Министерство науки и высшего образования Российской Федерации
НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

А.Г. ЗАДОРОЖНЫЙ

МОДЕЛИ ОСВЕЩЕНИЯ И АЛГОРИТМЫ ЗАТЕНЕНИЯ В КОМПЬЮТЕРНОЙ ГРАФИКЕ

Утверждено
Редакционно-издательским советом университета
в качестве учебного пособия

НОВОСИБИРСК
2020

УДК 004.92(075.8)
З-156

Рецензенты:
канд. техн. наук, доцент *В.С. Карманов*
доктор техн. наук, профессор *М.Э. Рояк*

Работа подготовлена на кафедре прикладной математики НГТУ

Задорожный А.Г.
З-156 Модели освещения и алгоритмы затенения в компьютерной графике: учебное пособие / А.Г. Задорожный. – Новосибирск: Изд-во НГТУ, 2020. – 80 с.

ISBN 978-5-7782-4308-8

В данном учебном пособии рассмотрены элементы теории освещения в компьютерной графики, включая модели локального освещения и алгоритмы затенения. Также рассмотрены и соответствующие функции графической библиотеки OpenGL, приведены соответствующие примеры. Пособие может быть рекомендовано как для самостоятельного изучения курса «Компьютерная графика», так и для подготовки к практическим и расчетно-графическим заданиям.

УДК 004.92(075.8)

ISBN 978-5-7782-4308-8

ФИЗИЧЕСКОЕ ОСВЕЩЕНИЕ

Создание реалистичного освещения в сцене является одной из самых больших проблем при разработке трехмерной графики. В реальности падающий луч света претерпевает огромное количество отражений и преломлений, что при компьютерном моделировании повторить нереально. Тем не менее, освещение в трехмерной сцене все же можно приблизить к реальному как за счет продвинутых моделей освещения, так и за счет развития аппаратного обеспечения.

Зрение (зрительное восприятие) – это способность воспринимать информацию путем преобразования энергии электромагнитного излучения светового диапазона, осуществляемая зрительной системой. В частности, цвет предметов есть результат интерпретации света, отраженного от предметов.

Обработка светового сигнала начинается на сетчатке глаза, происходит возбуждение фоторецепторов, далее идет передача и преобразование зрительной информации в нейронных слоях с формированием в затылочной доле коры больших полушарий зрительного образа.

Весь этот процесс проходит множество этапов восприятия, на каждом из которых возникают разного рода искажения, которые мозг и пытается нивелировать. Так устраняются сферическая и хроматическая аберрации, эффекты слепого пятна, проводится цветокоррекция, и многое другое. В тех же случаях, когда подсознательная обработка информации недостаточна либо избыточна, и возникают оптические иллюзии. В связи с этим видимое разумом изображение не вполне соответствует реальности.

Соответственно, созданием изображения является процесс формирования светового образа на сетчатке глаза таким образом, чтобы он интерпретировался мозгом как требуемый объект, и разнообразие вариантов освещения является одним из вариантов увеличения количества необходимой для этого информации.

На рис.1 приведен пример отображения куба и сферы в различных режимах визуализации. Видно, что переход в режим цветных граней

немного улучшает восприятие фигур, но не дает достаточно информации о трехмерности. Переход в каркасный режим с одной стороны улучает восприятие трехмерности, а с другой – затрудняет из-за обилия вспомогательных деталей. И только включение освещения позволяет мозгу классифицировать эти фигуры не как плоские (многоугольники и круг), а как трехмерные (куб и сфера).

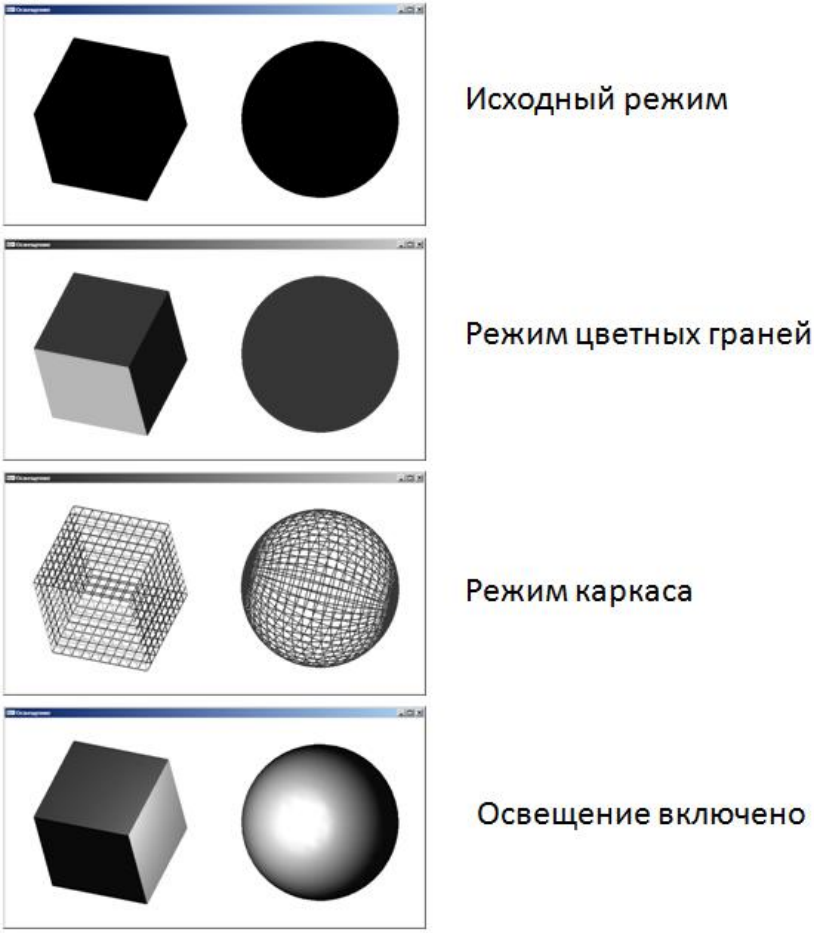


Рис.1. Куб и сфера в различных режимах отображения.

ВИДИМОЕ ИЗЛУЧЕНИЕ

В физической оптике, занимающейся изучением света, под светом понимается *видимое излучение* – электромагнитное излучение, воспринимаемое человеческим глазом. В широком же смысле под *светом* часто называют любое оптическое излучение с длинами волн порядка $10^{-9} - 10^{-4}$ метра, что включает в себя как ультрафиолетовый (10-400 нм) диапазон, так и инфракрасный (0.7-2000 мкм).

Видимое излучение является частью *оптического окна* – области спектра электромагнитного излучения, практически не поглощаемого земной атмосферой с длинами волн порядка длин волн 300—1500 нм. В ультрафиолетовой области прозрачность ограничена поглощением ультрафиолета озоновым слоем и водой, а в инфракрасной области — поглощением водой. Поэтому на сравнительно узкую видимую область спектра приходится более 40% энергии излучения Солнца у поверхности. В частности, небо выглядит голубым потому, что чистый воздух рассеивает синий свет лучше, чем, например, красный (у которого большая длина волны).

Чувствительность человеческого глаза к электромагнитному излучению зависит от длины волны излучения. Максимум чувствительности приходится на зеленый участок диапазона с длиной волны 555 нм, при удалении от которого чувствительность постепенно спадает до нуля, однако невозможно указать точные границы спектрального диапазона видимого излучения. Поэтому обычно в качестве коротковолновой границы спектрального диапазона света, принят участок с длинами волн в вакууме 380-400 нм, а в качестве длинноволновой границы – участок 760-780 нм.

При разложении луча белого цвета в призме образуется спектр, в котором излучения разных длин волн преломляются под разными углами. Цветам, входящие в этот спектр, соответствует излучение с малым разбросом длин волн – *монохроматическое излучение*. Основные спектральные цвета и их характеристики, представлены в табл.1. Такие цвета, как розовый или бежевый образуются только в результате сме-

шения нескольких монохроматических излучений с различными длинами волн.

В отличие от человека, в животном мире диапазон видимого излучения существенно шире, например, видеть в ультрафиолетовом диапазоне могут различные насекомые, птицы, рыбы, кошки и собаки. Но вот созданий, способных видеть в инфракрасном диапазоне, до сих пор не обнаружено, а под *инфракрасным зрением* обычно понимают способность воспринимать специальными сенсорами (терморецепторами) тепловое излучение, что свойственно некоторым видам змей и летучих мышей, в этом случае в мозгу формируется несфокусированное изображение теплых объектов.

Таблица 1

Основные спектральные цвета

Цвет	Диапазон волн, нм
Фиолетовый	≤ 450
Синий	$450 \div 480$
Голубой	$480 \div 510$
Зеленый	$510 \div 550$
Салатовый	$550 \div 570$
Желтый	$570 \div 590$
Оранжевый	$590 \div 630$
Красный	≥ 630

СВЕТОВОЙ ПОТОК

Свет состоит из *фотонов* – квантов электромагнитного излучения. Фотонов испускается огромное количество, например, обычная стоватная лампочка накаливания испускает порядка 10^{21} фотонов в секунду. Исходя из этого, свет можно рассматривать как непрерывный поток энергии и применять к нему статистические законы. Соответственно, в компьютерной графике свет рассматривается как непрерывный прямолинейный поток несталкивающихся между собой частиц.

ВЗАИМОДЕЙСТВИЕ С ПОВЕРХНОСТЬЮ

Когда луч света попадает на поверхность, он теряет часть своей энергии (физически, этот эффект связан с нагревом поверхности) и *отражается* под некоторым углом (reflection).

Если поверхность абсолютно гладкая, то лучи отражаются под тем же углом, под каким упали, в результате появляется *эффект блеска* (shininess), что можно наблюдать у любой глянцевой поверхности. Если же поверхность абсолютно шершавая, лучи отражаются по всем направлениям – *эффект рассеивания* (diffuse), он характерен для матовых поверхностей. В реальности каждый материал так или иначе, но обладает свойствами блеска и диффузии, при этом эффект блеска проявляется лишь при определенной позиции наблюдателя.

В зависимости от свойств поверхности свет может не только отразиться, но и проникнуть сквозь поверхность, в этом случае он переходит из одной среды в другую, меняя свое направление – *эффект преломления* (refraction).

После взаимодействия с поверхностью, луч света продолжает свой путь, достигая других объектов, пока не растеряет всю свою энергию или не вылетит за пределы пространства.

Эти многочленные взаимодействия весьма трудоемки для моделирования даже при использовании современного программного и аппаратного обеспечения.

ЗАТУХАНИЕ С РАССТОЯНИЕМ

С момента включения источника свет начинает распространяться в разные стороны от источника, образуя в общем случае сферу, радиус которой растет по мере удаления от источника.

Если рассмотреть пучок лучей, то момент времени t_1 он осветит некоторый участок сферы. В момент времени t_2 размер сферы увеличится – соответственно, увеличится и освещаемый данным пучком участок (рис.2). А поскольку плотность фотонов в пучке неизменна, второму участку достается меньше фотонов, в результате чего его яркость снизится по сравнению с первым участком.

Поскольку площадь поверхности сферы прямо пропорциональна квадрату ее радиуса ($s = 4 \cdot \pi \cdot r^2$), то яркость освещения I фрагмента сферы обратно пропорциональна квадрату расстояния от источника:

$$I = \frac{k}{r^2},$$

где k – величина, зависящая от яркости источника.

Данный закон справедлив в тех случаях, когда размер источника пренебрежимо мал.

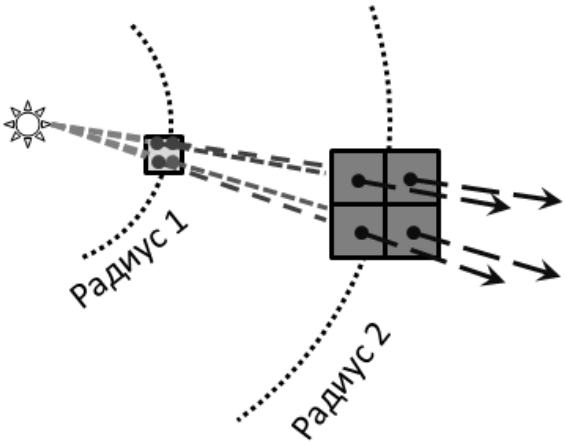


Рис.2. Распространение света во времени.

ПРОБЛЕМА ВЫБОРА ОСВЕЩЕНИЯ

Проблема правильного освещения в изображениях возникла задолго до появления компьютерной графики: первыми с этой задачей столкнулись еще художники, позже – фотографы и кинооператоры.

Выбор освещения зависит множества моментов: материалы объектов, геометрия сцены, действия в сцене, тип и цвет осветителя. В частности, прожекторный осветитель позволяет сконцентрировать внимание на каком-то определенном объекте, а фоновый или точечный – осветить сцену целиком. Также важен и цвет осветителя: искусственное освещение обычно имеет голубой или желтоватый оттенок, а уличное зависит от времени суток (например, красный свет во время заката).

РЕКОМЕНДАЦИИ

Существует множество приемов, с помощью которых можно осветить сцену таким образом, чтобы скрыть мелкие недостатки и подчеркнуть важные детали. Например, чтобы придать объем трехмерной модели, ее достаточно подсветить сзади – в этом случае появится отчетливая граница, визуально отделяющая объект от фона. А если требуется осветить половину объекта, то вторая его половина должна быть также подсвечена источником света с меньшей интенсивностью, в противном случае затененный участок будет скрыт в неестественной темноте (в реальности свет должен отразиться от множества поверхностей и хотя бы слабо, но подчеркнуть контур затененной стороны).

Существуют и общие рекомендации, как не нужно освещать сцену. Неудачное расположение осветителей может создать слишком темные участки в сцене, а сами объекты могут быть плохо видны из-за недостаточной или избыточной освещенности. Избыточное количество осветителей приводит к засвечиванию соответствующих участков сцены и появлению паразитных теней.

ТРЕХТОЧЕЧНОЕ ОСВЕЩЕНИЕ

Самым распространенным способом освещения важных объектов сцены является освещение из трех точек (трехточечная система).

В данном подходе источники света в кадре играют три различные роли. Для каждой из ролей может быть создано несколько осветителей, но общий эффект должен быть таким, как будто в сцене существуют основной (ключевой) источник света, более мягкий заполняющий осветитель и источник контрового света, отделяющего объекты от фона (рис.3).

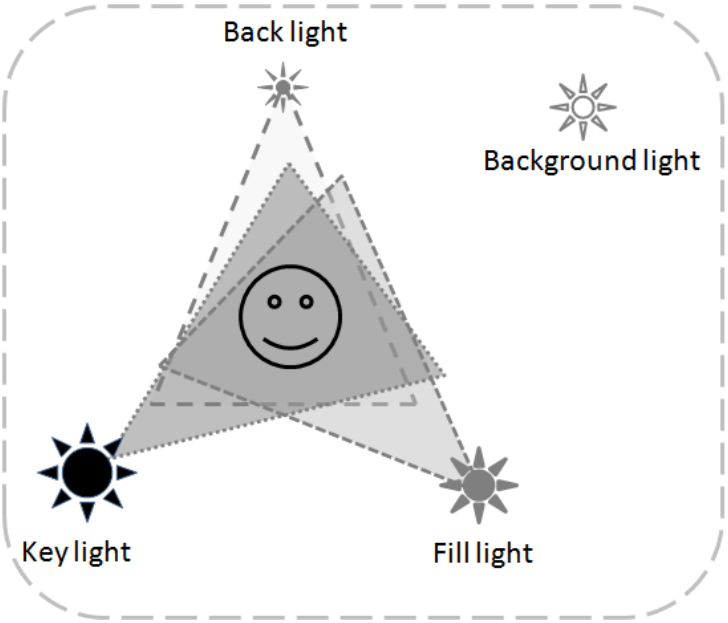


Рис.3. Схема освещения с трех точек.

Такой подход к освещению гарантирует не только освещение объектов сцены, но и появление на их поверхностях зеркальных бликов и теней. Освещение с трех различных точек позволяет добиться максимальной глубины сцены. Но в случаях сложных сцен данный подход может не сработать.

Ключевой источник света (key light) – это основной осветитель сцены. Обычно он располагается на некотором расстоянии справа или слева от камеры, что дает более яркое освещение с одной из сторон и увеличивает глубину кадра. Кроме того, именно этот источник света образует тени от объектов, позволяющие определить, с какой стороны освещена сцена.

Заполняющий источник света (fill light) не имеет ярко выраженного направления распространения лучей, они равномерно распределяются по затемненным областям сцены. Такой источник осветляет тени, создаваемые ключевым источником света, но при этом не создает свои тени. Обычно заполняющий источник света располагается по другую сторону от ключевого источника и имеет меньшую интенсивность.

Контровой свет (back light) располагается позади объектов и формирует небольшой ореол, позволяющий выделить объекты на фоне декораций. Контровой свет приводит к появлению бликов на кромках объектов, подчеркивая их форму и привлекая к ним дополнительное внимание.

Также есть и вспомогательный фоновый источник (background light), предназначенный для освещения обстановки за основными объектами сцены.

Система освещения с трех точек используется для основных объектов сцены. Так как производимый эффект зависит от положения камеры и ее ориентации по отношению к объекту съемки, перемещение камеры требует изменения настроек освещения. Таким образом, расположение источников света при освещении с трех точек зависит не от содержания сцены, а от положения камеры.

ВИРТУАЛЬНОЕ ОСВЕЩЕНИЕ

Создание реалистичного освещения сцены требует не меньших усилий, чем ее моделирование. Добиться этого непросто, ведь в виртуальном мире источники света работают совсем не так, как в реальности. Например, виртуальные источники света можно настроить так, что они будут не увеличивать, а уменьшать освещенность сцены, что принципиально невозможно в обычном мире.

В реальности обычной лампочки достаточно для освещения целой комнату, включая недоступные прямому освещению места, тем самым формируется рассеянное освещение, а вот в программных пакетах эти области останутся неосвещенными. Для исправления ситуации приходится либо устанавливать дополнительные осветители, либо использовать трудоемкие алгоритмы генерации затенения.

Самыми известными графическими программами для создания реалистичных сцен являются Maya и 3DSMAX от компании AUTODESK. В Maya доступно 6 различных источников света (табл.2), а генерация теней осуществляется двумя алгоритмами: более быстрым Shadow Map и более качественным Ray tracing. В 3DSMAX также доступно 6 источников света (табл.3), но реализовано больше алгоритмов генерации теней: Shadow Map, Ray traced shadows, Adv ray traced, Area shadows, Mental ray shadow map.

Разумеется, у этих профессиональных приложений существует масса альтернатив (более дешевых и более простых), например:

1) Blender;
2) SketchUp (бывший Google Sketchup);
3) CINEMA 4D;
4) K-3D;
5) MODO;
6) LightWave.

Таблица 2

Источники света в AUTODESK MAYA

№	Источник	Описание действия	Цель использования
1	Ambient (Фоновый)	Равномерное освещение	Общий световой фон
2	Directional (Направленный)	Освещение из бесконечности по направлению	Ключевой, заполняющий и контровой осветители
			Имитация солнечного света
			Общее освещения
3	Point (Точечный)	Освещение из точки во всех направлениях	Имитации лампочки
			Общее освещения
4	Spot (Прожектор)	Освещение из точки конусом по направлению	Имитация прожектора
			Ключевой и заполняющий осветители
			Получение световых пятен
5	Area (Прямоугольный)	Освещение из прямоугольника по направлению	Имитация узкой полоски света
6	Volume (Объемный)	Освещение в ограниченной области	Имитация световых эффектов: - пламени - тумана - фар автомобиля - света маяка

Таблица 3

Источники света в AUTODESK 3DSMAX

№	Источник	Описание
1	Omni (точечный)	Точечный источник
	Target Spot (направленный прожектор)	Прожектор в виде конуса или пирамиды
3	Free Spot (свободный прожектор)	Target Spot, но без точного указания направления
4	Target Direct (направленные параллельные лучи)	Направленный источник в форме параллелепипеда или цилиндра
5	Free Direct (свободные параллельные лучи)	Target Direct, но без точного указания направления
6	Skylight (небесный свет)	Солнечное освещение

ОСНОВНЫЕ ТИПЫ ИСТОЧНИКОВ СВЕТА

В табл.4 приведены основные источники освещения, используемые в компьютерной графике. В общем случае источники света характеризуются такими параметрами, как местоположение, направление и интенсивность излучения.

Таблица 4

Основные источники света

№	Источник	Описание
1	Фоновый (ambient)	Светится само пространство
2	Точечный (point)	Свет из точки распространяется шарообразно, во все стороны
3	Прожекторый (spot)	Свет из точки распространяется конусом в определенном направлении
4	Направленный, Удаленный (directional)	Свет распространяется из бесконечности
5	Объемный (volume)	Свет распространяется в/из некоторой области

ФОНОВЫЙ ИСТОЧНИК

В реальности свет, доходя до поверхностей, многократно между ними отражается и преломляется, порождая определенный уровень освещенности пространства. Просчитать эти многократные лучи света при моделировании крайне затруднительно, поэтому их результат часто аппроксимируется так называемым *фоновым освещением*. Поскольку может быть несколько источников фонового освещения (*локальное фоновое освещение*), то их имеет смысл объединить и выделить в отдельный, *глобальный фоновый источник света*.

Фоновое освещение характеризуется равномерной освещенностью всего пространства и не создает теней, поэтому его источник задается лишь интенсивностью.

Данный вариант освещения может использоваться для моделирования сумеречного освещения.

ЭМИССИОННЫЙ ИСТОЧНИК

Эмиссионное освещение (подвид фонового) используется только для самоподсветки задаваемого объекта, поэтому является равномерным, не создает тени и не распространяется на другие объекты.

Источник такого освещения задается лишь интенсивностью и может быть распределен по всей поверхности.

Эмиссионное освещение может использоваться для моделирования фар автомобиля, лампы накаливания.

ТОЧЕЧНЫЙ ИСТОЧНИК

Точечное освещение является наиболее распространенным и характеризуется равномерным по всем направления распространением света из бесконечно малой точки.

Источник такого освещения задается местоположением и интенсивностью.

Данный вариант освещения может использоваться для моделирования света лампы.

ПРОЖЕКТОРНЫЙ ИСТОЧНИК

Прожекторное освещение эмулирует работу прожектора (фонарика) и характеризуется равномерным конусным распространением света из бесконечно малой точки. Соответственно, для задания данного источника помимо интенсивности необходимо указать местоположение, направление и угол конуса распространения (отсечения), функцию распределения для прожекторного пятна (если константное, то освещенность не будет затухать к краям).

НАПРАВЛЕННЫЙ ИСТОЧНИК

Фактически, данный источник света является точечным или прожекторным, который удален от сцены на бесконечно далекое расстояние. Это приводит к тому, что все лучи распространяются параллельно, что позволяет, в частности, ускорить расчеты по модели освещения.

Источник такого освещения задается интенсивностью и направлением распространения.

Данный вариант освещения может использоваться для моделирования оконного или солнечного освещения.

ОБЪЕМНЫЙ ИСТОЧНИК

Это сложный источник, обладающий ненулевым размером поверхности и направлением распространения.

Сам источник можно визуализировать с помощью эмиссионного, а распространение света моделируется через комбинацию направленного и прожекторного источников. Например, излучающую поверхность можно представить в виде большого числа узконаправленных прожекторов.

ФАКТОРЫ, ВЛИЯЮЩИЕ НА ОСВЕЩЕННОСТЬ

На интенсивность как падающего, так и отраженного света оказывают влияние различные факторы, в частности:

1. Параметры источников света:
    1) Интенсивность излучения;
    2) Месторасположение;
    3) Направление излучения;
    4) Количество источников.
2. Затухание света с расстоянием.
3. Свойства материала поверхности:
    1) Поглощение света;
    2) Геометрическая структура;
    3) Прозрачность и полупрозрачность;
    4) Подповерхностное рассеивание.
4. Отражение от других объектов.
5. Позиция наблюдателя.
6. Направления нормалей.

Обычно интенсивность источника устанавливают в диапазоне $[0, 1]$. Но, в принципе, интенсивность источника можно установить и в диапазоне $[-1, 0]$, в этом случае источник используется не для освещения темной области, а для затенения светлой области.

Как бы то ни было, во избежание нарушения закона сохранения энергии результирующая интенсивность не должна превышать исходную.

УЧЕТ ПОГЛОЩЕНИЯ

При попадании света на поверхность, часть его интенсивности L поглощается поверхностью, а часть отражается с интенсивностью I:

$$I = L \cdot m,$$

где $m \in [0,1]$ – коэффициент поглощения.

Соответственно, при $m = 0$ весь падающий свет полностью поглощается, и получается эффект абсолютно черного тела. При $m = 1$ весь свет полностью отражается, вызывая эффект идеального зеркала.

УЧЕТ НЕСКОЛЬКИХ ИСТОЧНИКОВ

При наличии нескольких источников каждый из них просчитывается независимо, а их результат складывается:

$$I = \sum_{j=1}^{n} I_j,$$

где n – количество источников, I_j – интенсивность отраженного света для j-го источника.

В этом случае вполне возможна ситуация, когда в некоторой точке результирующая интенсивность окажется больше максимально возможной. Из этой ситуации существует три варианта выхода:

1) Уменьшить интенсивность источников;
2) Обрезать до максимума все, что превышает максимум (получится эффект засветки кадра);
3) Пронормировать интенсивность отраженного света обратно в диапазон $[0,1]$.

Очевидно, что первый вариант требует ручного подбора, а третий вариант – хранения и обработки каждой интенсивности. В связи с этим обычно используется второй вариант.

УЧЕТ ЗАТУХАНИЯ

С пройденным расстоянием интенсивность света L уменьшается (damping factor) с некоторым коэффициентом k_{DF}:

$$L' = L \cdot k_{DF}.$$

Обычно для расчета k_{DF} используется следующая формула:

$$k_{DF}\left(dist\right) = \frac{1}{k_{const} + k_{linear} \cdot dist + k_{quadr} \cdot dist^2},$$

где:

$dist$ – расстояние от источника до освещаемой точки,
k_{const} – постоянный фактор затухания,
k_{linear} – линейный фактор затухания,
k_{quadr} – квадратичный фактор затухания,

Здесь k_{const}, k_{linear} и k_{quadr} – некоторые вещественные константы, не равные нулю одновременно. Выбор значений этих констант для конкретной сцены осуществляется методом проб и ошибок. На рис.4 приведены примеры значений этих констант в зависимости от эффективного расстояния работы источника, когда интенсивность света падает со 100% (на расстоянии 0) до 0.1% (на расстоянии S), причем, большая часть света попадает в первые 20% диапазона. Также на рисунке представлен график падения интенсивности для $S = 100$.

S	k_{const}	k_{linear}	k_{quadr}
20	1.0	2.2e-1	2.0e-1
50	1.0	9.0e-2	3.2 e-2
100	1.0	4.5e-2	7.5e-3
200	1.0	2.2e-2	1.9 e-3
600	1.0	7.0e-3	2.0e-4

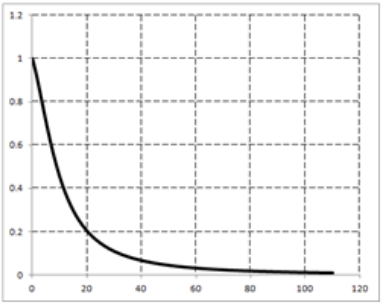


Рис.4. Затухание интенсивности да дистанции до 100.

НОРМАЛИ

Все модели освещения основаны на использовании понятия нормали. *Нормаль* – это вектор, который перпендикулярен поверхности, которую требуется осветить. *Нормаль к поверхности в точке* – это прямая, перпендикулярная к касательной плоскости в данной точке. *Вектор нормали* к поверхности в точке – это единичный вектор, приложенный к данной точке и параллельный направлению нормали.

В компьютерной же графике под *нормалью* понимается единичный вектор, перпендикулярный в точке к плоскости примитива. В принципе, нормаль может быть и не единична, и не перпендикулярна поверхности, но это уже будет сказываться на качестве освещения.

Нормаль для гладкой поверхности определяется однозначно, при этом в каждой точке можно задать два нормальных вектора, отличающихся знаком (направлением), что и позволяет определять ориентацию поверхности. Для лицевых граней нормаль считается направленной вовне.

Для ряда объектов нормаль известна аналитически, например, у сферы она совпадает с антинаправлением в центр, а у плоскости коэффициенты при переменных и являются соответствующими компонентами нормали.

Но в большинстве случаев объекты задаются набором примитивов, поэтому нормаль приходится рассчитывать через векторное произведение смежных ребер примитива. По определению, вектор $\vec{c}$ является векторным произведением векторов $\vec{a}$ и $\vec{b}$, если

$$\vec{c} = \left[\vec{a}, \vec{b}\right] = \vec{a} \times \vec{b} = \begin{vmatrix} i & j & k \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix} = \begin{pmatrix} a_y \cdot b_z - a_z \cdot b_y \\ a_z \cdot b_x - a_x \cdot b_z \\ a_x \cdot b_y - a_y \cdot b_x \end{pmatrix}.$$

РАСЧЕТ НОРМАЛЕЙ В ПЛОСКОСТИ

На рис.5 представлен пример расчета нормали путем векторного произведения в вершине A как для одного треугольника ABC, так и для лежащих в одной плоскости треугольников ABC, ACD, ADE.

Как можно видеть, направление нормали зависит от направления перебора смежных вершин (ребер). Если ребро $\overrightarrow{BA}$ векторно умножить на ребро $\overrightarrow{CA}$, то по правилу буравчика нормаль будет направлена вверх ("наружу"), соответственно, такая грань будет лицевой. Но если ребро $\overrightarrow{CA}$ векторно умножить на $\overrightarrow{BA}$, то нормаль окажется направленной вниз, такая грань уже становится нелицевой.

В случае, когда в плоскости находятся несколько треугольников, с общей вершиной A, тогда при обходе против часовой стрелки в каждом из треугольников в этой вершине нормаль будет направлена одинаково, но ее длина (модуль) для каждого треугольника будет уже своя. Соответственно, при расчете освещения нормали нужно приводить к одной длине путем нормализации.

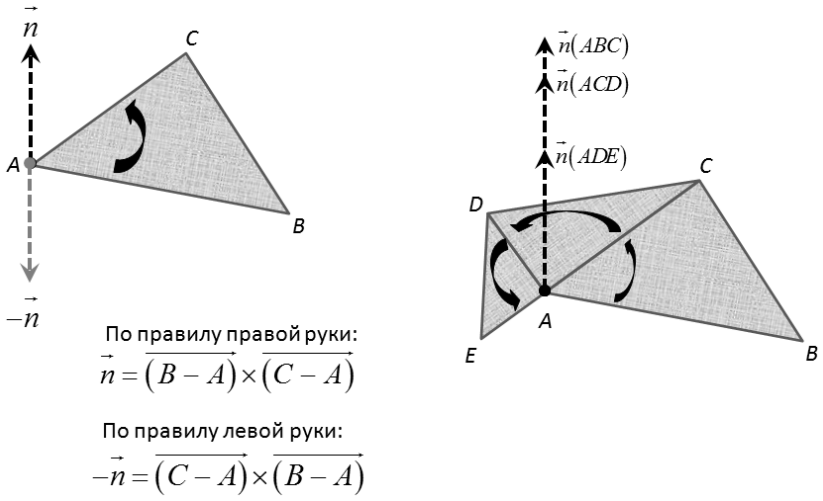


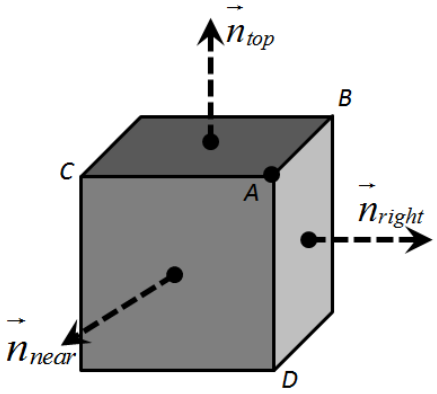
Рис.5. Расчет нормали в вершине треугольника.

РАСЧЕТ НОРМАЛЕЙ В ТРЕХМЕРНОМ ПРОСТРАНСТВЕ

Отдельный интерес представляет ситуация, когда нележащие в одной плоскости грани имеют общую вершину, например, вершину куба (рис.6). В этой ситуации получается, что вершина *A* является общей для трех граней, и ей соответствует 3 разнонаправленных нормали. Если так и оставить, получится освещение в режиме *плоских нормалей* (когда у каждой грани нормали независимы от нормалей других граней). Но можно нормали в общей вершине векторно сложить (рис.7), получив режим *сглаженных нормалей*:

$$\vec{n}_{smooth} = \vec{n}_{top} + \vec{n}_{right} + \vec{n}_{near}.$$

Пример сравнения режимов "плоской" и "сглаженной" нормалей при освещении куба справа представлен на рис.8. Как можно видеть, в режиме сглаживания нормалей пропадают четкие границы перехода между гранями. Этот эффект, с одной стороны улучшает реалистичность (в реальном мире редко где бывает несмазанный переход), но с другой – усложняет разбор геометрии.



$$\vec{n}_{top} = \overrightarrow{(B-A)} \times \overrightarrow{(C-A)}$$

$$\vec{n}_{right} = \overrightarrow{(D-A)} \times \overrightarrow{(B-A)}$$

$$\vec{n}_{near} = \overrightarrow{(C-A)} \times \overrightarrow{(D-A)}$$

Рис.6. Расчет нормалей в вершине куба.

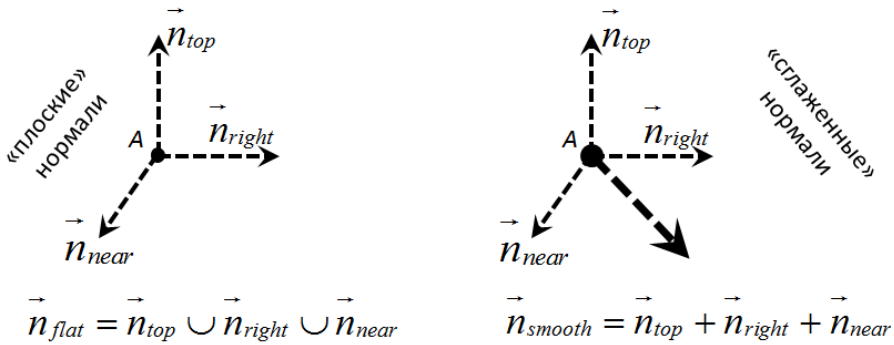


Рис.7. "Плоская" и "сглаженная" трехмерные нормали.

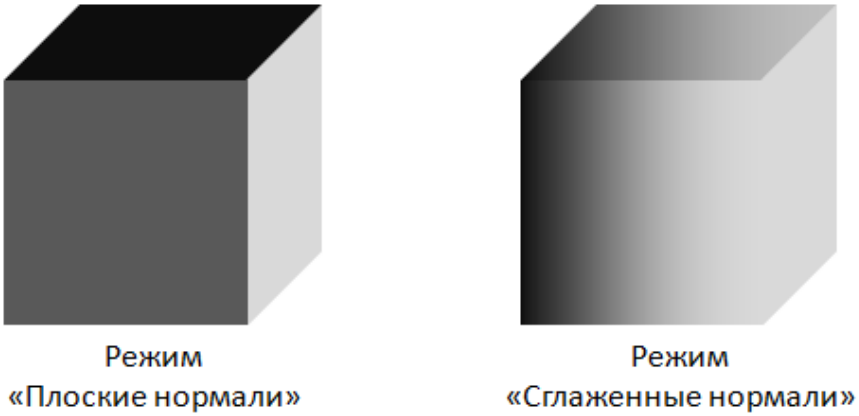


Рис.8. Освещение куба в режимах "плоской" и "сглаженной" нормалей.

ЭФФЕКТ СГЛАЖИВАНИЯ НОРМАЛЕЙ

Эффект сглаживания нормалей часто используется для повышения реалистичности сцены с недостаточным набором полигонов. С одной стороны, в природе мало где есть четкие границы между гранями, поскольку со временем все стирается (шлифуется). С другой стороны, сглаживание переходов между гранями вызывает эффект более качественной (высокополигональной) модели.

На рис.9 представлен пример подобной ситуации. Как можно видеть, использование сглаживания нормалей позволяет более чем в 2 раза сократить число полигонов (четырехугольников) для сферы.

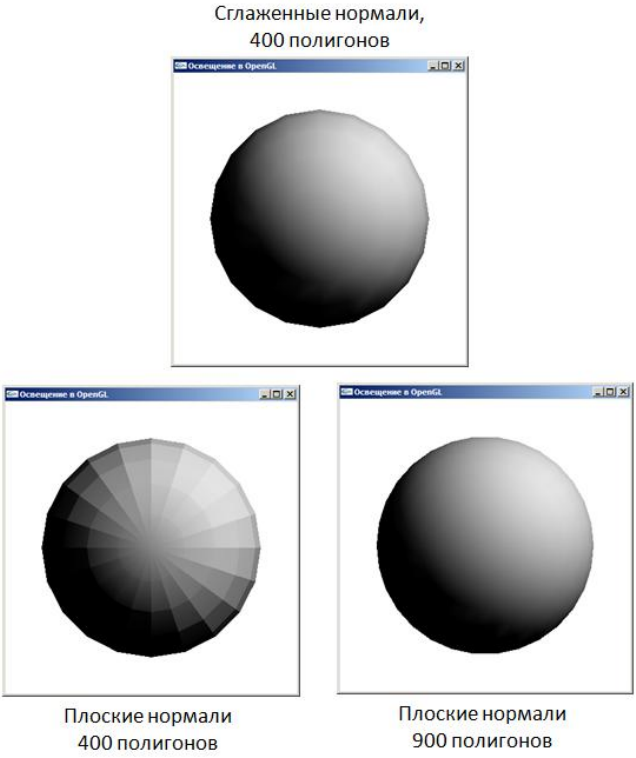


Рис.9. Освещение сферы в режимах "плоской" и "сглаженной" нормалей.

ВЛИЯНИЕ МОДЕЛЬНЫХ ПРЕОБРАЗОВАНИЙ

Вообще говоря, раз нормали задаются при указании вершины, то и на них должно оказывать влияние текущая система координат, установленная выполненными модельными преобразованиями (операциями сдвига, поворота и масштаба).

Поскольку нормаль есть вектор в однородных координатах, то операция сдвига не оказывает на нормаль влияния. В свою очередь, поворот системы координат одновременно меняет направление нормали и грани, но при этом не меняет длину самой нормали. Однако операция масштабирования уже может оказать влияние как на длину нормали, так и на ее ориентацию.

При однородном масштабировании фигура растягивается по всем осям одинаково, оказывая влияние на длину вектора, но не на его направление. При неоднородном же масштабировании коэффициенты масштаба к осям разные, что приводит к соответствующему искажению направления нормали, соответствующие примеры приведены на рис.10. И если длину нового вектора путем нормализации легко вернуть в исходное единичное состояние, то для восстановления направления нормали придется либо применить к ней обратное преобразование модели, либо заново пересчитать уже в новой геометрии.

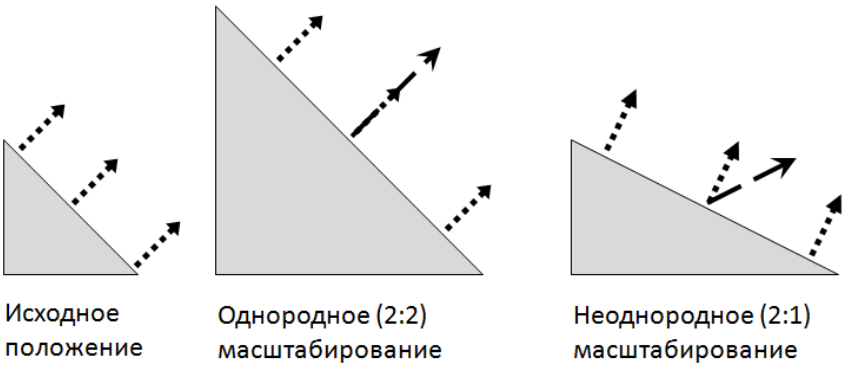


Рис.10. Влияние масштабирования на нормаль.

ТРЕХКОМПОНЕНТНОЕ ОСВЕЩЕНИЕ

Излучаемый источником свет с интенсивностью L доходит до поверхности, характеризующейся коэффициентом поглощения m, и отражается с интенсивностью I. Помимо свойств материала и угла падения на отраженный от поверхности свет может оказать влияние еще ряд факторов, например, затухание света с расстоянием.

Предполагается, что свет (как излучаемый, так и отраженный от поверхности) состоит из трех стандартных компонент (табл.5): фоновой, диффузной (иногда ее еще называют рассеянной) и зеркальной.

При вычислении компонент часто используется скалярное произведение двух векторов, которое по определению является произведением длин этих векторов на косинус угла между ними. А поскольку используются лишь единичные вектора, то справедливо следующее равенство:

$$cos(\vec{A}, \vec{B}) = (\vec{A} \cdot \vec{B}).$$

Таблица 5

Компоненты освещения

Вид излучения	Излучаемый свет	Отраженный свет	Свойство материала
Фоновое (ambient)	L_a	I_a	m_a
Диффузное (diffuse)	L_d	I_d	m_d
Зеркальное (specular)	L_s	I_s	m_s

ФОНОВОЕ ОСВЕЩЕНИЕ

Интенсивность фоновой составляющей рассчитывается по элементарному закону:

$$I_a = m_a \cdot L_a.$$

Из формулы видно, что фоновая составляющая освещенности не зависит от пространственных координат освещаемой точки и источника. Поэтому при моделировании освещения не имеет смысла брать более одного фонового источника света, поэтому обычно просто задается некое глобальное фоновое освещение всей сцены.

ДИФФУЗНОЕ ОСВЕЩЕНИЕ

Некоторые поверхности отражают падающий свет одинаково во всех направлениях (например, гипс). Такие поверхности называют абсолютно рассеивающими или *поверхностями Ламберта* (Ламберт описывал отражение света такими поверхностями еще в 1760 г.).

В большинстве же случаев свет при попадании на поверхность рассеивается достаточно равномерно во все стороны. Соответственно, при моделировании такого освещения учитывается только ориентация поверхности $\vec{n}$ (табл.6) и направление на источник света $\vec{s}$ (рис.11): максимальное освещение достигается лишь при падении света на поверхность под прямым углом. Из геометрических соображений следует, что если косинус угла θ выходит за пределы диапазона $[0,1]$, то поверхность не должна оказаться освещенной.

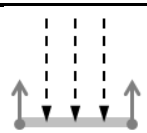
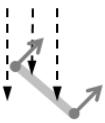
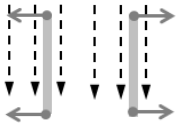
Диффузная составляющая I_d освещения рассчитывается по закону косинусов (*закон отражения Ламберта*):

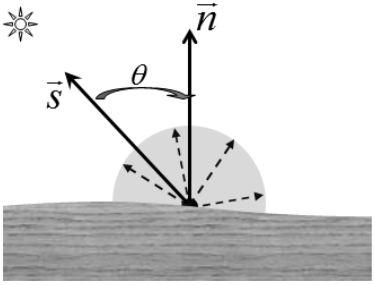
$$I_d = m_d \cdot k_d \cdot L_d,$$

где $k_d = max\{0,\ (\vec{s} \cdot \vec{n})\}$.

Таблица 6

Влияние ориентации поверхности на освещенность

Ориентация поверхности	Уровень освещенности	Угол падения	Угол θ	$\cos(\theta)$
	Максимум	90°	0°	1
	Средний	45° (0°, 90°)	45° (0°, 90°)	~0.7854 (0, 1)
		135° (90°, 180°)	−45° (−90°, 0°)	
	Нулевой	0°	±90°	0



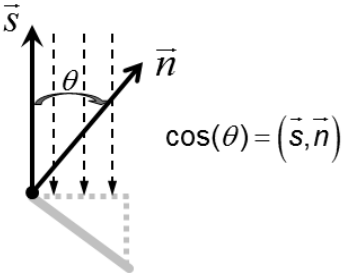


Рис.11. Диффузное рассеивание.

ЗЕРКАЛЬНОЕ ОСВЕЩЕНИЕ

При попадании на зеркальную поверхность свет отражается по закону *зеркального отражения* (угол падения равен углу отражения). Соответственно, при моделировании такого освещения необходимо учитывать не только ориентацию поверхности $\vec{n}$ и направление на источник света $\vec{s}$, но и направление на наблюдателя $\vec{v}$ (рис.12). Из геометрических соображений следует, что если косинусы углов θ и φ выходят за пределы диапазона $[0,1]$, то блика не должно быть.

Зеркальная составляющая I_s освещения рассчитывается по закону зеркального отражения *(закон отражения Фонга*):

$$I_s = m_s \cdot k_s \cdot L_s,$$

где $k_s = max\{0,\ (\vec{r} \cdot \vec{v})^b\}$, b – параметр блеска (shininess).

Исходя из геометрических представлений, вектор отражения $\vec{r}$ можно рассчитать через комбинацию векторов $\vec{n}$ и $\vec{s}$:

$$\vec{r} = 2 \cdot (\vec{n} \cdot \vec{s}) \cdot \vec{n} - \vec{s}.$$

Соответственно, скалярное произведение векторов $\vec{r}$ и $\vec{v}$ рассчитывается следующим образом:

$$(\vec{r} \cdot \vec{v}) = 2 \cdot (\vec{n} \cdot \vec{s}) \cdot (\vec{n} \cdot \vec{v}) - (\vec{s} \cdot \vec{v}).$$

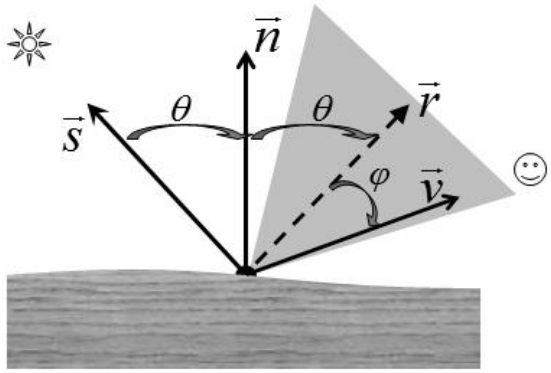


Рис.12. Зеркальное отражение с областью блика.

МОДЕЛИ ОСВЕЩЕНИЯ

Модели освещения (отражения) используются для имитации световых эффектов в компьютерной графике, когда свет аппроксимируется на основе физики света. Модели предназначены для вычисления цвета каждого пикселя или количества света, отраженного от различных поверхностей в сцене.

Цвет любой точки поверхности определяется множеством факторов:

- свойствами источников света,
- способностью объектов излучать (отражать, поглощать, пропускать, преломлять) свет,
- взаимодействием с другими объектами (подсветка, затенение, переотражения),
- цветом самой поверхности.

Поскольку при моделировании используется цветовая модель *RGB*, то в моделях освещения источник света представляется тремя независимыми однородными источниками: красным, зеленым и синим. Соответственно, у каждого из этих источников будет и свой отраженный однородный свет:

$$L = (L^R, L^G, L^B) \quad \leftrightarrow \quad I = (I^R, I^G, I^B).$$

Таким образом, *RGB*-цвет c в рассматриваемой точки поверхности, полученный по выбранной модели освещения, может быть рассчитан через суммарную интенсивность отраженного света в этой точке:

$$c = (I^R, I^G, I^B).$$

Поскольку однородные источники просчитываются независимо, то и у материала тоже есть соответствующие *RGB*-компоненты:

$$m = (m^R, m^G, m^B).$$

Существует большое количество моделей освещения, их можно по-разному классифицировать:

1. В *локальных моделях* учитывается только локальная геометрия (каждый объект рассматривает индивидуально, без взаимовлияния), такие модели еще называются *объектно-ориентированными*. В *глобальных моделях* учитываются и переотражение между поверхностями и прочие факторы (что и приводит к большим затратам вычислительных ресурсов).

2. В *физичных моделях* аппроксимируются свойства реальных объектов, такие модели учитывают особенности приповерхностной структуры поверхности (например, кожи) и поведение частиц материала (например, песка). В *эмпирических моделях* параметры могут не иметь физической интерпретации, но их правильный подбор позволяет получать весьма реалистичные изображения.

3. В *гладких моделях* поверхность рассматриваются как гладкая, (например, пластик). В *негладких моделях* поверхность рассматривается как набор гладких микрозеркал (например, у металлов).

4. В *изотропных моделях* поверхность однородна, соответственно, при повороте поверхности вокруг вектора нормали, освещение в точке не изменяется. В *анизотропных моделях* поверхность уже не однородна (например, есть царапины), в результате, при повороте поверхности вокруг вектора нормали, освещение в точке должно изменяется.

5. В *фотореалистичных моделях* качество изображения приближается к уровню фотографии. В *нефотореалистичных моделях* процесс изначально ориентированные на получение изображения, выглядящее как нарисованное человеком.

Самой распространенной моделью освещения является модель Фонга, которая относится к классу локальных, эмпирических, гладких, изотропных, фотореалистичных моделей. Помимо модели Фонга и ее простых модификаций в данной работе приведены и некоторые примеры из класса анизотропных, микрофасеточных и нефотореалистичных моделей, основанных на модификации параметров трехкомонентного освещения.

ОСНОВНЫЕ МОДЕЛИ

К основным моделям относятся модели Ламберта и Фонга, построенные на основе трехкопонентного освещения. На рис.13 представлено сравнение компонент освещения на примере сферы: фоновая компонента закрашивает сферу одним цветом, диффузная закрашивает градиентом цвета, а зеркальная только добавляет пятно блика.

У модели Фонга существует простая модификация Блинна-Фонга, а у модели Ламберта – модификация Wrap-around.

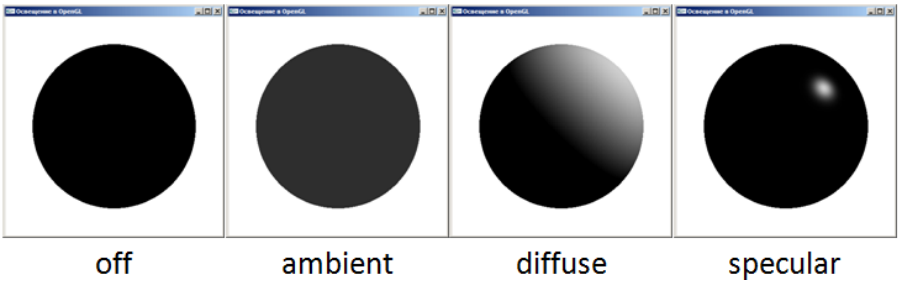


Рис.13. Фоновая, диффузная и зеркальная компоненты.

МОДЕЛЬ ФОНГА (PHONG)

В 1975 Фонг (Bui Tuong Phong) предложил модель освещения достаточно гладких поверхностей. Эта модель давно стала классикой и до сих пор остается самой популярной в компьютерной графике.

В общем виде *модель освещения Фонга* состоит из суммы фоновой, диффузной и зеркальной составляющей и имеет следующий вид:

$$I = I_a + I_d + I_s = m_a \cdot L_a + m_d \cdot k_d \cdot L_d + m_s \cdot k_s \cdot L_s.$$

МОДЕЛЬ БЛИННА-ФОНГА (BLINN-FONG)

При малых значениях параметра блеска область зеркального отражения становится довольно большой. В ситуации, когда угол между векторами $\vec{r}$ и $\vec{v}$ превышает 90°, соответствующее произведение становится отрицательным, и зеркальная компонента обнуляется. Это приводит к ситуации, когда граница зеркальной области становится ярко выраженной, что неестественно выглядит.

Для решения этой проблемы (а заодно и для и ускорения расчета зеркальной составляющей) Джеймсом Блинном (James F. Blinn) в 1977 году была предложена модификация модели освещения Фонга, получившая название *модель освещения Блинна-Фонга*, отличие заключается в способе расчета коэффициента зеркального отражения:

$$k_s = max\left\{0,\ \left(\vec{h}\cdot\vec{n}\right)^{\beta}\right\},$$

где $\vec{h}$ – вектор полупути, β – параметр блеска Блинна-Фонга.

Как можно видеть, в модели Блинна-Фонга вместо вектора $\vec{r}$ используется вектор $\vec{h}$ который показывает ориентацию площадки с максимальным отражением и рассчитывается как медианный для векторов $\vec{s}$ и $\vec{v}$:

$$\vec{h} = \frac{\vec{s}+\vec{v}}{\|\vec{s}+\vec{v}\|}.$$

Вектор $\vec{h}$ называется *вектором полупути* (halfway vector), т.к. если все три вектора $\vec{v}$, $\vec{s}$ и $\vec{n}$ лежат в одной плоскости, то угол между $\vec{h}$ и $\vec{n}$ составляет половину угла между $\vec{r}$ и $\vec{v}$.

При правильном подборе параметра β распределение зеркальной составляющей по поверхности для обеих моделей будут очень близкими.

МОДЕЛЬ ЛАМБЕРТА (LAMBERT)

В общем виде *модель освещения Ламберта* состоит из суммы фоновой и диффузной компонент:

$$I = I_a + I_d = m_a \cdot L_a + m_d \cdot k_d \cdot L_d.$$

Модель Ламберта является одной из самых простых моделей освещения. Данная модель очень часто используется как часть других моделей, поскольку практически в любой другой модели освещения можно выделить диффузную составляющую. Более-менее равномерная часть освещения (без присутствия какого-либо всплеска), как правило, будет представляться моделью Ламберта с определенными характеристиками. Данная модель может быть очень удобна для анализа свойств других моделей (за счет того, что ее легко выделить из любой модели и анализировать оставшиеся составляющие).

В частности, модель Ламберта является существенной частью модели Фонга, которая представляет собой комбинацию диффузной составляющей (модели Ламберта) и зеркальной составляющей.

МОДЕЛЬ WRAP-AROUND

Модель освещения Wrap-Around является модификация модели Ламберта, позволяя источнику освещать объект немного за границей основной тени. В этом случае коэффициент диффузии будет вычисляться с помощью дополнительного параметра $w \in [0,1]$:

$$k_d = max\left\{0,\ \frac{(\vec{s} \cdot \vec{n}) + w}{1 + w}\right\}.$$

Очевидно, что при $w = 0$ получается чистая модель Ламберта, а при $w = 1$ освещаться будет весь объект.

МИКРОФАСЕТОЧНЫЕ МОДЕЛИ

Модель освещения Ламберта хорошо работает только для сравнительно гладких поверхностей. Но существуют поверхности, составленные из выступающих микроэлементов, например, кожа состоит из микропор, а бархат – из волокон. Такие поверхности рассеивает свет во всех направлениях, но абсолютно неравномерно.

Для моделирования таких поверхностей были разработаны микрофасеточные модели (модели с микрогранями), которые предполагают, что поверхность представлена набором плоских V-образных микрограней (рис.14), ориентация нормалей которых относительно нормали к средней линии поверхности задается некоторым случайным распределением (например, Бэкмана или Гаусса), что и определяет модель освещения.

Такие модели учитывают взаимозакрывание и самозатенение микрограней. Каждая из этих микрограней вносит вклад, если она направлена вдоль вектора полупути $\vec{h}$.

Данный подход можно применять как для моделирования диффузных поверхностей (модель Орен-Наяра), так и для зеркальных (модель Кука-Торренса).

Рис.14. Гладкая и шероховатая поверхности.

МОДЕЛЬ GAUSS

В модели Гаусса считается, что микрограни подчиняются закону распределения Гаусса, коэффициент зеркальности принимает вид:

$$k_s = exp\left(-\left(\frac{(\vec{n}\cdot\vec{h})}{\mu}\right)^2\right),$$

где $\mu \in (0,1)$ – параметр гладкости поверхности.

МОДЕЛЬ OREN–NAYAR

Модель Орен-Наяра предназначена для шероховатых непрозрачных диффузных поверхностей, коэффициент диффузии имеет вид:

$$k_d = \alpha \cdot (A + B \cdot \gamma \cdot \delta),$$

где:

$\alpha = max\ \{0,\ (\vec{n}\cdot\vec{s})\},$

$\gamma = max\ \{0,\ (\vec{p}\cdot\vec{q})\},$

$\vec{p} = \vec{s} - \vec{n}\cdot(\vec{n}\cdot\vec{s}),$

$\vec{q} = \vec{v} - \vec{n}\cdot(\vec{n}\cdot\vec{v}),$

$\delta = \frac{1}{C}\cdot\sqrt{(1-D^2)\cdot(1-C^2)},$

$C = min\{(\vec{n}\cdot\vec{s}),\ (\vec{n}\cdot\vec{v})\},$

$D = max\{(\vec{n}\cdot\vec{s}),\ (\vec{n}\cdot\vec{v})\},$

$A = 1 - 0.5\cdot\sigma^2/(\sigma^2 + 0.33),$

$B = 0.45\cdot\sigma^2/(\sigma^2 + 0.09),$

$\sigma \in [0,1]$ – параметр неровности поверхности.

МОДЕЛЬ COOK-TORRANCE

Модель Кука-Торэнса предназначена для шероховатых глянцевых поверхностей (каждая микрогрань представляется идеальным зеркалом), отличие от зеркальной составляющей модели Фонга заключается в способе расчета коэффициента k_s:

$$k_s = \frac{D \cdot F \cdot G}{(\vec{v} \cdot \vec{n}) \cdot (\vec{s} \cdot \vec{n})}.$$

Коэффициент D определяет уровень шероховатости (наклон микрограни):

$$D = \frac{1}{4 \cdot \mu^2 \cdot (\vec{h} \cdot \vec{n})^4} \cdot exp\left(\frac{(\vec{h} \cdot \vec{n})^2 - 1}{m^2 \cdot (\vec{h} \cdot \vec{n})^2}\right),$$

где $\mu \in (0, 1)$ – среднеквадратичный наклон поверхности микрофасеток (шероховатость материала).

Геометрический член затухания (масштабный коэффициент) G описывает самозатенение и самоэкранирование микрограней:

$$G = min\left\{1,\ \frac{2 \cdot (\vec{h} \cdot \vec{n}) \cdot (\vec{v} \cdot \vec{n})}{(\vec{v} \cdot \vec{h})},\ \frac{2 \cdot (\vec{h} \cdot \vec{n}) \cdot (\vec{s} \cdot \vec{n})}{(\vec{v} \cdot \vec{h})}\right\}.$$

Для вычисления коэффициента Френеля F существует множество формул, но в данном случае целесообразнее применять аппроксимацию Шлика:

$$F = F_0 + [1 - (\vec{v} \cdot \vec{n})]^5 \cdot (1 - F_0),$$

где $F_0 = m_s \cdot L_s$ – количество отражаемого света при нормальном падении (перпендикулярно поверхности).

Для ряда материалов можно воспользоваться и более простой аппроксимирующей формулой:

$$F = \frac{1}{1 + (\vec{v} \cdot \vec{n})}.$$

АНИЗОТРОПНЫЕ МОДЕЛИ

При наличии мелких неровностей (как у металла), при вращении форма блика может меняться, в этом случае поверхность является *анизотропной*. При создании подобных моделей предполагается, что на поверхность равномерно нанесены прямые параллельные царапины.

МОДЕЛЬ WARD

Анизотропная модель Уорда рассчитывает коэффициент зеркальности по следующей формуле:

$$k_s = \frac{1}{\sqrt{(\vec{s}\cdot\vec{n})\cdot(\vec{v}\cdot\vec{n})}}\cdot\frac{(\vec{s}\cdot\vec{n})}{4\pi a_x a_y}\cdot exp\left(-2\frac{\left(\frac{(\vec{h}\cdot\vec{e}_x)}{a_x}\right)^2+\left(\frac{(\vec{h}\cdot\vec{e}_y)}{a_y}\right)^2}{1+(\vec{h}\cdot\vec{n})}\right),$$

где a_x и a_y – это параметры управления анизотропией (при их равенстве модель становится изотропной), $\vec{e}_x$ и $\vec{e}_y$ – это ортогональные вектора, задающие направление анизотропии.

МОДЕЛЬ MINNAERT

Эта модель была предложена для моделирования освещения Луны, но также подходит для других поверхностей, имеющих корпускулярную или губчатую поверхность, довольно хорошо она подходит для моделирования некоторых видов ткани, например вельвета.

$$k_s = (\vec{s}\cdot\vec{n})^{1+k}\big(1-(\vec{s}\cdot\vec{v})\big)^{1-k},$$

где k – параметр управления анизотропией.

НЕФОТОРЕАЛИСТИЧНЫЕ МОДЕЛИ

Данный класс моделей ориентирован на получение изображения, выглядящее как нарисованное человеком.

МОДЕЛЬ TOON SHADING

В этой модели освещенность, полученная по любой диффузной модели освещения, дискретизируется, приводя к появлению небольшого числа областей с постоянным освещением.

МОДЕЛИ GOOCH И HEMISPHERIC LIGHTING

В модели HEMISPHERIC LIGHTING диффузное освещение рассчитывается путем линейной интерполяции между цветами C_0 и C_1 стандартной функцией `lerp`:

$$I_d = lerp(C_0,\ C_1,\ \ 0.5 + (\vec{s} \cdot \vec{n})/2).$$

В модели GOOCH цвета C_0 и C_1 являются холодным и теплым оттенками одного цвета C.

МОДЕЛЬ BIDIRECTIONAL LIGHTING

Данную модель можно рассматривать как освещение сразу с двух противоположных сторон источниками с разными цветами:

$$I_d = C_0 \cdot max\{0,\ (\vec{s} \cdot \vec{n})\} + C_1 \cdot max\{0,\ -(\vec{s} \cdot \vec{n})\}.$$

МОДЕЛЬ LOMMEL-SEELIGER MODEL

$$I_d = \frac{max\{0,\ (\vec{s} \cdot \vec{n})\}}{max\{0,\ (\vec{s} \cdot \vec{n})\} + max\{0,\ (\vec{s} \cdot \vec{v})\}}.$$

АЛГОРИТМЫ ЗАТЕНЕНИЯ

Затенение является способом придания глубины изображениям (эмуляция трехмерности) путем изменения уровня темноты (светлоты) деталей изображения. Алгоритмы затенения (закраски грани) эмулируют локальное поведение света на поверхности объекта, в отличии от ресурсозатратных методов наложения теней и учета глобального освещения.

С развитием вычислительной техники появился даже специальный класс программ – шейдеры (от слова "shade" – "тень"), который был изначально предназначен для затенения граней. В дальнейшем шейдеры стали использоваться и для других задач, например, для расчета вершин граней.

Суть алгоритмов затенения заключается в закраске граней (примитивов) на основе информации о цвете (получаемом от выбранной модели освещения) в контрольных точках грани. Выбор таких точек и определяет метод затенения.

В основном используются 3 варианта затенения:

- плоское (однотонное) затенение,
- гладкое затенение по Гуро,
- гладкое затенение по Фонгу.

В отличие от плоского затенения, где цвета изменяются прерывисто на границах полигонов, при плавном (гладком) затенении цвет меняется от пикселя к пикселю, что приводит к плавному переходу цвета между двумя соседними полигонами.

Алгоритмы Гуро и Фонга используют билинейную интерполяцию. Но, в принципе, существуют и другие варианты, например, биквадратичная и сферическая линейная интерполяции.

ПЛОСКОЕ ЗАТЕНЕНИЕ

Алгоритм плоской закраски (flat shading) вызывает расчет по модели освещения только 1 раз, в одной контрольной точке, которая может быть как вершиной примитива, так и его центром. Полученный таким образом цвет применяется ко всем пикселям примитива. Этот алгоритм имеет смысл использовать, когда более продвинутые методы затенения слишком дороги с вычислительной точки зрения.

Недостатком такого подхода является резкая смена цвета у соседних граней. Зеркальная компонента тоже плохо визуализируются при плоском затенении: если контрольная вершина попала в область блика, то весь полигон принимает цвет блика, в противном случае блик теряется (даже если полигон попадает в нужную область). В связи с этим зеркальная компонента часто даже не включается в расчет плоского затенения.

Пример плоской закраски приведен на рис.15, где представлена часть сферы с двумя уровнями детализации – из 900 и 8100 четырехугольников. Отчетливо видно, что блик не принимает круглую форму.

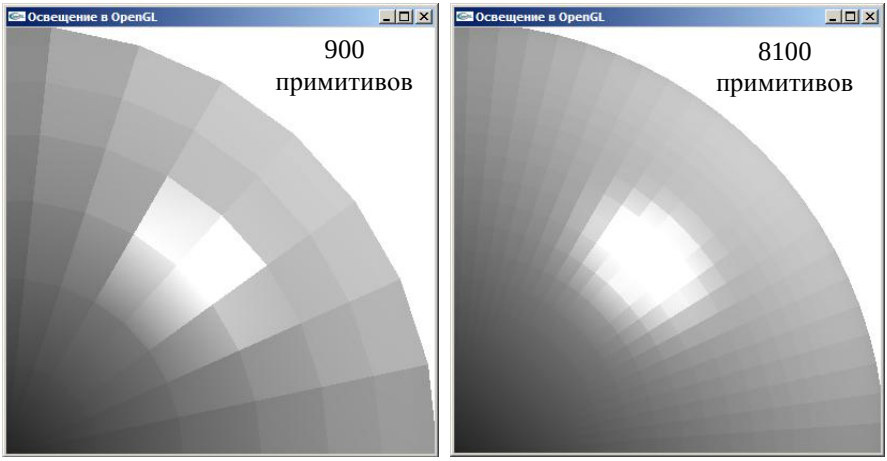


Рис.15. Затенение сферы по алгоритму плоской закраски.

ЗАТЕНЕНИЕ ПО ГУРО

В 1971 Henri Gouraud предложил алгоритм, названный в его честь. В алгоритме Гуро (Gouraud shading) контрольными точками являются вершины примитива, т.е. сначала цвет вычисляется в вершинах, а для вычисления значений в остальных точках примитива используется билинейная интерполяция.

Таким образом, переход цвета как внутри грани, так и между гранями будет плавным, но все равно возможна ситуация с потерей всего блика или его формы. Это связано с тем, что хотя бы одна контрольная точка должна попасть в область блика – только тогда пойдет интерполяция блика по примитиву. В частности, потеря блика происходит, когда блик попадает внутрь примитива, не затрагивая его вершин.

Пример закраски по Гуро приведен на рис.16, где (как и в случае с плоской закраской) представлена сфера с двумя уровнями детализации. Видно, что даже 8100 четырехугольников не позволяют получить ровный блик.

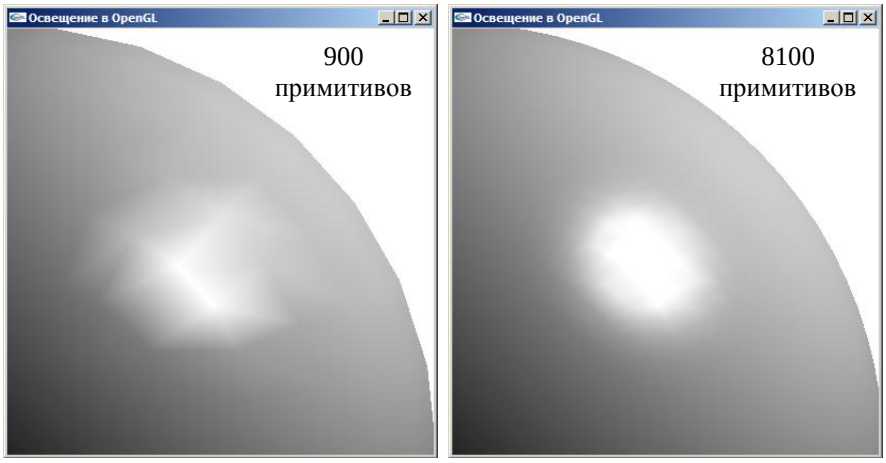


Рис.16. Затенение сферы по алгоритму закраски Гуро.

ЗАТЕНЕНИЕ ПО ФОНГУ

Алгоритм затенения (вместе с моделью освещения) Фонг впервые опубликовал в 1975. С развитием вычислительной техники подход Фонга стал базой для многих графических приложений благодаря балансу между скоростью вычислений и качеством изображения.

В отличие от затенения Гуро, которое интерполирует цвета по примитиву, в затенении Фонга (Phong shading) предполагается плавное изменение вектора нормали путем его линейной интерполяции между нормалями вершин примитива. Таким образом, контрольными точками являются все пиксели примитива.

Пример сравнения закраски по Гуро и Фонгу приведен на рис.17. Здесь на экране (размером 400×400) растеризована часть сферы (примерно 120 тысяч пикселей) с разными уровнями детализации. Здесь метод Гуро потребовал примерно в 4 раза меньше расчетов, увеличение же дискретизации сферы до 73000 примитивов (до сопоставимого качества) аналогично потребовало в 9 раза больше ресурсов, в результате метод Гуро оказался в 2.5 раз медленнее метода Фонга.

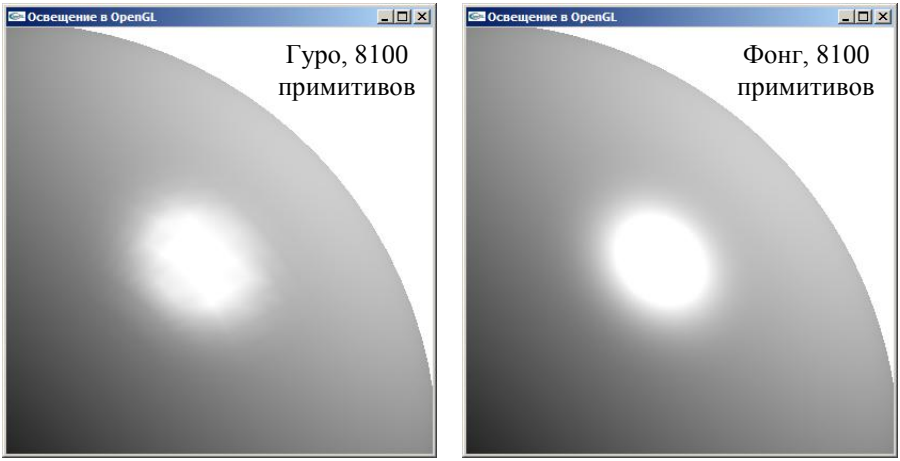


Рис.17. Затенение сферы по алгоритму закраски Фонга.

РЕАЛИЗАЦИЯ ОСВЕЩЕНИЯ В OPENGL

OpenGL, базируясь на моделях RGB и Фонга, дает разработчику весьма широкие возможности для настройки освещения. Однако при разработке даже относительно простого приложения придется немало поэкспериментировать с параметрами света, чтобы добиться приемлемых результатов.

В реальном мире любой источник освещения имеет свой спектр излучений, характеризующийся диапазоном длин излучаемых волн – фактически, это и есть *цвет источника*. Но тут есть важный нюанс: излучаемый свет может быть искажен, например, белый свет (как результат излучения во всем видимом спектре) Солнца, проходя полупрозрачную атмосферу Земли, преобразуется в желтый, красный и другие цвета.

Аналогично объекты реального мира по-разному отражают (преломляют, поглощают) приходящий от источника свет с различной длиной волны (этот уровень отражения и есть *цвет материала*), приобретая определенный цвет – *цвет объекта*. При этом *истинный цвет объекта* являете светом, отраженным от источника, излучающего во всем видимом спектре. Таким образом, при освещении синего объекта красным светом, цвет объекта получится черным (точнее, темно-серым из-за рассеянного в атмосфере света). Поскольку реальное взаимодействие между поверхностью материала и падающим светом достаточно сложное (например, при учете подповерхностых слоев водной глади), возникает проблема с адекватным выбором значений материала.

В модели освещения OpenGL все идентично: есть цвет источника, цвет материала и цвет объекта. И вследствие использования модели RGB можно независимо задавать уровни излучения и отражения для красной, зеленой и синей компонент составляющих освещенности.

По умолчанию, цвет объекта при включении освещения игнорируется, в результате для белого освещения все объекты принимают оттенки серой палитры, поскольку и цвета материала также по умолчанию задаются с одинаковыми значениями компонент. Соответственно,

для возвращения объекту изначально синего цвета необходимо увеличить значение **B**-компоненты материала, а значения компонент **R** и **G** – наоборот, ослабить.

При моделировании обычно используется белый цвет источника различной интенсивности, таким образом, цвет объектов в основном определяется свойствами материала (на цвет еще могут оказать влияние такие эффекты, как прозрачность и затуманивание).

Фоновые компоненты подбираются таким образом, чтобы вклад фоновой освещенности был невелик, но позволял слегка видеть контуры объектов. В реальности цвет бликов от источника блика совпадает с цветом самого источника света (практически не зависит от цвета материала), поэтому зеркальную компоненту материала имеет смысл задавать градиентом серого (остроту же бликов можно варьировать с помощью коэффициента блеска). Поскольку для появления блика нужны специальные условия, а фоновая компоненты мала, то большую часть результирующей освещенности вносит именно диффузная составляющая.

Замечание 1. В OpenGL у всех переменных (включая и параметры освещения) есть значения по умолчанию. В частности, при включении лампы №0 свет будет белым, при этом само пространство останется с тем цветом, каким был очищен буфер кадра (свет взаимодействует лишь с объектами, а не пространством).

Замечание 2. Переменные, которые задают связанные с цветом значения, представляют собой вектор из четырех вещественных или целочисленных компонент (модель RGBA), при этом альфа-компонента учитывается только для диффузной компоненты. Соответственно, переменные, которые задают связанные с положением или направлением значения, представляют собой аналогичный вектор в однородных координатах (если последняя компонента вектора равна нулю, тогда мстоположение рассматривается как направление).

Замечание 3. Булевские значения (`GL_TRUE` и `GL_FALSE`) параметров могут быть стандартно преобразованы в целочисленные (1 и 0) и вещественные (1.0 и 0.0) и обратно.

МОДЕЛЬ ОСВЕЩЕНИЯ И АЛГОРИТМ ЗАТЕНЕНИЯ

В OpenGL используется модель освещения Фонга, в соответствии, с которой цвет точки определяется несколькими факторами:

- свойствами материала (включая и цвет объекта);
- свойствами текстуры (здесь не рассматриваются);
- вектором нормали;
- параметрами источника света;
- положением наблюдателя.

При включении нескольких источников света результирующее значение может превысить свой максимум (1.0 для вещественного числа), в этом случае происходит отсечение значения (все, что больше 1.0 обрезается до 1.0), что эквивалентно эффекту засветки при фотографировании.

Вследствие использования модель Фонга в качестве модели освещения, для учета освещения используются три стандартных компоненты освещенности:

- фоновая (ambient);
- диффузная (diffuse);
- зеркальная (specular).

Поскольку расчет теней является нетривиально задачей, а модель Фонга – локальная, в OpenGL реализован лишь простейший вариант, когда освещение (затенение) рассчитывается независимо для каждого полигона. Сама же закраска граней выполняется по алгоритму Гуро.

Одной из характерных особенностей освещения в OpenGL является то, что оно может быть двухсторонним (освещать не только лицевую, но и нелицевую поверхность), однако такой режим приводит к удвоению расходов на просчет освещения.

Замечание. Для корректного расчета освещенности в точке необходимо использовать единичные нормали.

ТИПЫ ИСТОЧНИКОВ ОСВЕЩЕНИЯ

Поскольку используется модель освещения Фонга, то источники света (лампы) характеризуется своей *интенсивностью*, которая состоит из фоновой, диффузной и зеркальной составляющей для каждой из компонент цветовой модели RGB (фактически, интенсивность и есть цвет лампы). Всего в OpenGL реализовано 5 видов источников освещения, они приведены в табл.7.

Таблица 7

Типы источников света

Тип источника	Описание освещения	Генерация теней	Параметры задания
Глобальный фоновый	Равномерное освещение всего пространства	нет	Интенсивность (только фоновая)
Эмиссионный	Самоподсветка	нет	Интенсивность
Точечный	Равномерное излучение по всем направлениям из точки	есть	Интенсивность Позиция
Прожекторный	Излучение конуса света по направлению из точки	есть	Интенсивность Позиция Направление Угол расхождения Распределение
Направленный	Равномерное излучение в одном направлении из бесконечности	есть	Интенсивность Направление

ЗАДАНИЕ ОСВЕЩЕНИЯ

Для задания освещения в сцене требуется:

1) включить режим освещения;
2) настроить модель освещения;
3) задать свойства источников света;
4) задать свойства материала поверхности граней;
5) задать нормали к поверхности для каждой вершины грани.

Для задания параметров освещения могут быть использованы команды как с целочисленными параметрами, так и с вещественными. Соответственно, преобразование значений вещественного вектора (заданного в формате RGBA) в целочисленный выполняется путем линейной интерполяции из диапазона $[0, 1]$ в диапазон $[0, maxint]$, в случае другого диапазона соответствующее целочисленное значение не определено. Для прочих вещественных чисел перевод осуществляется путем округления до ближайшего целого.

ВКЛЮЧЕНИЕ РЕЖИМА ОСВЕЩЕНИЯ

По умолчанию режим освещения выключен в целях экономии ресурсов. Включение/выключение режима освещения выполняется командами `glEnable/glDisable` с указанием идентификатора `GL_LIGHTING`. До этого момента установка параметров освещения будет проигнорирована.

Всего в библиотеке доступно к использованию до 8 источников света, которые также включаются/выключаются командами `glEnable/glDisable` с указанием соответствующих констант-идентификаторов:

`GL_LIGHT0, GL_LIGHT1,...,GL_LIGHT7.`

ЗАДАНИЕ МОДЕЛИ ОСВЕЩЕНИЯ

Базовая модель освещения задается командой `glLightModel`:

```
void glLightModeli  (GLenum pname, GLint param ),
void glLightModelf  (GLenum pname, GLfloat param ),
void glLightModeliv (GLenum pname, const GLint *params ),
void glLightModelfv (GLenum pname, const GLfloat *params ).
```

Здесь первым параметром (`pname`) указывается идентификатор параметра модели освещения, а вторым (`param`) – его новое значение. Краткое описание параметров команды приведено в табл.8. Очевидно, что с помощью двух первых вариантов команды невозможно установить вектор фоновой компоненты.

С помощью данной команды можно установить следующие режимы расчетов освещения:

1. Включить/выключить режим корректного расчета углов зеркального отражения (по умолчанию считается, что наблюдатель находится в бесконечно удаленной точке – в этом случае углы всегда будет параллельным оси OZ).

2. Включить/выключить режим учета нелицевых граней (по умолчанию расчет освещения выполняется лишь для лицевых граней, поскольку нелицевые грани как бы не должны быть видны наблюдателю).

3. Включить/выключить режим отдельного расчета зеркальной компоненты с целью последующего наложения на текстуру (по умолчанию, зеркальная составляющая цвета вычисляется вместе с фоновой и диффузной).

4. Задать значение интенсивности глобального фонового освещения, которое не является включаемым источником света.

Таблица 8

Параметры функции glLightModel

Параметр pname	Описание	Параметр param (значение по умолчанию)
GL_LIGHT_MODEL_LOCAL_VIEWER	Корректный расчет зеркальных углов	GL_FALSE или 0.0
GL_LIGHT_MODEL_TWO_SIDE	Учет нелицевых граней	GL_FALSE или 0.0
GL_LIGHT_MODEL_COLOR_CONTROL	Отдельный расчет зеркальной компоненты	GL_SINGLE_COLOR
GL_LIGHT_MODEL_AMBIENT	Значение глобальной фоновой интенсивности	(0.2, 0.2, 0.2, 1.0)

ЗАДАНИЕ ПАРАМЕТРОВ ИСТОЧНИКА СВЕТА

Параметры источника света задаются командой `glLight`:

```
void glLighti (GLenum light, GLenum pname, GLint   param),
void glLightf (GLenum light, GLenum pname, GLfloat param),
void glLightiv(GLenum light, GLenum pname, GLint   *param),
void glLightfv(GLenum light, GLenum pname, GLfloat *param).
```

Здесь первым параметром (`light`) указывается идентификатор источника (`GL_LIGHT0, GL_LIGHT1,...,GL_LIGHT7`), вторым и третьим – идентификатор параметра освещения (`pname`) и его новое значение (`param`). Краткое описание параметров `pname` и `param` приведено в табл.9. Очевидно, что с помощью двух первых команд невозможно получить векторные значения параметров.

Каждый источник стандартно задается своей позицией и тремя компонентами излучения света (фоновой, диффузной и зеркальной). Также существуют дополнительные наборы параметров для задания прожекторного варианта и затухания с расстоянием.

По умолчанию источник является направленным (последняя компонента вектора позиции равна 0), соответственно, первые 3 компоненты этого вектора задают направление светового излучения.

Для получения точеного источника необходимо установить последнюю компоненту вектора позиции в 1, в этом случае первые 3 компоненты данного вектора задают позицию источника.

Прожекторный источник получается, когда последняя компонента вектора позиции установлена в 1 и задан полуугол излучения (угол отсечения) в пределах [0°, 90°]. Полуугол может принять еще и невходящее в указанный диапазон значение 180° – в этом случае прожектор фактически становится точечным источником, светящим во все стороны (на все 360°). Также у прожектора можно переключить фокусировку прожекторного пятна с константного на экспоненциальный.

Замечание. Позиция и направление освещения задаются в текущей модельной системе координат, например, при повороте сцены изменится и позиция источников.

Таблица 9

Параметры функции glLight

Параметр pname	Описание	Параметр param (по умолчанию)
GL_AMBIENT	Фоновая составляющая	$(0, 0, 0, 1)$
GL_DIFFUSE	Диффузная составляющая	$(1,1,1, 1)$ для LIGHT0
GL_SPECULAR	Зеркальная составляющая	$(0, 0, 0, 1)$ для остальных
GL_POSITION	Координаты или направление	$(0, 0, 1, 0)$ (направленный)
GL_SPOT_EXPONENT	Прожектор: фокусировка пятна, [0,128]	0
GL_SPOT_CUTOFF	Прожектор: полуугол излучения, [0,90] и 180	180 (как точечный)
GL_SPOT_DIRECTION	Прожектор: направление	$(0, 0, -1, 0)$
GL_CONSTANT_ATTENUATION	Коэффициенты затухания	$k_{const} = 1$
GL_LINEAR_ATTENUATION		$k_{linear} = 0$
GL_QUADRATIC_ATTENUATION		$k_{quadr} = 0$

ЗАДАНИЕ МАТЕРИАЛА

При расчете освещения примитивы просчитывается индивидуально. Соответственно, для каждого примитива можно задать *материал* – набор параметров для учета света, приходящего от источников освещения.

Свойства материала определяют фоновую, диффузную, зеркальную и эмиссионную составляющую материала и смогут быть заданы при помощи функции `glMaterial`:

```
void glMateriali ( GLenum face, GLenum pname, GLfloat param),
void glMaterialf ( GLenum face, GLenum pname, GLfloat param),
void glMaterialiv ( GLenum face, GLenum pname, GLfloat param),
void glMaterialfv ( GLenum face, GLenum pname, GLfloat param).
```

Параметр `face` определяет грань, для которой устанавливаются свойства, может принимать одно из следующих значений:

- ✓ `GL_FRONT` (лицевая грань);
- ✓ `GL_BACK` (нелицевая грань);
- ✓ `GL_FRONT_AND_BACK` (обе грани).

Краткое описание параметров `pname` и `param` приведено в табл.10. Очевидно, что с помощью двух первых команд невозможно задать векторные значения параметров. Значения параметров по умолчанию одинаковы как для лицевых, так и не для лицевых граней, при этом нелицевые грани (режим `GL_BACK`) обрабатываются лишь при включении двустороннего освещения.

Коэффициент блеска определяет "шероховатость" материала, является степенью косинуса в формуле зеркального отражения. Коэффициент влияет на площадь и яркость блика, для металлической поверхности его значение должно быть больше.

Как уже говорилось, подбор значений параметров материала является нетривиальной задачей, не может быть однозначно выполнен. Пример таких наборов значений приведен в табл.11.

По умолчанию, при включении освещения игнорируется текущий цвет, задаваемый командой glColor. Соответственно, цвет объекта задается свойствами материала.

Но, включив управление свойством материала с помощью текущего цвета, можно изменять одну из характеристик материала командой glColor. Этот подход может быть полезен при частом изменении одной из компонент материала.

Включение/выключение данного режима стандартно выполняется командами glEnable/glDisable с параметром GL_COLOR_MATERIAL.

Выбор компоненты материала, которую требуется менять таким образом, выполняется командой glColorMaterial:

```
void glColorMaterial ( GLenum face,  GLenum mode ).
```

Здесь параметр face определяет грань, для которой устанавливаются свойства, может принимать одно из следующих значений:

- ✓ GL_FRONT (лицевая грань);
- ✓ GL_BACK (нелицевая грань);
- ✓ GL_FRONT_AND_BACK (обе грани, значение по умолчанию).

Параметр mode определяет изменяемую компоненту материала и может принимать следующие значения:

- ✓ GL_EMISSION;
- ✓ GL_AMBIENT;
- ✓ GL_DIFFUSE;
- ✓ GL_SPECULAR;
- ✓ GL_AMBIENT_AND_DIFFUSE (значение по умолчанию).

Таблица 10

Параметры функции glMaterial

Параметр pname	Описание	Параметр param (значение по умолчанию)
GL_AMBIENT	Фоновая компонента	(0.2, 0.2, 0.2, 1)
GL_DIFFUSE	Диффузная компонента	(0.8, 0.8, 0.8, 1)
GL_SPECULAR	Зеркальная компонента	(0, 0, 0, 1)
GL_EMISSION	Эмиссионная компонента	(0, 0, 0, 1)
GL_SHININESS	Коэффициент блеска, [0,128]	0
GL_AMBIENT_AND_DIFFUSE	Аналогичен двойному вызову функции glMaterial с одинаковыми значениями параметров для GL_AMBIENT и GL_DIFFUSE	

Таблица 11

Пример наборов значений параметров материала

Название	Фоновая компонента			Диффузная компонента			Диффузная компонента			Блеск
Изумруд	0.0215	0.1745	0.0215	0.0756	0.6142	0.0756	0.6330	0.7278	0.6330	0.6
Рубин	0.1745	0.0117	0.0117	0.6142	0.0413	0.0413	0.7278	0.6269	0.6269	0.6
Обсидиан	0.0537	0.0500	0.0662	0.1827	0.1700	0.2252	0.3327	0.3286	0.3464	0.3
Жемчуг	0.2500	0.2072	0.2072	1	0.8290	0.8290	0.2966	0.2966	0.2966	0.088
Медь	0.1912	0.0735	0.0225	0.7038	0.2704	0.0828	0.2567	0.1376	0.0860	0.1
Золото	0.2472	0.1995	0.0745	0.7516	0.6064	0.2264	0.6282	0.5558	0.3660	0.4
Серебро	0.1922	0.1922	0.1922	0.5075	0.5075	0.5075	0.5082	0.5082	0.5082	0.4
Бронза	0.2125	0.1275	0.0540	0.7140	0.4284	0.1814	0.3935	0.2719	0.1667	0.2
Пластик черный	0	0	0	0.01	0.01	0.01	0.5	0.5	0.5	0.25
Пластик белый	0	0	0	0.055	0.055	0.055	0.7	0.7	0.7	0.25
Пластик желтый	0	0	0	0.5	0.5	0	0.6	0.6	0.5	0.25
Пластик красный	0	0	0	0.5	0	0	0.7	0.6	0.6	0.25
Пластик голубой	0	0.1	0.06	0	0.5098	0.5098	0.5019	0.5019	0.5019	0.25

РАСЧЕТ НОРМАЛЕЙ

Для правильного освещения нормали поверхности должны иметь единичную длину и задаваться непосредственно перед заданием вершины (будучи атрибутом вершины, нормаль может задаваться внутри операторных скобок `glBegin/glEnd`). При необходимости нормаль можно задавать один раз перед группой вершин, тогда у всей группы нормаль будет одинакова.

Поскольку вершины задаются после выполнения модельных преобразований, нормали в этих вершинах изменятся аналогично. Восстановление направления нормали автоматически выполняется библиотекой, однако по умолчанию режим автоматического нормирования векторов нормалей выключен.

Для корректного расчета освещения требуется единичная длина нормалей, но, в принципе, возможна работа и с неединичными нормалями – главное, чтобы длина у них была одинаковой. В общем случае, процедура нормализации заключается в следующем: сначала вычисляется длина вектора нормали, а затем каждая компонента вектора делится на это число. Данная процедура вычислительно затратна, поэтому в ряде случаев, например, при работе со сферами, имеет смысл вручную нормировать (все компоненты поделить на радиус сферы), не прибегая к процедуре автонормализации.

Включение/выключение режима автонормирования нормалей стандартно выполняется командой `glEnable`/`glDisable` с параметром `GL_NORMALIZE`. На рис.18 представлен пример растеризации сферы со включенным и выключенным режимами нормализации, длина задаваемых нормалей равнялась радиусу.

Но в OpenGL также предусмотрен и альтернативный (ускоренный) режим автонормализации – `GL_RESCALE_NORMAL`, который также включается и выключается командами `glEnable` и `glDisable`. В этом случае каждая компонента вектора будет умножена на одно и то же число, извлеченное из матрицы модельных преобразований. Данная процедура работает корректно только в случае, когда нормали масштабировались равномерно и изначально имели единичную длину.

Поскольку OpenGL позволяет задавать целочисленные нормали, значение компонент вектора автоматически масштабируется в стандартный вещественный диапазон [– 1,1].

Вектор нормали задается командой `glNormal3` (в зависимости от типа используемых данных):

```
void glNormal3b ( GLbyte nx, GLbyte ny, GLbyte nz );
void glNormal3bv ( const GLbyte *v );
void glNormal3d ( GLdouble nx, GLdouble ny, GLdouble nz );
void glNormal3dv ( const GLdouble *v );
void glNormal3f (  GLfloat nx, GLfloat ny, GLfloat nz );
void glNormal3fv ( const GLfloat *v );
void glNormal3i ( GLint nx, GLint ny, GLint nz );
void glNormal3iv ( const GLint *v );
void glNormal3s ( GLshort nx, GLshort ny, GLshort nz );
void glNormal3sv ( const GLshort  *v ).
```

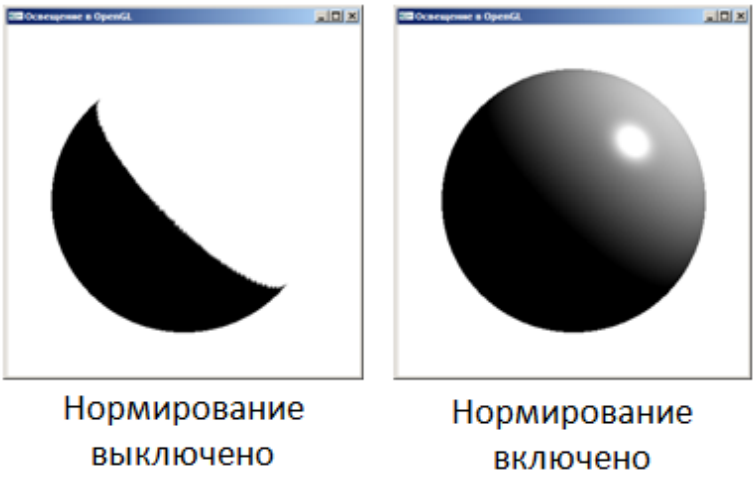


Рис.18. Влияние нормирования нормалей.

ПОЛУЧЕНИЕ СТАТУСА ПАРАМЕТРОВ ОСВЕЩЕНИЯ

Поскольку библиотека OpenGL является конечным автоматом, состояние библиотеки в любой момент времени является определенным. Каждый параметр библиотеки инициализирован значением по умолчанию и может быть изменен и проверен соответствующими командами.

Примерами таких параметров являются рассмотренные ранее параметры освещения. Проверка режимов на включенность и получение текущих значений параметров выполняется с помощью частично дублирующих друг друга команд `glIsEnabled`, `glGet`, `glGetLight` и `glGetMaterial`:

```
GLboolean glIsEnabled ( GLenum cap ),

void glGetBooleanv ( GLenum pname, GLboolean *params ),
void glGetIntegerv ( GLenum pname, GLint     *params ),
void glGetFloatv   ( GLenum pname, GLfloat   *params ),
void glGetDoublev  ( GLenum pname, GLfouble  *params ),

void glGetLightiv ( GLenum light,
                    GLenum pname, GLint   *params ),
void glGetLightfv ( GLenum light,
                    GLenum pname, GLfloat *params ),

void glGetMaterialiv( GLenum face,
                      GLenum pname,  GLint   *params ),
void glGetMaterialfv( GLenum  face,
                      GLenum  pname, GLfloat *params ),
```

где:

`cap` – идентификатор режима освещения,
`light` – идентификатор источника света,
`face` – идентификатор поверхности материала,
`pname` – идентификатор параметра,
`params` – значение материала (адрес переменной или вектора).

СТАТУС МОДЕЛИ ОСВЕЩЕНИЯ

Узнать текущее значение параметров модели освещения можно путем вызова команды `glGet`, краткое описание параметров которой приведено в табл.12. Как можно видеть, с помощью данной команды можно получить и статус освещения.

Таблица 12

Параметры функции glGet

Параметр `pname`	Параметр `param`
`GL_LIGHTING`	Булевское значение, показывающее включенность режима освещения
`GL_LIGHT0 ÷ GL_LIGHT7`	Булевское значение, показывающее включенность указанного источника света
`GL_MAX_LIGHTS`	Целочисленное значение, показывающее максимальное количество источников света
`GL_LIGHT_MODEL_AMBIENT`	Вектор в формате RGBA, содержащий значение глобальной фоновой компоненты
`GL_LIGHT_MODEL_LOCAL_VIEWER`	Булевское значение, показывающее включенность режима упрощенного расчета углов зеркального отражения
`GL_LIGHT_MODEL_TWO_SIDE`	Булевское значение, показывающее включенность режима учета нелицевых граней

СТАТУС ИСТОЧНИКА СВЕТА

Узнать текущее значение параметров источника света можно путем вызова команды `glGetLight`, параметры которой аналогичны параметрам `glLight`. В табл.13 приведено краткое описание параметров `pname` и `param`. При этом значения позиции и направления лампы может не совпадать с исходными вследствие использования модельно-видовых преобразований.

Таблица 13

Параметры функции glGetLight

Параметр `pname`	Параметр `param`
GL_AMBIENT	Вектор в формате RGBA, содержащий значение фоновой компоненты
GL_DIFFUSE	Вектор в формате RGBA, содержащий значение диффузной компоненты
GL_SPECULAR	Вектор в формате RGBA, содержащий значение зеркальной компоненты
GL_POSITION	Вектор в формате XYZW, содержащий значение позиции лампы
GL_SPOT_EXPONENT	Число, показывающее значение фокусировки пятна прожектора
GL_SPOT_CUTOFF	Число, показывающее значение угла прожектора
GL_SPOT_DIRECTION	Вектор в формате XYZW, содержащий значение направления прожектора
GL_CONSTANT_ATTENUATION	Число, показывающее значение постоянного фактора затухания
GL_LINEAR_ATTENUATION	Число, показывающее значение линейного фактора затухания
GL_QUADRATIC_ATTENUATION	Число, показывающее значение кубического фактора затухания

СТАТУС МАТЕРИАЛА

Узнать текущее значение параметров материала можно путем вызова команды `glGetMaterial`, параметры которой аналогичны параметрам `glMaterial`. В табл.14 приведено краткое описание параметров `pname` и `param`.

Проверить включенность режима учета цвета объекта при расчете освещения можно путем следующего вызова:

```
glIsEnabled ( GL_COLOR_MATERIAL ).
```

Таблица 14

Параметры функции glGetMaterial

Параметр `pname`	Параметр `param`
`GL_AMBIENT`	Вектор в формате RGBA, содержащий значение фоновой компоненты
`GL_DIFFUSE`	Вектор в формате RGBA, содержащий значение диффузной компоненты
`GL_SPECULAR`	Вектор в формате RGBA, содержащий значение зеркальной компоненты
`GL_EMISSION`	Вектор в формате RGBA, содержащий значение эмиссионной компоненты
`GL_SHININESS`	Число, показывающее значение шероховатости материала

СТАТУС ОСВЕЩЕНИЯ И АВТОНОРМИРОВАНИЯ

Проверку включенности освещения и ламп удобно выполнять командой `glIsEnabled`, указав в качестве значения параметра `cap` идентификатор освещенности (`GL_LIGHTING`) или лампы (`GL_LIGHT0, GL_LIGHT1,...,GL_LIGHT7`). Данная функция возвращает `GL_TRUE`, если включен указанный режим, иначе возвращает `GL_FALSE`.

С помощью этой же функции можно проверить и включения режима автонормировки нормалей:

```
glIsEnabled ( GL_NORMALIZE ).
```

ПРИМЕР РЕАЛИЗАЦИИ ОСВЕЩЕНИЯ

Для того, чтобы уже на практике рассмотреть особенности реализации освещения в OpenGL, необходимо создать соответствующий каркас графического приложения. Удобнее всего это сделать с использованием библиотеки GLUT. Пример кода приведен Приложении А. Здесь функция `LightOn` включает нужный источник света, а `DrawObj` вызывает функцию отрисовки объекта (сферы или плоскости), их код элементарен.

Функции рисования плоскости (в виде квадрата, состоящего из множества малых квадратов) и сферы приведены соответственно в Приложении Б и Приложении В. Оба объекта отрисовываются четырехугольниками. В принципе, для задания квадрата достаточно и одного четырехугольника, но тогда пропадает эффект бликования поверхности.

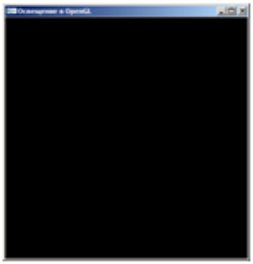
Функции включения/выключения и настройки различных ламп приведены в Приложении Г. У нулевой лампы оставлена только фоновая составляющая, а у других ламп – диффузная и зеркальная.

На рис.19 – рис.26 приведен ряд примеров, демонстрирующих влияние различных источников света и их параметров на освещение сферы и плоскости.

На рис.19 и рис.23 для плоскости и сферы показаны примеры использования различных типов источников света. Видно, что одновременное включение двух несогласованных друг с другом источников приводит к эффекту засвечивания участка поверхности, где их интенсивность максимальна.

На рис.20 и рис.24 показано влияние эффекта угасания луча с расстоянием для точечного источника с подобранными параметрами затухания.

На рис.21 и рис.25 продемонстрировано влияние угла распространения света для прожектора, а на рис.22 и рис.26 – влияние фокусировки (распределения интенсивности внутри пятна) прожектора.



Освещение
выключено

Фоновый
источник

Направленный
источник

Прожекторный
источник

Точечный
источник

Направленный
и прожекторный

Рис.19. Освещение грани различными лампами.

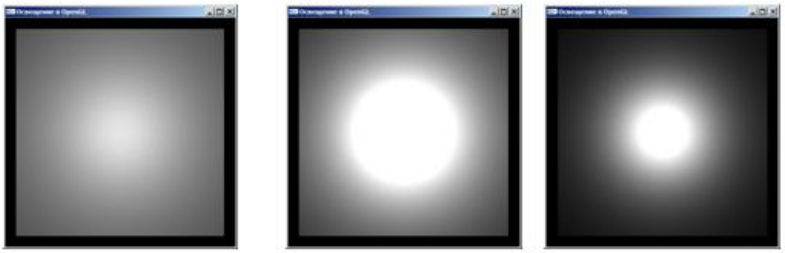


Рис.20. Освещение грани точечной лампой с затуханием.

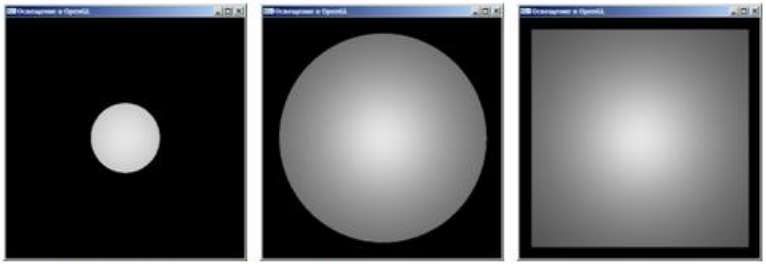


Рис.21. Освещение грани прожектором с разными углами.

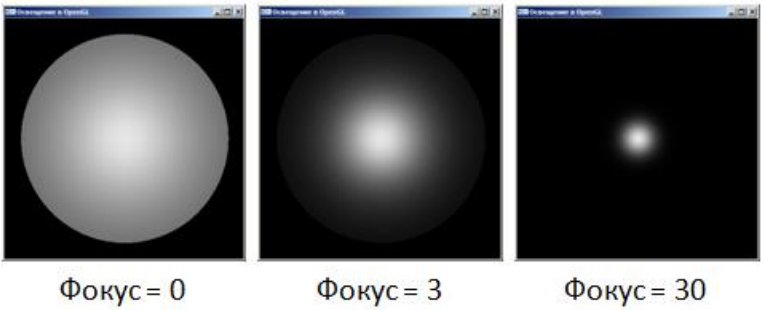
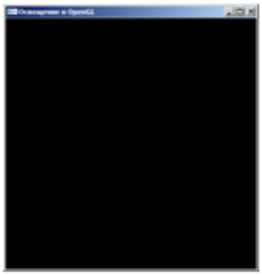
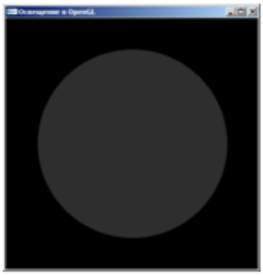


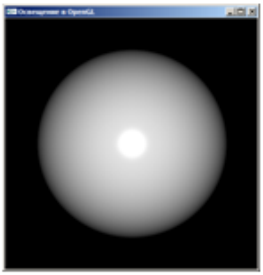
Рис.22. Освещение грани прожектором с разной фокусировкой.



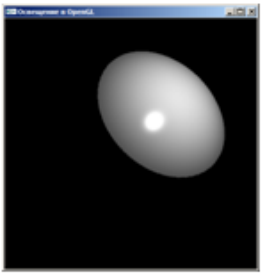
Освещение
выключено



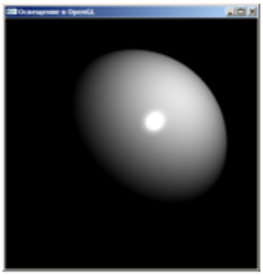
Фоновый
источник



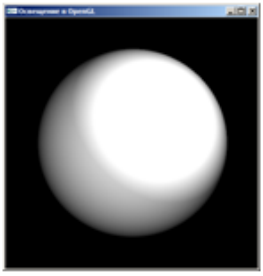
Направленный
источник



Прожекторный
источник



Точечный
источник



Направленный
и точечный

Рис.23. Освещение сферы различными лампами.

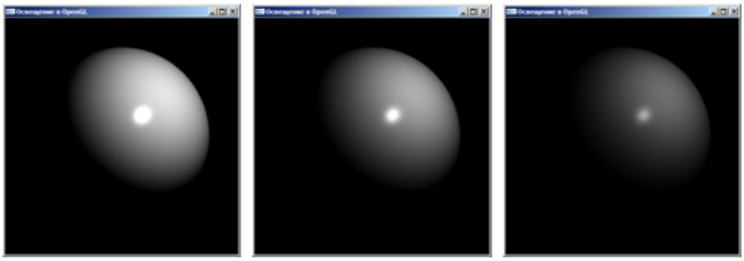


Рис.24. Освещение сферы с точечной лампой затуханием.

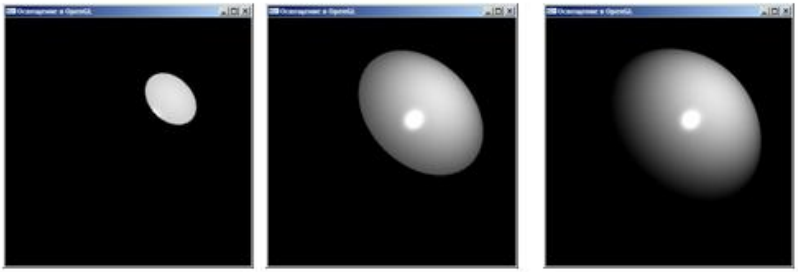


Рис.25. Освещение сферы прожектором с разными углами.

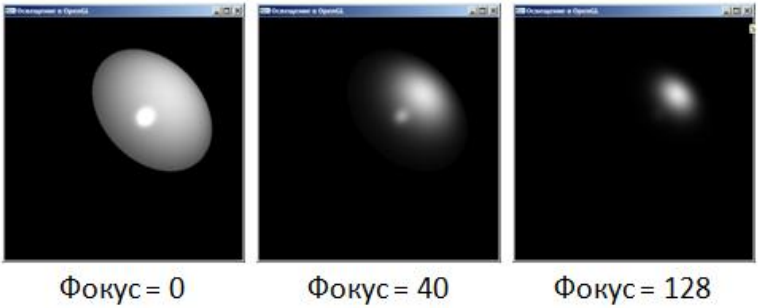


Рис.26. Освещение сферы прожектором с разной фокусировкой.

ПРИЛОЖЕНИЕ А.
КАРСКАС ПРИЛОЖЕНИЯ

```
#include "glut.h"

/* разрешение окна */
GLint Width = 400;
GLint Height = 400;

/* вектора для свойств освещения и материала */
float nul[] = { 0, 0, 0, 1 };
float one[] = { 1, 1, 1, 1 };
float min[] = { 0.1, 0.1, 0.1, 1 };
float max[] = { 0.9, 0.9, 0.9, 1 };


/* Функция инициализации режимов двойной буферизации и
освещения */
void Init(void)
{
 if( !glIsEnabled(GL_DEPTH_TEST) ) glEnable(GL_DEPTH_TEST);
 if( !glIsEnabled(GL_LIGHTING) ) glEnable(GL_LIGHTING);
 if(!glIsEnabled(GL_NORMALIZE)) glEnable(GL_NORMALIZE);

 /* Отключение глобального освещения */
 glLightModelfv( GL_LIGHT_MODEL_AMBIENT, nul);
}
```

```
/* Функция вывода на экран */
void Display(void)
{
 glClearColor(0, 0, 0, 0);
 glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
 Init();
 LightOn();
 DrawObj();
 glFinish();
}
/* Функция изменения размеров окна и вьюпорта*/
void Reshape(GLint w, GLint h)
{
 Width = w;    Height = h;
 glViewport(0, 0, w, h);
 glMatrixMode(GL_PROJECTION); glLoadIdentity();
 glOrtho(-w/2, w/2, -h/2, h/2, -10000, 10000);
 glMatrixMode(GL_MODELVIEW); glLoadIdentity();
}
/* Головная программа */
void main(int argc, char *argv[])
{
 glutInit(&argc, argv);
 glutInitDisplayMode(GLUT_RGB | GLUT_DEPTH);
 glutInitWindowSize(Width, Height);
 glutCreateWindow("Освещение");
 glutDisplayFunc(Display);
 glutReshapeFunc(Reshape);
 glutMainLoop();
}
```

ПРИЛОЖЕНИЕ Б.
ФУНКЦИЯ РИСОВАНИЯ ПЛОСКОСТИ

```
void DrawPlane()
{
	glMaterialfv(GL_FRONT, GL_AMBIENT, max);
	glMaterialfv(GL_FRONT, GL_DIFFUSE, max);
	glMaterialfv(GL_FRONT, GL_SPECULAR, nul);
	glMateriali(GL_FRONT, GL_SHININESS, 100);

	float s = 360;
	float s0 = -s/2, s1 = -s0;
	float ds = s / 1000;

	glNormal3f( 0, 0, 1 );
	glColor3f(1, 1, 1);

	glBegin(GL_QUADS);
	for (float x = s0; x < s1; x += ds)
	{
		for (float y = s0; y < s1; y += ds)
		{
			glVertex3f(x     , y     , 0);
			glVertex3f(x     , y + ds, 0);
			glVertex3f(x + ds, y + ds, 0);
			glVertex3f(x + ds, y     , 0);
		}
	}
	glEnd();
}
```

ПРИЛОЖЕНИЕ В.
ФУНКЦИЯ РИСОВАНИЯ СФЕРЫ

```
float get_sin(float fi)
{
 const float pi = 3.141592;
 float rad = fi * pi / 180;
 return ( sin(rad) );
}

float get_cos(float fi)
{
 const float pi = 3.141592;
 float rad = fi * pi / 180;
 return ( cos(rad) );
}

void DrawSphere()
{
 glMaterialfv(GL_FRONT, GL_AMBIENT, max);
 glMaterialfv(GL_FRONT, GL_DIFFUSE, max);
 glMaterialfv(GL_FRONT, GL_SPECULAR, max);
 glMateriali(GL_FRONT, GL_SHININESS, 100);

 /* сдвиг сферы вглубь */
 glLoadIdentity();
 glTranslatef(0,0,-300);

 float s = 500;
 float r = 150;
 float d = 360 / s;
```

```
 glBegin(GL_QUADS);
 for( float te = 0; te < 180; te += d )
  {
       float t0 = te;
       float t1 = te + d;
       float z0 = r * get_cos(t0);
       float z1 = r * get_cos(t1);

       for( float fi = 0; fi < 360; fi += d )
       {
        float f0 = fi;
        float f1 = fi + d;

        { // верхняя полусфера
         float x0 = r * get_cos(f0) * get_sin(t1);
         float x1 = r * get_cos(f1) * get_sin(t1);
         float y0 = r * get_sin(f0) * get_sin(t1);
         float y1 = r * get_sin(f1) * get_sin(t1);
         glNormal3f(x0, y0, z1);    glVertex3f(x0, y0, z1);
         glNormal3f(x1, y1, z1);    glVertex3f(x1, y1, z1);
        }

        { // нижняя полусфера
         float x1 = r * get_cos(f0) * get_sin(t0);
         float x0 = r * get_cos(f1) * get_sin(t0);
         float y1 = r * get_sin(f0) * get_sin(t0);
         float y0 = r * get_sin(f1) * get_sin(t0);
         glNormal3f(x0, y0, z0);    glVertex3f(x0, y0, z0);
         glNormal3f(x1, y1, z0);    glVertex3f(x1, y1, z0);
        }

       }
 }
glEnd();
}
```

ПРИЛОЖЕНИЕ Г.
ФУНКЦИИ ЗАДАНИЯ ЛАМП

```
/* Включение лампы №0: фоновая подсветка */
void SetLamp0(bool on)
{
	glLightfv(GL_LIGHT0, GL_AMBIENT , min);
	glLightfv(GL_LIGHT0, GL_DIFFUSE , nul);
	glLightfv(GL_LIGHT0, GL_SPECULAR, nul);
	if(on) glEnable(GL_LIGHT0); else glDisable(GL_LIGHT0);
}
/* Включение лампы №1: направленный свет */
void SetLamp1(bool on)
{
	float  dir[] = { 0, 0, 1, 0 };
	glLightfv(GL_LIGHT1, GL_AMBIENT , nul);
	glLightfv(GL_LIGHT1, GL_DIFFUSE , one);
	glLightfv(GL_LIGHT1, GL_SPECULAR, one);
	glLightfv(GL_LIGHT1, GL_POSITION, dir);
	if(on) glEnable(GL_LIGHT1); else glDisable(GL_LIGHT1);
}
/* Включение лампы №2: точечный свет */
void SetLamp2(bool on)
{
	float  pos[] = { 200, 200, 300, 1 };
	glLightfv(GL_LIGHT2, GL_AMBIENT , nul);
	glLightfv(GL_LIGHT2, GL_DIFFUSE , one);
	glLightfv(GL_LIGHT2, GL_SPECULAR, one);
	glLightfv(GL_LIGHT2, GL_POSITION, pos);
	if(on) glEnable(GL_LIGHT2); else glDisable(GL_LIGHT2);
}
```

```
/* Включение лампы №3: прожекторынй свет */
void SetLamp3(bool on)
{
	float  pos[] = { 200, 200, 300, 1 };
	float  dir[] = { -200, -200, -300, 0 };

	glLightfv(GL_LIGHT3, GL_AMBIENT , nul);
	glLightfv(GL_LIGHT3, GL_DIFFUSE , one);
	glLightfv(GL_LIGHT3, GL_SPECULAR, one);
	glLightfv(GL_LIGHT3, GL_POSITION, pos);

	glLighti(GL_LIGHT3, GL_SPOT_EXPONENT, 30);
	glLighti(GL_LIGHT3, GL_SPOT_CUTOFF, 20);
	glLightfv(GL_LIGHT3, GL_SPOT_DIRECTION, dir);
	if(on) glEnable(GL_LIGHT3); else glDisable(GL_LIGHT3);
}

/* Включение лампы №4: точечный свет с затуханием */
void SetLamp2f(bool on)
{
	float  pos[] = { 200, 200, 300, 1 };

	glLightfv(GL_LIGHT4, GL_AMBIENT, nul);
	glLightfv(GL_LIGHT4, GL_DIFFUSE, one);
	glLightfv(GL_LIGHT4, GL_SPECULAR, one);
	glLightfv(GL_LIGHT4, GL_POSITION, pos);

	glLightf(GL_LIGHT4, GL_CONSTANT_ATTENUATION,  0);
	glLightf(GL_LIGHT4, GL_LINEAR_ATTENUATION,    5e-3);
	glLightf(GL_LIGHT4, GL_QUADRATIC_ATTENUATION, 0);
	if(on) glEnable(GL_LIGHT4); else glDisable(GL_LIGHT4);
}
```

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Задорожный А.Г.* Введение в двумерную компьютерную графику с использованием библиотеки OpenGL: учебное пособие / А. Г. Задорожный, Д. В. Вагин, Ю. И. Кошкина. – Новосибирск: Изд-во НГТУ, 2018. – 102 с. – Режим доступа: https://elibrary.nstu.ru/source?id=73094.

2. *Задорожный А.Г.* Введение в трехмерную компьютерную графику с использованием библиотеки OpenGL: учебное пособие / А. Г. Задорожный, М. Г. Персова, Ю. И. Кошкина. – Новосибирск: Изд-во НГТУ, 2018. – 99 с. – Режим доступа: https://elibrary.nstu.ru/source?id= 75699.

3. *Роджерс Д.* Математические основы машинной графики / Д. Роджерс, Дж. Адамс. – М.: Мир, 2001. – 604 с.

4. *Залогова Л.А.* Компьютерная графика / Л. Залогова. – М., 2005. – 319 с.

5. *Дегтярев В.М.* Инженерная и компьютерная графика: учебник / В.М. Дегтярев, В.П. Затыльникова. – М., 2010. – 238 с.

СОДЕРЖАНИЕ

ФИЗИЧЕСКое освещение 3
Видимое излучение 5
Световой поток 7
Взаимодействие с поверхностью 7
Затухание с расстоянием 8
Проблема выбора освещения 9
Рекомендации 9
Трехточечное освещение 10
ВИРТУАЛЬНОЕ ОСВЕЩЕНИЕ 12
Основные типы источников света 15
Фоновый источник 16
Эмиссионный источник 16
Точечный источник 16
Прожекторный источник 17
Направленный источник 17
Объемный источник 17
Факторы, влияющие на освещенность 18
Учет поглощения 19
Учет нескольких источников 19
Учет затухания 20
Нормали 21
Расчет нормалей в плоскости 22
Расчет нормалей в трехмерном пространстве 23

Эффект сглаживания нормалей 25
Влияние модельных преобразований 26
Трехкомпонентное освещение 27
Фоновое освещение 28
Диффузное освещение 28
Зеркальное освещение 30
МОДЕЛИ ОСВЕЩЕНИЯ 31
Основные модели 33
Модель Фонга (Phong) 33
Модель Блинна-Фонга (Blinn-Fong) 34
Модель Ламберта (Lambert) 35
Модель Wrap-Around 35
Микрофасеточные модели 36
Модель Gauss 37
Модель Oren–Nayar 37
Модель Cook-Torrance 38
Анизотропные модели 39
Модель Ward 39
Модель Minnaert 39
Нефотореалистичные модели 40
Модель Toon Shading 40
Модели Gooch и Hemispheric lighting 40
Модель Bidirectional Lighting 40
Модель Lommel-Seeliger model 40
АЛГОРИТМЫ ЗАТЕНЕНИЯ 41
Плоское затенение 42

Затенение по Гуро 43
Затенение по Фонгу 44
РЕАЛИЗАЦИЯ ОСВЕЩЕНИЯ В OPENGL 45
Модель освещения и алгоритм затенения 47
Типы источников освещения 48
Задание освещения 49
Включение режима освещения 49
Задание модели освещения 50
Задание параметров источника света 52
Задание материала 54
Расчет нормалей 58
Получение статуса параметров освещения 60
Статус модели освещения 61
Статус источника света 62
Статус материала 63
Статус освещения и автонормирования 63
ПРИМЕР РЕАЛИЗАЦИИ ОСВЕЩЕНИЯ 64
ПРИЛОЖЕНИЕ А. Карскас приложения 69
ПРИЛОЖЕНИЕ Б. Функция рисования плоскости .. 71
ПРИЛОЖЕНИЕ В. Функция рисования сферы 72
ПРИЛОЖЕНИЕ Г. Функции задания ламп 74
БИБЛИОГРАФИЧЕСКИЙ СПИСОК 76

Задорожный Александр Геннадьевич

МОДЕЛИ ОСВЕЩЕНИЯ И АЛГОРИТМЫ ЗАТЕНЕНИЯ В КОМПЬЮТЕРНОЙ ГРАФИКЕ

Учебное пособие

В авторской редакции

Выпускающий редактор И.П. Брованова
Дизайн обложки А.В. Ладыжская

Налоговая льгота – Общероссийский классификатор продукции
Издание соответствует коду 95 3000 ОК 005-93 (ОКП)

Подписано в печать 08.12.2020. Формат 60 × 84 1/16. Бумага офсетная.
Тираж 100 экз. Уч.-изд. л. 4,65. Печ. л. 5. Изд. №264/19. Заказ № 49
Цена договорная

Отпечатано в типографии
Новосибирского государственного технического университета
630073, г. Новосибирск, пр. К. Маркса, 20