Эффективные алгоритмы распараллеливания процедуры рендеринга при формировании реалистических изображений

Романюк А.Н., доцент, к.т.н.

Винницкий государственный технический университет (Украина)

Для многих современных приложений, таких как виртуальная реальность, научная визуализация, синтез фотографических изображений требуется графическая производительность, превосходящая возможности современных вычислительных машин. Сети рабочих станций или даже суперкомпьютеры оказываются не в состоянии формировать реалистические изображения с необходимой производительностью в реальном масштабе времени [1,2].

Распараллеливание процесса рендеринга является одним из возможных способов решения поставленной задачи на существующей материально-технической базе.

Известные подходы к распараллеливанию процесса рендеринга отражают глобальный подход к проблеме: они генерируют полноэкранное изображение путём композиции сформированных частей сложной сцены, либо «склейкой» полных изображений частей экрана, то есть одновременно формируют различные фрагменты одного изображения [1,2,3].

На этапе конечной визуализации (рендеринг) геометрическая информация трансформируется в элементы изображения. На этом этапе учитываются и отображаются свойства материалов, текстур, освещение, прозрачности. На яркость видимых точек изображения влияют различные параметры и факторы: расстояние к источнику света, расстояние к точке зрения, углы наклона поверхностей относительно источника света и точки зрения, спектральные характеристики источника света, отбивные свойства поверхности, нелинейности в системе отображения и пр.

На практике вполне приемлемые графические изображения получают упрощенными методами, которые не учитывают точно все физические и психофизиологические факторы.

Повышения производительности рендеринга можно достичь за счет:

1) интеграции существующих методов в единый мощный алгоритм, в котором были бы задействованы все методы ускорения производительности;

2) разработки новых, более быстрых алгоритмов рендеринга;

3) распараллеливания процедуры рендеринга;

4) аппаратной реализации алгоритмов нижнего уровня рендеринга.

К числу наиболее важных и распространенных процедур рендеринга относят закраску элементов изображения, к производительности которой предъявляются повышенные требования. Непосредственно процессу закраски предшествует триангуляция полигональной области. Такая процедура позволяет в дальнейшем, с применением линейного интерполирования, легко рассчитать интенсивности света каждого составного пикселе как внутри, так и на ребрах треугольника. В качестве элементарных полигонов треугольники используются чаще других из-за простоты их геометрии и расчетных формул. Актуальным вопросом триангуляции является задача нахождения оптимального соотношения размеров треугольников, на которые разбивается исходный объект, что в конечном итоге обеспечивает сбалансированную загрузку рендеров.

В случаях, когда вычислительной мощности одного процесса рендеринга не достаточно, используют параллельный рендеринг.

Существующие подходы к реализации параллельного рендеринга основаны на:

- композиции изображений, при которой каждый процесс рендеринга генерирует полноэкранное изображение части сложной сцены, а полное изображение получается наложением частичных изображений;

- разделение экрана, когда каждый процесс рендеринга генерирует полное изображение части экрана, а изображение всего экрана получается «склейкой» всех частичных изображений.

Композиционная схема требует сложной аппаратуры композиции, что ограничивает количество возможных атрибутов у пикселей. Кроме того, нагрузка на процессы рендеринга может быть не сбалансирована, что усложняет управление.

Схема разбиения экрана предполагает наличие подпрограммы, которая позволяет распределять поступающие полигоны для закраски различных частей экрана. Кроме того, для адаптивного регулирования нагрузки на процессы рендеринга необходимо проводить динамическое, а не статическое разбиение экранного кадра. Склейка полного изображения не требует сложных специальных операций.

Предлагаются новые подходы распараллеливания процедуры закраски, отличительная особенность которых состоит в реализации вычислительного процесса на уровне составных треугольников, полученных триангуляцией исходной полигональной области. Это позволяет использовать предложенные принципы совместно с ранее рассмотренными в полной независимости от их вычислительного процесса.

При классическом подходе к реализации рендеринга используются два линейных интерполятора, которые формируют координаты точек, составляющих ребра треугольника. По расчетному значению приращения интенсивности по всем ребрам треугольника осуществляется кодовая интерполяция. Для формирования координат внутренних точек треугольника используется строчный линейный интерполятор. Для определения интенсивности внутренних точек треугольника используется соответствующий кодовый интерполятор, который можно совместить с кодовым интерполятором, формирующим ребра треугольника. Таким образом,

классическая реализация рендеринга требует два линейных интерполятора, один кодовый и один строчный.

Один из предлагаемых подходов к реализации параллельной закраски полигона треугольника основан на встречной кодовой интерполяции. Он состоит в том, что значения интенсивности в сканирующей строке треугольника формируют два кодовых интерполятора, которые работают навстречу друг другу из начальной и конечной точек сканирующей строки. Линейная интерполяция ребер треугольника происходит аналогично классическому методу рендеринга. Предложенный принцип позволяет ускорить кодовую интерполяцию сканирующей строки в два раза, а также уменьшить погрешность рендеринга за счет распределения ее между двумя кодовыми интерполяторами. По сравнению с классическим методом дополнительно требуются один кодовый и один строчный интерполяторы.

Часто при рендеринге полигональных областей уже известна хотя бы одна внутренняя точка (точка затравки), ограниченная треугольником, или её легко определить. Информацию о координатах этой точки, а также значения приращения интенсивности можно использовать для волновой заливки треугольника. При таком подходе заданный треугольник предварительно ограничивается контуром определённого цвета. Процесс интерполяции начинается с внутренней точки, активизируя 4^x соседних кодовых интерполятора, работающих в 4^x направлениях. На каждом шаге любой кодовый интерполятор активизирует неактивные по отношению к себе кодовые интерполяторы. При достижении границы треугольника интерполятор переходит в неактивное состояние. Для предлагаемого подхода необходимо определить адрес и интенсивность точки-затравки. Это легко сделать по известному методу "деления пополам" используя микрооперации сложения и сдвига.

Один из возможных путей ускорения процесса рендеринга состоит в независимом вычислении интенсивностей пикселей для четных и нечетных точек сканирующей строки. Первый кодовый интерполятор определяет интенсивность только нечетных точек сканирующей строки треугольника, используя удвоенное значение приращения интенсивности в сканирующей строке ($2*\Delta I$). Интенсивность четных точек сканирующей строки определяет второй кодовый интерполятор. Данный метод дает возможность ускорить кодовую интерполяцию сканирующей строки в два раза. Дополнительные аппаратные затраты у данного подхода, как и у предыдущего метода, составляют один кодовый и один строчный интерполяторы.

Вышеизложенный принцип можно распространить и на большее количество кодовых интерполяторов. Их оптимальное число не превышает четырех, это связано с тем, что величины $2*\Delta I$, $3*\Delta I$, $4*\Delta I$ можно определить при помощи одной операции сдвига и одного сложения. Определение приращения интенсивности большей кратности требует дополнительных вычислительных затрат, что в большинстве случаев не оправданно.

Распараллеливание можно осуществить путем разбивки треугольника с помощью средних линий на составные треугольники. Полученные таким образом четыре составных треугольника подобны исходному треугольнику и равны между собой. При интерполяции полигона треугольника приращение вдоль его стороны величина постоянная, поэтому можно считать, что после интерполирования каждая точка полигона одного составного треугольника будет отличаться от соответствующей ей точки полигона другого составного треугольника лишь на определенную константу. При таком подходе нет необходимости закрашивать всю область, ограниченную исходным треугольником. Достаточно осуществить закраску полигона одного составного треугольника. Используя полигон этого треугольника как шаблон, с помощью микроопераций суммирования формируют полигоны остальных составных треугольников. Эффективно совместить процесс закраски составного треугольника и формирования значений интенсивности для остальных составных треугольников с помощью трех дополнительных сумматоров.

В этом случае процессор рендеринга включает в себя 4 строчных интерполятора по одному на каждый составной треугольник для определения координат пиксела, 2 линейных интерполятора для формирования сторон треугольника, 1 кодовый интерполятор для формирования значений интенсивности пикселей на сторонах и внутри составного треугольника и три сумматора, которые формируют значения интенсивности пикселей для трех составных треугольников.

Использование данного метода позволяет ускорить процесс закраски в четыре раза, так как площадь закрашиваемого составного треугольника в четыре раза меньше площади исходного треугольника, а закраска оставшейся части треугольника происходит параллельно и не требует дополнительных временных затрат. При этом достигается сбалансированная загрузка всех составных рендеров.

Предлагаемые подходы позволяют существенно повысить производительность рендеринга Гуро и Фонга и могут быть использованы в высокопроизводительных средствах реалистических изображений.

Литература

1. В. Штрассер, А. Шиллинг, Г. Книттель Архитектуры высокопроизводительных графических систем// Открытые системы, №5(13), 1995г., стр 53-60.

2. В. Гилой, Г. Расселер Новые стандарты высокореалистичного рендеринга в реальном времени// Открытые системы, №5(13), 1995г., стр 35-44.

3. Molnar S., Eyles J., etc.: PixelFlow - High-Speed Rendering Using Image Composition,SIGGRAPH'92.