Машинная графика
Computer Graphics

Лекция 16.

«Построение реалистичных изображений»

План лекции

- Модели освещения
- Модели закрашивания граней
- Метод Гуро
- Метод Фонга
- Текстура
- Методы фрактальной геометрии
- Метод трассировки лучей
- Конвейер построения реалистичных изображений в OpenGL

Основные понятия

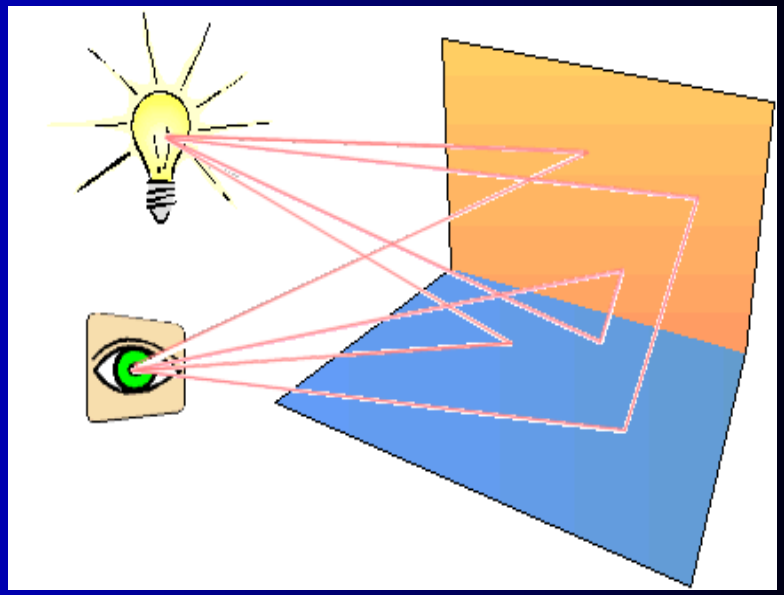
- **Модель освещения -**
  - это математическое представление физических свойств источников света и поверхностей, а также их взаимного расположения.
- **Простая модель освещения -**
  - основана на вычислении интенсивности отражаемого объектом света от точечного источника.

Illumination Models

Illumination: transport of luminous flux from light sources via direct & indirect paths.

Lighting: computing intensity reflected from 3D point in scene.

Shading: assigning pixel colour.



Illumination Models:

- Empirical: simple approximations to observed phenomena.

- Physically-based: model actual physics of light interactions.

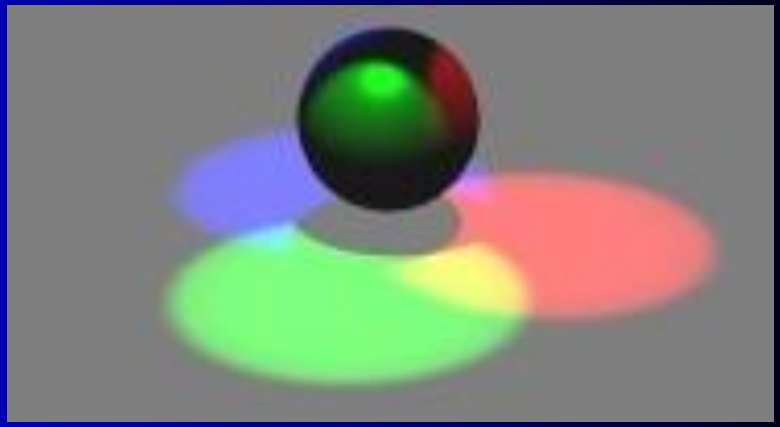
Two Components of Illumination

Light Sources:

-Emittance spectrum (colour)

-Geometry (position and colour)

-Directional attenuation



Surface Properties:

-Reflectance spectrum (colour)

-Geometry (position, orientation, and micro-structure)

-Absorption

Simplifications:

- Use only direct illumination from emitters to reflectors

- Ignore geometry of emitters, use only geometry of reflectors

Фоновое освещение (Ambient Reflection)

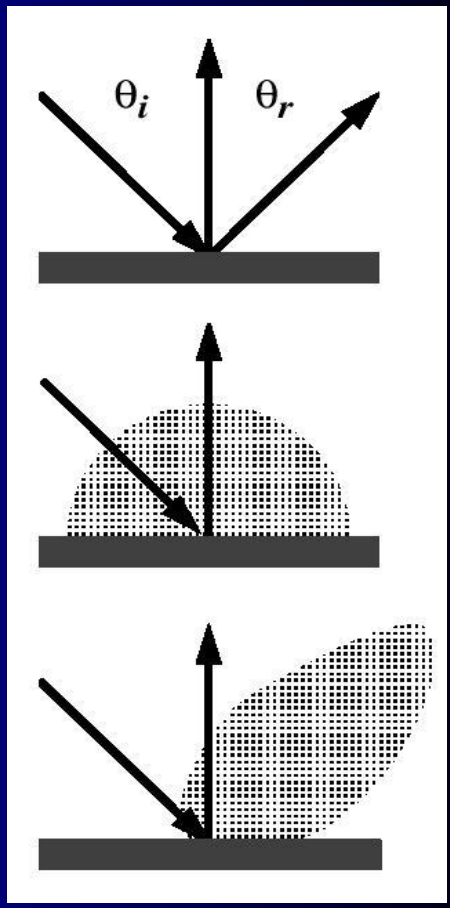
- Простейшая модель освещения.
- Подразумевает ровное освещение для всех объектов сцены со всех сторон, I_a
- Суммарное количество отражённых рассеянных лучей произвольной поверхностью определяется коэффициентом фонового отражения, k_a
- Таким образом цвет некоторой поверхности зависит от $I = I_a k_a$
- Данная модель не учитывает и не использует никаких физических законов вообще!

Простая модель освещения

Световая энергия, падающая на поверхность, может быть поглощена, отражена или пропущена.

Источники освещения – точечные, направленные (прожекторы) и светоизлучающие поверхности (неоновые лампы). Освещение может так же быть «рассеянным» (фоновым).

- Идеальное отражение:
  - зеркало
  - закон отражения
- Идеальное рассеивание:
  - матовая поверхность
  - закон Ламберта (Lambert)
- Реальное отражение:
  - Глянцевые поверхности (световые блики)
  - модели Фонга (Phong) и Блинна (Blinn)



Простая модель освещения

- Свойства отраженного света зависят от строения, направления и формы источника света, от ориентации и свойств поверхности.

Диффузное отражение происходит при равномерном по всем направлениям рассеивании света, поэтому создается иллюзия, что поверхности имеют одинаковую яркость независимо от углов обзора. Рассеянный свет практически всегда присутствует в реальной обстановке.

Зеркально отраженный свет не рассеивается. Угол отражения от идеальной отражающей поверхности равен углу падения. В любом другом положении наблюдатель не видит зеркально отраженный свет. Благодаря зеркальному отражению на блестящих предметах появляются блики.

Диффузное отражение

- Пример поверхности с диффузным отражением – кусочек мела.
- Описываются законом отражения Ламберта (*Lambertian* reflection).
- Отражённый свет равномерно рассеивается во всех направлениях.
- Для определённых поверхностей яркость зависит только от угла между векторами направления падающего света и нормалью к поверхности.
- Коэффициент рассеивания определён для каждой поверхности (материала) отдельно.
- Объекты только с диффузным отражением часто выглядят слишком яркими. Следует добавлять в модель фоновое освещение.
- Физические основы диффузного отражения проработаны слабо.

Диффузное отражение

- Свет точечного источника отражается от идеального рассеиватель по закону косинусов Ламберта:

$$I = I_l \; k_d \cos\theta,$$

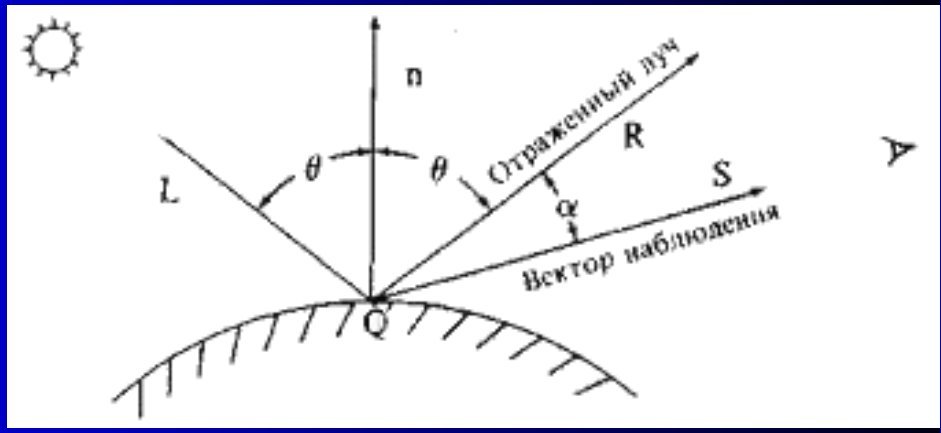
- где I_l – интенсивность источника света, I – интенсивность отраженного света, k_d – коэффициент диффузного отражения, θ – угол между направлением света и нормалью к поверхности.
- При освещении объекта точечным источником на него падает также свет, отраженный от соседних объектов. Этот свет будет рассеянным (фоновое освещение). Ему соответствует распределенный источник. Для вычисления интенсивности рассеянного света используется формула вида:

$$I = I_a \, k_a + I_l \, k_d \cos\theta,$$

- где I_a – интенсивность рассеянного света, k_a – коэффициент диффузного отражения рассеянного света $(0<=k_a<=1)$.

Зеркальное отражение

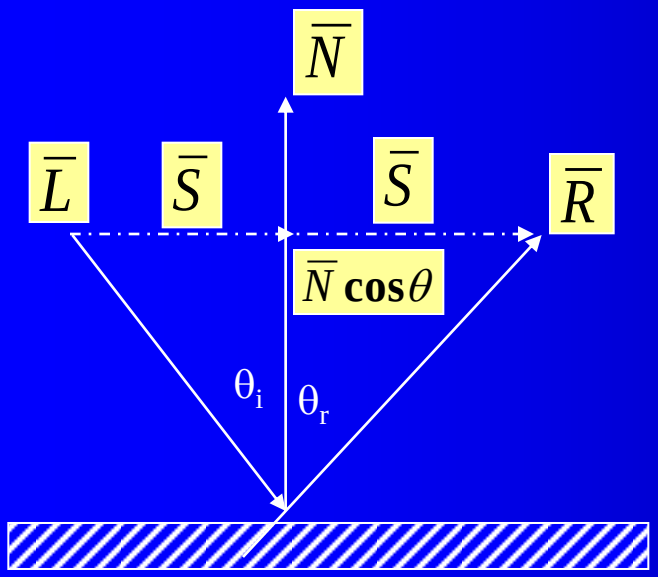
Интенсивность зеркально отраженного света зависит от угла падения, длины волны падающего света и свойств вещества. Зеркальное отражение света является направленным.



Угол отражения от идеальной отражающей поверхности (зеркала) равен углу падения, в любом другом положении наблюдатель не видит зеркально отраженный свет. Это означает, что вектор наблюдения **S** совпадает с вектором отражения **R**, и угол равен нулю. Если поверхность не идеальна, то количество света, достигающее наблюдателя, зависит от пространственного распределения зеркального отраженного света. У гладких поверхностей распределение узкое или сфокусированное, у шероховатых - более широкое.

Зеркальное отражение: идеальная отражающая поверхность

Закон отражения



$\theta_r = \theta_i$

Все лучи лежат в одной плоскости.

Вычисление вектора отражённого света по вектору падающего света **L** относительно **N**:

Вектора $\overline{L}$ и $\overline{N}$ нормализованы

Проекция $\overline{L}$ на $\overline{N}$ - $\overline{N}\cos\theta$

Из вычитания векторов:

$$\overline{S} = \overline{N}\cos\theta - \overline{L}$$

Аналогично:

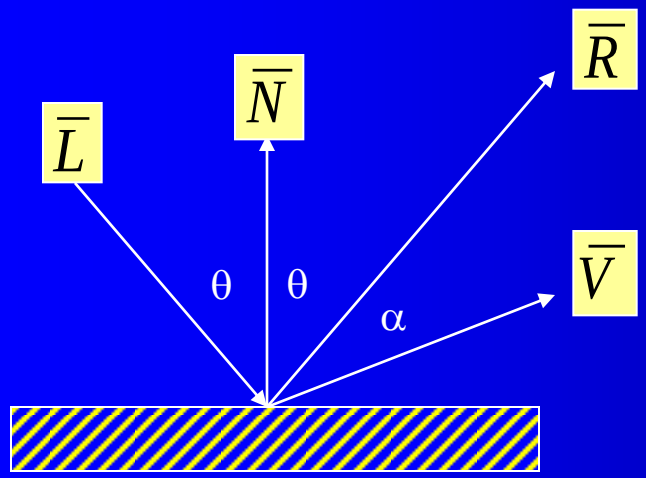
$$\overline{R} = 2\overline{N}\cos\theta - \overline{L}$$

Заменяя $\overline{N}\cdot\overline{L}$ вместо θ:

$$\overline{R} = 2\overline{N}\cdot(\overline{N}\cdot\overline{L}) - \overline{L}$$

Для видимого спектра поверхность зеркала должна иметь шероховатость менее 0.5 мкм. Иначе – лучи с большей длиной волны отражаются сильнее (красный сильнее синего).

Зеркальное отражение: модель Фонга (The Phong model)



$$I_\lambda = I_p k_s \cos^n \alpha$$

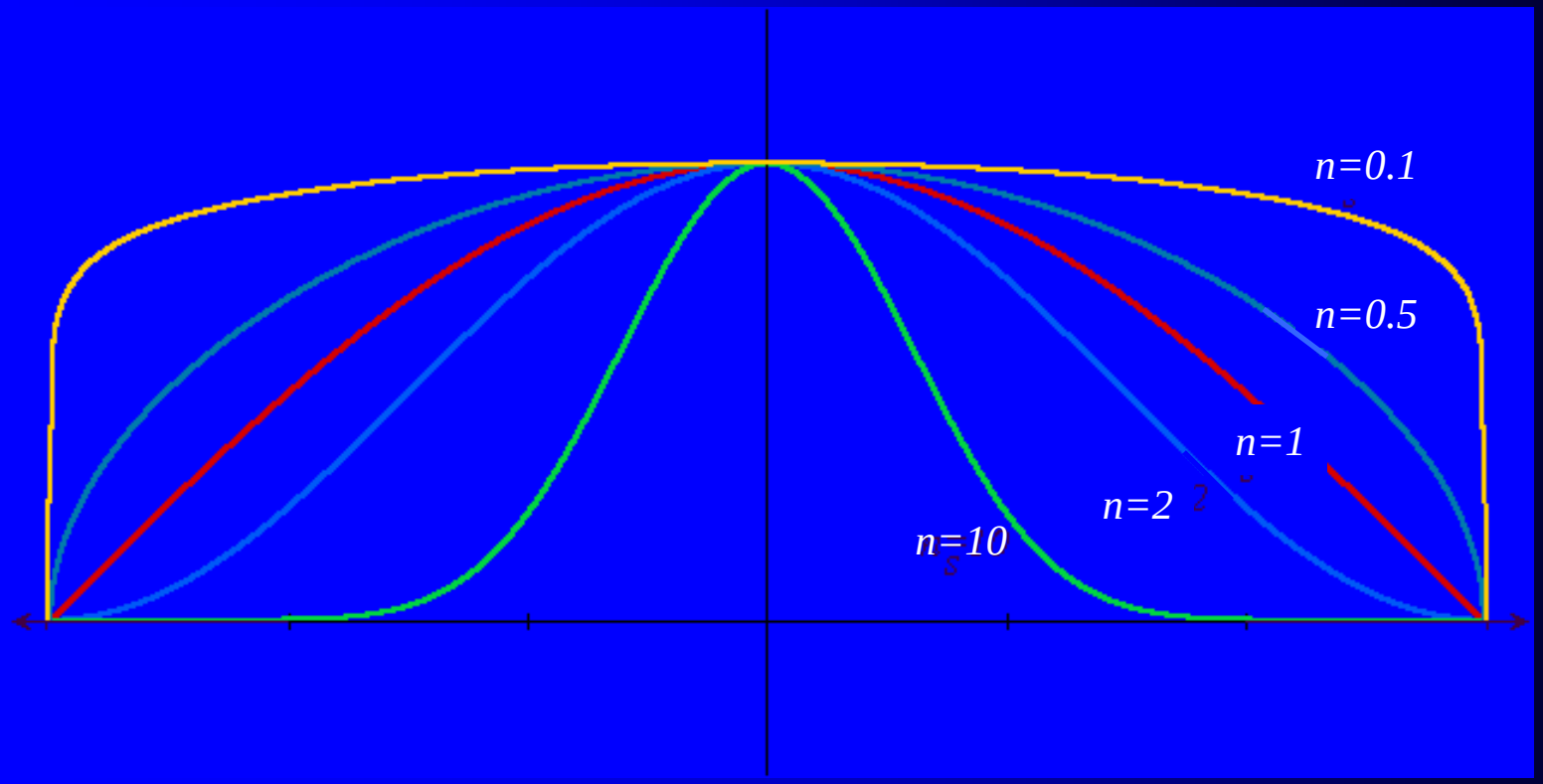
где k_s – коэффициент зеркального отражения

При наблюдении зеркального отражения максимум интенсивности отражённого света имеется при $\alpha = 0$, и падает (в зависимости от свойств поверхности) при увеличении угла α.

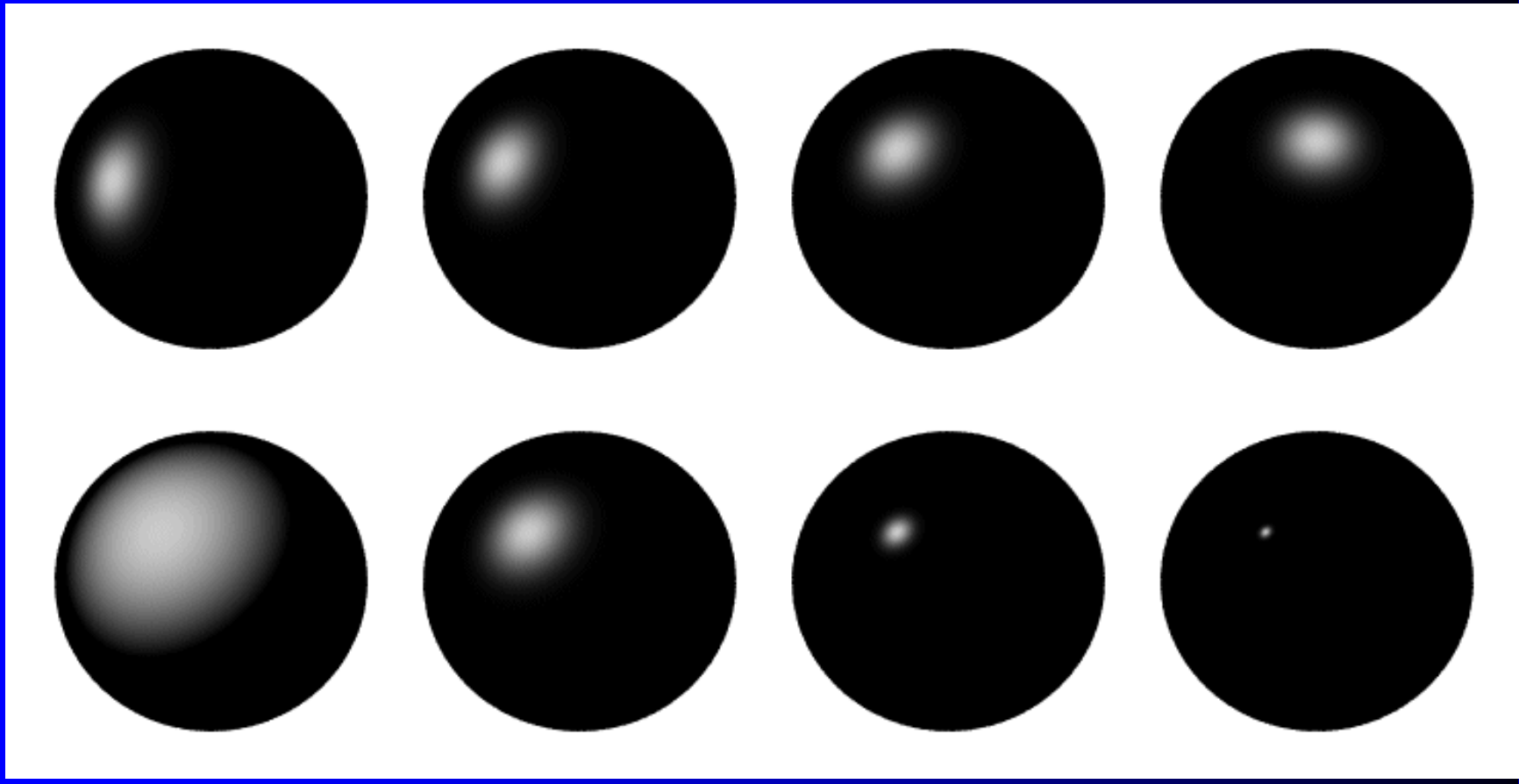
По модели Фонга уменьшение интенсивности зависит от $\cos^n \alpha$, где n – показатель зеркального отражения.

Для идеального отражателя, $n = \infty$.

Модель Фонга – влияние параметра зеркального отражения *n*

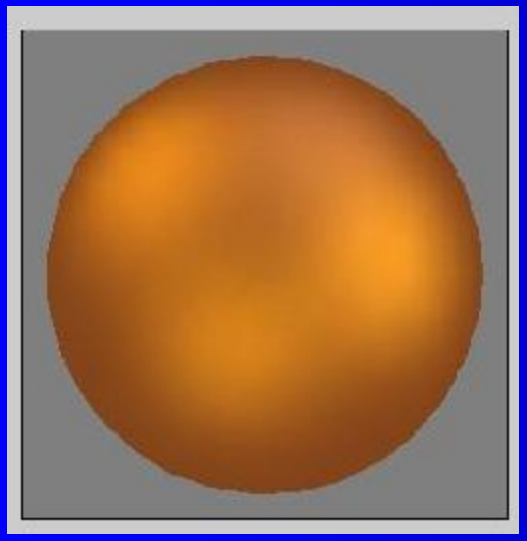


Зеркальное отражение: модель Фонга примеры

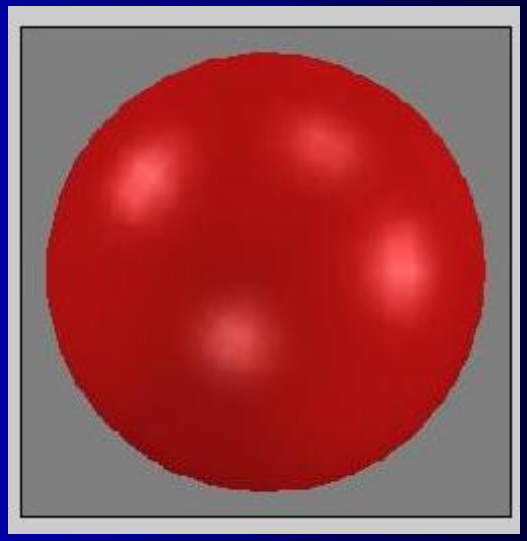


Моделирование различных материалов

Фоновое	0.52
Диффузное	0.00
Зеркальное	0.82
Блеск	0.10
Интенсивность	0.31



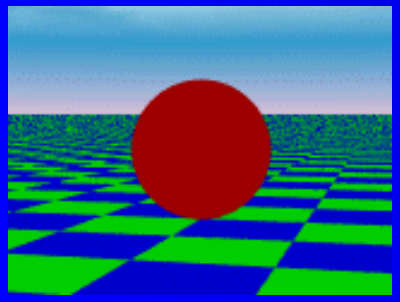
Фоновое	0.39
Диффузное	0.46
Зеркальное	0.82
Блеск	0.75
Интенсивность	0.52



Простая модель освещения

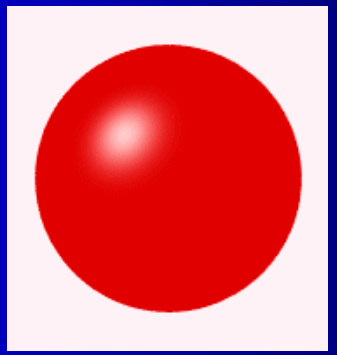
$$I_\lambda = I_a k_a + I_p [k_d \cos\theta + k_s \cos^n \alpha]$$

Фоновое освещение



Рассеянное освещение

Зеркальное освещение



Простая модель освещения с учётом расстояния от источника

$$I = I_a k_a + \frac{I_t}{d + K}\left(k_d \cos\theta + k_s \cos^n \alpha\right)$$

- где *ks* – константа, которая определяется экспериментально; α – угол между вектором отраженного луча и вектором наблюдения; *n* – степень, аппроксимирующая пространственное распределение зеркально отраженного света.
- Если используется несколько источников света, то их эффекты суммируются. Вычислительная сложность алгоритма построения объекта с использованием простой модели освещенности довольно высока.

$$I_\lambda = I_a k_a + \sum_{p=1}^{lights} \frac{I_p}{d_p + k} I_p [k_d \cos\theta + k_s \cos^n \alpha]$$

При цветном изображении расчет интенсивности для каждого цвета производится отдельно, что предопределяет существенные вычислительные затраты. Для упрощения расчетов интенсивности в точках объектов используется интерполяция.

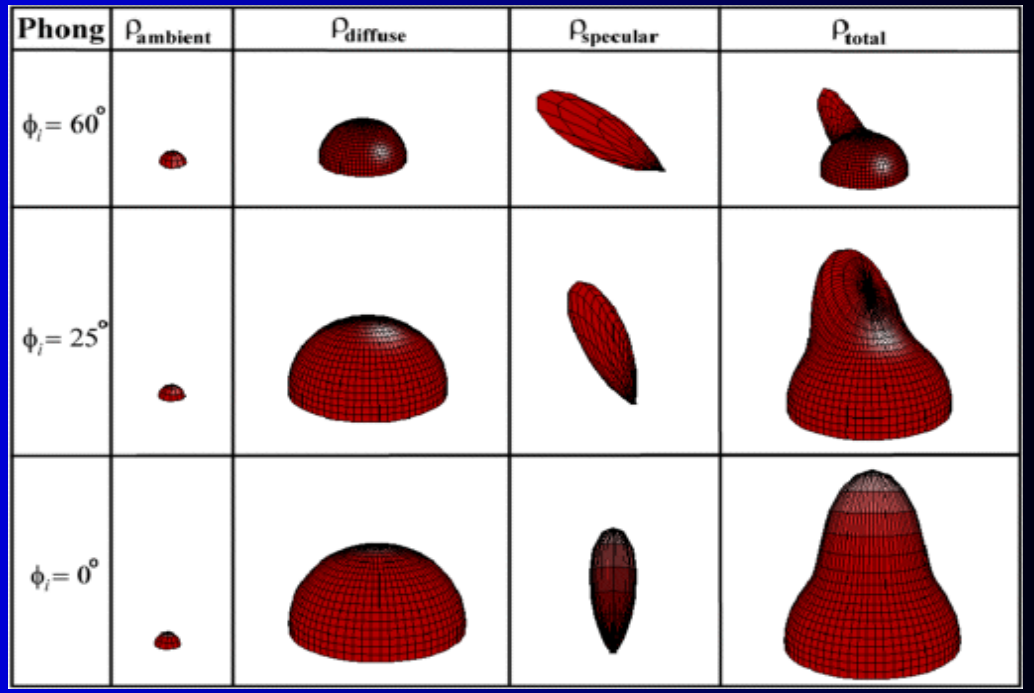
There may be a number of Point Sources

Данная модель имеет название стандартной (простой) и является эмпирической:

$$I_\lambda = I_a k_a + \sum_{p=1}^{lights} I_p [k_d (\bar{N} \bullet \bar{L}) + k_s (\bar{V} \bullet \bar{R})^n]$$

Стандартная ЛОКАЛЬНАЯ МОДЕЛЬ ОСВЕЩЕНИЯ

(LOCAL ILUMINATION MODEL)



Модели закрашивания граней

1. Простейшая/плоская закраска
   - -вычисляется нормаль к грани
   - -с помощью некоторой модели освещённости вычисляется интенсивность освещения грани в некоторой точке (любой!)
   - -все пиксели грани окрашиваются в данный цвет.

2. Метод Гуро
   - -вычислить нормаль к каждой грани
   - -определить «нормали» в каждой вершине, усреднением нормалей примыкающих граней
   - -определить интенсивность (цвет) для каждой вершины
   - -закрасить грань с помощью линейной интерполяции цветов в вершинах граней

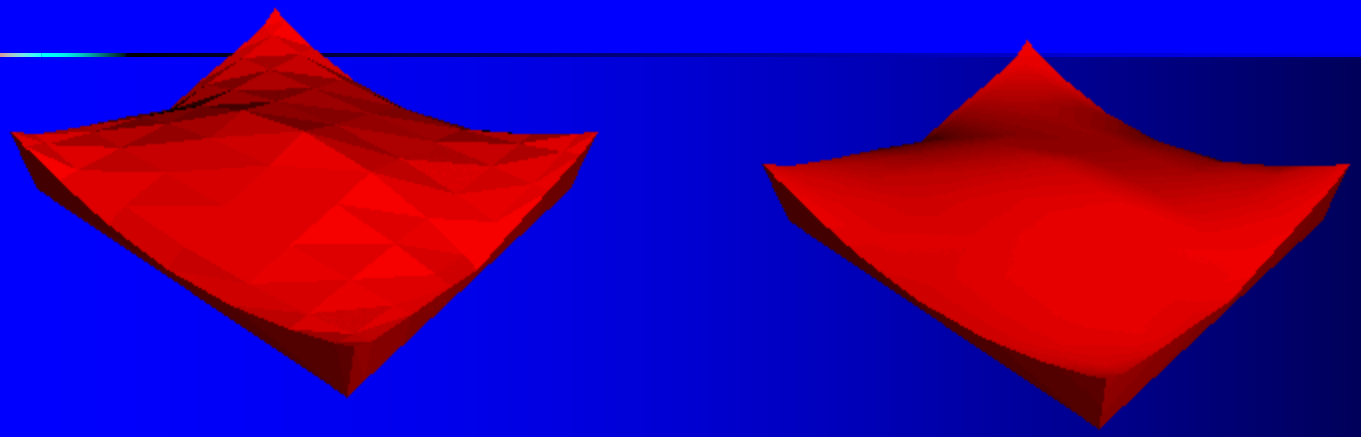
Модели закрашивания граней

- Первые опыты по линейной интерполяции освещённых поверхностей проводили Wylie, Romney, Evans and Erdahl.
- Гуро (Gouraud) обобщил их подход для произвольных полигонов.

Закраска (затенение) по Гуро - это метод линейной интерполяции освещенности в пределах одного полигона. Он был изобретен в 1971 и носит имя своего изобретателя. Это простой и эффективный метод придания ощущения изогнутости для ровного полигона. С точки зрения физики – данный метод не корректен.

Этот метод также часто используется для сокращения глубины прорисовываемой сцены путем имитации исчезновения удаленных объектов в тумане.

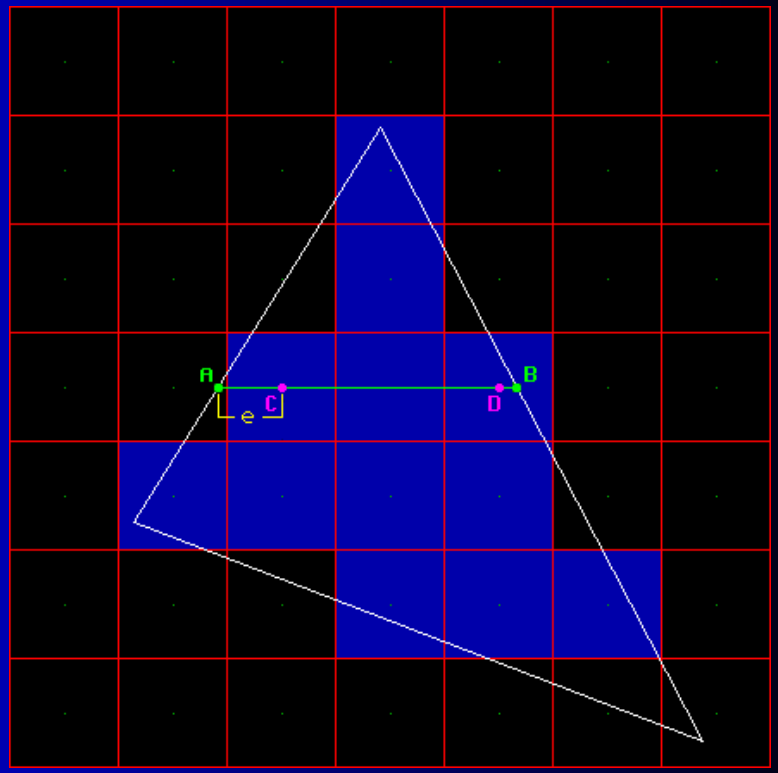
Пример работы метода Гуро



Слева изогнутая поверхность (состоящая из множества треугольных полигонов, или попросту - треугольников), выведенная на экран с помощью простейшей модели, где каждая треугольная грань (полигон) имеет равномерную освещенность, справа та же поверхность, но с применением затенения по Гуро.

Интерполяции освещения по полигону

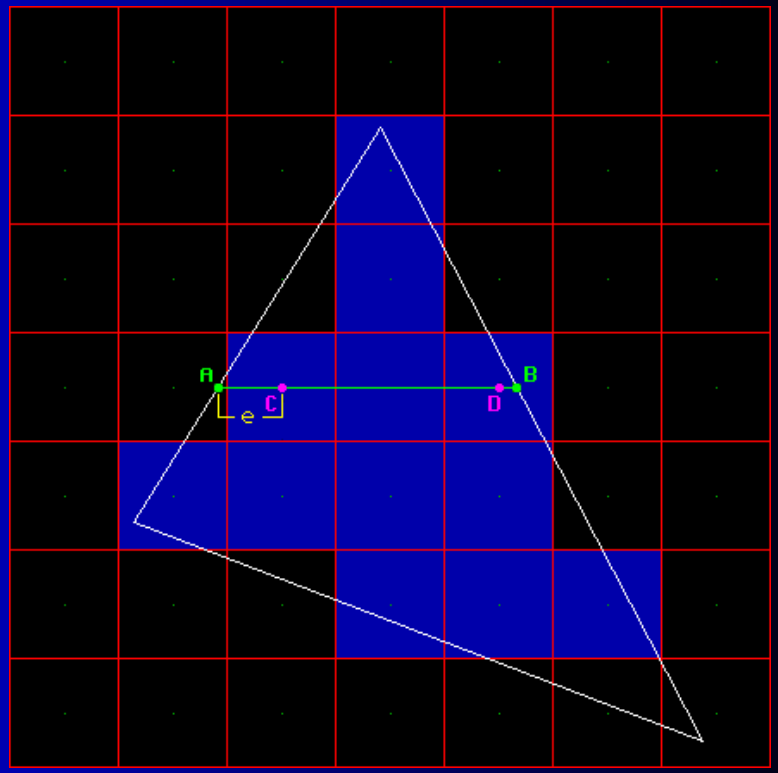
Предположим, что освещенность в вершинах уже вычислена, и допустив, что вдоль кромок полигона она изменяется линейно, вычислить ее значение в точках A и B на каждое изменение координаты Y будет несложно. Соответственно перед началом дальнейших вычислений мы знаем освещенность в точках A и B.



Интерполяции освещения по полигону

Мы хотим просчитать линию A-B. Но мы имеем дело с пикселизованным экраном, поэтому мы сможем вывести на экран только линию, состоящую из трех пикселей C- D. Центр первого пикселя, C, не совпадает с центром точки A, а центр пикселя D не совпадает с точкой B.

Можно смещения в доли пикселя и не учитывать, но это будет справедливо только для больших полигонов, а для небольших это смещение учитывать надо. Особенно, если на полигон будет наложена текстура..



Интерполяции освещения по полигону

Сначала определим градиент изменения освещенности по всей длине линии. Произведем это обычным способом.

Gradient = (Bs - As) / (Bx - Ax);

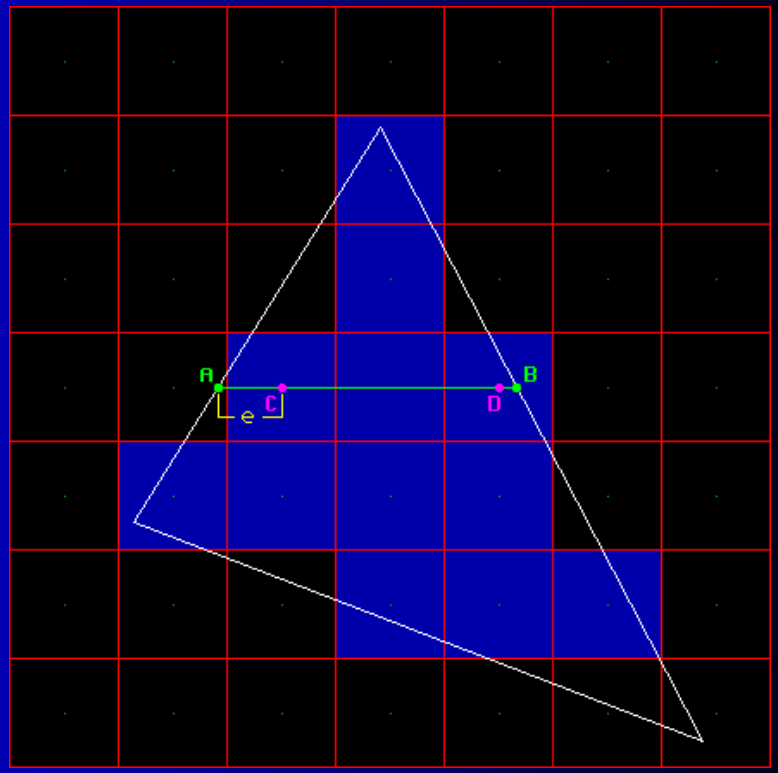
где: As и Bs - оттенок в точках A и B

Bx и Ax - значения X в точках A и B соответственно градиент показывает относительную величину изменения освещенности в пересчете на единицу изменения координаты X.

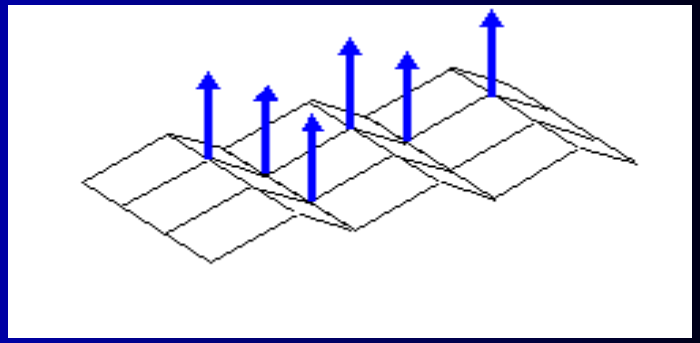
Теперь мы определим точное значение (shade) в точке C.

Cs = (1 - frac(Ax)) * Gradient;

величина **frac(Ax)** определяет смещение **e**



Недостатки метода Гуро



Метод Гуро обеспечивает непрерывное изменение интенсивности при переходе от одной грани к другой без разрывов. Однако он не обеспечивает *непрерывности изменения интенсивности* (т.е. непрерывности первой производной), в результате чего возможны дефекты изображения.

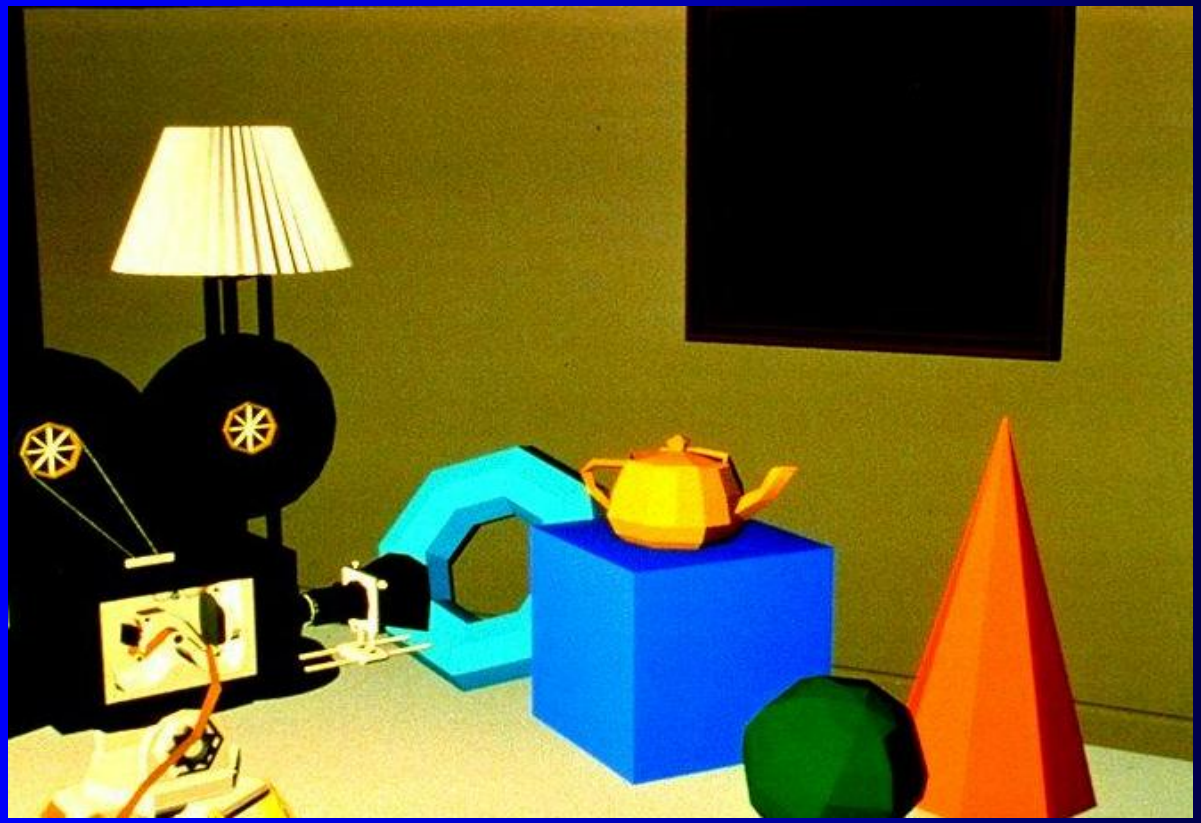
Закраска Гуро обычно используется для поверхностей с преимущественно *диффузным рассеянием* света. Другой недостаток, который может проявиться в методах градиентной закраски, связан с "черепитчатой" поверхностью и представлен на рисунке. Нормали к рёбрам одинаково направлены - следовательно различий в интенсивности от вершины к вершине не возникает, и поверхность будет закрашена как плоская.

При попытке использования метода для криволинейных поверхностей – ошибки будут очень заметны.

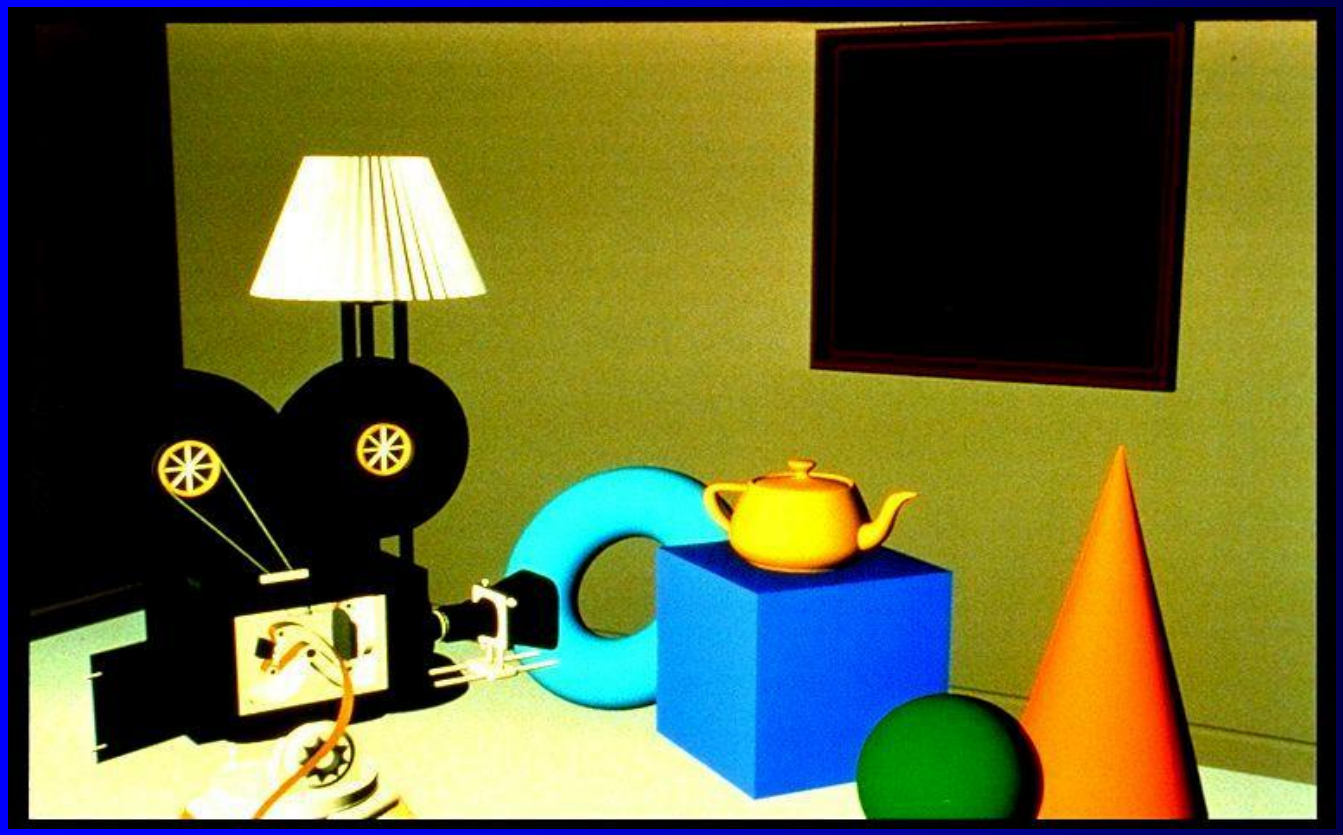
Модели закрашивания граней

2. Метод Гуро
   - вычислить нормаль к каждой грани
   - определить «нормали» в каждой вершине, усреднением нормалей примыкающих граней
   - определить интенсивность (цвет) для каждой вершины
   - закрасить грань с помощью линейной интерполяции цветов в вершинах граней
3. Метод Фонга
   - пункты а) и б) – аналогично Гуро
   - с учётом линейной интерполяции вычислить нормали в каждой точке каждой грани
   - для каждой точки вычислить свой цвет.

Плоская закраска



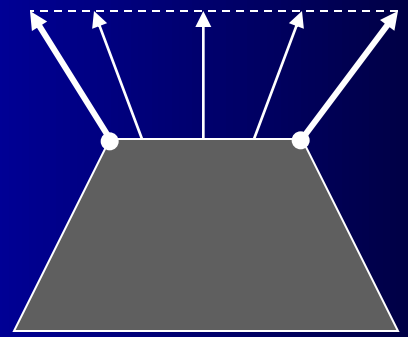
Закраска методом Гуро (Gouraud shading)



Метод Фонга (Phong shading)

- Нормали вычисляются в каждой вершине.
- Вектора нормалей интерполируются вдоль каждой плоскости.
- Освещённость вычисляется в каждой точке поверхности по нормали в данной точке.

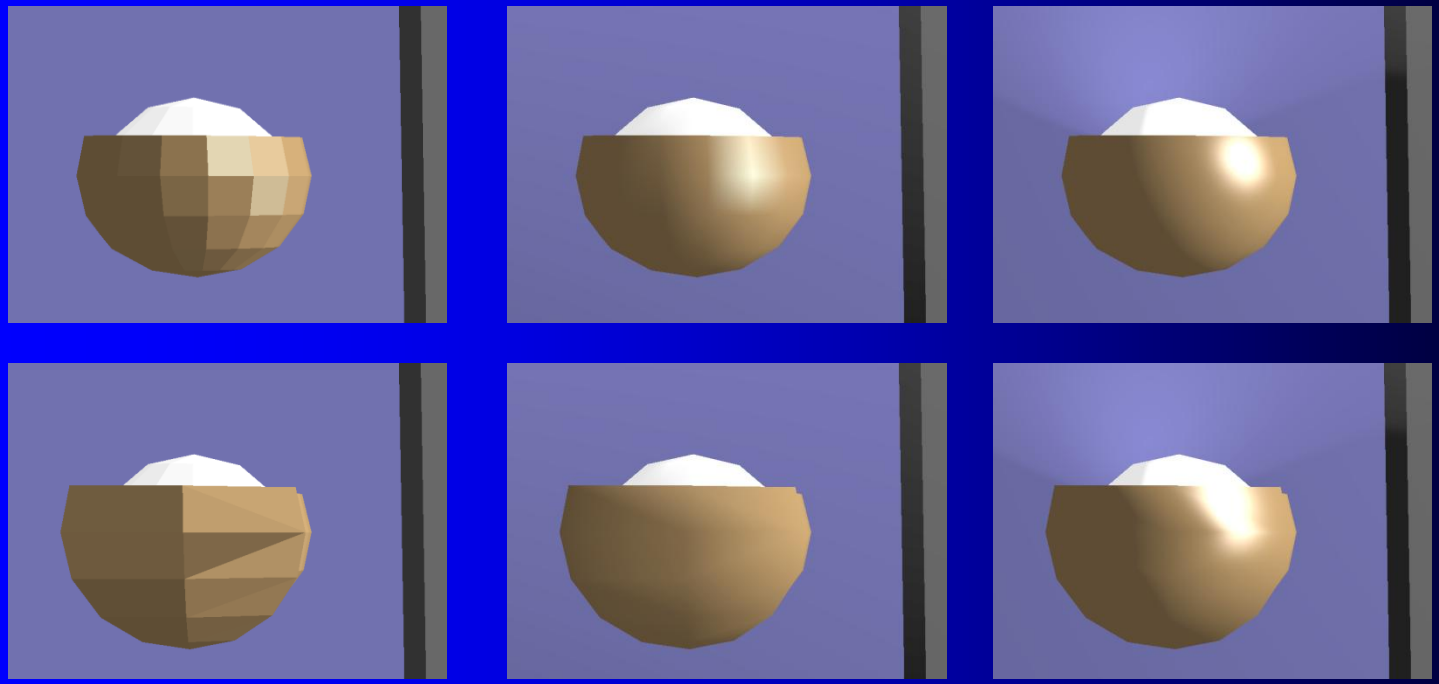
Интерполированные вектора нормалей



Пример закраски по Фонгу

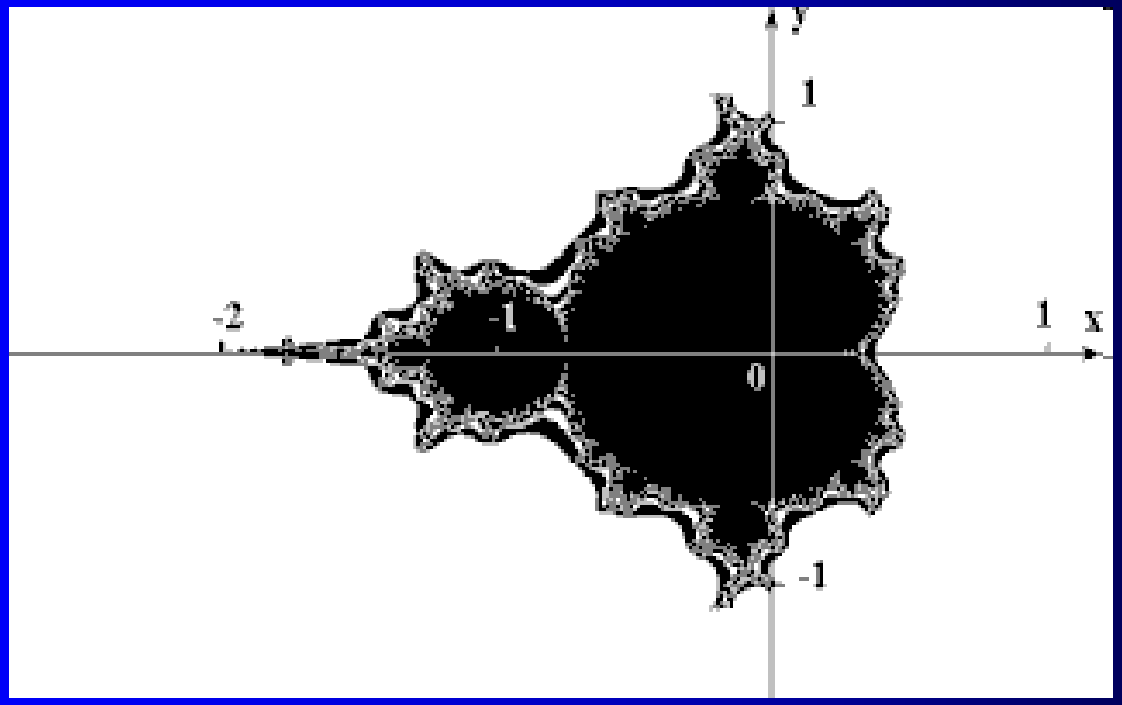
Сравнение методов

Плоская **Гуро** **Фонг**



Более редкая решётка

Фракталы



Фракталы

- Фракталы с большой точностью могут описывать многие физические явления и природные образования: горы, облака, деревья, ландшафты... Впервые фрактальную природу нашего мира подметил Бенуа Мандельброт. Слово «фрактал» происходит от латинского *fractus*, что означает «дробный», и *frahgere* — «ломать». Главной особенностью фракталов является их *бесконечное* самоподобие.

Фракталы

- Фрактальный объект обладает двумя базовыми характеристиками: бесконечное число деталей в любой точке и определённое самоподобие частей объекта и общего изображения объекта.
- Фрактальные функции широко используются в качестве инструментов для реалистичного построения природных объектов, бесконечно сложных узоров и картин... В машинной графике метод построения фрактальных поверхностей первыми применили Карпентер, Фурнье и Фассел.
- Фракталы — это процедуры, рекурсивно применяющиеся к частям объекта, который они описывают.

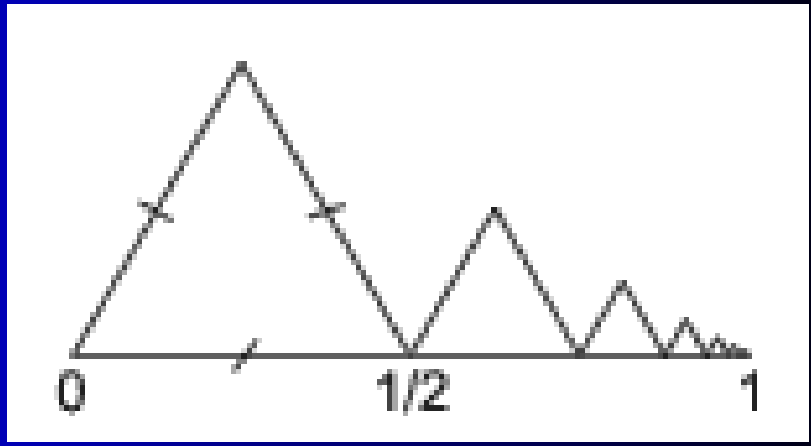
Фракталы

- Фрактальная поверхность состоит из случайно заданных полигональных или биполиномиальных поверхностей. Одно из преимуществ таких поверхностей в том, что можно получить любой уровень их детализации, независимо от того, насколько близко мы к ним находимся. В машинной графике фракталы строятся простыми и быстрыми итерационными алгоритмами.

Фракталы

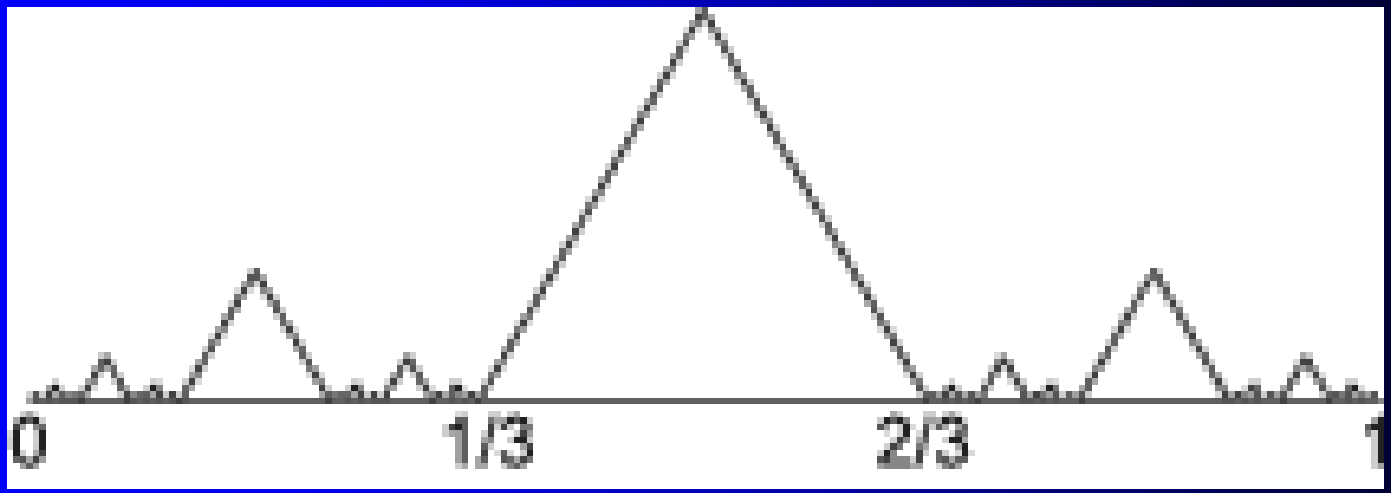
- Пример: линия-фрактал, имеющая бесконечное число максимумов и минимумов на отрезке (0, 1).
- Построить эту функцию можно так: разбиваем отрезок на 1/2, строим равносторонний треугольник; одну из сторон делим на 2 и от него строим следующий треугольник с меньшей стороной и так до бесконечности...

$$f(x) = \begin{cases} x * \cos(\pi / x) \\ 0, x = 0 \end{cases}$$



Фракталы

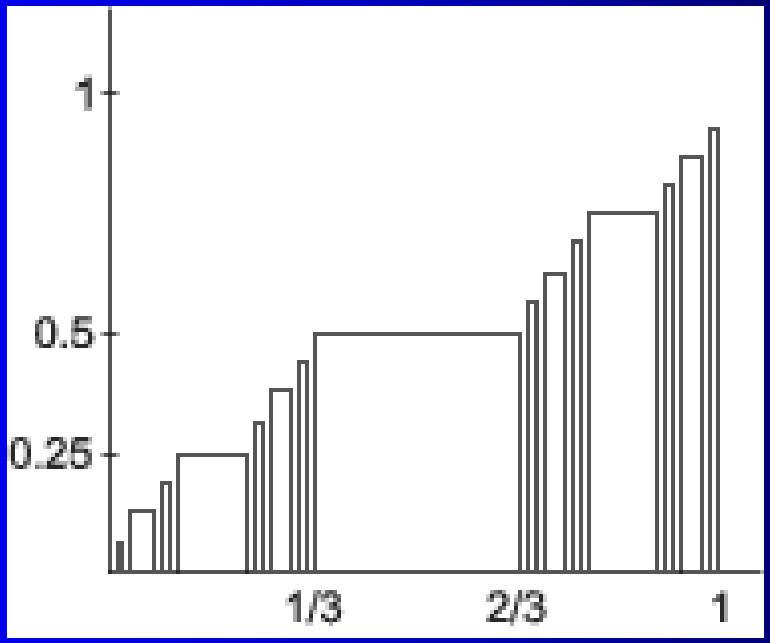
- Если делить отрезок не на две, а на три части, то можно построить другую линию



Такие функции обладают интересными и необычными свойствами. Например, данный фрактал имеет такие свойства: число треугольников («крыш») бесконечно; на такой линии расположено *бесконечное* число точек, которые не защищены «крышей» — это точки 0, ..., 1/9, 2/9, ..., 1/3, 2/3, ..., 1.

Фракталы

- Рассмотрим еще некоторые виды фракталов. На **рисунке** изображена лестница с бесконечным числом ступеней.



Фракталы

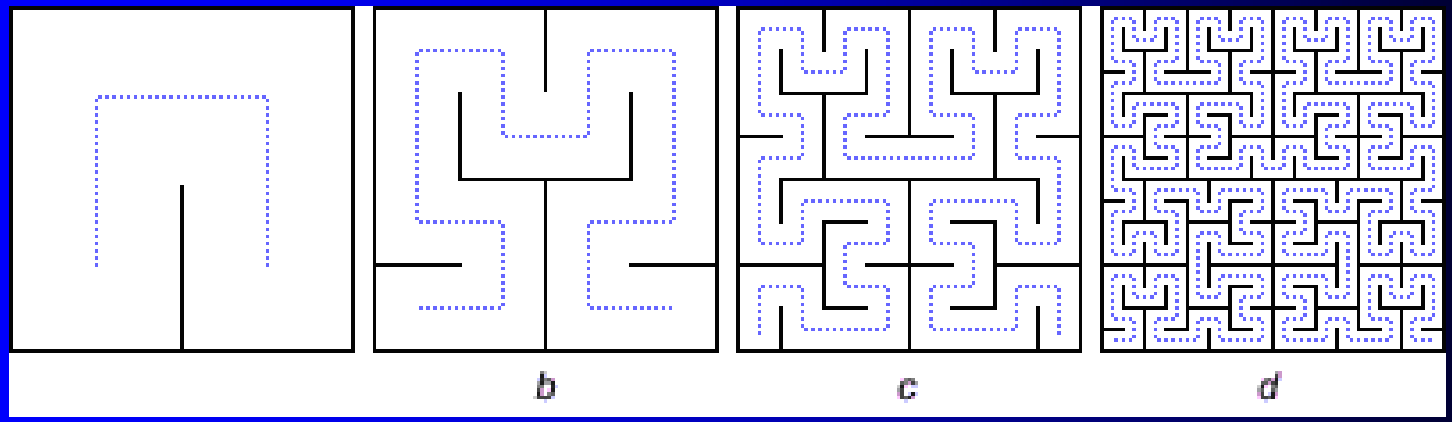
- Вычислим общую длину такой лестницы: 1 * 1/3 — первая ступень, 2 * 1/9 — вторые ступени, 4 * 1/27 — третьи ступени и т. д. Общая длина ступеней равна единице: 1 * 1/3 + 2 * 1/9 + 4 * 1/27 + ... = 1. Но при этом существует бесконечно много точек, которые не принадлежат ступеням, это так называемые «мокрые» точки: 0, ..., 1/9, 2/9, ..., 1/3, 2/3, ..., 1. (Они называются «мокрыми», потому что если начать лить на ступени воду, то она свободно просочится между стенками ступеней и попадет как раз в «мокрые» точки.)

Фракталы

- Этим простым примером иллюстрируется важнейшее положение современной математики о том, что *часть* равна *целому*. Мы только что показали, что между ступенями имеется бесчисленное число «мокрых» точек, образующих разрывы. Но, тем не менее, сумма длин всех ступеней строго равна единице! То есть «часть», составленная из длин всех ступеней, равна «целому», представляющему собой отрезок единичной длины!

Линия Пеано

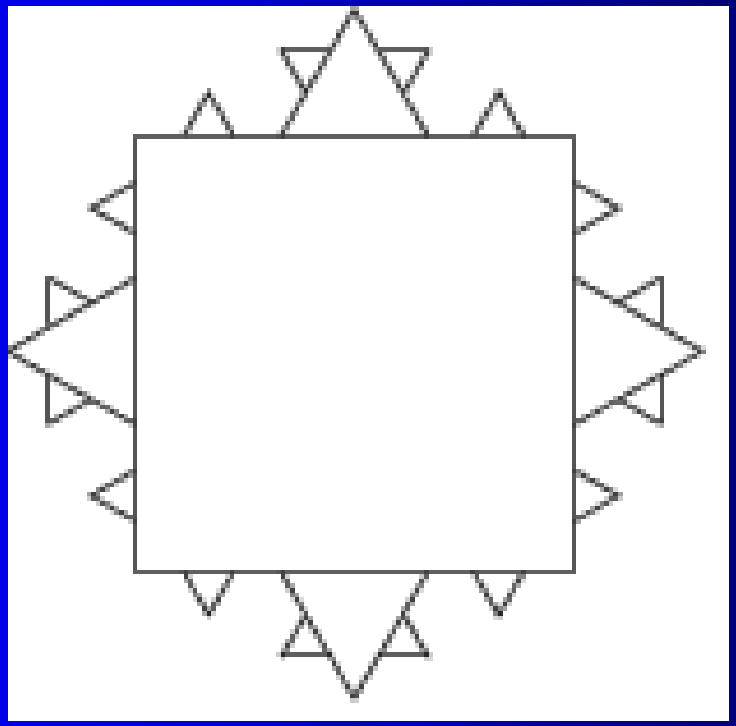
- Линия Пеано — это непрерывная линия, проходящая через все точки квадрата. Когда число замен линий на скобки будет стремиться к бесконечности в квадрате на **рисунке** не останется пустых мест:



Фрактал Коши

- «Бесконечно колючая линия», периметр P которой вычисляется по формуле: P = 3 * (4/3)n, где n — номер итерации (на рисунке показана линия для n=3).
- Если n бесконечно, то длина линии бесконечна. Примечательно то, что в *конечном* куске плоскости можно уместить *бесконечно* длинную линию.

Фрактал Коши



Фракталы

2D фракталы и 3D фракталы

Метод трассировки лучей

Метод трассировки лучей (ray tracing) – один из наиболее мощных и универсальных методов создания реалистичных изображений

Ключевая задача метода трассировки -определение освещённости произвольной точки объекта сцены и той части световой энергии, которая от него уходит в заданном направлении

Методы трассировки

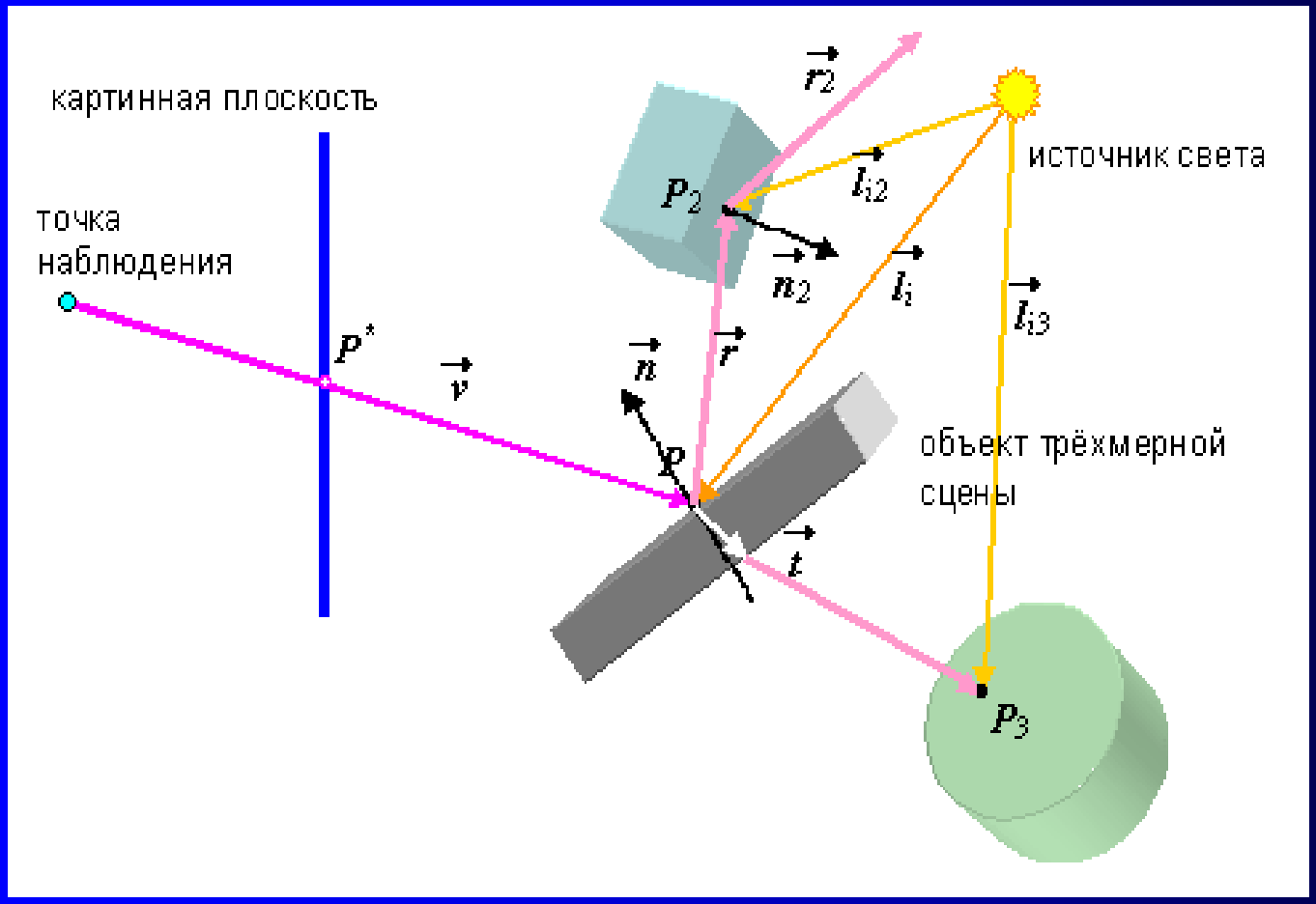
Прямая трассировка

- ***Принцип :*** рассчитываются пути лучей света, идущие от источников до объектов сцены. Если такой первичный луч попадает в прозрачный объект, то преломляясь и теряя интенсивность, он идёт дальше и, возможно, попадает на другой объект, в свою очередь отражаясь от него. Так, многократно отражаясь и преломляясь, какая-то часть лучей приходят в точку наблюдения - т.е. эти лучи определяют цвет некоторых пикселей экранной плоскости. Если ограничиться некоторыми ***первичными лучами***, выходящими из источников и попадающими только на объекты сцены или в камеру, а затем точно так же поступить со ***вторичными*** **лучами**, то время вычисления сцены уже будет конечным, а результат достаточно реалистичным..
- ***Недостатки :*** даже небольшое изменение точки наблюдения приведёт к пересчёту всей сцены. Недостаток данного метода - огромное количество "лишних" вычислительных операций.

Обратная трассировка

- Метод разработан в 80-х годах, основополагающими считаются работы Уиттеда и Кея.
- Метод позволяет значительно сократить перебор световых лучей.
- Согласно этому методу отслеживание лучей производится не от источников света, а в обратном направлении - от точки наблюдения через каждый пиксель картинной плоскости. Таким образом, учитываются только те лучи, которые формируют изображение.

Обратная трассировка



Обратная трассировка

На рисунке представлена сцена £ рассчитываемая по методу обратной трассировки лучейPt Из точки наблюдения через пиксель картинной плоскости восстанавливается луч зрения упираясь в точку P на видимой грани объекта трёхмерной сценыPt Эта точка освещается с помощью первичных лучей£ идущих от нескольких источников светаPt На рисунке показан один из источниковPtPt Кроме того£ если в этой точке грань является отражающей£ то восстановив вектор отражения £симметричный относительно нормали к грани£отследим пересечение этого вектора с другими объектами сцены и проведём расчёт освещённости для точки P½Pt

Обратная трассировка

- Таким образом, расчёт цвета пикселя P* в общем случае определяется освещённостью всех поверхностей сцены, видимых по лучу зрения с учётом многократного отражения, преломления и поглощения. Совершенно очевидно, что для убыстрения вычислений необходимо ограничиться небольшим числом отражённых или преломлённых лучей, либо вести оценку порога минимальной интенсивности света, приходящего по лучу зрения. Интенсивность в точке P некоторой поверхности сцены может быть рассчитана по следующей формуле:

$$I = k_a I_a + k_d \sum_i \frac{I_{l_i}}{d_{l_i} + k_{l_i}} (\bar{n} \cdot \bar{l}_i) + k_s \sum_i \frac{I_{l_i}}{d_{l_i} + k_{l_i}} \cdot \frac{R(\theta_i) \cdot D(\gamma)}{(\bar{n} \cdot \bar{l}_i) \cdot (\bar{n} \cdot \bar{v})} + k_s I_r R(\theta_r) \cdot e^{-\alpha_r d_r} + k_t I_t T(\theta_t) \cdot e^{-\alpha_t d_t}$$

Обратная трассировка

- ***Первое слагаемое*** - вклад фонового освещения; **ka** - коэффициент "участия" поверхности в фоновой засветке.
- **Второе слагаемое** - учёт прямого диффузного освещения поверхности источниками света, близкими к точечным; **kd** - коэффициент диффузного освещения.
- **Третье слагаемое** - учёт зеркального отражения источников света от поверхности по лучу зрения в соответствии со свойствами поверхности; **ks** - коэффициент зеркальности.
- **Четвёртое слагаемое** - учёт интенсивности света, приходящего по отражённому лучу зрения, с условием поглощения в среде, где проходит луч. **Ir** - интенсивность отражённого луча, рассчитываемая в свою очередь по предыдущей формуле для точки **P2**.
- **Пятое слагаемое** - учёт интенсивности света, приходящего по преломлённому лучу зрения, с условием поглощения в среде; **kt** - коэффициент прозрачности, **It** - интенсивность преломлённого луча, рассчитываемая по предыдущей формуле для точки **P3**.

Обратная трассировка

- *Таким образом£* ***алгоритм обратной трассировки лучей включает в себя:***

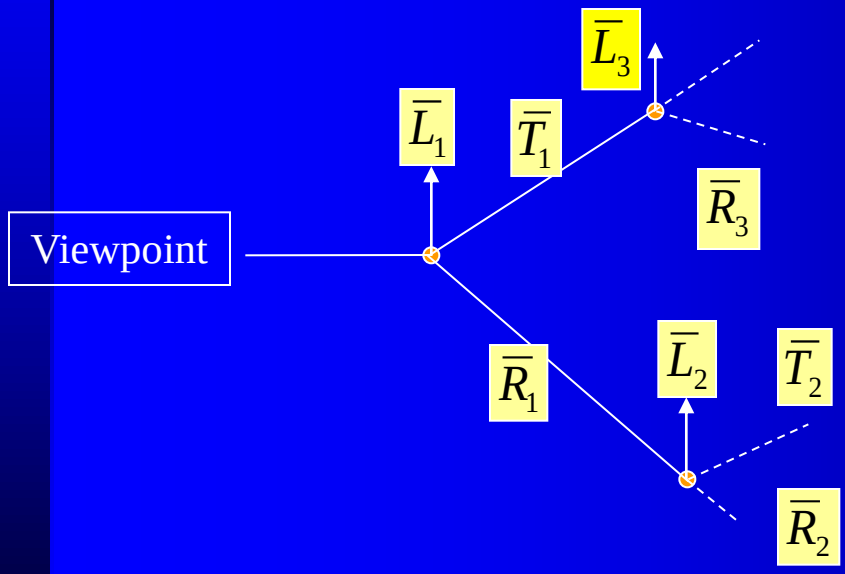
  º¿ *расчёт освещённости сцены с учётом положения источников света с параметрами яркости£ цвета£ направленности и положения объектов для расчёта теневых масок*Pt

  ½¿ *расчёт видимости граней объектов из точки наблюдения с учётом положения точки наблюдения и вида проецирования*Pt

  ¼¿ *расчёт цветовых компонент по освещённости и цвету граней с учётом оптических свойств поверхности и среды для каждого пикселя картинной плоскости*

Recursive ray tree

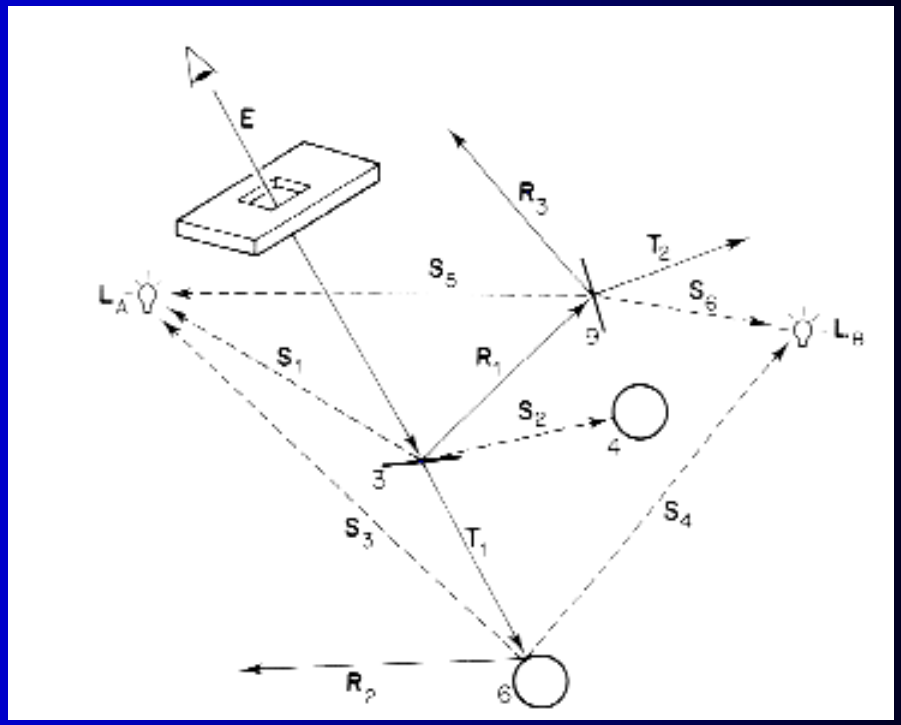
- Reflection and Transmission Rays spawn other rays.
  - Shadow rays test only for occlusion.
- The complete set of rays is called a *Ray Tree.*



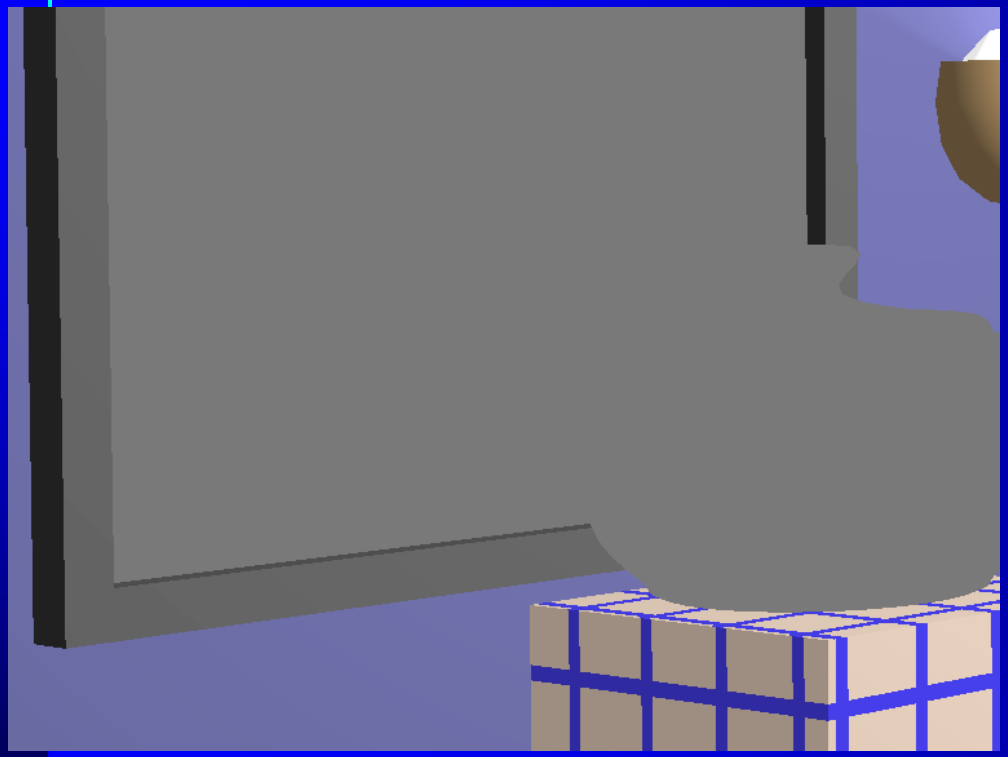
Light Source ray determines colour of current object.

Recursive ray tree

- Reflection and Transmission Rays spawn other rays.
  - Shadow rays test only for occlusion.
- The complete set of rays is called a *Ray Tree.*



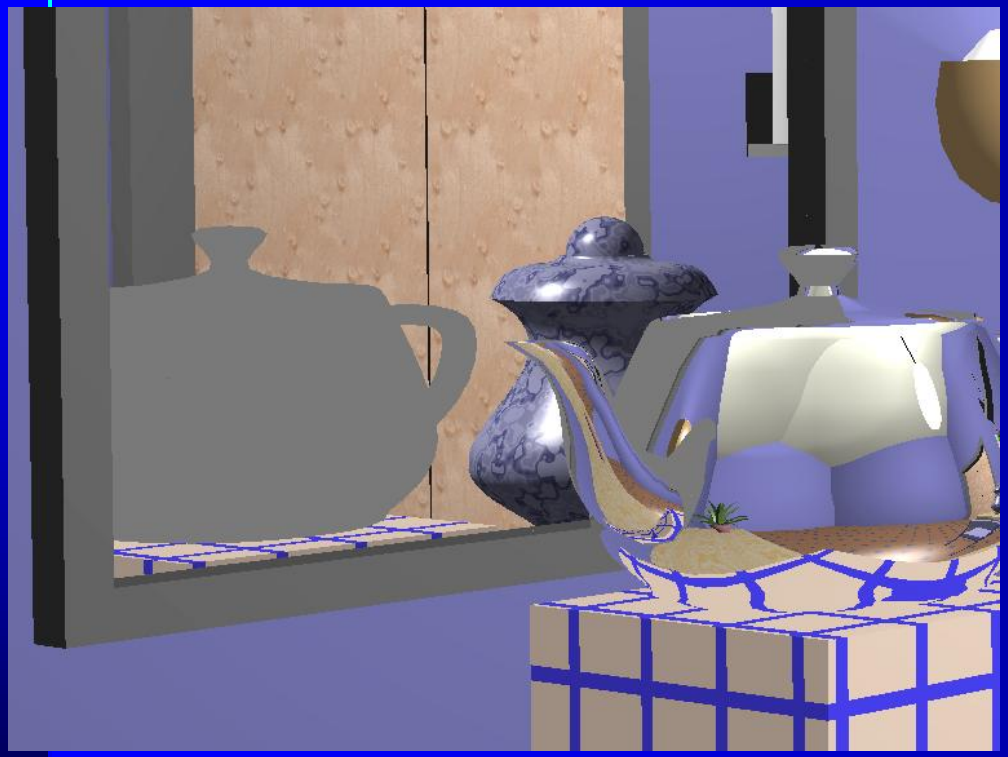
Test Scene



Ray tree depth 1.

Note only ambient shade on mirror and teapot

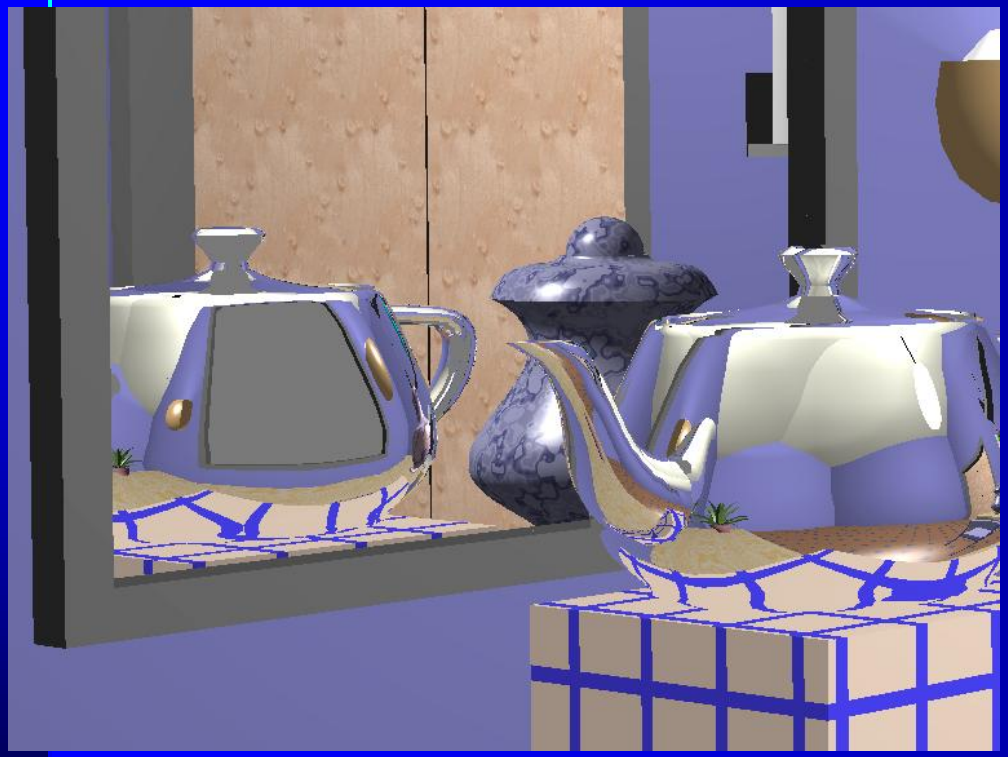
Test Scene



Ray tree depth 2.

Note only ambient shade on reflection of mirror and teapot.

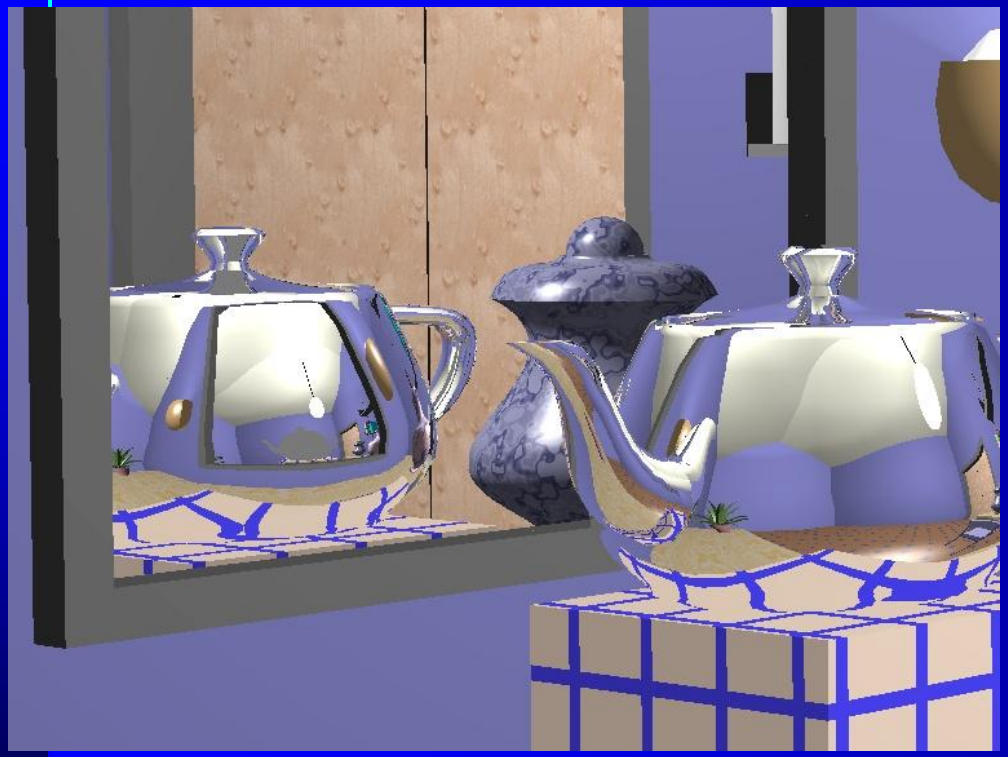
Test Scene



Ray tree depth 3.

Note only ambient shade on reflection of mirror in teapot.

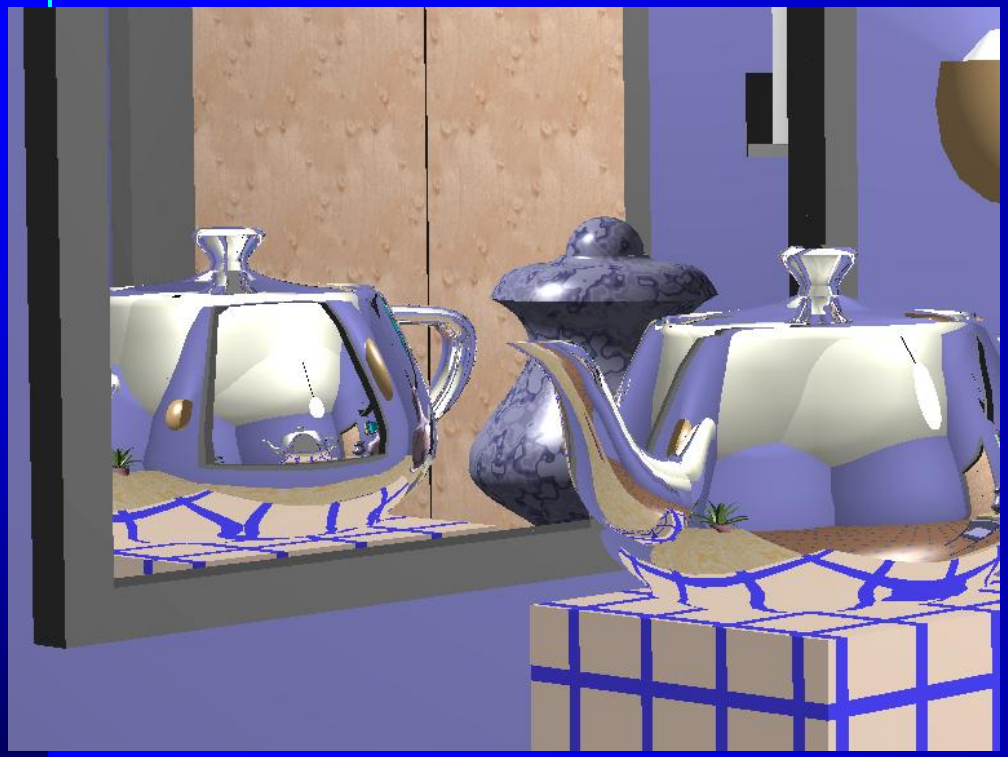
Test Scene



Ray tree depth 4.

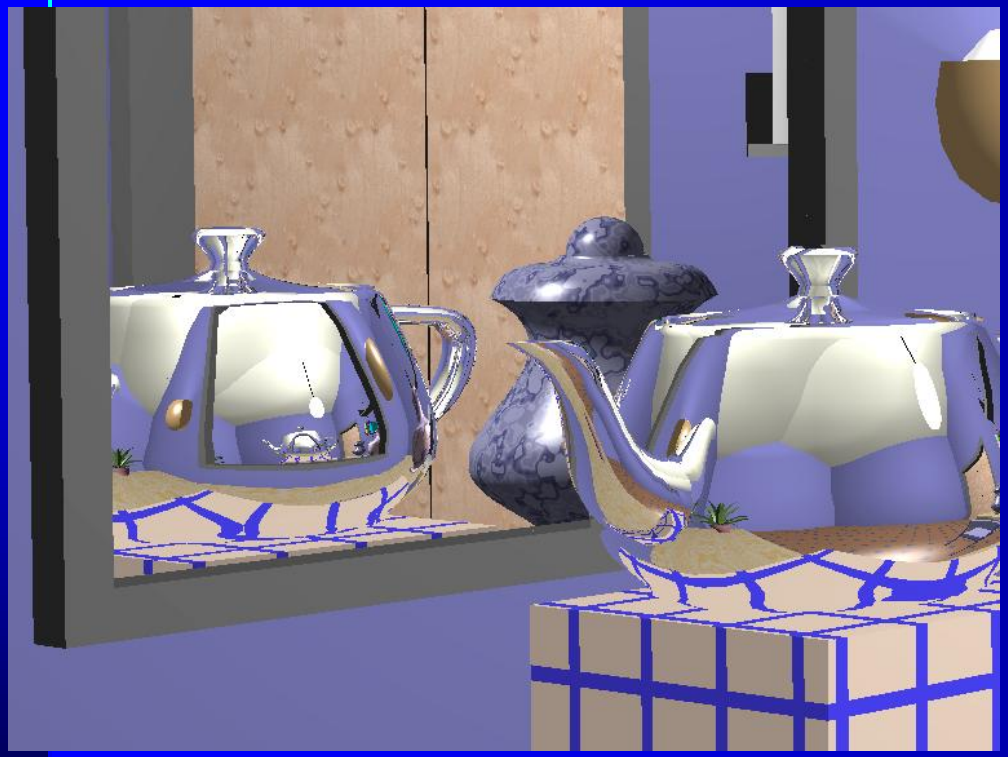
Note ambient shade on reflection of teapot in reflection of mirror in teapot.

Test Scene



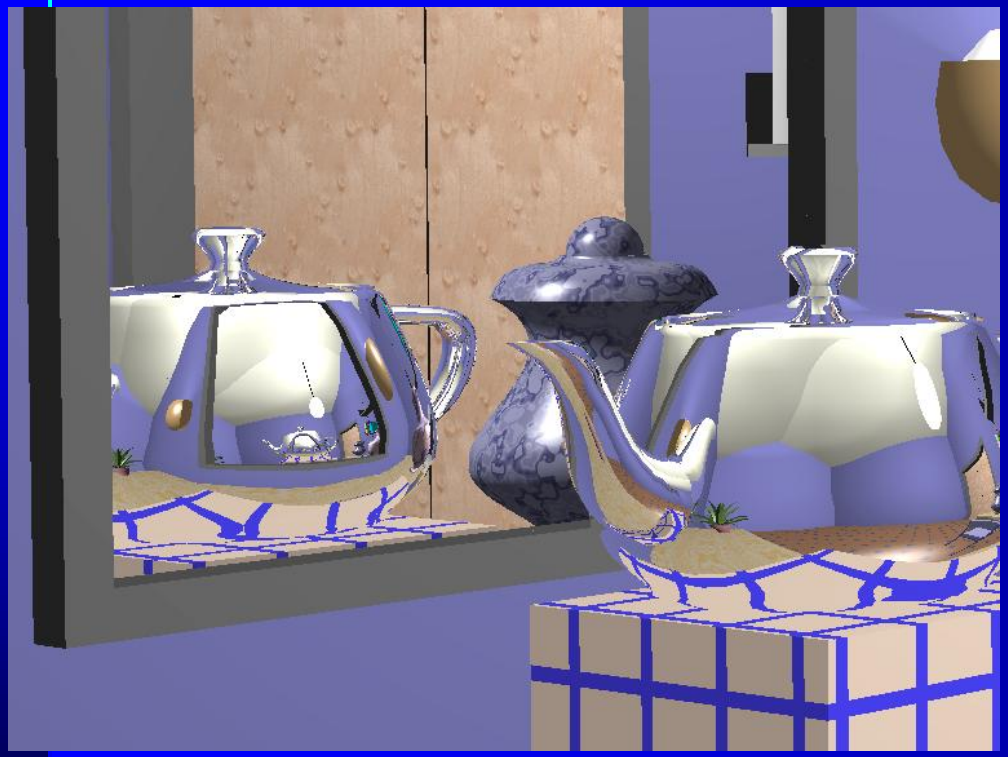
Ray tree depth 5.

Test Scene



Ray tree depth 6.

Test Scene

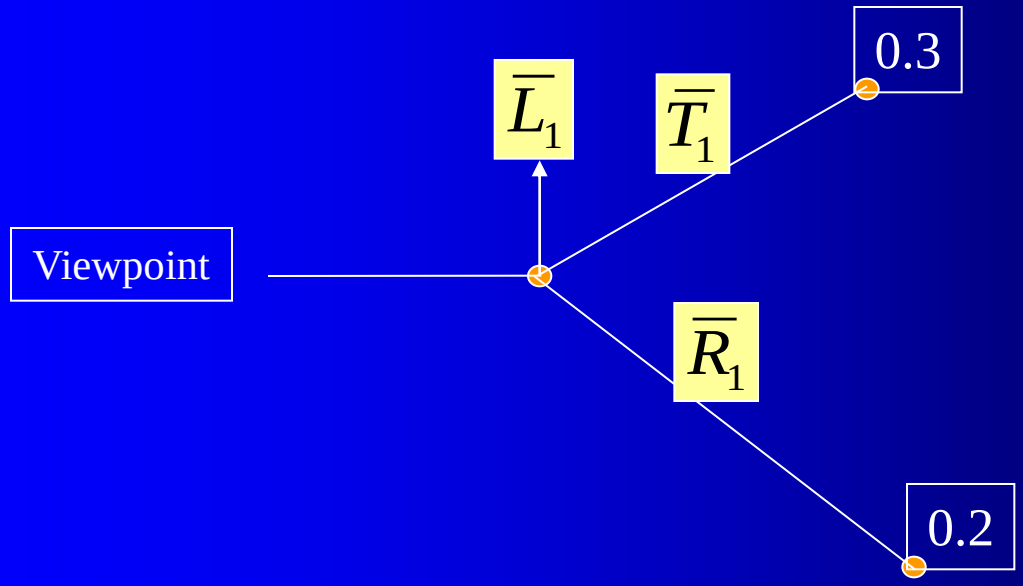


Ray tree depth 7.

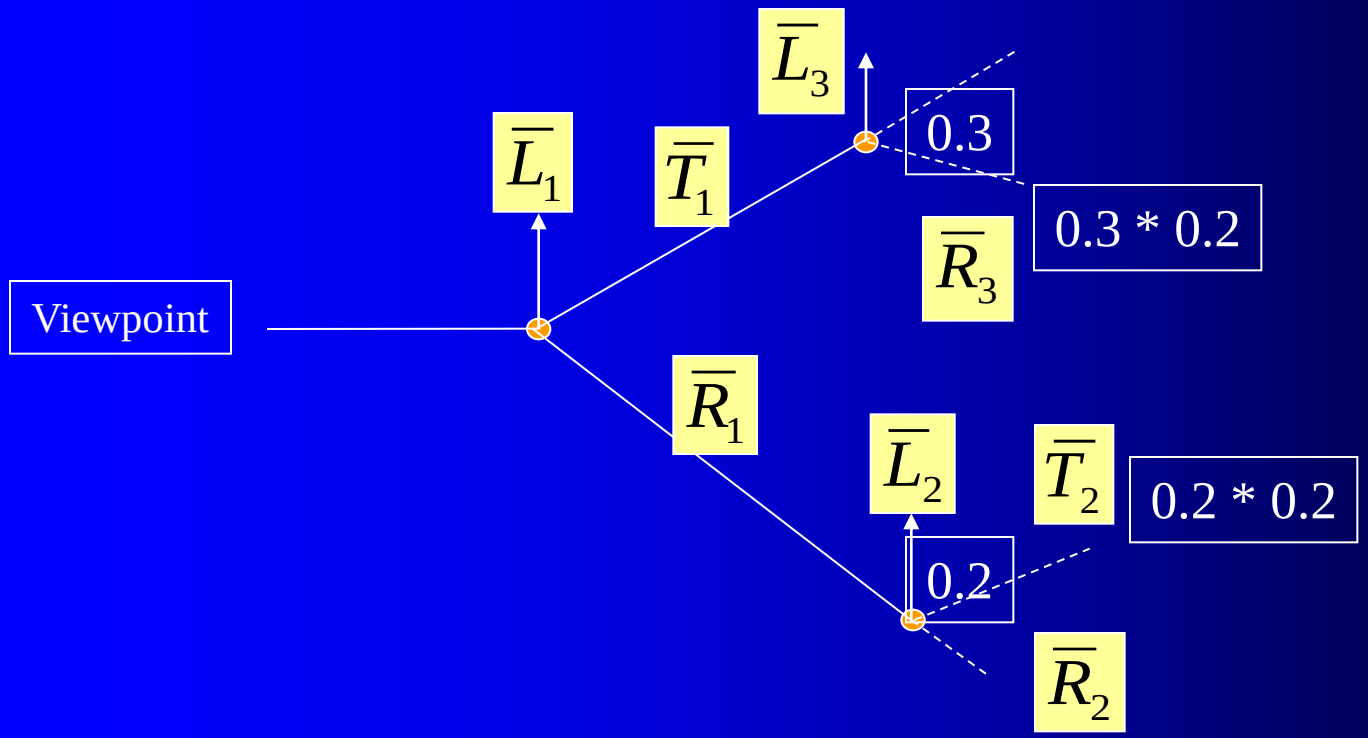
When to stop ?

- Need to know when to stop the recursion.
- Can define a fixed depth.
- Hall introduced *adaptive tree depth control.*
  - Calculate maximum contribution of a ray to a pixels final value.
  - Multiply contribution of ray's ancestors down the tree.
  - Stop when below some threshold, perhaps stack overflow.
  - May miss major contribution this way (culled bright pt)

Adaptive tree depth control



Adaptive tree depth control



Global vs. Local illumination

- In both an object hit by a ray, if lit by a light source, is illuminated by a local illumination model, i.e with specular, diffuse & ambient terms.
- Global: a reflected ray, a shadow feeler, and a transmission ray (if appropriate) are also cast into the scene.
- Phong term only reflects light source.
  - Need to adjust local illumination terms to normalise total light values.
  - Inconsistent if local and global specular terms used together as local term spreads light source, global term does not.

Increased reflectivity

Increased transmissivity

Incorrect result

- Effect of not normalising reflection and transmission – light appears to be created.
  - Reflection & transmission = 100%

Problems with Ray-tracing

- A serious problem with Ray tracing is rays are traced from the eye.
  - Refraction is not physically correct.
- Shadow rays are cast only to light sources
  - Lights reflected in mirrors do not cast shadows
  - Shadows of transparent objects don't exhibit refraction.
  - Still need local illumination for diffuse shading.

Speeding up Ray Tracing

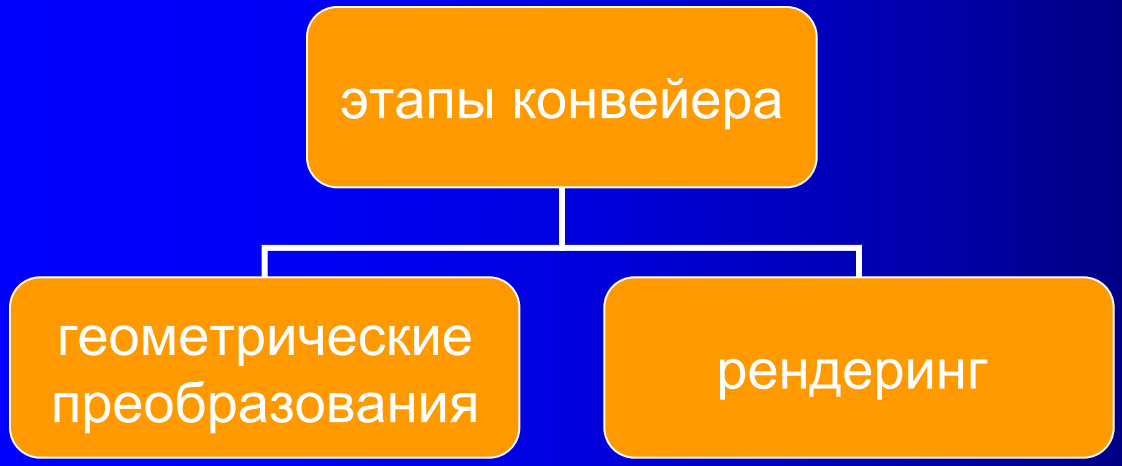
- Ray tracing is slow, not real-time.
- Use appropriate extents for objects.
- Ray tracing is inherently parallel.
- Use item buffers – z-ordered lists, store closest object per pixel.
- Use light buffer – z-ordered list per light ray used for shadowing.

Формирование реалистических изображений

Стадии построения реалистических изображений при помощи конвейера.

Основные этапы построения

- **Графический конвейер** – это логическая совокупность вычислений, которые выполняются последовательно и дают на выходе синтезируемую сцену. Вычисления в конвейере разделены на несколько этапов, в каждом из которых аппаратно или программно выполняется определенная функция.



- **Большинство приложений используют стандартную схему построения 3D сцен. Для сокращения временных затрат на реализацию используются аппаратные возможности с применением прикладных интерфейсов API.**

Стадия геометрических преобразований

- **Разбиение геометрических моделей на примитивы.** На начальном этапе производится описание трехмерной сцены, изображение которой необходимо синтезировать. Трехмерные объекты обычно конструируются из графических примитивов, преимущественно из треугольников, поскольку треугольник является простейшим полигоном, однозначно задает простейшую плоскость в пространстве и гарантировано обеспечивает решение задачи декомпозиции. Любая поверхность может быть аппроксимирована сеткой треугольников и, если такая сетка достаточно хорошо составлена, то с ее помощью можно представить любую поверхность с необходимой точностью.
- **Модельные преобразования.** Включает аффинные операции переноса, поворота и изменение масштаба. Преобразования позволяют перемещать объекты в сцене и манипулировать сюжетом.

Стадия геометрических преобразований

- **Освещение.** На этом этапе выбираются модели освещения и вычисляется освещенность объектов. Модель освещения описывает тип используемых источников света. Освещенность и тонирование поверхностей объектов определяется расположением источников света и их типом, а также оптическими свойствами материала, из которого выполнены поверхности. Общепринятые модели освещения включают рассеянный свет, направленный и точечный источник света.
- **Видовые преобразования.** Здесь определяются новые координаты для всех вершин примитивов, исходя из положения наблюдателя и направления его взгляда. Сцена проецируется на экранную систему координат. Для отображения трехмерного объекта на двумерный экран или другое внешнее устройство используется математическое преобразование, называемое проецированием. Точки, определяющие отрезки прямых, кривые и другие элементы проецируются на двумерную плоскость, при этом виртуальная проекционная плоскость, называемая картинной плоскостью, помещается между объектом и наблюдателем, перпендикулярно направлению взгляда. Проводятся проекционные линии от точек объекта к наблюдателю. Точки пересечения картинной плоскости с лучами проецирования являются соответствующими точками проекции.

Стадия геометрических преобразований

- **Удаление невидимых поверхностей.** На этом этапе из списка примитивов исключаются полностью невидимые, оставшиеся позади или сбоку поверхности. В программах машинной графики для удаления невидимых частей изображения наибольшего распространения получили метод трассировки лучей, метод построчного сканирования и метод *Z*-буфера.

Стадия рендеринга

- **Рендеринг** – это процесс преобразования объекта или сцены, созданных в приложении трехмерной графики, для вывода на экран. На стадии рендеринга определяются пиксели изображения и их адреса. В отличии от стадии геометрических преобразований, в процессе рендеринга объем операций с плавающей точкой значительно меньше и в основном состоит из простых операций над пикселями.

- **Наложение текстуры или текстурирование.** Это этап, посредством которого на поверхность объекта накладывается некоторое изображение, называемое изображением текстуры. В общем контексте конвейера визуализации этот метод открывает нетривиальные возможности.
  - текстуры можно использовать для того, чтобы показать материал, из которого сделан объект;
  - с помощью текстурирования можно наглядно представить физические свойства объектов в приложениях научной визуализации. Например, данные о температуре кодируются цветом и наносятся на объект, позволяя видеть, как геометрия влияет на протекание процессов теплопереноса;
  - текстуры дают возможность моделировать световые эффекты, например отражение, при создании фотореалистичных изображений.

Стадия рендеринга

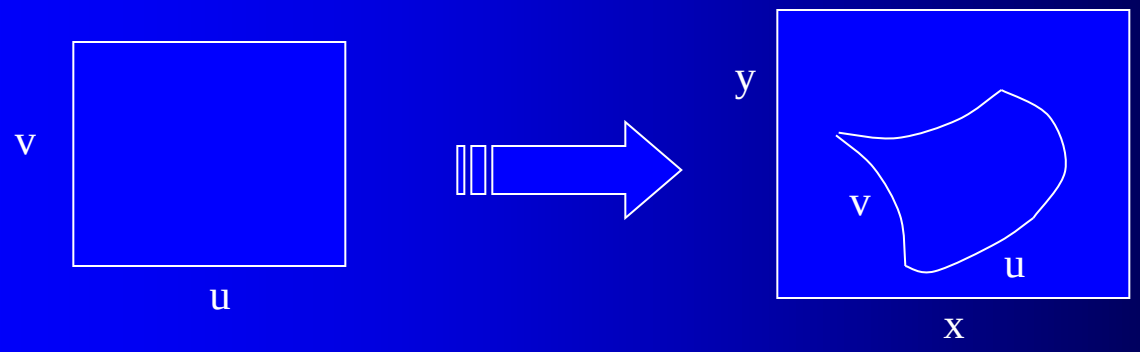
- **Закраска примитивов.** Для того, чтобы создавать правдоподобные изображения применяют специальные алгоритмы закраски для имитации неравномерного освещения. Для выполнения этой задачи используют чаще всего три модели: плоское закрашивание, закрашивание по Гуро и по Фонгу.
- **Сглаживание, финальная обработка.** На этом заключительном этапе рендеринга происходит обработка всей результирующей сцены. Поскольку графические устройства является дискретными, то при представлении объекта, имеющего наклонные линии, возникает эффект ступенчатости. Алгоритм сглаживания (antialiasing) снижает проявление данного эффекта. Метод полноэкранного сглаживания (full screen antialiasing) позволяет значительно улучшить качество выводимой сцены за счет расчета изображения с большим расширением и последующим усреднением значений цвета соседних пикселов. Подобным образом устраняют резкие границы между полигональными областями. На этом этапе применяются и другие эффекты, такие как смазывание (smooth), туман (fogging), которые придают сцене большую реалистичность.

Texture mapping

- Method of improving surface appearance by adding details on surface.

Texture mapping

- Image is ‘pasted’ onto a polygon.
- Image is called a *Texture map*, it’s pixels are often referred as a *Texels* and have coordinates (u,v)
- *Texture coordinates* are defined for each vertex and interpolated across the polygon.



Texture mapping = 2D image warp

2D texture Space

Parameterisation

3D Object space

Model Transformation

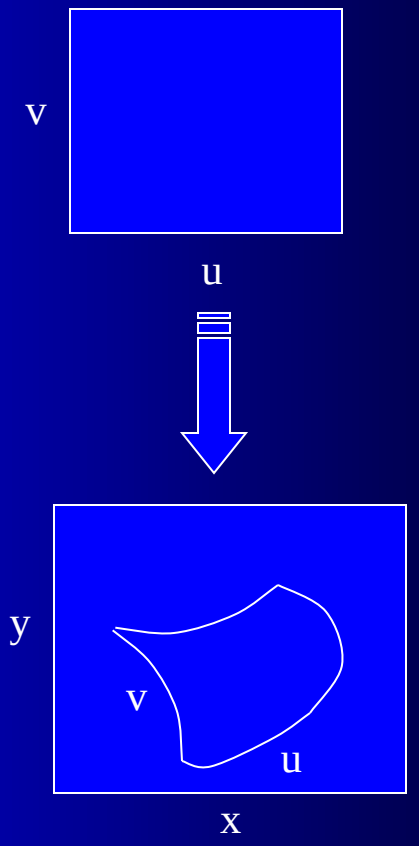
3D World space

Viewing Transformation

3D Camera space

Projection

2D Image Space



Polygonal Texture mapping

Texture to Screen transformation composed of 3 transformations :

1. Parameterization may be written as a linear transformation from texture space to object Space (TO)

2. Modelling and viewing transformation from object space to camera Space (OC)

3. Projection from camera space to screen space (CS) (not shown)

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \underbrace{\begin{bmatrix} . & . & . \\ . & . & . \\ . & . & . \end{bmatrix}}_{TO} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

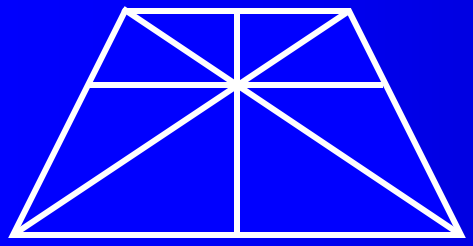
$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \underbrace{\begin{bmatrix} . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \end{bmatrix}}_{OC} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \underbrace{\begin{bmatrix} . & . & . \\ . & . & . \\ . & . & . \end{bmatrix}}_{TS=TO \bullet OC \bullet CS} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

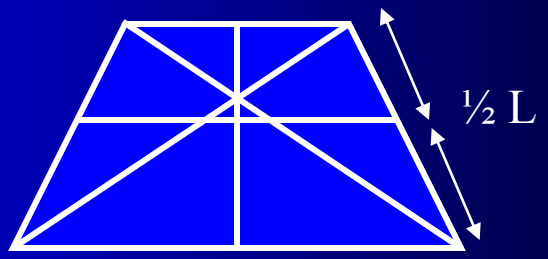
Interpolation of texture coordinates

- Same problem as for Gouraud shading – need to break down into triangles for rotational invariance.
- Linear interpolation leads to incorrect perspective.

Perspective interpolation



Linear interpolation

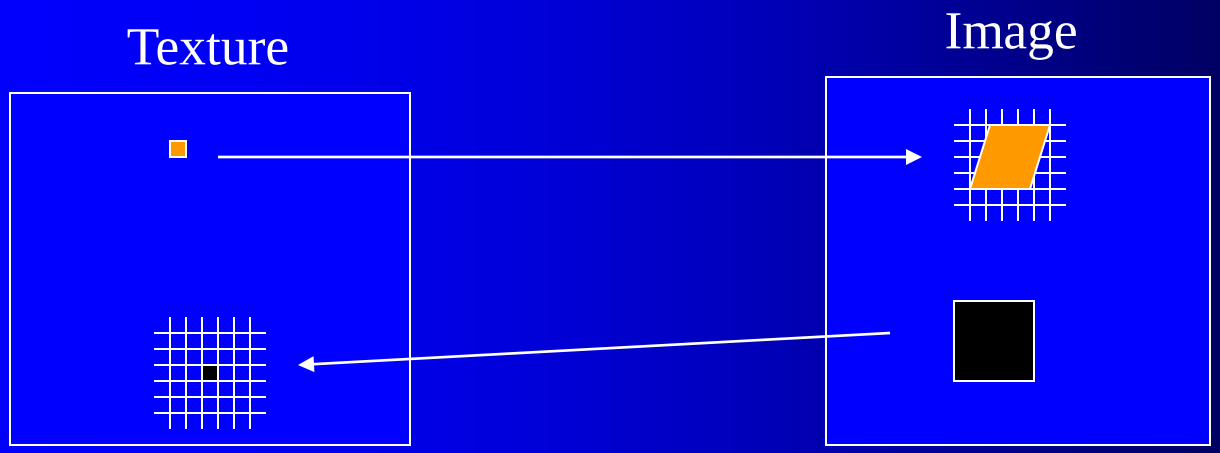


Texture filtering

- Can map texture in 2 ways :
  - From (u,v) space to (x,y) – forward mapping.
  - From (x,y) back to (u,v) – reverse mapping.
- Forward mapping is linear projective map.
- Reverse mapping is inverse – also linear projective map.
  - The inverse of a projective transformation is also a projective transformation.

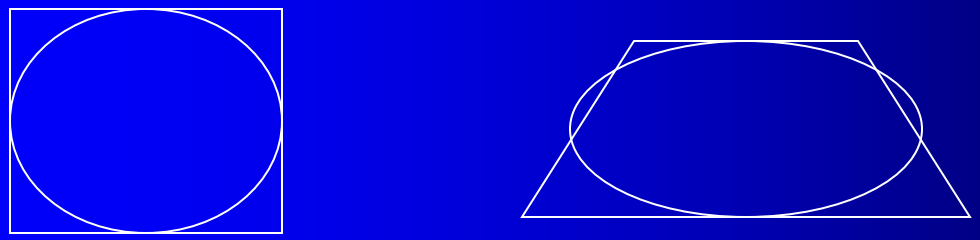
Texture filtering

- Pixel footprint changes from pixel to pixel
  - No single filter.
- Resampling theory : two cases
  - Magnification – interpolate texel values.
  - Minification – average texel values.



Texture filtering

- Need to filter texels to avoid aliasing.
- Useful filter shape is a weighted ellipse whose dimensions are controlled by the projective warp. – EWA filter.



Texture mapping summary

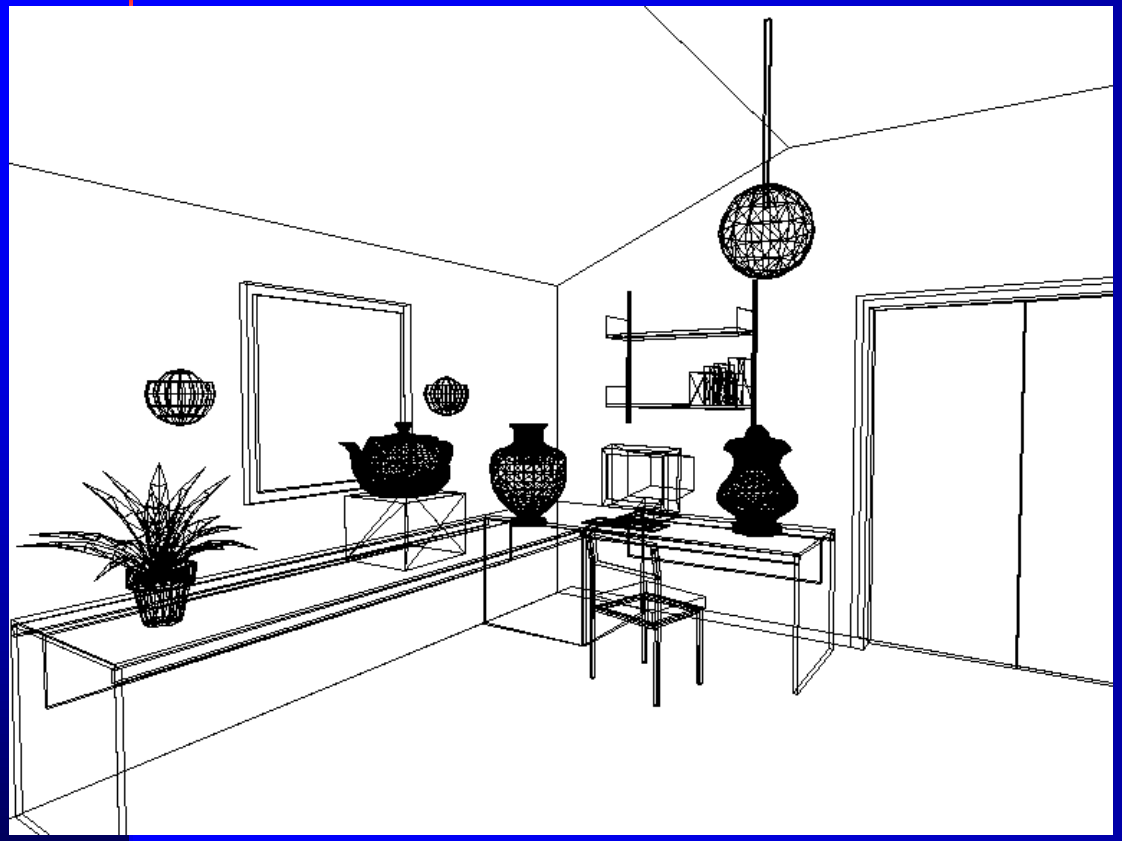
- Paste a 2D image onto projected geometry.
- Required transformation is a 2D image warp.
- Need for projective interpolation.
- Texture filtering.
  - EWA
  - Mip-map.

Texture mapping example

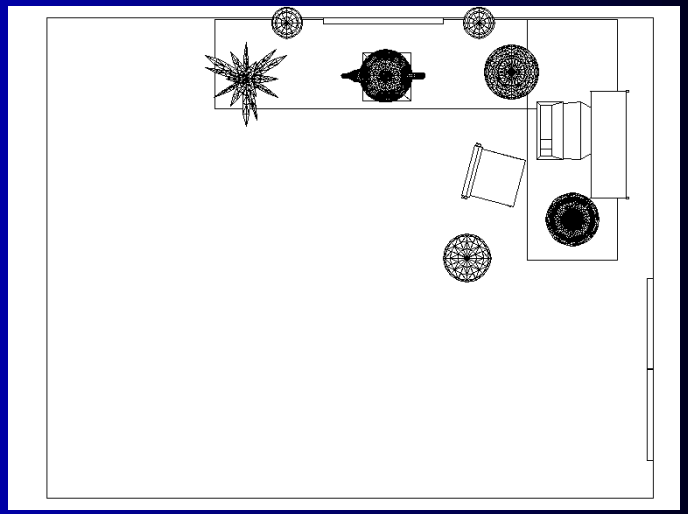
Global Illumination

- Extends the Local Illumination Model to include:
  - Reflection (one object in another)
  - Refraction (Snell's Law)
  - Transparency (better model)
  - Shadows (at point, check each light source)
  - Antialiasing (usually means supersampling)

Wireframe view of a test scene



Orthographic view from above



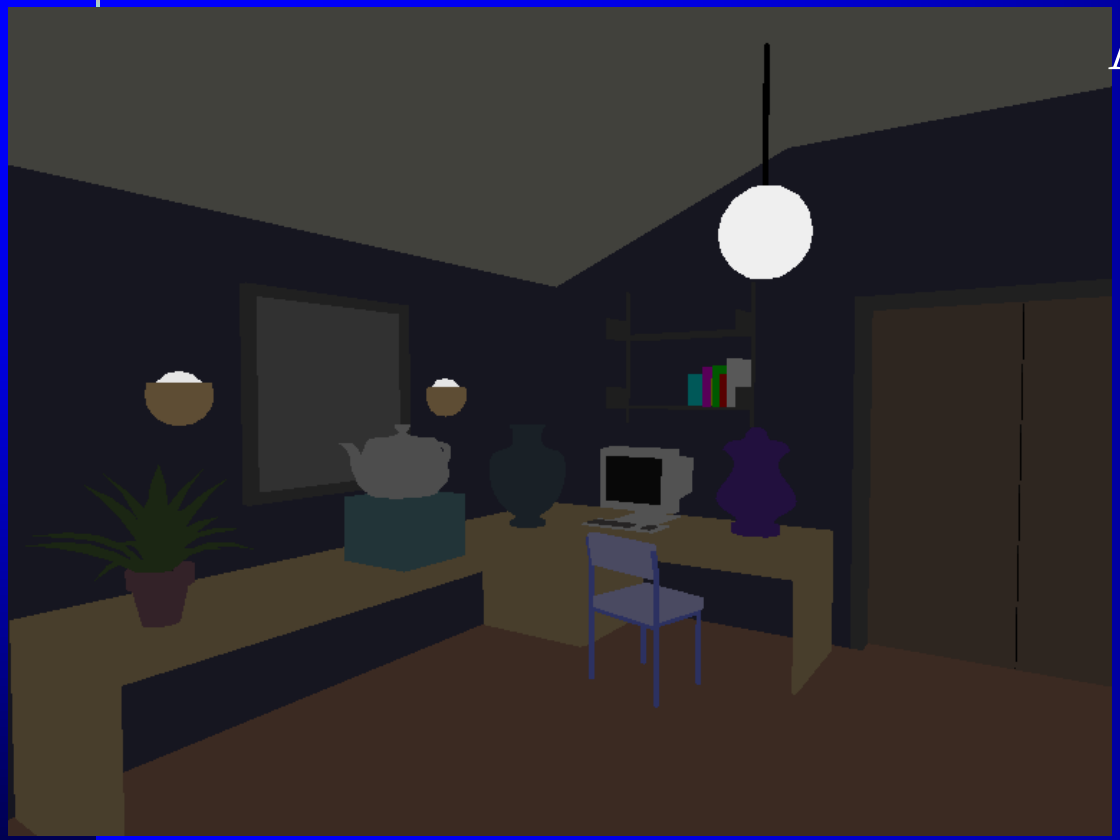
Test Scene

High quality rendering of test scene.

Note :

- Mirror and chrome teapot.
- Shadows on floor.
- Shiny floor.

Locally illuminated test scene



Ambient term only

Locally illuminated test scene

Phong shading.

Ambient and Diffuse terms only

Notes :

- Highlight on wall from light is in the wrong place; screen space interpolation.
- We cannot illuminate the lights with the light sources – wrong side !

Locally illuminated test scene.

Phong shading.

Ambient, diffuse and Specular terms.

Locally illuminated test scene

Flat shading.

Note : Mach bands.

Locally illuminated test scene

Gouraud shading.

Ambient, diffuse and Specular terms.

Note: artefacts on wall.

Solution to Gouraud artefacts

Gouraud shading.

Re-triangulated mesh.