УДК 004.925


А.А. Суконщиков, В.В. Круглов
Вологодский государственный университет

ВИЗУАЛИЗАЦИЯ ПАДАЮЩИХ ТЕНЕЙ В 3D-ГРАФИКЕ РЕАЛЬНОГО ВРЕМЕНИ ПО ТЕХНОЛОГИИ SHADOW MAPPING С ИСПОЛЬЗОВАНИЕМ DIRECTX 11

Одним из важнейших эффектов для компьютерной графики являются падающие тени. Их наличие крайне важно для восприятия трехмерной сцены, так как зачастую только по теням можно судить о взаимном расположении объектов. При этом в графических API и конвейере визуализации видеокарты нет отдельных алгоритмов, напрямую реализующих тени. Но это не означает, что эффективная визуализация теней невозможна. В данном тексте рассмотрены основные принципы и техники, позволяющие в своем сочетании осуществить визуализацию падающих теней в трехмерной графике реального времени. Также описаны отдельные возможности DirectX 11, позволяющие упростить этот процесс.

3D-графика реального времени, DirectX 11, Shadow mapping, Projective texturing.

Реализация трехмерной графики реального времени сегодня практически всегда основывается на двух неразрывно связанных техниках: программировании состояния видеокарты для подготовки к визуализации при помощи вызовов графического API и непосредственно визуализации – вызова графического конвейера, аппаратно поддерживаемого видеокартой. В процессе визуализации на основе геометрического описания объектов трехмерной сцены формируется растровое изображение, которое и передается на экран. Предварительно установленные через вызовы API состояния видеокарты служат для конвейера в качестве параметров, влияющих на его работу. Отдельно стоит упомянуть шейдеры – небольшие программы, описывающие работу программируемых стадий конвейера, которые также должны быть переданы на видеокарту перед вызовом конвейера.

Из вышесказанного следует, что, с одной стороны, программирование трехмерной графики реального времени производится в достаточно жестко установленных рамках (единый конвейер, определенные программируемые стадии, конкретно установленные состояния), а с другой – в предоставляемых разработчикам программных возможностях (шейдеры, вызовы API) заложен достаточный потенциал для множества всевозможных графических эффектов.

Одним из важнейших для компьютерной графики эффектов являются падающие тени, то есть тени, которые освещенный объект отбрасывает на другие объекты. Наличие падающих теней крайне важно для восприятия трехмерного изображения, ведь, как отмечено в [1], зачастую именно по ним можно судить о положении объекта в пространстве. Например, если тень от стоящего на земле самолета будет всегда находиться прямо под ним, то тень от взлетевшего самолета может находиться на различном расстоянии от него в зависимости от положения Солнца. Таким образом, при взгляде на этот самолет сверху только по одной лишь тени можно будет судить о его положении.

В графических API или конвейере визуализации видеокарты не предусмотрено отдельных специальных алгоритмов, непосредственно реализующих тени, но предоставляемых ими программных возможностей общего назначения вполне достаточно для воплощения теней без существенных потерь в производительности.

Целью данного материала является подробный разбор общего существующего подхода к реализации динамических падающих теней, а также его адаптация и применение для визуализации средствами конкретного графического API Direct3D 11 в составе DirectX 11.

В ходе достижения поставленной цели будут решаться задачи, связанные с изучением и описанием общей технологии визуализации падающих теней и связанных процедур, с поиском составляющих API DirectX, при помощи которых будет производиться реализация теней и с применением полученных знаний для визуализации теней на конкретном примере.

Проективное текстурирование

В основе описываемого в данной работе подхода к реализации теней лежит техника проективного текстурирования. Согласно [2], задача его состоит в том, чтобы текстура (растровое изображение, хранимое в памяти как прямоугольная матрица) накладывалась на трехмерную сцену со стороны подобно изображению с кинопроектора в кинотеатре.

При обычной визуализации сначала (на стадии вершинного шейдера) каждая вершина, из которых состоит любой трехмерной объект, проецируется из трехмерного пространства на плоскость, соответствующую нашему экрану. Затем (при растеризации) все эти спроецированные вершины преобразуются в пиксели, которые и формируют изображение на экране. Таким образом, проецируя вершины на плоскость экрана, мы определяем, какой из пикселей экрана будет соответствовать той или иной вершине. А те пиксели, которым не попало в соответствие ни одной вершины, получают свои данные путем интерполяции.

Проективное же текстурирование основано на том, что помимо плоскости экрана добавляется еще одна плоскость, на которую будут проецироваться вершины – плоскость накладываемой текстуры. При проецировании вершин (на стадии вершинного шейдера) на эту дополнительную плоскость будет получено соотношение типа «вершина – координаты пикселя на текстуре». После растеризации координаты вершин будут преобразованы в параметры пикселей на экране. Таким образом, взяв координаты одного такого пикселя экрана, можно по имеющемуся соотношению получить соответствующие координаты пикселя на проецируемой текстуре и прибавить цвет пикселя текстуры к цвету пикселя экрана. По результатам выполнения данного действия для всех пикселей экрана и будет достигнут эффект спроецированной текстуры.

Наглядно логика проективного текстурирования представлена на рисунке ниже. При обработке каждого пикселя результирующего изображения через интерполированные координаты объекта сцены происходит обращение к связанным с ними координатам проецируемой текстуры.

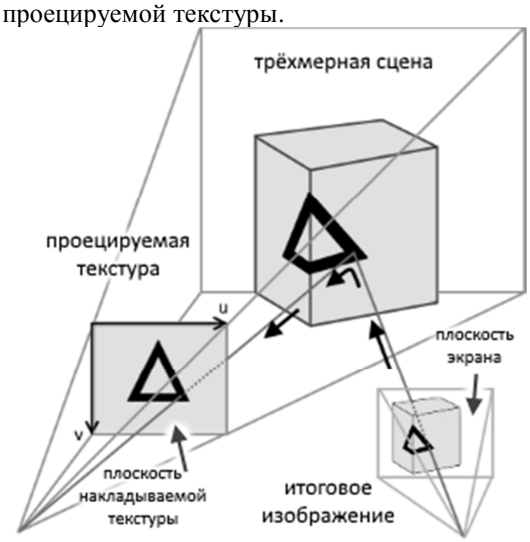

Рис. 1. Логика визуализации с использованием проективного текстурирования Shadow mapping (наложение теней)

Общая логика реализации теней согласно [3] может быть разделена на два шага по количеству вызовов конвейера визуализации и состоит в следующем.

На первом шаге выполняется фиктивная визуализация всей трехмерной сцены из позиции источника света, порождающего тень. Фиктивна она потому, что в результате не будет формироваться изображение сцены. Но при этом в процессе такой визуализации производится заполнение буфера глубины – отдельной текстуры, в которую записывается расстояние ближайших к зрителю (в данном случае – к источнику) фрагментов сцены. Данные из буфера глубины будут использоваться для построения теней. Буфер глубины, используемый для реализации теней, именуется картой теней (shadow map).

На втором шаге производится обычная визуализация с позиции зрителя, наблюдающего сцену. Но после растеризации, на стадии пиксельного шейдера, когда вычисляются значения цветов каждого пикселя визуализируемого объекта, применяется проективное текстурирование, где в роли проецируемой текстуры выступает карта теней.

Как было отмечено ранее, каждый пиксель визуализируемого объекта содержит свои координаты в пространстве, интерполированные по вершинам геометрии данного объекта. Имея их и координаты источника света, формирующего тень, можно вычислить расстояние от источника до каждого конкретного пикселя, что и выполняется.

При проецировании карты теней на сцену каждый пиксель получает в соответствие значение из буфера глубины, содержащее расстояние от источника до ближайшего к нему фрагмента поверхности.

Получив таким образом расстояние от источника до ближайшего фрагмента сцены в каждом пикселе и расстояние от него же до каждого конкретного фрагмента сцены, можно, сравнив их, определить: находится ли данный конкретный фрагмент в тени или же освещен падающим светом. Если данные значения совпадают, то данный фрагмент – ближайший к источнику и не может быть затенен. В противном случае свет от источника до данного фрагмента перекрыт более близким объектом, следовательно – фрагмент находится в тени.

Получение карты теней, ортографическая проекция

Для отображения любых объектов трехмерной сцены их необходимо спроецировать из пространства на плоскость экрана. В этом случае обычно используется перспективное проецирование, так как перспектива (уменьшение размеров объектов с увеличением расстояния до них) присуща реальному миру в том виде, как наблюдает его человек.

Но для получения карты теней такое проецирование не подходит. Как можно увидеть на примере рисунка 1: размер проецируемой на объект текстуры зависит от того, насколько далеко этот объект расположен от центра проекции. Если использовать перспективную проекцию и для получения карты теней, то эффект будет тот же: тени будут увеличиваться при отдалении объекта от источника света. Но при освещении солнечным светом (т.е. когда источник находится очень далеко) размер теней всегда одинаков. Поэтому для имитации солнечного света можно либо разместить источник очень далеко и, оперируя большими числами, получать расстояние до объекта с заметными ошибками, либо согласно [4] использовать ортографическое проецирование, где перспектива не учитывается. Пример ортографического проецирования координат объекта на плоскость показан на рисунке 2.

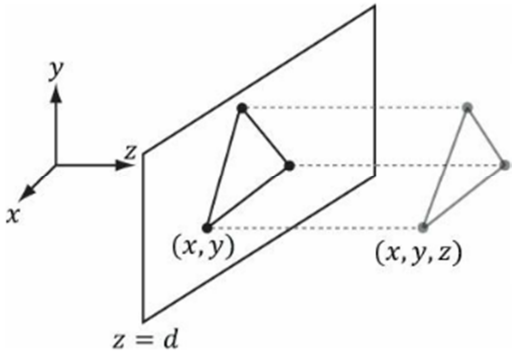

Рис. 2. Проецирование на плоскость при ортографической системе координат

Для выполнения ортографического проецирования необходимо сначала преобразовать координаты объектов сцены в ортографическую систему координат из перспективной.

В трехмерной графике для работы с расположением геометрии объектов в пространстве используется декартова система координат с тремя осями: x, y, z. Для каждой вершины ее расположение по всем трем осям хранится в виде вектора. Для преобразования из одного координатного пространства в другое используют матрицы. Результатом векторно-матричного произведения является новый вектор, указывающий расположение той же вершины, но уже в другом пространстве координат.

Процесс получения матрицы, преобразующей координаты вершин в ортографическое пространство координат, изложен в [4]. Сама матрица, участвующая в векторно-матричном произведении, представлена на рисунке 3. Единица в конце является признаком того, что вектор обозначает позицию вершины, а у позиции может быть не только направление, но и расстояние относительно центра пространства координат.

Рис. 3. Векторно-матричное произведение с матрицей преобразования в ортографическое пространство координат

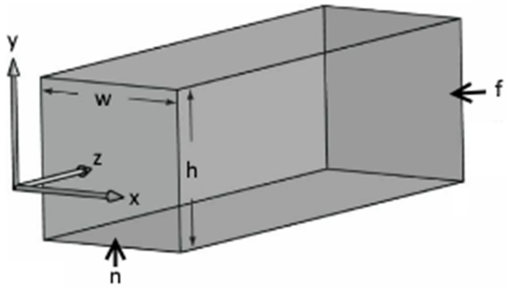

Рис. 4. Ортографическое пространство координат, в которое попадают вершины после преобразования

Собственные тени, связь с системой освещения

В трехмерной графике для визуализации освещенности объектов применяется алгоритм его расчета, где результирующее освещение формируется из трех составляющих: ambient (непрямое освещение), diffuse (освещение шершавого объекта) и specular (блики от света на гладком объекте). Как видно из рисунка 5, модель освещения сама формирует собственное затенение объекта.

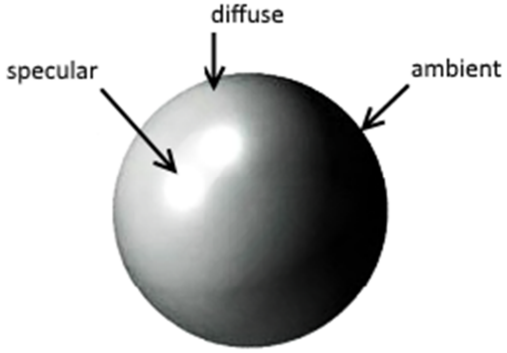

Рис. 5. Составляющие модели освещения в трехмерной графике

Более того, данный алгоритм в пиксельном шейдере и использует для расчетов интерполированные по пикселям значения нормалей каждой вершины, что делает освещение плавным.

Алгоритм же визуализации теней основан на проективном текстурировании с использованием текстур относительно невысокого разрешения, а значит – дает менее плавный результат.

При простом применении техники shadow mapping в цвета тени окрашиваются все фрагменты, не находящиеся под прямым освещением. Это ведет к тому, что собственные тени модели освещения перекрываются тенями техники shadow mapping, выдавая менее качественное и плавное изображение.

Решение данной проблемы достаточно простое – нужно запретить значению, полученному при расчете падающей тени, влиять на ambient-составляющую освещения. Вместо этого значение тени будет модифицировать только diffuse и specular-составляющие, не нарушая тем самым собственные тени объектов.

Особенности воплощения shadow mapping с использованием API DirectX 11

Для того чтобы иметь возможность создавать и использовать карту теней (shadow map), прежде всего нужно создать пустую текстуру нужного разрешения с bind-флагами: D3D11_BIND_DEPTH_STENCIL и D3D11_BIND_SHADER_RESOURCE, что позволит использовать ее сперва в качестве буфера глубины (при генерировании карты теней), а затем – для проективного текстурирования.

В Direct3D для передачи в память видеокарты используются не сами текстуры, а специальные объекты (интерфейсы) – представления (view). Соответственно, для созданной текстуры нужно сформировать Depth Stencil View и Shader Resource View. Детально процесс получение view'ов изложен в [4].

При сравнении расстояния от незатененного фрагмента до источника с соответствующим значением в карте теней может возникнуть непредвиденное расхождение значений (хотя они должны быть равны) из-за того, что карта теней представляет собой растровое изображение. Это приводит к накладыванию теней на те фрагменты, где их быть не должно. Поэтому для вызова конвейера, генерирующего карту теней, желательно задать состояние растеризации

(rasterizer state) с набором параметров, сглаживающих последующие ошибки проективного текстурирования.

```
typedef struct D3D11_RASTERIZER_DESC {
        [...]
        INT DepthBias;FLOAT
        DepthBiasClamp;FLOAT
        SlopeScaledDepthBias;
        [...]
} D3D11_RASTERIZER_DESC;
```

При проецировании карты теней на объекты сцены, когда производится наложение теней, в DirectX 11 имеются встроенные возможности, позволяющие получить более сглаженные тени.

Если для простого чтения значения из текстуры используется SamplerState, позволяющий получить только одно значение, то для обращения к карте теней имеется специальный SamplerState – SamplerComparisonState. При его использовании из карты теней сначала автоматически считывается 4 значения, а потом производится билинейная интерполяция для вычисления одного усредненного на их основе. Такая техника, носящая название PCF (Percentage Close Filtering) позволяет формировать тени с мягкими краями без серьезных потерь в производительности.

В результате суммирования всего вышесказанного накапливается достаточно информации, чтобы выполнить визуализацию сцены с реализацией теней.

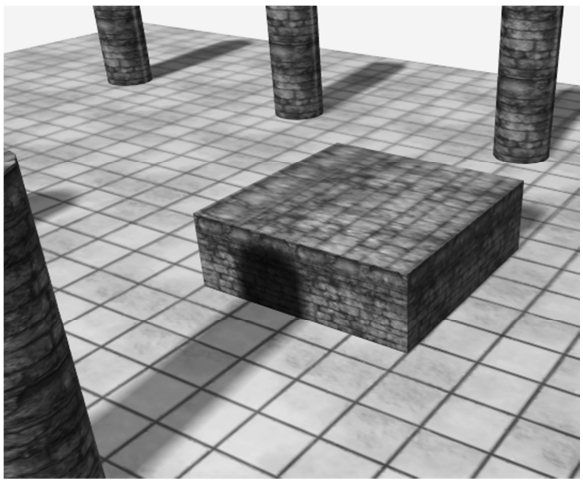

Рис. 6. Визуализация трехмерной сцены с падающими тенями

Результат визуализации сцены с наличием динамических (т.е. на каждом кадре) генерируемых падающих теней по технологии shadow mapping с использованием графического API Direct3D 11 продемонстрирован на рисунке 6.

При визуализации данной сцены была использована карта теней с разрешением 1024×1024 пикселей. Генерация карты теней выполнена предварительной установкой соответствующего RasterizerState. Взятие значения из проецируемой карты теней для каждого фрагмента производилось с использованием SamplerComparisonState, что обеспечило достаточный уровень сглаживания даже для относительно небольшого размера текстуры карты теней.

Таким образом, в результате исследования данной темы были рассмотрены, исследованы и на основании этого подробно изложены основные принципы и техники, лежащие в основе процесса визуализации падающих теней в трехмерной графике реального времени, был изучен набор вспомогательных программных возможностей DirectX 11, которые могут быть использованы для упрощения реализации теней и избавления от ошибок их отображения. В качестве подтверждения корректности полученных знаний была произведена визуализация тестовой сцены.

Литература

1. Shadow Mapping: GPU-based Tips and Techniques / AMD Developer Central. – URL : https://developer.amd.com/wordpress/media/2012/10/Isidoro-ShadowMapping.pdf (дата обращения: 03.03.2022). – Text : Electronic.

2. Projective Texturing. – URL : http://www.rastertek.com/dx11tut43.html (дата обращения: 03.03.2022). – Text : Electronic.

3. Tutorial 16 : Shadow mapping. – URL : http://www.opengl-tutorial.org/intermediatetutorials/tutorial-16-shadow-mapping (дата обращения: 03.03.2022). – Text : Electronic.

4. Luna, F. Introduction to 3D game programming with Directx® 11 / Luna F. – Copy-right © Mercury learning and information llc, 2012. – 754 p.

A.A. Sukonshchikov, V.V. Kruglov
Vologda State University

RENDERING SHADOWS IN REALTIME 3D-GRAPHICS WITH DIRECTX 11 USING THE SHADOW MAPPING TECHNIQUE

Falling shadows are one of the main effects in 3D computer graphics. Its presence is important for perception of 3D scenes because shadows often become the only factor which allows to understand the objects' location. Despite this there aren't any special efficient built-in algorithm neither in graphics API nor in rendering pipeline. However, that doesn't mean that it is impossible to render shadows in realtime 3D graphics efficiently. In this text we cover the main principles and techniques which, if combined, make it possible to efficiently visualize falling shadows in realtime 3D graphics. Here we also describe DirectX 11 features which make this process easier.

Realtime 3D graphics, DirectX 11, Shadow mapping, Projective texturing.