

Содержание

Введение	4
1 Аналитическая часть	5
1.1 Анализ необходимых качеств реалистического изображения	5
1.2 Формализация объектов синтезируемой сцены	5
1.3 Анализ алгоритмов удаления невидимых линий и ребер поверхностей	6
1.3.1 Алгоритм, использующий Z-буфер	6
1.3.2 Алгоритм обратной трассировки лучей	7
1.3.3 Алгоритм обратной трассировки путей	7
1.3.4 Алгоритм Робертса	7
1.3.5 Алгоритм художника	8
1.4 Реализация теней	8
1.5 Построение отражений и преломления	9
1.5.1 Модель освещения Ламберта	9
1.5.2 Модель освещения Фонга	10
1.5.3 Модель освещения Френеля	10
1.6 Реализация глубины резкости	11
1.6.1 Трассировка лучей по поверхности линзы	11
1.6.2 Использование накопительного буфера	12
1.6.3 Слоистая глубина резкости	12
1.7 Реализация подповерхностного рассеивания	12
2 Конструкторская часть	15
2.1 Требования к программному обеспечению	15
2.2 Расчет лучей на поверхности тела	15
2.3 Разработка алгоритмов	18
Список использованных источников	21

Введение

Компьютерная графика является неотъемлемой частью современного мира в особенности она востребована в сфере кинопроизводства и игровой промышленности. Именно поэтому синтез реалистического изображения является одной из важнейших задач.

Подобные условия требуют создания средств и методик, учитывающих такие оптические явления как преломление, отражение и рассеивание света, зависящие от свойств визуализируемых объектов. Но рост точности и детальности разрабатываемых алгоритмов зачастую приводит к более высоким затратам по времени и памяти.

Целью данной работы является разработка программы, позволяющей создать реалистическое изображение с учетом подповерхностного рассеивания, на основе трехмерной сцены наполненной объектами (сфера, куб, конус, цилиндр, полигональная модель) и источниками света, положение, количество и свойства которых задается пользователем.

Задачи.

1. Выявить неотъемлемые качества реалистического изображения.
2. Провести анализ существующих алгоритмов.
3. Выбрать оптимальные пути для решения основной задачи.
4. Реализовать выбранные алгоритмы.
5. Создать программный продукт для решения задачи.

1. Аналитическая часть

В этом разделе будут рассмотрены модели представления трехмерных тел, формализованы объекты, наполняющие сцену, и требования, предъявляемые к работе программы и конечному результату. Будут рассмотрены алгоритмы построения трехмерного изображения, методы закраски, модели освещения.

1.1. Анализ необходимых качеств реалистического изображения

Большинство алгоритмов, изученных в курсе компьютерной графики нацелены на корректное отображение геометрии объектов и их положения в сцене, однако для построения реалистического изображения этих аспектов недостаточно.

Первым аспектом, наполняющим сцену объемом можно считать тень. Однако большинство моделей подразумевает источник света материальной точкой, из-за чего все тени получаются четкими и однородными, что в общем случае неверно.

Подобным образом, для построения изображения стоит учитывать такую важную составляющую оптических систем, как линза. Изображения предметов на фотографии или у нас перед глазами получаются четкими только в случае, если они находятся в фокальной плоскости, иначе изображение размывается, что также необходимо учитывать.

Не стоит забывать, что объекты в сцене состоят из различных материалов, что вносит на рассмотрение блестящие и прозрачные предметы, с присущими явлениями отражения, преломления и рассеивания.

1.2. Формализация объектов синтезируемой сцены

Сцена состоит из следующего набора объектов.

1. Геометрический объект – непосредственное представление объектов реального мира внутри сцены. В число рассматриваемых тел входят параметрические модели, такие как куб, сфера, конус, цилиндр, а также тела произвольной формы. Данная модель предназначена исключительно для хранения геометрического представления и иерархии составного тела.

Для описания каждого объекта используются:

- куб – координаты двух диагональных вершин;
- сфера – координаты центра и радиус;

- конус – координаты центра основания, радиус основания, высота;
 - цилиндр – координаты центра нижнего основания, радиус, высота;
 - тело произвольной формы – набор полигонов с указанием нормали к поверхности, определяющей наружную часть объекта. В качестве основной единицы такой модели был выбран треугольник, так как с его помощью можно представить полигон любой формы.
2. Источник света – геометрическое тело, обладающее свойством свечения. Для источника света присущи такие вырожденные случаи геометрического тела, как точка (точечный источник света, направленный свет).
 3. Карта окружения – куб, опоясывающий все объекты сцены и бесконечно удаленный от них. Служит для определения фонового изображения и создания дополнительных отражений.
 4. Камера – геометрическое тело, определяющее видимую часть сцены. Положение задается координатой центра, направление взгляда совпадает с осью Z тела.

1.3. Анализ алгоритмов удаления невидимых линий и ребер поверхностей

Прежде всего предстоит выбрать основной алгоритм визуализации изображаемой сцены. При выборе необходимо обращать внимание на следующие аспекты: быстродействие, требовательность к памяти, возможность реализации эффектов, описанных в пункте 1.1.

1.3.1 Алгоритм, использующий Z -буфер

Смысл алгоритма заключается в использовании двух буферов: буфера кадра и Z -буфера, хранящих атрибуты и информацию о координате Z каждого пикселя соответственно.

Z -буфер инициализируется минимальным значением координаты, а буфер кадра – информацией о пикселе, описывающем фон. Глубина каждого нового пикселя при подсчете сравнивается со значением, которое уже есть в Z -буфере. В случае, если новый пиксель расположен ближе к наблюдателю, то информация о нем заносится в буфер кадра и происходит редактирование Z -буфера.

Данный алгоритм достаточно прост в реализации и, в силу отсутствия сортировок, не требует больших затрат по времени и памяти (для современных компьютеров), однако требуются дополнительные техники для учета отражений, прозрачности и теней.

1.3.2 Алгоритм обратной трассировки лучей

Алгоритм берет свое начало из законов оптики: наблюдатель видит объект, только если до него доходят лучи испускаемого света, прошедшие некоторый путь от источника.

Однако, вопреки физической модели, трассировка лучей происходит в обратном направлении – от наблюдателя к источнику. В противном случае приходится рассматривать множество лучей, которые никогда не попадут в наблюдаемую область.

Предполагая, что наблюдатель располагается непосредственно за экраном, в процессе работы алгоритма для каждого пикселя выпускается луч. Если луч попадает в объект сцены, то из найденной точки пересечения в направлении источников света посылаются так называемые «теневые» лучи, благодаря которым определяется итоговая интенсивность точки, а в следствии и пикселя. Иначе пиксель высвечивается цветом фона.

Алгоритм построен на физически верной модели, что дает широкий простор для реализации различных визуальных явлений, но такая гибкость и обобщенность требует большого количества вычислений (возможность усовершенствования за счет параллельных вычислений, так как каждый пиксель в данном методе не зависит от окружающих).

1.3.3 Алгоритм обратной трассировки путей

Описанный выше алгоритм учитывает только прямое освещение, что ограничивает возможности воссоздания оптических эффектов. Как следует из названия, алгоритм производит построение пути, пройденного светом от источника до наблюдателя, путем рекурсивной работы алгоритма трассировки лучей, для очередной найденной точки пересечения.

Несмотря на кажущуюся простоту алгоритма, он является достаточно затратным в плане вычислений, так как для получения корректного результата в конкретной точке необходим расчет нескольких путей в различных направлениях. Помимо этого, возникают проблемы, связанные с множественным отражением и глубиной рекурсии.

1.3.4 Алгоритм Робертса

Данный алгоритм решает задачу удаления невидимых ребер и граней только с выпуклыми телами, работает только в объектном пространстве.

У следующие этапы выполнения:

- удаление ребер, экранируемых самим телом;
- удаление невидимых ребер, экранируемых другими телами сцены;
- удаление невидимых ребер, образованных в результате протыкания.

Алгоритм обладает высокой точностью, в силу того, что все вычисления происходят в объектном пространстве, однако тела должны быть выпуклыми, что не подходит для нашей задачи.

1.3.5 Алгоритм художника

Данный алгоритм работает аналогично тому, как художник рисует картину - то есть сначала рисуются дальние объекты, а затем более близкие. Наиболее распространенная реализация алгоритма - сортировка по глубине, которая заключается в том, что произвольное множество граней сортируется по ближнему расстоянию от наблюдателя, а затем отсортированные грани выводятся на экран в порядке от самой дальней до самой ближней.

Данный алгоритм требует меньше памяти в сравнении с остальными, однако так же, как и алгоритм Z-буфера требует дополнительных методов отображения визуальных эффектов.

Вывод

Так как цель работы – создание реалистического изображения, то лучшим выбором станет алгоритм трассировки путей. Данный алгоритм предоставляет удобную базу для реализации оптических эффектов, а затраты по времени не окажут особого значения, так как объекты в сцене – статические, поэтому можно сконцентрироваться на качестве получаемого изображения.

1.4. Реализация теней

Выбранный в предыдущем пункте алгоритм трассировки лучей, позволяет легко реализовать отрисовку тени.

Корректное построение достигается за счет того, что после нахождения ближайшей точки пересечения очередного луча, из нее посылаются «теновые» лучи к каждому источнику света в сцене. Если теневой луч не достигает источника света (имеется более близкое пересечение с каким-либо объектом сцены), то точка не освещена, иначе – в точке рассчитывается интенсивность, согласно модели освещения.

Помимо этого, также решается проблема построения мягких теней, так как имеется возможность испускания множества теневых лучей к одному объекту, что позволит определить видимую часть и итоговую интенсивность.

1.5. Построение отражений и преломления

Нахождение отражений и преломления также является логическим продолжением алгоритма трассировки лучей. Используя нормаль к поверхности и направление отраженного луча, нетрудно найти падающий и преломленный лучи, для которого аналогично применяется трассировка. Пример расположения лучей на границе объекта представлен на рисунке 1.1.

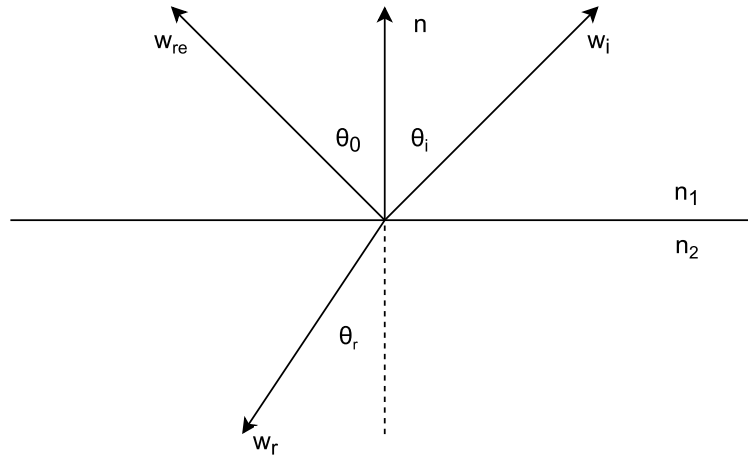


Рисунок 1.1 – Пример расположения лучей на границе объекта: w_i - направление падающего луча, w_{re} - направление отраженного луча, w_r - направление преломленного луча

Интенсивность света, распространяющегося в направлении взгляда w_0 из точки пространства p , может быть описана при помощи следующей формулы

$$L_0(p, w_0) = \int_{S^2} f(p, w_0, w_i) L_i(p, w_i) \cos(\theta_i) dw_i, \quad (1.1)$$

где w_i - направление падающего света в точке p ; f - функция рассеяния; L_i - интенсивность света, падающего по направлению w_i ; θ_i - угол между нормалью к поверхности и вектором направления падающего света. Значение интеграла вычисляется по поверхности единичной сферы. Такой метод описания дает возможность реализовать любую модель освещения и легко произвести подмену при необходимости, за счет введения функции рассеяния, показывающую какая часть падающего света отражается в указанном направлении.

$$f(p, w_0, w_i) \in [0, 1] \quad (1.2)$$

1.5.1 Модель освещения Ламберта

Данная модель предназначена для представления идеального диффузного рассеивания, то есть подразумевается, что свет при попадании на поверхность равномерно распространяется во всех направлениях

$$f_d(p, w_0, w_i) = k_d, \quad (1.3)$$

где k_d - коэффициент диффузного отражения.

1.5.2 Модель освещения Фонга

Данная модель базируется на модели Ламберта и добавляет зеркальную составляющую, то есть блик на поверхности объекта. Для достижения этого эффекта к формуле (1.3) добавляется следующая функция зеркального рассеивания

$$f_s(p, w_0, w_i) = k_s \cos^\alpha(\theta_{re}) = k_s \left(\frac{w_0 w_{re}}{|w_0| |w_{re}|} \right)^\alpha, \quad (1.4)$$

где θ_{re} – угол между отраженным лучом и лучом взгляда, k_s - коэффициент зеркального отражения, α - коэффициент блеска. Пример расположения лучей представлен на рисунке 1.2.

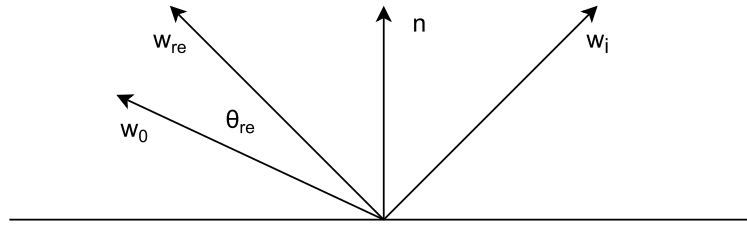


Рисунок 1.2 – Расположение лучей при расчете модели Фонга: w_0 - направление взгляда, w_{re} - направление отраженного луча, w_i - направление падающего луча

1.5.3 Модель освещения Френеля

В отличие от представленных эмпирических моделей, данная – физически корректна и опирается на решение уравнений Максвелла

$$F(p, w_0, w_i) = \frac{1}{2}(r_\perp^2 + r_\parallel^2), \quad (1.5)$$

где r_\parallel - коэффициент отражения для света с параллельной поляризацией, r_\perp - коэффициент отражения для света с перпендикулярной поляризацией. Реализация метода разделяет материалы на два типа: проводники и диэлектрики (согласно их физическим свойствам).

1. Отражение для диэлектрика

$$r_\parallel = \frac{n_2 \cos(\theta_i) - n_1 \cos(\theta_r)}{n_2 \cos(\theta_i) + n_1 \cos(\theta_r)} \quad (1.6)$$

$$r_\perp = \frac{n_1 \cos(\theta_i) - n_2 \cos(\theta_r)}{n_1 \cos(\theta_i) + n_2 \cos(\theta_r)}, \quad (1.7)$$

где θ_r - угол между нормалью и преломленным лучом, который может быть вычислен по закону Снеллиуса.

2. Отражение для проводника

В отличие от диэлектриков коэффициент преломления для металлов - комплексное число $\bar{n} = n + ik$, где n - показатель преломления, k - коэффициент экстинкции.

$$r_{\perp} = \frac{a^2 + b^2 - 2a \cos(\theta_i) + \cos^2(\theta_i)}{a^2 + b^2 + 2a \cos(\theta_i) + \cos^2(\theta_i)} \quad (1.8)$$

$$r_{\parallel} = r_{\perp} \frac{\cos^2(\theta_i)(a^2 + b^2) - 2a \cos(\theta_i) \sin^2(\theta_i) + \sin^4(\theta_i)}{\cos^2(\theta_i)(a^2 + b^2) + 2a \cos(\theta_i) \sin^2(\theta_i) + \sin^4(\theta_i)}, \quad (1.9)$$

$$\text{где } a^2 = \frac{\sqrt{(n^2 - k^2 - \sin^2(\theta_i))^2 + 4n^2 k^2} + (n^2 - k^2 - \sin^2(\theta_i))}{2}, b^2 = \frac{\sqrt{(n^2 - k^2 - \sin^2(\theta_i))^2 + 4n^2 k^2} - (n^2 - k^2 - \sin^2(\theta_i))}{2},$$

$$\text{тогда } a^2 + b^2 = \sqrt{(n^2 - k^2 - \sin^2(\theta_i))^2 + 4n^2 k^2}, \text{ при этом } \bar{n} = \frac{\bar{n}_2}{\bar{n}_1} = \frac{n_2 + ik_2}{n_1 + ik_1} = \frac{n_1 n_2 + k_1 k_2}{n_1^2 + k_1^2} + i \frac{k_2 n_1 - n_2 k_1}{n_1^2 + k_1^2}, \text{ то есть } n = \frac{n_1 n_2 + k_1 k_2}{n_1^2 + k_1^2}, \text{ а } k = \frac{k_2 n_1 - n_2 k_1}{n_1^2 + k_1^2}.$$

Однако в таком виде не получится добиться зеркального блика, для этого введем

$$\text{функцию } \delta_{w_0} = \begin{cases} 1, & \text{если } w = w_0 \\ 0, & \text{иначе.} \end{cases} \quad \text{Тогда}$$

$$\begin{aligned} L_0(p, w_0) &= \int_{S^2} \delta_{w_0}(w_i) F(p, w_0, w_i) L_i(p, w_i) \cos(\theta_i) dw_i = \\ &= F(p, w_0, w_{re}) L_{re}(p, w_{re}) \cos(\theta_{re}), \end{aligned}$$

где w_{re} - зеркально отраженный луч для w_0 . Но желаемый результат

$$L_0 = F(p, w_0, w_{re}) L_{re}(p, w_{re}),$$

поэтому необходимо нормировать функцию $\delta_{w_{re}}$

$$f_s(p, w_0, w_i) = \frac{\delta_{w_{re}}}{\cos(\theta_{re})} F(p, w_0, w_i) \quad (1.10)$$

1.6. Реализация глубины резкости

Желаемый результат по достижению глубины резкости может быть достигнут следующими способами.

1.6.1 Трассировка лучей по поверхности линзы

Трассировка лучей также позволяет достаточно очевидно реализовать глубину резкости, засчет симуляции множества лучей проходящих через оптическую систему (линзу).

К примеру можно реализовать тонкую линзу, пропуская луч из точки картинной плоскости в произвольные точки на поверхности диска, представляющего собой линзу. Если объект находится в фокусе, то все лучи попадут в одну точку, в результате чего

изображение будет четким, иначе – множество различных значений излучения создадут размытое изображение. Пример прохождения лучей через линзу представлен на рисунке 1.3.

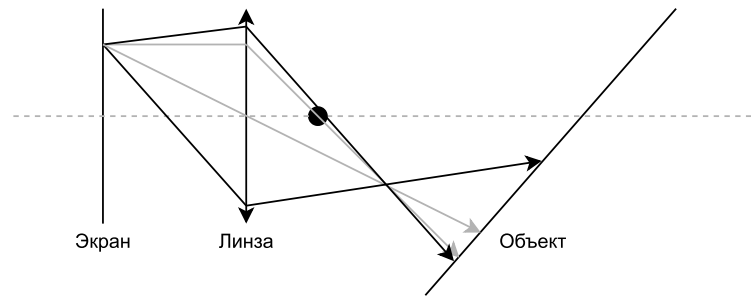


Рисунок 1.3 – Лучи, попадающие в одну точку, после прохождения через тонкую линзу

1.6.2 Использование накопительного буфера

Метод подразумевает, что для получения необходимого эффекта, нужно будет получить множество буферов кадра, для каждого из которых положение камеры будет изменяться согласно параметрам камеры. Результирующее изображение получается суммированием и усреднением полученных значений в накопительном буфере на каждой новой итерации алгоритма.

1.6.3 Слоистая глубина резкости

Алгоритм реализует поставленную задачу, путем разделения сцены на «слои», согласно глубине изображения в точке сцены. Далее к полученным слоям применяется алгоритм фильтрации, для достижения необходимого размытия текущего слоя. Это метод менее затратен в плане вычислений, так как для сцены требуется вычислить только один буфер кадра, разбитый на несколько слоев.

Вывод

В программе будет использоваться метод распространения лучей по поверхности линзы, так как он позволит реализовать практически любую оптическую систему и произвести подмену в случае необходимости.

1.7. Реализация подповерхностного рассеивания

Подповерхностное рассеивание описывает механизм распространения света, при котором свет, проникая внутрь полупрозрачного тела через его поверхность, рассеивается внутри самого тела, многократно отражаясь от частиц тела в случайном направлении и на

нерегулярные углы. В итоге свет выходит из объекта в выходной точке, отличной от точки вхождения в объект.

Нетрудно заметить, что необходимый результат можно достичь, произведя дополнительное интегрирование соотношения (1.1) по поверхности тела A , что потребует введения дополнительной *функции поверхностного рассеивания* $S(p_0, w_0, p_i, w_i)$

$$L_0(p_0, w_0) = \int_A \int_{H^2} S(p_0, w_0, p_i, w_i) L_i(p_i, w_i) \cos(\theta_i) dw_i dA. \quad (1.11)$$

Однако задание функции рассеивания даже для плоского случая достаточно проблематично, даже без учета составных тел. Поэтому на практике применяется другое соотношение для S , называемое *разделяемой функцией поверхностного рассеивания*

$$S(p_0, w_0, p_i, w_i) = (1 - F_r(\cos \theta_0)) S_p(p_0, p_i) S_w(w_i), \quad (1.12)$$

где $1 - F_r$ - формула отражения Френеля (1.5), показывающая часть света, доходящую до наблюдателя после выхода за границы объекта; $S_p(p_0, p_i)$ - функция, описывающая часть света, способную пройти расстояние от точки входа до точки выхода внутри тела; $S_w(w_i) = \frac{1 - F_r(\cos \theta_i)}{c\pi}$ - аналог первой части выражения для света, попадающего в границы объекта. Нормализующий коэффициент c служит для выполнения условия $\int_{H^2} S_w(w_i) \cos \theta dw = 1$ (интегрирование по половине сферы, отсекаемой поверхностью).

Для демонстрации упрощения подставим (1.12) в (1.11)

$$\begin{aligned} L_0(p_0, w_0) &= \int_A \int_{H^2} S(p_0, w_0, p_i, w_i) L_i(p_i, w_i) \cos(\theta_i) dw_i dA = \\ &= \int_A \int_{H^2} (1 - F_r(\cos \theta_0)) S_p(p_0, p_i) S_w(w_i) L_i(p_i, w_i) \cos(\theta_i) dw_i dA = \\ &= (1 - F_r(\cos \theta_0)) \int_A S_p(p_0, p_i) \int_{H^2} S_w(w_i) L_i(p_i, w_i) \cos(\theta_i) dw_i dA \end{aligned} \quad (1.13)$$

Несмотря на то, что мы разделили интеграл на части, внутренняя часть которого вполне укладывается в имеющееся соотношение для отражения (1.1), остается проблема с поддержкой геометрии произвольных тел, поэтому введем следующее приближение и получим окончательную формулу

$$S_p(p_0, p_i) \approx S_r(|\overline{p_0 p_i}|) \quad (1.14)$$

$$\begin{aligned} L_0(p_0, w_0) &= (1 - F_r(\cos \theta_0)) \int_A S_p(p_0, p_i) \int_{H^2} S_w(w_i) L_i(p_i, w_i) \cos(\theta_i) dw_i dA \approx \\ &\approx (1 - F_r(\cos \theta_0)) \int_A S_r(|\overline{p_0 p_i}|) \int_{H^2} S_w(w_i) L_i(p_i, w_i) \cos(\theta_i) dw_i dA \end{aligned} \quad (1.15)$$

Вывод

В данном разделе были описаны модели представления моделей в сцене и свойства, необходимые для физически корректной визуализации объектов с учетом подповерхностного рассеивания. В качестве основного алгоритма удаления невидимых линий был выбран алгоритм трассировки лучей, на основании которого будет строиться реализация отражений, теней, глубины резкости и подповерхностного рассеивания.

2. Конструкторская часть

В данном разделе будут рассмотрены требования к программному обеспечению, некоторые расчетные соотношения и схемы разработанных алгоритмов.

2.1. Требования к программному обеспечению

Программа должна предоставлять следующий функционал:

- возможность интерактивного изменения положения объектов и источников света в сцене;
- возможность интерактивного добавления/удаления объектов в сцене;
- возможность интерактивного изменения свойств объектов (коэффициент восприятия света, коэффициент преломления, тип материала объекта, интенсивность и цвет излучения);
- возможность интерактивного масштабирования и поворота объектов сцены.

К программе предоставляются следующие требования:

- программа должна корректно реагировать на любые действия пользователя;
- программа должна предоставлять приемлемое время отклика (не более 1 сек) при работе в интерактивном режиме.

2.2. Расчет лучей на поверхности тела

Выражение для нахождения отраженного w_{re} и преломленного w_r лучей, могут быть найдены при помощи заданных лучей падения w_i и нормали n (рисунок 1.1).

Для упрощения вычислений работа будет производиться с единичными векторами \widehat{w}_{re} , \widehat{w}_r , \widehat{w}_i и \widehat{n} .

Отраженный луч

Отраженный луч может быть представлен в виде комбинации векторов падения и нормали

$$\widehat{w}_{re} = \alpha \widehat{w}_i + \beta \widehat{n}.$$

Из равенства углов падения и отражения:

$$\widehat{w}_{re} \widehat{n} = \widehat{w}_i \widehat{n} = \alpha \widehat{w}_i \widehat{n} + \beta \widehat{n}^2 = \alpha \widehat{w}_i \widehat{n} + \beta$$

$$\beta = \widehat{w}_i \widehat{n} - \alpha \widehat{w}_i \widehat{n}.$$

Так как вектора единичные

$$\begin{aligned} |\widehat{w}_{re}| &= |\widehat{w}_i| = 1 \quad \uparrow^2 \\ |\widehat{w}_{re}|^2 &= |\widehat{w}_i|^2 = 1, \end{aligned}$$

$$\begin{aligned} 1 = w_{re}^2 &= (\alpha \widehat{w}_i + \beta \widehat{n})^2 = \alpha^2 \widehat{w}_i^2 + 2\alpha\beta \widehat{w}_i \widehat{n} + \beta^2 \widehat{n}^2 = \alpha^2 + 2\alpha\beta \widehat{w}_i \widehat{n} + \beta^2 = \\ &= \alpha^2 + 2\alpha(\widehat{w}_i \widehat{n} - \alpha \widehat{w}_i \widehat{n}) \widehat{w}_i \widehat{n} + (\widehat{w}_i \widehat{n} - \alpha \widehat{w}_i \widehat{n})^2 = \\ &= \alpha^2 + \underline{2\alpha(\widehat{w}_i \widehat{n})^2} - 2\alpha^2(\widehat{w}_i \widehat{n})^2 + (\widehat{w}_i \widehat{n})^2 - \underline{2\alpha(\widehat{w}_i \widehat{n})^2} + \alpha^2(\widehat{w}_i \widehat{n})^2 = \\ &= (\widehat{w}_i \widehat{n})^2 + (1 - (\widehat{w}_i \widehat{n})^2)\alpha^2, \end{aligned}$$

$$1 - (\widehat{w}_i \widehat{n})^2 = (1 - (\widehat{w}_i \widehat{n})^2)\alpha^2.$$

$$\text{Если } 1 - (\widehat{w}_i \widehat{n})^2 = 0 \Rightarrow \widehat{w}_i \widehat{n} = \pm 1 \Rightarrow \widehat{w}_i \parallel \widehat{n} \Rightarrow w_{re} = w_i,$$

иначе $\alpha^2 = 1 \Rightarrow \alpha = \pm 1$, так как падающий и отраженный луч не могут находиться по одну сторону от нормали, то $\alpha = -1$, $\beta = 2\widehat{w}_i \widehat{n}$, а $\widehat{w}_{re} = 2(\widehat{w}_i \widehat{n})\widehat{n} - \widehat{w}_i$.

Таким образом

$$w_{re} = \widehat{w}_{re}|w_i| = (2(\widehat{w}_i \widehat{n})\widehat{n} - \widehat{w}_i)|w_i| = (2\frac{w_i n}{|w_i||n|} \frac{n}{|n|} - \frac{w_i}{|w_i|})|w_i| = 2\frac{w_i n}{n^2}n - w_i. \quad (2.1)$$

Преломленный луч

Аналогично для преломленного луча

$$\widehat{w}_r = \alpha \widehat{w}_i + \beta \widehat{n}.$$

По закону Снеллиуса

$$\begin{aligned} n_1 \sin(\theta_i) &= n_2 \sin(\theta_r) \quad \uparrow^2 \\ n_1^2 \sin^2(\theta_i) &= n_2^2 \sin^2(\theta_r) \\ n_1^2 (1 - \cos^2(\theta_i)) &= n_2^2 (1 - \cos^2(\theta_r)). \end{aligned} \quad (2.2)$$

Так как $\cos \theta_i = \widehat{w}_i \widehat{n}$ и $\cos \theta_r = \widehat{w}_r \widehat{n}$, то выражение (2.2) преобразуется в

$$\begin{aligned} n_1^2 (1 - (\widehat{w}_i \widehat{n})^2) &= n_2^2 (1 - (\widehat{w}_r \widehat{n})^2) = n_2^2 (1 - (\alpha \widehat{w}_i \widehat{n} + \beta \widehat{n})^2) = \\ &= n_2^2 (1 - (\alpha^2 (\widehat{w}_i \widehat{n})^2 + 2\alpha\beta (\widehat{w}_i \widehat{n}) + \beta^2)) \end{aligned} \quad (2.3)$$

Так как векторы \widehat{w}_i и \widehat{w}_r единичные

$$1 = \widehat{w}_r^2 = (\alpha \widehat{w}_i + \beta \widehat{n})^2 = \alpha^2 + 2\alpha\beta(\widehat{w}_i\widehat{n}) + \beta^2, \quad (2.4)$$

$$2\alpha\beta(\widehat{w}_i\widehat{n}) = 1 - \alpha^2 - \beta^2. \quad (2.5)$$

Подставим (2.5) в (2.3)

$$\begin{aligned} n_1^2(1 - (\widehat{w}_i\widehat{n})^2) &= n_2^2(1 - (\alpha^2(\widehat{w}_i\widehat{n})^2 + 1 - \alpha^2 - \beta^2 + \beta^2)) \\ n_1^2(1 - (\widehat{w}_i\widehat{n})^2) &= n_2^2(1 - (\widehat{w}_i\widehat{n})^2)\alpha^2. \end{aligned}$$

Аналогично, если $1 - (\widehat{w}_i\widehat{n})^2 = 0$, то $w_{re} = -w_i$, иначе $\alpha = \pm \frac{n_1}{n_2}$. Так как падающий и преломленный лучи не могут находиться по одну сторону от нормали, то $\alpha = -\frac{n_1}{n_2}$. Подставим полученное значение в (2.4)

$$\begin{aligned} \beta^2 - 2\frac{n_1}{n_2}(\widehat{w}_i\widehat{n})\beta + \frac{n_1^2 - n_2^2}{n_2^2} &= 0 \\ D = 4\frac{n_1^2}{n_2^2}(\widehat{w}_i\widehat{n})^2 - 4\frac{n_1^2 - n_2^2}{n_2^2} &= \frac{4}{n_2^2}(n_2^2 - n_1^2(1 - (\widehat{w}_i\widehat{n})^2)). \end{aligned}$$

Пусть $D' = n_2^2 - n_1^2(1 - (\widehat{w}_i\widehat{n})^2)$, тогда $D = \frac{4}{n_2^2}D'$. В силу того, что $\frac{4}{n_2^2} > 0$, сравнение знака дискриминанта можно проводить с величиной D' .

Если $D' < 0$ - преломления нет.

Если $D' = 0$

$$\begin{aligned} \beta = \frac{n_1}{n_2}(\widehat{w}_i\widehat{n}) \Rightarrow \widehat{w}_r &= -\frac{n_1}{n_2}\widehat{w}_i + \frac{n_1}{n_2}(\widehat{w}_i\widehat{n})\widehat{n} \Rightarrow \\ \Rightarrow w_r &= -\frac{n_1}{n_2}w_i + \frac{n_1}{n_2}\frac{w_i n}{n^2}n = \frac{n_1}{n_2}\left(\frac{w_i n}{n^2}n - w_i\right). \end{aligned} \quad (2.6)$$

Если $D' > 0$

$$\beta = \frac{2\frac{n_1}{n_2}(\widehat{w}_i\widehat{n}) \pm \sqrt{D}}{2} = \frac{2\frac{n_1}{n_2}(\widehat{w}_i\widehat{n}) \pm \frac{2}{n_2}\sqrt{D'}}{2} = \frac{n_1(\widehat{w}_i\widehat{n}) \pm \sqrt{D'}}{n_2}$$

Для определения знака подставим α и β в выражение для \widehat{w}_r и проверим, что \widehat{w}_r и \widehat{n} расположены по разные стороны от границы пересечения, то есть $\widehat{w}_r\widehat{n} < 0$

$$\begin{aligned} \widehat{w}_r\widehat{n} &= \left(-\frac{n_1}{n_2}\widehat{w}_i + \frac{n_1(\widehat{w}_i\widehat{n}) \pm \sqrt{D'}}{n_2}\widehat{n}\right)\widehat{n} < 0 \\ -\frac{n_1}{n_2}(\widehat{w}_i\widehat{n}) + \frac{n_1(\widehat{w}_i\widehat{n}) \pm \sqrt{D'}}{n_2} &= \pm \frac{\sqrt{D'}}{n_2} < 0 \Rightarrow \\ \Rightarrow \beta &= \frac{n_1(\widehat{w}_i\widehat{n}) - \sqrt{D'}}{n_2} \Rightarrow \widehat{w}_r = -\frac{n_1}{n_2}\widehat{w}_i + \frac{n_1(\widehat{w}_i\widehat{n}) - \sqrt{D'}}{n_2}\widehat{n}. \end{aligned}$$

Таким образом итоговое значение w_r

$$w_r = \frac{n_1}{n_2}\left(\frac{w_i n}{n^2}n - w_i\right) - \frac{|w_i| D'}{|n| n_2}n. \quad (2.7)$$

2.3. Разработка алгоритмов

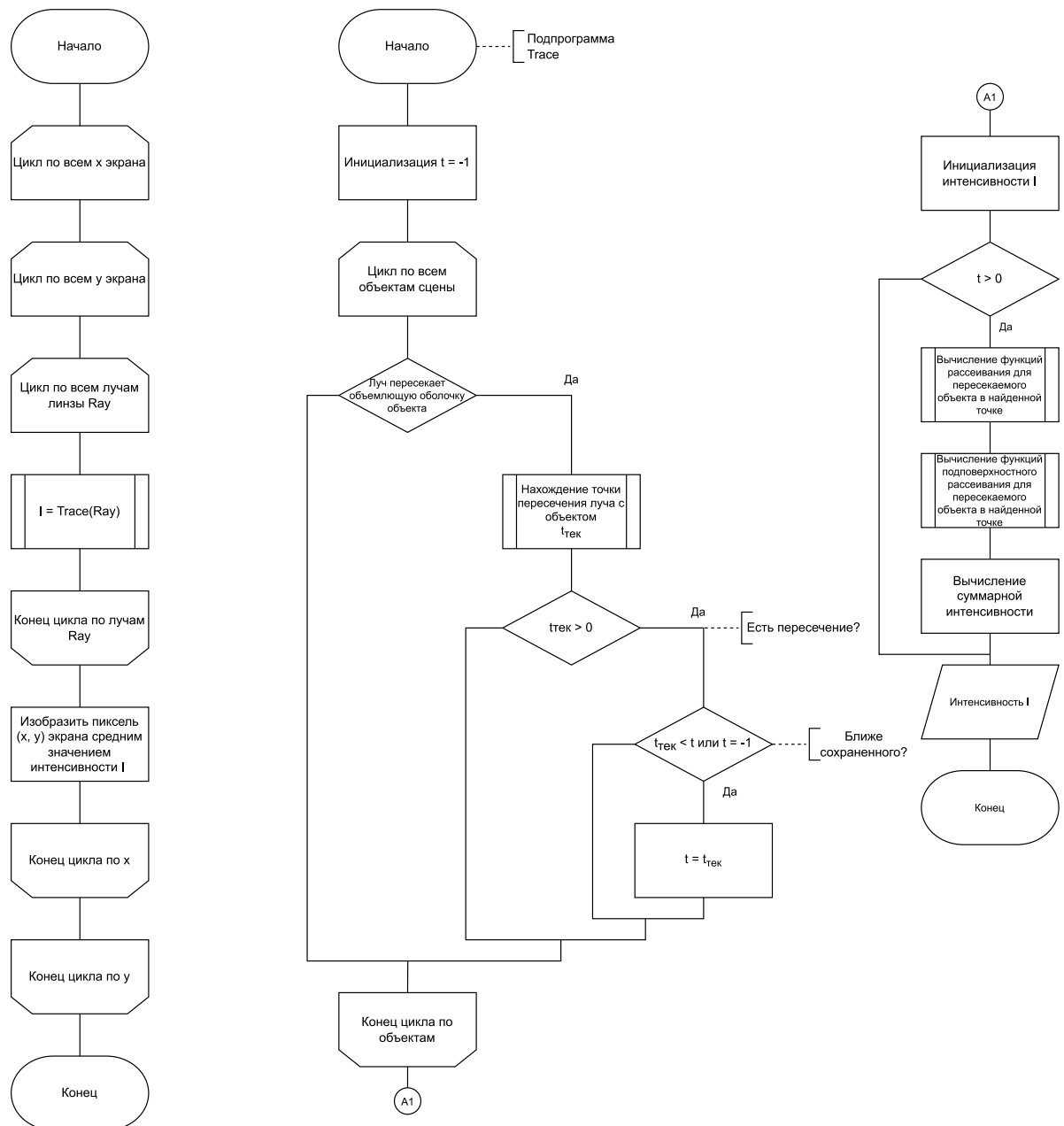


Рисунок 2.1 – Схема алгоритма трассировки лучей

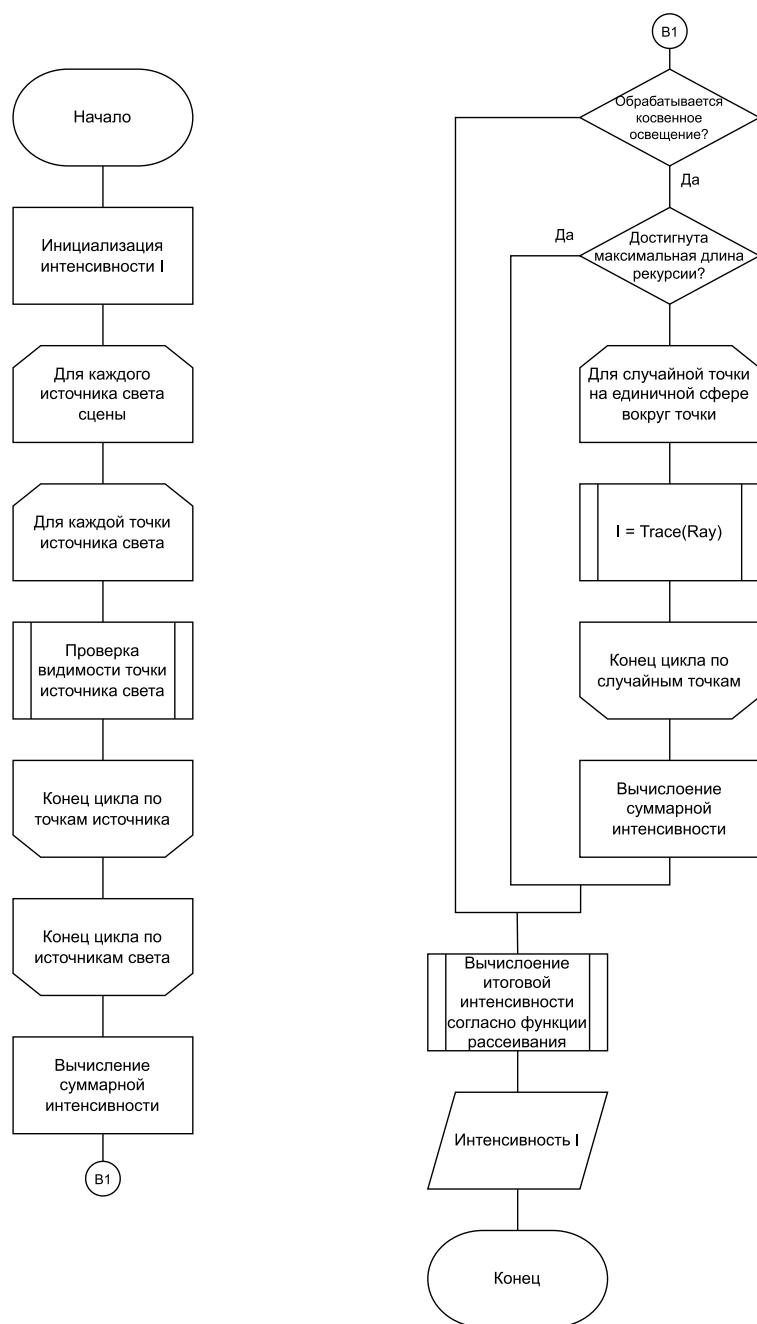


Рисунок 2.2 – Схема алгоритма вычисления функции рассеивания



Рисунок 2.3 – Схема алгоритма вычисления функции подповерхностного рассеивания

Список использованных источников