УДК 004.946

РЕНДЕРИНГ В ТЕХНОЛОГИЯХ ДОПОЛНЕННОЙ РЕАЛЬНОСТИ НА МОБИЛЬНЫХ ПЛАТФОРМАХ С ИСПОЛЬЗОВАНИЕМ VUFORIA

Маслов А.С., Белов Ю.С.

Московский государственный технический университет имени Н.Э. Баумана, филиал, Калуга, e-mail: maslow.tema@yandex.ru

За последние несколько лет дополненная реальность (AR) стала все более распространенной технологией потребительского уровня. Основной движущей силой развития дополненной реальности была эволюция мобильных и портативных устройств и алгоритмов компьютерного зрения. Несмотря на то, что системы дополненной реальности не слишком фокусируются на визуальных эффектах, увеличение реальности возможно только с помощью дополнительной графики. Это делает рендеринг еще одним важным аспектом дополненной реальности. Цель данной статьи – дать более углубленное понимание процесса визуализации контента в приложениях с технологиями дополненной реальности. В данной работе описаны различия между программным и аппаратным рендерингом, а также произведено сравнение аппаратного и программного рендеринга по времени визуализации контента. Рассмотрены существующие решения для визуализации 2D и 3D графики, такие как OpenGL, OpenGL ES, также рассмотрен конвейер визуализации OpenGL 2.0. Рассмотрены такие шаги, как вершинный шейдер и пиксельный шейдер (фрагментный шейдер). Дано понимание того, как именно происходит обработка изображения с камеры для распознавания маркера и рендеринга объекта относительно камеры и данного маркера.

Ключевые слова: дополненная реальность, особенности маркера, отслеживание маркера, рендеринг, Vuforia, OpenGL, OpenGL ES

RENDERING IN AUGMENTED REALITY TECHNOLOGIES ON MOBILE PLATFORMS USING VUFORIA

Maslov A.S., Belov Yu.S.

Bauman Moscow State Technical University, branch, Kaluga, e-mail: maslow.tema@yandex.ru

Over the past few years, augmented reality (AR) has become an increasingly common consumer-level technology. The main driving forces behind the development of augmented reality were the evolution of mobile and portable devices and computer vision algorithms. Although augmented reality systems don't focus too much on visual effects, the only way to increase reality is to use additional graphics. This makes rendering another important aspect of augmented reality. The purpose of this article is to provide a more in-depth understanding of the content visualization process in applications with augmented reality technologies. This paper describes the differences between software and hardware rendering, and compares hardware and software rendering in terms of content rendering time. Existing solutions for rendering 2D and 3D graphics such as OpenGL, OpenGL ES are considered, as well as The OpenGL 2.0 visualization pipeline.steps such as vertex Shader and pixel Shader (fragment Shader) are Considered. It gives an understanding of how the camera image is processed to recognize the marker and render the object relative to the camera and this marker.

Keywords: augmented reality, marker features, tracking marker, rendering, Vuforia, OpenGL, OpenGL ES

Исследователи компьютерной графики в основном имеют дело с низкоуровневыми графическими библиотеками, такими как OpenGL. Существующие встроенные 3D-графические библиотеки на мобильных платформах, таких как Android Graphics, обычно не могут соответствовать требованиям систем дополненной реальности. OpenGL ES и Direct3D mobile стали незаменимыми для 3D-приложений на мобильных платформах (Android, iOS и Windows Mobile соответственно). На данный момент они являются единственными эффективными низкоуровневыми графическими библиотеками.

Еще одним решением для рендеринга на мобильном устройстве является удаленный рендеринг. При удаленном рендеринге сервер с вычислительно высокой мощностью выполняет задачу генерации изображения и отправляет окончательную модель на мобильное устройство [1]. Была предложена система, в которой кластер ПК способен обрабатывать сеансы удаленной визуализации на основе потоковой передачи MPEG-видео со сложными 3D-моделями. Предложенная структура позволила мобильным устройствам, таким как смартфоны, визуализировать объекты, состоящие из миллионов текстурированных полигонов на стороне сервера и мультимедийных возможностей на стороне клиента, предполагая очень быстрое сетевое соединение. Это, конечно, не очень практично в условиях низкого качества связи. Поэтому данная статья фокусируется на оригинальной визуализации.

Цель исследования: дать более углубленное понимание процесса визуализации контента в приложениях с технологиями

дополненной реальности. Показать различие программного и аппаратного рендеринга контента. Рассмотреть существующие решения для визуализации 2D и 3D графики, такие как OpenGL и OpenGL ES. Рассмотреть конвейер визуализации в OpenGL 2.0. Определить, как именно происходит обработка изображения с камеры для распознавания маркера и рендеринга объекта относительно камеры и данного маркера.

Методы исследования.
Программный рендеринг
в сравнении с аппаратным рендерингом

Рендеринг относится к процессу создания автоматизированных изображений с помощью компьютерных программ [2]. Это может быть выполнено с помощью аппаратного или программного рендеринга. Рендеринг программного обеспечения осуществляется исключительно с помощью компьютерного кода или приложений. Аппаратный рендеринг выполняется с помощью компьютерного чипа, который возвращает изображения непосредственно на экран.

Программный рендеринг обрабатывается без помощи какого-либо оборудования и выполняется в процессоре, в то время как аппаратный рендеринг опирается на графический блок (GPU). Программный рендеринг медленнее аппаратного (рис. 1), поскольку GPU – это специализированная вычислительная архитектура, разработанная с нуля с учетом рендеринга, и она может обрабатывать большие блоки данных параллельно, что имеет решающее значение для алгоритмов, используемых в компьютерной графике. В идеале программные алгоритмы рендеринга должны быть переведены непосредственно на аппаратное обеспечение. Однако это невозможно, поскольку аппаратный и программный рендеринг используют два совершенно разных подхода. Более подробно:

– Программное обеспечение рендеринга содержит 3D сцену, подлежащую рендерингу, или некоторые соответствующие ее разделы в памяти и дискретизируют ее пиксель за пикселем. Другими словами, сцена статична и всегда присутствует, но визуализатор имеет дело с одним пикселем за раз.

– Аппаратный рендеринг работает наоборот. Все пиксели присутствуют постоянно, но рендеринг видит сцену по одному треугольнику за раз, загружая каждый из них в буфер кадров. Аппаратное обеспечение не имеет понятия о других объектах, только один треугольник известен в определенное время.

Начиная с Android 3.0 (уровень API 11), конвейер рендеринга Android поддерживает аппаратное ускорение. Это означает, что все операции рисования, выполняемые на холсте, используют графический процессор. Из-за увеличения требуемых ресурсов приложения с включенным аппаратным ускорением потребляют больше памяти.

При низкоуровневом рендеринге разработчик имеет полный контроль над процессом. Объекты определяются как набор вершин и перерисовываются на каждом кадре. Какой бы ни была система визуализации, этот подход является самым низким уровнем. OpenGL и Direct3D Mobile являются наиболее широко используемыми низкоуровневыми графическими библиотеками. Direct3D Mobile доступен только для платформы Windows Mobile.

С помощью OpenGL можно создавать 2D и 3D графику с помощью графического процессора. OpenGL использует так называемый графический конвейер, показанный на рис. 2, для преобразования примитивов (точек, линий и т.д.) в пиксели. Основная идея, лежащая в основе конвейера, заключается в следующем. Сначала вводится ряд вершин слева, загружая их в конвейер. Затем выполняется несколько промежуточных шагов. В конце конвейера мы получаем желаемое изображение.

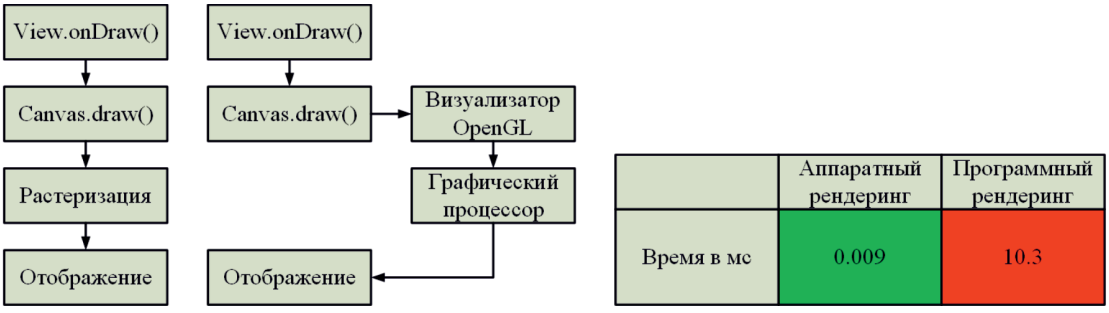

Рис. 1. Сравнение программного (слева) и аппаратного (справа) рендеринга на Android

Рис. 2. Конвейер рендеринга OpenGL 2.0

Вершинный шейдер и пиксельный шейдер – это единственные шаги, которые разработчик может запрограммировать в конвейере. Остальные шаги в конвейере выполняются автоматически. Шейдеры в основном передают данные на графический процессор и сообщают ему, какие вычисления будут выполняться.

OpenGL ES (OpenGL для встраиваемых систем) – это подмножество OpenGL. Несмотря на то, что он предлагает большую часть функциональности OpenGL, он был разработан для поддержки встроенных систем, таких как смартфоны, планшеты, КПК и игровые приставки, поэтому он более легкий, чем его предшественник. Например, каждая функция, которую он предоставляет, может быть непосредственно сопоставлена с базовой реализацией. Это упрощает разработку драйвера и уменьшает размер кода драйвера [3]. Кроме того, некоторые избыточности OpenGL были удалены.

Существует широкий спектр графических библиотек более высокого уровня, работающих поверх OpenGL ES, таких как libGDX28, который является очень популярным и современным кроссплатформенным графическим фреймворком Java, и AndEngine, который является широким 2D игровым движком.

Вершинный шейдер: вершинный шейдер запускается на каждой вершине, подлежащей рендерингу, то есть если мы рендерим спрайт, содержащий только четыре вершины, то вершинный шейдер будет запущен четыре раза, чтобы вычислить цвет и другие атрибуты для каждой вершины в спрайте.

Фрагментный шейдер: фрагментный шейдер запускается на каждом пикселе экрана, что означает, что если мы рендерим полный экран с высоким разрешением на телефоне Android, то этот шейдер будет вызван 1920×1080 раз.

Эти шейдеры не могут существовать поодиночке, их нужно вызывать вместе. Они вместе образуют программу. Во-первых, вершинный шейдер определяет атрибуты для каждой вершины на экране. Затем все пиксели делятся на подмножество пикселей, которые выполняются с помощью фрагментного шейдера. Наконец, полученные пиксели рисуются на экране.

Поскольку мы работаем на платформе Android, нам нужен интерфейс между OpenGL ES и базовой оконной системой родной платформы Android. EGL (Embedded System Graphics Library) делает это за нас. Он обрабатывает управление графическим контекстом, привязки поверхности и буфера, синхронизацию рендеринга и обеспечивает «высокопроизводительный, ускоренный, смешанный режим 2D и 3D рендеринга».

Теперь, перед последним шагом, который является фактическим рендерингом нашей модели, нам нужно инициализировать процесс рендеринга. Достигается это путем создания объекта модели, инициализируя необходимые обработчики с помощью программы, которую создаем из шейдеров. Эти обработчики будут использоваться в реальном рендеринге, который вызывается на каждом кадре. Реалистичное увеличение трехмерной среды может быть достигнуто только в том случае, если объекты непрерывно визуализируются в соответствии с их назначенным местоположением в трехмерном пространстве и точкой зрения камеры.

Результаты исследования и их обсуждение

Правильная система дополненной реальности требует задания трех преобразований, которые связывают системы координат виртуальных объектов, окружающую среду, камеру и создаваемое ею изображение.

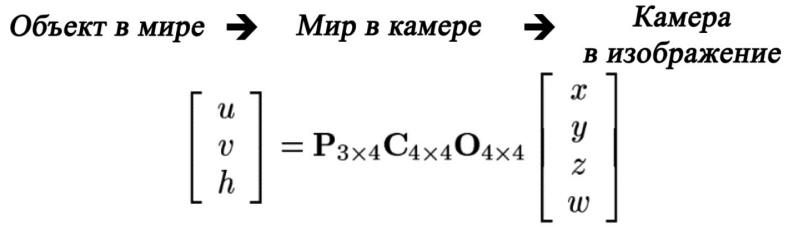

Рис. 3. Преобразование объекта в изображение

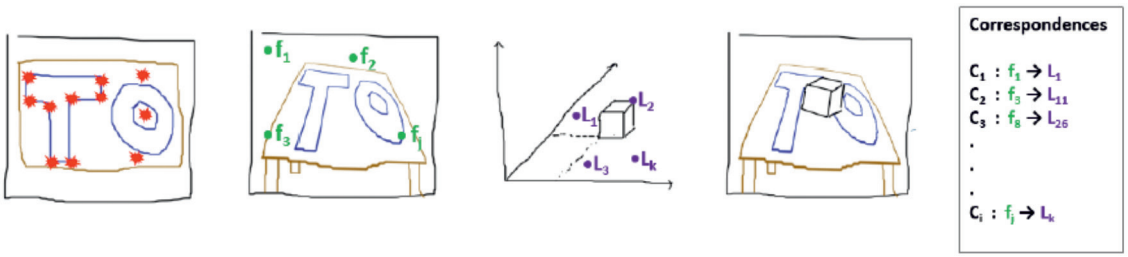

Рис. 4. Процесс визуализации

ModelViewProjection – это Матрица 4×4, содержащая позу отслеживаемого результата. Для каждого отслеживаемого объекта, обнаруженного и отслеживаемого Vuforia, SDK предоставляет нам позу отслеживаемого объекта; поза представляет собой комбинацию положения и ориентации отслеживаемого локального опорного кадра относительно 3D-опорного кадра камеры [4]. Мы можем извлечь эту позу с помощью функции getpos. Поза может быть описана с помощью вращения (матрица 3×3) и перемещения (матрица или вектор 1×3), которые переносят объект из опорной позы в наблюдаемую позу в трехмерном пространстве.

Матрица вращения определяет ориентацию камеры и, следовательно, то, как цель поворачивается относительно плоскости камеры. Перевод матрицы описывает источник камеры. Это положение камеры в виртуальном трехмерном мире. Источник сообщает, где находится цель и как видно цель с камеры. Например, значение <0,0,0> означает, что камера и цель находятся в одном и том же положении, а значение <0,0,5> означает, что цель находится на расстоянии 5 единиц в направлении обзора камеры.

Эта матрица позы определяет, где находится цель по отношению к камере, и позволяет системе визуализировать AR-контент. Однако матрица позы не говорит, как камера расположена относительно маркера или цели. Если это необходимо, то матрица поз должна быть инвертирована.

После выполнения всех операций мы получаем следующий результат, представленный на рис. 5. Vuforia распознаёт с помощью компьютерного зрения загруженный маркер и выводит определенный контент поверх распознанного маркера [5, 6].

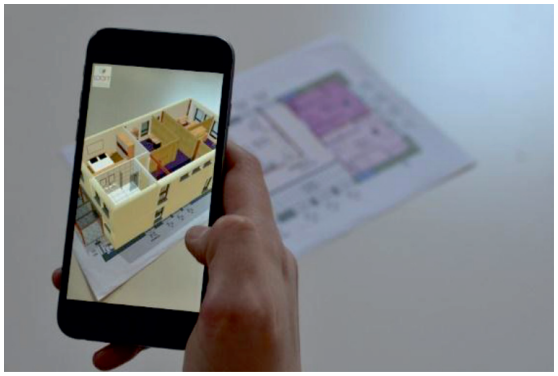

Рис. 5. Результат распознавания маркера и вывод изображения поверх маркера

Заключение

В статье были рассмотрены существующие решения для рендеринга в системе дополненной реальности, показаны низкоуровневые библиотеки графического рендеринга и проведено сравнение программного и аппаратного рендеринга. Затем были представлены некоторые функции OpenGL, которые могут быть использо-

ваны для рендеринга моделей, связанных с AR, и указали, что OpenGL ES 2.0 достаточно для удовлетворения потребностей дополненной реальности на мобильных устройствах. Была показана тесная связь между отслеживанием позы и процессом рендеринга.

Список литературы

1. Wagner D., Schmalstieg D. Experiences with Handheld Augmented Reality. 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality. 13–16 Nov. 2007. P. 1–13.

2. Akenine-Moller T., Haines E. Real-Time Rendering, Fourth Edition. CRC Press. 2018. 158 p.

3. Lee H.-Y., Baek N.-H. OpenGL ES 1.1 Implementation Using OpenGL. The Kips Transactions: parta, 2009. P. 159–168.

4. Vuforia documentation. [Electronic resource]. URL: https://library.vuforia.com/getting-started/overview.html (date of access: 25.12.2020).

5. Шапиро Л., Стокман Д. Компьютерное зрение: учебное пособие / под ред. С.М. Соколова; перевод с англ. А.А. Богуславского. 4-е изд. М.: Лаборатория знаний, 2020. 763 с. [Электронный ресурс]. URL: https://e.lanbook.com/book/135496 (date of access: 25.12.2020).

6. Маслов А.С., Белов Ю.С. Различные виды мишеней и их распознавание в фреймворке дополненной реальности Vuforia // Научное обозрение. Технические науки. 2020. № 1. С. 15–19.