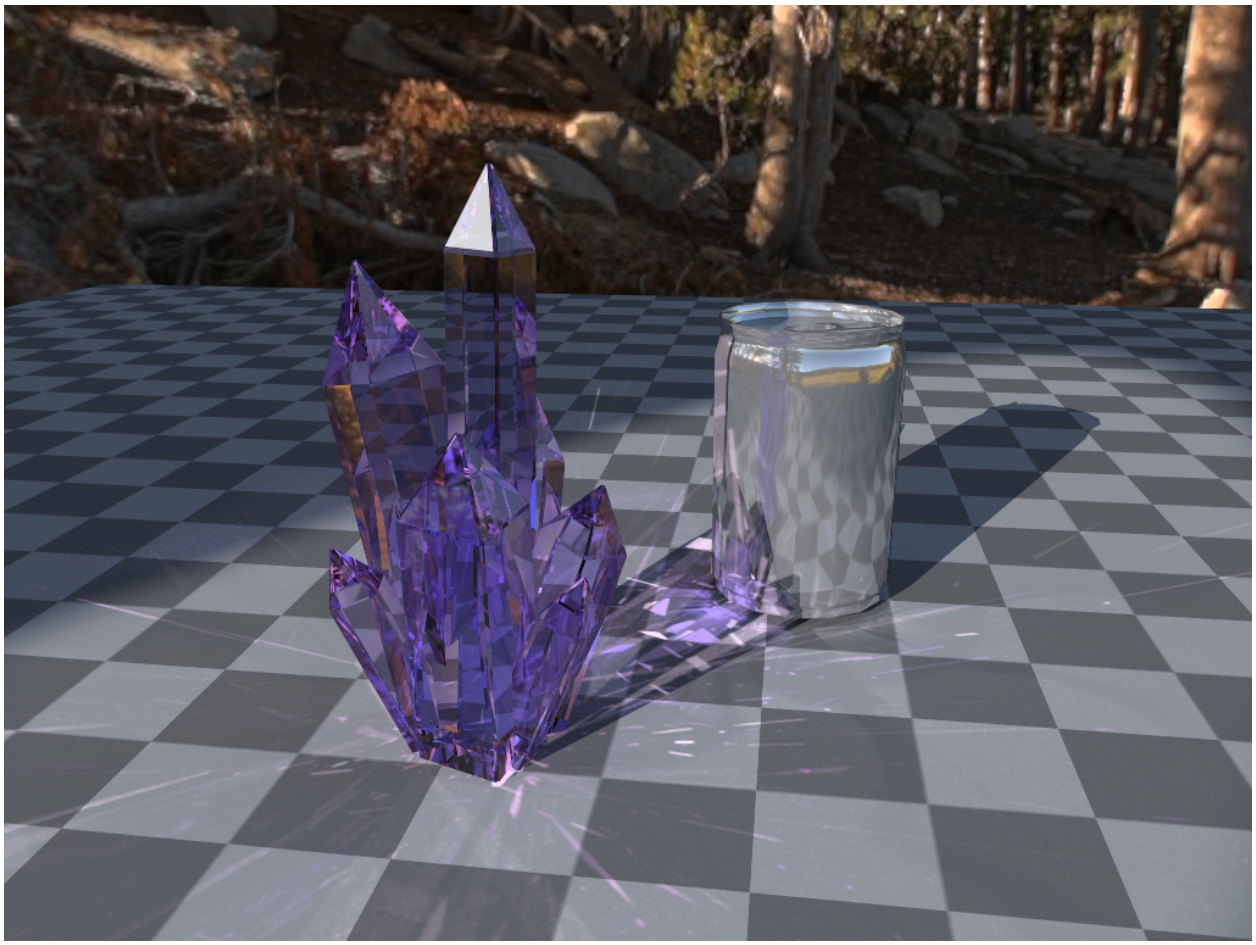
Визуализация кристалла

Авторы задания: Владимир Афанасьев, Владимир Фролов, Александра Афанасьева

Цель задания - реализовать визуализацию полигональных объектов с помощью трассировки лучей.



Базовая часть (15 баллов)

В базовой части требуется реализовать алгоритм трассировки лучей с учетом корректных преломлений и отражений.

Требования

Расчёт изображения

- Камера с перспективной точечной проекцией

- Цветное RGB изображение
- Использование законов отражения и преломления света
- Использование формул Френеля
- Использование локальной модели освещения (Фонга)

Сцена

- Кристалл (преломляющий и отражающий объект), заданный в виде полигональной модели
- Плоское зеркало
- Сферическое окружение, либо окружение в виде комнаты “cornell box”, либо более сложные варианты помещений.
- Кристалл должен стоять на участке плоскости.

Взаимодействие с пользователем

- В текстовом файле настроек должны задаваться следующие **параметры**:
    - Координаты камеры (**CameraPos(x,y,z)**)
    - Направление взгляда камеры (**ViewDir(x,y,z)**)
    - Углы обзора камеры по горизонтали и вертикали (**ViewAngle(x,y)**)
    - Разрешение изображения (**Resolution(x, y)**)
    - Максимальное число отражений/преломлений (**MaxDepth**) - глубина трассировки, или минимальная энергия луча (**MinEnergy**)
    - Параметры, задающие поведение алгоритмов из бонусной части (показатель поглощения, количество лучей на пиксель при антиалиасинге, цвет кристалла и т.п.)

  Формат файла произвольный, должен быть описан в Readme.

- Минимум **3 различных сценария запуска**, реализованных через bat файлы. Сценарии должны отличаться параметрами рендеринга, использованием различных алгоритмов, положением камеры, демонстрацией различных дополнительных элементов задания. **Изображения**, полученные при использовании каждого сценария, нужно включить в архив **в папку img**.
- Результирующая картинка должна сохраняться в файл (можно использовать любой из общепринятых форматов) и может выводиться на экран. **Время работы** программы при заданных параметрах должно быть указано **в readme** и не должно превышать **2 минут** (хотя бы для одного набора параметров, указанного в readme).

Правила оформления работы

Внимание! При невыполнении указанных требований работа может не проверяться!

Архив с заданием должен быть залит в систему курса. В случае превышения максимального размера архива в системе нужно разбить его на части средствами архиватора. Заливать архив на файлообменники можно только в случае невозможности

залить его в систему, по предварительному согласованию с проверяющими.

Формат архива:

1. Папка **src** (исходный код)
    - Файлы исходного кода
    - Файлы проекта
    - **НЕ нужно** включать в архив файлы со следующими расширениями (пройти поиском и удалить): **.sdf**, **.ncb**, папки **Debug**, **Release, ipch**. Это позволяет существенно экономить место.
2. Папка **bin** (исполняемый код - конфигурация **Release, 32 бит**). *Обязательно проверьте, что **программа запускается из папки bin**. Желательно, на другой машине.*
    - Исполняемый файл
    - Библиотеки, необходимые для запуска
    - Данные (модели, текстуры, файл настроек). Дублировать данные в папке src не нужно.
3. Папка **img** (визуализированные изображения сцены + файлы настроек для них)
4. Файл **Readme.txt**
    - Фамилия, имя, отчество, группа
    - Операционная система
    - Оборудование (процессор, видеокарта, объём памяти)
    - Управление программой (формат задания настроек в файле настроек, описание интерфейса)
    - Время работы программы для каждого варианта настроек
    - Реализованные пункты из бонусной части

Дополнительная часть (максимум 5 баллов)

Алгоритм, эффект	**Способ демонстрации**	**Баллы (от 0 до X)**
BVH-дерево или kd- дерево	Сложный объект с большим числом полигонов	**+5**
Трассировка путей	Несколько диффузных объектов в сцене, светящиеся объекты в сцене, взаимное вторичное освещение и каустики.	**Максимум +5**: ● расчет вторичного диффузного освещения **+2** ● расчет каустиков **+2** ● прогрессивный расчет **+1** ● теневые лучи **+2** ● многократная выборка по значимости **+3**

		● MLT и другие более сложные методы **+5**
Фотонные карты	Каустики на плоскости, либо на объектах	**Максимум +5**: ● упрощенный сбор в текстуру на плоскости: с модификацией ближайшего текстеля **+1**, с обратной интерполяцией и модификацией 4 соседних текстелей **+2**. ● финальный сбор с фиксированным радиусом на плоскости **+3** ● финальный сбор с фиксированным радиусом при помощи ускоряющих структур на произвольных объектах **+4** ● финальный сбор k-ближайших (можно на плоскости) **+4** ● прогрессивная схема расчета **+1** ● более сложные методы сбора (например *[1]* - см в разделе Литература), фильтрация и другие методы улучшения результата) **+2**
Ambient Occlusion	Мягкое затенение объектов сцены другими объектами при освещении панорамой	**+2**
Текстурирование	Минимум одна текстурированная модель	**Максимум +3**: ● простая выборка текстур (билинейная интерполяция) **+1** ● использование mipmap уровней с трилинейной интерполяцией **+1** ● Более сложные техники выборки текстур (бикубическая или анизотропная) **+1**
Моделирование глубины резкости	Рендеринг как минимум для 3-х положений камеры	**+2**
Эффект поглощения внутри кристалла	Показатель поглощения должен задаваться в текстовом файле	**+1**
Антиалиасинг	Возможность	**+1**

	посмотреть на результат как с реализацией антиалиасинга, так и без нее	
Простые неполигональные объекты	Например, пузыри внутри кристалла	**+1**
Параллелизм	Ускорение расчёта при включении параллелизма. Количество потоков должно задаваться в файле настроек.	**+1**
Мягкие и/или цветные тени от кристалла	Протяжённый/площадной источник света, цветной кристалл	**Максимум +2:** ● мягкие тени от протяженных источников **+1** ● цветные тени **+1**

Материалы

Модели

Модель кристалла содержится в шаблоне к заданию.

Ресурсы по изучению трассировки лучей

1) ray-tracing.ru
2) raytracegroundup.com

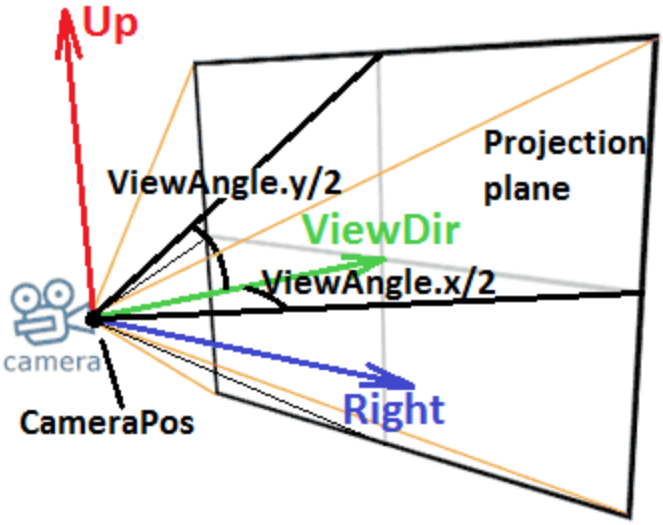
Литература

[1] Lars Schjøth, Jeppe Revall Frisvad, Kenny Erleben, and Jon Sporring. 2007. Photon differentials. In *Proceedings of the 5th international conference on Computer graphics and interactive techniques in Australia and Southeast Asia* (GRAPHITE '07). ACM, New York, NY, USA, 179-186. DOI=10.1145/1321261.1321293 http://doi.acm.org/10.1145/1321261.1321293

Подсказки к решению: база

Модель камеры

Предлагается реализовать модель перспективной камеры с точечной проекцией.



Камера не должна быть наклонена на бок, т.е. **ось z** (если принимать её за вертикальное направление), **ViewDir** и **Up** должны лежать в одной плоскости. Векторы **ViewDir**, **Up**, **Right** образуют ортонормированный базис и задают модельно-видовую матрицу.

Обратная трассировка лучей (краткое описание алгоритма)

1. Для каждого пиксела изображения вычисляется луч. Луч задается точкой испускания (положением камеры) и направлением. Для генерации направления луча можно использовать следующие функции.

   Простая генерация направления луча, проходящего через центр пиксела

```
// Подготовим 2 вектора, длина которых равна половине плоскости проекции
float3 Right_HalfPlane = Right * tan(ViewAngle.x/2);
float3 Up_HalfPlane = Up * tan(ViewAngle.y/2);

// x, y - координаты пиксела, w, h - разрешение изображения
float3 EyeRayDir(float x, float y, float w, float h)
{
  float xNorm = (x+0.5f)/w * 2.0f - 1.0f;
  float yNorm = (y+0.5f)/h * 2.0f - 1.0f;
  float3 ray_dir = ViewDir + Right_HalfPlane*xNorm + Up_HalfPlane*yNorm;
```

```
  return normalize(ray_dir);
}
```

Обратите внимание, что соотношение разрешения картинки по вертикали и горизонтали **h / w** должно быть равно соотношению сторон плоскости проекции **tan(ViewAngle.y/2) / tan(ViewAngle.x/2)**. Иначе изображение будет растянутое.

Генерация направления луча с учетом модельно-видовой матрицы и матрицы проекции

Данная функция может быть полезна, если необходимо получать геометрически совпадающие изображения при помощи OpenGL и трассировки лучей.

```
// x, y - координаты пиксела, w, h - разрешение изображения
float3 EyeRayDir2(float x, float y, float w, float h)
{
  float4 pos( 2.0f * (x + 0.5f) / w - 1.0f,
              -2.0f * (y + 0.5f) / h + 1.0f, 0.0f, 1.0f );
  pos = g_mViewProjInv*pos; // (mView*mProj)^-1*pos
  pos /= pos.w;
  pos.y *= (-1.0f);
  return normalize(pos.xyz);
}
```

2. Вычисляется пересечение луча с объектом сцены. В случае, если луч не пересекается ни с одним объектом, вычисляется его пересечение со сферическим окружением. При этом нужно посчитать сферические координаты направления этого луча и сделать выборку из сферической текстуры по соответствующим координатам, а не делать расчёт пересечения луча со сферой конечного радиуса.

3. Когда луч пересекается с поверхностью одного из объектов, выполняются следующие действия:

- Вычисление освещенности точки по локальной модели (Ламберта, Фонга). Идеальные зеркала и стекла при этом освещать не нужно, локальные модели освещения предназначены для учета рассеивающих свойств поверхности. Освещенность от каждого отдельного источника складывается.

- Трассировка отраженных и преломленных лучей (выполняется рекурсивно). При этом:

    - Если луч попал на зеркальную поверхность, вычисляется отраженный луч в соответствии с законом отражения света.

- Если луч попал на диффузную поверхность, в классической трассировке лучей дальнейшие переотражения не учитываются, и рекурсия останавливается. В стохастической трассировке лучей и трассировке путей используется отраженный случайный луч с равномерным или косинусоидальным распределением.

- Если луч попал на преломляющую поверхность, определяется направление идеально отраженного и преломленного лучей (согласно закону Снелла). На основе направления входящего луча и направлений отраженного и преломленного луча, используя формулы Френеля, определяются коэффициенты отражения **R** и пропускания **T**. То есть определяется, какая часть энергии приносится в эту точку отражённым лучом, а какая - преломлённым.

4. Для новых лучей (отраженного и преломленного) рекурсивно запускаем шаги 2 - 3.

5. После рекурсивной трассировки этих лучей нужно умножить их цвет (полученный из сферического окружения, а возможно и из цвета кристалла, если кристалл цветной) соответственно на коэффициенты **R** и **T** и получить результирующий цвет текущего луча.

После заданного числа отражений и преломлений (глубины трассировки) либо при достижении минимальной учитываемой интенсивности рекурсию нужно остановить.

Вычисление пересечений
Для вычисления пересечения луча с произвольной полигональной моделью можно для каждого треугольника использовать алгоритм пересечения луча с треугольником. Формулы для расчета пересечения луча со сферой предлагается вывести самостоятельно.

Формулы Френеля
При отражении и преломлении интенсивность лучей распределяется по закону Френеля. В базовой части требуется корректный расчет интенсивности по формулам Френеля без учёта поляризации. Направление преломленного луча считается согласно закону преломления света. Коэффициенты преломления реальных материалов есть здесь.

Код для расчета коэффициента отражения R от поверхности диэлектрика по формулам Френеля при идеальном отражении/преломлении:
ci - косинус угла падения (угла между падающим лучом и нормалью)
n - относительный коэффициент преломления границы раздела сред

```
float FresnelReflectance(float& ci, float& n)
{
```

```
  float ci2 = ci * ci;
  float si2 = 1.0f - ci2;
  float si4 = si2 * si2;
  float a = ci2 + n * n - 1.0f;
  float sqa = 2 * sqrtf(a) * ci;
  float b = ci2 + a;
  float c = ci2 * a + si4;
  float d = sqa * si2;
  return (b - sqa) / (b + sqa) * (1.0f + (c - d) / (c + d)) * 0.5f;
}
```

Коэффициент пропускания поверхности **T** считается исходя из закона сохранения энергии: **R + T = 1**.
Случай полного внутреннего отражения света, пришедшего изнутри среды, нужно обрабатывать отдельно: **R = 1**, **T = 0**.

Подсказки к решению (дополнительная часть задания)

Трассировка путей

Значительное отличие трассировки путей от трассировки лучей состоит в том, что при трассировке путей луч один на всем своем пути, а в трассировке лучей луч при взаимодействии с объектом расщепляется на 2 или более лучей. Т.е. трассировка лучей - рекурсивный алгоритм, а трассировка путей - нет (хотя и может быть реализована через рекурсию).

Для моделирования случайного луча с косинусоидельным распределением может быть полезна следующая функция, позволяющая по 2 случайным числам с равномерным распределением (r1 и r2) сгенерировать вектор с косинусоидальным (степень косинуса задается в аргументе power) распределением вокруг направления direction. В аргумент hit_norm необходимо передавать нормаль в точке поверхности, на которой генерируется отраженный луч. Это делается для предотвращения ситуации, когда отраженный луч уходит под поверхность.

```
float3 MapSampleToCosineDistribution(float r1, float r2, float3 direction, float3 hit_norm, float
power)
{
  float e = power;
  float sin_phi, cos_phi;

  sincosf(2*r1*3.141592654f, &sin_phi, &cos_phi);

  float cos_theta = pow(1.0f-r2,1.0f/(e+1.0f));
  float sin_theta = sqrtf(1.0f-cos_theta*cos_theta);
```

```
  float3 deviation;
  deviation.x = sin_theta*cos_phi;
  deviation.y = sin_theta*sin_phi;
  deviation.z = cos_theta;

  float3 ny = direction;
  float3 nx = normalize(cross(ny, make_float3(1.04,2.93f,-0.6234f)));
  float3 nz = normalize(cross(nx, ny));
  swap(ny,nz);

  float3 res = nx*deviation.x + ny*deviation.y + nz*deviation.z;

  float invSign = dot(direction, hit_norm) > 0.0f ? 1.0f : -1.0f;

  if(invSign*dot(res, hit_norm) < 0.0f) // reflected ray is below surface
    res = -nx*deviation.x + ny*deviation.y - nz*deviation.z;

  return res;
}
```

Обратите внимание, что каустики в трассировке путей получаются автоматически, если источники света имеют ненулевой размер и используется неявная стратегия сэмплирования.

Фотонные карты

Фотонная карта предназначена для того, чтобы запоминать информацию о падающих на поверхность фотонах. В простейшем случае с каждой поверхностью необходимо связать линейный массив, в который должны записываться координаты точки (двумерные, в пространстве поверхности) и цвет попавшего в поверхность фотона. Этот массив и называется фотонной картой. Вы можете сделать только один такой массив - для плоскости, на которой находится кристалл. Для расчета освещения при помощи метода фотонных карт на произвольных объектах, как правило, используются различные ускоряющие структуры (например, kd-дерево).

Текстурирование объектов сцены

Обратите внимание на необходимость реализации как минимум билинейной интерполяции при выборке из текстуры (за трилинейную или анизотропную фильтрацию начисляются баллы в соответствии с графой “текстурирование”). При отсутствии фильтрации совсем, баллы за текстурирование могут быть снижены на 0.5.

Моделирование глубины резкости

Моделирование эффекта глубины резкости можно сделать, смещая случайным образом

позицию луча (по x и y) в пределах некоторого диска, перпендикулярного направлению взгляда. Направление луча должно меняться так, чтобы получать на изображении область резкости, а не целиком размытую картинку.

Реализация эффекта поглощения

Поглощение света в сплошной однородной среде описывается законом Бугера-Ламберта-Бера (экспоненциальное затухание). Для реализации эффекта поглощения в кристалле нужно для каждого луча учитывать длину участков пути, пройденных ими в сплошной среде (в данном случае - в кристалле), и домножать интенсивность лучей, прошедших через кристалл, на рассчитанный коэффициент пропускания.

Антиалиасинг

Для борьбы с алиасингом предлагается пускать бОльшее количество лучей через различные точки пикселя (multisampling).

Реализация цветного кристалла и цветных теней

- Если реализовано поглощение в кристалле, следует задать разные коэффициенты поглощения по R, G, B каналам, что определит цвет кристалла.
- Если поглощения нет, нужно при пересечении с поверхностью кристалла домножать цвет луча, на цвет кристалла (простейшее моделирование поглощения). Например, если кристалл голубой, то компоненты r и g луча при прохождении через кристалл обнуляются.
- С теневыми лучами происходит то же самое. Домножая цвет источника света (например, белый) на цвет кристалла, получаем цвет тени.