УДК 76.022

Гугучкин А.В.
Студент
Харин А.Д.
студент
Научный руководитель: Бригаднов И.А., д. ф.-м. н
Санкт-Петербургский горный университет императрицы Екатерины II
Российская Федерация, Санкт-Петербург

ПЕРСПЕКТИВЫ ПРОГРАММНОГО УСКОРЕНИЯ МЕТОДИК ТРАССИРОВКИ ЛУЧЕЙ В СФЕРЕ ВИЗУАЛИЗАЦИИ 3D ГРАФИКИ

Аннотация

В данной работе будут приведены существующие методы упрощения реализации трассировки лучей с использованием технологий постпроцессинга в пространстве экрана и иерархии ограничивающих объемов, будет отмечен активный к ним интерес в научной среде, а также результат этого интереса в виде значимых улучшений этих технологий.

Решение поставленных в работе задач осуществлялось на основе применения общенаучных методов исследования в рамках сравнительного и логического анализа, а также посредством анализа структуры.

Предложенные в данной работе технологии, как возможный путь развития оптимизации методик трассировки лучей, могут серьезно повлиять на будущее современных медиа проектов, требуя тем самым в производстве меньшие бюджеты, и, открывая тем самым больше творческого простора для авторов.

Ключевые слова: *трассировка лучей, трассировка пути, 3D графика, отражения в пространстве экрана, иерархия ограничивающих объемов, трассировка вокселей конусами;*

Guguchkin A.V.
Student
Kharin A.D.
Student
Academic advisor: ***Brigadnov I.A., Grand PhD in Physics and Mathematics***
St. Petersburg Mining University after Catherine the Great
Russian Federation, Saint-Petersburg

PROSPECTS FOR SOFTWARE ACCELERATION OF RAY TRACING TECHNIQUES IN 3D GRAPHICS VISUALIZATION

Summary

In this paper, the existing methods of simplifying the implementation of ray tracing using postprocessing technologies in the space of the screen and a hierarchy of bounding volumes will be given, will be noted active interest in the scientific community, as well as the result of this interest in the form of significant improvements in these technologies.

The solution of the tasks set in this work was carried out on the basis of the application of general scientific methods of research in the framework of comparative and logical analysis, as well as through the analysis of the structure.

The technologies proposed in this paper, as a possible way to develop the optimization of ray tracing techniques, can seriously affect the future of modern media projects, thus requiring less budget in the production, and, thus, opening more creative space for the authors.

Keywords: *ray tracing, path tracing, 3D graphics, screen space reflections, bunding volume hierarchy, voxel cone tracing;*

Введение

Наиболее важный компонент фотореализма в компьютерной графике обеспечивается физически корректной аппроксимацией переноса света. Помимо прямого освещения от источников света, существует косвенное освещение, создаваемое отражением световых лучей на других поверхностях сцены. В компьютерной графике процесс вычисления освещения поверхности с учетом как прямого, так и косвенного освещения широко известен как глобальное освещение.[1]

Практически все визуальное медиа на сегодняшнее день создается с использованием различных методик трассировки лучей, и каждая крупная компания пытается оптимизировать определенные методики под свои цели и задачи, и чем дальше мы приближаемся к реализму в освещении, тем быстрее замедляется видимый нам прогресс в визуальной составляющей.

Объектом исследования является физически корректное поведение света.

Предметом исследования являются методики трассировки лучей в сфере визуализации трехмерных изображений.

Целью исследования является поиск и анализ методов программной акселерации методик трассировки лучей.

Задачи исследования:

- анализ популярнейших методик трассировки лучей, обзор их преимуществ и недостатков;

- исследование существующих проблем в области трассировки лучей;
- исследование алгоритмов оптимизации методик трассировки лучей;
- анализ возможных методов оптимизации методик трассировки лучей в будущем;

Описанные в работе методы программного ускорения трассировки лучей демонстрируют улучшение производительности с небольшой потерей качества восприятия по сравнению с полным решением трассировки лучей.

Принцип работы обратной трассировки лучей

Трассировка лучей представляет собой совокупность математических методов, разработанных для более точного моделирования распространения света из реального мира в трехмерной графике. В своей работе, опубликованной в 1980 году и называющейся "Улучшенная модель освещения для затененного отображения", был введен термин "обратная трассировка лучей", а также описаны основные принципы этого метода в трехмерной визуализации.[2]

Обратная трассировка лучей - это модифицированный метод броска лучей, в котором трассировка начинается из камеры и не заканчивается при столкновении луча с объектом. Вместо этого, генерируются новые лучи из точки соприкосновения, и их направление определяется материалом объекта. Этот подход делает метод рекурсивным, что влияет на его производительность, но позволяет корректно отображать отражения, преломления и тени.[4]

При генерации лучей алгоритм создает луч, исходящий из камеры и направленный к каждому пикселю на плоскости. Цвет пикселя определяется в соответствии с трехмерным объектом, который пересекается лучом. Эти

процессы составляют один образец, и для рендеринга сцены выполняется несколько образцов. Схему работы обратной трассировки лучей можно увидеть на рисунке 1.

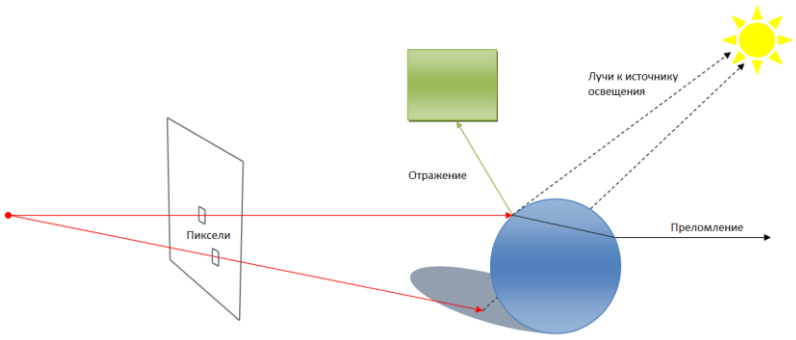

Рисунок 1. Схема работы обратной трассировки лучей[1]

Реализация обратной трассировки лучей включает многократный поиск точек пересечения сгенерированных лучей и поверхностей, определение нормалей этих поверхностей в точках пересечения и вычисление преломленных и отраженных лучей. Поиск точек пересечения является наиболее ресурсоемким процессом.[5]

Принцип работы трассировки пути

Принцип работы трассировки пути был впервые описан в статье Джеймса Каджия "Уравнение рендеринга" в 1986 году. В этой статье был введен термин "трассировка пути" и описаны основные принципы этого метода.

Трассировка пути является улучшенным методом распределенной трассировки лучей, представленной Робертом Куком в его работе "Распределенная трассировка лучей" в 1984 году. В этой работе была

[1] https://mavink.com/explore/Ai-GPU-Structure

предложена концепция стохастического распределения лучей во времени и пространстве.[6]

Интегрирование Монте-Карло является техникой численного интегрирования, которая используется в трассировке пути для вычисления интеграла$f(x_n)$ путем аппроксимации конечного числа случайных выборок в области интегрирования. [7]

Обобщённая форма выглядит следующим образом (1):

$$F_N \approx \frac{1}{N}\sum_{n=0}^{N}\frac{f(x_n)}{\rho(x_n)}, \quad (1)$$

Схему работы трассировки пути можно увидеть на рисунке 2.

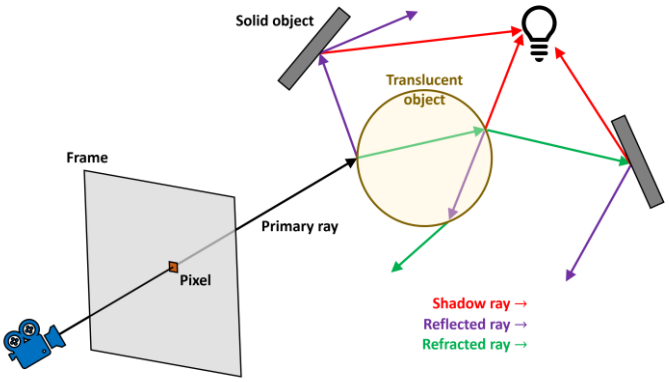

Рисунок 2. Схема работы трассировки пути [2]

Одной из ключевых преимуществ трассировки пути по сравнению с другими методами трассировки лучей является возможность достижения высокого уровня производительности и воспроизведения сложных оптических эффектов, таких как глобальная окклюзия, каустика, глубина резкости и размытие, благодаря использованию большего числа выборок на пиксель, чем один образец.[8,9]

[2] https://www.pvsm.ru/razrabotka-igr/340429

Основные нерешенные проблемы обратной трассировки лучей

Однако трассировка пути также имеет нерешенные проблемы. Когда в сцене присутствуют несколько источников света, затенение каждого объекта отдельно становится сложной задачей из-за большого количества лучей, требующихся для расчетов, и когерентности [10]. При трассировке траектории отраженных лучей их направление напрямую зависит от шероховатости поверхности материала: лучи от гладких поверхностей отражаются практически в одном направлении, тогда как лучи от шероховатых поверхностей отражаются в разных направлениях, что снижает когерентность. Это приводит к проблеме поиска, которая требует значительного времени для выполнения [3,11]. Также требуется выбор подмножества источников света или объектов.

Существующие структуры ускорения, основанные на статических сценах, плохо справляются с рендерингом в реальном времени [12]. При работе с динамическими сценами в реальном времени эти ускоряющие структуры должны постоянно перестраиваться, что требует значительных вычислительных ресурсов. В индустрии рендеринга возникает противоречие между пользовательским опытом и возможностями рендеринга. Снижение вычислительной нагрузки и оптимизированные структуры ускорения становятся необходимостью.

Основные нерешенные проблемы трассировки пути

Трассировка пути также имеет свои нерешенные проблемы. В ее основе лежит физически правильное моделирование переноса света через пути, соединяющие камеру и источники света. При использовании трассировки пути для визуализации сцены необходимо рассчитать несколько преломлений и отражений, чтобы получить конечное значение

пикселя. Из-за усреднения пространства путей, начальный шум становится неотъемлемой частью моделирования переноса света [14,15].

Трассировка пути может правильно рассчитать цвет пикселя с бесконечным временем, но из-за ограниченных вычислительных ресурсов и временных ограничений это не возможно; трассировка пути останавливает отражение лучей при достижении пороговой глубины или с помощью метода русской рулетки. В результате на экране появляется много шума [16]. Для снижения уровня шума до приемлемого уровня требуется значительное количество выборок, например, не менее 5000 выборок на пиксель, что требует больших вычислительных ресурсов [17].

На рисунке 3 показаны примеры визуализации с разным количеством выборок при использовании трассировки пути.

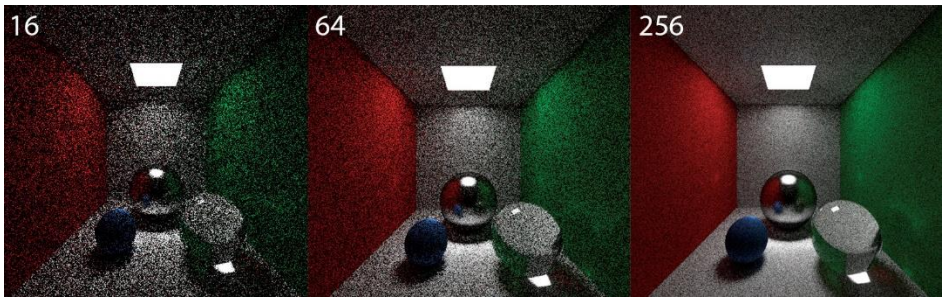

Рисунок 3. (слева направо) Визуализация трассировкой пути 16 выборок на пиксель/64 выборки на пиксель/256 выборок на пиксель [3]

Основные принципы работы технологии отражения в пространстве экрана (SSR)

Избавиться как от шума, так и от постоянной необходимости рассчитывать точки столкновения световых лучей с полигонами может помочь постпроцессинг.

[3] https://ru.stackoverflow.com/questions/1210358/%D0%94%D0%B2%D1%83%D0%BD%D0%B0%D0%BF%D1%80%D0%B0%D0%B2%D0%BB%D0%B5%D0%BD%D0%BD%D0%B0%D1%8F-%D1%82%D1%80%D0%B0%D1%81%D1%81%D0%B8%D1%80%D0%BE%D0%B2%D0%BA%D0%B0-%D0%BF%D1%83%D1%82%D0%B5%D0%B9-bidirectional-path-tracing

Такие эффекты как освещение, глобальное затенение, отражения можно отделить от геометрии и реализовать эти эффекты как особый вид постпроцессинга в пространстве экрана (Screen Space).

Принцип подобного подхода заключается в идее отложенного освещения, состоящего из двух проходов: в первом проходе, геометрическом, рисуется вся сцена и различная информация сохраняется в набор текстур, называемых G–буффером, который, обычно, в себя включает три текстуры: карту цвета и карту нормалей – текстуры, о которых говорилось в первом разделе про физически корректный шейдинг, а также карта глубины, которая в свою очередь дает нам экранные координаты, о том, насколько «далеко» от камеры пиксель.

Карту цвета, нормалей, глубины можно увидеть на рисунке 4.

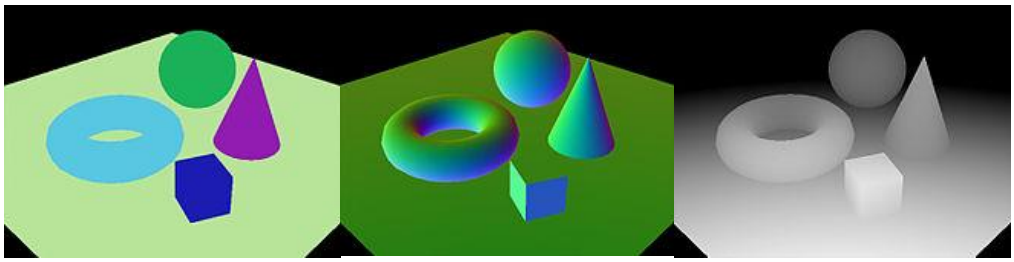

Рисунок 4. Карта цвета[4] Карта нормалей Карта глубины [5]

Сохранённая в G–буфере графическая информация как раз и может использоваться для реализации постпроцессинга в пространстве экрана, один из методов которого именуется отражения в пространстве экрана (SSR).[18]

Для реализации отражения данная технология использует только те ресурсы, которые видит виртуальная камера. Если какой–то объект хотя бы частично остаётся за кадром, то и его отражение не будет прорисовываться до конца, также источник света за границей экрана не будет влиять на

[4] https://habr.com/ru/post/244367/
[5] https://habr.com/ru/post/244367/

отражение. Такие ограничения разработчики маскируют, замыливая отражающую поверхность или растягивая края текстуры.

Используя SSR надо было следить за тем, чтобы между отражающей поверхностью и камерой ничего не находилось, так как это корректно отразить было невозможно, что видно на рисунке 5.[21]

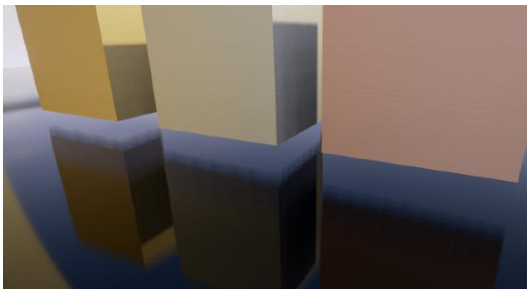

Рисунок 5. Артефакты SSR – не видимые в камере нижние поверхности кубов, не отражаются на плоскости [6]

Гибридная технология отражения в пространстве экрана и чистой трассировки лучей

Современные методы, основанные на растеризации, такие как отражения в пространстве экрана, часто страдают от артефактов на вне экранном контенте. Кроме того, трассировка лучей, вероятно, является наиболее эффективным способом обработки многократных отражений на поверхностях произвольной формы.[4]

Описанный в данном разделе алгоритм из статьи «Динамические отражения в реальном времени для реалистичной визуализации 3D-сцен», спроектированный и разработанный Даниэлем Валенте де Маседо и Марией Андрейя Формико Родригес сочетает растеризацию, технику SSR с чистым алгоритмом трассировки лучей. Схематическое представление подобной технологии можно увидеть на рисунке 6.

[6] https://mobillegends.net/planar-reflection-on-mobile-bug-ue4-answerhub

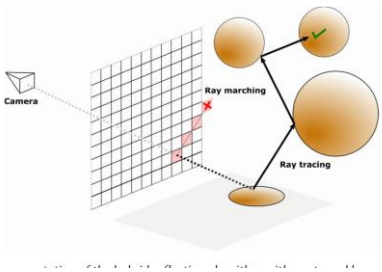

Рисунок 6. Схематическое представление гибридного алгоритма отражения с трассировкой лучей отскоков. Трассировка лучей ищет точку пересечения для отражения в пространстве экрана. Если она не найдена, тот же пиксель альтернативно отображается с помощью трассировки лучей.[24]

Результаты демонстрируют значительное улучшение производительности при очень незначительной потере качества гибридного алгоритма по сравнению с полным решением трассировки лучей. Кроме того, подобный метод хорошо масштабируется для реалистичных динамических сцен с трехмерными объектами.[23]

Отражения моделируются с помощью аппроксимации из комбинации SSR с использованием предопределенной ограничивающей рамки, которая содержит буферы, связанные с ее гранями, для представления сцены. Это используется в качестве резервной функции для исправления частей сцены, где метод добавления отражений в пространстве экрана не сработал.[24]

Итоги визуализации разными методам можно увидеть на рисунке 7. Разбивку по времени визуализации итогового изображения различными методами можно увидеть на рисунке 8.

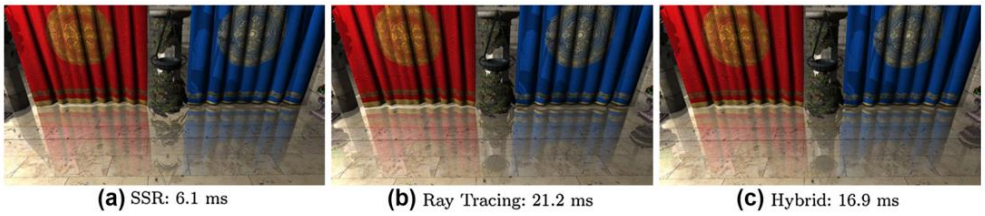

Рисунок 7. Итоговые изображения и соответствующее время их обработки[23]

Lighting process	SSR (ms)	Ray tracing (ms)	Hybrid (ms)
G-buffer	0.9	0.9	0.9
Shadow	4.5	4.5	4.5
SSR	0.3	0.0	0.3
Ray tracing	0.0	15.1	10.8
Light (one point light)	0.2	0.2	0.2
Final result	6.1	21.2	16.9

Рисунок 8. Разбивка по времени визуализации итогового изображения различными методами[23]

Впоследствии, к комбинированному изображению применяется фильтр по гауссу, чтобы смягчить часть артефактов, созданных технологией SSR, особенно в областях перехода между SSR и в пространстве объекта, как показано на рисунке 9.

Рисунок 9. Слева гибридное изображение и визуальные артефакты (выделены кружком), которые могут появиться из-за процесса слияния SSR и трассировки лучей. Справа артефакты были уменьшены путем применения размытия Гаусса[23]

Что касается генерации теней, то перед генерацией отражений гибридный алгоритм использует кубическую карту теней.[25] Наконец, все освещения рассчитываются и объединяются с картами отражений и теней, в результате чего получается окончательная композиция сцены.

Влияние разрешения на процесс рендеринга технологии SSR, трассировки лучей и гибридных алгоритмов можно проследить на рисунке 10.

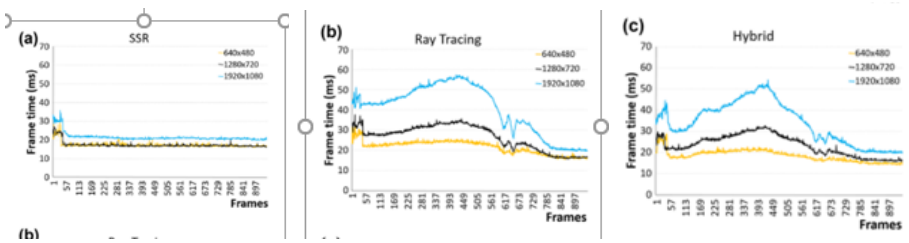

Рисунок 10. Влияние разрешения на процесс рендеринга в алгоритмах SSR, трассировки лучей и гибридном алгоритме[23]

Результаты демонстрируют улучшение производительности при использовании гибридного алгоритма, с небольшой потерей качества восприятия по сравнению с полным решением трассировки лучей.

Основные принципы работы технологии иерархии ограничивающих объемов (BVH)

При генерации реалистичной графики каждый луч должен пересекаться с каждым полигоном традиционным методом рендеринга, что будет потреблять много вычислительных ресурсов. Поэтому оптимизация алгоритма трассировки лучей в основном сосредоточена на том, как уменьшить пересечение света и полигонов в сцене для повышения эффективности алгоритма.[24]

Иерархии ограничивающих объемов (BVH) являются первыми ускоряющими структурами пространственных данных для сокращения вычислений пересечений.[3,25]

BVH есть ничто иное, как упорядоченная карта положения объектов в сцене, в виде дерева.

Существует и два типа BVH:

- Статические - которые перестраиваются после того, как мы изменили какой–либо объект в сцене, однако, после того как они были построены, они ускоряют время рендеринга сцены.
- Динамические - позволяют обновлять объекты индивидуально, таким образом, что при перестроении BVH время на это намного меньше, но, в свою очередь, увеличивается время последующей визуализации.

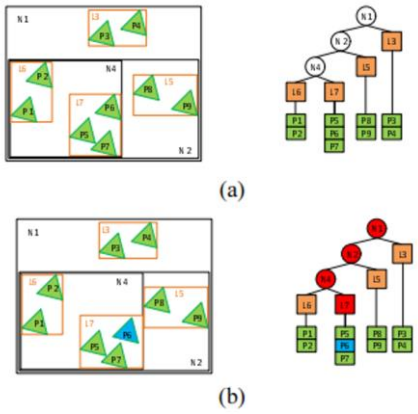

Рисунок 11. Обновление дерева BVH: (a) дерево BVH текущего кадра и (b) обновленное дерево BVH следующего кадра[25]

Вместо того, чтобы проверять пересечение луча с каждым треугольником в сцене по отдельности, система трассировки лучей будет проверять, пересекается ли граничный объем узла с лучом или нет.

Если же блок пересечен не был, то и никакой работы над этим блоком BVH производить не требуется. Если пересечение было установлено, алгоритм получает набор меньших блоков для пересекаемого объекта. Дополнительные BVH пересечения происходят до тех пор, пока алгоритм не получит фактический список полигонов, в которые попадает луч.

Пример реализации поиска пересечения с помощью BVH можно проследить на рисунке 12.

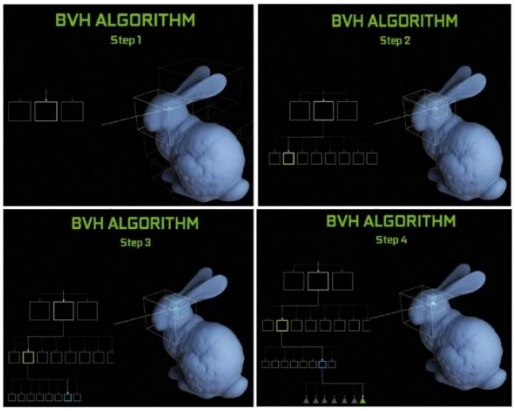

Рисунок 12. Бросок луча на кролика, чтобы выяснить, в какой треугольник он попадает. Требуется всего 4 двоичные проверки (да/нет)[7]

Это недетерминированная операция, что означает, что невозможно точно сказать, насколько много лучей может быть просчитано в секунду, поскольку это зависит от сложности сцены и структуры BVH.

Оптимизация иерархии ограничивающих объемов по средствам алгоритма FBVH

Алгоритм оптимизации трассировки лучей FBVH, предложенный в работе «Алгоритм трассировки лучей в реальном времени для динамической сцены» двумя китайскими учеными Тяньхань Гао и Инь Ли, работает, уменьшая количество пересечений световых и объектных поверхностей в сцене, алгоритм вводит иерархическую структуру ограничивающего поля для эффективной организации геометрических поверхностей. Достигается цель сокращения вычислительной избыточности и повышения эффективности алгоритма.[3]

Во время построения стандартной BVH устанавливается порог для размера отрисованной сцены. Объекты, размер которых превышает этот порог, определяются в сцене как сложные объекты. Напротив, объект, размер которого меньше или равен этому порогу, определяется как простой

[7] https://www.hardwaretimes.com/what-is-ray-tracing-and-how-does-it-work-are-nvidias-rtx-gpus-worth-it/?share=twitter

объект. Граничная область строится отдельно для простых и сложных объектов с помощью различных методов построения граничной области сцены.

Для сложных объектов используется метод построения внутреннего ограничительного поля, для простых – в свою очередь внешнего. Применение этого метода позволяет уменьшить сложность пересечения лучей и избыточность, а также повысить эффективность алгоритма. .

Сначала используется метод построения сверху вниз для создания узлов BVH для каждого ограничивающего поля, в процессе разбиения в качестве базовой оси выбирается самая длинная ось всех примитивных граничных окон. Затем, согласно алгоритму Surface Area Heuristic (SAH), на котором строится стратегия разбиения FBVH, выбирается местоположение раздела и происходит выбор двух плоскостей в качестве левого и правого поддерева.

Наконец, для быстрого построения древовидной структуры BVH используется метод построения снизу вверх.

Когда информация о положении объекта меняется в динмической сцене, FBVH способен быстро обновить недействительную структуру и повысить эффективность рендеринга. FBVH учитывает структурную инвариантность простых объектов в реальных времени. Когда сцена динамически меняется, логическая структура исходного ограничительного поля простого одиночного объекта не меняется. Затем простая структура данных быстро обновляется методом построения сверху вниз. После этого ограничительная область каждого объекта в сцене принимается за узел, и дерево BVH строится по принципу «снизу вверх».

Такой подход позволяет сохранить логическую структуру ограничивающей рамки одиночного объекта от изменений при динамическом изменении сцены. Все отдельные объекты в сцене рассматриваются как объекты низкого уровня, поэтому мы можем быстро

обновлять недействительные структуры, используя методы построения «сверху вниз» и «снизу вверх».

Заключение

В работе были проанализированы основные проблемы современных методов трассировки лучей, а именно ресурсозатратность просчета столкновений световых лучей с геометрией в сцене и зашумление картинки, по причине ограничения количества выборки образцов на кадр.

В работе был показан подход гибридного рендера с использованием постпроцессинга в пространстве экрана, что существенно снижает время визуализации отражений при трассировке, что делает этот метод применимым для не сложных сцен.

А улучшенная структура иерархии ограничивающих объемов – FBVH, способная быстро обновить недействительную структуру позволяет повысить эффективность рендеринга в особо динамичных сценах рендера в реальном времени или во время работы в окне рендер программы

Все описанные в работе методики оптимизации трассировки лучей демонстрируют улучшение производительности с небольшой потерей качества восприятия по сравнению с полным решением трассировки лучей.

Использованные источники:

1. McGuire M. et al. Real-time global illumination using precomputed light field probes //Proceedings of the 21st ACM SIGGRAPH symposium on interactive 3D graphics and games. – 2017. – С. 1-11.
2. Whitted T. An improved illumination model for shaded display //Proceedings of the 6th annual conference on Computer graphics and interactive techniques. – 1979. – С. 14.
3. Gao T., Li Y. Real-Time Ray Tracing Algorithm for Dynamic Scene //Innovative Mobile and Internet Services in Ubiquitous Computing: Proceedings

of the 13th International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS-2019). – Springer International Publishing, 2020. – С. 125-131.

4. Liu E. et al. Cinematic rendering in UE4 with real-time ray tracing and denoising //Ray Tracing Gems: High-Quality and Real-Time Rendering with DXR and Other APIs. – 2019. – С. 289-319.

5. Corso A. D. et al. Interactive stable ray tracing //Proceedings of High Performance Graphics. – 2017. – С. 1-10.

6. Cook R. L., Porter T., Carpenter L. Distributed ray tracing //Proceedings of the 11th annual conference on Computer graphics and interactive techniques. – 1984. – С. 137-145.

7. Lin W. et al. A detail preserving neural network model for Monte Carlo denoising //Computational visual media. – 2020. – Т. 6. – С. 157-168.

8. Zeltner T., Georgiev I., Jakob W. Specular manifold sampling for rendering high-frequency caustics and glints //ACM Transactions on Graphics (TOG). – 2020. – Т. 39. – №. 4. – С. 149: 1-149: 15.

9. Keller A. et al. The iray light transport simulation and rendering system //ACM SIGGRAPH 2017 Talks. – 2017. – С. 1-2.

10. Barringer R., Andersson M., Akenine- Möller T. Ray Accelerator: Efficient and Flexible Ray Tracing on a Heterogeneous Architecture //Computer Graphics Forum. – 2017. – Т. 36. – №. 8. – С. 166-177.

11. Koskela M. et al. Blockwise multi-order feature regression for real-time path-tracing reconstruction //ACM Transactions on Graphics (TOG). – 2019. – Т. 38. – №. 5. – С. 1-14.

12. Wang X., Zhang R. Rendering transparent objects with caustics using real-time ray tracing //Computers & Graphics. – 2021. – Т. 96. – С. 36-47.

13. Moreau P., Pharr M., Clarberg P. Dynamic Many-Light Sampling for Real-Time Ray Tracing //High Performance Graphics (Short Papers). – 2019. – С. 21-26.

14. Fascione L. et al. Path tracing in production: part 1: modern path tracing //ACM SIGGRAPH 2019 Courses. – 2019. – С. 1-113.
15. Fascione L. et al. Path tracing in production-part 2: making movies //ACM SIGGRAPH 2017 Courses. – 2017. – С. 1-32.
16. Pediredla A. et al. Path tracing estimators for refractive radiative transfer //ACM Transactions on Graphics (TOG). – 2020. – Т. 39. – №. 6. – С. 1-15.
17. Deng H. et al. A practical path guiding method for participating media //Computational Visual Media. – 2020. – Т. 6. – С. 37-51.
18. Hofmann N. et al. Hierarchical multi-layer screen-space ray tracing //Proceedings of High Performance Graphics. – 2017. – С. 1-10.
19. Yalçıner B., Sahillioğlu Y. Voxel transformation: scalable scene geometry discretization for global illumination //Journal of Real-Time Image Processing. – 2020. – Т. 17. – С. 1585-1596.
20. Lambru C. et al. Comparative Analysis of Real-Time Global Illumination Techniques in Current Game Engines //IEEE Access. – 2021. – Т. 9. – С. 125158-125183.
21. Widmer S. et al. An adaptive acceleration structure for screen-space ray tracing //Proceedings of the 7th Conference on High-Performance Graphics. – 2015. – С. 67-76.
22. Andrade P. et al. Ray-Traced Reflections in Real-Time Using Heuristic Based Hybrid Rendering //2014 Brazilian Symposium on Computer Games and Digital Entertainment. – IEEE, 2014. – С. 240-248.
23. de Macedo D. V., Rodrigues M. A. F. Real-time dynamic reflections for realistic rendering of 3D scenes //The Visual Computer. – 2018. – Т. 34. – С. 337-346.
24. Macedo D. V. D., Serpa Y. R., Rodrigues M. A. F. Fast and realistic reflections using screen space and GPU ray tracing—a case study on rigid and deformable body simulations //Computers in Entertainment (CIE). – 2018. – Т. 16. – №. 4. – С. 1-18.

25. Wyman C., Dai Z. Imperfect voxelized shadow volumes //Proceedings of the 5th High-Performance Graphics Conference. – 2013. – С. 45-52.