УДК 608.2

Зенкевич Д. А., Преподаватель кафедры информационных и робототехнических систем, НИУ «БелГУ» Россия, г. Белгород

Zenkevich D.A., Assistant, Department of Information and Robotic Systems, NRU "BelSU" Russia, Belgorod

Гончаров Д.В., Преподаватель кафедры информационных и робототехнических систем, НИУ «БелГУ» Россия, г. Белгород

GoncharovD.V., Assistant, Department of Information and Robotic Systems, NRU "BelSU" Russia, Belgorod

Свиридова И.В., Ассистент кафедры прикладной информатики и информационных технологий, НИУ «БелГУ» Россия, г. Белгород

Sviridova I.V., Assistant, Department of Applied Informatics and Information Technology, NRU "BelSU" Russia, Belgorod

ФеоктистовА.С., ПреподавательСПОИнжиниринговогоколледжа НИУ «БелГУ» Россия, г. Белгород

Feoktistov A.S., Lecturer of STR of Engineering College NRU "BelSU" Russia, Belgorod

ПРОЦЕСС ПОЛУЧЕНИЯ ИЗОБРАЖЕНИЯ ПО ТРЕХМЕРНОЙ МОДЕЛИ В РЕАЛЬНОМ ВРЕМЕНИ

THE PROCESS OF OBTAINING AN IMAGE BY A THREE-DIMENSIONAL MODEL IN REAL TIME

Аннотация: в данной статье описан процесс получения изображения по трехмерной модели, описан алгоритм растеризации.

Ключевые слова: растеризация, трехмерная модель.

Abstract: This article describes the process of obtaining images using a three-dimensional model, describes the algorithm of rasterization.

Keywords:rasterization, three-dimensional model.

Терминология

ScreenSpaceReflaction (SSR) — это метод повторного использования данных пространства экрана для расчета отражений. Данная техника достаточно ресурсозатратна, но при грамотном ее использовании можно добиться отличных результатов, и при этом сохранить высокую производительность.

AmbientOcclusion (AO) — это метод затенения изображения, имитирующий глобальное затенение. Если говорить конкретно об EEVEE, то в данном случае корректнее будет использовать аббревиатуру SSAO (ScreenSpaceAmbientOcclusion). Еще точнее сказать, GTAO (GroundTruthAmbientOcclusion), но об этом мы уже поговорим в главе, посвященной этому методу. Алгоритм SSAO работает в режиме реального времени и имитирует рассеянное непрямое освещение и соответствующее затенение в экранном пространстве. SSAO был разработан отделом исследований немецкой компании Crytek, при разработке графических компонентов игрового движка CryEngine 2. Первая игра вышедшая на этом движке и использовавшая данную технологию была Crysis, которая вышла осенью 2007 года. Позже данный алгоритм и его модификации были использованы во многих других игровых движках, таких как UnrealEngine, Unity и многих других.

PBR (Physicallybasedrendering) — это философия компьютерной графики, которая стремится визуализировать графику максимально близко к тому, что мы видим в реальном мире. Конечно же, до реального мира ей очень далеко, но на практике это означает использование некоторого набора текстур (шероховатость, альбедо, карта нормалей, карта отражений...), которые дадут итоговый фотореалистичный результат. Также под этим термином стоит понимать, что один и тот же материал будет выглядеть

идентично (практически) в любом другом программном обеспечении с поддержкой PBR.

Утечка/протечка света (Lightleaking) — это проблема всех движков рендеринга в реальном времени. Данным термином называют появление света в тех местах, где по законам физики его быть не должно. Универсального решения данной проблемы не существует. В книге вы столкнетесь с некоторыми опциями, которые помогут полностью либо частично от этого избавиться.

Окклюдер (Occluder) — в контексте компьютерной графики окклюдером называют любой объект, который стоит на пути света и может отбрасывать тень. Это слово очень удобно использовать вместо «объект, который отбрасывает тень/непрозрачный объект».

Основная информация

Для рассмотрения процесса получения изображения по трёхмерной модели в реальном времени нам потребуется движок рендеринга, встроенный в некое программное обеспечение. EEVEE (ExtraEasyVirtualEnvironmentEngine) — это движок рендеринга в реальном времени, встроенный в Blender и использующий OpenGL. Он ориентирован на скорость и интерактивность при работе с материалами. Несмотря на то, что основной областью его применения является настройка материалов, он отлично подходит и для финальной визуализации.

EEVEE является PBR-рендером (Physicallybasedrendering), что делает его совместимым с огромным количеством других PBR-рендеров. Они используют одни и те же ноды для создания материалов и, несмотря на различные алгоритмы работы, выдают практически идентичный результат. Во многих случаях определить разницу «на глаз» затруднительно или же

невозможно вовсе. Пример представления объектов в пространстве представлен на рисунке 1.

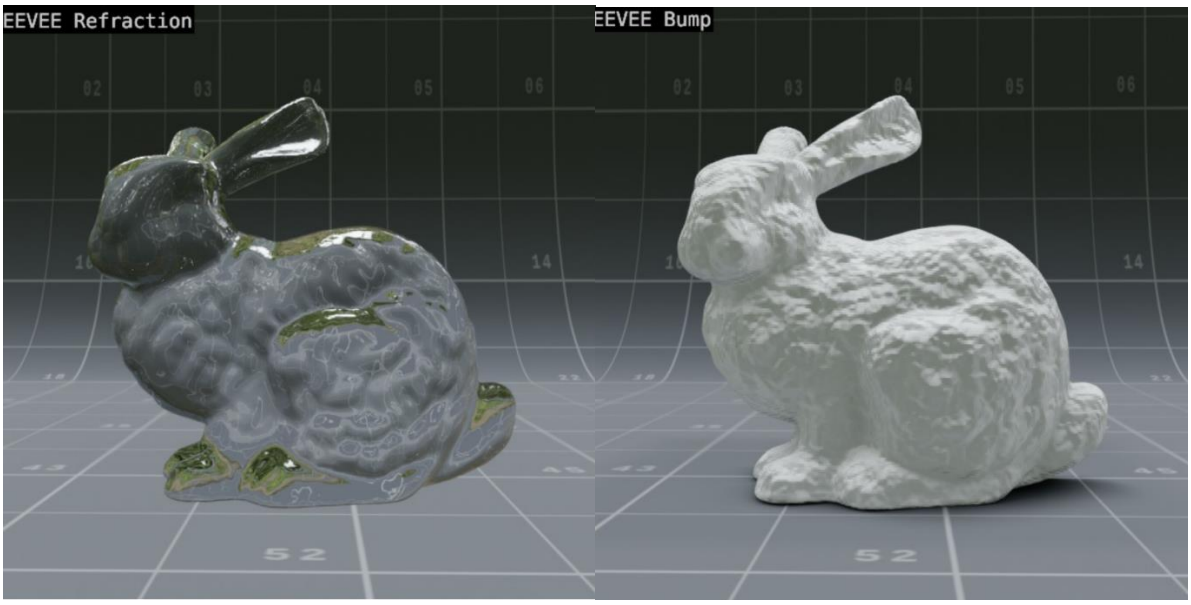

Рисунок 1 – Пример представления объекта

Из-за ограничений рендеринга в реальном времени не все функции доступны в EEVEE. Производительность сильно зависит от количества BSDF-нодов, присутствующих в дереве материалов. То же самое касается и количества активированных дополнительных эффектов, таких как ScreenSpaceReflaction (SSR), AmbientOcclusion (AO), Bloom и т.д.

Что такое растеризация?

Компьютерная графика в реальном времени уже давно использует метод, называемый растеризацией (rasterization), для отображения трехмерных объектов на двумерном экране. Это быстро. И результат получается очень хорошим, хоть и не всегда настолько хорош, как то, что может сделать трассировка лучей.

При растеризации объекты на экране создаются из мешей, состоящих из треугольников или многоугольников, которые создают трехмерные модели объектов. Вершины каждого треугольника пересекаются с вершинами других

треугольников разных размеров и форм. С каждой вершиной связано множество различной информации, такой как: ее положение в пространстве, информация о цвете, текстуре и ее «нормали», которая используется для определения того, как поверхность 7 объекта обращена к камере.

Затем компьютеры преобразуют треугольники 3D-моделей в пиксели на 2D-экране. Каждому пикселю может быть назначено начальное значение цвета из данных, хранящихся в вершинах треугольника. Дальнейшая обработка или «затенение» (shading) пикселя, включая изменение его цвета на основе того, как источники света в сцене освещают тот или иной участок меша, и применение одной или нескольких текстур к пикселю, объединяются, чтобы сформировать окончательный цвет, примененный к пикселю.

Это все трудозатратно, с точки зрения вычислений. Для всех объектов в сцене могут использоваться миллионы полигонов, а на экране может содержаться примерно 8 миллионов пикселей (4K-дисплей). И каждый кадр или изображение, отображаемое на экране, обычно обновляется с частотой от 30 до 90 раз каждую секунду.

Кроме того, буфер памяти (временное пространство), отведенный для ускорения процесса, используется для предварительной визуализации будущих кадров до их отображения на экране. Глубина или «z-буфер» также используется для хранения информации о глубине пикселя, чтобы гарантировать, что ближайшие к камере объекты в пространстве экрана отображаются на вашем мониторе, а объекты за ними остаются скрытыми.

Базовый алгоритм растеризации

На самом деле существует не один, а несколько алгоритмов растеризации, но эти алгоритмы основаны на одном и том же общем принципе. Все эти алгоритмы являются вариантами одной и той же идеи.

Именно на эту идею или принцип мы будем ссылаться, когда будем говорить о растеризации.

Что это за идея? Процесс рендеринга по существу можно разделить на две основные задачи: видимость и затенение. Растеризация, чтобы сказать вещи быстро, по сути, является методом решения проблемы видимости. Видимость состоит в том, чтобы определить, какие части трехмерных объектов видны камере. Некоторые части этих объектов могут быть недоступны, потому что они находятся вне видимой области камеры или скрыты другими объектами.

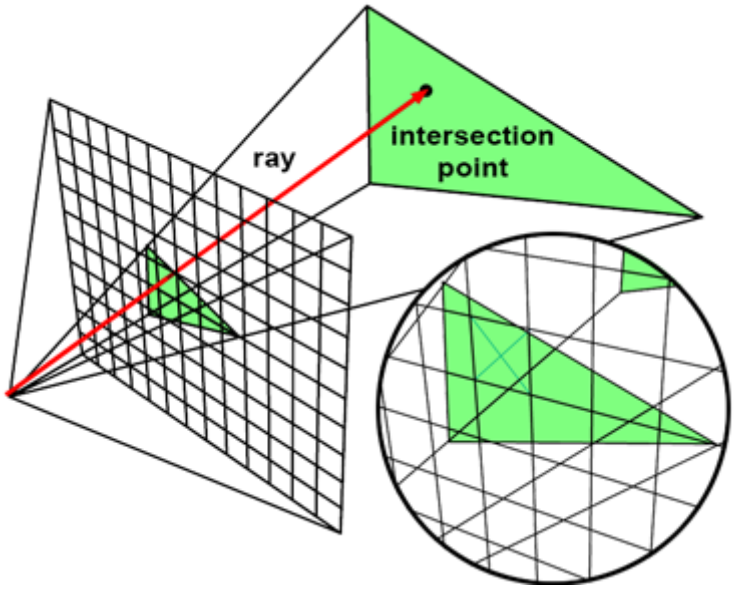

Рисунок 2

Решение этой проблемы может быть реализовано двумя способами. Можно либо проследить луч через каждый пиксель в изображении, чтобы узнать расстояние между камерой и любым объектом, который этот луч пересекает (если есть). Объект, видимый через этот пиксель, является объектом с наименьшим расстоянием пересечения (обычно обозначаемым как t). Это техника, используемая в трассировке лучей. В этом случае

создаетсяизображение, зацикливаясь на всех пикселях изображения, отслеживая луч для каждого из этих пикселей и затем выясняя, пересекаются ли эти лучи с какими-либо объектами в сцене. Другими словами, алгоритм требует двух основных циклов. Внешний цикл выполняет итерации по пикселю в изображении, а внутренний цикл выполняет итерации по объектам в сцене:

```
for (each triangle in scene) {
   // STEP 1: project vertices of the triangle using perspective projection
   Vec2f v0 = perspectiveProject(triangle[i].v0);
   Vec2f v1 = perspectiveProject(triangle[i].v1);
   Vec2f v2 = perspectiveProject(triangle[i].v2);
   for (each pixel in image) {
      // STEP 2: is this pixel contained in the projected image of the triangle?
if (pixelContainedIn2DTriangle(v0, v1, v2, x, y)) {
image(x,y) = triangle[i].color;
}
   }
}
```

Растеризация использует противоположный подход. Чтобы решить проблему наглядности, он фактически «проецирует» треугольники на экран, другими словами, мы переходим от трехмерного представления к двухмерному представлению этого треугольника, используя перспективную проекцию. Это легко сделать, проецируя вершины, составляющие треугольник, на экран (используя перспективную проекцию, как мы только что объяснили). Следующим шагом в алгоритме является использование некоторой техники для заполнения всех пикселей изображения, которые покрыты этим 2D-треугольником. Эти два шага показаны на рисунке 2. С

технической точки зрения они очень просты в выполнении. Шаги проецирования требуют только разделения перспективы и переназначения результирующих координат из пространства изображения в пространство растра.

Как выглядит алгоритм по сравнению с подходом трассировки лучей? Во-первых, вместо того, чтобы сначала перебирать все пиксели изображения, при растеризации, во внешнем цикле, перебираются все треугольники в сцене. Затем во внутреннем цикле перебираются все пиксели изображения и выясняется, содержится ли текущий пиксель в «проецируемом изображении» текущего треугольника (рисунок 3).

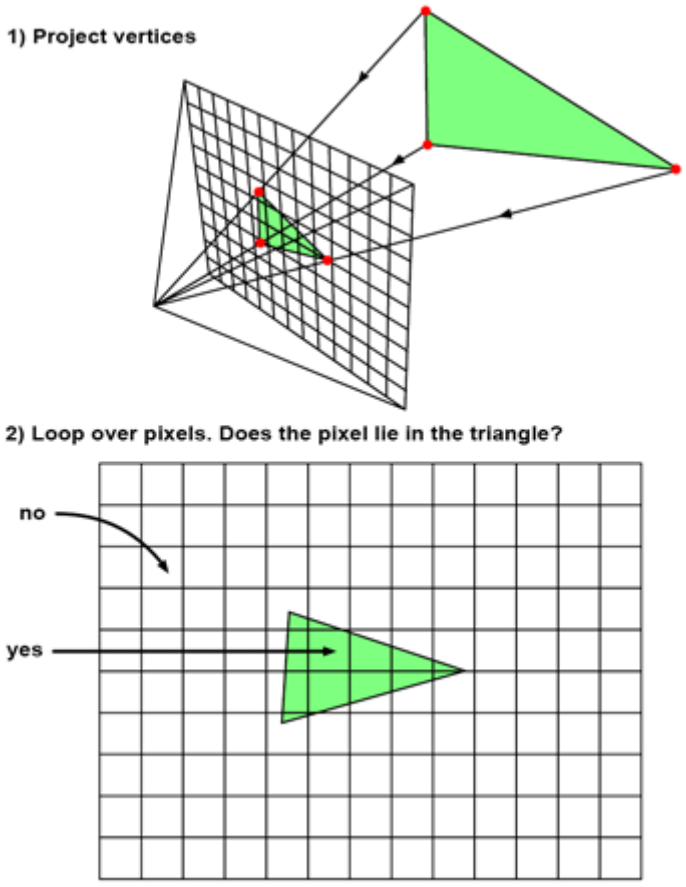

Рисунок 3

Другими словами, внутренние и внешние циклы двух алгоритмов меняются местами.

```
for (each triangle in scene) {
   // STEP 1: project vertices of the triangle using perspective projection
   Vec2f v0 = perspectiveProject(triangle[i].v0);
   Vec2f v1 = perspectiveProject(triangle[i].v1);
   Vec2f v2 = perspectiveProject(triangle[i].v2);
   for (each pixel in image) {
      // STEP 2: is this pixel contained in the projected image of the triangle?
if (pixelContainedIn2DTriangle(v0, v1, v2, x, y)) {
image(x,y) = triangle[i].color;
}
   }
}
```

В итоге:

- Преобразование геометрии в треугольники упрощает процесс. Если все примитивы преобразованы в примитив треугольника, мы можем написать быстрые и эффективные функции для проецирования треугольников на экран и проверки, находятся ли пиксели в этих 2D треугольниках.
- Растеризация является объектно-ориентированной. В ней проецируется геометрия на экран и определяется их видимость, перебирая все пиксели изображения.
- Алгоритм опирается в основном на два метода: проецирование вершин на экран и выяснение, находится ли данный пиксель в 2D-треугольнике.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ

1. Аббасов, И.Б. Двухмерное и трехмерное моделирование в 3dsMAX / И.Б. Аббасов. - М.: ДМК, 2012. - 176 с.

2. Ганеев, Р.М. 3D-моделирование персонажей в Maya: Учебное пособие для вузов / Р.М. Ганеев. - М.: ГЛТ, 2012. - 284 с.

3. Зеньковский, В. 3D-моделирование на базе VuexStream: Учебное пособие / В. Зеньковский. - М.: Форум, 2011. - 384 с.

4. Зеньковский, В.А. 3D моделирование на базе VuexStream: Учебное пособие / В.А. Зеньковский. - М.: ИД Форум, НИЦ Инфра-М, 2013. - 384 с.

5. Климачева, Т.Н. AutoCAD. Техническое черчение и 3D-моделирование. / Т.Н. Климачева. - СПб.: BHV, 2008. - 912 с.

6. Пекарев, Л. Архитектурное моделирование в 3dsMax / Л. Пекарев. - СПб.: BHV, 2007. - 256 с.