Трассировка лучей. Алгоритмы поиска пересечений

Алексей Игнатенко

Лекция 8

30 ноября 2006

На прошлой лекции

- Полигональные модели
  - Текстура используется как средство передачи освещения и параметров материала
    - Проективные текстуры
    - Световые поля как текстуры
- Точечные модели
  - Подходят для сложных моделей
    - Проще в обработке, быстрее в визуализации
  - Проблема реконструкции поверхности при приближении

На лекции

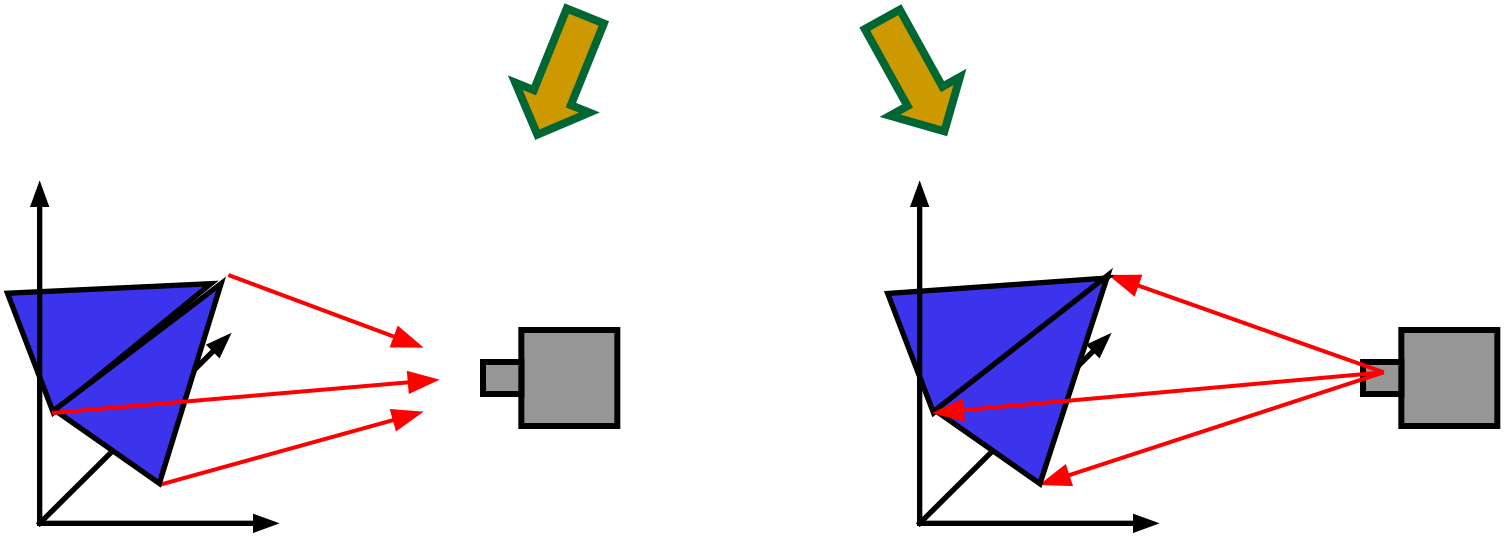
- Трассировка лучей.
- Сравнение с алгоритмами растеризации
- Методы поиска пересечений
- Интерактивная трассировка лучей

Часть 1/3

ТРАССИРОВКА VS. РАСТЕРИЗАЦИЯ

Экранизация в компьютерной графике

Два основных подхода



Растеризация:

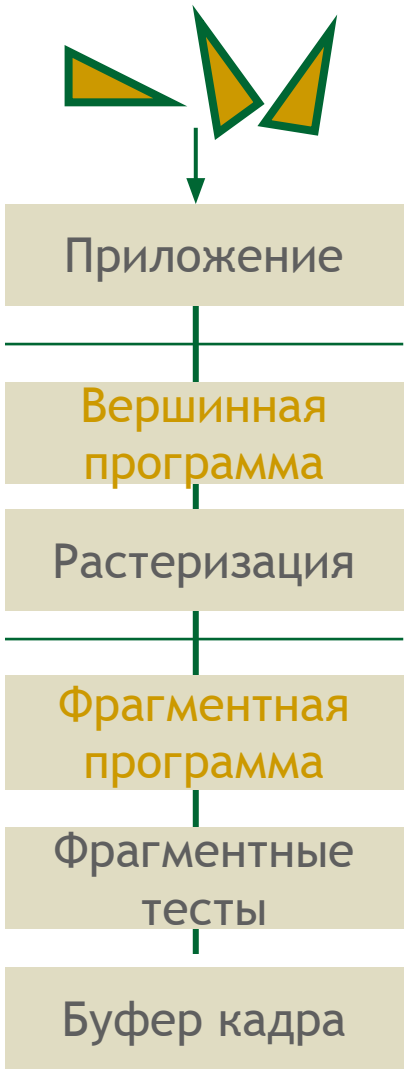
Прямая проекция геометрии

Трассировка лучей:

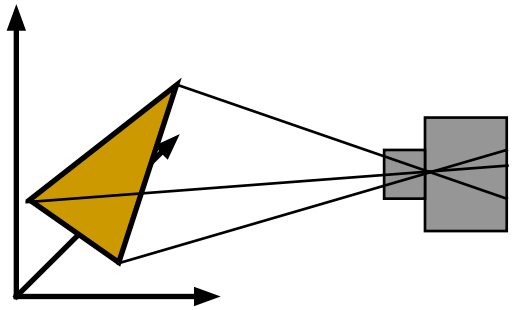
Обратная проекция пикселей изображения

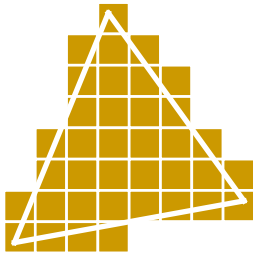
Растеризация

- Конвейер
  - Успешная технология
  - Аппаратная поддержка
- Достоинства
  - Простой и проверенный алгоритм
  - Все быстрее и быстрее
  - Полная программируемость уже скоро



Растеризация: особенности

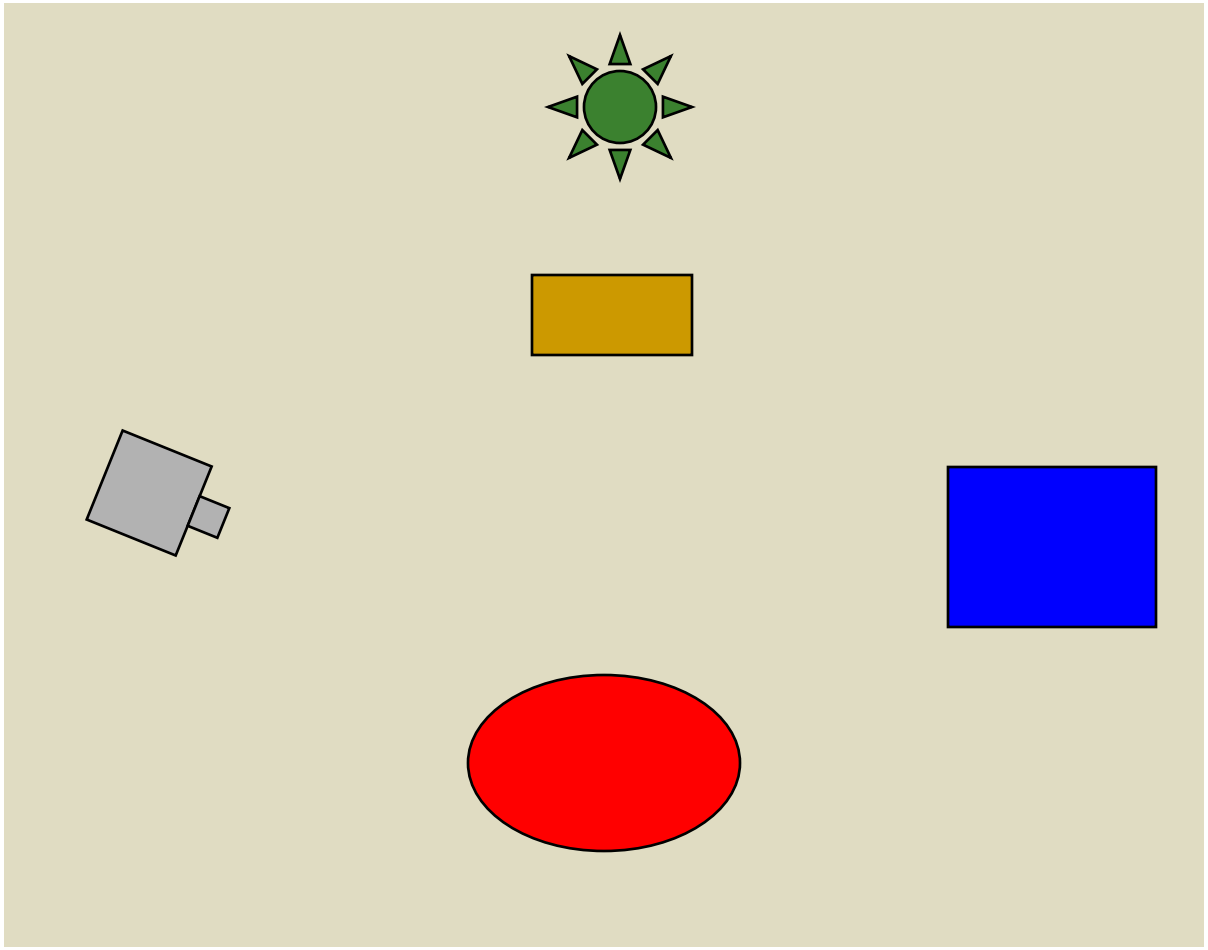


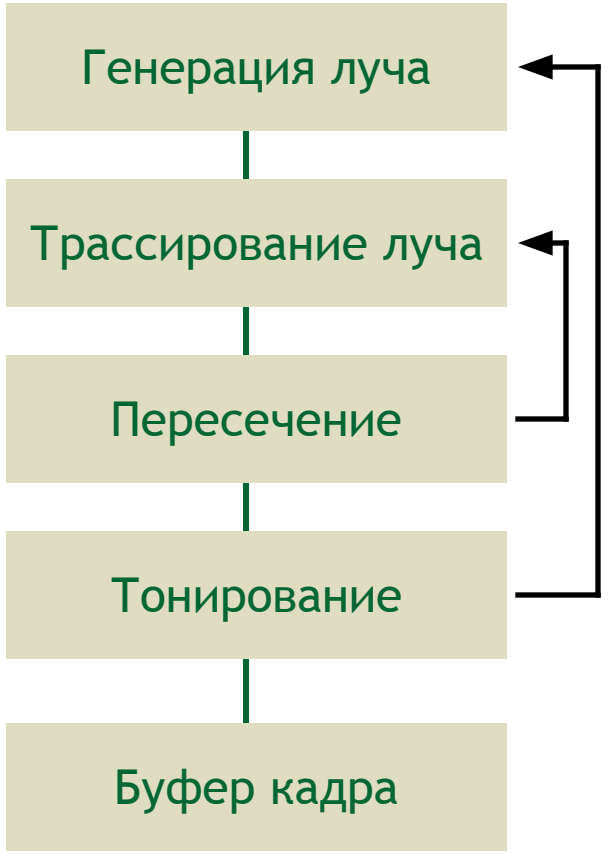


- Базовая операция всей компьютерной графики
  - Построчное сканирование по треугольнику
- Последовательная обработка всех треугольников по одному
  - Невозможно работать более, чем с одним треугольником за раз

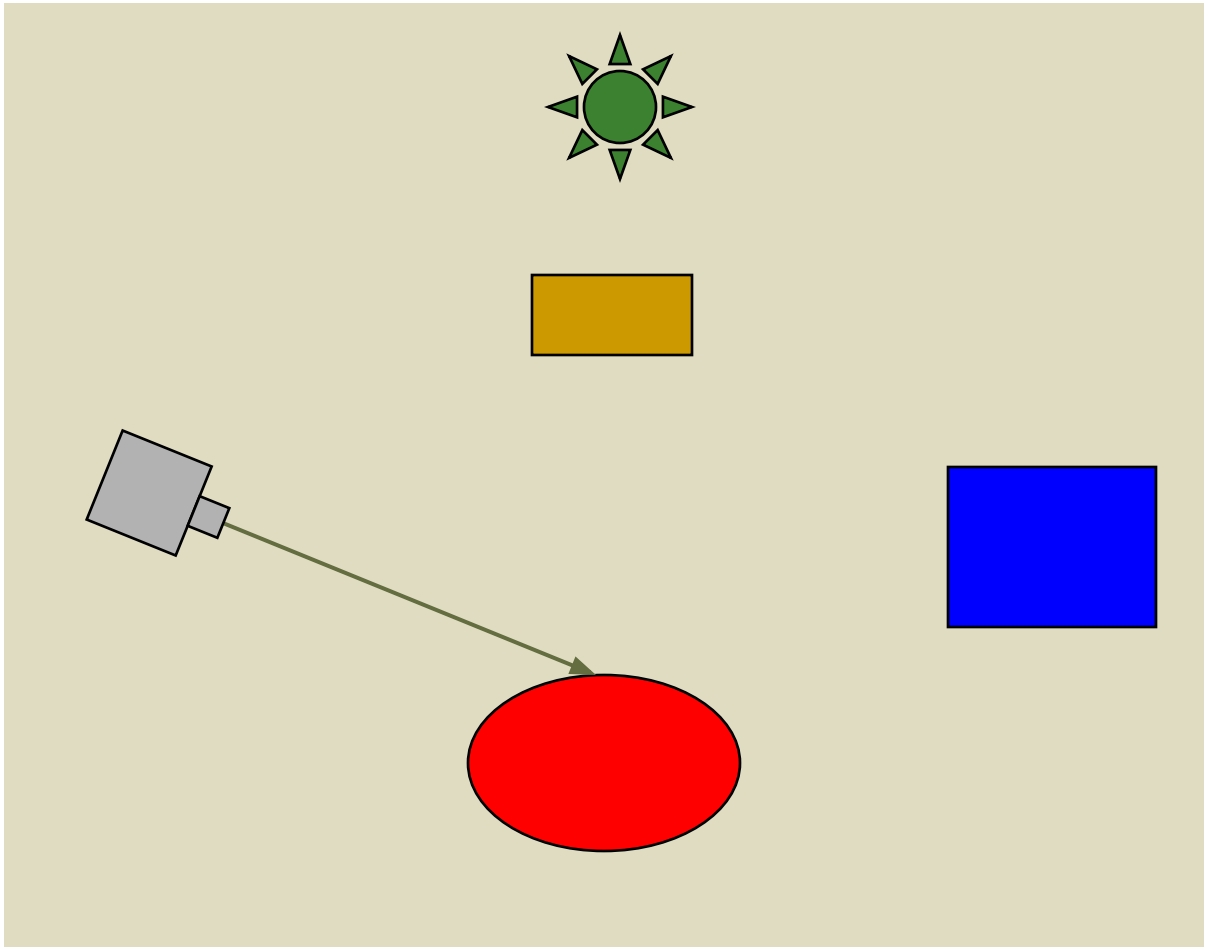
  - Но большинство реалистичных эффектов требуют доступа ко всей сцене: тени, отражения, глобальное освещение!

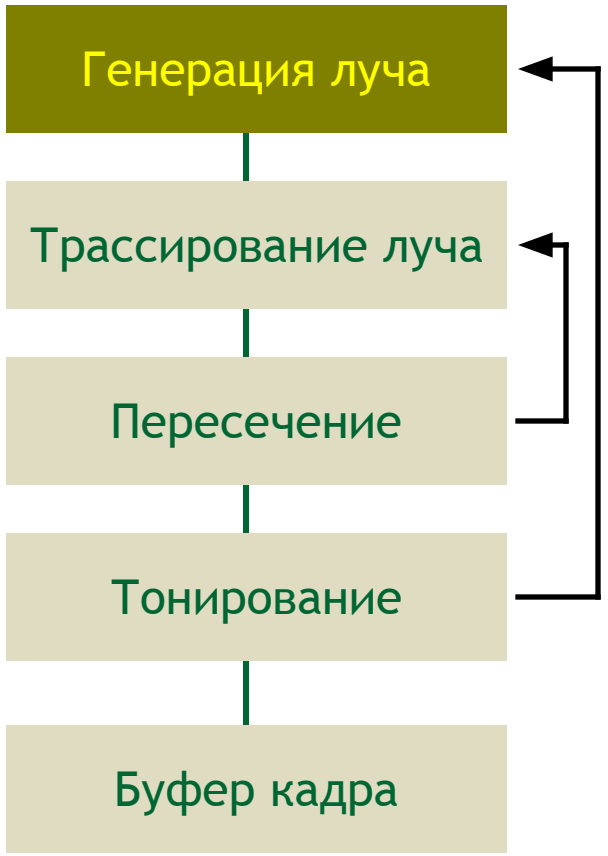
Трассировка лучей



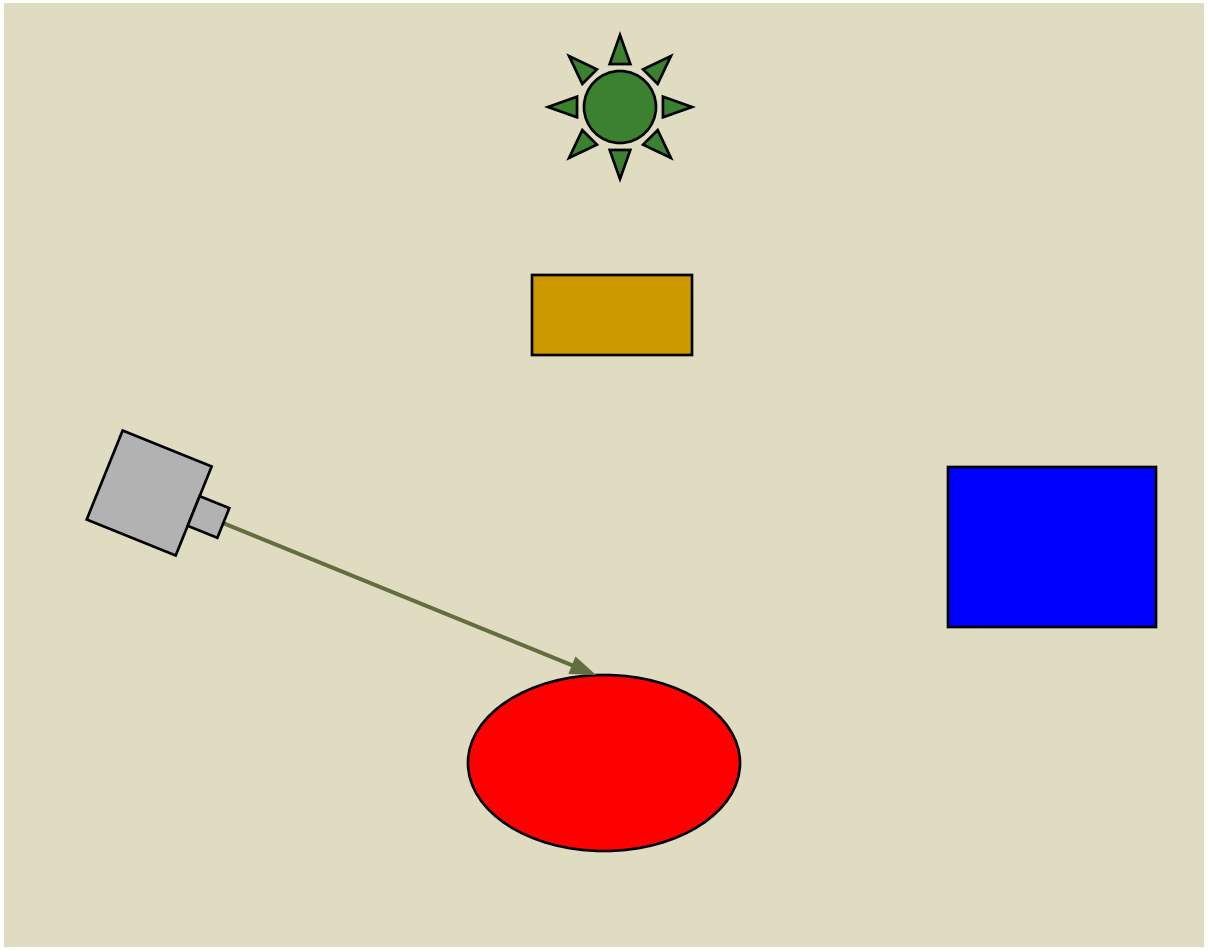


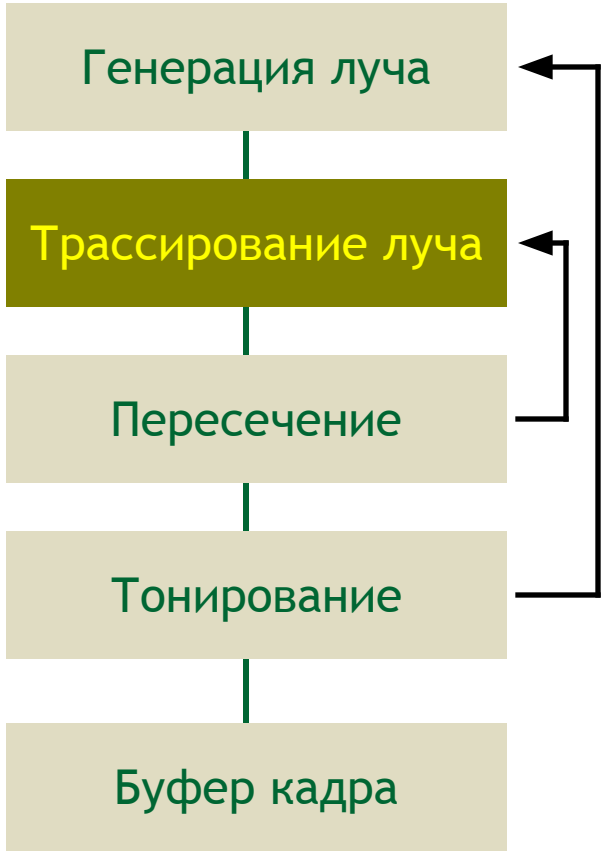
Трассировка лучей



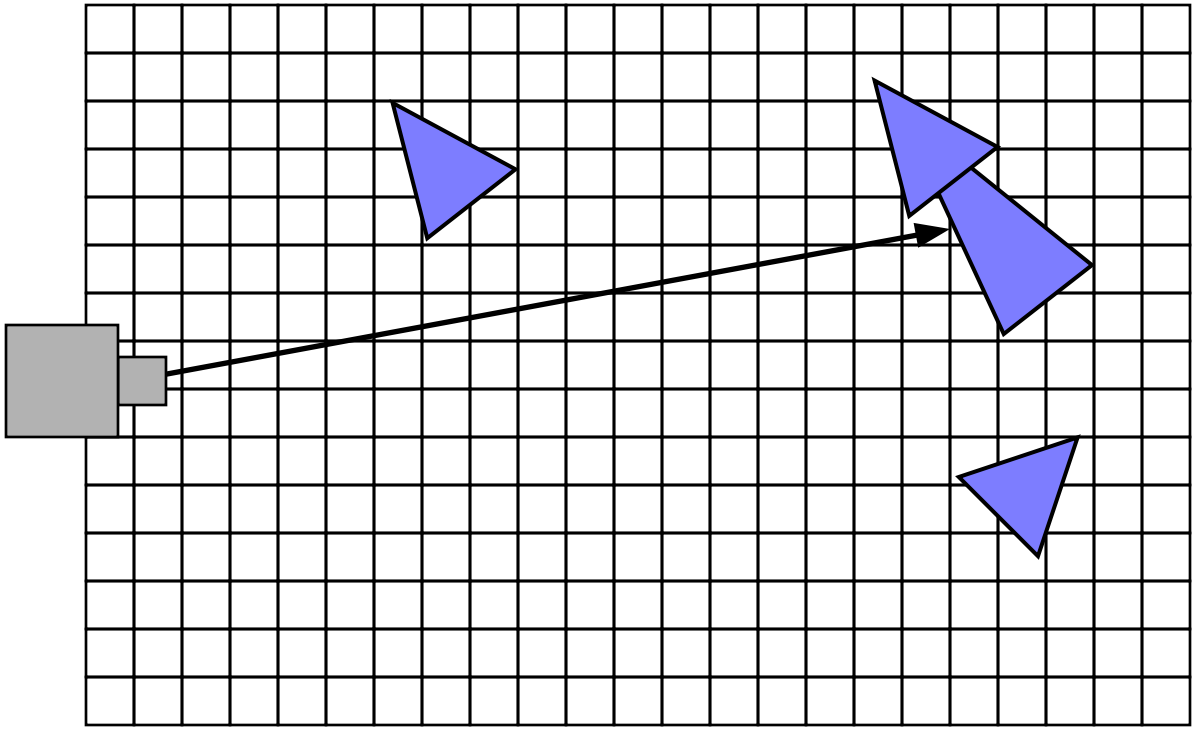


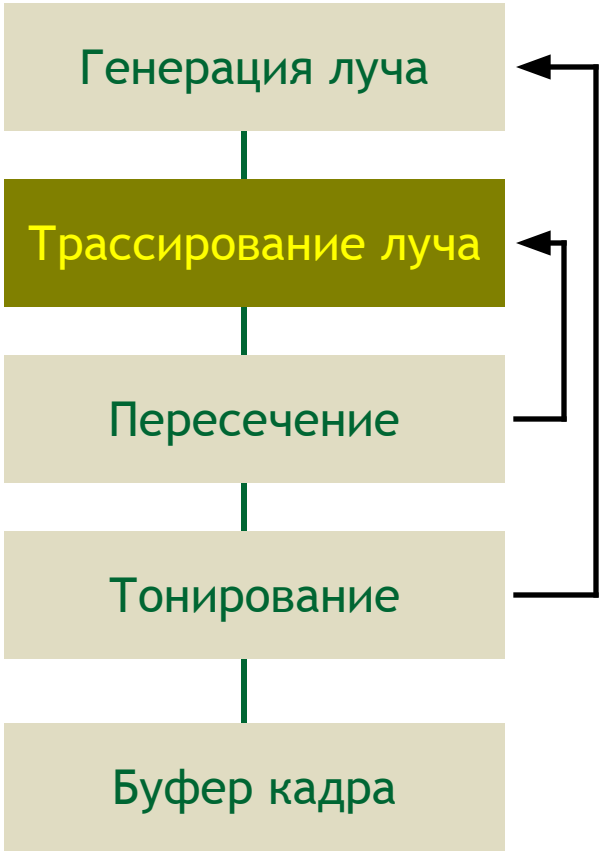
Трассировка лучей



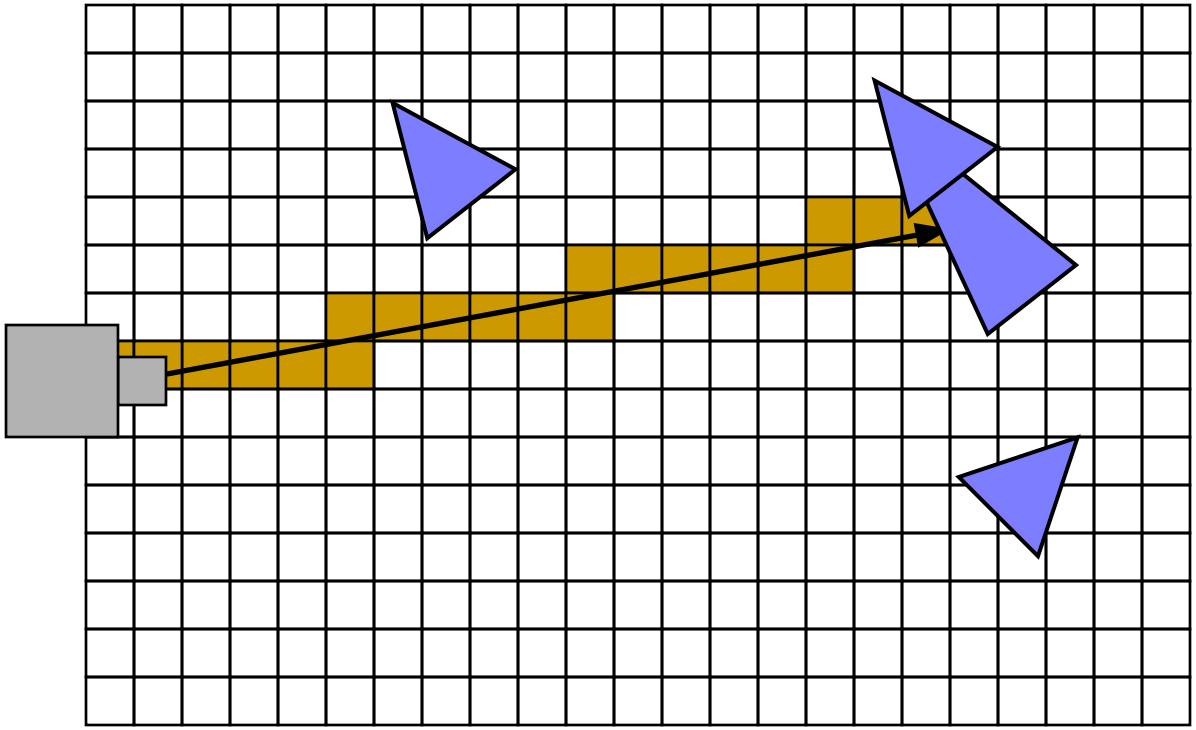


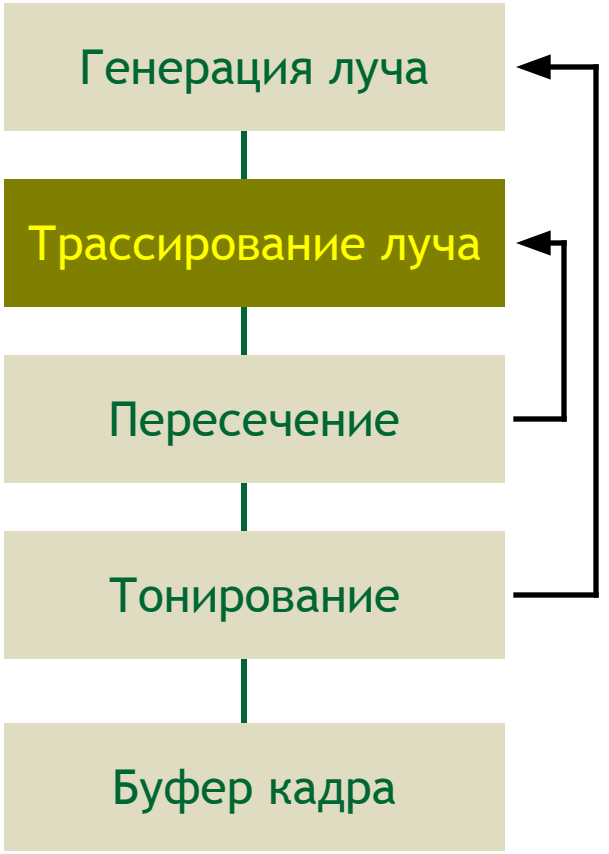
Трассировка лучей: трассирование луча



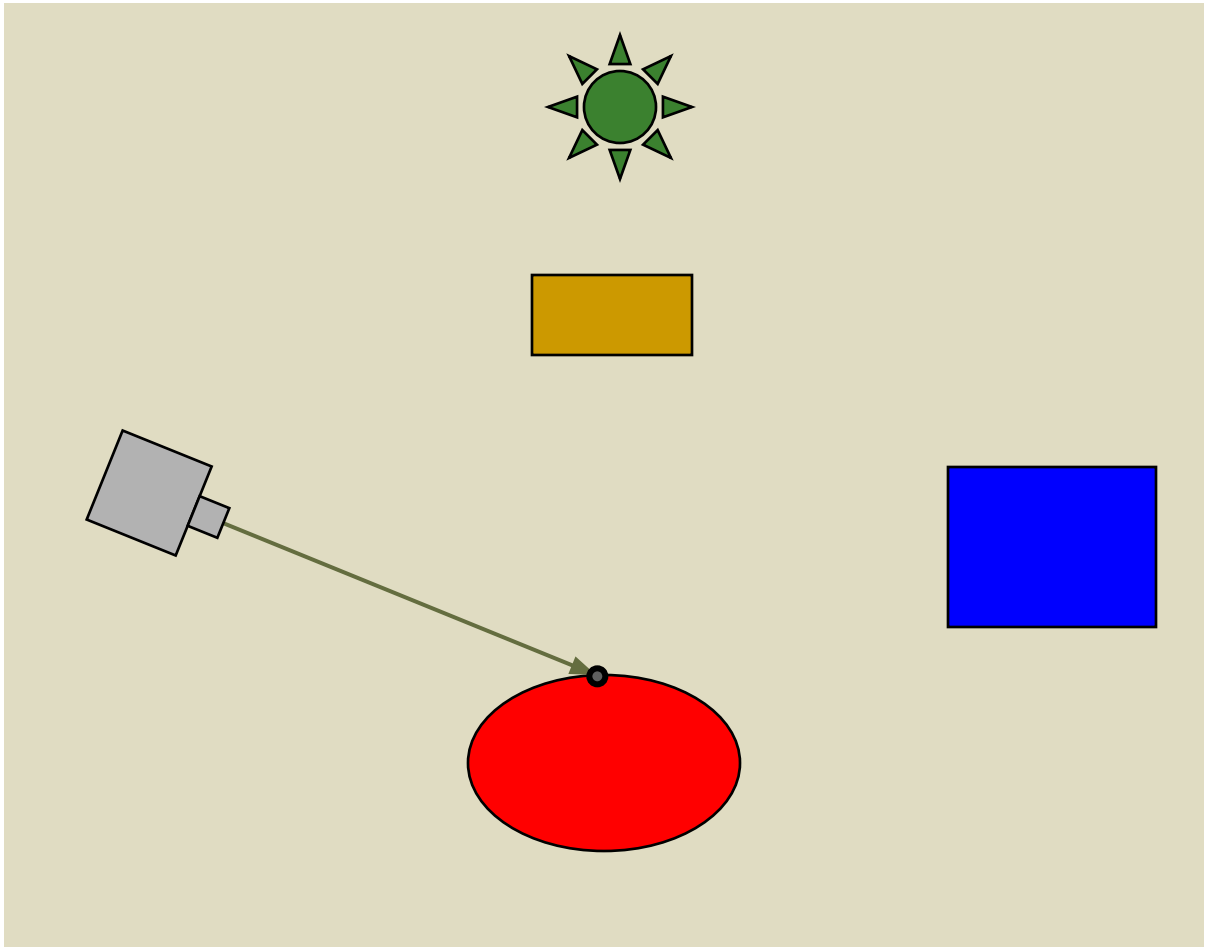


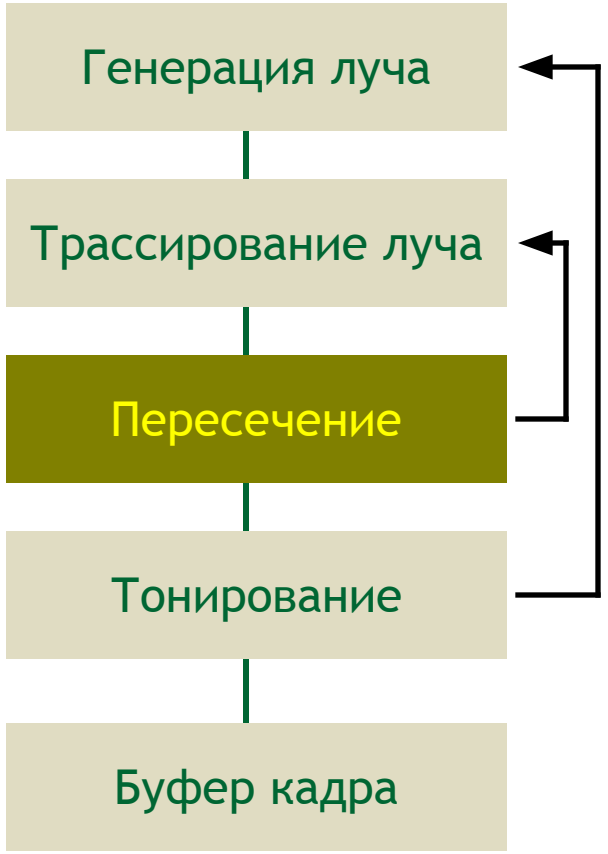
Трассировка лучей : трассирование луча



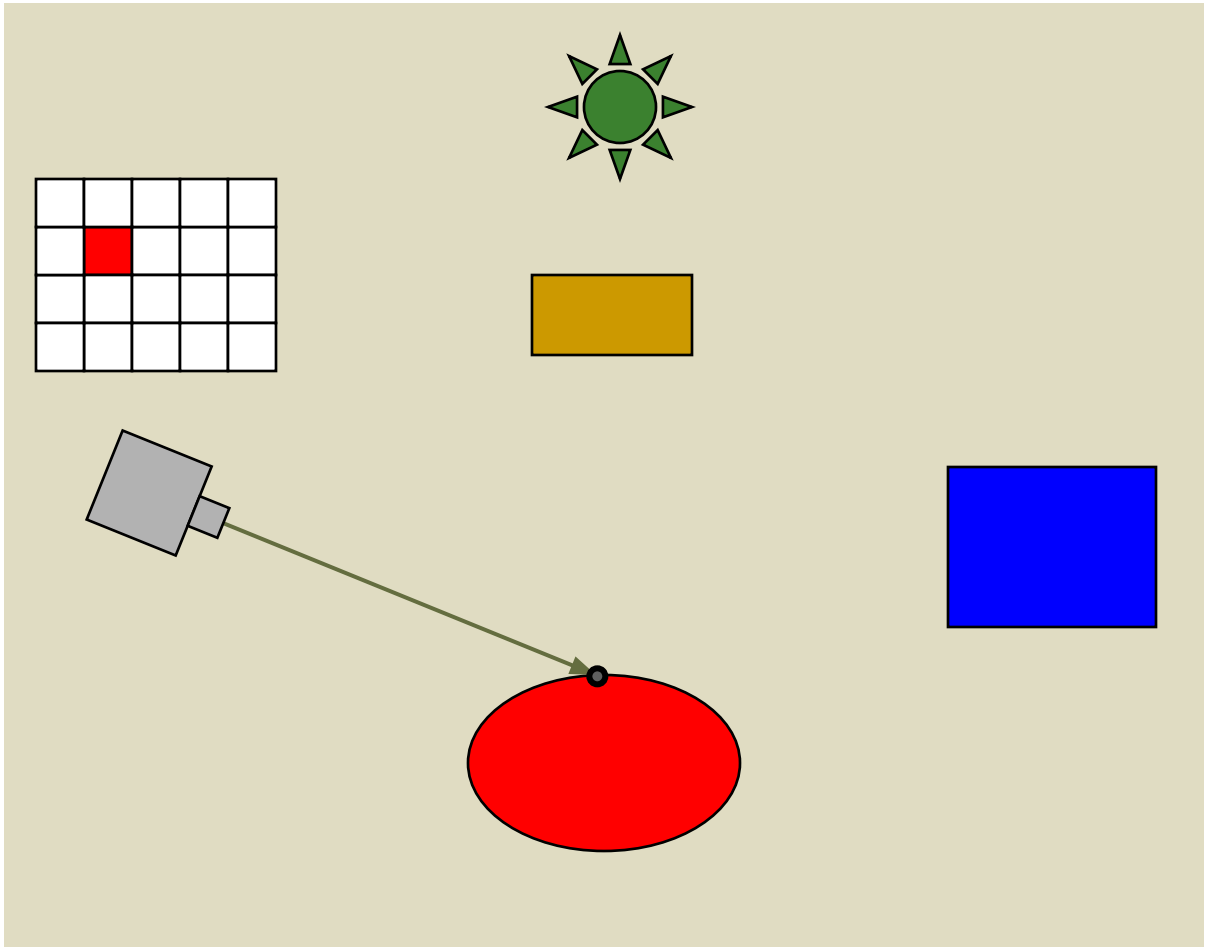


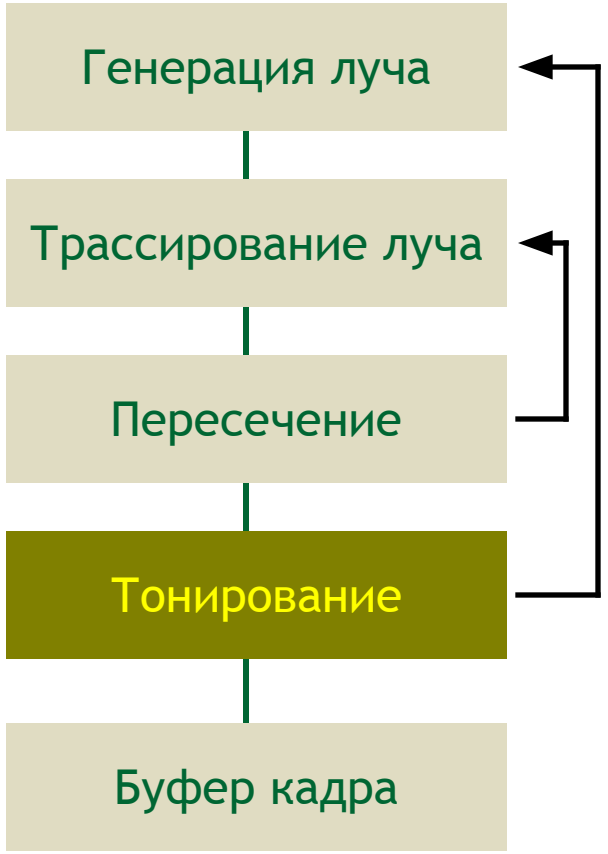
Трассировка лучей



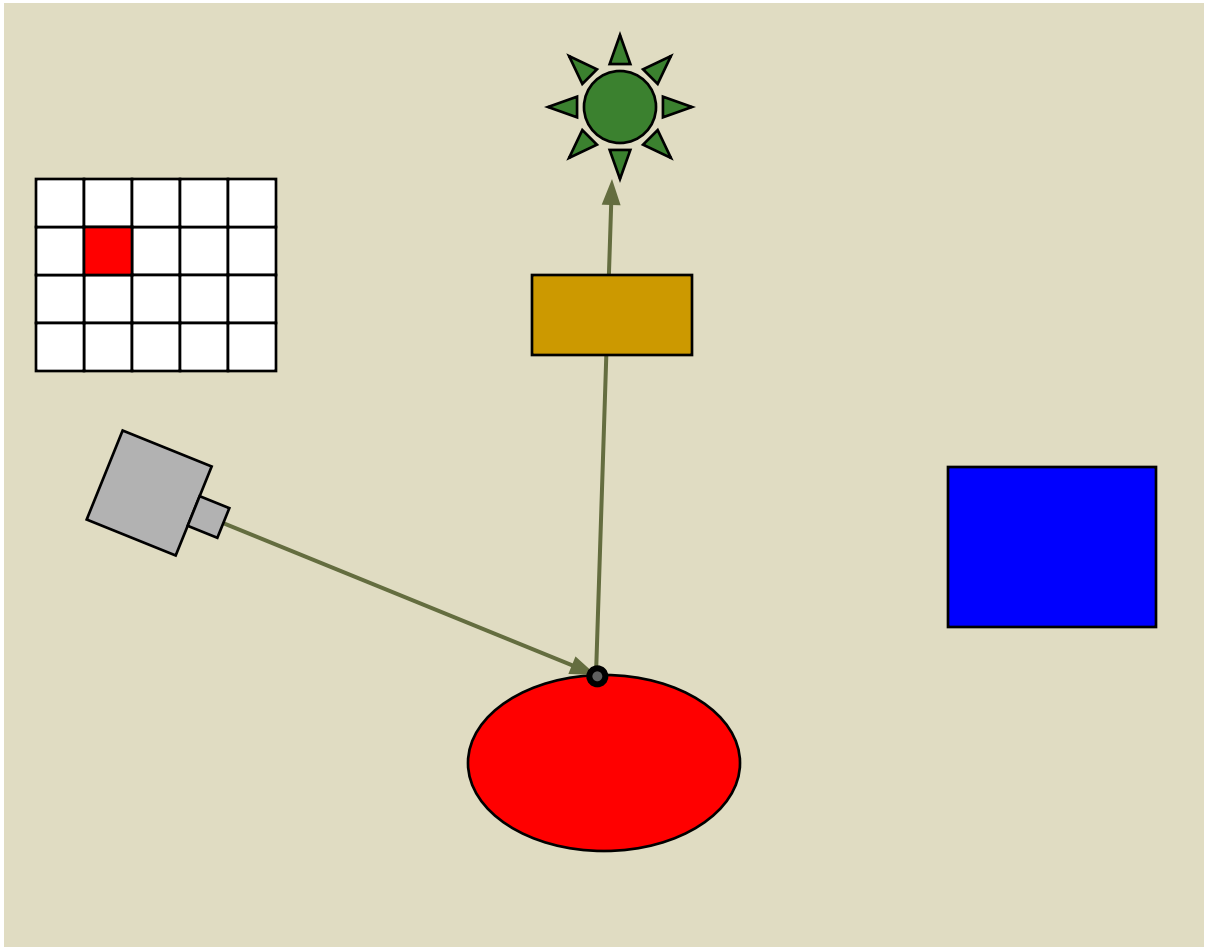


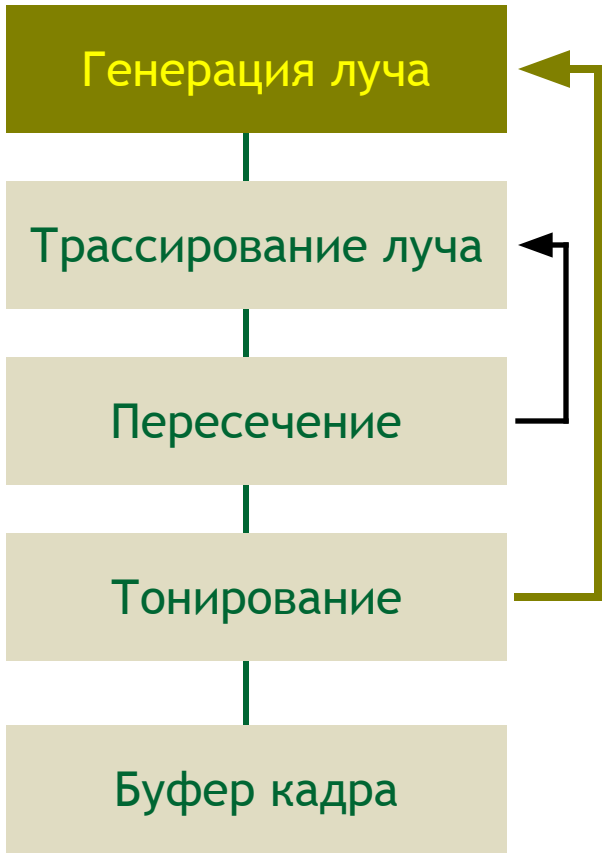
Трассировка лучей



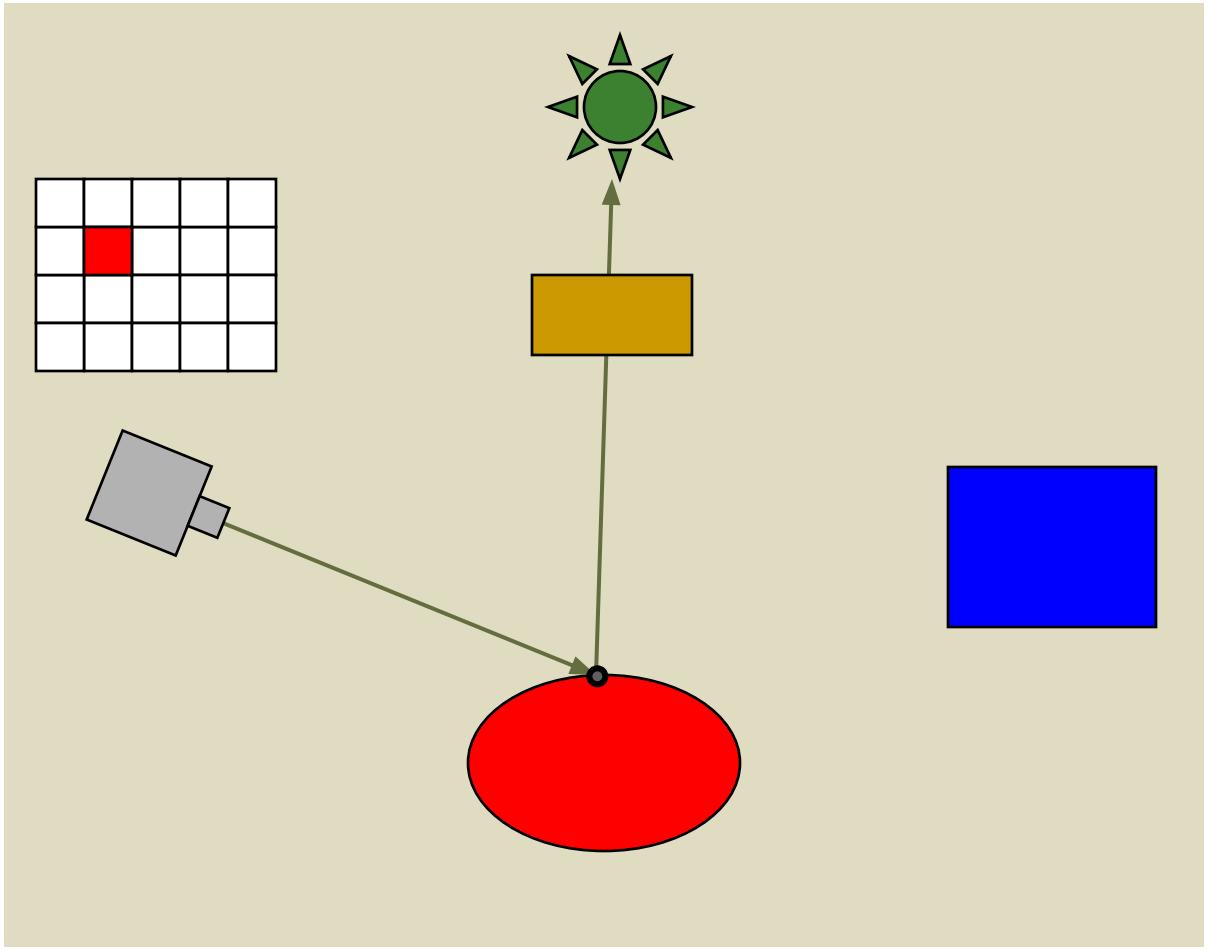


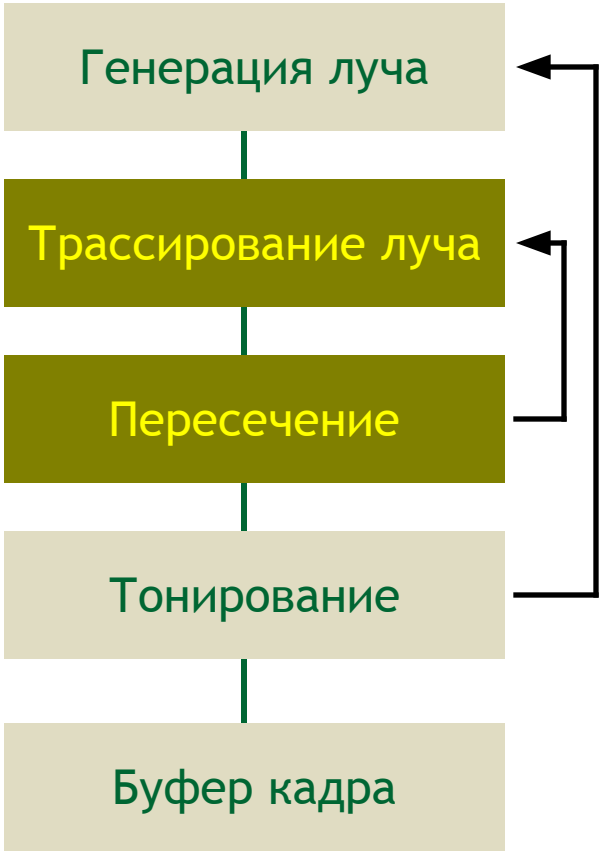
Трассировка лучей



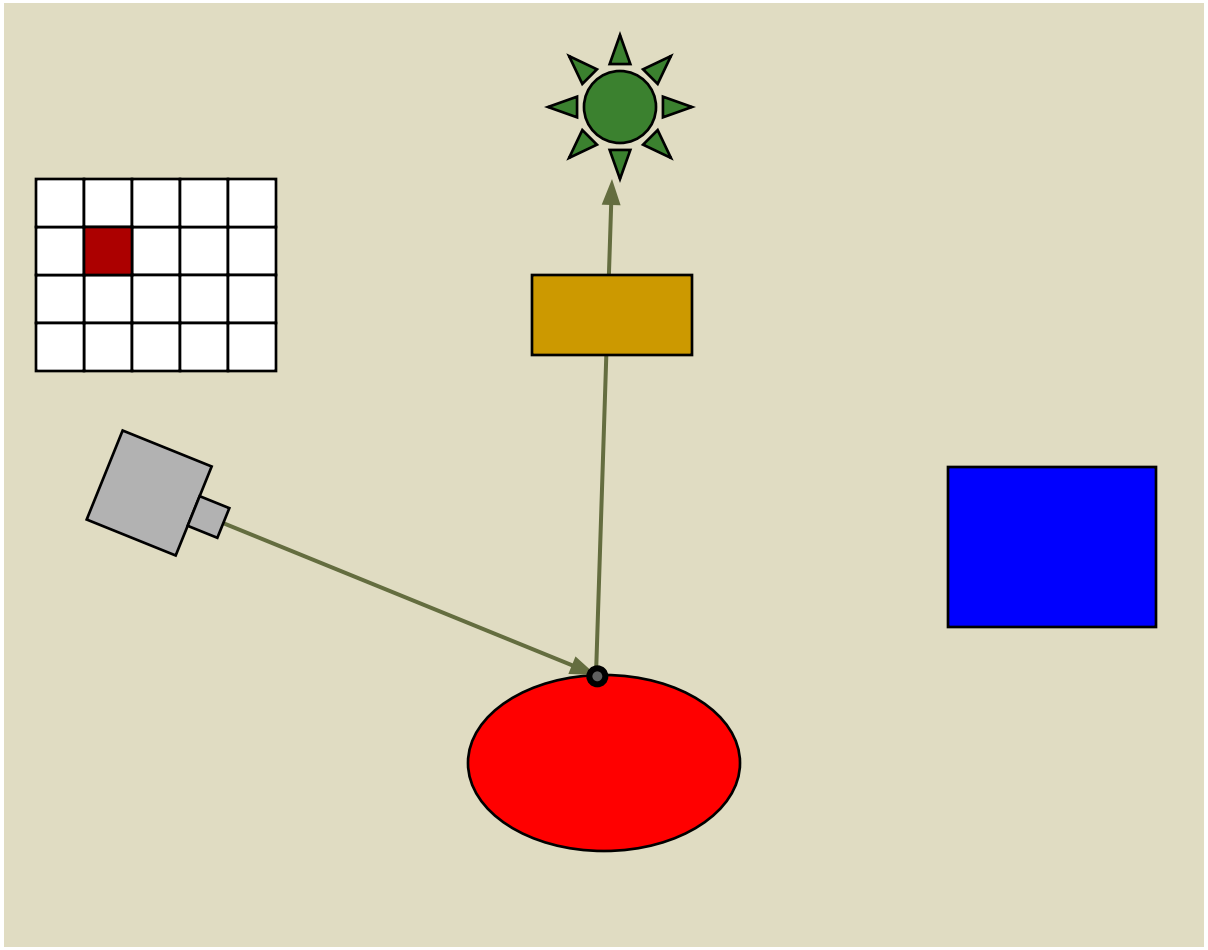


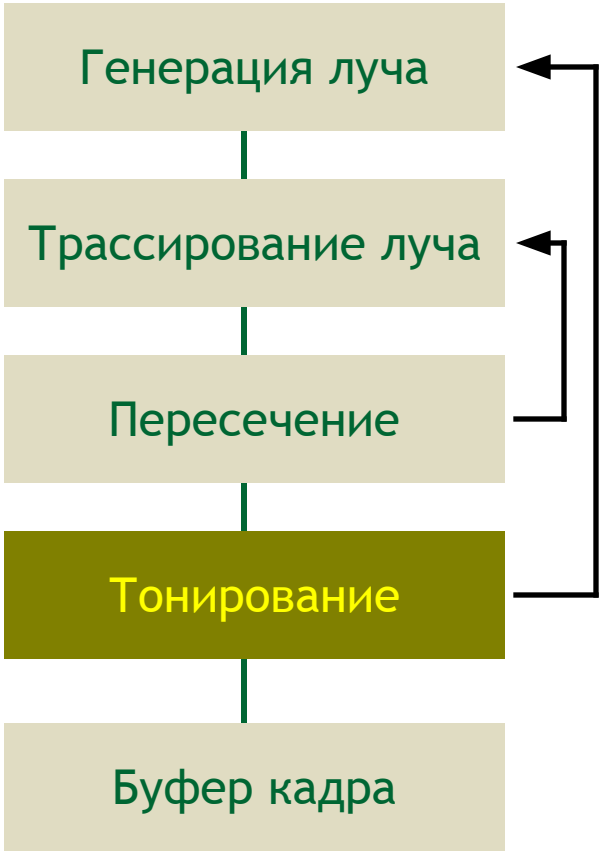
Трассировка лучей



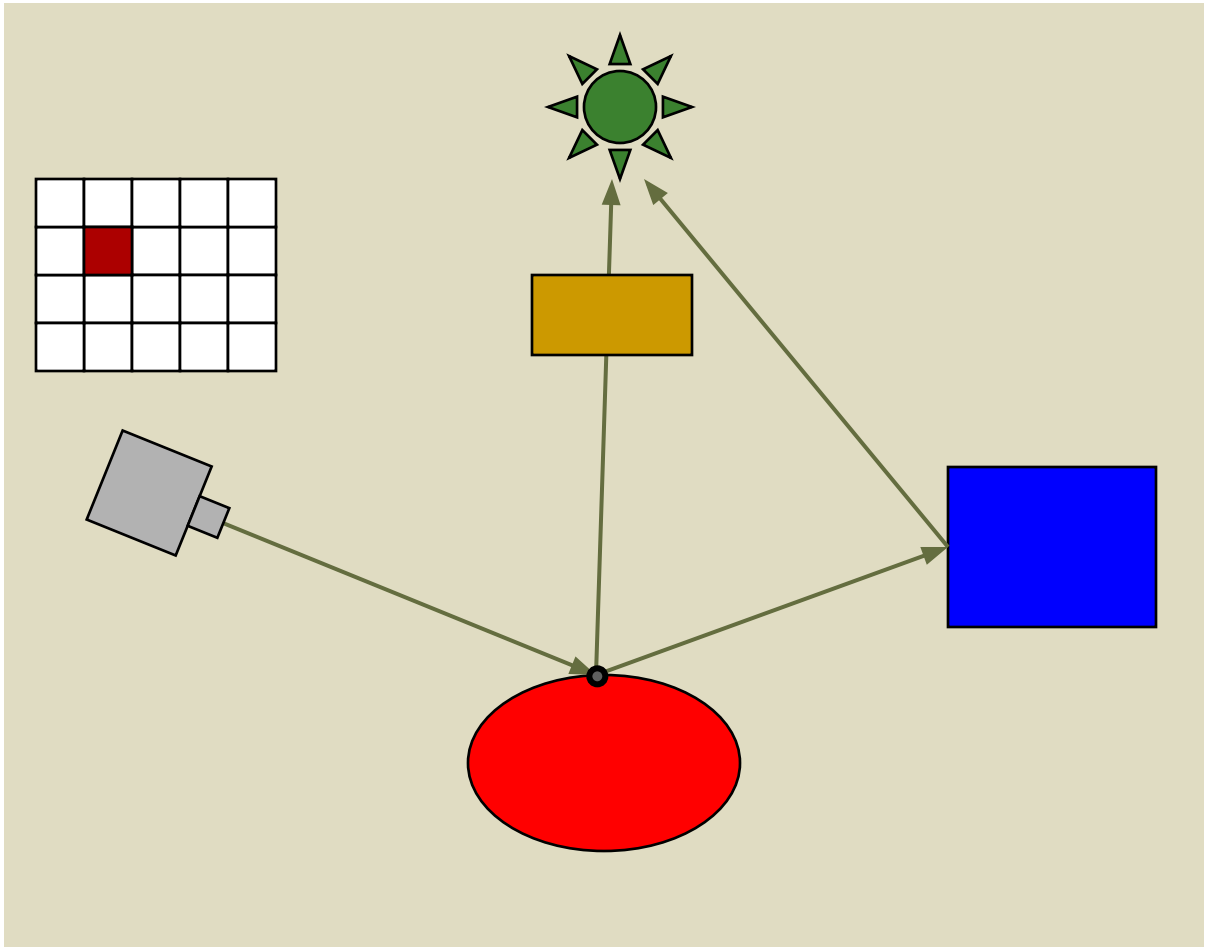


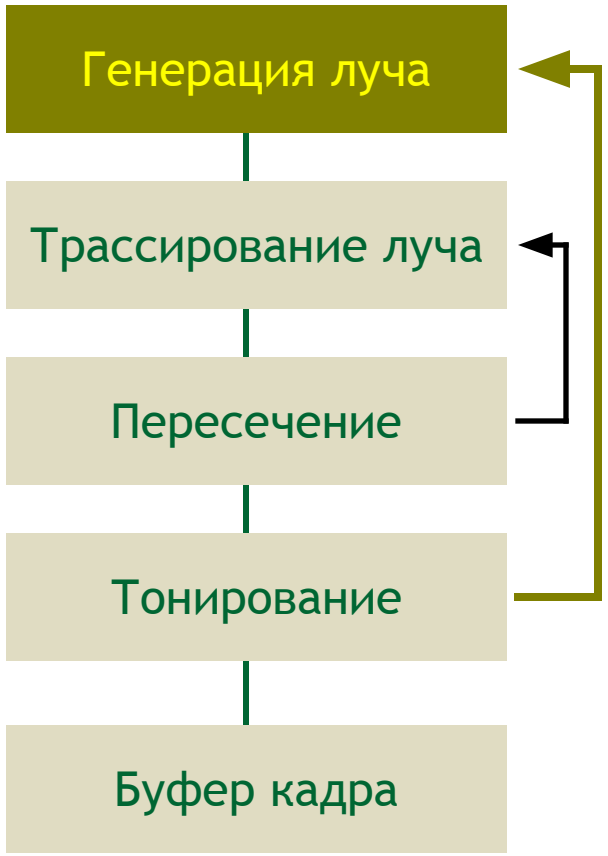
Трассировка лучей



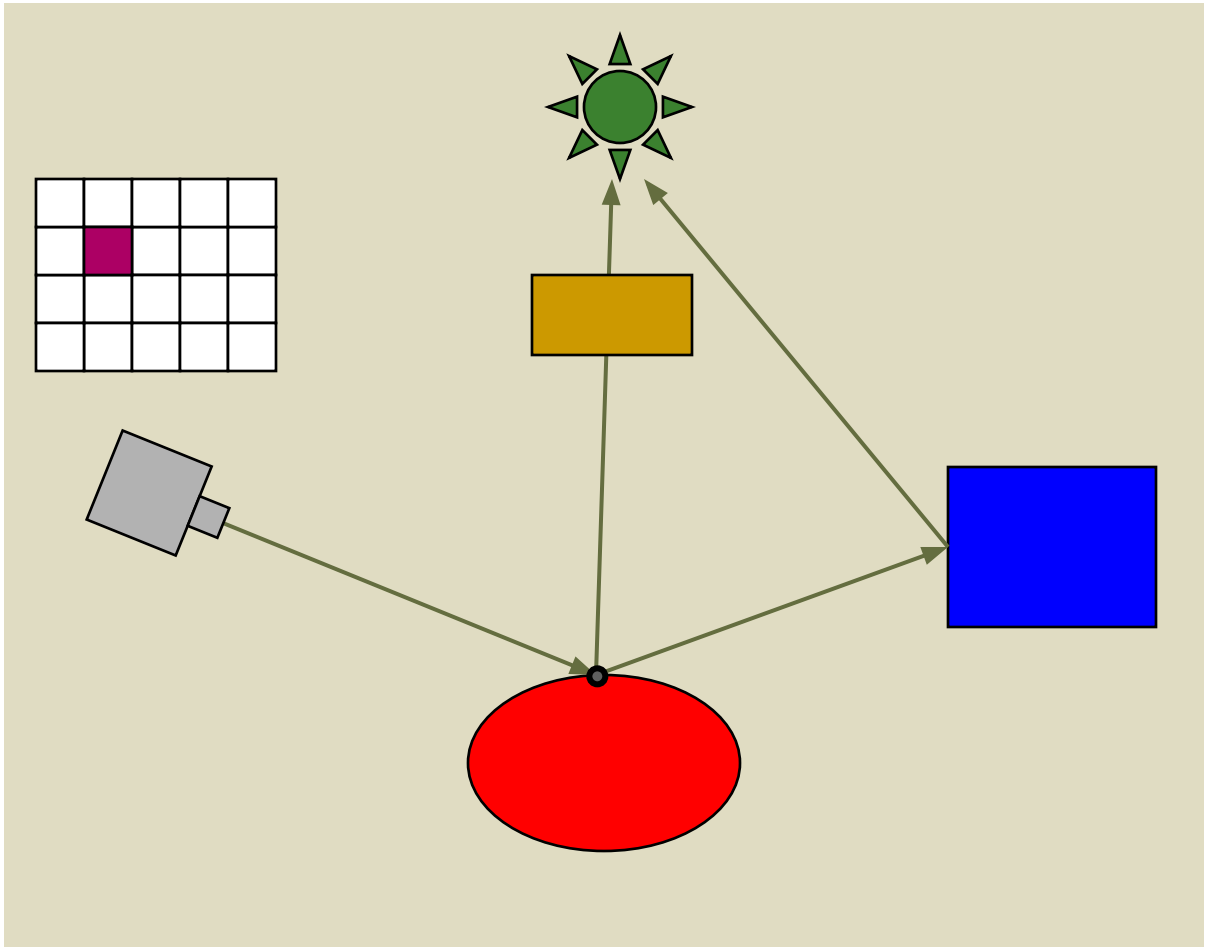


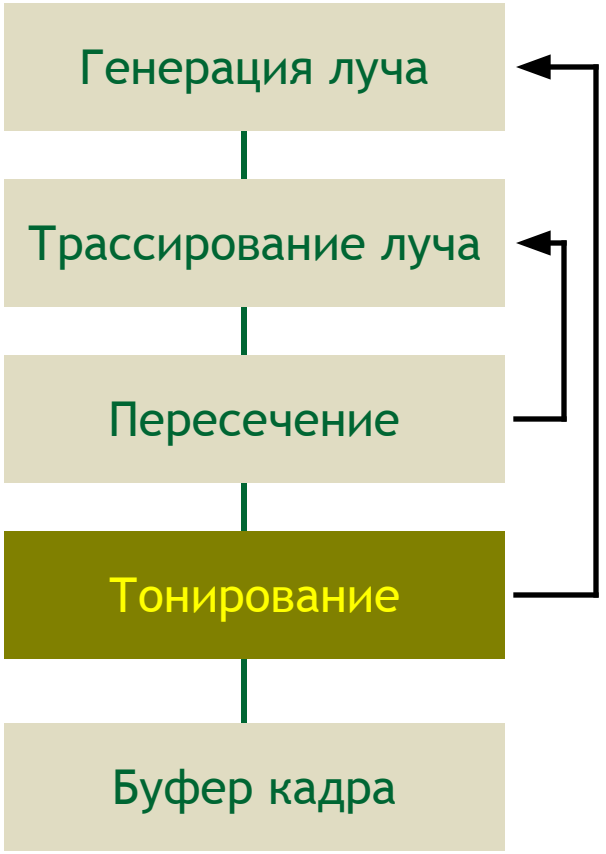
Трассировка лучей



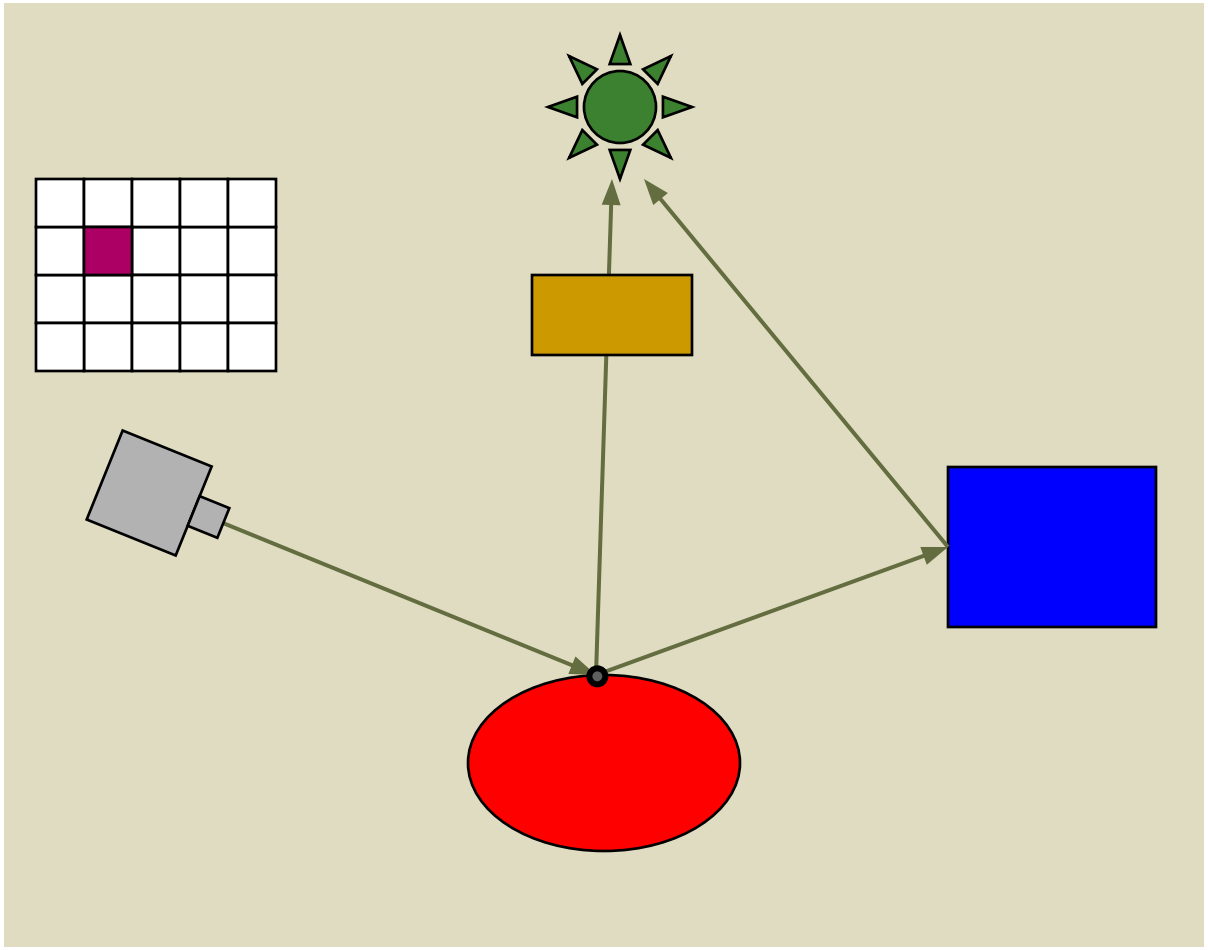


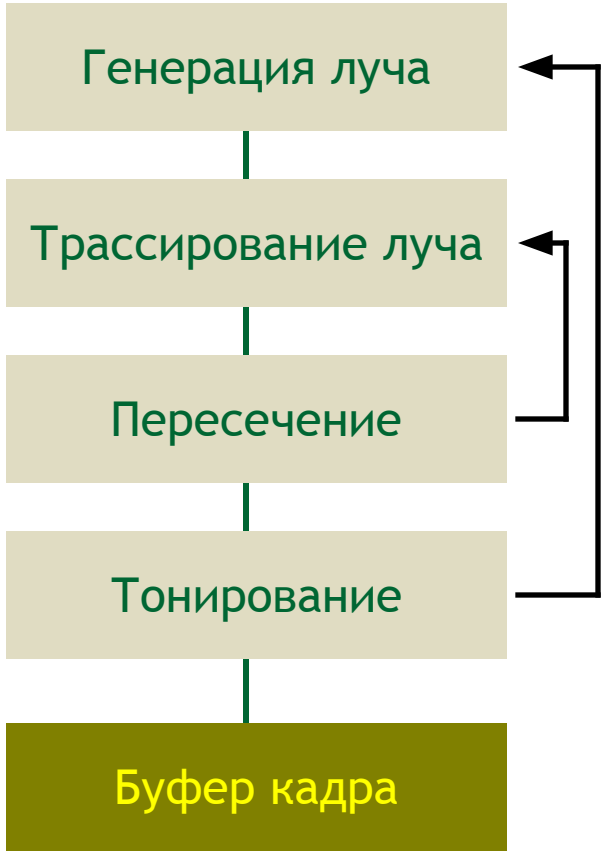
Трассировка лучей



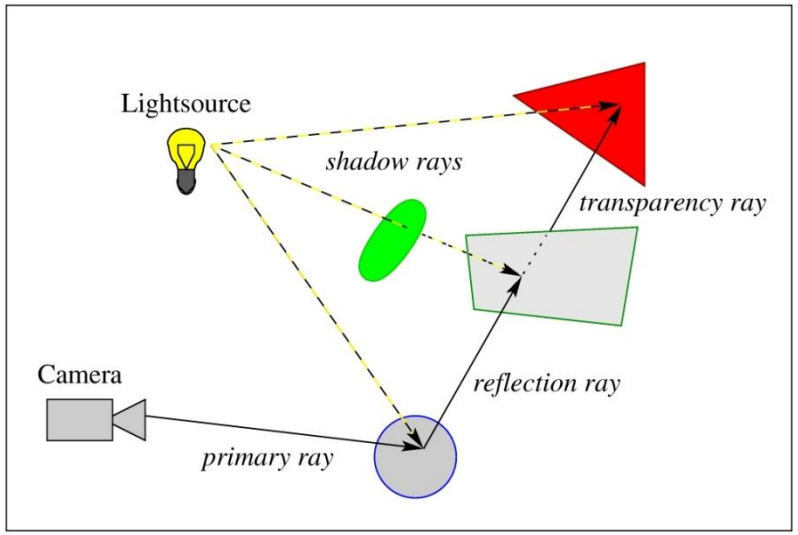


Трассировка лучей





Трассировка лучей: свойства



- Глобальное освещение
- Параллелизм
- Расширямость
- Вычисления только по запросу
- Попиксельные операции

Растеризация vs. Трассировка лучей

- Определение: растеризация
  - Дан набор лучей и притимивов, вычислить подмножество лучей, пересекающихся с примитивом
  - 2D сетка (экран) как индекс
- Определение: трассировка лучей
  - Дан луч и набор примитивов, вычислить подмножество примитивов, пересекающихся с лучом
  - 3D иерархическая структура как индекс

Растеризация vs. Трассировка лучей

- 3D индекс в мировом пространстве
  - Ограничивает динамику сцены (перестройка)
  - Масштабируемость O(log n)
  - Произвольные наборы лучей
- 2D сетка в пространстве экрана
  - Регулярная дискретизация

Растеризация vs. Трассировка лучей

- Слияние: 2D-сетка + 3D-структура в мировом пространстве
  - Растеризация становится ближе к трассировке
  - Те же самые ограничения (динамика сцены)
    - Но индекс может быть менее детализированным
  - Вычисления делятся на аппаратные и программные
    - Увеличение сложности, вопросы обмена данными...

Растеризация vs. Трассировка лучей

- Попиксельная эффективность
  - Функции тонирования имеют одинаковую сложность
  - Растеризация
    - Инкрементное вычисление между пикселями
      - Строчная развертка
    - Лишние операции из-за z-буфера (overdraw)
  - Трассировка
    - Нет инкрементных вычислений
    - Нет лишних операций

Растеризация vs. Трассировка лучей

- Достоинства вычислений «по запросу»
  - Только требуемые вычисления
    - □ эффективность
    - Пример: не нужно вычислять всю карту освещения
  - Не требуется передискретизация данных
    - □ точность
  - Подгрузка данных только по требованию □ ресурсы

Растеризация vs. Трассировка лучей

- Аппаратная поддержка
  - Растеризация имеет полную аппаратную поддержку
    - Быстрое развитие
    - Высокая производительность, параллелизм, поточная обработка
  - Трассировка в основном реализуется программно
    - Требуются гибкие потоки управления, рекурсия, гибкий ввод/вывод
    - Требуется виртуальная память, инкреметная подгрузка
    - Требуется полная поддержка циклов
    - Сильно зависит от кэширования
    - □ Нет аппаратной поддержки

Часть 2/3

ОСНОВЫ ТРАССИРОВКИ И АЛГОРИТМЫ ПОИСКА ПЕРЕСЕЧЕНИЙ

Трассировка поверхностей

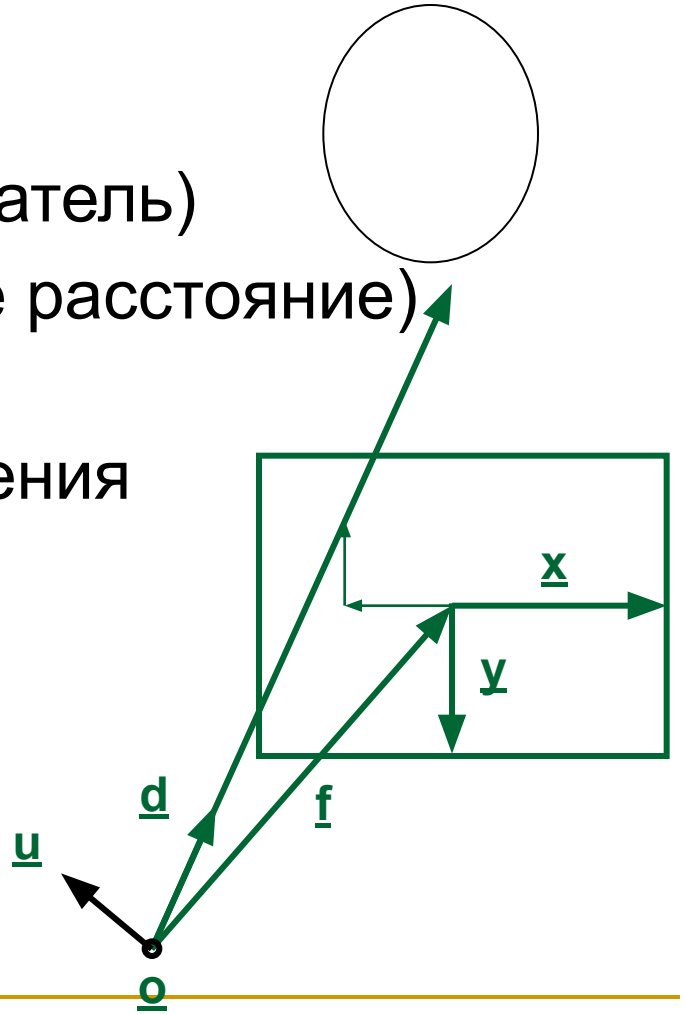
- Предположение: пустое пространство полностью прозрачно
- Поверхности
  - Трехмерные геометрические модели объектов
- Материалы поверхностей
  - Отражение, поглощение, пропускание и т.п.
- Освещение
  - Положение и характеристики источников света

Основные шаги

- Генерация первичных лучей
  - Rays from viewpoint into 3D scene
- Трассировка лучей
  - Первое пересечение с геометрией сцены
- Тонирование
  - Излучение (radiance) переносится с лучом
  - В точке пересечения входящее излучение вычисляется с помощью дополнительных лучей

Генерация лучей

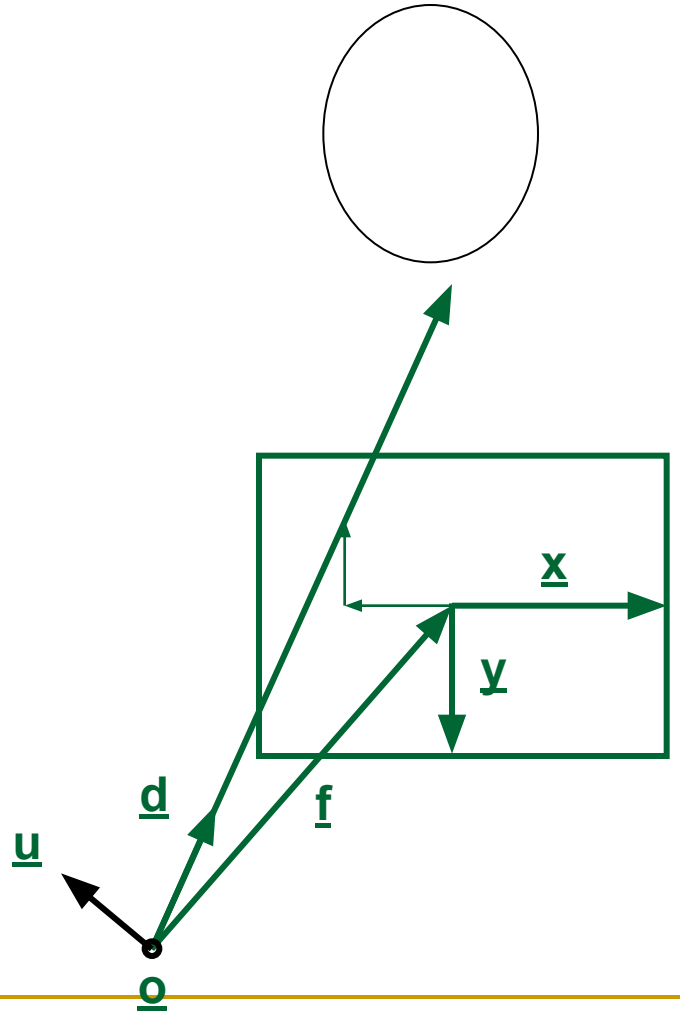
- Камера-обскура
  - o: Центр проекции (наблюдатель)
  - f: Вектор зрения (фокусное расстояние)
    x, y: Оконные координаты
  - xres, yres: Размер изображения



Ray Generation

- Камера-обскура

```
for (x= 0; x < xres; x++)
  for (y= 0; y < yres; y++)
  {
    d= f + 2(x/xres - 0.5)·x
         + 2(y/yres - 0.5)
·y;
    d= d/|d|; // Normalize
    col= trace(o, d);
    write_pixel(x,y,col);
  }
```



Представления луча и объектов

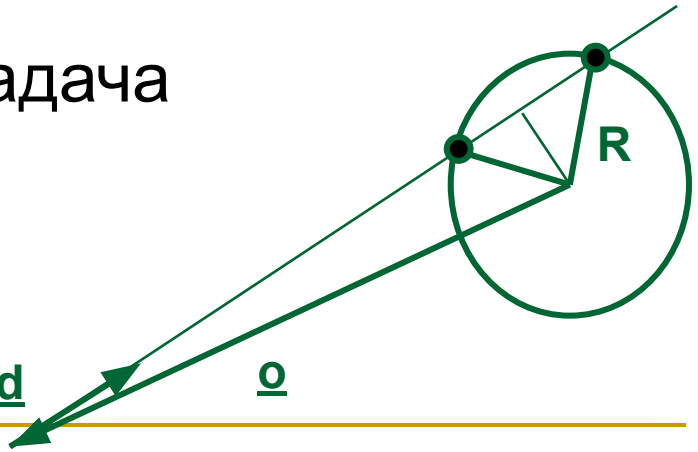
- Луч: $\underline{r}(t)=\underline{o}+t\,\underline{d}$
  - $\underline{\mathbf{o}}=(o_x, o_y, o_z)$, $\underline{\mathbf{d}}=(d_x, d_y, d_z)$
- Геометрия сцены
  - Plane: $(\underline{p}-\underline{a})\cdot\underline{n}=0$
    - Implicit definition ($\underline{\mathbf{n}}$: surface normal, $\underline{\mathbf{a}}$: point one surface)
  - Sphere: $(\underline{p}-\underline{c})\cdot(\underline{p}-\underline{c})-r^2=0$
    - $\underline{\mathbf{c}}$: sphere center, $\boldsymbol{r}$: sphere radius
  - Triangles: Plane plus 2D coordinates

Пересечение луча со сферой

- Сфера
  - Сфера в начале координат ($x^2 + y^2 + z^2 - 1 = 0$)
  - Подставляем уравнение для луча

    $$t^2(d_x^2 + d_y^2 + d_z^2) + 2t\,(d_x o_x + d_y o_y + d_z o_z) + (o_x^2 + o_y^2 + o_z^2) - 1 = 0$$

  - Вариант: геометрическая задача



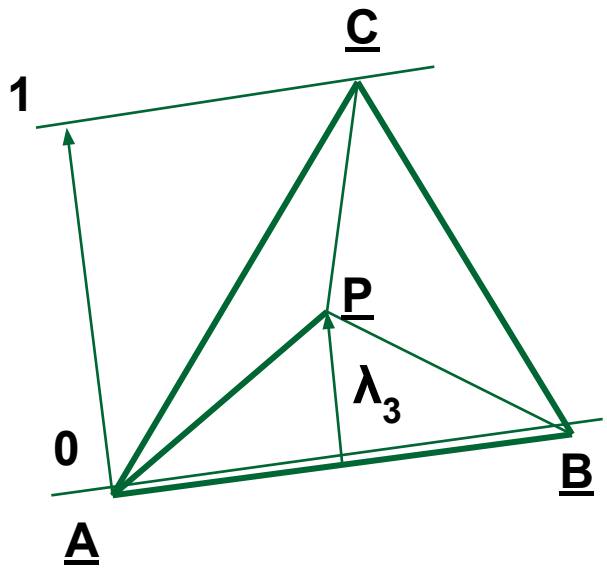
Пересечение луча с плоскостью

- Плоскость
  - Уравнение плоскости: $\underline{p} \cdot \underline{n} - D = 0$, $|\underline{n}| = 1$
    - *Неявное представление*
    - Нормаль: $\underline{n}$
    - Перпендикуляр до (0, 0, 0): D
    - Заменяем $\underline{o} + t\underline{d}$ на p
    - $(\underline{o} + t\underline{d}) \cdot \underline{n} - D = 0$
    - Решаем для t:

$$t = \frac{D - o \cdot n}{d \cdot n}$$

Пересечение луча с треугольником

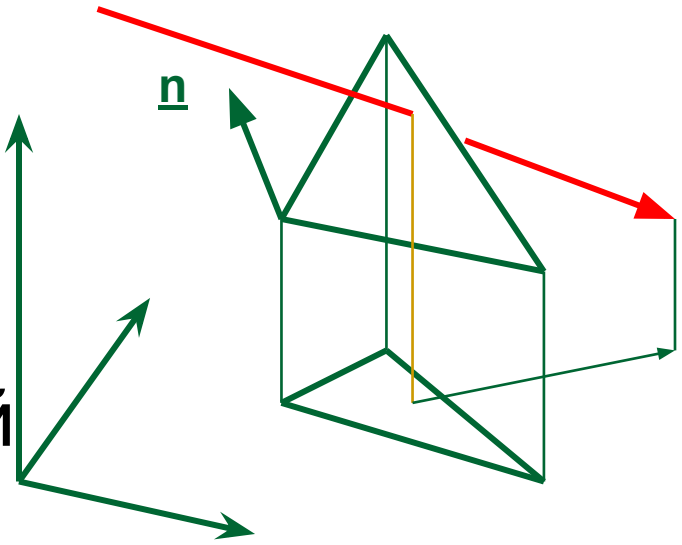
- Барицентрические координаты
  - Невырожденный трк. ABC
    $\underline{P} = \lambda_1 \underline{A} + \lambda_2 \underline{B} + \lambda_3 \underline{C}$
  - $\lambda_1 + \lambda_2 + \lambda_3 = 1$
  - $\lambda_3 = \angle(APB) / \angle(ACB)$ etc
    - Relative area



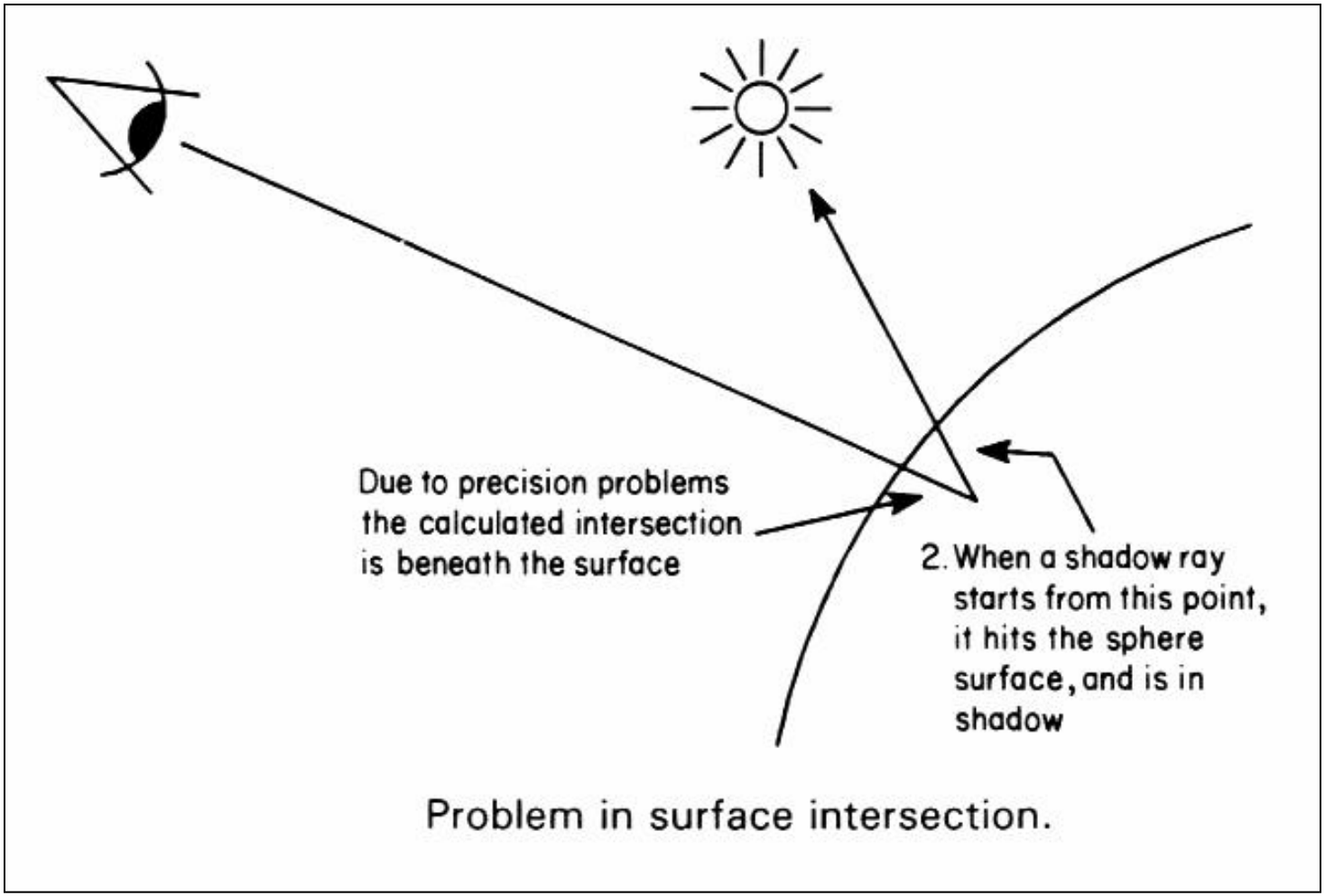
- Пересечение, если все $\lambda_i >= 0$

Пересечение луча с треугольником

- Пересечение с плоскостью треугольника
- Дана 3D-точка пересечения
  - Спроецировать точку на плоскость xy, xz, yz
  - Можно использовать любую плоскостью
  - Плоскость и 2D-положения вершин можно вычислить заранее
- Провести барицентрический тест



Проблема точности



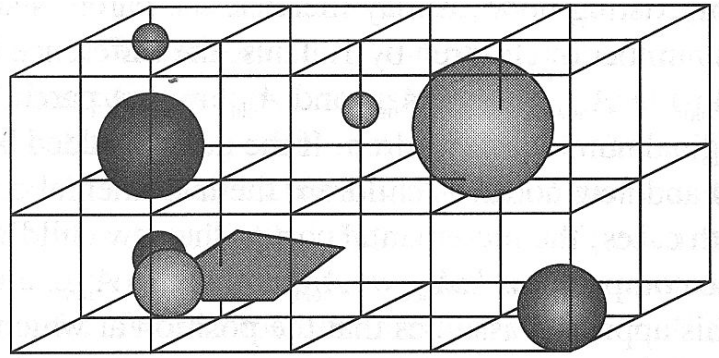
Problem in surface intersection.

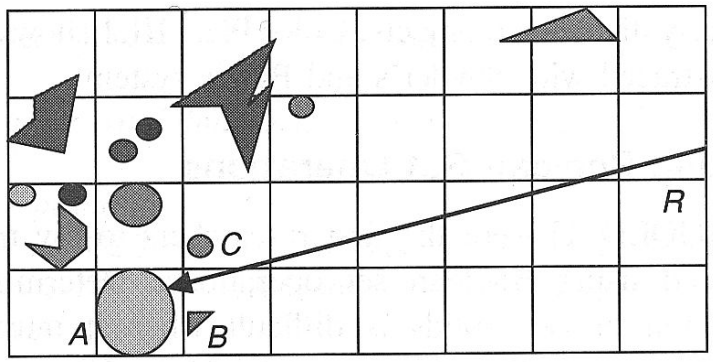
Ускорение трассировки

- Пересечение луча со всеми объектами и сортировка для поиска ближайшего пересечения
  - Очень дорого!
- Ускорение алгоритма пересечения
  - Небольшой эффект
- Уменьшение количества пересечений
  - Разбиение пространства (часто иерархическое)
    - Сетки, октодеревья, BSD и kd-деревья, деревья ограничивающих объемов
    - 5D разбиение (позиция и направление)

Сетки

- Построение сетки
  - Начинаем с описывающего параллелепипеда
  - Треугольники разбиваются по вокселям
- Трассировка
  - Алгоритм Брезенхема в 3D
  - Останавливаемся, если пересечение найдено в текущем вокселе





Сетка: проблемы

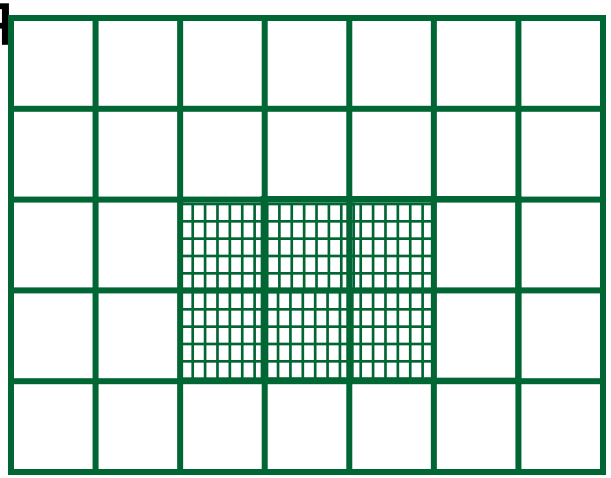
- Обход сетки
  - Перечисление вокселей вдоль луча
  - Очень простой алгоритм, возможна аппаратная реализация
- Разрешение сетки
  - Очень сильно зависит от сцены
  - Невозможна адаптация к локальной плотности примитивов
    - Проблема «Чайника на стадионе»
  - Возможное решение: иерархические сетки

Сетка: проблемы

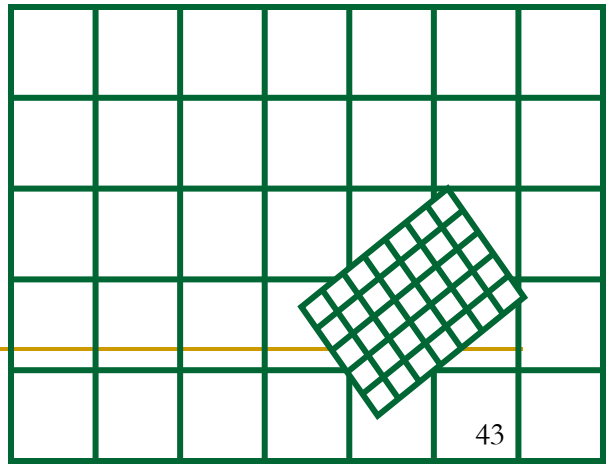
- Объекты в нескольких вокселях
  - Хранить только ссылки на объекты
  - Хранить информацию о найденных пересечениях в кэше
    - Предел: хранить индекс луча в каждом треугольнике

Иерархические сетки

- Простой алгоритм построения
  - Рекурсивное создание сеток в вокселях с высокой плотностью
  - Проблема: какое должно быть разрешение на каждом уровне?

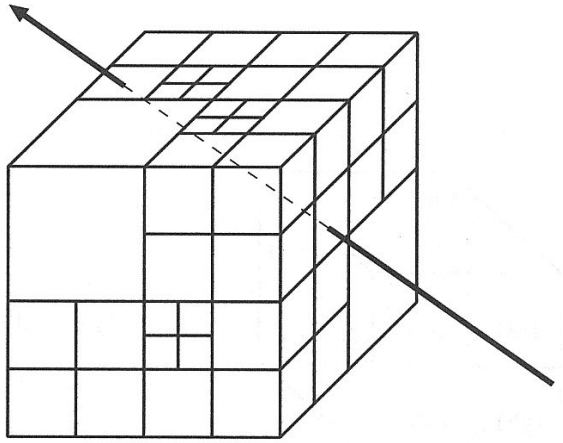


- Улучшения алгоритма
  - Разделить сетки для объектов
  - Проблема: что считать объектами?



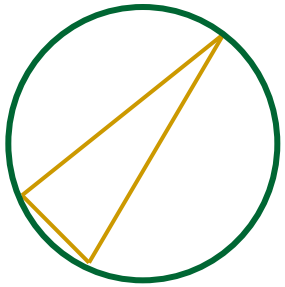
Октодерево

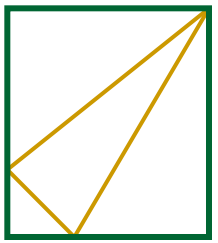
- Иерархическое разбиение пространства
  - Адаптивное рекурсивное разбиние пространства на 8 равных частей
- Проблемы
  - Достаточно сложный алгоритм обхода
  - Сложные регионы сходятся медленно

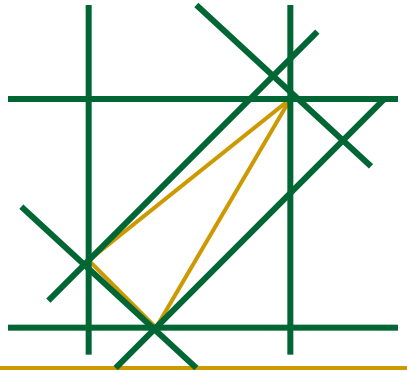


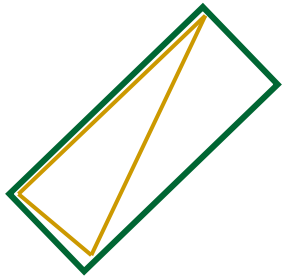
Описывающие объемы

- Идея
  - Вычислять пересечение с объектом только если луч пересекает простой описывающий объем
- Возможные описывающие объемы:
  - Сфера
  - Выровненный по осям описывающий параллелепипед
  - Описывающий параллелепипед



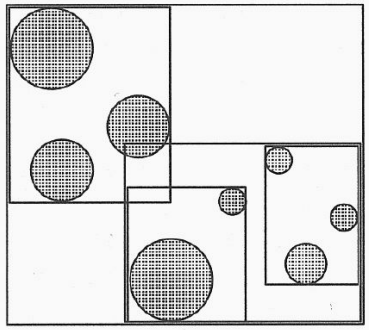


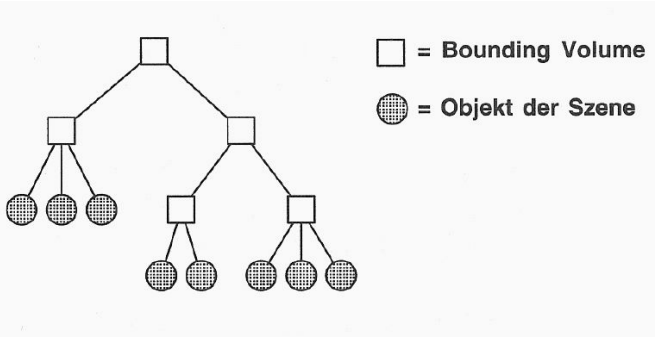




Иерархия описывающих сфер

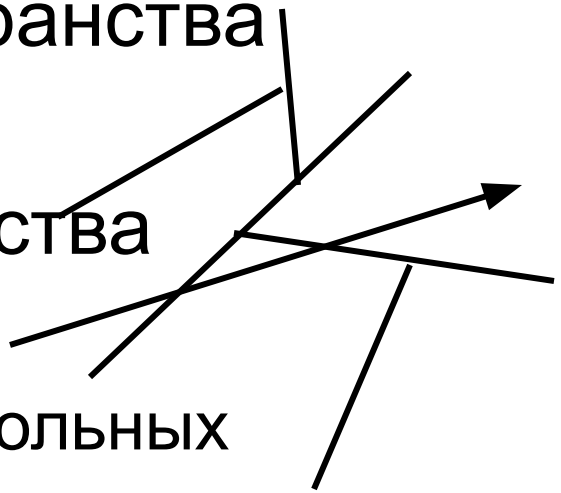
- Идея:
  - Разбиваем рекурсивно
- Преимущества:
  - Очень хорошая адаптивность
  - Эффективный обход O(log N)
- Проблемы
  - Как располагать описывающие объемы?

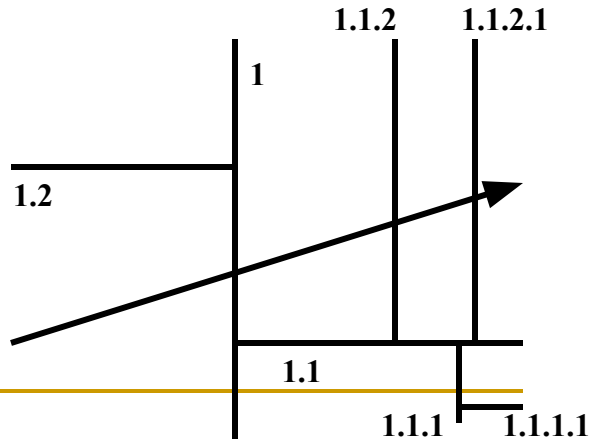




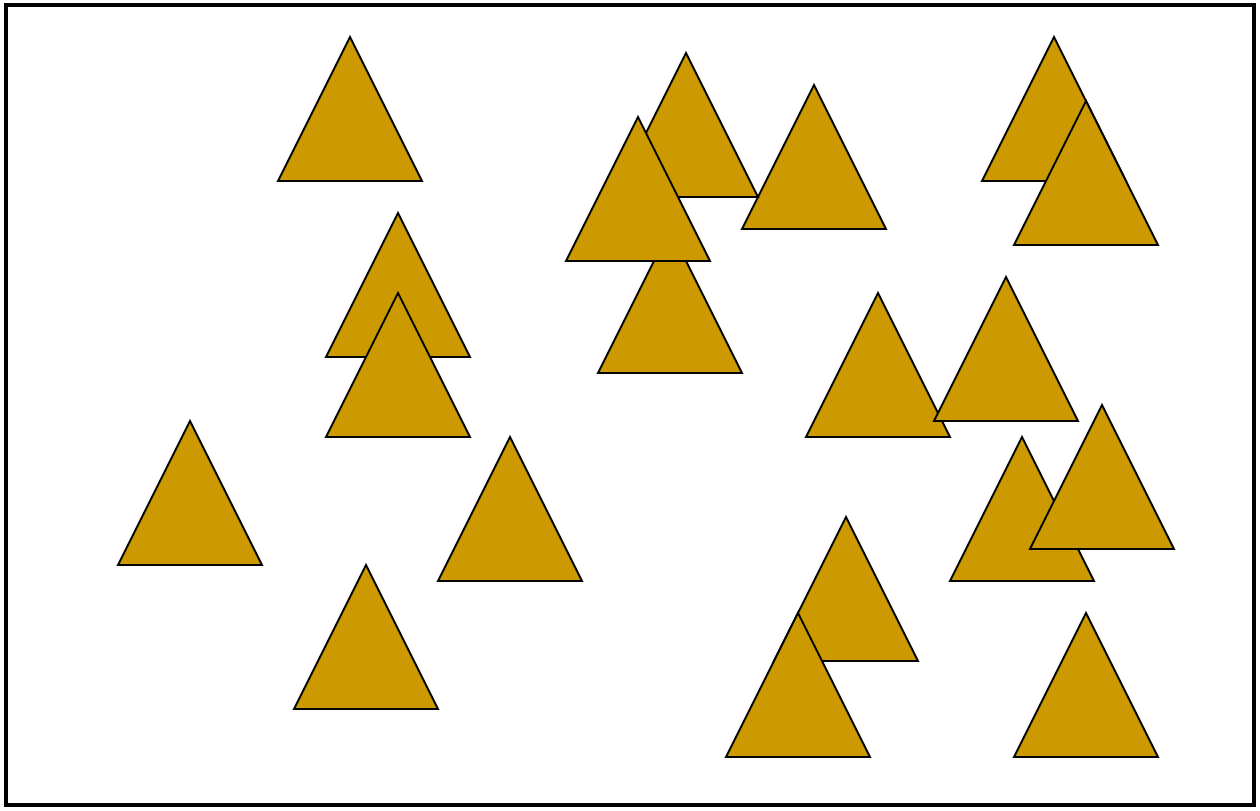
BSP- и Kd-деревья

- Рекурсивное разбиение пространства на полупространства
- Двоичное разбиение пространства (BSP):
  - Разбиение плоскостями в произвольных положениях
- Kd-деревья
  - Разбиение выровненными относительно осей плоскостями

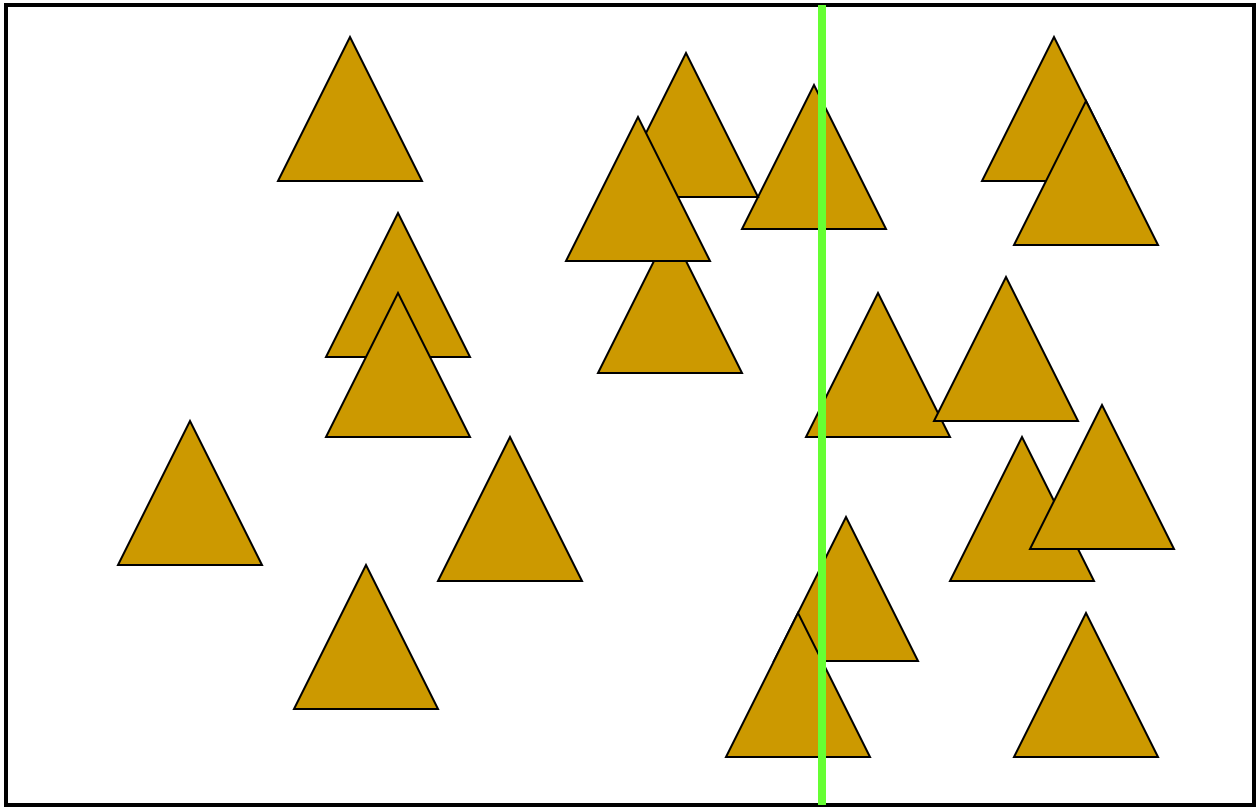




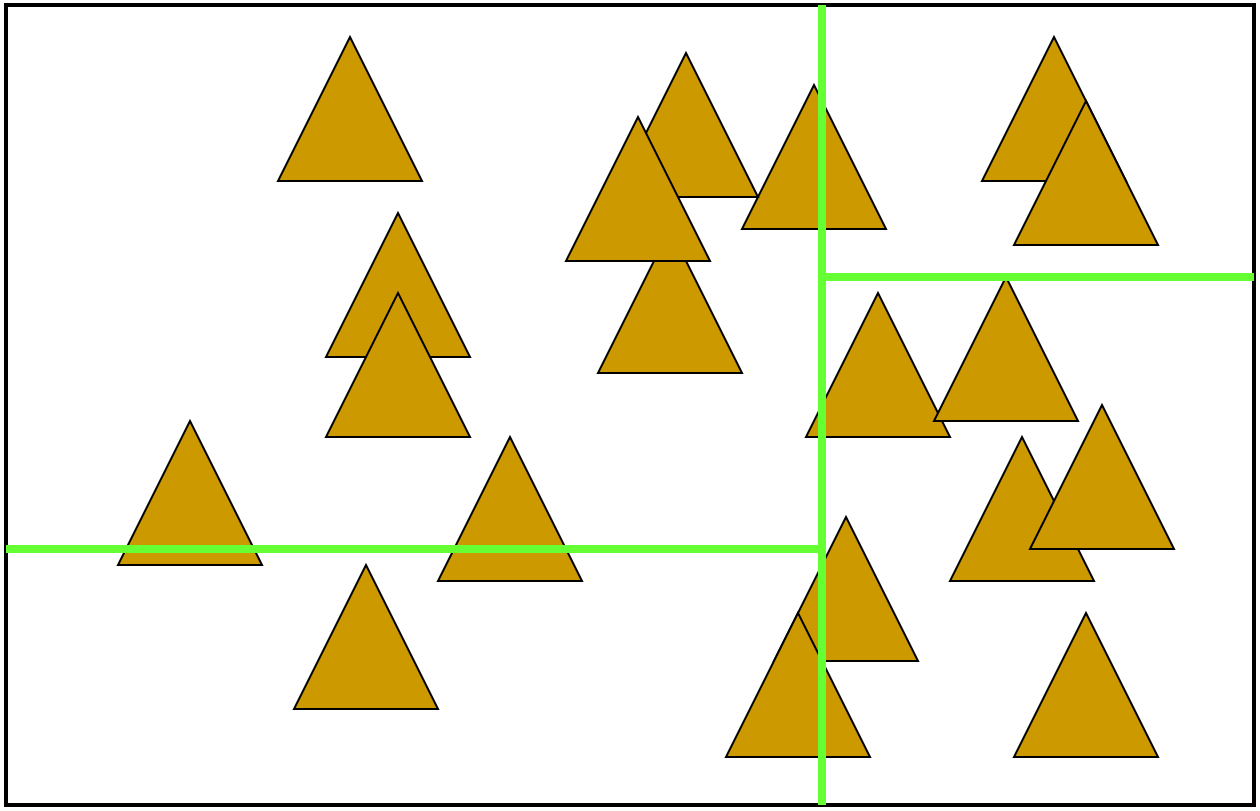
Построение kD-дерева



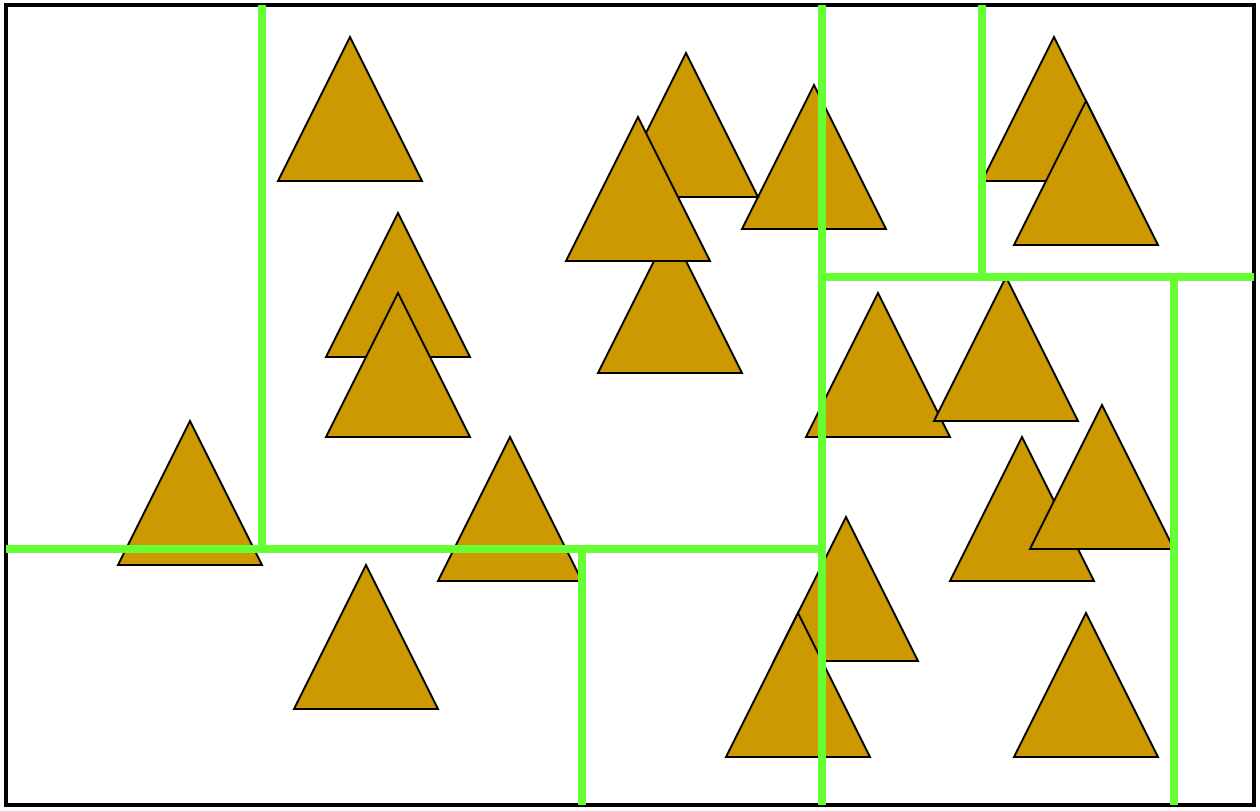
Построение kD-дерева



Построение kD-дерева



Построение kD-дерева



ТРАССИРОВКА: ИНТЕРАКТИВНАЯ ТРАССИРОВКА

Интерактивная трассировка лучей

- В: Что такое интерактивная трассировка лучей?
- О: Это обычная трассировка + оптимизации, оптимизации, оптимизации...

- Оптимизации могут быть и алгоритмическими!

Что можно оптимизировать?

- 1. Построение пространственного индекса
- 2. Алгоритм трассирования луча
- 3. Вычисление освещения

Оптимизации: построение пространственного индекса

- kD-деревья
  - Адаптивны
  - Компактны
  - Быстрый обход
- За счет хорошо построенного дерева можно получить увеличение скорости **в несколько раз**!
- Проблема: где провести разбивающую плоскость?
  - Учет вероятностей попадания луча в разные полуплоскости
- Проблема: где остановить разбиение?

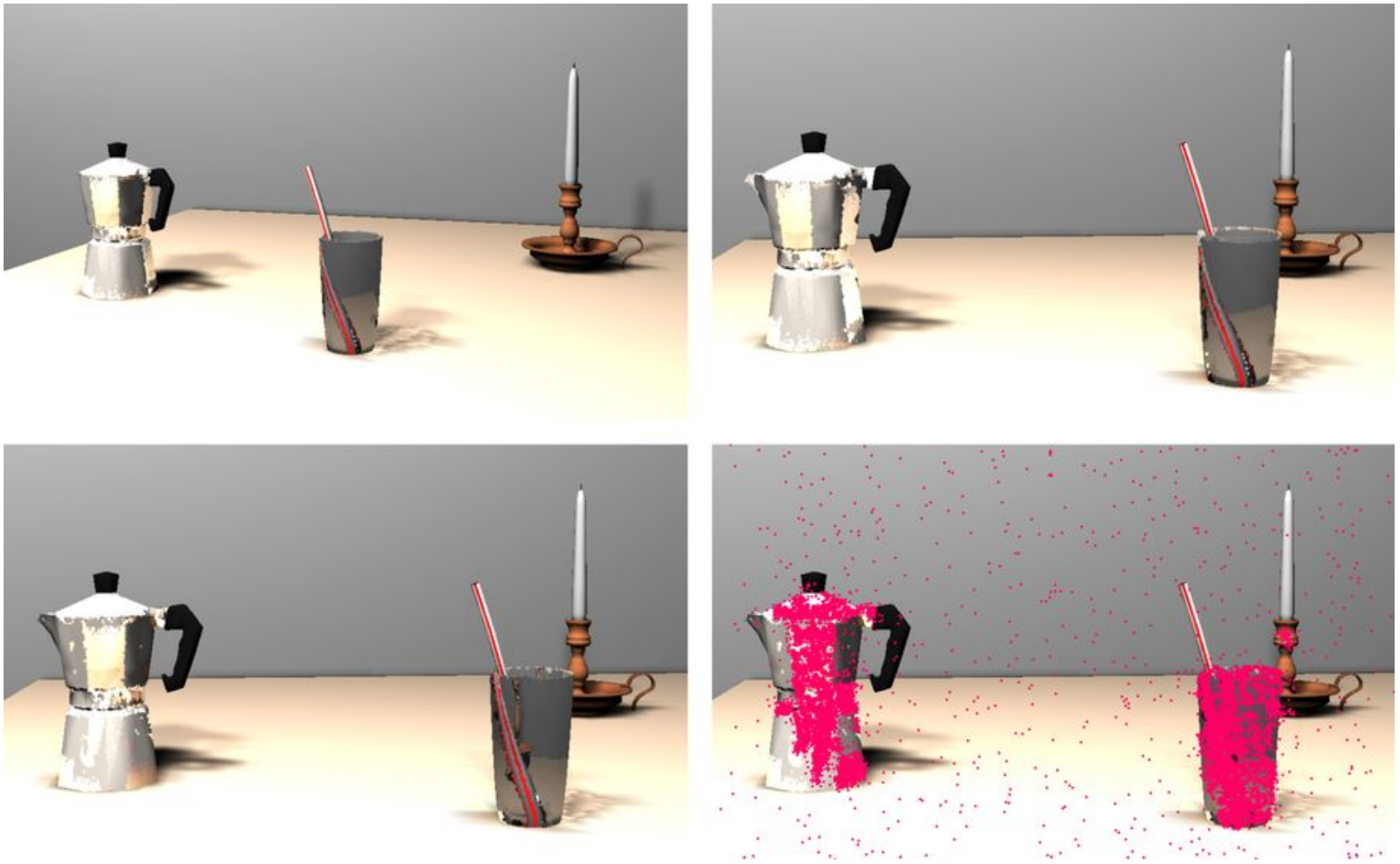
Оптимизации: Алгоритм трассирования луча

- Оптимизация структуры данных для узла дерева
  - Учет процессорного кэша
- Оптимизация цикла трассировки
  - Никаких рекурсий
  - Минимизация операцией со стеком
  - Параллелизация: SIMD, многоядерность
  - Когерентность

Оптимизации: Алгоритм трассирования луча

- Трассировка луча все равно очень дорога
- Два варианта:
  - Трассировать больше лучей в секунду
    - рассмотрели
  - Трассировать меньше лучей на кадр
    - Использование растеризационной аппаратуры
    - Технологии на основе изображений
    - Интерполяция результатов трассирования

Алгоритм трассирования луча: корректирующие текстуры



Оптимизации: Вычисление освещения

- Проблема: вычисление интеграла
- Подходы:
  - Квази-статические (квази Монте-Карло)
  - Гибридные
    - Использование растеризационной аппаратуры
    - Технологии на основе изображений

Почему сейчас?

- Успех растеризации и отсутствие прогресса остановили развитие алгоритмов трассировки лучей в 90е
  - Мало низкоуровневой оптимизации, не использовалась когерентность и т.п.
- В начале 2000х развитие алгоритмов позволило догнать аппаратные методы
  - За счет софтверной оптимизации
  - В основном на сложных сценах
- Сейчас программируемые аппаратные ускорители позволили еще более ускорить трассировку
  - Все равно неудобно – ориентация на растеризацию
  - Надежды на следующее поколение

Причины использования интерактивной трассировки

- Реалистичные изображения по умолчанию
- Физическая корректность
- Поддержка массивных сцен
- Интеграция различных типов примитивов
- Декларативное описание сцен
- Интерактивное глобальное освещение

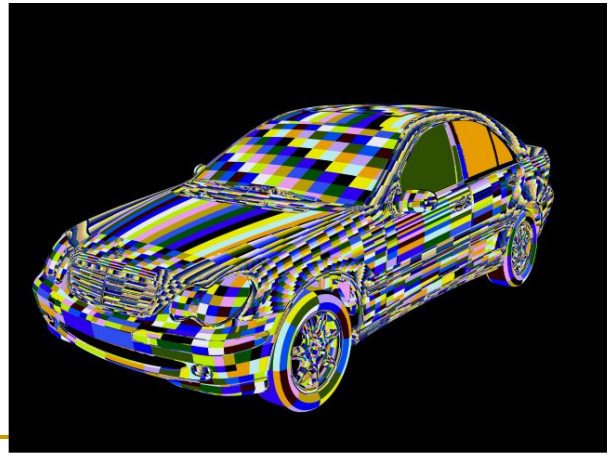
Реалистичные изображения по умолчанию

Volkswagen Beetle with correct shadows and (multi-)reflections on curved surfaces

Физическая корректность

Fully ray traced car head lamp, faithful visualization requires up to 50 rays per pixel

Физическая корректность



Rendered directly from trimmed NURBS surfaces, with smooth environment lighting

Физическая корректность

Textured Phong for comparison

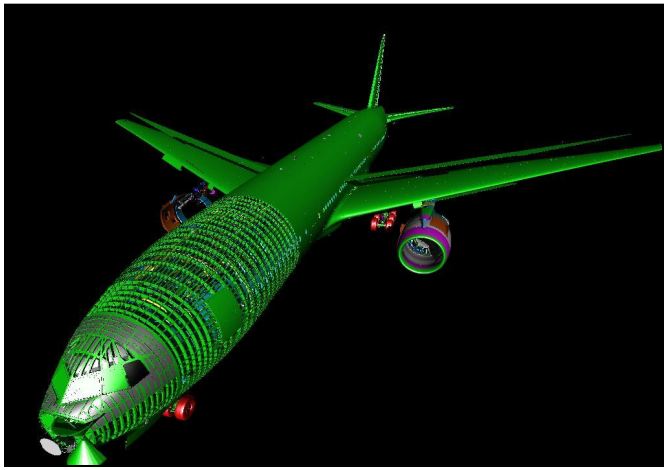
Rendered with accurately measured BTF data that accounts for micro lighting effects

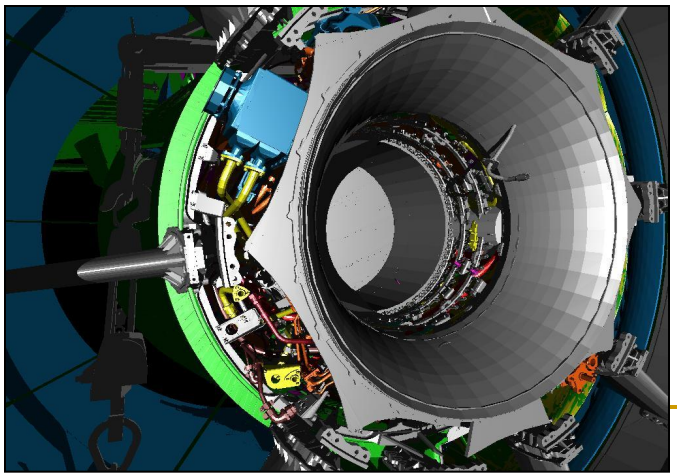
BTF Data Courtesy R. Klein, Uni Bonn

Физическая корректность

VR scene illuminated from realtime video feed, AR with realtime environment lighting

Поддержка массивных сцен

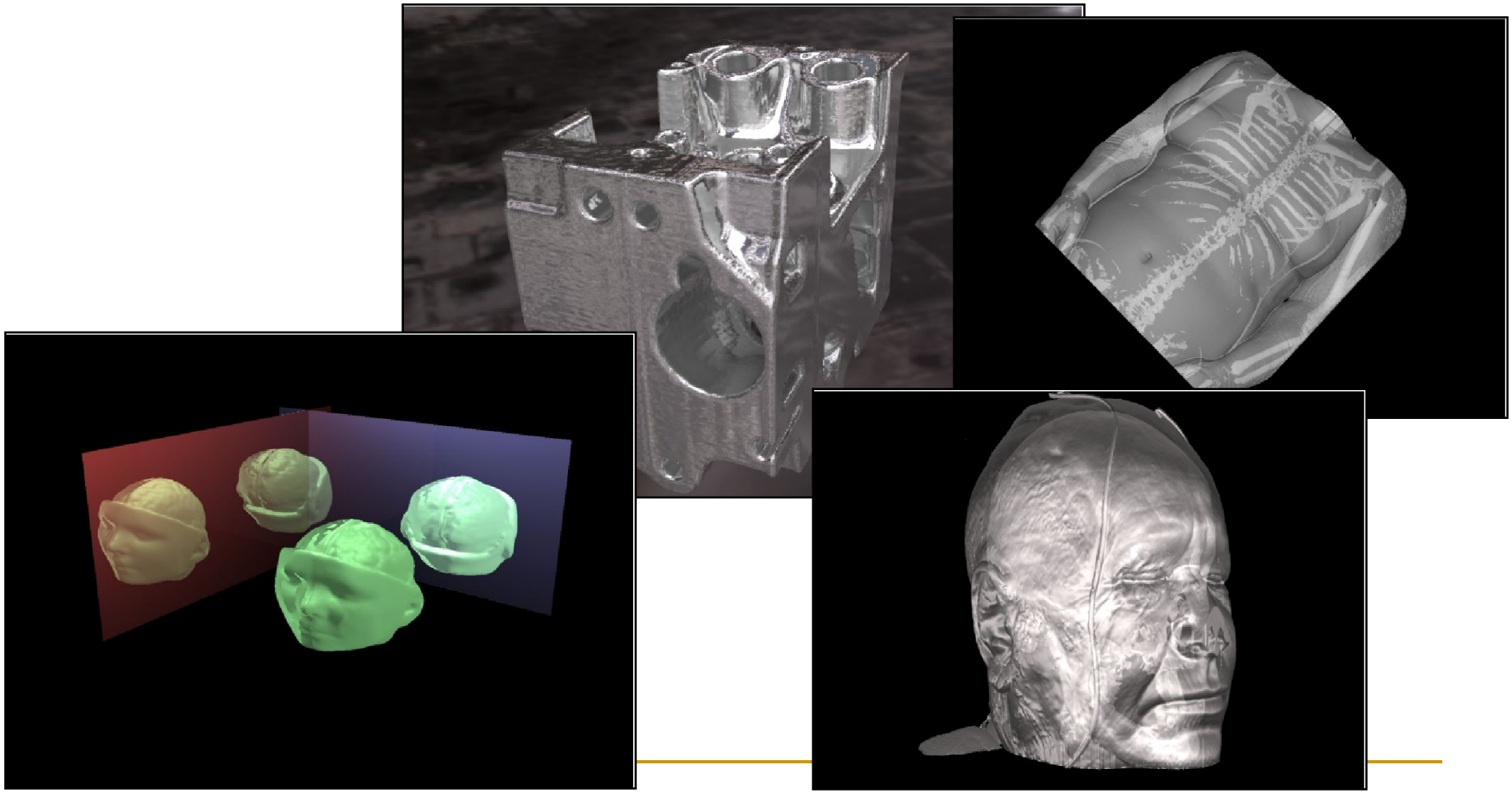




Интеграция различных типов примитивов

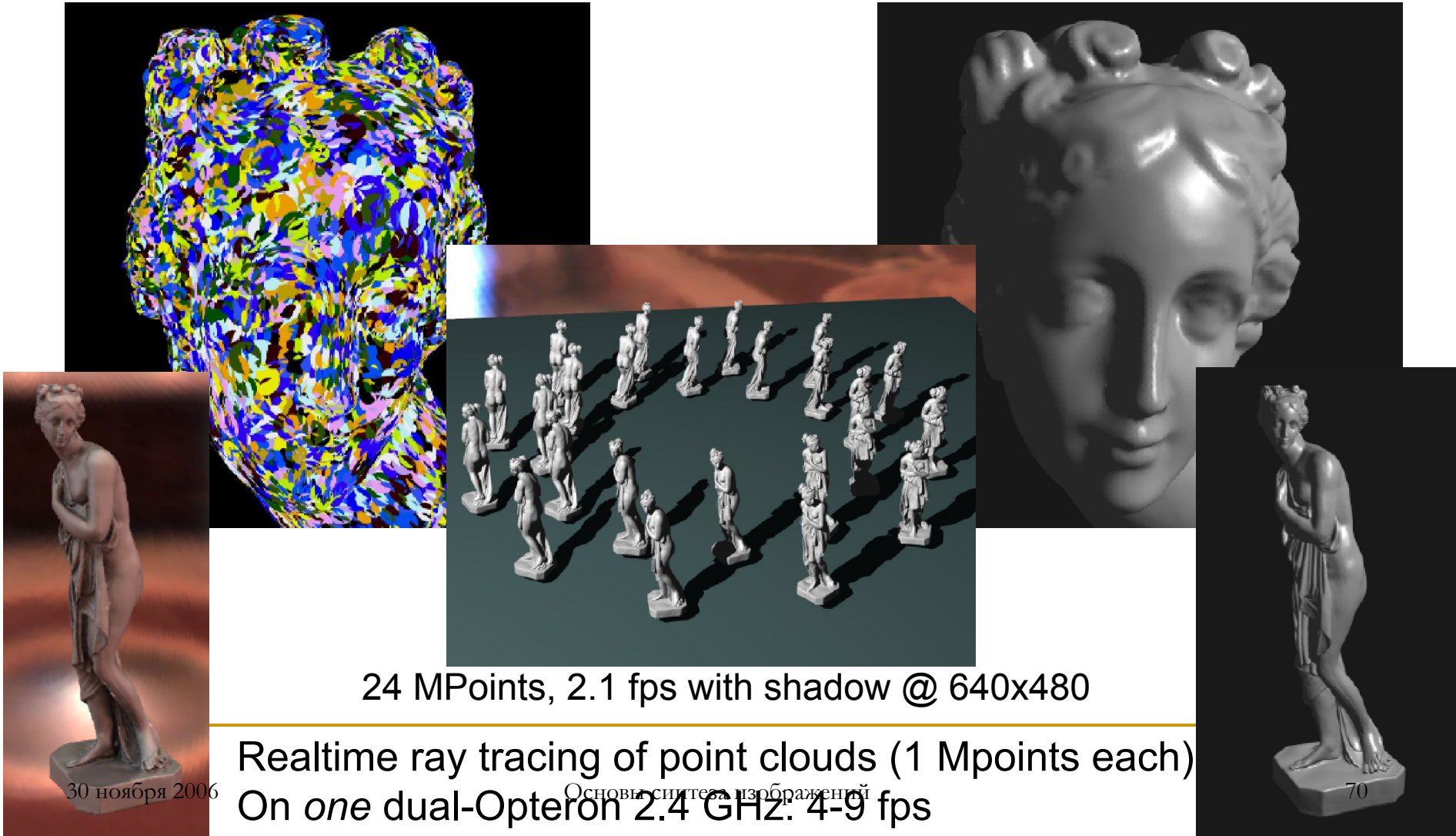
Triangles, Bezier splines, and subdivision surfaces fully integrated

Интеграция различных типов примитивов



Volume visualization using multiple iso-surfaces in combination with surface rendering

Интеграция различных типов примитивов



24 MPoints, 2.1 fps with shadow @ 640x480

Realtime ray tracing of point clouds (1 Mpoints each)
On *one* dual-Opteron 2.4 GHz: 4-9 fps

Декларативное описание сцен

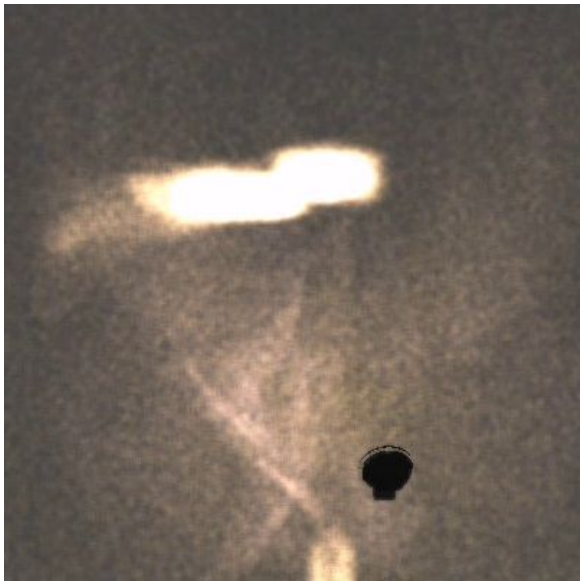
- Декларативный интерфейс задания сцены
  - Приложение задает всю сцену за раз
  - Экранизация полностью выполняется на уровне трассировщика (например, в железе)
- Достоинства
  - Сильно упрощает программирование
  - Возможно полное аппаратное ускорение

Глобальное освещение

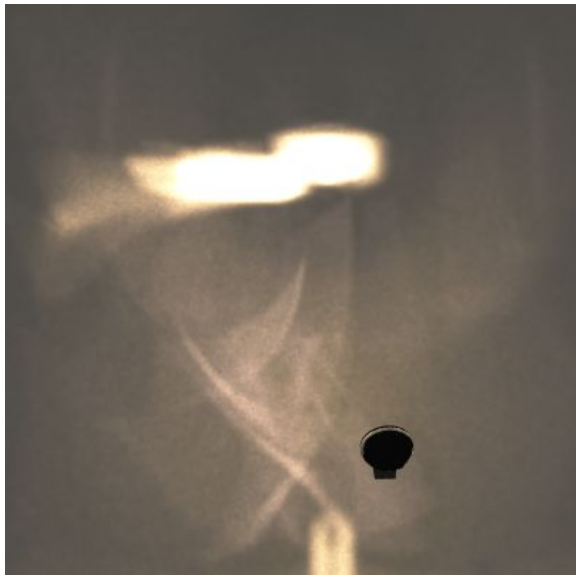
Conference room (380 000 tris, 104 lights) with full global illumination in realtime

Глобальное освещение

250k / 3 fps



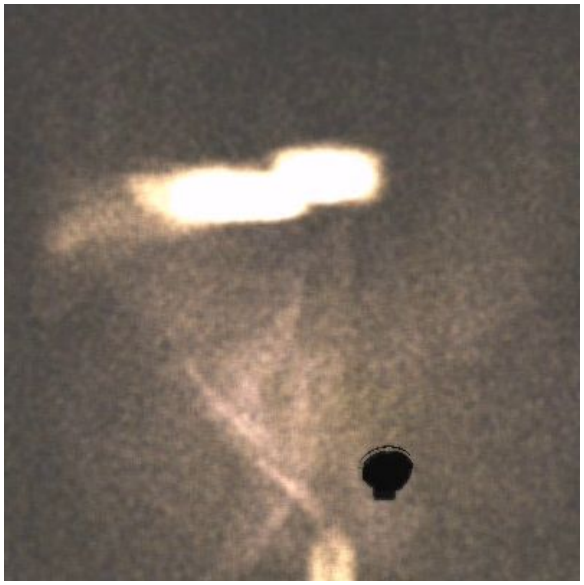
25M / 11 fps



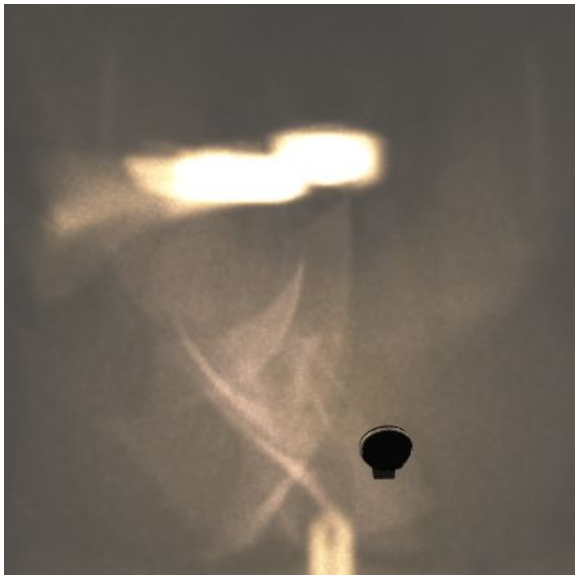
Light pattern from a car head lamp computed in realtime using photon mapping:
Left: realtime update, middle: accumulated in 30s, right: photograph of real pattern

Глобальное освещение

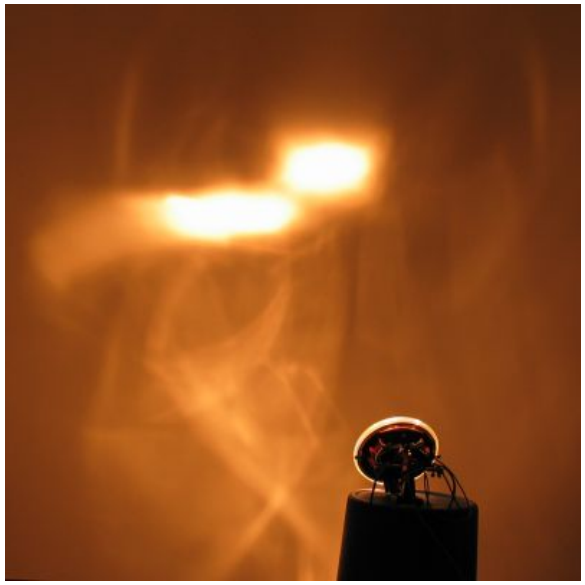
250k / 3 fps



25M / 11 fps



Photograph



Light pattern from a car head lamp computed in realtime using photon mapping:
Left: realtime update, middle: accumulated in 30s, right: photograph of real pattern

Проблемы интерактивной трассировки лучей

- Динамические сцены
  - Изменения геометрии обновление пространственного индекса
- Подходы
  - Деление сцен исходя из темпоральных характеристик
  - «Ленивый» индекс

Проблемы интерактивной трассировки лучей

- Эффективное устранение ступенчатости и блестящие (glossy) отражения
  - Нужно много лучей для корректного результата

Проблемы интерактивной трассировки лучей

- Аппаратная поддержка
  - Сейчас вся mainstream-поддержка разрабатывается под растеризацию
- Возможные решения
  - Многоядерные CPU – перспективно!
  - Cell: Нет кэша
  - GPU: ограничения на поток управления
  - Специальная аппаратура

Итоги

- Трассировка лучей = быстрый поиск пересечения
- Трассировка vs. Растеризация
- Интерактивная трассировка лучей = оптимизация

Материалы

- В презентации использованы слайды из курса “Interactive Ray Tracing”, представленнего на конференции SIGGRAPH’2005