СОДЕРЖАНИЕ

ВВЕДЕНИЕ

1 Аналитическая часть

1.1 Преломление лучей

Когда свет проходит из одной среды в другую, например из воздушной в водную и т.п., луч отклоняется, такое явление называется преломление. Его пример видно на рисунке 1.1, из-за отклонения лучей света наблюдателю кажется, что объект разрывается.

Рисунок 1.1 – Карандаш в стакане с водой, преломление света нарушает визуальную непрерывность.

Закон описывающий направления отклонения, изображенном на рисунке 1.2, называется закон Снелиуса [1]:

$$\frac{\sin\theta_1}{\sin\theta_2} = \frac{n_2}{n_1}, \tag{1.1}$$

где:

— θ_1, θ_2 — угол падения и преломленного угла луча,

— n_1, n_2 — коэффициенты плотности среды,

— v_1, v_2 — соответствующие скорости распространения света в средах,

— *normal* — нормаль к поверхности N,

— P — вектор, описывающий направление падающего луча в пространстве,

— Q — исходящего.

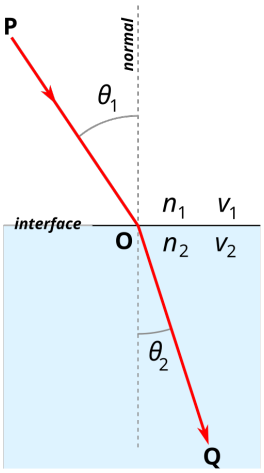

Рисунок 1.2 – Преломление одного луча.

Важное замечание – если луч падает их более оптически плотной среде ($n_1 > n_2$), под определенным углом (критическим) явление преломления вырождится в случай внутреннего отражения. И значение угла θ_1 находится при $\theta_2 = \pi/2$, как:

$$\sin\theta_1 = \frac{n_2}{n_1} \cdot \sin\theta_2 = \frac{n_2}{n_1}, \tag{1.2}$$

и угол исходящего луча θ_2 будет равен θ_1.

Направление исходящего луча также вычисляется следующим выражением:

$$Q = r \cdot P + \left[r \cdot c - \sqrt{1 - r^2 \cdot (1 - c^2)}\right] \cdot N, \tag{1.3}$$

где $r = n_1/n_2$ и $c = -\cos\theta_1 = -N \cdot P$.

Если луч падает под критическим углом, то

$$r^2 \cdot (1 - c^2) = r^2 \cdot \sin^2\theta_1 = \frac{n_1^2}{n_2^2} \cdot \frac{n_2^2}{n_1^2} = 1, \tag{1.4}$$

то есть подкоренное меньше нуля, если происходит внутреннее отражение.

Направление отраженного луча определяется, как:

$$Q = P - 2 \cdot (N \cdot L) \cdot N = P + 2 \cdot c \cdot N \tag{1.5}$$

1.2 Задача

Формализованная задача представлена в виде диаграммы на рисунке 1.3.

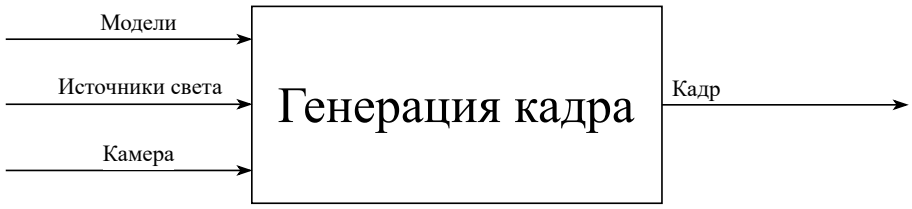

Рисунок 1.3 – IDEF0 диаграмма формализованной задачи.

1.3 Объекты

Сцена содержит в себе следующие объекты:

— модели — видимые объемные фигуры произвольной формы с параметрами:

1) цвет поверхности,
2) зеркальность,
3) коэффициент рассеивания,
4) прозрачность,
5) коэффициент преломления (оптическая плотность среды)
6) положение в пространстве (глобальные координаты, поворот, масштаб);

— источники света, с параметрами:

1) цвет,
2) интенсивность излучения,
3)
4) положение в пространстве (глобальные координаты, поворот, масштаб);

— точки обзора (камеры) — являются наблюдателями на сцене, всегда присутствует хотя бы одна камера, с параметрами:

1) разрешение (в том числе задает соотношение сторон изображения),

2) угол обзора,

3) положение в пространстве (глобальные координаты, поворот, масштаб).

1.3.1 Описание

Для представления объемных сущностей в компьютерной графике ниже представлены [2, с.с. 341–350]:

1) аналитическая — в виде уравнения поверхности, позволяет описать примитивную геометрию объекта, например прямой, плоскости, шара, тора и т.п., в пространстве;

2) воксельная — использует блоки для построения;

3) полигональная — в виде многогранника, позволяет описывать сложную геометрию.

1.3.2 Представление

Для решения задачи необходимо выбрать то представление объекта, которого будет достаточно для представления любого на сцене. Так как объекты могут быть любой формы, и описываются сущности имеющие объем, выбран полигональное представление.

1.4 Алгоритмы удаления невидимых поверхностей

Для корректного изображения сцены, нужно удалить невидимые поверхности, так как задача — генерация изображений с учетом преломления света, а значит предполагается, что некоторые объекты пропускают свет, таким образом геометрически объект, расположенный за прозрачной материей, не будет доступен, однако визуально 2-й объект будет виден. Кроме того, за счет преломления света на кадре могут быть

Для удаления невидимых поверхностей в большинстве случаев используют следующие алгоритмы:

— алгоритм Робертса [1, с. 303],

— алгоритм Художника [1, с. 387],

— алгоритм, использующий Z-буфер [1, с. 375],

— алгоритм обратной трассировки лучей [1, с. 432].

1.4.1 Алгоритм Робертса

Алгоритм основан на анализе нормалей граней и определении их ориентации относительно наблюдателя. Для этого он использует матрицы тела T размером $M \times 4$, где M – количество поверхностей объекта.

Основные этапы алгоритма:

1) удаления граней, экранируемых самим телом;

2) удаление частей, экранируемых другими объектами;

3) устранение отрезков, пересекающих плоскости.

Преимущества:

— не требует создания буферов размером пропорциональному размеру экрана,

— не использует растеризованные формы объектов,

— эффективен для выпуклых объектов без самопересечений.

Недостатки:

— предназначен для работы с выпуклыми многогранниками;

— при преломлении света нарушается базовое предположение о прямолинейном распространении лучей;

— неприменим к прозрачным и полупрозрачным материалам, где «невидимые» грани влияют на итоговое изображение через преломление.

Классический алгоритм Робертса критически ограничен при работе с преломляющими материалами, поскольку требует модификации для трассировки преломлённых лучей и учёта вклада всех поверхностей в формирование изображения, а не только "видимых"в геометрическом смысле. Поэтому этот алгоритм не подходит для решения этой задачи.

1.4.2 Алгоритм Художника

Основные этапы алгоритма:

1) вычисление глубины каждого полигона,

2) сортировка этих полигонов по глубине,

3) проверка перекрытий и разрешение конфликтов,

4) последовательная отрисовка полигонов от дальних к ближним по списку.

Преимущества:

— возможность реализации прозрачности,

— не требует дополнительного буфера.

Недостатки:

— требует предварительной сортировки,

— сложности обработки перекрывающихся многоугольников,

— при изменении положения наблюдателя требует повторной сортировки.

Алгоритм не подходит для визуализации отраженных и преломленных объектов, поэтому он не подходит для решения поставленной задачи.

1.4.3 Алгоритм, использующий Z-буфер

Алгоритм, работающий в пространстве изображения, использует буфер глубин каждого пикселя буфера кадра, в результате чего на изображении будут только видимые поверхности.

Основные этапы алгоритма:

1) заполнить буфер кадра b_{frame} фоновым значением,

2) заполнить Z-буфер b_z минимальным значением,

3) растеризировать все объекты

4) для каждого пикселя $point$ вычислить его глубину $z(point)$, если $b_z(point) < z(point)$, то присвоить $b_z(point)$ значение $z(point)$ и $b_{frame}(point)$ значение цвета поверхности.

Преимущества:

— подходит для обработки разных представлений тел,

— не требует предварительной сортировки списка объектов сцены,

— обладает линейной сложностью,

— не случается конфликтных ситуаций.

Недостатки:

— ошибка точности, связанная с перспективным преобразованием,

— предполагает прямолинейное распространение света,

— не позволяет реализовать прозрачность,

— слишком прост в реализации, чтобы визуализировать сложные сцены с большим количеством зависимостей.

Алгоритм использующий Z-буфер, не позволяет решить поставленную задачу, а значит — не подходит.

1.4.4 Алгоритм обратной трассировки лучей

Алгоритм «грубой силы», наиболее приближен к физическим моделям распространения света, потому что просчитывается каждый луч. Изначально рассматривались 2 вида трассировки: прямая – от источника света до наблюдателя и обратная. Прямое трассирование луча является менее эффективным способом решения задачи, так как большая часть выпущенных из источника лучей не

дойдут до наблюдателя. Второй вид, наоборот, позволяет вычислять только луч, влияющие на результат.

В настоящее время разработано множество модификаций, преследующие различные цели. За счет того, что просчитывается каждый луч, доступна реализация различных оптических явлений, в том числе преломления и прозрачности.

Основные этапы алгоритма:

1) для каждого пикселя экрана сформировать луч,

2) найти ближайшее пересечение луча с моделью на сцене.

Преимущества:

— подходит для обработки разных представлений тел,

— наиболее приближен к физическим моделям распространения света,

— позволяет реализовать оптические явления.

Недостатки:

— большой объем работы,

— сложность вычислений.

За счет того, что алгоритм предоставляет возможность реализации поставленной задачи, за счет включения в алгоритм логики вычисления преломленных и отраженных лучей.

1.4.5 Выбранный алгоритм

Так как все описанные алгоритмы, кроме обратной трассировки,не учитывают искажений преломления света, выбран именно этот алгоритм.

1.5 Освещение

Модели освещения рассматривают 2 вида: локальная и глобальная.

1.5.1 Модели освещения

Локальная модель освещения учитывает модели индивидуально, что не подходит для реализации задачи, так как именно другие объекты будут видны в результате преломления. Что учитывает глобальная [1, с. 464, с. 502, с. 548].

Уравнение интенсивности:

$$I = k_\alpha \cdot I_\alpha + k_d \cdot \sum_{i=1}^{N}(I_{L_i} \cdot \vec{n} \cdot \vec{L_i}) + k_s \cdot \sum_{i=1}^{N}(I_{L_i} \cdot (\vec{S} \cdot \vec{L_i})^n) + k_s \cdot I_s + k_t \cdot I_t, \quad (1.6)$$

где коэффициенты

— k_α — фоновый,

— k_d — диффузный,

— k_s — зеркальный,

— k_t — преломленного луча,

— n — аппроксимирующий распределение лучей от отражения,

и вектора

— $\vec{L}$ — направления до точечного источника света,

— $\vec{S}$ — направление до наблюдателя,

— $\vec{n}$ — нормали к поверхности.

1.5.2 Тени

Для определения интенсивности затененных областей удобно использовать алгоритм трассировки лучей, для это испускаются зондирующие лучи до источников света, если пересечение установлено, то область находится в тени [1, с. 517]. Однако, когда рассматриваются прозрачные модели, точка может не быть затененной, поэтому для каждого зонда и источника коэффициент затенения от одного объекта составляет [2, с. 368]:

$$k_{shadow,i} = k_{t,i} \quad (1.7)$$

Если луч пересекает несколько объектов, то коэффициент для N_O полных пересечений моделей на отрезке до источника света будет:

$$k_{shadow} = \prod_{i=1}^{N_O} k_{t,i} \tag{1.8}$$

1.5.3 Полная интенсивность

Итоговая интенсивности в точке вычисляется по формуле 1.9.

$$I = k_\alpha \cdot I_\alpha + \sum_{i=1}^{N_L} \left[\prod_{j=1}^{N_O} k_{t,j} \cdot (k_d \cdot \vec{n} \cdot \vec{L_i} + k_s \cdot (\vec{S} \cdot \vec{L_i})^n) \right] \cdot I_{L_i} + k_s \cdot I_s + k_t \cdot I_t \tag{1.9}$$

1.6 Выводы из аналитической части

В аналитической части достигнуты следующие цели:

— формализована задача,

— рассмотрены представления объектов,

— проанализированы доступные алгоритмы удаления невидимых граней,

— описаны модели освещения.

В результате выбраны

1) полигональное представление объектов, которое позволит изобразить модели любой формы;

2) алгоритм обратной трассировки лучей, для поддержки преломления лучей света;

3) глобальная модель освещения, так как она учитывает положения и освещенность других объектов, что и требуется для преломления света.

2 Конструкторская часть

2.1 Требования к программе

Программа должна иметь графический пользовательский интерфейс, который предоставляет возможность

— загрузку сцены из специального файла,

—

2.2 Используемые структуры данных

2.2.1 Объект

2.2.2 Камера

2.2.3 Источник света

2.3 Схемы алгоритмов

2.4 Выводы из конструкторской части

3 Технологическая часть

4 Исследовательская часть

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. David F. Rogers. Procedural elements for computer graphics – 1998. – 2-е издание. – 695 с.

2. John F. Hughes, Andries Van Dam, Morgan MvGuire. Computer Graphics Principles and Practice – 2014. – 3-е издание. - 1260 с.

3. Tomas Akenine-Möler, Eric Haines, Naty Hofman. Real-time rendering – 2018. – 4-е издание. – 1199 с.

4. Mat Pharr, Wenzel Jakob, Greg Humphreys. Physically based rendering – From theory to implementation – 2017. – 3-е издание. – 1270 с.

5. Steve Klabnik, Carol Nichols. The Rust Programming Language // Руководство. Языки программирования. – 2022. – 2-е издание. – 560 с.

6. Документация библиотеки cpu_time [эл. ресурс]. – URL: https://docs.rs/cpu-time/latest/cpu_time/ (дата обращения: 5 октября 2025 г.).

7. Документация текстового редактора «NVim» [эл. ресурс]. – URL: https://neovim.io/doc/ (дата обращения: 5 октября 2025 г.).