Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Робототехника и комплексная автоматизация (РК)

КАФЕДРА Системы автоматизированного проектирования (РК6)

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ

НА ТЕМУ:

«Методы оптимизации рендеринга деревьев в Unreal Engine 5»

Студент РК6-85Б _____ **Самарин С.Д.**
(Подпись, дата) И.О. Фамилия

Руководитель _____ **Витюков Ф.А.**
(Подпись, дата) И.О. Фамилия

2025 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ
Заведующий кафедрой РК6
А.П. Карпенко

«_____» _____ 2023 г.

З А Д А Н И Е
на выполнение научно-исследовательской работы

по теме: Методы оптимизации рендеринга деревьев в Unreal Engine 5

Студент группы РК6-85Б

Самарин Савва Денисович
(Фамилия, имя, отчество)

Направленность НИР (учебная, исследовательская, практическая, производственная, др.) учебная
Источник тематики (кафедра, предприятие, НИР) кафедра

График выполнения НИР: 25% к 2 нед., 50% к 3 нед., 75% к 5 нед., 100% к 6 нед.

Техническое задание: Требуется создать сцену с значительным числом деревьев в Unreal Engine 5. Провести анализ производительности полученной системы. Исследовать доступные методы решения проблем рендеринга и повышения производительности. Провести анализ результатов применения изученных методов.

Оформление научно-исследовательской работы:

Расчетно-пояснительная записка на 19 листах формата А4.
Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.):

Дата выдачи задания «25» апреля 2025 г.

Руководитель НИР _____ **Витюков Ф.А.**
(Подпись, дата) И.О. Фамилия

Студент _____ **Самарин С.Д.**
(Подпись, дата) И.О. Фамилия

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Создание элементов растительности является одной из самых трудоёмких задач в компьютерной графике. Эта задача обусловлена необходимостью обработки больших массивов объектов, сконцентрированных на ограниченном пространстве. В связи с подобными задачами возникает ряд проблем, влияющих как на визуальное качество сцены, так и на оптимизационные параметры. Настоящая работа посвящена анализу современных методов и особенностей рендеринга объектов растительности в Unreal Engine 5. Основной акцент сделан на изучении проблем обработки полупрозрачных текстур и их взаимодействия, а также методов и способов повышения производительности.

Для разработки проекта был выбран графический движок Unreal Engine 5.2, обладающий значительным набором инструментов для разработки 3D-приложений и совместимостей с другими приложениями в данной области. Значительными преимуществами Unreal Engine является полностью открытый исходных код, гарантирующий одновременно свободную доступность программы и полную возможность изучения внутреннего устройства, активное взаимодействие разработчиков с сообществом, обеспечивающее постоянное совершенствование и оптимизацию работы, а также огромный функционал с мировой известностью, которым пользуются ведущие разработчики игровой индустрии, дизайнеры и кинематографы.

Разработка в Unreal Engine 5 грубо делится на 2 взаимодействующие системы: скрипты Blueprint и модифицированной код C++. Каждая из систем обладает своими преимуществами в зависимости от задач. Написание в коде C++ является абсолютно приоритетным для написания логики событий, начиная от реализации математических алгоритмов и заканчивая работой поведения ИИ. Система Blueprint также позволяет значительно проще и без серьёзных знаний программирования реализовывать значительную часть функционала, приоритетной для написания в коде, однако зачастую это сопряжено с худшей производительностью и в целом это является плохим тоном программирования. Наилучшее применение данной системы — это написание скриптов для задач,

которые сложно осуществить в коде, зачастую такие задачи сопряжены с настройкой материалов или некоторых актёров, например, камеры и её управления. В соответствии с поставленными целями всю работу над проектом рационально выполнять при помощи системы Blueprint, не прибегая к использованию кода.

1. Создание объектов растительности

Для добавления растительности в среде Unreal Engine 5 (UE5) существует собственный режим Foliage, позволяющий насыщать местность любыми объектами растительности, от травы до деревьев. Программные инструменты UE5 позволяют создавать объекты растительности самостоятельно, однако зачастую более рациональное решение заключается в создании объектов в специализированном программном обеспечении, позволяющем создавать статическую сетку растительности и прочую информацию необходимую движку, с последующим добавлением подобных объектов с помощью экспорта. Также Unreal Engine позволяет добавлять растительности с помощью поддерживаемых наборов и плагинов.

Для добавления объектов и текстур травы существует библиотека Quixel Megascans, которая обладает рядом наборов доступных для свободного использования. Для добавления объектов деревьев и кустов существует сервис FAB, созданный разработчиками движка. Данный сервис располагает широким спектром моделей, доступных к свободному использованию в личных целях.

Добавление объектов в среду редактора после установки нужного набора можно осуществить при помощи импортирования объектов растительности в проект при помощи функционала управляющего приложения разработчика Epic Games Launcher. Для этого необходимо перейти в секцию Unreal Engine, затем перейти на вкладку Library и внизу для установленного пакета выбрать проект, к которому необходимо добавить объекты набора.

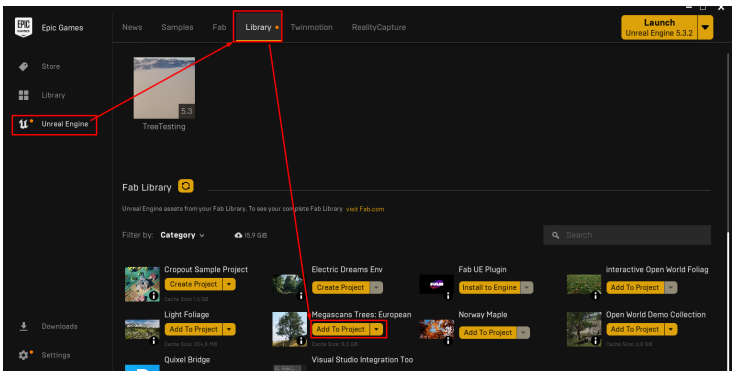

Рисунок 1. Добавление расширений в проект

Для помещения растительности в сцену необходимо перейти в режим Foliage и перетащить имеющиеся статические сетки объектов растительности в поле используемых объектов В режиме Foliage существует множество инструментов, наиболее важными для поставленной задачи являются инструменты Paint и Erase. Первый инструмент позволяет добавлять объекты растительности на имеющиеся объекты ландшафта или другие статические объекты при помощи инструмента схожего с кистью, при этом существует возможность регулирования размера кисти, кучности добавления, а также различных псевдослучайных вариаций стандартных параметров объекта, таких как размер и наклон. Второй инструмент позволяет удалять выбранные объекты растительности с указанной области.

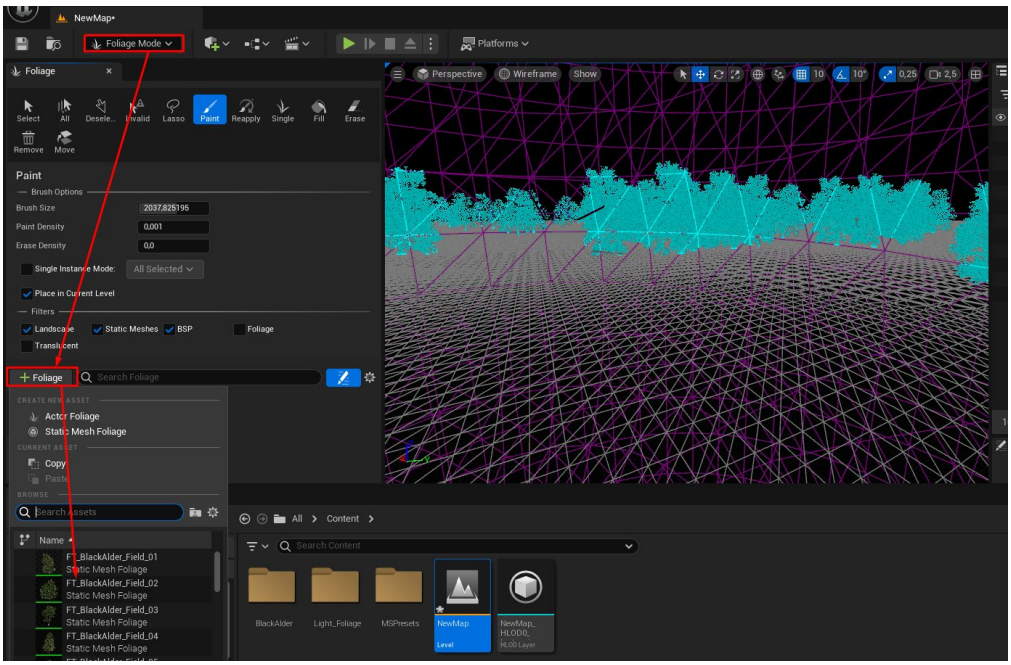

Рисунок 2. Добавление растительности в режиме Foliage

После этого была создана простая сцена с плоскостью без текстур для минимизации воздействия на производительность и при помощи режима Paint в сцену было добавлено 300 объектов деревьев, при этом 100 из них были помещены отдельно для возможности дальнейшей оценки оптимизации объектов вне поля зрения наблюдателя.

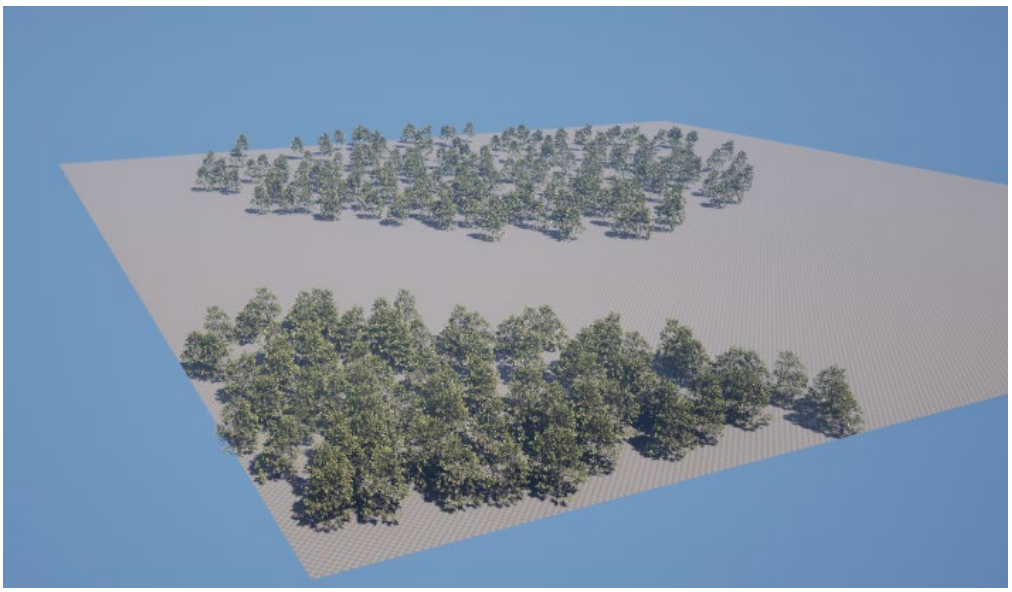

Рисунок 3. Сцена после добавления объектов растительности

2. Измерение вычислительных нагрузок

Для изначальных измерений была отключена симуляция ветра. После этого были получены данные производительности сцены с нескольких ракурсов и режимов отображения. Первый ракурс имеет точку обзора над всеми 300-ми деревьями, захватывая большую часть деревьев непосредственно в поле зрения. Второй ракурс ставит точку наблюдения на уровень деревьев с целью захвата всех объектов, при этом многие из которых перекрывают друг друга. Третий ракурс наблюдает за большей группой на высоте деревьев, при этом исключая из поля зрения 100 изолированных объектов, с целью изучения влияния нахождения объектов вне поля зрения.

Ниже представлены результаты замеров производительности по трём выбранным ракурсам обзора точки наблюдения.

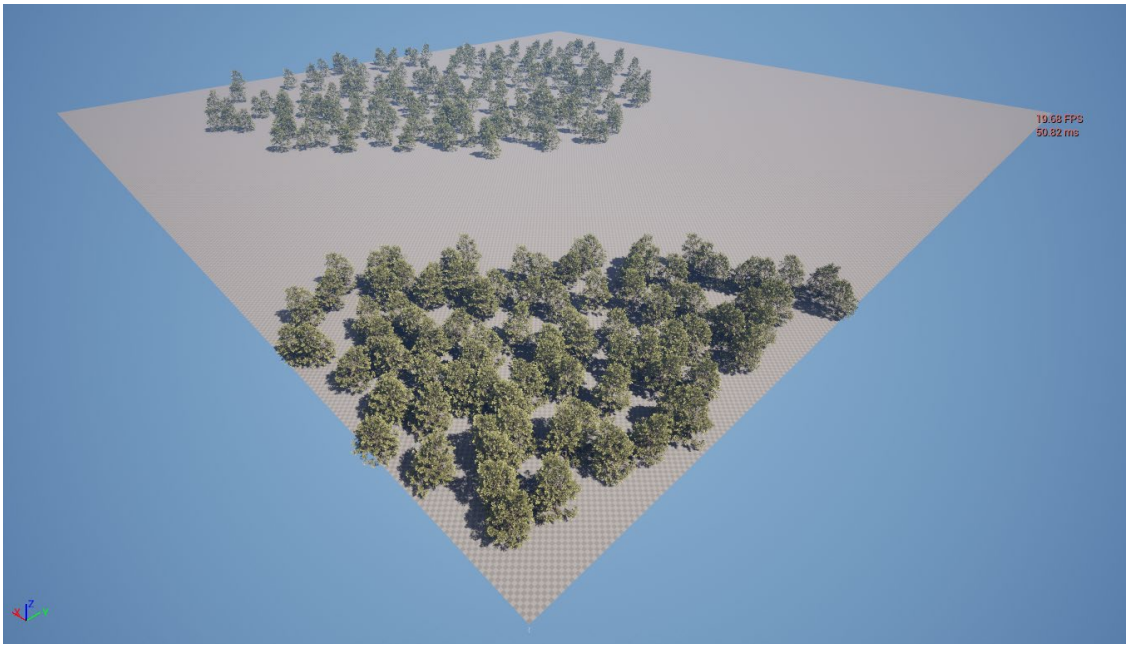

Рисунок 4. Ракурс №1 до оптимизации, 19 FPS

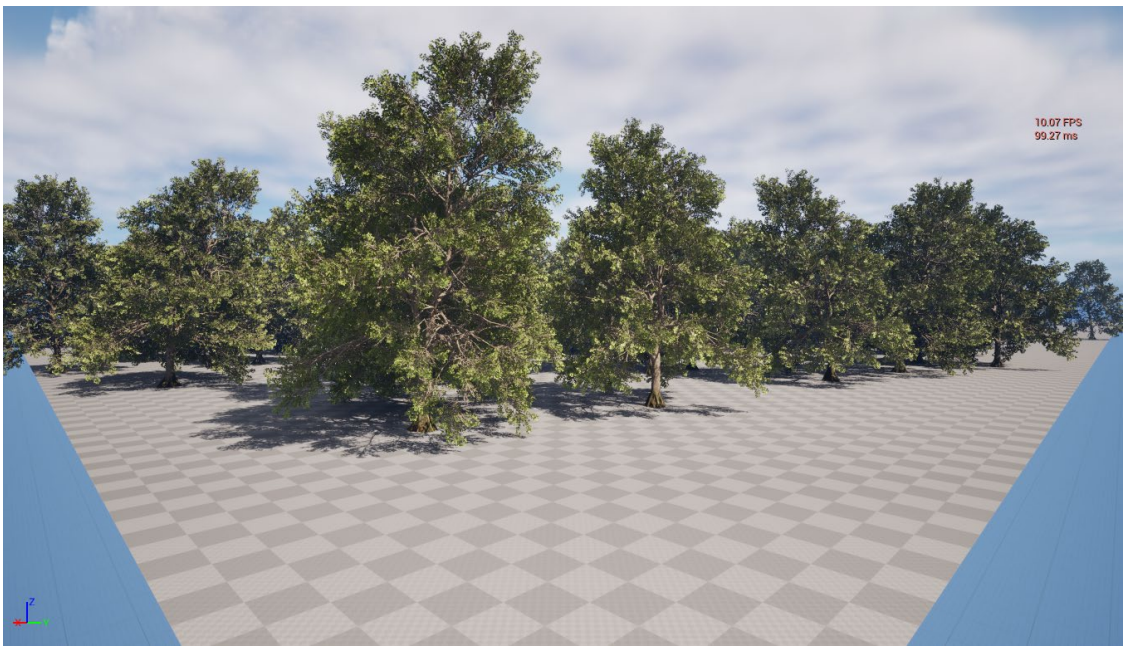

Рисунок 5. Ракурс №2 до оптимизации, 10 FPS

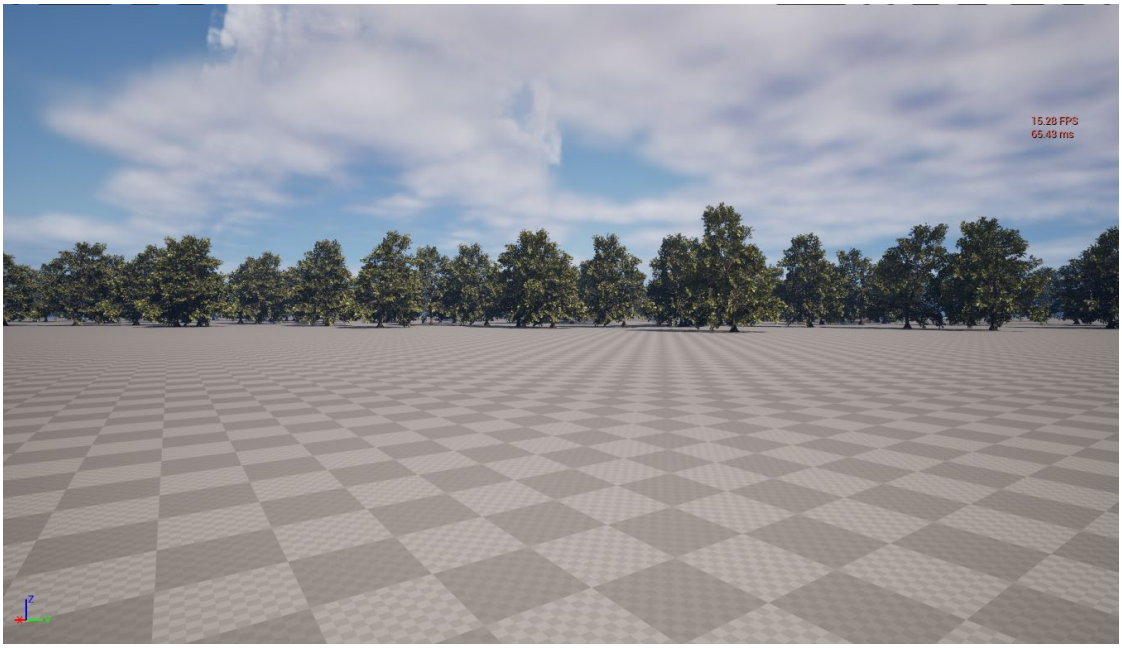

Рисунок 6. Ракурс №3 до оптимизации, 15 FPS

3. Методы устранения проблем рендеринга

Одной из первоочередных проблем рендеринга является проблема альфа сортировки. Для рендеринга кадра применяется множество вычислительных методов. Важными для исследования в рамках данной работы являются следующие:

1) 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.

2) 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.

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

1) 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.

2) 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.

3) 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.

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

5) Специфика проблем. Указанные проблемы не ограничиваются рендерингом объектов растительности, однако они проявляются наиболее выраженно 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. 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 следующими выражениями:

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

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

где объект final представляет результат блендинга двух объектов на текущем этапе, previous является предыдущим результатом, fragment – текущий объект, подлежащий смешиванию, а параметры rgb и α отвечают за цвет и прозрачность объекта соответственно.

Алгоритм порядко-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:

1) 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.

2) 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.

3) 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.

4) 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.

Второй значимой проблемой рендеринга является проблема перерисовки (overdraw). Большое количество элементов в сцене часто приводит к снижению производительности, ключевым фактором которого является избыточная отрисовка объектов. Для решения проблемы перерисовки применяются следующие современные подходы:

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

2) 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.

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

Помимо упомянутых методов, для решения проблем оптимизации возможно уменьшение интенсивности анимации ветра, отключение физики и коллизии. В зависимости от задач, поставленных определённым объектам подобные изменения не повлекут за собой существенного падения в качестве.

4. Анализ результатов оптимизации

В процессе оптимизации была выключена функция Nanite, установленная по умолчанию. Проблема в работе данной системы заключается в необходимости настройки взаимодействия объекта с данной системой. Применённые объекты деревьев имели необходимо настроенные уровни детализации, однако по причине отсутствия взаимодействия с системой Nanite были недоступны к использованию настройки уровней LOD. После этого было использование инстанцирование одинаковых объектов растительности, проведена ручная настройка дальности рендеринга объектов при использовании ракурсов с перекрывающимися объектами и изменены параметры детализации теней.

Были получены следующие результаты оптимизации для выбранных ракурсов:

1. 33 FPS. Прирост производительности в среднем составил 73%.
2. 17 FPS. Прирост производительности в среднем составил 70%.
3. 27 FPS. Прирост производительности в среднем составил 80%

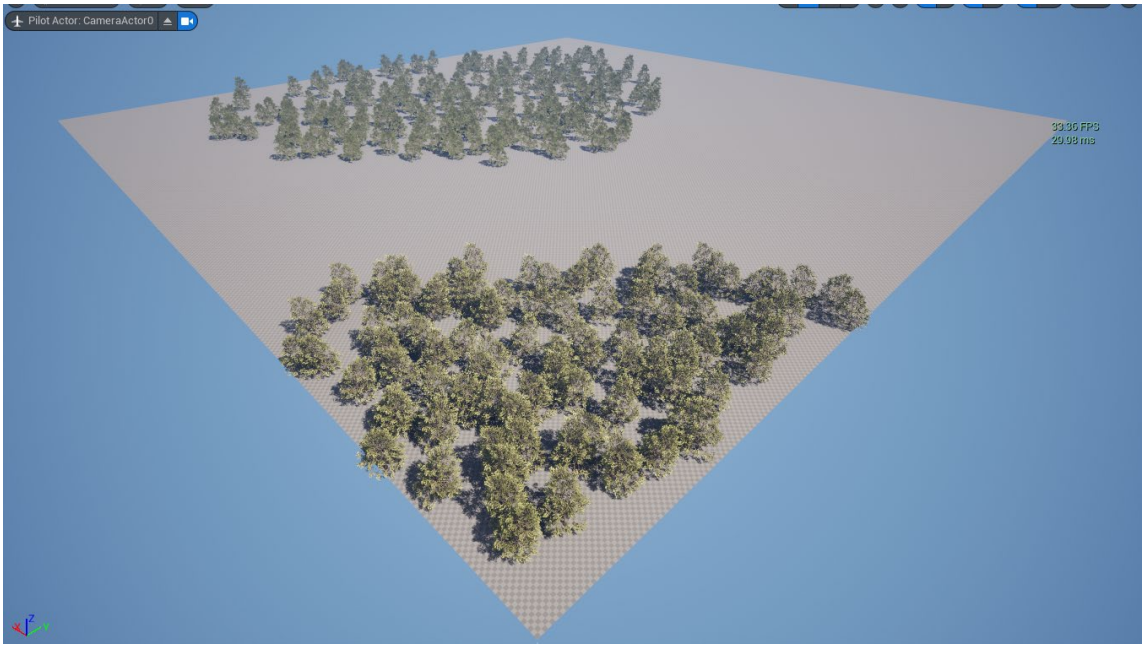

Рисунок 7. Ракурс №1 после оптимизации, 33 FPS

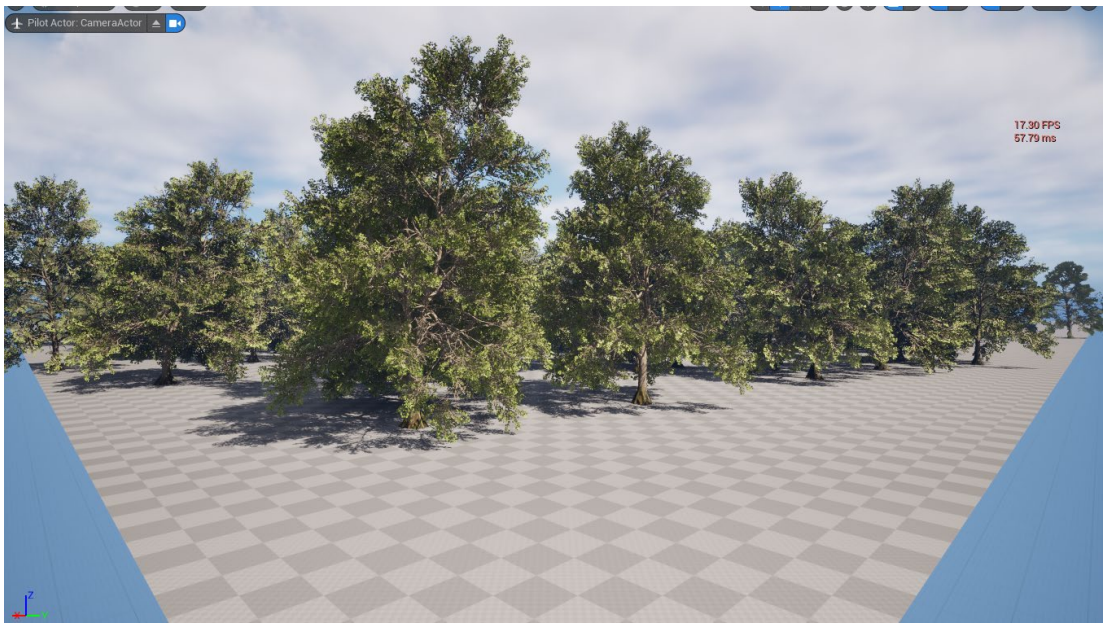

Рисунок 8. Ракурс №2 после оптимизации, 17 FPS

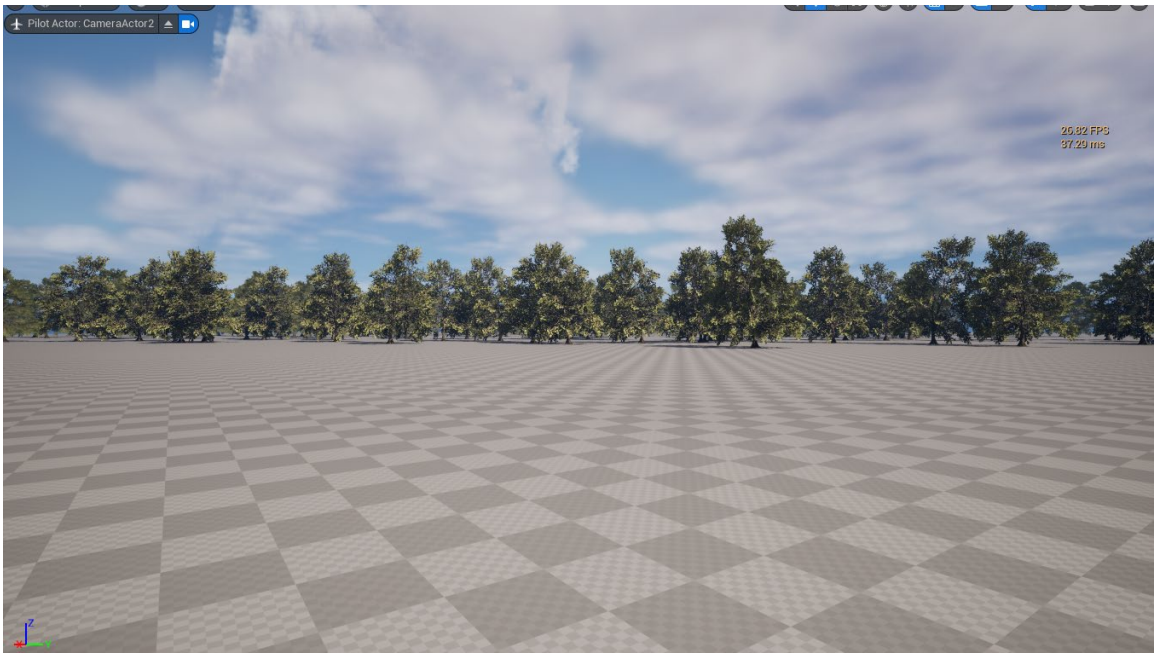

Рисунок 9. Ракурс №3 после оптимизации, 27 FPS

В среднем после применения методов оптимизации наблюдается рост производительности от 70%.

ЗАКЛЮЧЕНИЕ

В рамках исследования были проанализированы актуальные проблемы, связанные с рендерингом объектов растительности в Unreal Engine 5. Проведено построение значительного числа объектов растительности со стандартным освещением, после чего был сделан анализ производительности в зависимости от параметров точки наблюдения и настроек среды UE5. Рассмотрены основные аспекты, включая обработку полупрозрачных объектов и оптимизацию производительности. Также были описаны и систематизированы альтернативные методы и подходы, направленные на повышение эффективности рендеринга и минимизацию ошибок при обработке объектов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Польский С.В. Алгоритм художника / Е.В. Мышенков, А.С. Летин // Компьютерная графика. – 2008. — С. 15-17.

2. Using Transparency in Materials / [Электронный ресурс] // Epic Games Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/using-transparency-in-unreal-engine-materials (дата обращения: 06.05.2025).

3. Creating and Using LODs / [Электронный ресурс] // Epic Games Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/creating-and-using-lods-in-unreal-engine (дата обращения: 06.05.2025).

4. Visibility and Occlusion Culling / [Электронный ресурс] // Epic Games Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/visibility-and-occlusion-culling-in-unreal-engine (дата обращения: 06.05.2025).

5. Viewport Modes / [Электронный ресурс] // Epic Games Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/viewport-modes-in-unreal-engine (дата обращения: 06.05.2025).

6. Lumen Global Illumination and Reflections / [Электронный ресурс] // Epic Games Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/lumen-global-illumination-and-reflections-in-unreal-engine?application_version=5.0 (дата обращения: 08.05.2025).