МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение
высшего образования «Санкт-Петербургский политехнический
университет Петра Великого»

Институт компьютерных наук и кибербезопасности

Высшая школа технологий искусственного интеллекта

Направление: 02.03.01 «Математика и компьютерные науки»

Лабораторная работа №2 по дисциплине «Алгоритмические основы компьютерной графики» по теме: «Алгоритм построения теней»

Студент,
группы 5130201/20102 _____ Тищенко А. А.

Преподаватель _____ Курочкин М. А.

«_____»_____ 2025г.

Санкт-Петербург, 2025

Содержание

Введение

Отображение теней является важной задачей компьютерной графики, так как тени позволяют повысить реалистичность сцены.

Тени делятся на собственные и падающие (или проекционные) (см. Рис. 1). Собственной тенью A называется неосвещённая часть поверхности. Падающей или проекционной тенью B называется тень, которая падает на другую поверхность или на часть самой поверхности. Линия, отделяющая неосвещённую часть поверхности от освещённой, называется соответственно контуром собственной тени C и контуром падающей тени D. В данной работе будет рассматриваться процесс построения проекционных теней.

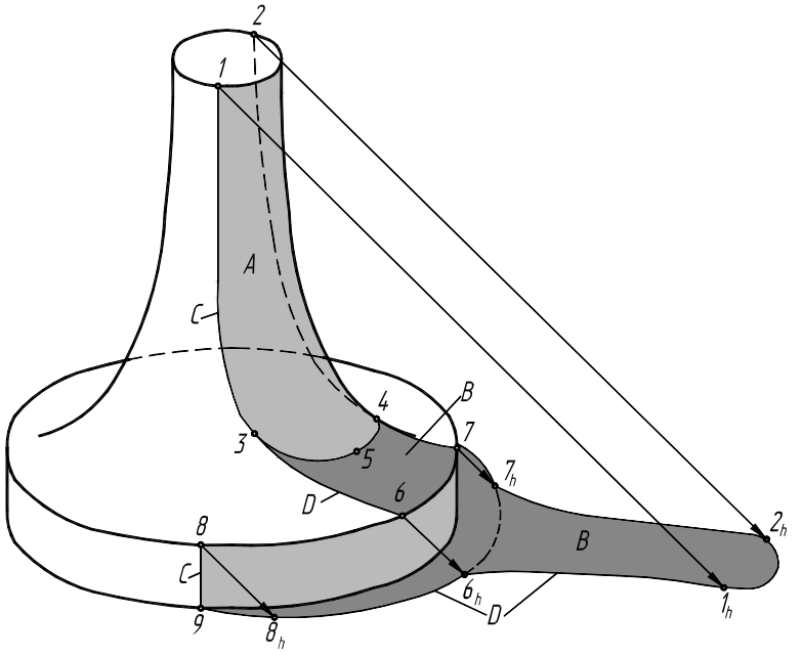

Рис. 1. Собственные и падающие (проекционные) тени.

Реальная тень состоит из двух частей: полутени и полной тени (см. Рис. 2). Полная тень — это центральная, темная, резко очерченная часть, а полутень — окружающая ее более светлая часть. Распределенные источники света конечного размера создают как тень, так и полутень: в полной тени свет вообще отсутствует, а полутень освещается частью распределенного источника. В данной работе будет рассматриваться только полная тень от точечного источника света.

Иногда отдельно рассматривают тени от полупрозрачных и окрашенных материалов (см. Рис. 3). Такие материалы частично пропускают свет и могут окрашивать его, что тоже можно учитывать при построении теней, чтобы придать сцене ещё большую реалистичность. Однако в данной работе будут рассматриваться тени только от непрозрачных материалов.

В компьютерной графике существует несколько распространённых подходов к вычислению теней, каждый из которых имеет свои области применения и ресурсоёмкость:

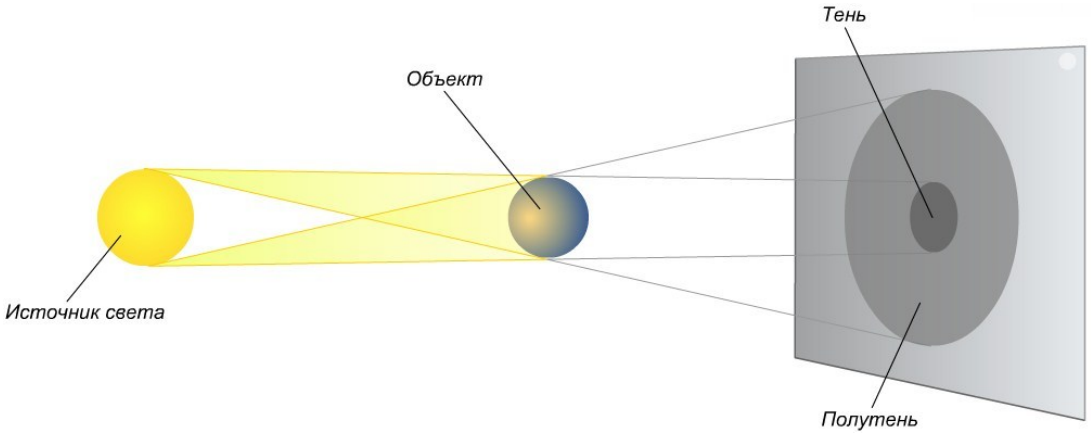

Рис. 2. Полутень и полная тень.

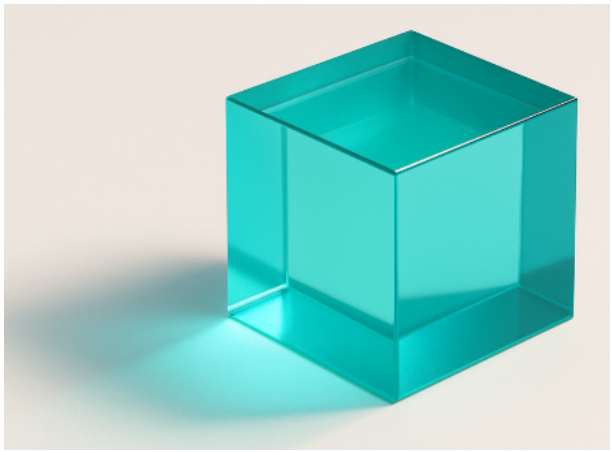

Рис. 3. Тень от полупрозрачного окрашенного материала.

- **Планарные проекционные тени (projective shadows)**: проецирование геометрии объекта на опорную плоскость вдоль направления света. Метод прост и быстр, хорошо подходит, когда нужно получить тень на одной плоскости;
- **Shadow Mapping**: построение карты глубины из пространства источника света и последующая проверка видимости. Широко используется в интерактивной графике, масштабируется на сложные сцены;
- **Shadow Volumes**: построение объёмов тени по силуэтам объектов и проверка попадания точки в объём. Обеспечивает чёткие границы, но сложнее в реализации;
- **Трассировка лучей**: физически корректная проверка видимости по лучам от точки к источнику. Даёт высокое качество, но обычно дороже по вычислениям.

В данной работе реализован первый подход — *планарные проекционные тени* при *направленном источнике света на бесконечности*. Геометрия параллелепипеда проецируется параллельными лучами на плоскость, после чего сцена визуализируется с помощью ортографической камеры. Такой выбор позволяет сфокусироваться на

линейной алгебре построения тени и наглядно проиллюстрировать ключевые шаги алгоритма при умеренной сложности реализации.

1 Постановка задачи

Дано: 3D-сцена:

- Параллелепипед P, заданный координатами вершин $\{v_i \in \mathbb{R}^3\}_{i=1}^8$.
- Плоскость H, заданная точкой $p_0 \in \mathbb{R}^3$ и нормалью $n_H \in \mathbb{R}^3$, $|n_H| = 1$.
- Источник света L, находящийся на бесконечности положительной части оси Z.
- Наблюдатель O, заданный позицией $o \in \mathbb{R}^3$ и ориентацией (широта ϕ и долгота θ).

Требуется: Построить полную проекционную тень, отбрасываемую параллелепипедом на плоскость.

2 Алгоритм построения теней

2.1 Шаги алгоритма

1. Определить лицевые грани параллелепипеда:

   - Для грани F_j, заданной вершинами $\{v_a, v_b, v_c\}$, нормаль вычисляется следующим образом:

   $$n_j = \frac{(v_b - v_a) \times (v_c - v_a)}{\|(v_b - v_a) \times (v_c - v_a)\|}$$

   - Грань F_j параллелепипеда считается лицевой, если скалярное произведение её нормали n_j и вектора направления света d_L отрицательно:

   $$n_j \cdot d_L < 0$$

   - Поскольку источник света направлен вдоль отрицательной оси Z ($d_L = (0, 0, -1)$), критерий упрощается до:

   $$n_3(F_j) > 0$$

2. Проекция лицевых граней на плоскость H.

   - Для каждой лицевой грани F_j выполняется параллельная проекция вершин $\{v_k\}_{k=1}^4$ на H вдоль d_L.
   - Для вершины $\{v_k\} = (x_k, y_k, z_k)$:

   $$v'_k = v_k + t\, d_L, \quad t = \frac{n_H \cdot (p_0 - v_k)}{n_H \cdot d_L}$$

   - Для каждой лицевой грани F_j, из проецированных вершин в порядке, соответствующем F_j, формируется теневой многоугольник, после чего он добавляется в структуру данных.

3. Построить вид сцены из заданной точки наблюдения.

   - Применим матрицу сцены к каждой точке v (при этом координаты точек переводятся в однородные координаты посредством добавлением скалярного множителя $w = 1$), преобразующую мировые координаты в систему координат камеры:

     – Матрица трансляции:

   $$T(o) = \begin{pmatrix} 1 & 0 & 0 & -o_x \\ 0 & 1 & 0 & -o_y \\ 0 & 0 & 1 & -o_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

– Матрица вращения вокруг оси Y на угол θ:

$$R_y(\theta) = \begin{pmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

– Матрица вращения вокруг оси X на угол ϕ:

$$R_x(\phi) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\phi) & -\sin(\phi) & 0 \\ 0 & \sin(\phi) & \cos(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

– Итоговая матрица:

$$M_{view} = R_x(\phi) * R_y(\theta) * T(o)$$

- Далее применяется ортографическая проекция:

$$M_{ortho} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Последним шагом является визуализация сцены.

3 Результаты

Для реализации метода планарной проекции тени при направленном источнике света был использован Python 3.13 и библиотеки numpy, matplotlib. На рис. 4–6 представлены результаты работы алгоритма построения теней для повёрнутого параллелепипеда и плоскости с трёх разных ракурсов со следующими параметрами:

Координаты вершин параллелепипеда (после поворота на 30°, 30°, 0°):

- Вершина 0: [5.21, 3.06, 7.03]
- Вершина 1: [9.54, 3.06, 4.53]
- Вершина 2: [10.04, 4.79, 5.40]
- Вершина 3: [5.71, 4.79, 7.90]
- Вершина 4: [6.08, 2.06, 8.53]
- Вершина 5: [10.41, 2.06, 6.03]
- Вершина 6: [10.91, 3.79, 6.90]
- Вершина 7: [6.58, 3.79, 9.40]

Параметры освещения:

- Широта источника света: 90°
- Долгота источника света: 0°
- Вектор направления луча света: (0.000, 0.000, -1.000)

Параметры плоскости проекции:

- Точка плоскости: [0, 0, 0]
- Нормаль плоскости: [0, 0, 1]

Источник света зафиксирован в одной позиции. Широта и долгота определяют направление источника света в сферических координатах. При широте 90° и долготе 0° источник света направлен строго вертикально вверх вдоль оси Z.

Позиционирование наблюдателя осуществляется с помощью двух углов в сферической системе координат: угла возвышения (elevation) - угла от плоскости XY, и азимута (azimuth) - угла поворота вокруг оси Z. Наблюдатель моделируется ортографической камерой.

На рис. 4 представлена визуализация сцены с видом сверху, где наблюдатель находится в той же позиции, что и источник света. В этом ракурсе тени не видно, так как она закрывается параллелепипедом.

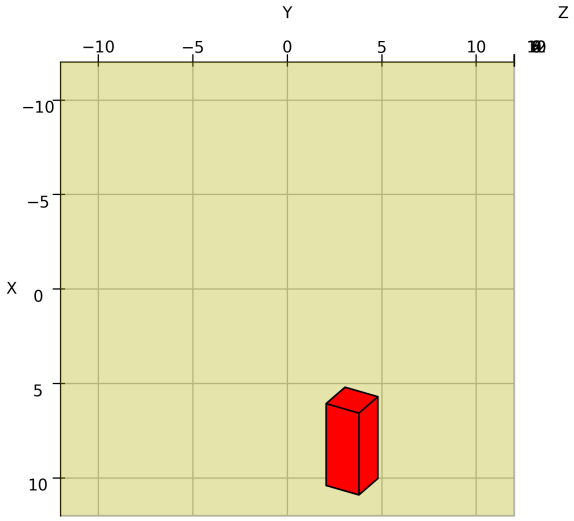

Рис. 4. Вид сверху (elevation=90°, azimuth=0°)

На рис. 5 показан вид сбоку, который позволяет увидеть тень от параллелепипеда на плоскости.

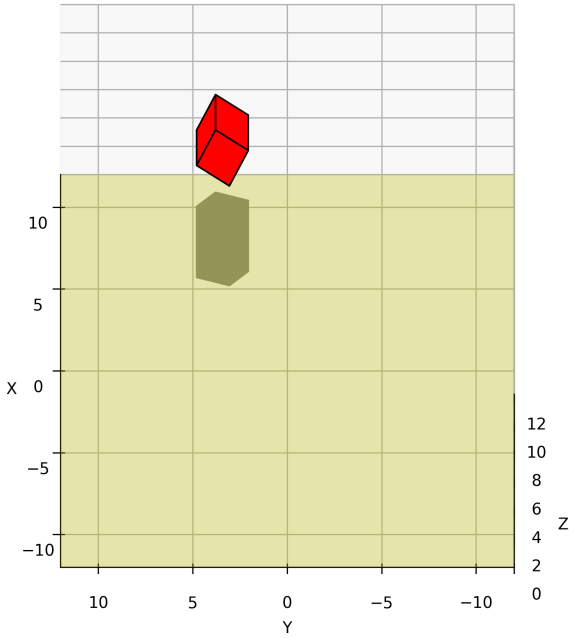

Рис. 5. Вид сбоку (elevation=60°, azimuth=180°)

На рис. 6 представлен вид под углом, демонстрирующий трёхмерную структуру

параллелепипеда и проецируемых теней, что позволяет оценить корректность работы алгоритма с различных точек обзора.

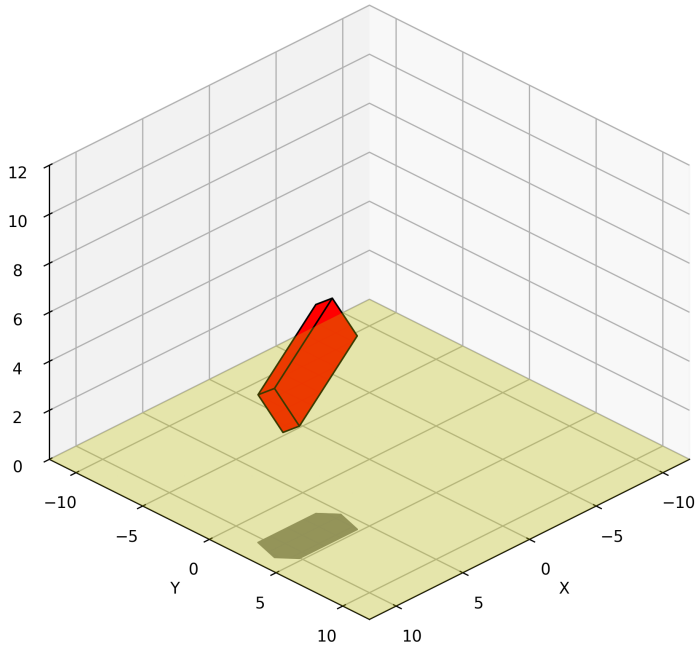

Рис. 6. Вид под углом (elevation=30°, azimuth=45°)

4 Сравнение с библиотечным алгоритмом

В качестве библиотечного эталона выбран **планарный алгоритм теней на основе матрицы проекции** (OpenGL-совместимая shadow-projection matrix на плоскость). Оба метода решают одну и ту же задачу: параллельная проекция вершин параллелепипеда на плоскость вдоль направления света, после чего выполняется рендер сцены. Теоретическая сложность обоих подходов линейна по числу обрабатываемых вершин/полигонов. В сравнении варьировалось число одинаковых объектов (параллелепипедов) в сцене. Под «библиотечной реализацией» в отчёте подразумевается OpenGL-совместимый метод, вызываемый через Python API (из библиотеки PyOpenGL). Все вычисления выполнялись на CPU.

Для корректности оценки учитывалось только *время геометрических вычислений проекции* и сборки теневых полигонов; накладные расходы на отрисовку графиками исключались. Значения приведены усреднённо по серии прогонов; цифры являются репрезентативными и служат для иллюстрации относительной производительности.

N	Наш алгоритм, мс	Библиотечный, мс	Разница, %
1	1.51	1.04	45.1
5	3.80	2.82	34.9
10	6.23	5.31	17.3
20	11.76	10.26	14.6
50	27.41	25.57	7.2
100	54.38	50.92	6.8

Таблица 1. Сравнение времени построения тени при различном числе объектов N

Выводы по сравнению. Оба подхода масштабируются линейно по числу объектов. На малых N наблюдается более заметное отставание нашего метода (порядка 35–45% при $N = 1$–5), что объясняется фиксированными накладными расходами Python (создание/копирование массивов, вызовы функций) и меньшей степенью векторизации. По мере роста сцены вычислительная часть доминирует, накладные амортизируются, и разница снижается до 6–7% (при $N \geq 50$).

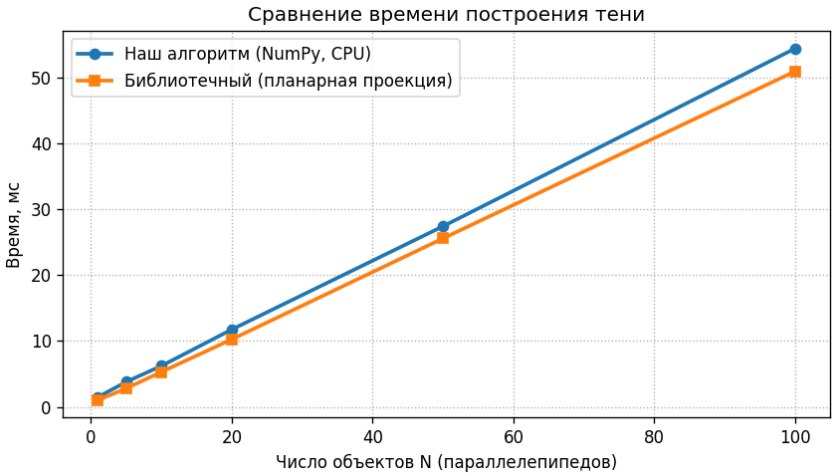

Рис. 7. График сравнения времени построения тени: зависимость от числа объектов N

Заключение

В данной работе рассмотрены основные подходы к построению теней и реализован метод планарной проекции тени параллелепипеда на плоскость при направленном источнике света. Реализация выполнена на языке Python с использованием библиотек `numpy` и `matplotlib`. Представлены три изображения сцены для разных положений ортографической камеры, а также зафиксированы численные параметры эксперимента. Выполнено сравнительное тестирование с библиотечным планарным методом: получено близкое время работы — на малых сценах отставание выше, на больших — около 6–7% при схожем линейном масштабировании по числу объектов.

Список литературы

[1] Мухин О. И., «Компьютерная графика». URL https://stratum.ac.ru/education/textbooks/kgrafic/additional/addit28.html (дата обращения 29.08.2025 г.)