

УДК 004.92

Исследование особенностей рендеринга деревьев в Unreal Engine 5

Самарин Савва Денисович

Витюков Федор Андреевич (научный руководитель)

МГТУ им. Н.Э. Баумана, Москва, Россия

Проведено исследование проблем, возникающих при рендеринге деревьев в трехмерном движке Unreal Engine 5. Рассмотрены такие аспекты, как избыточная отрисовка объектов, оптимизация анимации, устранение световых артефактов, а также современные методы и технологии, способствующие решению связанных с рендерингом задач.

Ключевые слова: 3D-моделирование, альфа-сортировка, проблема перерисовки, повышение производительности, порядко-независимая прозрачность

Введение. Создание элементов растительности в компьютерной графике представляет собой сложную задачу, обусловленную необходимостью обработки больших массивов объектов, часто сконцентрированных в ограниченном пространстве сцены. Это приводит к возникновению оптимизационных проблем и затрудняет достижение высокого визуального качества. Настоящее исследование посвящено анализу современных методов и особенностей рендеринга объектов растительности в Unreal Engine 5. Основной акцент сделан на изучении обработки полупрозрачных текстур, решении задач избыточной отрисовки объектов в едином экранном пространстве, исследовании особенностей анимации деревьев и устранении световых артефактов в итоговом рендере.

Проблема альфа- сортировки. Для рендеринга кадра применяется множество вычислительных методов. Важными для исследования в рамках данной работы являются следующие.

Z-буферизация. Данный метод основан на использовании двумерного массива буфера глубины (Z-буфера), в котором для каждого пикселя экрана фиксируется расстояние до ближайшего объекта. Если фрагмент сцены находится дальше уже зарегистрированного ближайшего объекта, то он не обрабатывается. Основным ограничением метода является невозможность корректной обработки полупрозрачных и пересекающихся объектов.

Z-сортировка (алгоритм художника). Этот метод предполагает рендеринг объектов в порядке от наиболее удаленных к наиболее близким к наблюдателю. Несмотря на свою концептуальную простоту, метод характеризуется высокой вычислительной сложностью, что ограничивает его применимость в сценах с большим количеством объектов [1].

При рендеринге сложных сцен, включающих элементы растительности, возникают следующие ключевые проблемы.

Z-конфликты. Данная проблема связана с пересечением объектов, при котором алгоритм рендеринга не может однозначно определить, какой

из объектов должен быть отображен поверх другого. Это приводит к визуальным артефактам, таким как мерцание изображений и некорректное отображение объектов.

Ошибки буфера глубины. Проблема проявляется в неверном определении порядка отрисовки объектов, особенно при их пересечении. Ошибки чаще всего возникают в сценах с большим количеством пересекающихся элементов, таких как листья деревьев, что затрудняет корректное определение последовательности рендеринга.

Неверный порядок блендинга. Для корректного отображения полупрозрачных объектов необходимо их рендерить в порядке от наиболее удаленных к наиболее близким к наблюдателю. Нарушение этого порядка приводит к искажению визуального результата.

Проблемы производительности. Высокая вычислительная сложность рендеринга большого числа объектов, характерная для сцен с растительностью, существенно снижает производительность.

Специфика проблем. Указанные проблемы не ограничиваются рендерингом объектов растительности, однако они проявляются наиболее выражено в таких сценах из-за необходимости обработки значительного количества элементов, таких как листья и ветви.

В Unreal Engine 5 реализована экспериментальная функция порядко-независимой прозрачности (Order-Independent Transparency, OIT), которая представляет собой усовершенствованный подход к рендерингу полупрозрачных объектов [2]. Данный алгоритм имеет сходство с методом Z-буферизации, но отличается тем, что обрабатывает каждый пиксель экрана независимо от порядка объектов. Алгоритм включает следующие этапы.

1. Рендеринг непрозрачных объектов. На первом этапе отрисовываются все непрозрачные объекты. Объекты, расположенные дальше от камеры, отбрасываются, а для отрисованных объектов сохраняются их цветовые характеристики и значения глубины.

2. Обработка полупрозрачных объектов. Для каждого пикселя экрана, связанного с полупрозрачными объектами, фиксируются цвет объекта, уровень глубины и степень непрозрачности.

3. Блендинг и формирование итогового изображения. На заключительном этапе выполняется смешивание (блендинг) цветов и параметров всех объектов для каждого пикселя. Процесс блендинга продолжается последовательно между парами объектов, формируя промежуточные результаты, пока не будут смешаны все объекты, после чего создается окончательное изображение.

Процесс блендинга в алгоритме порядко-независимой прозрачности (OIT) описывается выражениями

$$\begin{aligned} final.rgb = & fragment.rgb * fragment.\alpha + \\ & + (1 - fragment.\alpha) * previous.rgb; \end{aligned}$$

$$final.\alpha = fragment.\alpha + (1 - fragment.\alpha) * previous.\alpha,$$

где объект *final* представляет результат блендинга двух объектов на текущем этапе; *previous* — предыдущий результат; *fragment* — текущий объект, подлежащий смешиванию; параметры *rgb* и α отвечают за цвет и прозрачность объекта соответственно.

Алгоритм порядка-независимой прозрачности (OIT) эффективно решает перечисленные проблемы рендеринга благодаря строгому порядку обработки объектов на попиксельной основе. Однако данный метод находится в стадии активной доработки, и его основным недостатком является высокая вычислительная стоимость, особенно при изменении положения камеры или анимации объектов, таких как деревья, когда глубина объектов постоянно изменяется.

Помимо OIT, существуют альтернативные методы, направленные на решение отдельных аспектов рендеринга.

Устранение Z-конфликтов. Для предотвращения конфликтов глубины объектам можно назначать приоритет сортировки (Sort Priority) или применять смещение глубины (Depth Bias), что позволяет корректно определять порядок их отрисовки.

Обработка пересекающихся объектов. Для решения проблем, связанных с пересечением объектов, используется наклонная шкала смещения глубины (Slope Scale Depth Bias), которая улучшает точность определения порядка рендеринга.

Порядок блендинга полупрозрачных объектов. В Unreal Engine 5 проблемы с порядком смешивания полупрозрачных объектов, возникающие при их пересечении, без применения OIT решаются преимущественно вручную путем настройки параметров объектов и их сортировки.

Оптимизация производительности. Для повышения производительности применяется метод предварительного обхода глубины (Z-prepass). Этот подход предусматривает генерацию упрощенных моделей перед финальным рендерингом, что позволяет исключить из обработки объекты, перекрываемые непрозрачными элементами.

Проблема перерисовки (overdraw). Большое количество элементов в сцене часто приводит к снижению производительности, ключевым фактором которого является избыточная отрисовка объектов (Overdraw). Для решения проблемы перерисовки применяются следующие современные подходы.

Foliage Instancing. Данный метод позволяет объединять объекты одного типа, что существенно сокращает количество вызовов отрисовки и повышает эффективность рендеринга.

Настройка уровней детализации (Level of Detail, LOD). Использование LOD обеспечивает снижение детализации объектов, расположенных на значительном удалении от наблюдателя, что уменьшает общее количество элементов, подлежащих рендерингу [3].

Методы отсечения объектов. Применение технологий отсечения, таких как GPU Culling, Instance Culling и Nanite Foliage Culling, позволяет исключить из обработки объекты, находящиеся за пределами поля зрения наблюдателя [4].

Для анализа участков с высокой интенсивностью вызовов отрисовки в Unreal Engine 5 предусмотрен режим визуализации сложности шейдеров (Shader Complexity Mode), который помогает выявить проблемные зоны и оптимизировать производительность [5].

Проблемы оптимизации анимационных циклов. Анимация растительности в компьютерной графике необходима для моделирования взаимодействия объектов с окружающей средой, включая симуляцию воздействия ветра, реакции на контакт с игроком и другие динамические процессы. В Unreal Engine 5 реализовано несколько методов анимации растительности, наиболее значимыми из которых являются:

- вертексная анимация — позволяет симулировать движение деревьев под действием ветра с указанием направления, типа воздействия и силы;
- физическая анимация — позволяет симулировать взаимодействие с объектами, характеризуется высокой ресурсоемкостью и, как правило, применяется для ключевых или близко расположенных объектов;
- скелетная анимация — обладает возможностью точной настройки анимации деревьев;
- анимация глобального ветра — универсальная настройка поведения растительности на всей обрабатываемой сцене.

Выбор метода анимации определяется задачами проекта. Вертексная и глобальная анимации обладают наибольшей производительностью, но их возможности ограничены примитивным визуальным представлением без качественной настройки и отсутствием поддержки сложных взаимодействий с окружением. Эти методы наиболее эффективны для объектов, расположенных на значительном удалении от наблюдателя, где взаимодействие минимально или отсутствует. Физическая и скелетная анимации, несмотря на высокую вычислительную стоимость, обеспечивают более реалистичное и детализированное поведение и применяются для ключевых объектов или элементов, находящихся вблизи игрока.

Для повышения производительности анимации растительности в Unreal Engine 5 применяются следующие методы оптимизации:

- упрощение детализации анимации для удаленных объектов. Оптимизация достигается путем снижения параметров интенсивности ветра и частоты анимационных циклов, что уменьшает вычислительную нагрузку при рендеринге объектов, находящихся на значительном расстоянии от наблюдателя;
- инстанцирование анимации. Использование инстанцирования для объектов одного типа, таких как идентичные или удаленные деревья, позволяет оптимизировать обработку анимации за счет объединения вычислений для однотипных элементов.

Световые артефакты. Основной причиной возникновения световых артефактов при рендеринге деревьев в Unreal Engine 5 являются проблемы обработки полупрозрачных объектов, а также высокая плотность движущихся элементов в сцене. Эти факторы приводят к следующим ошибкам:

Мерцанию теней. Данная проблема аналогична Z-конфликтам при рендеринге объектов и возникает из-за отсутствия четкого порядка обработки пересекающихся элементов, что приводит к нестабильному отображению теней.

Возникновению просветов. Ошибка связана с неполным перекрытием непрозрачными объектами лучей света, вызванным некорректным определением порядка рендеринга.

Возникновению шумов и зернистости. Эти артефакты обусловлены техническими ограничениями технологий освещения, таких как Lumen, при обработке сцен с большим количеством движущихся объектов [6].

Решение первых двух проблем связано с устранением ошибок в порядке рендеринга пересекающихся объектов, в частности, посредством настройки параметров смещения глубины (Depth Bias). Третья проблема обусловлена ограничениями технологии Lumen, и ее устранение требует снижения вычислительной нагрузки, например, за счет уменьшения количества объектов или интенсивности их анимации.

Заключение. В рамках исследования были проанализированы актуальные проблемы, связанные с рендерингом объектов растительности в Unreal Engine 5. Рассмотрены основные аспекты, включая обработку полупрозрачных объектов, оптимизацию производительности и устранение световых артефактов. Также были описаны и систематизированы альтернативные методы и подходы, направленные на повышение эффективности рендеринга и минимизацию ошибок при обработке объектов.

Литература

- [1] Польский С.В., Мышенков Е.В., Летин А.С. *Алгоритм художника. Компьютерная графика*. Москва, Московский государственный университет леса (МГУЛ), 2008, с. 15–17.
- [2] Using Transparency in Materials. URL: <https://dev.epicgames.com/documentation/en-us/unreal-engine/using-transparency-in-unreal-engine-materials> (accessed 06.05.2025).
- [3] Creating and Using LODs. URL: <https://dev.epicgames.com/documentation/en-us/unreal-engine/creating-and-using-lods-in-unreal-engine> (accessed 06.05.2025).
- [4] Visibility and Occlusion Culling. URL: <https://dev.epicgames.com/documentation/en-us/unreal-engine/visibility-and-occlusion-culling-in-unreal-engine> (accessed 06.05.2025).
- [5] Viewport Modes. URL: <https://dev.epicgames.com/documentation/en-us/unreal-engine/viewport-modes-in-unreal-engine> (accessed 06.05.2025).
- [6] Lumen Global Illumination and Reflections. URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/lumen-global-illumination-and-reflections-in-unreal-engine?application_version=5.0 (accessed 08.05.2025).

Tree rendering features research in Unreal Engine 5

Samarin Savva Denisovich

Vityukov Fedor Andreevich (*scientific advisor*)

BMSTU, Moscow, Russia

This research examines challenges in rendering trees within Unreal Engine 5, focusing on issues such as translucent object processing, overdraw management, animation optimization, and light artifact correction. It also explores contemporary methods and technologies for addressing rendering-related problems.

Keywords: 3D modeling, alpha sorting, overdraw, performance improvement, order-independent transparency