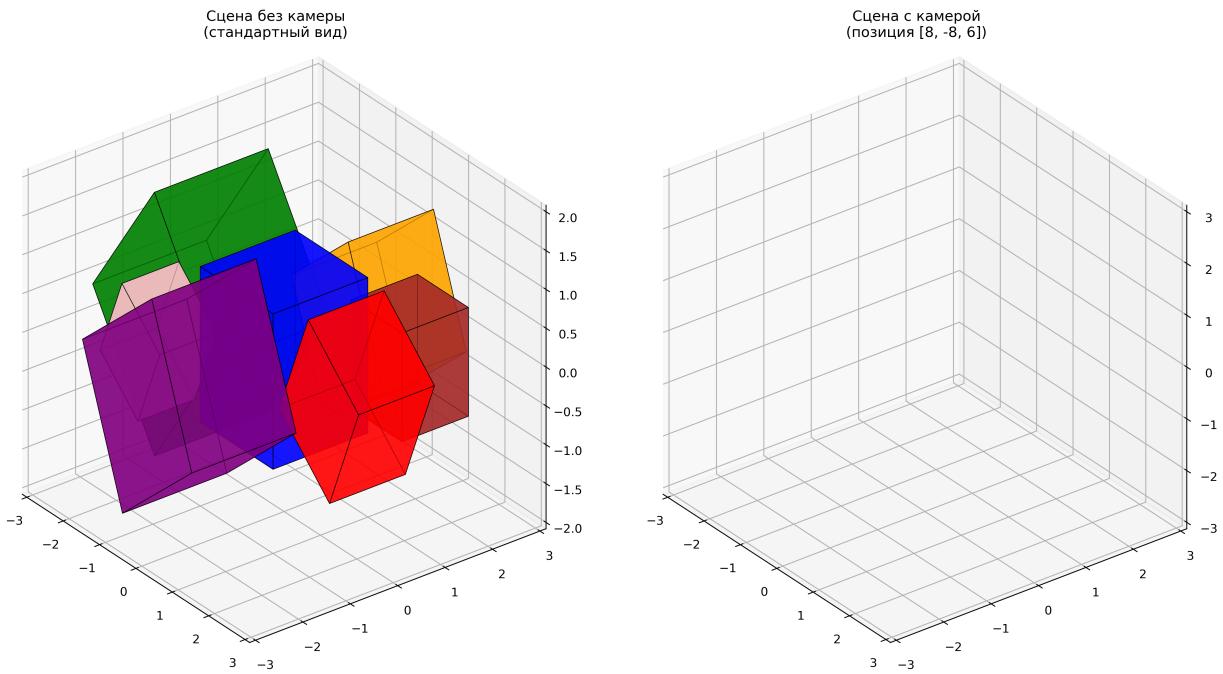


Рис. 11: Сравнение сцены с камерой и без камеры

## Эффект расстояния

## Основные результаты

### Ключевые выводы

1. **Однородные координаты** позволяют объединить линейные и аффинные преобразования в единую матричную форму.
2. **Некоммутативность преобразований:** порядок применения матриц важен.  $TS \neq ST$ ,  $TR \neq RT$ .
3. **Вращение вокруг произвольной оси** можно реализовать с помощью экспоненциальной матрицы  $R_v(\theta) = e^{J\theta}$ .
4. **Теорема вращения Эйлера** утверждает, что любое вращение можно представить как композицию трех вращений вокруг координатных осей.
5. **Матрица камеры** позволяет преобразовать сцену так, как будто камера находится в начале координат.
6. **Перспективная проекция** создает эффект глубины, делая дальние объекты меньше ближних.

### Практическая значимость

Полученные результаты имеют важное значение для:

- Разработки 3D-графических приложений
- Создания компьютерных игр

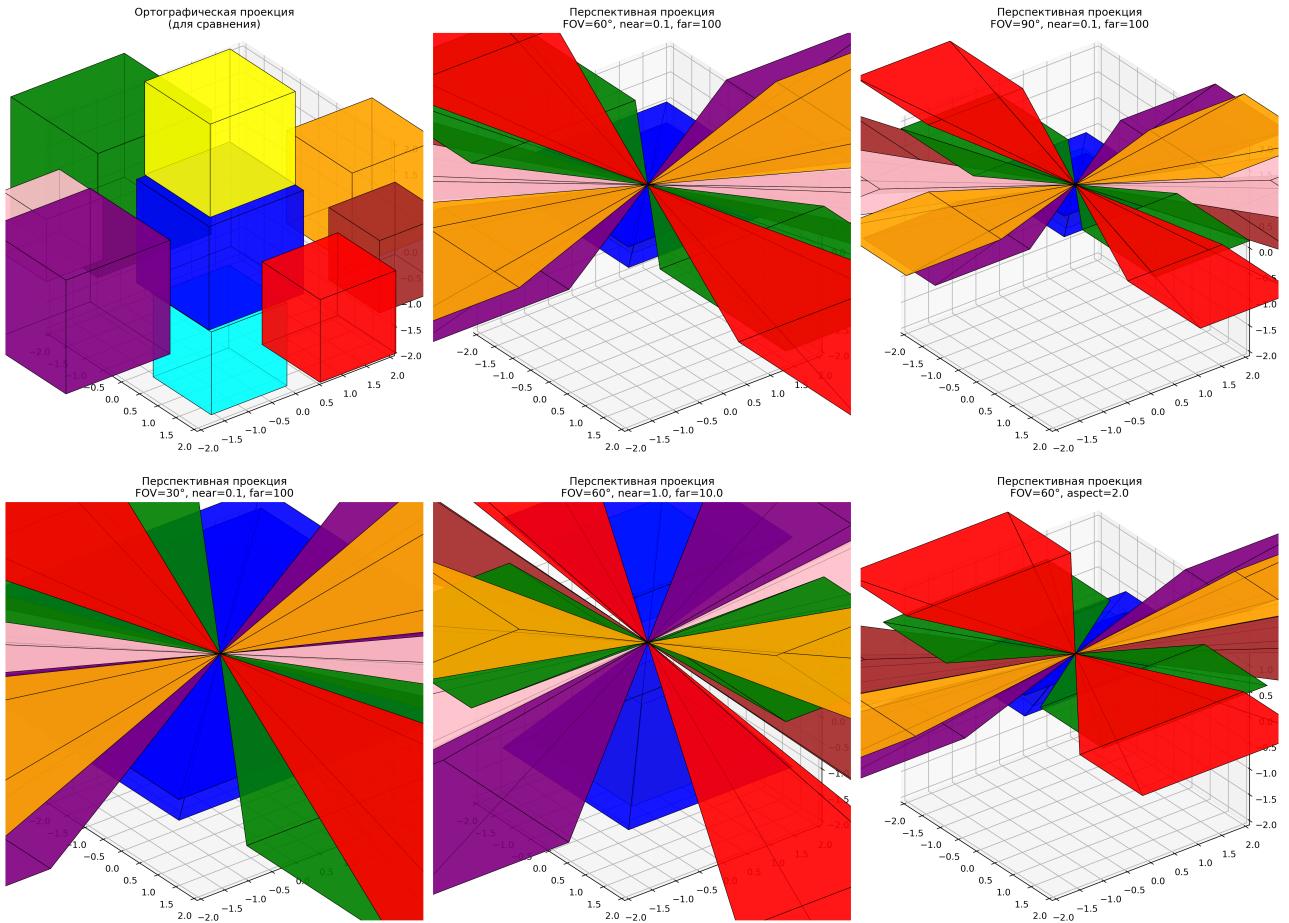


Рис. 12: Перспективные проекции с различными параметрами

- Моделирования в CAD-системах
- Компьютерной анимации
- Визуализации научных данных

## Математические основы

Все преобразования в данной работе основаны на матричной алгебре и теории групп преобразований. Использование однородных координат позволяет представить все аффинные преобразования в виде матричного умножения, что обеспечивает эффективную реализацию в компьютерной графике.

## Заключение

В ходе выполнения лабораторной работы №3 были успешно изучены и реализованы основные матричные преобразования, используемые в 3D-графике:

- Создание и отрисовка базовых 3D объектов
- Масштабирование и перемещение объектов
- Вращение вокруг произвольных осей

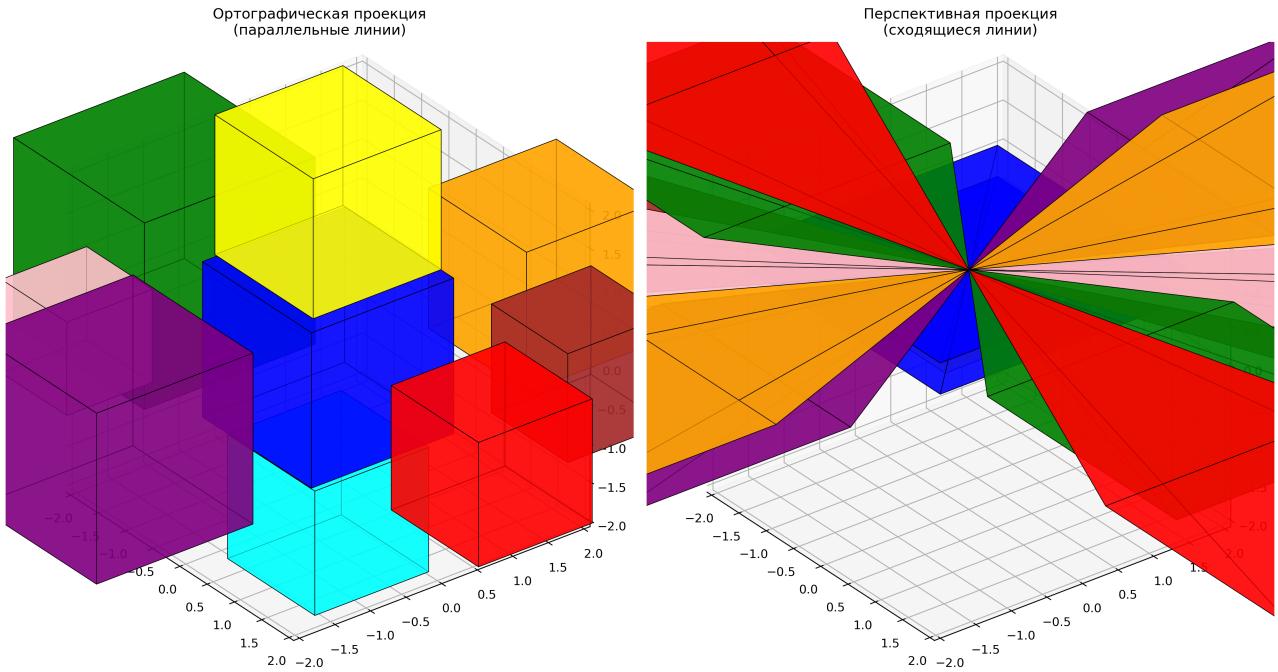


Рис. 13: Сравнение ортографической и перспективной проекций

- Реализация системы камеры
- Перспективная проекция

Все преобразования были реализованы с использованием матриц  $4 \times 4$  в однородных координатах, что обеспечивает единообразный подход к различным типам преобразований. Полученные результаты демонстрируют фундаментальные принципы компьютерной графики и могут быть использованы для дальнейшего изучения более сложных алгоритмов 3D-рендеринга.

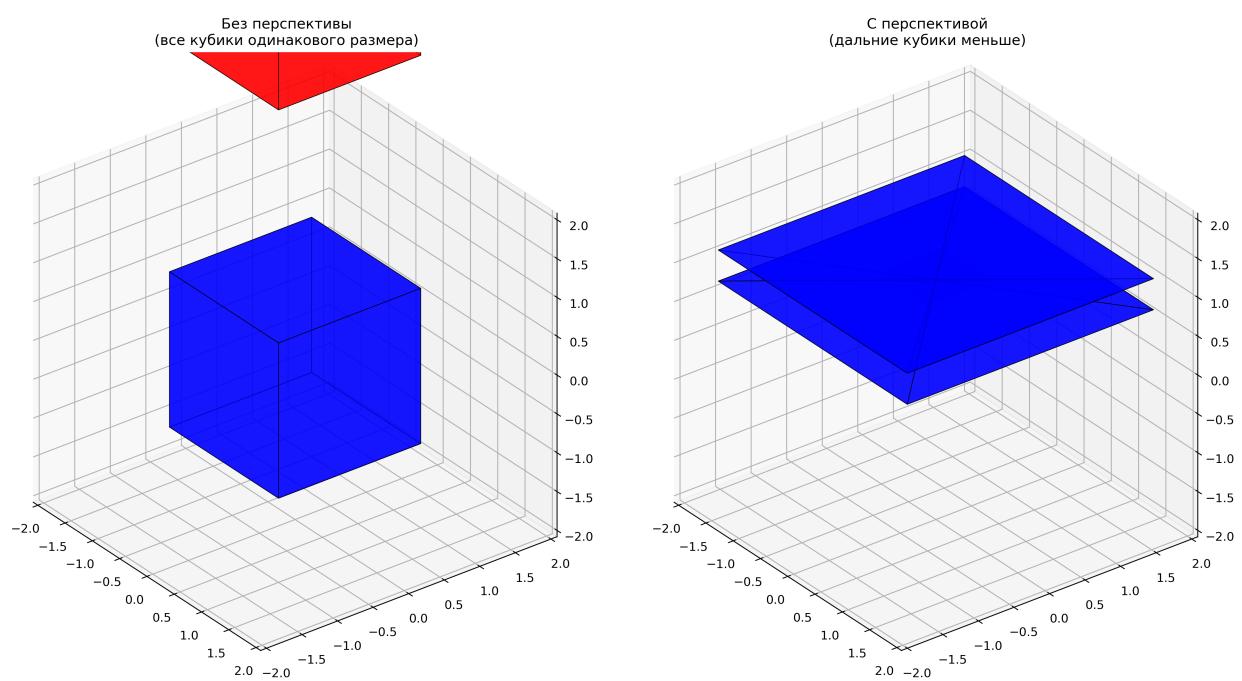


Рис. 14: Эффект расстояния в перспективной проекции