

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное 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»

КАФЕДРА «Системы автоматизированного проектирования (РК-6)»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ
НА ТЕМУ:
«Разработка методов создания ландшафта и элементов
окружения для Role-Playing Game проектов в Unreal Engine
5»

Студент РК6-85Б
(Группа)

(Подпись, дата)

Самарин С.Д.
(Фамилия И.О.)

Руководитель ВКР

(Подпись, дата)

Витюков Ф.А.
(Фамилия И.О.)

Нормоконтролёр

(Подпись, дата)

Грошев С.В.
(Фамилия И.О.)

2025 г.

УТВЕРЖДАЮ

Заведующий кафедрой

РК6

(индекс)

Карпенко А.П.

(подпись)

(дата)

ЗАДАНИЕ на выполнение выпускной квалификационной работы

Студент группы:

Самарин Савва Денисович
(фамилия, имя, отчество)

Тема выпускной квалификационной работы «Разработка методов создания ландшафта и элементов окружения для Role-Playing Game проектов в Unreal Engine 5»

При выполнении ВКР:

	Используются / Не используются	Да/Нет
1)	Литературные источники и документы, имеющие гриф секретности	Нет
2)	Литературные источники и документы, имеющие пометку «Для служебного пользования», иных пометок, запрещающих открытое опубликование	Нет
3)	Служебные материалы других организаций	Нет
4)	Результаты НИР (ОКР), выполняемой в МГТУ им. Н.Э.Баумана	Нет
5)	Материалы по незавершенным исследованиям или материалы по завершенным исследованиям, но ещё не опубликованные в открытой печати	Нет

Тема квалификационной работы утверждена распоряжением по факультету:

Название факультета:

Дата и рег. номер распоряжения:

Часть 1. Аналитическая часть

Разобрать полный цикл создания статических и анимированных 3d-моделей и сцен. Изучить материалы и средства разработки, предоставляемые трехмерным движком Unreal Engine 5.
Исследовать и освоить методы моделирования ландшафта и принципы наложения ландшафтных материалов. Провести анализ экспорта объектов и проблем, возникающих при интеграции в среде Unreal Engine 5. Оптимизировать параметры отрисовки ландшафта.

Часть 2. Практическая часть 1. Создание ландшафта.

Создать ландшафт пользуясь инструментами программного обеспечения World Creator 3. Экспортировать и интегрировать ландшафт в среду 3d-движка Unreal Engine 5. Добавить материалы ландшафта. Настроить освещение. Добавить простейшие объекты имитирующие водную поверхность. Создать элементы окружения и растительности.

Часть 3. Практическая часть 2. 3d-моделирование.

Создать сцены и 3d-модели объектов наполнения, пройдя полный цикл создания статических моделей. Добавить анимацию объектов в сцене. Создать и настроить частицы шерсти. Выполнить экспорт ассетов и их импорт в движок Unreal Engine 5.

Часть 4. Практическая часть 3. Создание элементов RPG окружения.

Создать RPG камеру с ограниченным перемещением. Добавить отображение игровых параметров пользователя и базовой системы взаимодействия с предметами. Настроить отображения и возможность взаимодействия с полученными объектами с помощью системы инвентаря.

Оформление выпускной квалификационной работы

Расчетно-пояснительная записка на __ листах формата А4.

Перечень графического (илюстративного) материала (чертежи, плакаты, слайды и т.п.):

<i>Работа содержит __ графических листов формата А4.</i>

Дата выдачи задания: «10» февраля 2025 г.

В соответствии с учебным планом выпускную квалификационную работу выполнить в полном объеме в срок до «__» июня 2025 г.

Руководитель квалификационной работы	_____	Витюков Ф.А.	_____
	(подпись)	(инициалы, фамилия)	(дата)
Студент	_____	Самарин С.Д.	_____
	(подпись)	(инициалы, фамилия)	(дата)

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

ФАКУЛЬТЕТ

РК

УТВЕРЖДАЮ

КАФЕДРА

РК6

ГРУППА

РК6-85Б

Заведующий кафедрой

РК6

(индекс)

Карпенко А.П.

(подпись)

(дата)

**КАЛЕНДАРНЫЙ ПЛАН
выполнения выпускной квалификационной работы**

Студента

Самарина Саввы Денисовича

(фамилия, имя, отчество)

Тема квалификационной работы «Разработка методов создания ландшафта и элементов
окружения для Role-Playing Game проектов в Unreal Engine 5»

№ п/п	Наименование этапов выпускной квалификационной работы	Сроки выполнения этапов		Отметка о выполнении	
		план	факт	Должность	ФИО, подпись
1.	Задание на выполнение работы. Формулирование проблемы, цели и задач работы	10.02.2025		Руководитель ВКР	Витюков Ф.А.
2.	1 часть. Аналитическая часть	24.02.2025		Руководитель ВКР	Витюков Ф.А.
3.	Утверждение окончательных формулировок решаемой проблемы, цели работы и перечня задач	29.02.2025		Заведующий кафедрой	Карпенко А.П.
4.	2 часть. Практическая часть 1	20.04.2025		Руководитель ВКР	Витюков Ф.А.
5.	3 часть. Практическая часть 2	16.05.2025		Руководитель ВКР	Витюков Ф.А.
6.	4 часть. Практическая часть 3	25.05.2025			
7.	1-я редакция работы	28.05.2025		Руководитель ВКР	Витюков Ф.А.
8.	Подготовка доклада и презентации	04.06.2025		Руководитель ВКР	Витюков Ф.А.
9.	Заключение руководителя	8.06.2025		Руководитель ВКР	Витюков Ф.А.
10.	Допуск работы к защите на ГЭК (нормоконтроль)			Нормоконтролер	Грошев С.В.
11.	Внешняя рецензия			Секретарь ГЭК	
12.	Защита работы на ГЭК	24.06.2025		Секретарь ГЭК	

Руководитель квалификационной работы

(подпись)

(инициалы, фамилия)

(дата)

Студент

(подпись)

(инициалы, фамилия)

(дата)

// Здесь должно быть направление на защиту

АННОТАЦИЯ

Работа направлена на изучение и создание элементов ландшафта и объектов составляющих основную архитектуру проектов типа RPG (Role-Playing Game).

В проекте рассмотрено создание карты ландшафта, материалов визуального наполнения местности, RPG камеры с ограничением возможности изменения угла обзора и направления перемещения, объектов растительности и методов повышения производительности сцены.

Для наполнения игрового окружения были смоделированы и экспортированы 3d-объекты с использованием редактора Blender.

После создания наполнения были настроены элементы пользовательского управления и наблюдения, соответствующего RPG проекту.

Также в процессе реализации некоторых этапов были произведена оптимизация для улучшения производительности и ресурсоёмкости проекта.

Тип работы: выпускная квалификационная работа.

Тема работы: «Разработка методов создания ландшафта и элементов окружения для Role-Playing Game проектов в Unreal Engine 5».

Объекты исследований: 3d-моделирование, разработка внутриигровых систем, повышение производительности, разработка RPG окружения.

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

Game engine (игровой движок) – набор ключевых компонентов программного обеспечения, используемых для разработки 2d и 3d-приложений. Как правило, инструменты движка абстрагированы от специфики конкретной задачи, предоставляя широкий спектр как программных инструментов взаимодействия посредством кода, так и при помощи встроенного интерфейса редактора для непосредственной настройки параметров и визуального состояния приложения или отдельных элементов.

Unreal Engine 5 (UE5) – игровой движок, разрабатываемый и поддерживаемый компанией Epic Games, которая предоставляет открытый доступ к исходным файлам программы при индивидуальной некоммерческом использовании.

3d-model (3d-модель) – математическое представление объекта в трехмерном пространстве.

3d-modeling (3d-моделирование) – процесс создания 3d-модели объекта.

3d hard-surface modeling (3d-моделирование твердых поверхностей) – процесс создания 3d-моделей объектов, имеющих гладкие поверхности и острые углы. Как правило, в данную категорию входят неживые объекты, например, механизмы, машины, здания, мебель.

3d organic modeling (3d-моделирование органических объектов) – процесс создания 3d-моделей объектов живой природы и объектов, имеющих похожие на них формы. В данную категорию входят такие объекты моделирования, как люди, животные, гуманоиды, растения.

Текстурирование – процесс добавления деталей и параметров 3d-моделям с целью имитации визуальных характеристик реальных объектов. Осуществляется путём наложения 2d-изображений или текстур на 3d-объект в развёрнутом виде, где наложенная текстура интерпретируется как определённое визуальное свойство на соответствующей точке поверхности объекта.

High-poly модель (высокополигональная модель) – максимально детализированная версия 3d-модели. Имеет большое количество полигонов (от

нескольких сотен тысяч до миллионов), в основном используется для дальнейшего запекания текстур.

Low-poly модель (низкополигональная модель) – модель, содержащая относительно низкое количество полигонов (несколько тысяч), при этом сохраняющая основные геометрические свойства объекта. Низкополигональные модели широко используются при отрисовке в реальном времени, поскольку затраты на их обработку приемлемы для получения достаточно плавного изображения.

Actor (актёр) – в рамках движка UE5 любой объект, который может быть размещен на уровне. Базовый программный класс – AActor.

Actor Component – специальный тип объекта, который может быть присоединен к выбранному Actor объекту как подобъект (subobject). Как правило, используется для внедрения функциональности, общей для различных Actor объектов. Базовый программный класс – UActorComponent.

Transform (трансформ) – данные о местоположении, повороте и масштабе объекта. Представляются матрицей преобразований.

Scene Component (USceneComponent) – класс, производный от UActorComponent. Представляет собой компонент, который может иметь свой трансформ на сцене. Данный компонент используется для внедрения функциональности, не требующей геометрического представления.

Полигональная сетка – набор вершин, рёбер и граней, определяющий внешний вид многогранного объекта.

Полигон – многоугольник, являющийся гранью или набором граней 3d-сетки. Основные типы: треугольник (tri), четырёхугольник (quad) и n-gon (5 или более вершин).

Sculpting (скульптинг) – процесс создания и детализации 3d-моделей с помощью 3d-кистей.

Rendering (рендеринг, отрисовка) – процесс получения 2d-изображения по имеющейся 3d-модели.

Polycount – количество полигонов модели.

Static Mesh (UStaticMesh) – класс, представляющий собой статический геометрический объект. Хранит данные о полигональной сетке модели.

Текстура – изображение, накладываемое на поверхность 3d-модели. Может содержать одно или несколько свойств поверхности, например, цвет, жёсткость (roughness), смещение (displacement), направление нормалей (normal map) и т.д.

Материал – набор свойств, определяющих поведение света при отражении от поверхности. Материалы также могут использовать одну или несколько текстур.

Static Mesh Component (UStaticMeshComponent) – компонент, который может иметь статический меш и набор материалов, применимых к нему.

Geometry instancing (инстансинг, дублирование геометрии) – техника, позволяющая отрисовывать множество однотипных элементов за один проход.

Instanced Static Mesh Component (UInstancedStaticMeshComponent, ISMC) – класс, производный от UStaticMeshComponent, позволяющий использовать механизм инстансинга геометрии.

LOD (Level of Detail, уровни детализации) – техника, позволяющая подменять разные по детализации версии модели в зависимости от дистанции между камерой и объектом, либо в зависимости от процента площади экрана, занимаемой моделью.

Ретопология – процесс изменения топологии полигональной сетки модели с целью ее упрощения/улучшения.

UV-unwrapping (UV-развертка) – процесс создания развертки 3d-модели. Представляет собой процесс получения соответствия между координатами поверхности 3d-модели и координатами на текстуре (U по горизонтали, V по вертикали).

Texture baking (запекание текстур) – процесс переноса деталей поверхности high-poly модели на набор текстур.

Hierarchical Instanced Static Mesh Component (HISMC, UHierarchicalInstancedStaticMeshComponent) – класс, производный от UInstancedStaticMeshComponent, позволяющий использовать LOD.

Central Processing Unit (CPU) – центральный процессор.

Graphics Processing Unit (GPU) – графический процессор (видеокарта).

Frames per second (FPS) – количество кадров в секунду.

Rendering Hardware Interface (RHI) – в UE5 надстройка над множеством графических API (например: DirectX, OpenGL, Vulkan), позволяющая писать независимый от платформы код.

Asset (ассет) – в контексте компьютерных игр, объект, представляющий собой единицу контента. Игровыми ассетами являются 3d-модели, текстуры, материалы, аудиофайлы, и т.д. Как правило, созданием ассетов занимаются художники, дизайнеры, музыканты.

Reference (референс) – изображение, используемое художником в процессе 3d-моделирования для получения дополнительной информации о моделируемом объекте.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....	12
1. Постановка задачи.....	14
2. Создание ландшафта.....	15
2.1. Разработка каркаса ландшафта.....	15
2.2. Настройка материалов.....	18
2.3. Реализация импорта в Unreal Engine.....	21
2.4. Оптимизация сетки ландшафта	27
3. 3d-моделирование	31
3.1. Обзор интерфейса	31
3.2. Базовые методы моделирования	34
3.3. Методы полигонального моделирования объектов	37
3.4. Методы «скульптуинга».....	45
3.5. Экспорт и импорт.....	52
4. Создание элементов RPG окружения.....	54
4.1. Добавление растительности и воды.....	54
4.2. Создание сетки NavMesh.....	58
4.3. Создание RPG камеры	59
4.4. Оптимизация рендеринга растительности	61
4.4.1. Создание объектов растительности	61
4.4.2. Измерение вычислительных нагрузок.....	64
4.4.3. Методы устранения проблем рендеринга	66
4.4.4. Анализ результатов оптимизации	70
4.5. Добавление пользовательского интерфейса	73
ЗАКЛЮЧЕНИЕ	80
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	81
ПРИЛОЖЕНИЕ А. Графическая часть выпускной квалификационной работы	83

ВВЕДЕНИЕ

Разработка 3D-приложений в современной индустрии представляет собой широкий спектр прикладных задач, требующих глубоких знаний и навыков в области компьютерной графики, программирования, 3d-моделирования, текстурирования, оптимизации производительности, симуляции физического взаимодействия и создания пользовательского интерфейса. Для эффективной реализации непосредственных навыков разработчика требуется наличие мощного программного обеспечения, способного предоставить необходимый функционал с наглядным отображением и доступом к полученному результату в ходе создания. Таким программным обеспечением являются игровые движки.

В данной работе для интеграции результатов полученных прикладных навыков и создания элементов проекта жанра RPG был выбран графический движок Unreal Engine 5.3, обладающий значительным набором инструментов для разработки 3D-приложений и совместимостей с другими приложениями в данной области. Значительными преимуществами Unreal Engine является обширный функционал с возможностью подключения дополнительных модулей и полностью открытый исходных код, который осуществляет доступность программы и полную возможность изучения внутреннего устройства.

Разработка в Unreal Engine 5 делится на 2 взаимодействующие системы: скрипты Blueprint и модифицированной код C++. Каждая из систем обладает своими преимуществами в зависимости от поставленных промежуточных задач. Написание в коде C++ является приоритетным для написания логики событий, начиная от реализации математических алгоритмов и заканчивая работой поведения ИИ. Система Blueprint также позволяет значительно проще и без углублённых знаний программирования реализовывать значительную часть функционала, приоритетной для написания в коде, однако зачастую это сопряжено с худшей производительностью. Наилучшее применение данной системы — это написание скриптов для задач, которые сложно осуществить в коде, зачастую такие задачи сопряжены с настройкой материалов или инструментов взаимодействия между пользователем и программой.

Создание карты является основополагающей задачей значительного числа проектов игровой индустрии и симуляций. Ландшафт прежде всего необходим для расположения и перемещения объектов по его поверхности, создания определённого рельефа и биомов. Добавление растительности и освещения могут помочь в создании реалистичных пейзажей, добавлении атмосферы или преград.

Как правило движок оснащён базовым функционалом, позволяющим добавлять простейшие объекты геометрии, камеры, освещения или ландшафта. Но зачастую, для крупных проектов разработчику приходится создавать конкретные сложные модели, построение которых невозможно без навыков 3D-моделирования. Unreal Engine 5 располагает всем необходимым функционалом создания геометрии для объектов и наложения на них соответствующих материалов и текстур, но на практике лучше всего для 3D-моделирования подходят более специализированные программные продукты, такие как Autodesk Maya, Autodesk 3ds Max или Blender. Более того подобные продукты оснащены встроенными методами экспорта, позволяющего непосредственно переносить созданные модели в среду UE5.

В качестве основного предмета 3D-моделирования была выбрана программа Blender, предоставляющая продвинутый и доступный функционал, с широкой поддержкой передачи данных между Unreal Engine 5.

1. Постановка задачи

В рамках данной работы рассматривается исследование процессов и формирования методов проектирования проектов в среде Unreal Engine 5. В качестве практических задач требуется создать определённую карту (ландшафт) в среде UE5 и добавить на её основе функционал и наполнение характерное проектам типа Role-Playing Game.

Рассматриваемая система состоит из следующих компонентов.

- Карта (ландшафт, представляющий собой сетку поверхности рельефа).
- Окружающая среда:
 - текстурное наполнение (трава, земля, песок);
 - лес (деревья, кустарники);
 - вода (озера, реки, море).

Пользователь должен иметь доступ к следующим параметрам.

- Статусная панель (здоровье, энергия)
- Инвентарь
- Управление ограниченной камеры (обзор под фиксированным углом, ограничения на максимальную и минимальную дальность до объекта)

Для выполнения работы были использованы средства следующих программ:

- Unreal Engine 5 – разработка внутриигровых систем, настройка ландшафта и материалов, интеграция 3d-объектов, создание RPG окружения;
- Blender – создание 3d-моделей объектов;
- World Creator 3 – создание ландшафта и материалов;

2. Создание ландшафта

2.1. Разработка каркаса ландшафта

Для создания RPG карты была составлена приближенная схема расположения значимых объектов и элементов рельефа (рис. 1).

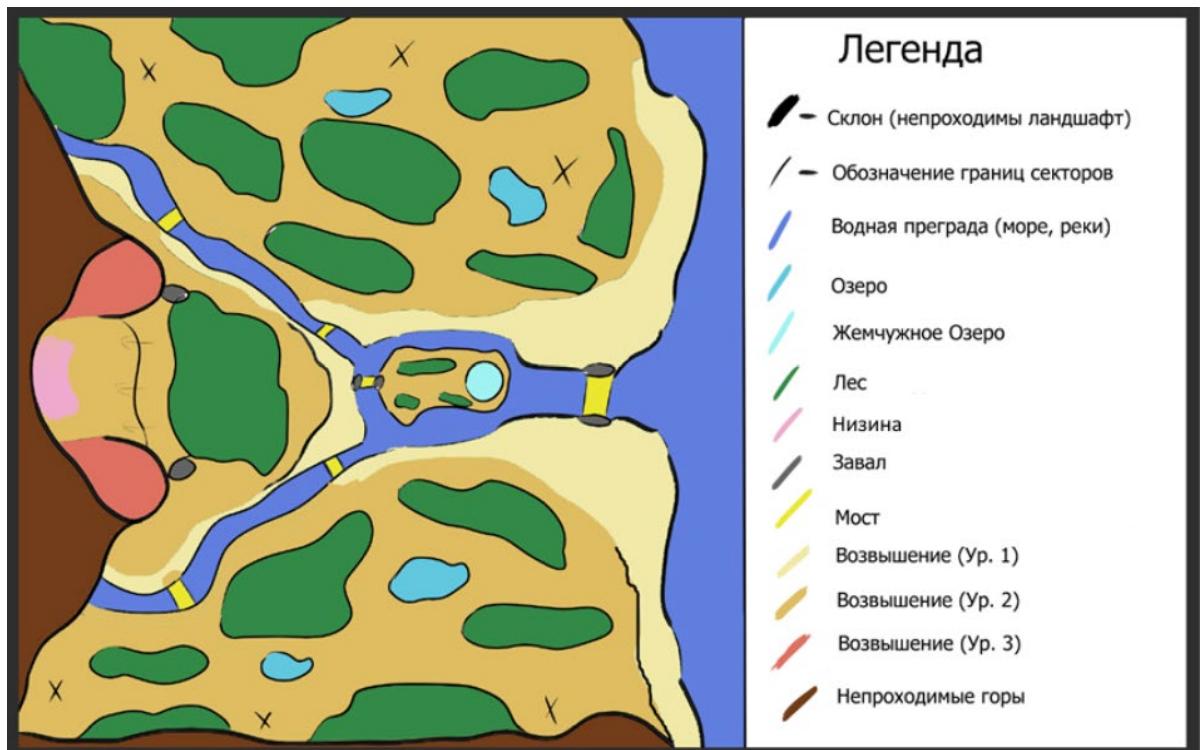


Рисунок 1. Приближенная схема RPG карты

В функционал Unreal Engine 5 входят средства, позволяющие создавать и настраивать ландшафт в самом редакторе [5], однако стоит отметить что возможности движка в некоторых аспектах серьёзно уступают специализированным программам, таким как Gaea, World Machine, World Creator и прочие. Каждая из вышеперечисленных программ обладает своими особенностями, но в рамках выполнения работы выбор остановился на World Creator 3. Подобные редакторы основывают свою генерацию на широком ряде настраиваемых фильтров, базовых формах и различных модификаторах, позволяющих имитировать различные типы воздействия на местность, накладывать или смешивать между собой разные рельефы, добавлять маски или включать разные случайные формы.

В сравнении с аналогами World Creator 3 обладает рядом функций серьёзно облегчающих тонкую настройку желаемого результата. Самой отличительной

чертой является возможность динамического разбиения всего пространства карты на квадратные сегменты указанного числа степени двух. Каждый такой квадрат или кластер квадратов можно вручную делать ниже или выше, регулируя влияние выбранных ячеек на соседние, тем самым допуская теоретически ручное создание любого желаемого ландшафта.

Ранее более тонкую настройку ландшафта можно было осуществлять разве что при помощи наложения масок и подбора необходимых фильтров и модификаций в конкретном локальном месте, что не сопоставимо с трудозатратами при непосредственной манипуляции над сегментами, при том что World Creator 3 не избавился от функций масок. Также в отличии от World Machine и Gaea работает без необходимости указания узлов («нодов») через которые осуществляется всё взаимодействие с ландшафтами. Вместо этого World Creator обладает рядом стандартных параметров настройки и набором добавляемых фильтров.

В процессе создания в первую очередь нужно выбрать стартовый шаблон ландшафта. Каждый из них полностью настраиваем и служит в большей степени как удобный начальный шаг. Далее ландшафт можно изменять, изменения ключи («сиды») генерации, настраивая параметры точности каркасной сетки, а также добавлять различные стили местности и их влияние на общий результат.

Далее после придания необходимой формы накладываются фильтры для добавления эффектов, таких как эрозия разных типов, выравнивание, высушивание, сглаживание острых углов и так далее.

В процессе создания активно использовались фильтры сглаживания пространства, создания «мягких» краёв и эрозии, огрубление обрывов.

Промежуточный результат создания ландшафта без наложения материалов представлен на рисунке 1 и рисунке 2.

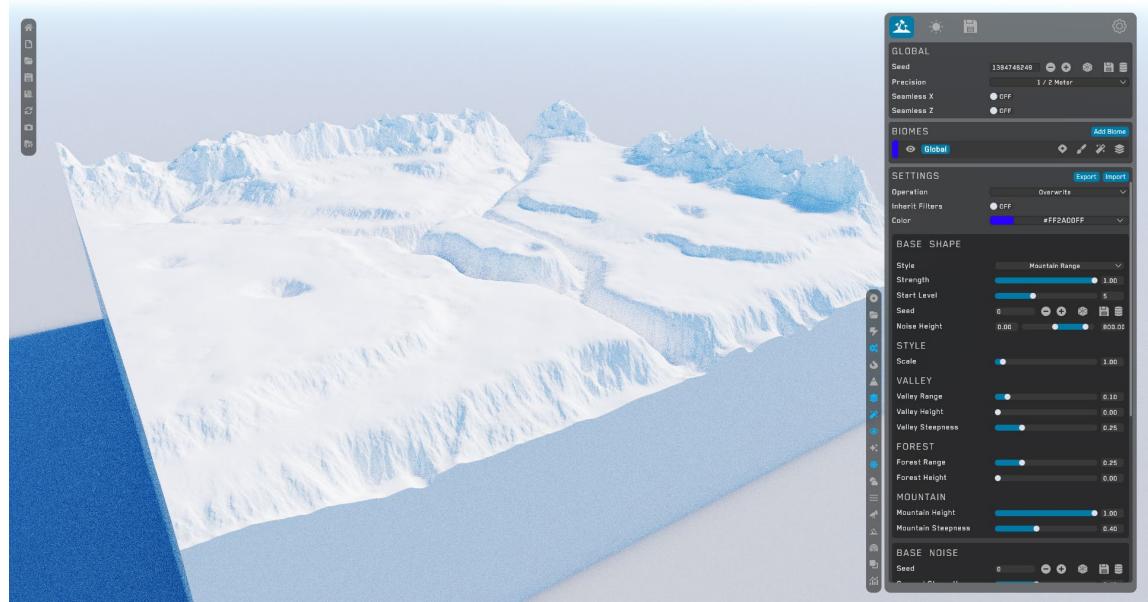


Рисунок 2. Промежуточный ландшафт в World Creator 3, вид сбоку

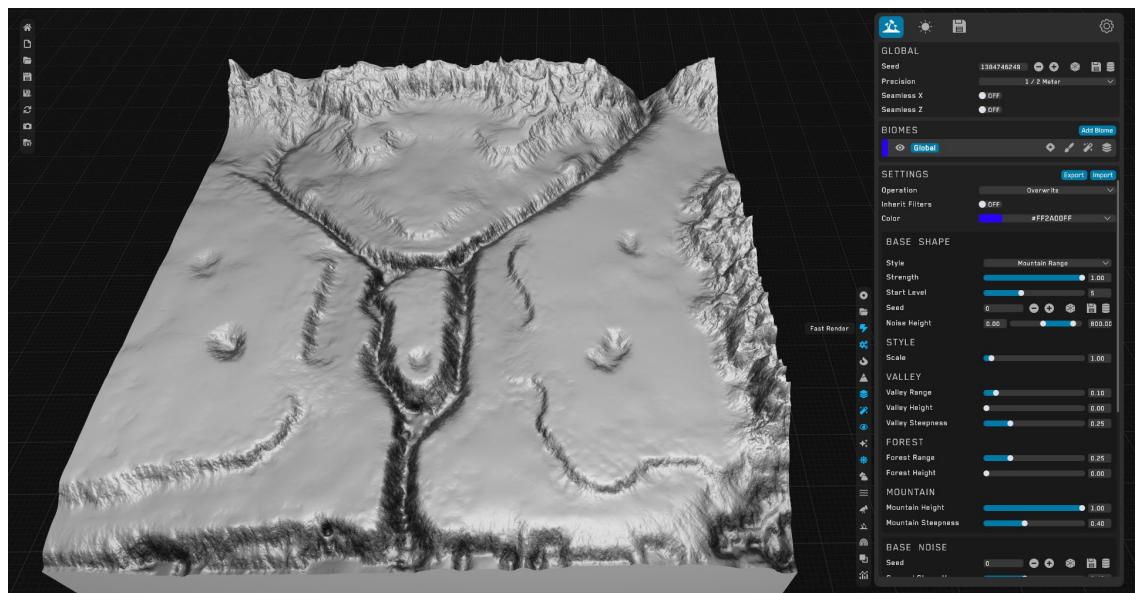


Рисунок 3. Промежуточный ландшафт в World Creator 3, вид сверху под наклоном в фильтре быстрого рендера

Белый вид местности является стандартным представлением при отсутствии каких-либо покрывающих материалов.

2.2. Настройка материалов

Зачастую добавлять материалы можно перед полным определением формы ландшафта, поскольку наиболее удобные и универсальные правила, по которым определяется область для используемого материала, будут менять своё положение вместе с изменением рельефа, предоставляя более качественную картину.

Чтобы определить область, к которой будет определён материал необходимо задать ряд правил распределения. Данные правила либо учитывают имеющийся рельеф, либо напрямую исходят из указанной им маски.

Первый способ позволяет учитывать крутизну склонов, впадины, обрывы, холмы и плоскости, ограничивать зоны по высоте или жёсткости, добавление различных случайных «шумов» по всей площади и многое другое. Данный способ наиболее подходит для задания биомов характерных определённом рельефу, например, задание гор, снега, равнинной земли или травы, впадин или морского дна, песка на берегу водоёмов и так далее.

Второй способ с использованием масок может использоваться для всех вышеперечисленных целей, однако, он не универсален и не будет менять положение области каждый раз, когда происходит изменение ландшафта. Данный способ наиболее рационален для выделения нестандартных областей, например, расширение береговой линии в определённых местах, добавления области лесного наста и другие.

Помимо этого, каждое правило распределения обладает набором модификаторов. Они могут влиять на конечный вид области самыми разными способами начиная от добавления случайных шумов по краям и заканчивая полным инвертированием или добавлением масок таких как, например, диаграмма Вороного произвольного дробления.

После установления области необходимо установить параметры материала.

Существуют два основных способа – при помощи градиентов и при помощи текстур.

Для градиентного метода имеется 5 ячеек, каждая из которых может принять градиент цветовой палитры и любую указанную область.

Текстурный метод требует импортировать в программу 5 ключевых компонентов материала поверхности: карту цветов, карту нормалей, карту жёсткости, карту затенений, карту смещений. Одновременным преимуществом и недостатком этого метода перед градиентным является возможность и необходимость точного задания параметров того как материал ляжет на указанную поверхность, в то время как градиентный метод может дать быстрые результаты.

В рамках практики были использован текстурный метод, ресурсы для которого установлены при помощи библиотеки Quixel Megascans, предоставляющей доступ пользователям Unreal Engine к текстурам, считанным с реальных объектов в высоком разрешении. Были добавлены 6 текстур материалов – лесного настила, снега, гор, песка, каменной поверхности и травы. Для добавления текстуры в материал текстурным методом требуется перенести необходимые вышеупомянутые карты в ячейки редактора World Creator 3.

Для создания областей леса была использована пользовательская маска, с размытием по краям. Для снега выбрана определённая высота, случайная девиация по краям (distortion), размытие (blur), случайные шумы (noise offset) и простые потоки (simple flow), симулирующие осадки. Для гор выбрана высота, параметр жёсткости отвечающий за крутизну, простые потоки, размытие и шумы по краям. Для песка выбрана пользовательская маска со случайной девиацией по краям и размытием. Для каменной поверхности выбрана крутизна. Для травы выбран параметр распределения ровности местности. Иерархия наложения в соответствии с порядком перечисления, где лесной настил находится в верху списка и накладывается поверх остальных материалов.

Результат наложения материалов представлен ниже на рисунках 4 и 5.

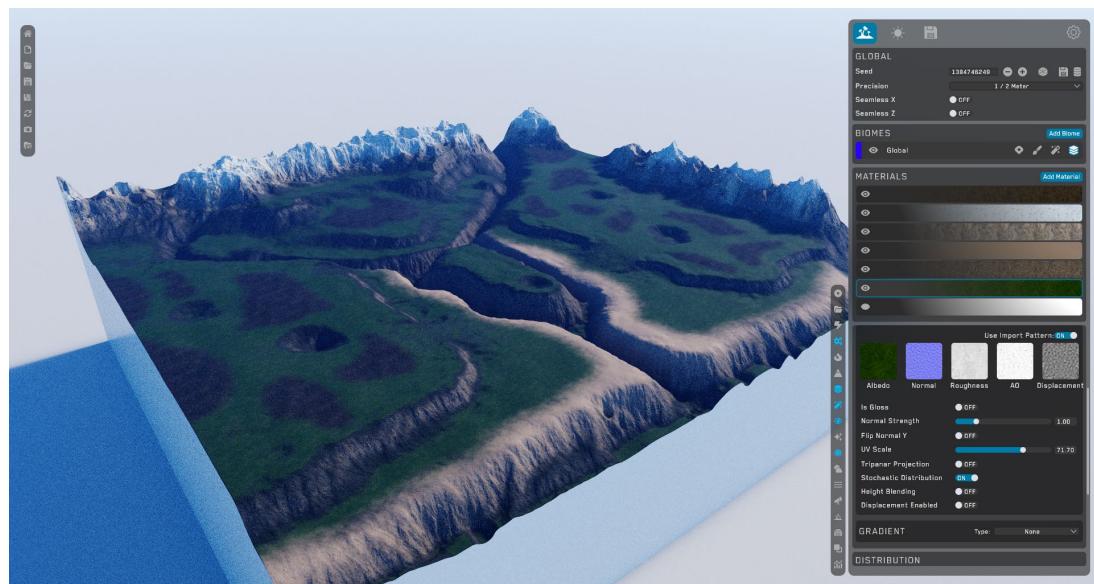


Рисунок 4. Итоговый ландшафт в World Creator 3, вид сбоку

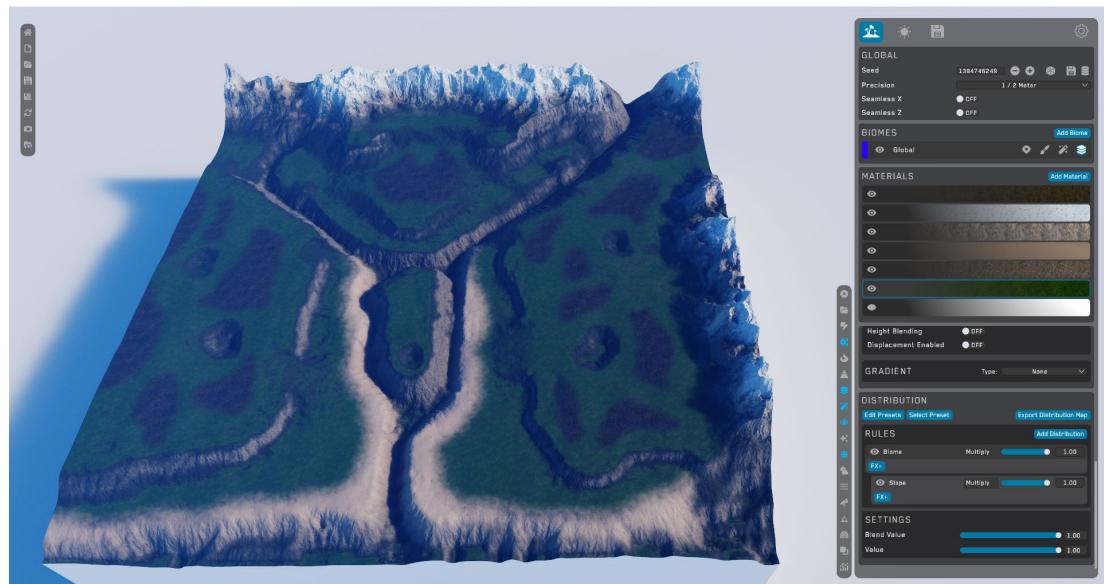


Рисунок 5. Итоговый ландшафт в World Creator 3, вид сверху под наклоном

2.3. Реализация импорта в Unreal Engine

Перед импортом в Unreal Engine требуется определить необходимые для экспорта элементы внутри World Creator 3. World Creator в отличии от World Machine и Gaea не обладает возможностью установления меток экспорта на конкретные узлы за отсутствием таковых в рабочей среде программы. Вместо этого существует отдельная вкладка экспорта, позволяющая экспортировать следующие параметры:

- 1) Цветовая карта (Color Map), содержащая все имеющиеся цвета на ландшафте.
- 2) Карта высот (Height Map), состоящая из монохромного изображения, где степень белого определяет высоту.
- 3) Сетка (Mesh).
- 4) Карта отражаемости (Metalness Map)
- 5) Карта нормалей (Normal Map), цвета на которой отражают различные углы на поверхности ландшафта.
- 6) Карта жёсткости (Roughness Map)
- 7) Splat карта (Splat Map), монохромное изображение отражающую весовую маску, эквивалентную области распределения отдельного материала.

В процессе работы над проектом сначала были экспортированы карта цветов, высот, нормалей и жёсткости, представляющие минимальный необходимый набор для успешного импорта ландшафта в среду Unreal Engine. Однако, в дальнейшем это выявило ряд недостатков, в частности более низкое качество и отсутствие возможности дальнейшей настройки внутри редактора Unreal.

Второй способ включал в себя экспорт карты высот и Splat карт, а также всех материалов, ранее установленных при помощи библиотеки Quixel Megascans.

Для успешного экспорта любых карт в Unreal Engine требуется соблюдать ряд стандартов размерности.

Карта высот должна передаваться в формате raw, глубиной 16 бит, с порядком Little Endian в единичном канале R красного цвета. Также следует включить параметр автоматической нормализации. Поскольку точность ландшафта была изначально задана в 1 квадрат каждые 1/2 метра, общим числом 1024x1024, редактор World Creator 3 автоматически предлагает размер экспортируемой карты 2048x2048. Ближайшая принимаемая величина в Unreal Engine 5 – 2041x2041, которая связана с внутренним устройством движка и по этой причине рекомендуется экспортировать материалы в соответствующем формате. Также для экспорта Splat Map следует указать Single Channel. Остальные параметры экспорта можно оставить по умолчанию для прочих карт.

После экспорта всех необходимых карт в редакторе Unreal Engine 5 требуется произвести импорт в разделе landscapes в панели Import From File и указать карту высот, после чего в случае верно заданных параметров импорта сгенерируется ландшафт без материалов [4]. Результаты импорта ландшафта в базовый уровень Unreal Engine 5 представлены на рисунке 6.

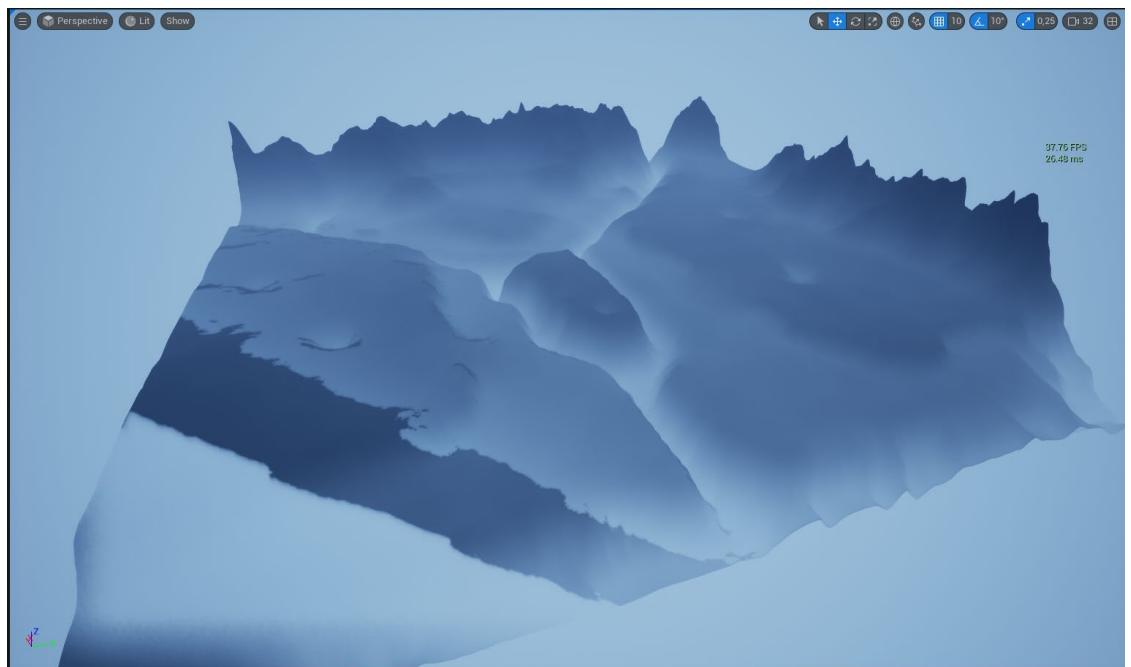


Рисунок 6. Результат импорта карты высот в Unreal Engine

Для создания материала требуется создать Blueprint класс. Согласно конвенции имён в Unreal Engine, файлы материалов должны иметь префикс «M_». Файл назван M_Terrain2K.

Чтобы включить все 6 вышеупомянутых текстур в состав материала требуется составить следующую схему в редакторе Blueprint (рис. 7):

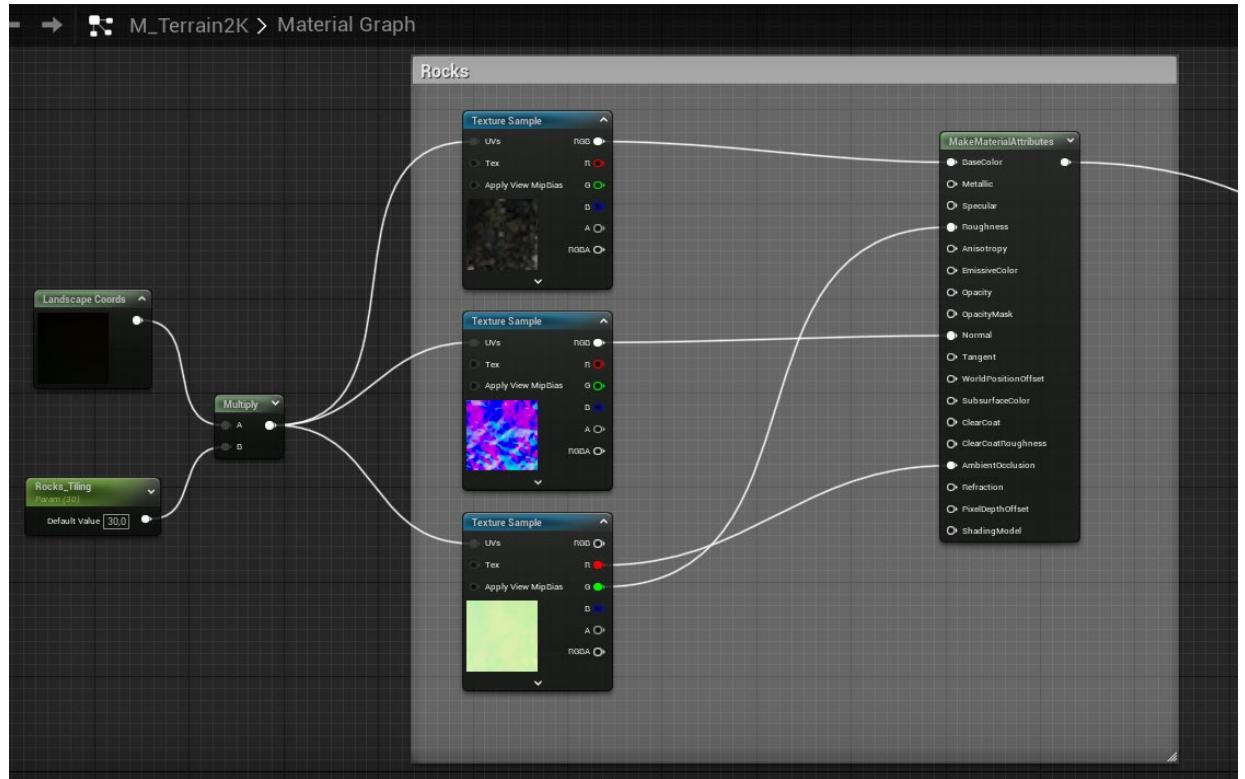


Рисунок 7. Схема включения одной текстуры в рабочее поле системы Blueprint

В данной схеме программа определяет UV координаты ландшафта, которые затем умножаются на изменяемый параметр «тайлинга», ответственного за размер текстур и передаёт эти параметры в 3 необходимых карты имеющегося материала: карту цветов, нормалей и затемнений, которые затем передают результирующие значения в узел создающий понятный для среды набор атрибутов материала [3]. Также в строке Sampler Source для каждой карты необходимо задать параметр Shared Wrap, что позволяет использовать несколько ресурсов одновременно.

После выполнения схожих действий для всех 6 материалов каждому выделяется определённый слой при помощи узла Landscape Layer Blend, с которым позже можно будет взаимодействовать и настраивать параметры смешивания между собой. Каждому слою были присвоены соответствующие имена на английском языке. Для всех слоёв кроме травы был выбран режим смешивания LB Weight Blend. Для травы – LB Height Blend для предотвращения

смешивания слоя травы с слоем каменной поверхности. После этого материалы смешения слоёв передаются выходному узлу материала.

После создания основного класса материала удобно использовать не базовый объект, а создать его инстанцию (material instance), который наследует весь функционал родительского материала, но не оказывает влияние на него при изменении параметров.

Далее инстанция добавляется в соответствующее единственное поле материала ландшафта. Теперь чтобы материал отобразился требуется сгенерировать всю необходимую информацию карты. Для этого в режиме Landscape Mode в разделе Paint для каждого из допустимых 6 слоёв нужно сгенерировать информацию слоя с опцией Weight-Blended Layer [6]. Затем перейдя во вкладку Manage, а затем в Import следует импортировать все Splat карты соответствующих слоёв вместе с картой высот, тем самым заново генерируя ландшафт. Результат импортирования материалов представлен ниже (рис. 8):



Рисунок 8. Импортированный ландшафт в Unreal Engine 5

На полученном результате можно легко заметить регулярность накладываемых текстур. Существует ряд способов, добавляющих нерегулярности в процесс наложения текстур делая их повторение значительно менее заметным. В процессе работы наиболее оптимальным способом было

выбрано добавление схожей текстуры с отличным «тайлингом». Для этого сначала требуется добавить текстуру вышеупомянутым способом через передачу трёх карт и задания атрибутов материала. Далее был добавлен сегмент, отвечающий за шумы, которые будут определять процесс смешивания двух слоёв. После этого два материала смешиваются при помощи звена Blend Material Attributes с использованием маски из составленных ранее шумов. После чего результат смешивания перенаправляется в слой травы. Результаты представлены ниже на рисунках 9 и 10:

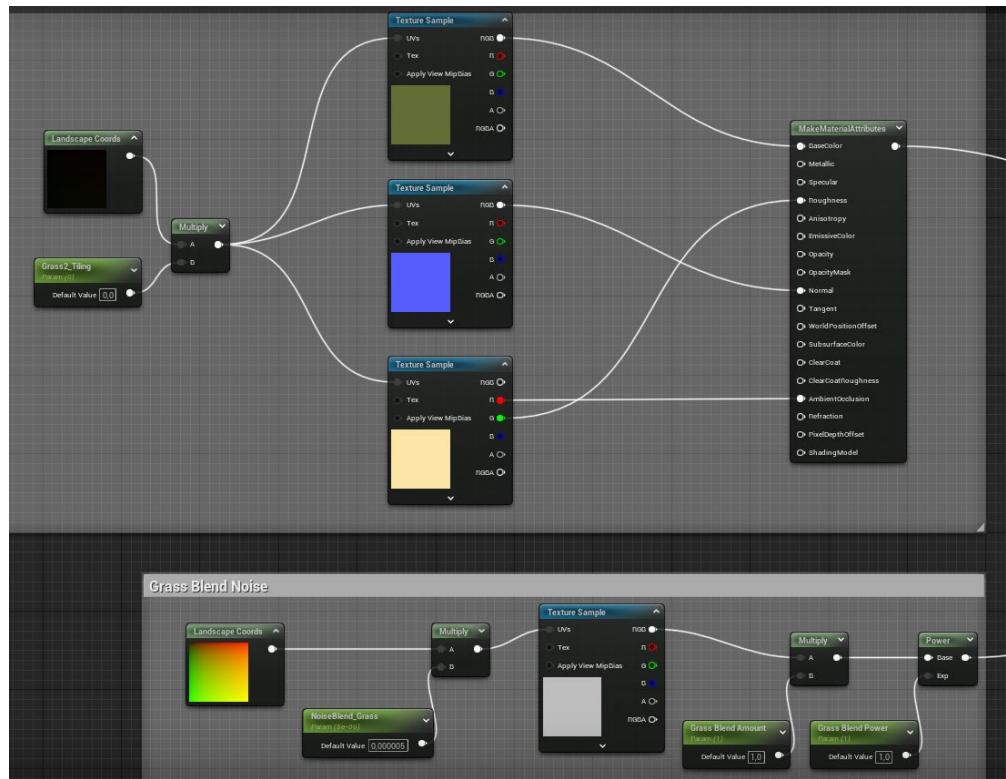


Рисунок 9. Blueprint схема добавления текстуры и шумов

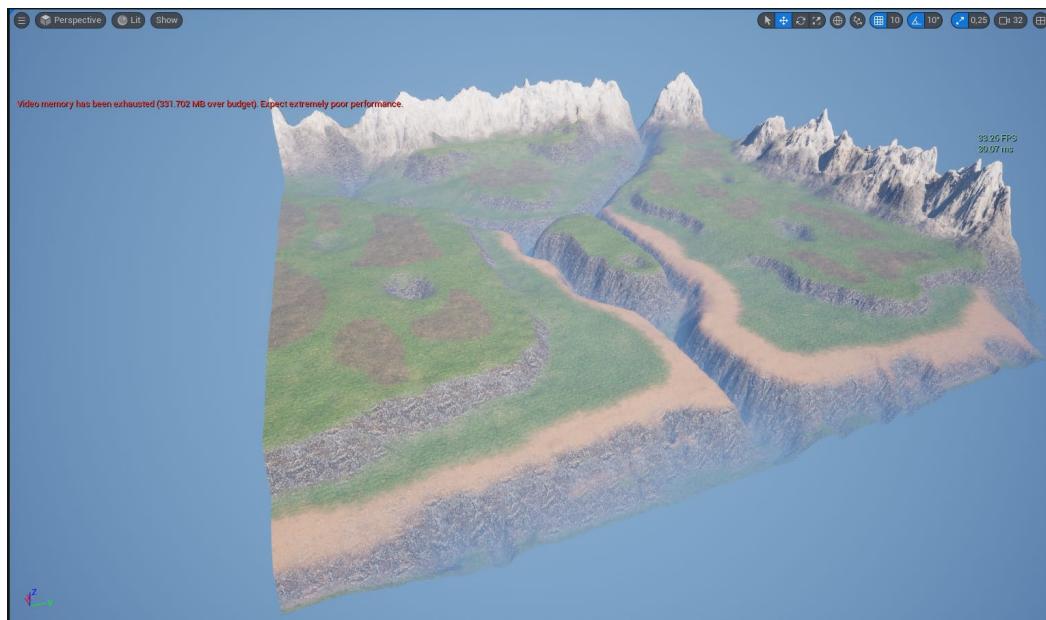


Рисунок 10. Ландшафт после добавления нерегулярностей

2.4. Оптимизация сетки ландшафта

Из вышеперечисленных материалов легко заметить низкие параметры производительности даже при крупных планах. Зачастую показатель FPS (Frames Per Second) находится в диапазоне от 30 до 40. В данном проекте на это в значительной мере влияют показатели GPU (Graphical Processor Unit), освещение и ландшафт.

Основной проблемой, связанной с ландшафтом, является результирующая сетка карты после экспорта [8]. Если переключится в Wireframe mode будет получена следующая картина, указывающая на значительное перенасыщение сетки (рис. 11).

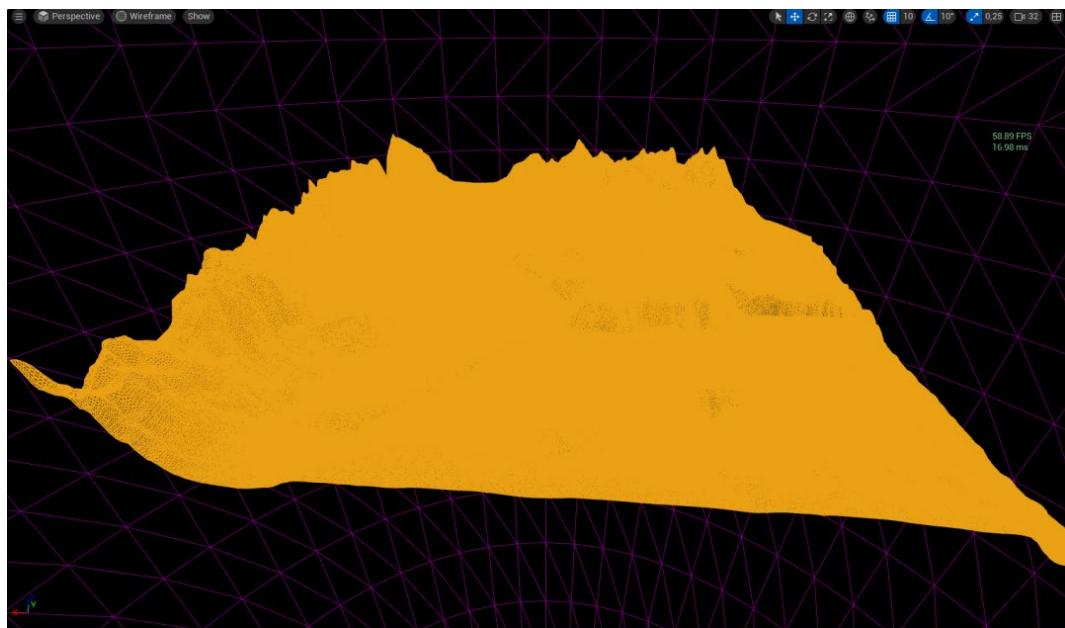


Рисунок 11. Сетка ландшафта до оптимизации, 59 FPS

Подобная детальность является абсолютно излишней и более того не учитывает положение наблюдателя для оптимизации отрисовки при помощи LOD, которые регулируют насколько детальной является сетка, где более высокие уровни отвечают за более дешёвое и грубое распределение. Стоит отметить что использование грубой сетки неминуемо ведёт к созданию более квадратному и примитивному ландшафту, поэтому требуется соблюдать баланс производительности и качества.

По умолчанию редактор Unreal Engine 5 устанавливает параметр максимального числа LOD равное -1, что автоматически допускает

использование максимально возможного числа LOD. Однако, остальные параметры по умолчанию устанавливают приоритетным самый низкоуровневый и детальный LOD 0. Были изменены следующие настройки:

- 1) LOD 0 Screen Size: 0,5 → 1,15
- 2) LOD 0: 1,25 → 1,9
- 3) Other LODs: 3,0 → 1,5

Ниже представлены результаты до и после изменения сетки в режиме Unlit с целью минимизировать влияние света на продуктивность в данных тестах (рис. 12):

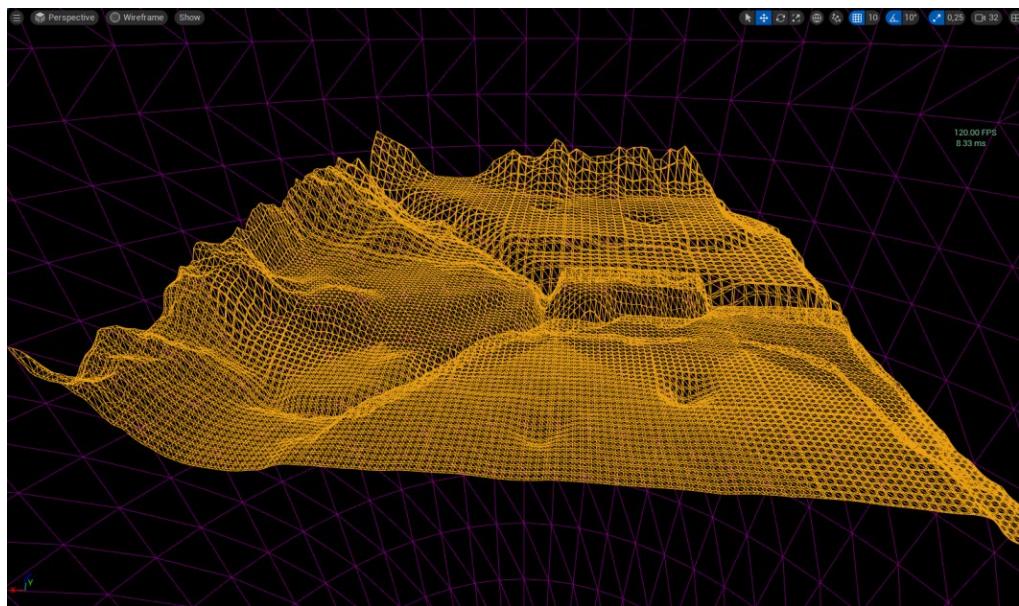


Рисунок 12. Сетка ландшафта после оптимизации, максимальные 120 FPS

Далее представлен сам ландшафт для сравнения влияния упрощения сетки (рис. 13 и рис. 14):

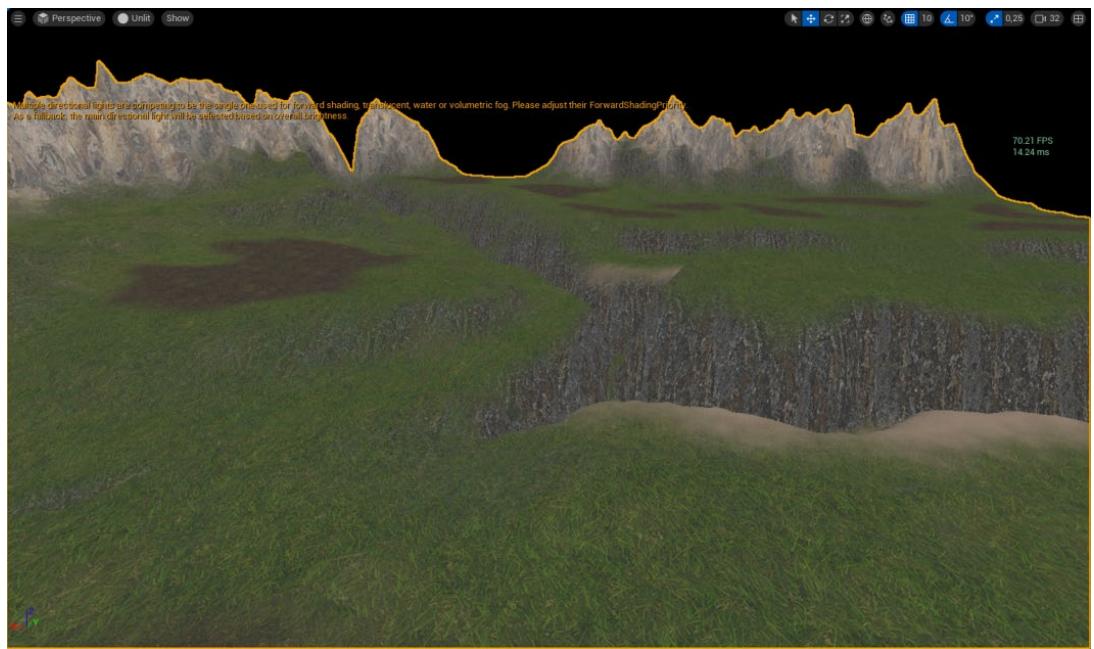


Рисунок 13. Ландшафт до оптимизации, 70 FPS

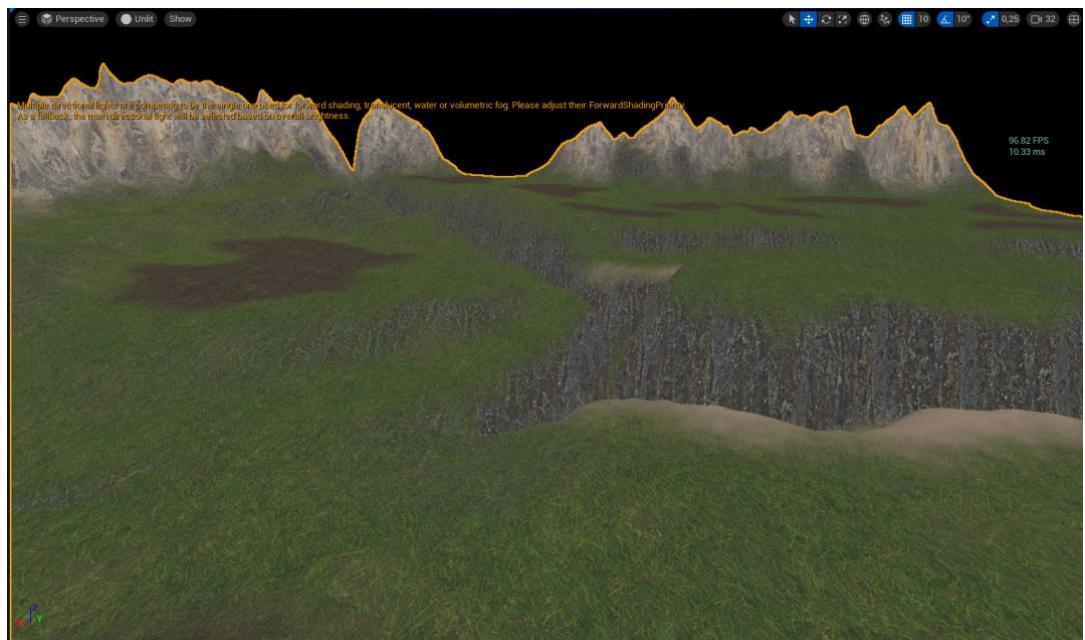


Рисунок 14. Ландшафт после оптимизации, 97 FPS

Как видно влияние на производительность существенная, при этом визуальное и функциональное качество [7] не теряется. Для наглядности ниже приведена таблица со сравнением производительности до и после оптимизации сетки в разных режимах (таблица 1):

Таблица 1. Параметры производительности после до и после оптимизации

№	Wireframe mode	Unlit mode	Lit mode

1	40→70	66→79	20→25
2	47→96	70→97	23→27
3	59→120	110→120	32→43
4	44→80	69→90	23→26

3. 3d-моделирование

3.1. Обзор интерфейса

Стандартный интерфейс Blender представляет из себя ряд интерактивных окон. По центру располагается главное окно взаимодействия – Viewport (рис. 15).

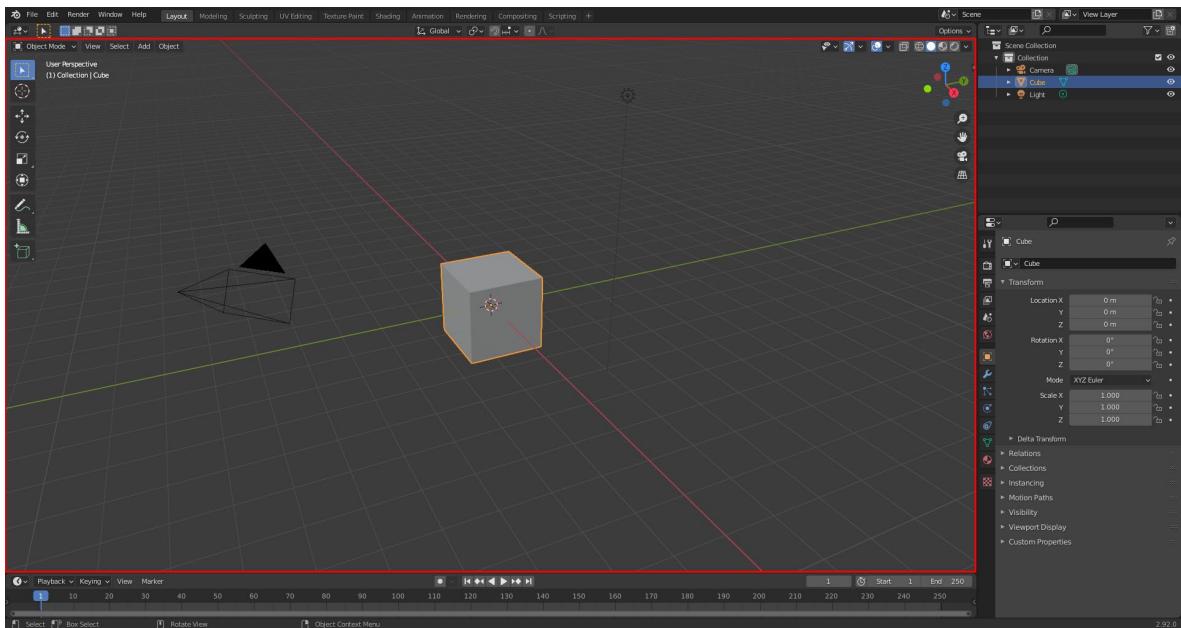


Рисунок 15. Окно Viewport Blender выделено красным

Данное окно предоставляет доступ к результату четырёх режимов отображения (рис. 16):

- 1) Wireframe – режим отображения сетки объектов, в которой видны вертексы и грани построения полигонов.
- 2) Solid – режим отображения твердотельных объектов без освещения и материалов.
- 3) Material Preview – режим отображения материалов, но без корректного рендеринга и освещения.
- 4) Rendered – режим отображения рендера во Viewport. Стоит отметить, что по умолчанию в данном режиме есть отличия с конечным рендером, вызываемом опцией Render → Render Image или по нажатию горячей клавиши F12, которые можно настроить в настройках Render Viewport.

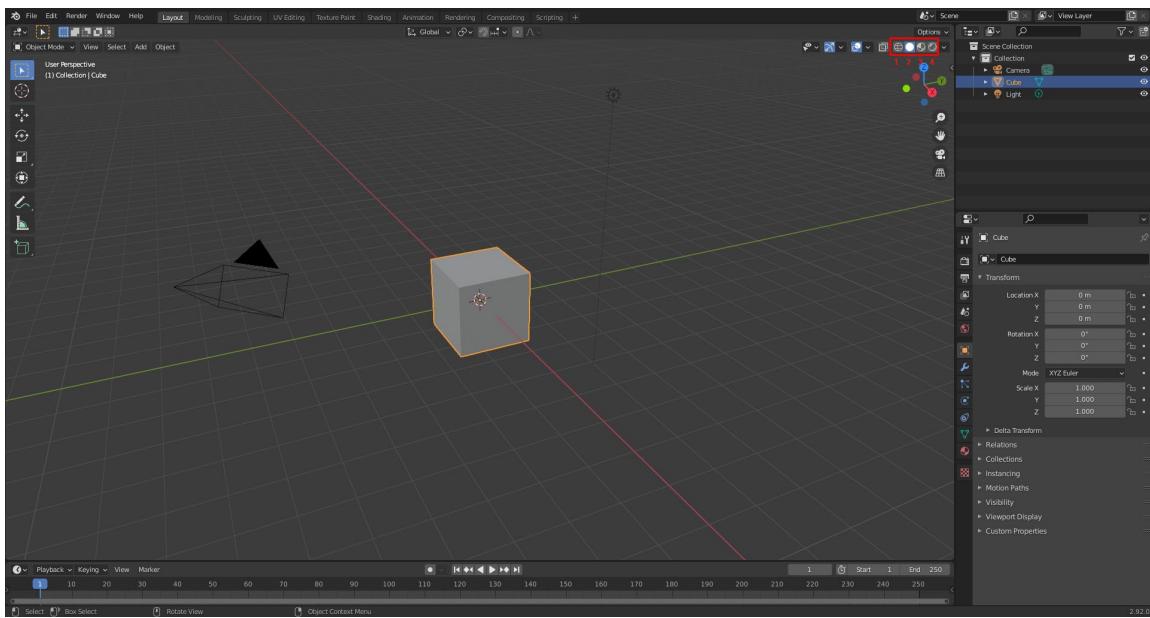


Рисунок 16. Режимы отображения, выделены красным

По умолчанию, помимо четырёх режимов отображения, можно переключаться между десятью режимами рабочей среды. Наиболее значимые из них:

- 1) Layout – стандартный режим взаимодействия с 3D-объектами. В данном режиме удобно создавать базовые геометрические формы и объекты, выделять, перемещать, масштабировать и настраивать группы существующих моделей.
- 2) Modeling – режим непосредственного моделирования сетки выделенных объектов.
- 3) Sculpting – режим «скульптуинга», позволяющий модифицировать сетку объектов по методам схожим с работой с глиной.
- 4) UV Editing – режим изменения и расположения UV-развёртки объектов, которая отвечает за то, какие полигоны будут получать или иные части накладываемой текстуры.
- 5) Texture Paint – режим создания текстур на самом объекте. В данном режиме представляется возможность рисовать, имитируя кисточку, добавлять смазывания и оттягивания имеющихся цветов текстуры, заливать области и накладывать маски.
- 6) Shading – режим настройки материала для выделенного объекта.

Из важных элементов также стоит отметить наличие навигационного окна в правом верхнем углу (рис. 17). Данное окно предоставляет доступ к объектам, их иерархии, некоторым свойствам, подсвеченным индикаторами и фильтрами, позволяющими влиять на отображение и взаимодействие с объектами, например, выключать их во Viewport, в Render или отключать возможность их выделения.

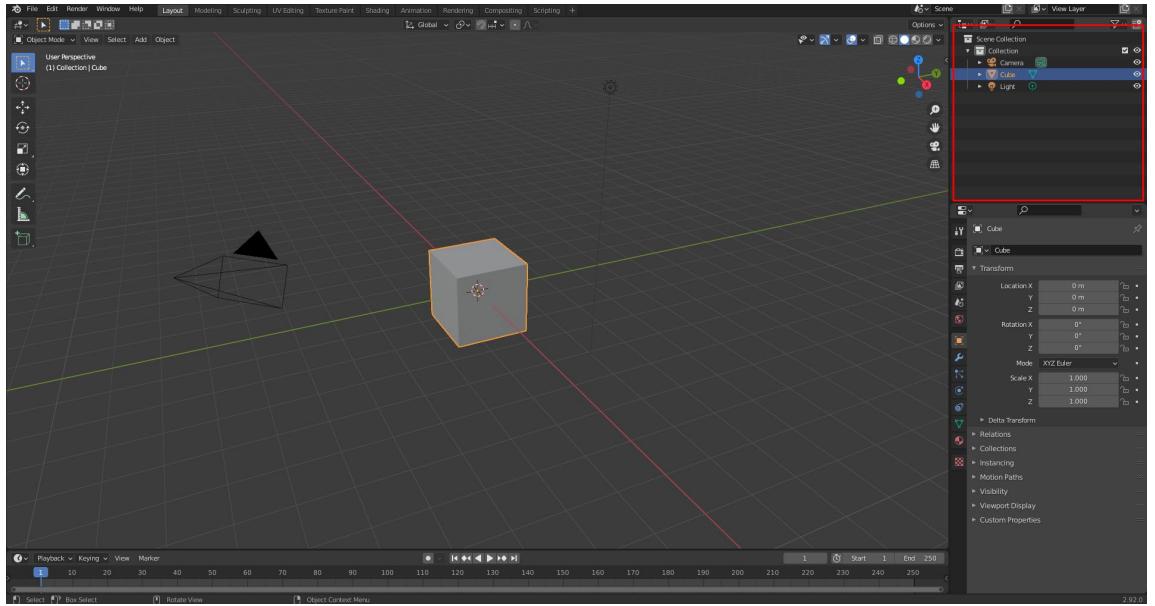


Рисунок 17. Окно навигации, выделено красным

3.2. Базовые методы моделирования

Предварительно для осуществления любого взаимодействия, требуется создать базовые объекты. В блендере по умолчанию для создания объектов можно использовать комбинацию клавиш «Shift + A». По нажатию данной комбинации на месте курсора всплывает окно выбора создаваемого объекта (рис. 18). Самые часто используемые объекты находятся в меню Mesh, Curve, Light и Camera.

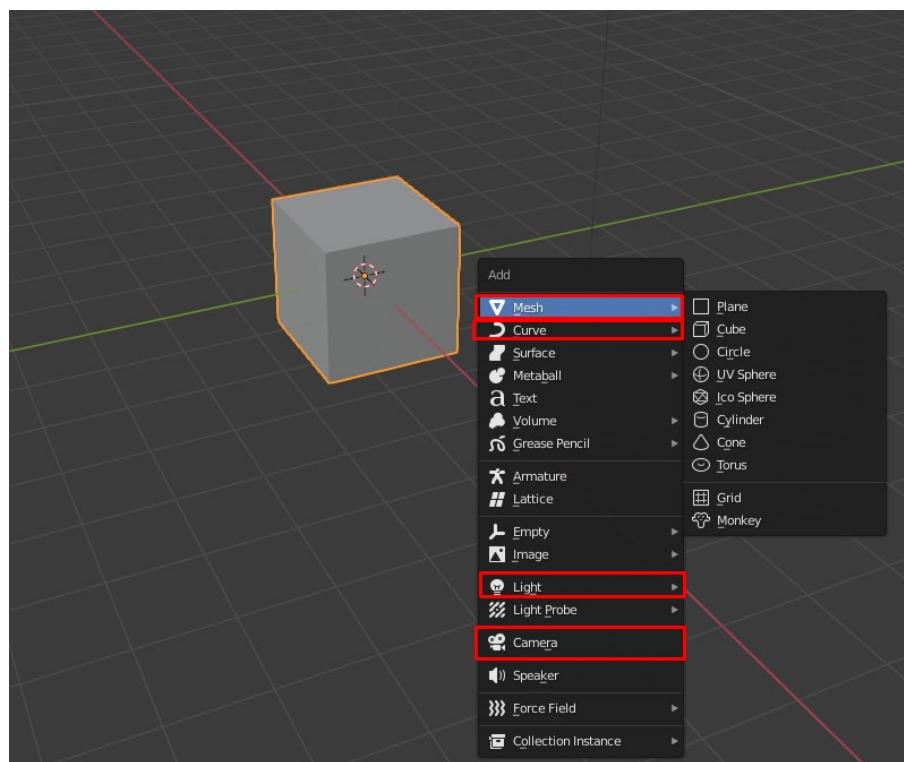


Рисунок 18. Меню добавления объектов

В меню Mesh пользователю предлагается выбрать из набора следующих простых геометрических объектов:

- 1) Plane – квадратная, односторонняя плоскость с четырьмя вертексами по краям. Стоит заметить, что по умолчанию данный объект на рендере будет отображаться лишь, с одной стороны.
- 2) Cube – объёмный куб с восемью вертексами по краям.
- 3) UV Sphere – сфера, по умолчанию состоящая из 32 вертикальных и 16 горизонтальных огибающих рёбер.
- 4) Cylinder – цилиндр, основания которого по умолчанию задаются как круги с 32 вертексами.

Остальные геометрические объекты представляют меньший интерес за счёт меньшего практического смысла. Также во время создания объектов в левом нижнем углу появляется вкладка меню последнего действия, позволяющая изменять параметры только что созданного объекта. Изменения включают в себя размер объекта в метрах, количество сегментов, расположение и вращение в пространстве и другие.

Меню Curve позволяет создавать кривые Безье, которые могут быть использованы для задания определённых линий для расположения объектов с помощью модификатора Align, а также для выдавливания их в объёмные трубы.

В меню Light доступен выбор четырёх типов источников света:

- 1) Point – точечный источник света, который имеет одну точку испускания света, яркость и направление.
- 2) Sun – источник имитирующий солнце, играющий роль большого объекта, все лучи которого падают на объекты под единым углом в независимости от местонахождения этих объектов на сцене.
- 3) Spot – источник света с определённым телесным углом.
- 4) Area – плоскость произвольной формы, одна сторона которой испускает свет.

Опция Camera позволяет добавлять камеры, через которые в последствии может осуществляться рендер сцены.

В дальнейшем созданные объекты можно изменять при помощи опций в панели слева или при помощи горячих клавиш. При нажатии на горячую клавишу Tab или при переходе в режим Modeling пользователю становится доступен функционал изменения сеток выделенных объектов как показано на рисунке 19.

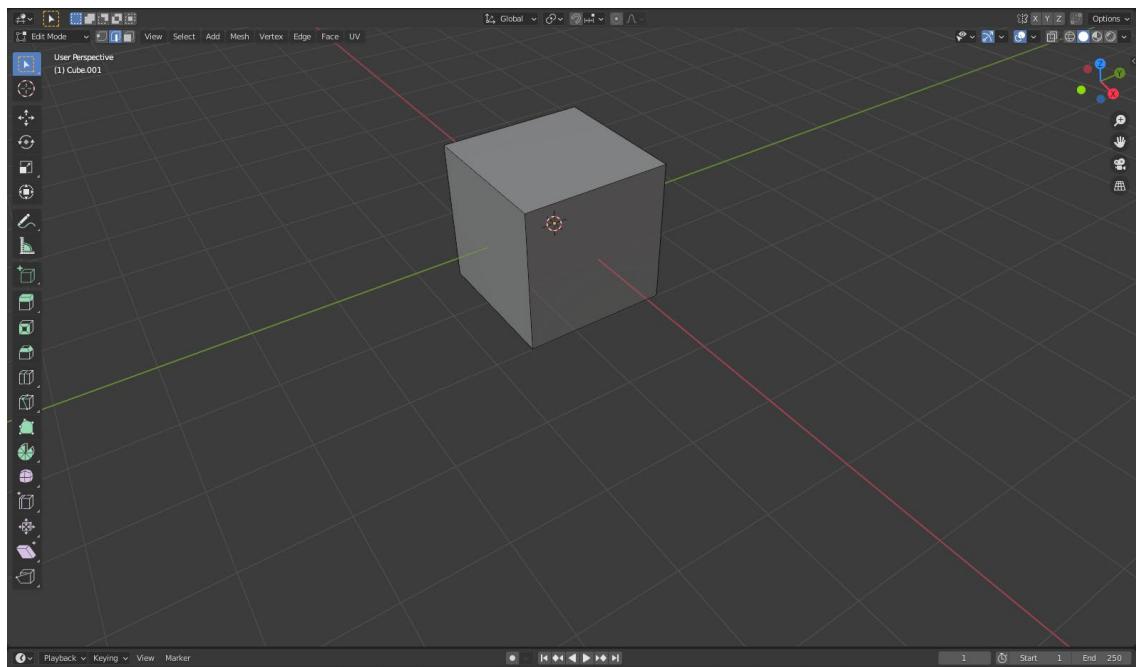


Рисунок 19. Окно Layout в режиме редактирования сетки

В данном режиме в ранее выделенных объектах можно выбирать отдельные вертексы, грани или рёбра. Переключения между режимами выделения подобъектов доступно в панели в левом верхнем углу или по нажатию клавиш «1», «2» или «3» соответственно. При этом после перехода в режим редактирования прежние функции масштабирования и перемещения теперь применимы и к выделенным подобъектам.

В этом режиме удобно пользоваться функцией X-Ray, дающую сквозную видимость через объект. Данная функция позволяет выделять подобъекты тела, скрытые за его гранями без необходимости вращаться вокруг тела.

Среди наиболее часто применяемых модификаций стоит выделить функции масштабирования на клавишу «S», перемещения на клавишу «G», выдавливание объектов на клавишу «E» и удаление объектов на клавишу «X». Более продвинутые функции Inset на клавишу «I», функция Bevel на клавишу «B» и функция Loop Cut на комбинацию клавиш «Ctrl + R» позволяют создавать на выделенном ребре отмасштабированное внутреннее ребро, фаску или набор граней по периметру соответственно.

3.3. Методы полигонального моделирования объектов

Используя вышеупомянутые представляется возможным создать базовый «блокинг» сцены «Фабрика». Под «блокингом» подразумевается первоначальное расположение геометрических объектов, схожих по форме с конечным вариантом в сцене (рис. 20).

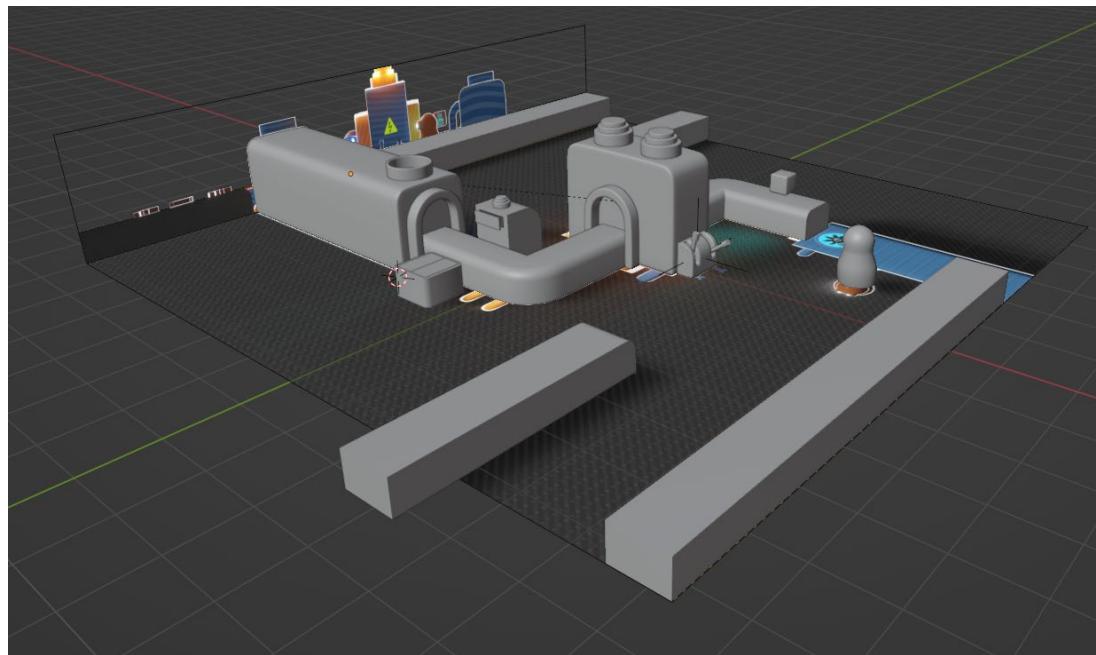


Рисунок 20. «Блокинг» сцены «Фабрика»

Создание арок требует добавление окружности, удаление нижнего полукруга и выдавливания нижних вертексов перпендикулярно вниз. Создание корпуса робота потребовало добавления UV-сферы и последовательного удаления, добавления и масштабирования нижних подобъектов. Для создания конвейера потребовалось последовательное выдавливание передней грани куба и добавление фаски на всём протяжении по бокам.

Для остальных элементов необходимо использование настройки модификаторов. Окно модификаторов указано на рисунке 21.

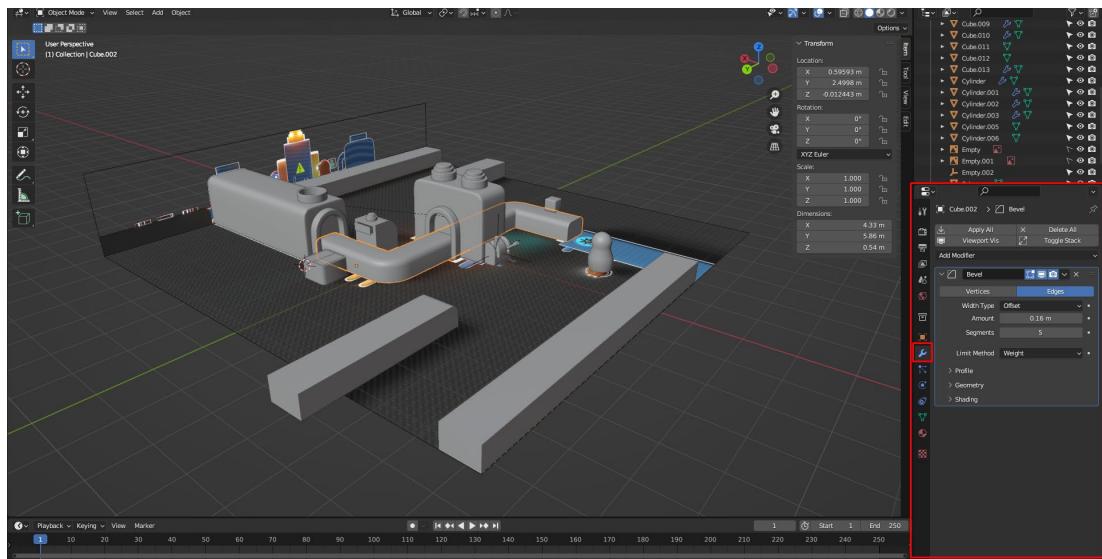


Рисунок 21. Окно модификаторов, выделено красным

В данном окне возможно присвоение объектам различного набора из списка доступных модификаторов. Одними из главных преимуществ модификаторов являются строгая иерархия применения по их взаиморасположению и тот факт, что модификаторы не влияют непосредственно на конечную сетку объекта, до тех пор, пока они не были применены и их эффекты не были перенесены на сетку.

Для создания фасок на объектах необходимо добавить модификатор **Bevel**, который имеет опции задания размеров фаски и количества сегментов, образуемых при построении. Также при создании некоторых объектов на данном этапе могут понадобится модификаторы **Solidify** добавления объёма фигуре перпендикулярно её нормалям, **Mirror** отражения сетки объекта вокруг определённых осей с центром в указанной точке и модификатор **Subdivision Surface**, делающий более дробную сетку объекта, для того чтобы сглаживание объектов производилось более плавно.

На втором этапе работы была добавлена конвейерная лента, винты и созданы проёмы для арок (рис. 22).

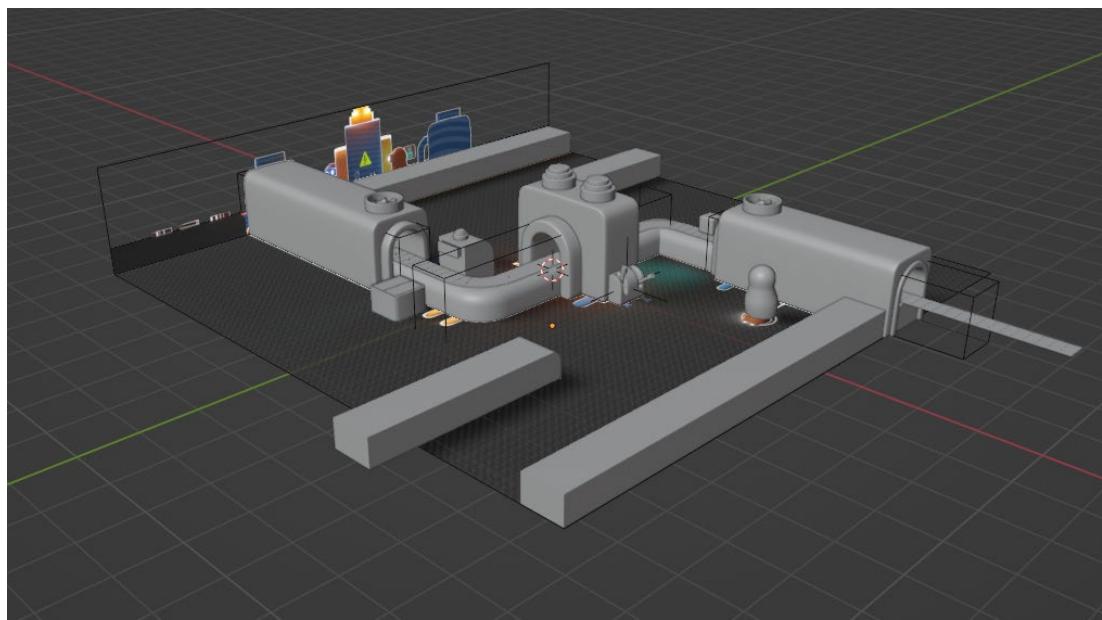


Рисунок 22. Сцена «Фабрика» на втором этапе моделирования

Для создания направления ленты была создана единая прямоугольная пластина и на её основе создан массив элементов при помощи модификатора Array. Направление для ленты задаётся посредством модификатора Curve, берущего за основу существующую кривую и располагающую объекты вдоль осей выбранной кривой, которая была создана на основе ранее созданного конвейера.

Для создания вентиляторов была создана сфера и из неё получен окружный стержень. Листы винтов были созданы при помощи добавления плоскости Plane и затем изменены через модификаторы Subdivision Surface и Simple Deform, позволяющий искривлять объект вокруг одной осей. Затем данные листы были размножены копированием и закреплены модификатором Align к центру стержня. Результат создания объекта вентилятора на рисунке 23.

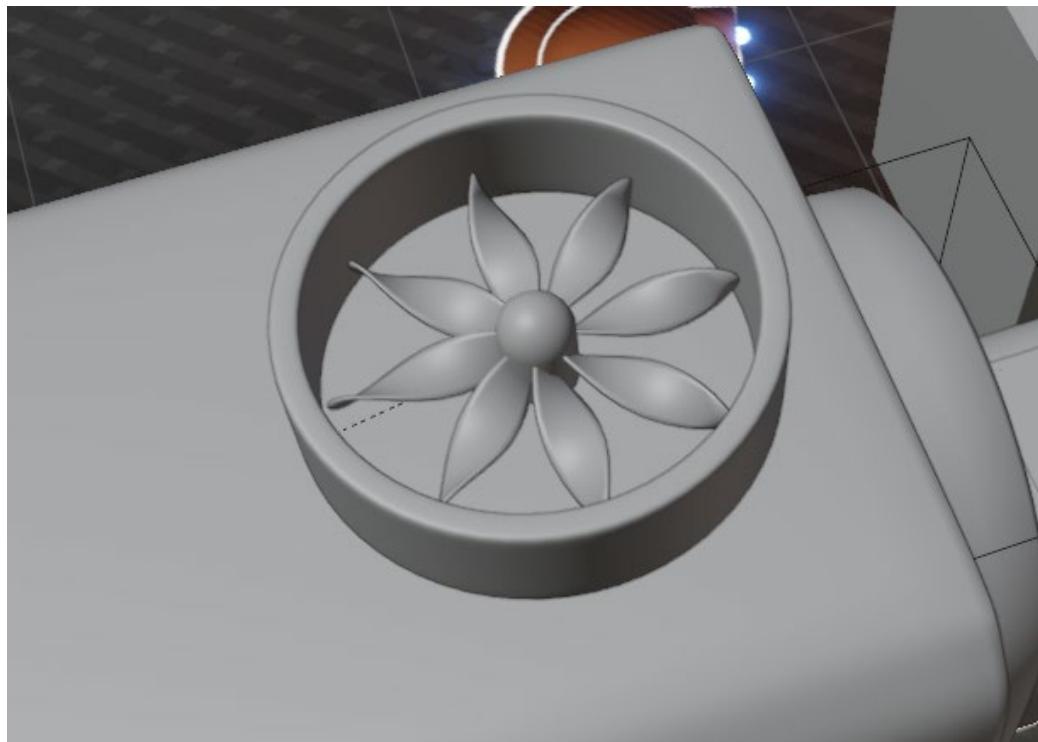


Рисунок 23. Объект вентилятор

Для создания отверстий был использован модификатор Boolean. Данный модификатор позволяет совершать взаимодействия между двумя объектами – объектом который использует модификатор и объектом, который указан в настройках данного модификатора. У этого модификатора присутствует 3 режима работы. Intersect оставляет лишь пересечение двух объектов, Union совмещает два объекта, Difference выполняет вычитание одного объекта из другого, при этом второй объект перестаёт быть твердотельным и оставляет после себя лишь остов для возможности его перемещения и изменения конечного объекта вычитания. Последний режим был применён для создания арок при помощи вычитания скопированных арок.

На третьем этапе были созданы трубы и провод зарядки для робота путём создания кривой и добавления модификатора Solidify. Маска робота и глаза с использованием уже известных методов изменения сетки объектов. Также был добавлен элемент «Лайк» над конвейером, созданный путём выдавливания и соединения вертексов по контуру объекта (рис. 24).

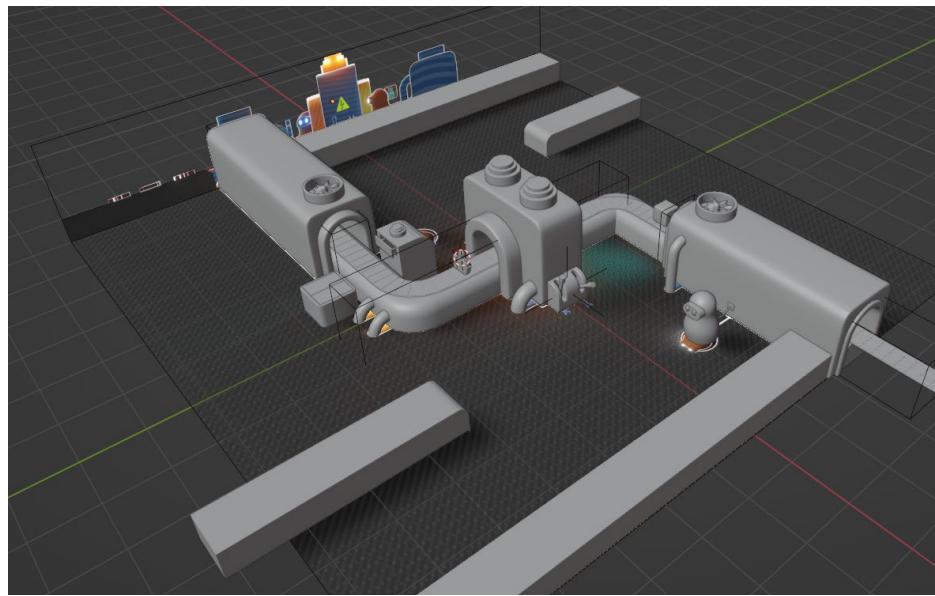


Рисунок 24. Сцена «Фабрика» на третьем этапе моделирования

На четвёртом этапе было добавлено освещение без материалов. Внутри арок были добавлены объекты Area Light, также был создан источник света Sun над сценой. Также была добавлена камера для корректировки угла обзора сцены для корректного построения освещения с учётом требуемого результата сцены «Фабрика». Освещение сцены производится только в режиме отображения Render (рис. 25).

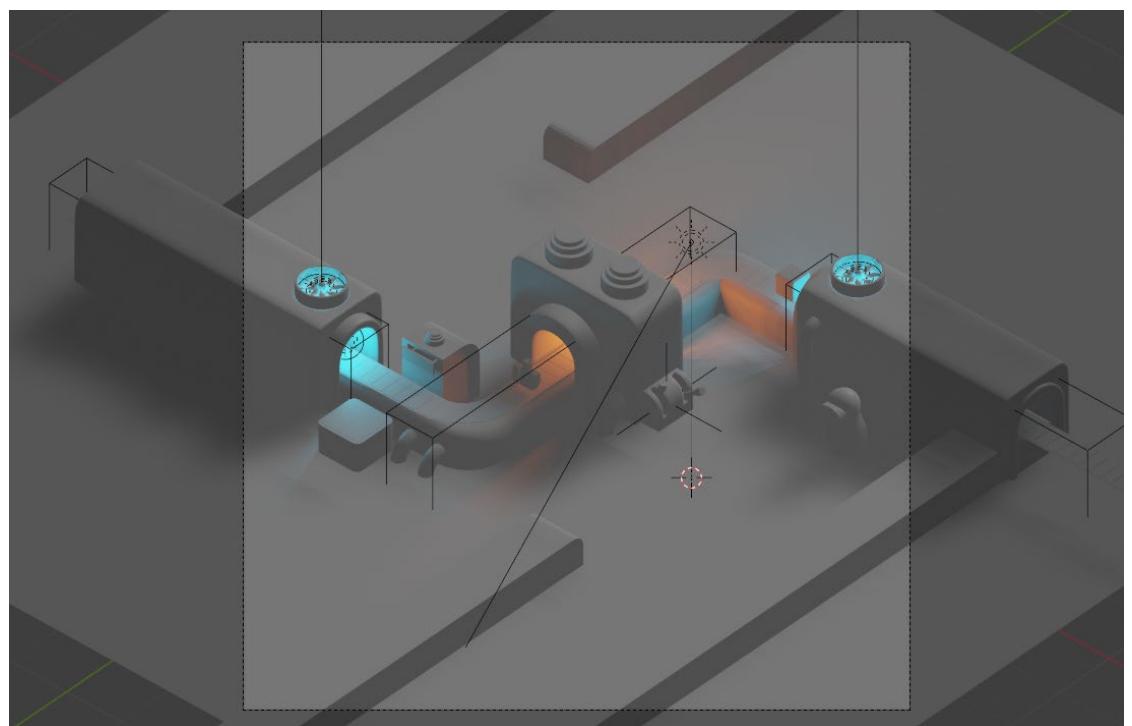


Рисунок 25. Сцена «Фабрика» на четвёртом этапе моделирования

Стоит отметить что обзор через установленную активную камеру осуществляется нажатием клавиши «0» на NumPad. При этом камера даёт ортогональный обзор.

Последним пятым этапом объектам сцены добавляются материалы и текстуры. Редактирование текстур осуществляется в меню Shading. После создания нового материала для тела появляется доступ к его свойствам (рис. 26).

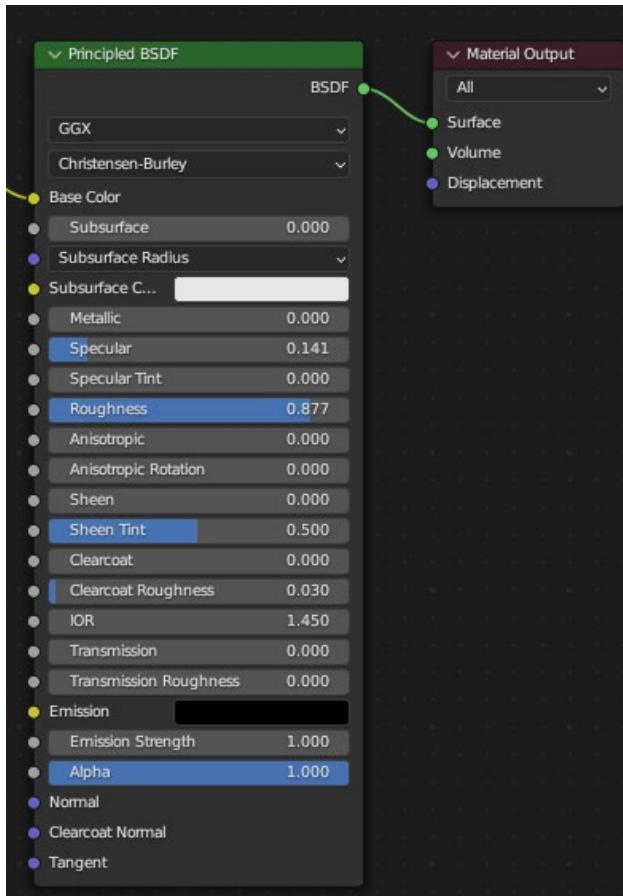


Рисунок 26. Свойства материала в меню Shading

Наиболее важные параметры:

- 1) Base Color – отвечает за цвет объекта.
- 2) Metallic – отражает металличность объекта, позволяя имитировать свойства отражения света материалов из металла.
- 3) Specular – отвечает за блики.
- 4) Roughness – отвечает за грубоность поверхности объекта. Низкие значения сделают объект идеально гладким на вид, позволяя отражать больше света.
- 5) Emission – цвет эмиссионного света материала.

6) Emission Strength – сила эмиссии.

Для создания простых материалов достаточно указать определённый цвет и параметры отражения света.

Для того чтобы добавить текстуру на объект достаточно перенести изображение текстуры в меню снизу и подключить его к узлу Base Color. Далее текстуре требуется назначить способ, по которому текстура накладывается на объект. Для этого создаётся панель Mapping и Texture Coordinate. Конечный результат создания текстуры для куба представлен на рисунке 27.

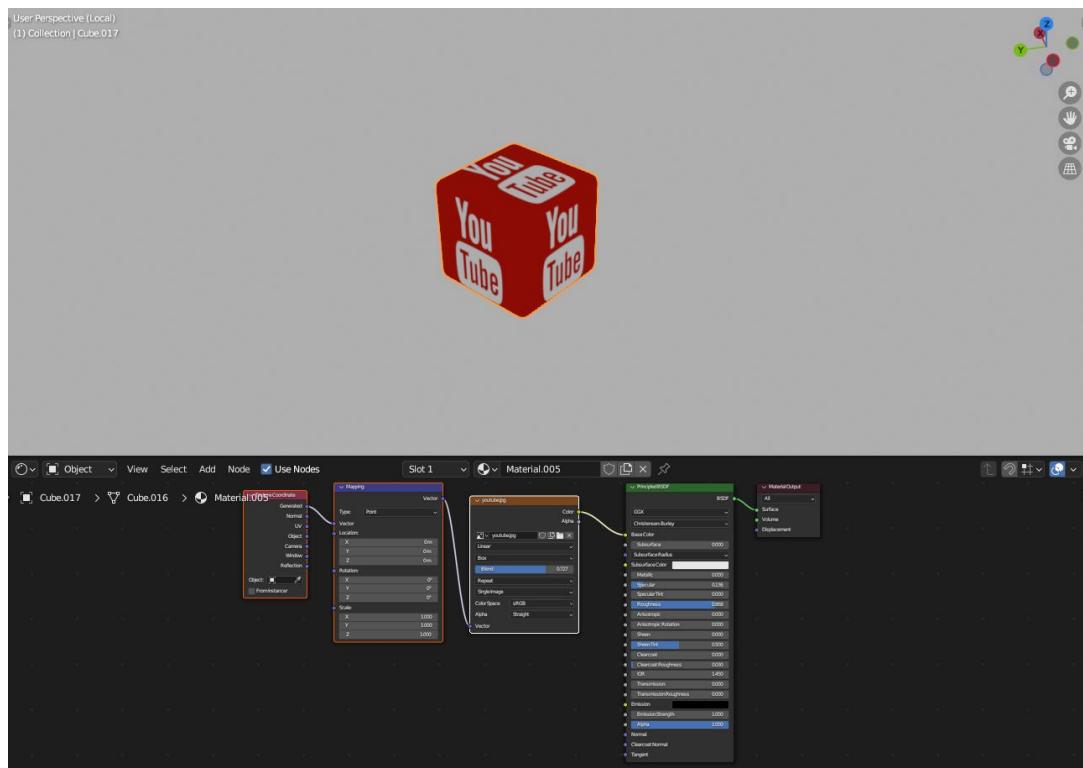


Рисунок 27. Результат создания материала для куба «YouTube»

Для создания материалов для робота требуется использования масок и эмиссии света. Поскольку робот в процессе работы был преобразован в единый объект, для того чтобы добавить на него части разные цвета необходимо воспользоваться картами масок. Глаза робота не были преобразованы в подобъект робота и поэтому им достаточно задать параметры Emission. Для светящихся цилиндров был применён подход создания различных материалов и определения для них разных наборов вертексов тела. Один набор вертексов получил простой материал, имеющий оранжевый цвет, а другой – чёрный

материал, излучающий жёлтый свет. Ниже представлены окончательный рендер данной сцены в Blender (рис. 28).



Рисунок 28. Окончательный рендер сцены «Фабрика» в Blender

3.4. Методы «скульптинга»

Для создания сцены «Енот» упомянутые выше методы не подойдут поскольку форма объекта не представляет собой подобие стандартной геометрической фигуры. Для создания подобных объектов в Blender существует режим Sculpting.

Скульптинг – это процесс 3D-моделирования, при котором используется инструменты воздействия на сетку объекта с помощью инструментов схожих с теми что используются при обработке глиняных скульптур.

Первоначально для моделирования были созданы несколько сфер для головы, тела, ушей, глаз, носа и лап енота. Далее данные сферы были растянуты в более схожие с результатом формы. Также был создан куб для коробки и цилиндр для хвоста. Для их построения можно использовать инструменты упомянутые в прошлой секции. Хвост был создан при помощи добавления цилиндра и последующего выдавливания масштабирования передней грани цилиндра. Коробка была создана путём масштабирования сторон, удаление верхней грани и вытягивания боковых граней в стороны. После придания примитивным объектам подходящей формы имеет смысл переходить к непосредственно к «скульптингу».

Основными инструментами «скульптинга» в процессе работы были функции:

- 1) Draw Sharp – тонкая, острые кисть позволяющая наращивать либо вдавливать материал в месте нажатия.
- 2) Clay Strips – инструмент схожий с наложением материала шпателем. При нажатии на объект наращивает либо вдавливает слои в месте нажатия.
- 3) Smooth – инструмент сглаживания поверхности.
- 4) Pinch – инструмент сжатия соседних объектов в центр места нажатия.
- 5) Grab – инструмент позволяющий тянуть за объект для добавления деформаций по направлению.

Стоит заметить, что для удобной работы с инструментами «скульптинга» необходимо создать более дробную сетку для объекта, поскольку данные инструменты могут работать некорректно при разряженной сетки и создавать слишком острые изменения. По этой причине при переходе в режим Sculpting требуется перейти во вкладку Remesh и уменьшить размер вокселей, а затем нажать кнопку Remesh.

При помощи данных инструментов были добавлены углубления в местах для ушей и рта, наращены массы в местах бровей, вытянуты лапы и наращены плечи при помощи инструмента Clay Strips. Результат первоначального «скульптинга» представлен ниже на рисунке 29.



Рисунок 29. Первый этап моделирования сцены «Енот»

Далее отельные объекты ушей, торса и лап были совмещены с исходным объектом головы и сглажены на стыках при помощи инструмента Smooth. Были проделаны углубления для носа, при помощи инструмента Draw Sharp были созданы дуги вокруг глаз и более острая обводка рта (рис. 30).



Рисунок 30. Второй этап моделирования сцены «Енот»

Как упоминалось ранее, для удобной работы с инструментами «скульптинга» необходимо увеличение числа элементов в сетке объекта. Ниже на рисунке 31 представлена сетка объекта после преобразований.

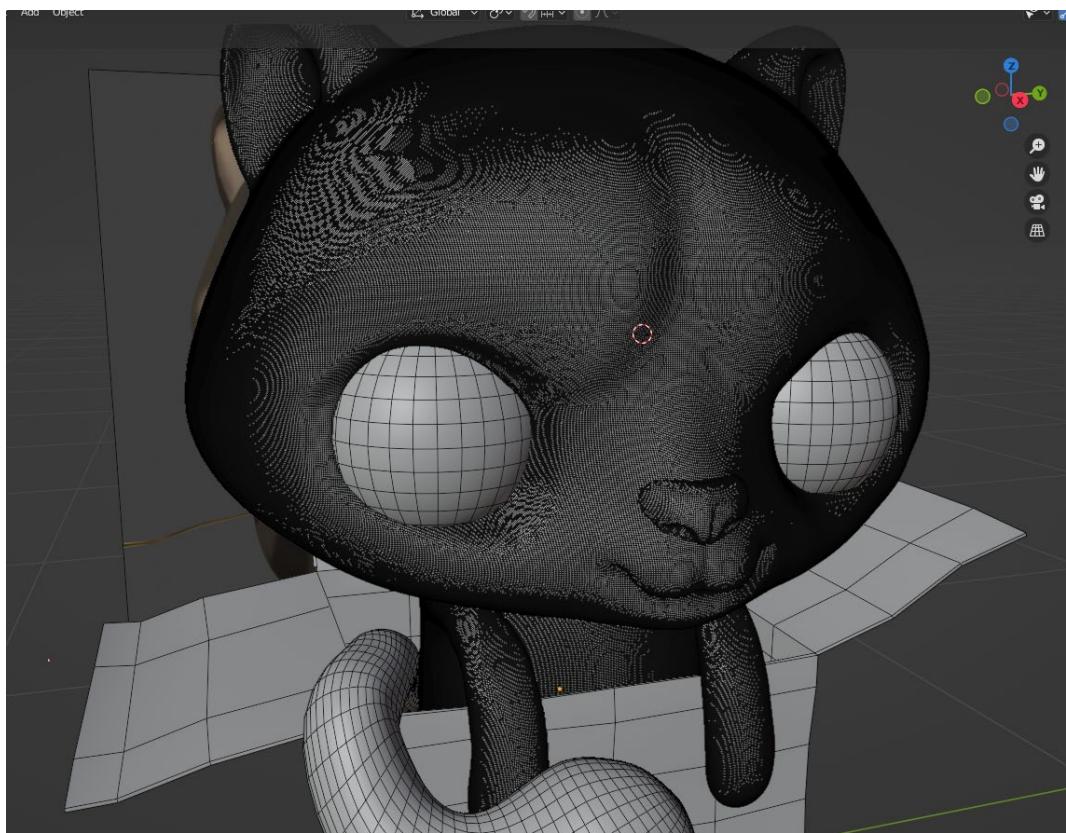


Рисунок 31. Сетка объекта до применения «запечки»

При этом серьёзно страдает производительность обработки и рендера объекта. Чтобы избежать проблем с производительностью можно воспользоваться функцией «запечки» рельефа объекта в определённую текстуру, карты рельефа и нормалей которой можно будет использовать для имитации детального объекта на менее полигональной, упрощённой модели. После применения метода «запечки» и применения полученных текстур на новой модели были получены результаты, указанные на рисунке 32 и рисунке 33:

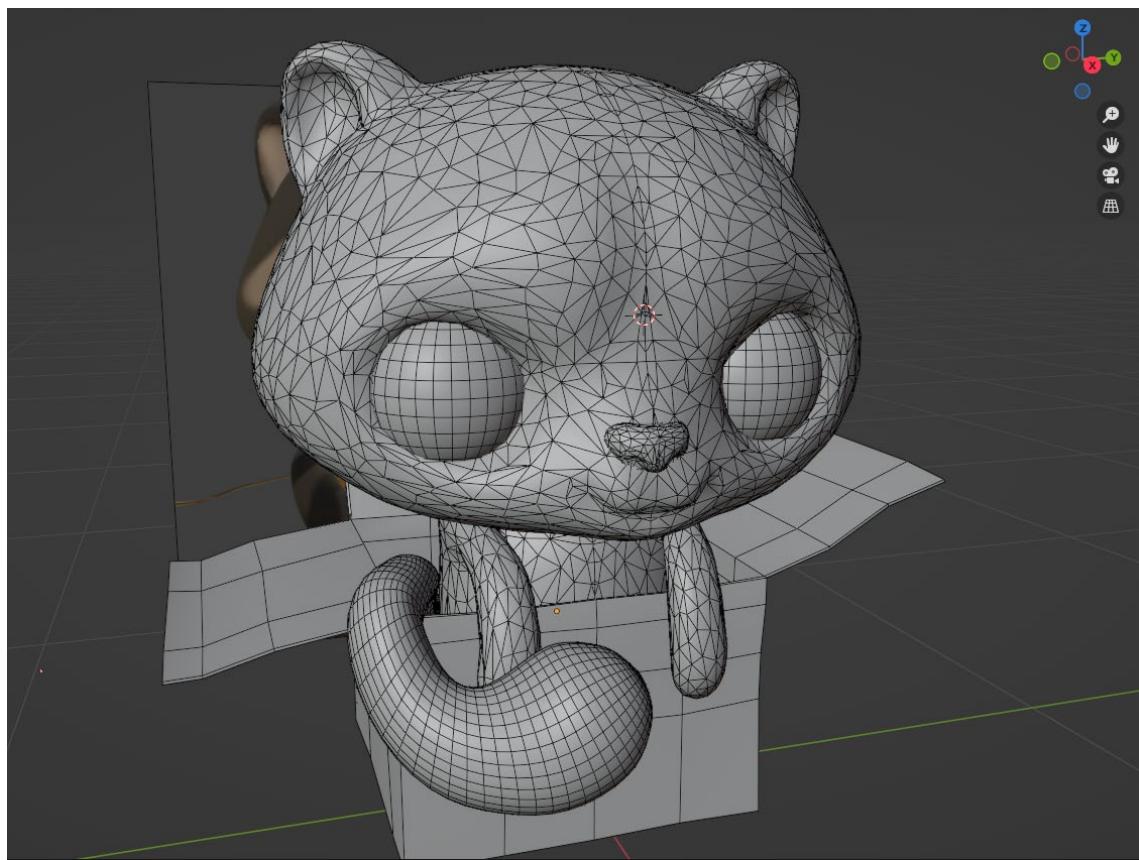


Рисунок 32. Сетка объекта после применения «запечки»



Рисунок 33. Вид объекта после применения «запечки»

Далее во вкладке UV Editing была автоматически получена UV-сетка объекта при помощи функции UV → Smart UV project, а затем произведена функция unwrap для расположения сетки на плоскости для дальнейшего наложения текстур.

После создания UV развёртки можно создать материал. Далее используя кисти во вкладке Texture Paint можно покрасить енота в необходимые цвета, а затем присвоить материалу необходимые параметры взаимодействия со светом во вкладке Shading. Для добавления текстур на коробку во вкладке Texture Paint существует способ покраски через трафарет, дающий возможность добавлять одинаковые рисунки отпечатков лап на объекте (рис. 34).

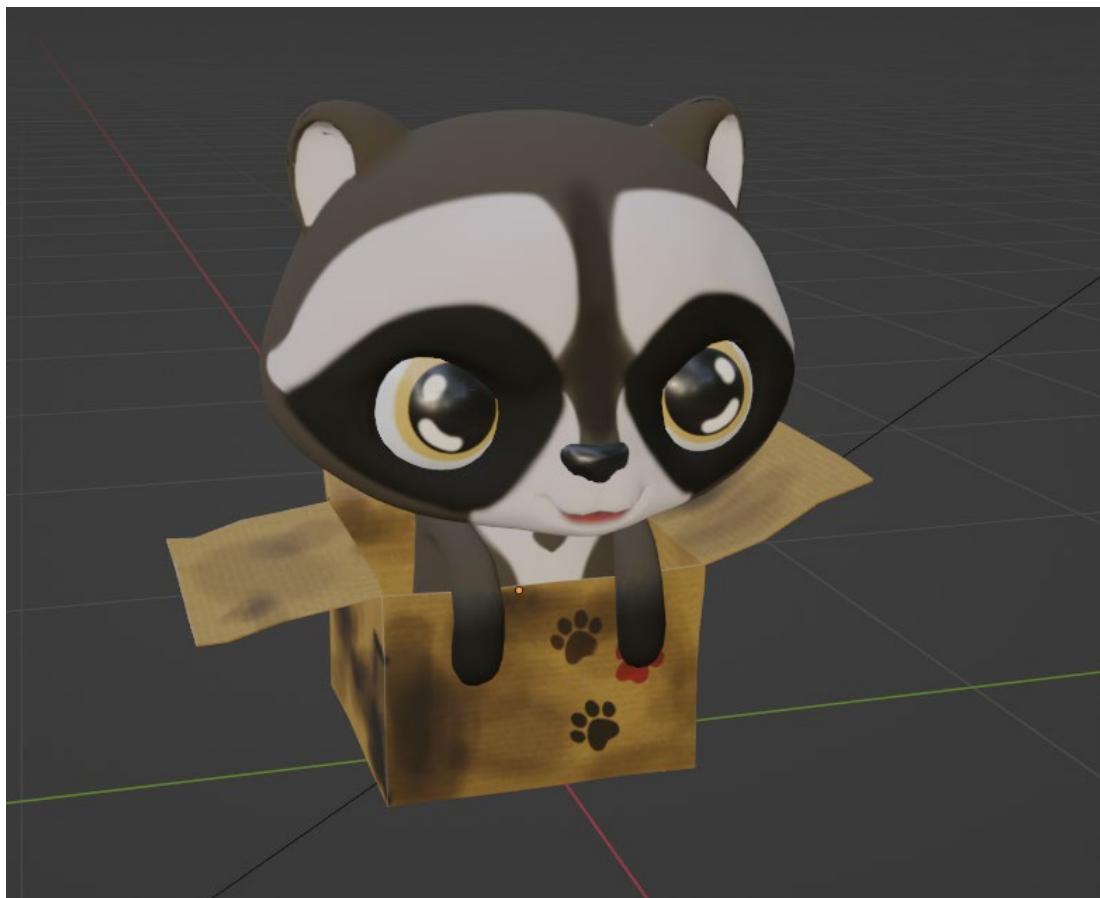


Рисунок 34. Сцена «Енот» после наложения материалов и текстур

Далее необходимо преступить к созданию шерсти. Шерсть в Blender создаётся при помощи добавления Particle Setting в меню Particles, доступном в правом нижнем углу. После выбора опции Hair программа создаёт на всём объекте отдельные элементы похожие на стержни. В настройках можно изменять

различные параметры, отвечающие за количество элементов, их скученность, «грязноту», длину, кривизну и многие другие.

Для того чтобы выделять определённые зоны в которых будут появляться элементы волос необходимо переключится на вкладку Weight Paint в левом верхнем углу вместо Object Mode и затем при помощи кисти добавить или удалить области распределения. После добавления нескольких зон и присвоения им различных систем волос для имитации поведения разнородной шерсти можно приступить к укладыванию волос по направлению. Для этого требуется перейти во вкладку Particle Edit в левом верхнем углу. Данный режим предоставляет доступ к инструментам схожим с причёсыванием, позволяя укладывать или расpusшать, удалять или добавлять, удлинять или укорачивать шерсть в месте нажатия. После применения данных инструментов был получен конечный результат. Более того в сцену было добавлено несколько вспомогательных объектов и освещение схожее с требуемым референсом (рис. 35).



Рисунок 35. Сцена «Енот» после окончательного рендера в Blender

3.5. Экспорт и импорт

Для переноса файлов из Blender в Unreal Engine 5 существует множество методов со своими плюсами и минусами.

Первый метод заключается в экспорте файлов в формате FBX. Преимущество данного метода состоит в возможности точной настройки элементов экспорта и настройке качества передачи данных. Основным минусом является невозможность передачи объектов освещения, некоторых внутренних настроек объектов и неспособность переносить модификаторы. Последнее может быть несущественно поскольку после воссоздание конечной сцены в Blender все модификаторы можно без осложнений применить к геометрии и воспользоваться упомянутым экспортом.

Второй метод использует файлы glTF2.0. Данный формат имеет меньшее число опций, но отличается тем что экспортирует данные в сыром формате, который хранит больше информации и позволяет переносить элементы освещения и более корректно устанавливать материалы для объектов.

Третий способ передачи данных состоит в установке взаимной связи между приложениями Blender и Unreal Engine 5 при помощи плагина Send2UE. Данный плагин устанавливается для Blender и открывает доступ к меню Pipeline, в котором можно настроить пути и способы передачи данных из Blender. Для корректной работы со стороны Unreal Engine 5 требуется открыть доступ удалённого изменения в настройках проекта и добавить несколько плагинов взаимодействия с Python, которые установлены по умолчанию в новых версиях Unreal Engine 5.

После создания базовой сцены и импорта полученных материалов в Unreal Engine 5 были получены конечные сцены рендера в среде Unreal Engine 5. Результаты переноса сцены «Фабрика» и сцены «Енот» представлены на рисунках 36 и 37 соответственно.

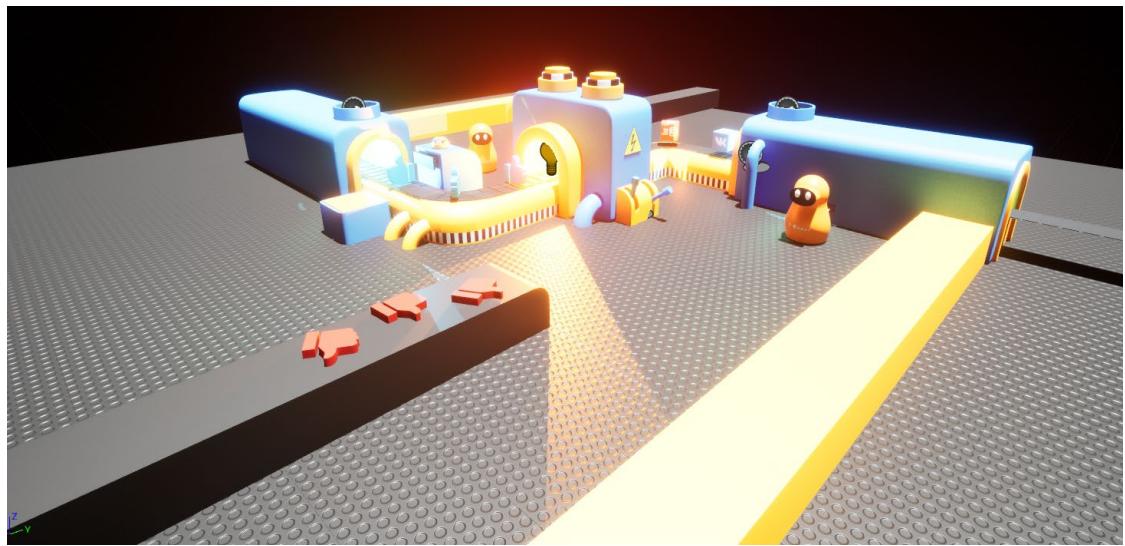


Рисунок 36. Рендер сцены «Фабрика» в Unreal Engine 5



Рисунок 37. Рендер сцены «Енот» в Unreal Engine 5

Стоит отметить что даже при корректном экспорте всех материалов из среды в Blender в Unreal Engine 5 сцена не будет выглядеть идентично в обоих программах. Это связано как с внутренними механизмами рендеринга [9], так и с разным отношением данных программ к различным элементам, например, к шерсти [10].

4. Создание элементов RPG окружения

4.1. Добавление растительности и воды

Для водных препятствий в Unreal Engine существует большой набор плагинов. Для создания воды рек и моря был использован стандартный плагин Water, при помощи которого можно поместить плоскость WaterBodyCustom, имитирующую поверхность воды.

Для создания воды в озёрах были созданы собственные материалы в которых были реализованы возможности по отражению света, перемещению водной глади и имитации прозрачности на глубине. После создания класса Blueprint для материала было создано два алгоритма перемещения двух волн разных по величине, направлению и скорости для предотвращения регулярности (рис. 38). Параметры Water1 и Water2 отвечают за величину волн, XSpeed1, XSpeed2 – за скорость волн по направлению X, YSpeed1, YSpeed2 – за скорость волн по направлению Y.

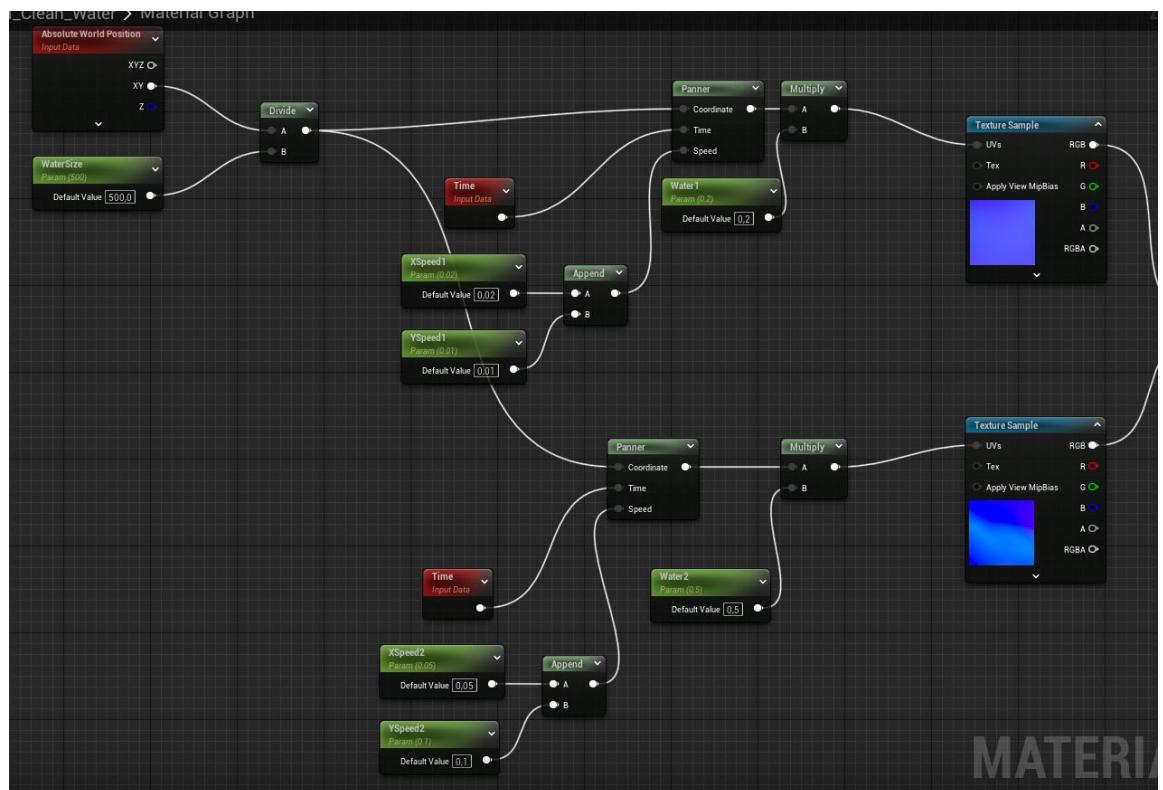


Рисунок 38. Фрагмент схемы Blueprint реализации хождения волн

После этого добавляются параметры DepthOpacity и FadeDistance, которые затем передаются в узел DepthFade позволяющий установить атрибут

прозрачности воды (рис. 39). Характеристики отражаемости и жёсткости также передаются в виде параметров.

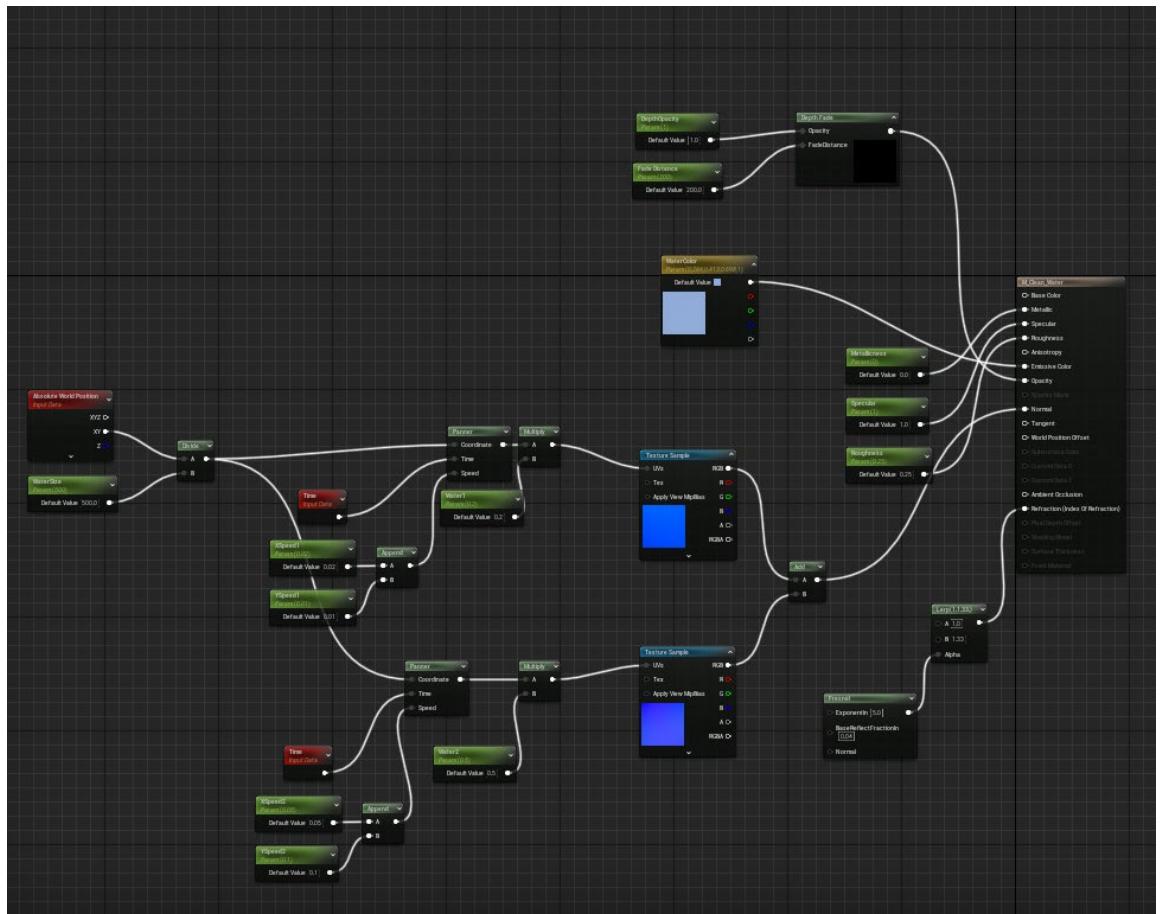


Рисунок 39. Конечная схема Blueprint материалов воды

Стоит также отметить что при установлении какой-либо константы в виде параметра её значение может быть в дальнейшем изменено в дочерних инстанциях материала, не требуя изменения самого материала для точной настройки конкретного объекта общего типа.

В качестве объекта, использующего материал воды, наилучшим образом подходит куб или плоскость. Далее для четырёх внешних озёр были наложены плоскости с одним материалом воды, в то время как для центрального озера был создана собственная инстанция и определены иные параметры для выделения из числа прочих. Результат добавления озёр (рис. 40):



Рисунок 40. Ландшафт после добавления водных объектов

Для добавления растительности в среде Unreal Engine 5 существует собственный режим Foliage, позволяющий насыщать местность любыми объектами растительности, от травы до деревьев. Добавить подобные объекты можно с помощью экспорта из сторонних программных продуктов, позволяющих создавать статическую сетку растительности и прочую информацию необходимую движку, или с помощью существующих наборов и плагинов. Ранее упомянутая библиотека Quixel Megascans обладает рядом наборов доступных для свободного использования, однако для поставленной задачи было принято использовать менее качественные объекты в угоду производительности.

Для помещения растительности необходимо во вкладке Foliage перетащить имеющиеся статические сетки растительности в требуемое поле, а затем настроить масштаб объектов и насыщенность их посадки. Добавление происходит в режиме кисти позволяя в реальном времени добавлять регулируемое количество деревьев необходимых для выполнения задания.

После установки всех необходимых объектов были отключены динамические тени для улучшения производительности (рис. 41).



Рисунок 41. Ландшафт после добавления растительности

4.2. Создание сетки NavMesh

Для создания сетки в среде движка Unreal существует класс объектов NavMesh [1]. Для выполнения задачи создания сетки навигации необходимыми являются актёры NavMeshBoundsVolume, отвечающий за области создания сетки, и NavMeshModifies, позволяющий модифицировать или удалять определённые области существующей сетки [2].

При добавлении актёра NavMeshBoundsVolume автоматически добавляется его менеджер – RecastNavMesh-Default, в котором имеются все возможности задания параметров, по которым генерируется сетка, указывая предельную высоту двух соседних ячеек сетки, плавность и прочее.

Актёр NavMeshModifies по умолчанию удаляет область сетки проходимости вокруг себя, что удобно использовать для ограничения доступа к объектам типа озёр, леса или определённым местам.

Также существует актёр NavLinkProxy, позволяющий другим движущимся актёрам перепрыгивать между двух указанных точек связи, что может быть реализовано в создании моста для перехода.

Результаты создания и отображения навигационной сетки ландшафта представлены ниже на рисунке 42.



Рисунок 42. Результаты создания NavMesh сетки навигации, выделено зелёным

4.3. Создание RPG камеры

Основными параметрами, характеризующими камеру для проектов типа RPG, являются:

- Фиксированный угол обзора;
- Сохранение расстояния до ближайшего объекта при перемещении;
- Возможность удаления и приближения камеры в определённых границах;
- Возможность вращения вокруг точки наблюдения;
- Возможность перемещения камеры при помощи курсора или нажатия клавиш.

Для создания RPG камеры необходимо создать базовую интерактивную единицу – Pawn, которые по умолчанию не имеет визуального представления и функционала, однако может быть модифицирован и использован различными системами движка. Затем в этот объект добавляется камера [12] на удалении и под необходимым углом, который фиксируется через модифицируемый компонент SpringArm. Стоит отметить что при перемещении камеры точка отсчёта объекта ведётся относительно начала координат, что позволит при включении коллизии с поверхностями получать информацию об удалении камеры от поверхности, так как точка наблюдения находится непосредственно на ней.

Для реализации сохранения расстояния до ближайшего объекта и возможности изменения этого расстояния необходимо написать Blueprint скрипт, который способен оценивать текущее расстояние камеры до точки начала координат внутри Pawn объекта. Затем необходимо предоставить этому скрипту допуск к событиям прокрутки колесика мыши при помощи событий IA_Zoom, который реагирует на прокрутку колёсика мыши и затем модифицирует длину объекта SpringArm, увеличивая или уменьшая его длину, в ограниченных параметрах.

Для вращения вокруг точки наблюдения были создано событие IA_Rotate, которое в ответ на сигналы «+» и «-» изменяет угол поворота SpringArm относительно точки наблюдения.

Для добавления перемещения были использованы две системы. Первая на основе событий IA_XAxis и IA_YAxis воспринимает команды «W» и «S», а также «A» и «D» для перемещения назад-вперёд и влево-вправо соответственно. При этом изменяется положения SpringArm вместе с удерживаемой камерой в системе Pawn. Вторая система построена на отслеживании положения курсора мыши пользователя. В случае нахождения курсора вблизи границ экрана, программой реализован просчёт скорости необходимого перемещения в соответствующую сторону в зависимости от того, насколько курсор близок к краю.

По итогам была получена настроенная RPG камера с функцией плавного перемещения при помощи передвижения указателя мыши или установленных четырёх клавиш, фиксированного вертикального угла обзора, возможностью кругового вращения и изменения расстояния от камеры до поверхности в заданном пределе (рис. 43).



Рисунок 43. Демонстрация положения RPG камеры на максимальном удалении

4.4. Оптимизация рендеринга растительности

4.4.1. 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 (рис. 44).

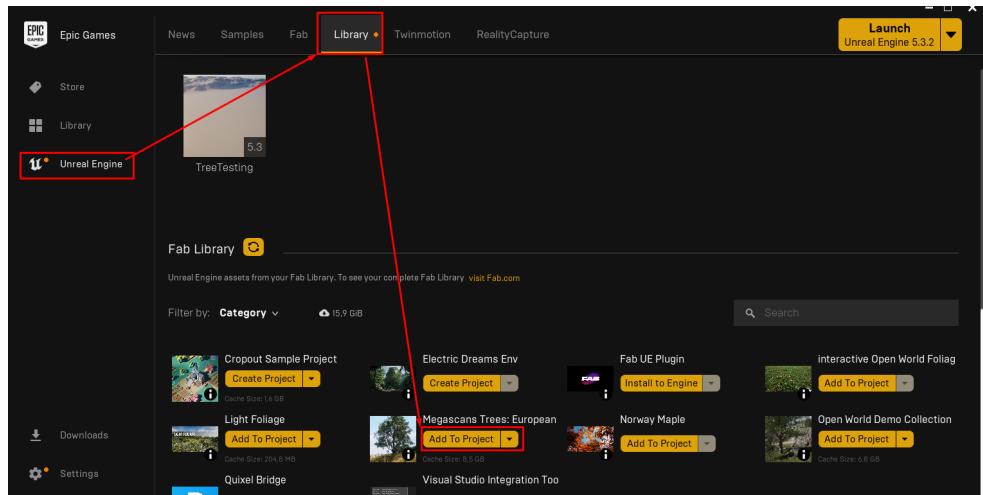


Рисунок 44. Добавление расширений в проект

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 (рис. 45). 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.

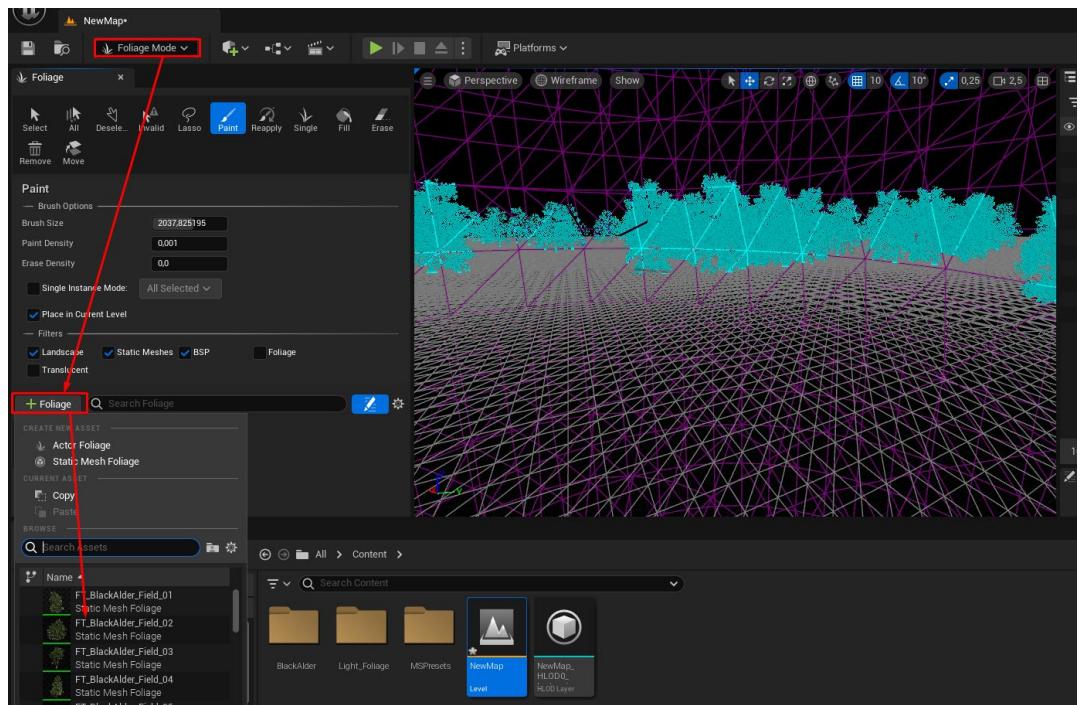


Рисунок 45. Добавление растительности в режиме Foliage

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 (рис. 46).

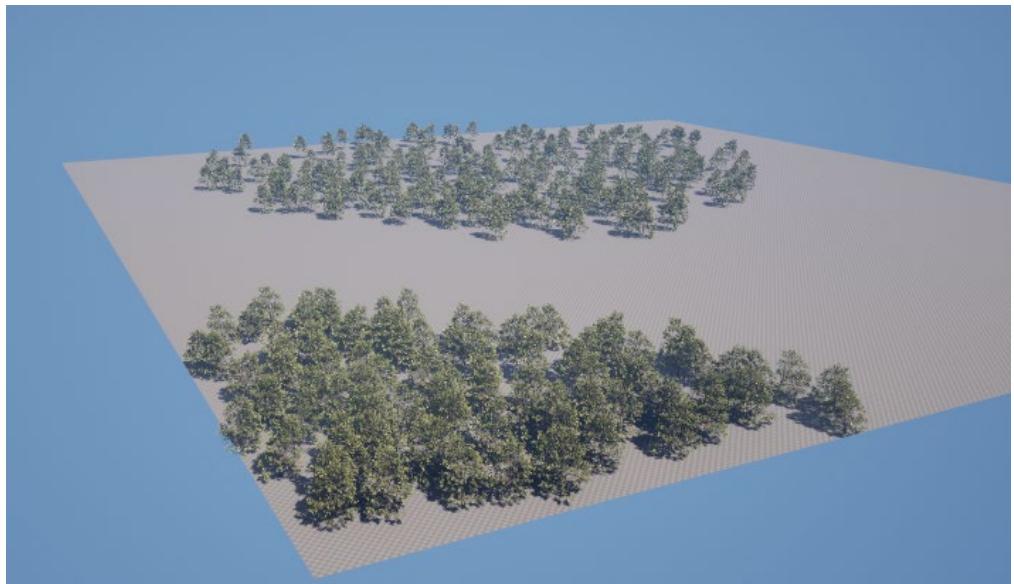


Рисунок 46. Сцена после добавления объектов растительности

4.4.2. 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 на рисунках 47, 48 и 49 представлены результаты замеров производительности по трём выбранным ракурсам обзора точки наблюдения.

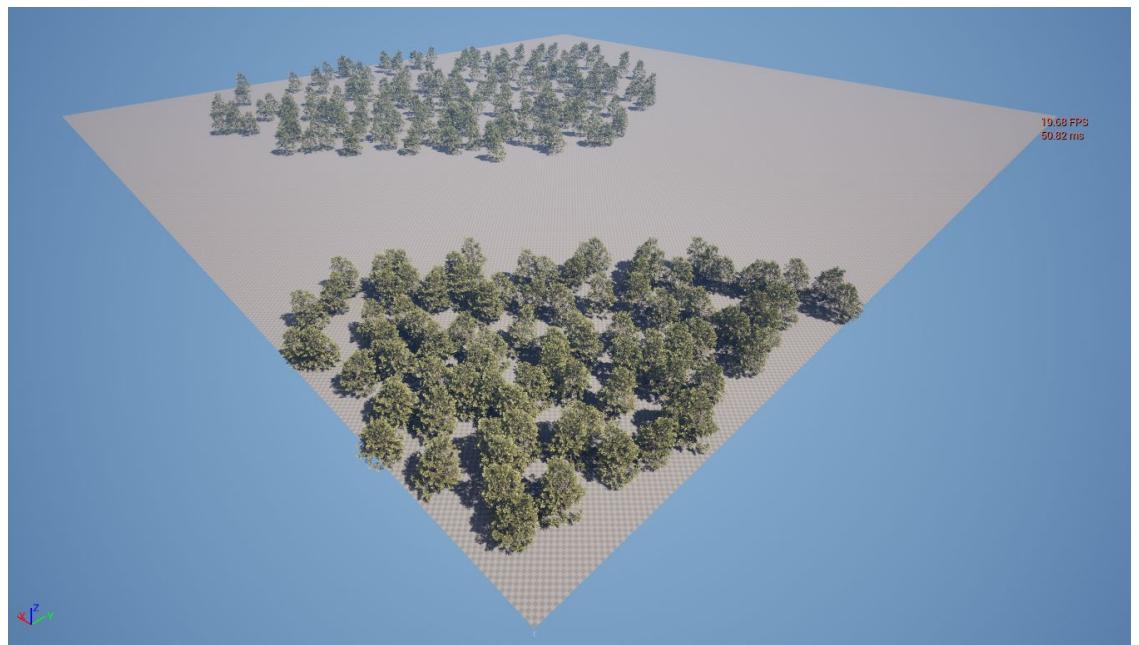


Рисунок 47. Ракурс №1 до оптимизации, 19 FPS

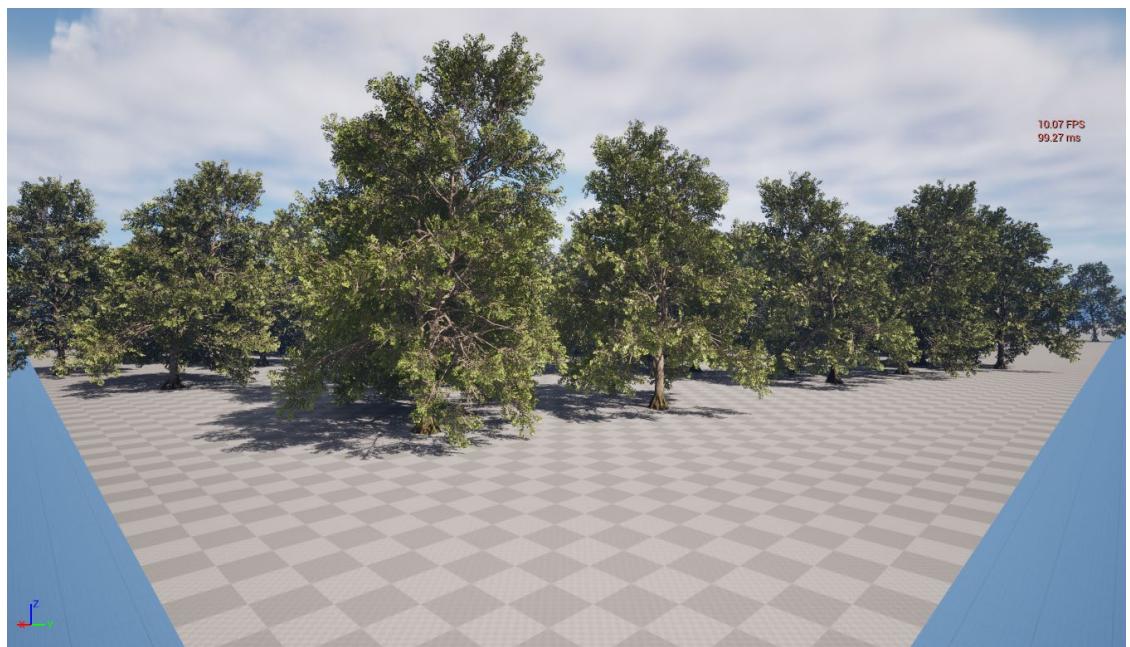


Рисунок 48. Ракурс №2 до оптимизации, 10 FPS

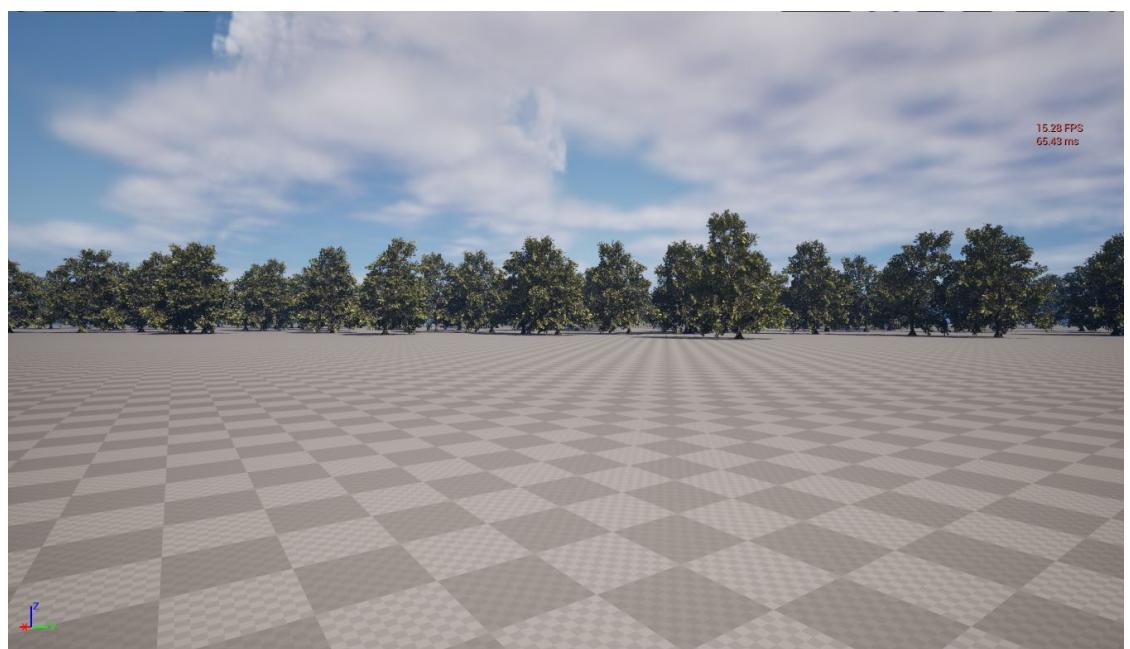


Рисунок 49. Ракурс №3 до оптимизации, 15 FPS

4.4.3. 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:

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.

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 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:

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.

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

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.

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:

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 смещивания 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.

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:

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.

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 сложности шейдеров (Shader Complexity Mode), который помогает выявить проблемные зоны и оптимизировать производительность.

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

4.4.4. 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 необходимые настроенные уровни детализации, однако по причине отсутствия взаимодействия с системой Nanite были недоступны к использованию настройки уровней LOD. После этого было использовано 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 (рис. 50, рис. 51 и рис. 52):

33 FPS. Прирост производительности в среднем составил 73%.

17 FPS. Прирост производительности в среднем составил 70%.

27 FPS. Прирост производительности в среднем составил 80%

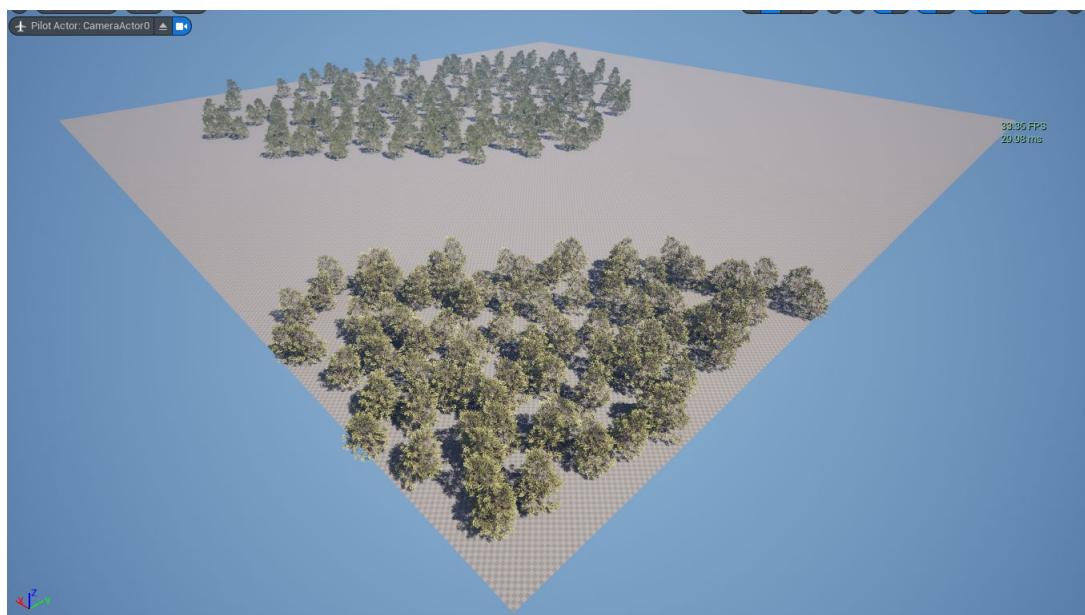


Рисунок 50. Ракурс №1 после оптимизации, 33 FPS



Рисунок 51. Ракурс №2 после оптимизации, 17 FPS

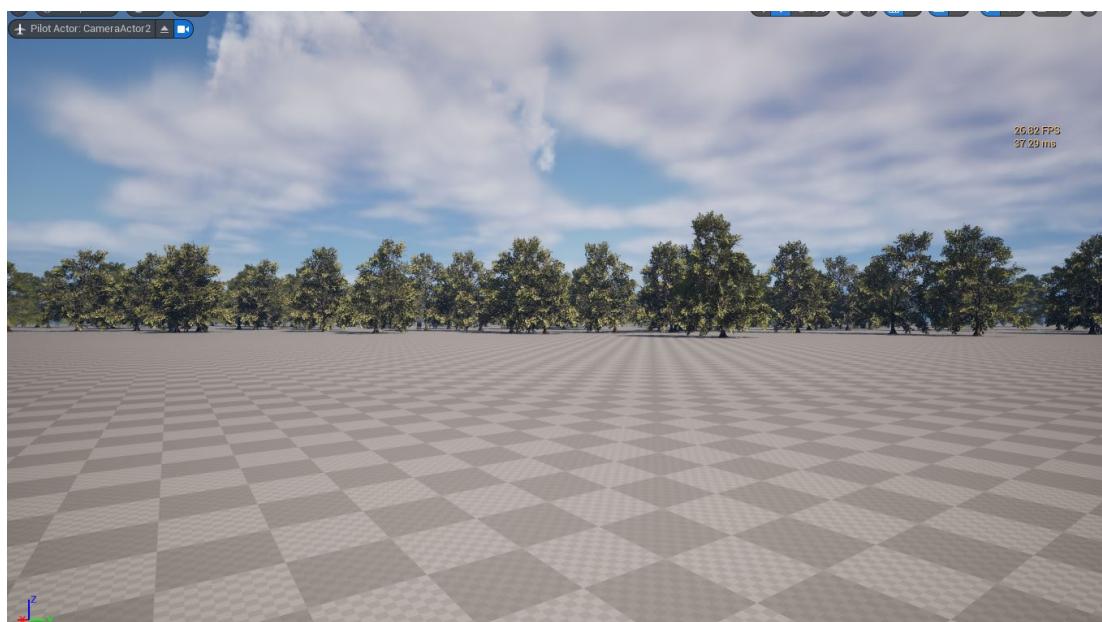


Рисунок 52. Ракурс №3 после оптимизации, 27 FPS

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

Использование полученных методов оптимизации, а также уменьшении числа избыточных объектов в исходном проекте представлено ниже на рисунках 53 и 54.



Рисунок 53. Свободная камера обзора, 15 FPS



Рисунок 54. Фиксированная камера обзора пользователя, 36 FPS

Здесь при значительном увеличении числа объектов, добавлении ландшафта, текстур и травы наблюдаются параметры в 15FPS для свободной камеры с широким углом обзора и 36 FPS для камеры, через которую будет получен доступ обзора пользователю. Таким образом результат повышения производительности в рамках изначального проекта составил порядка 50%.

4.5. Добавление пользовательского интерфейса

Для добавления пользовательского интерфейса необходимо использование одной из трёх встроенных подсистем Unreal Engine 5 для создания 2d-приложений Slate, Canvas или Unreal Motion Graphics (UMG) [11]. Несмотря на значительный выбор, документация разработчиков по большей степени затрагивает UMG. Для создания пользовательского интерфейса необходимо использование объекта WidgetBlueprint, который позволяет добавлять, как стандартные элементы интерфейса – окна, рамки, сортировочные панели, кнопки и т.п., так и модифицировать их через встроенный Blueprint скрипт.

Ниже на рисунке 55 представлено стандартное окно объекта WidgetBlueprint.

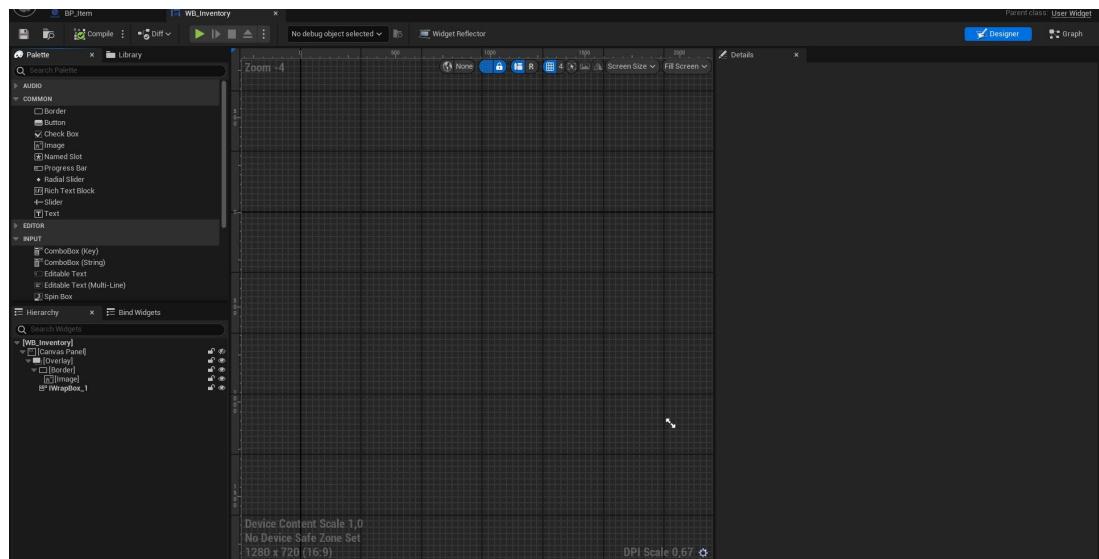


Рисунок 55. Базовое окно WidgetBlueprint

Далее при помощи следующих элементов был создан простое визуальное отображение пустого инвентаря пользователя:

- Canvas Panel – представляет собой замещающий элемент холста, который предназначен для предоставления ориентации всем объектам находящимся внутри относительно текущего состояния монитора пользователя. Не имеет визуального представления.
- Overlay – объект служащий для выделения и помещения других объектов в обозначенную область на холсте. Не имеет визуального представления.

- Border – объект определяющий непосредственные границы дочерних объектов внутри себя. Позволяет задать цвет, который будет отображен если ни один из дочерних объектов не будет его перекрывать.
- Image – объект, позволяющий добавить любое изображение на своё место.
- WrapBox – сортировочный объект замощения, который позволяет добавлять другие элементы произвольных визуальных массивов (например ячеек инвентаря) внутрь себя с переносом на новую строку.

Вид WidgetBlueprint после добавления всех необходимых элементов (рис. 56)

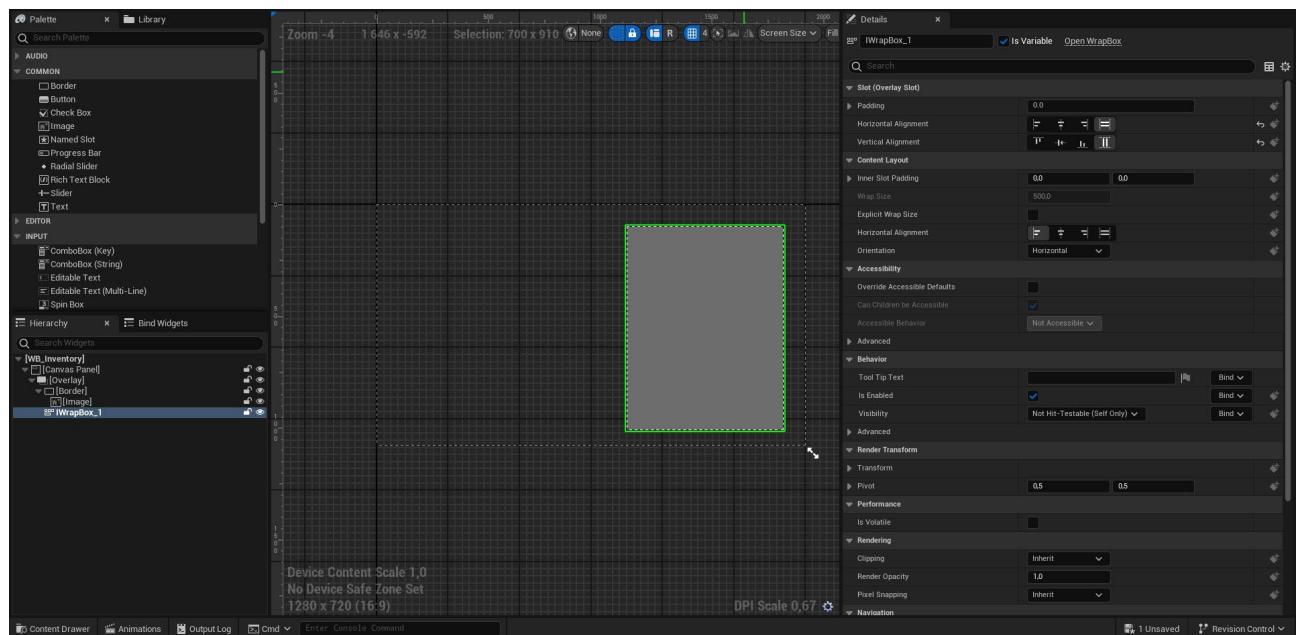


Рисунок 56. WidgetBlueprint инвентаря

Для вызова полученного элемента интерфейса необходимо изменить существующий Blueprint скрипт для пользовательского контроллера (рис. 57).

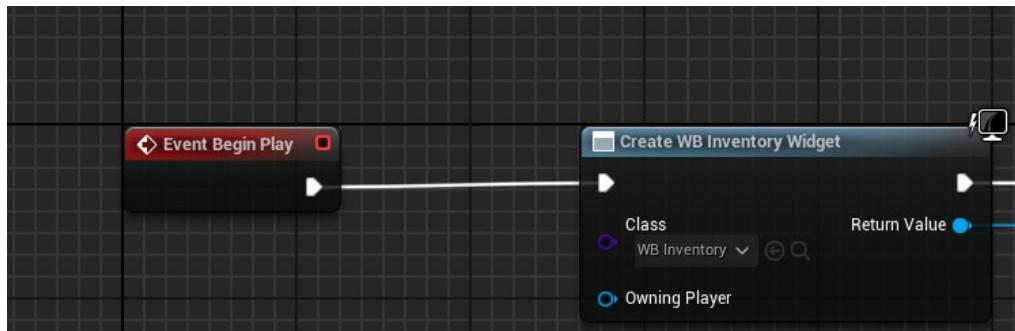


Рисунок 57. Blueprint скрипт вызова инвентаря

В процессе работы программы вызов панели инвентаря выглядит следующим образом (рис. 58):

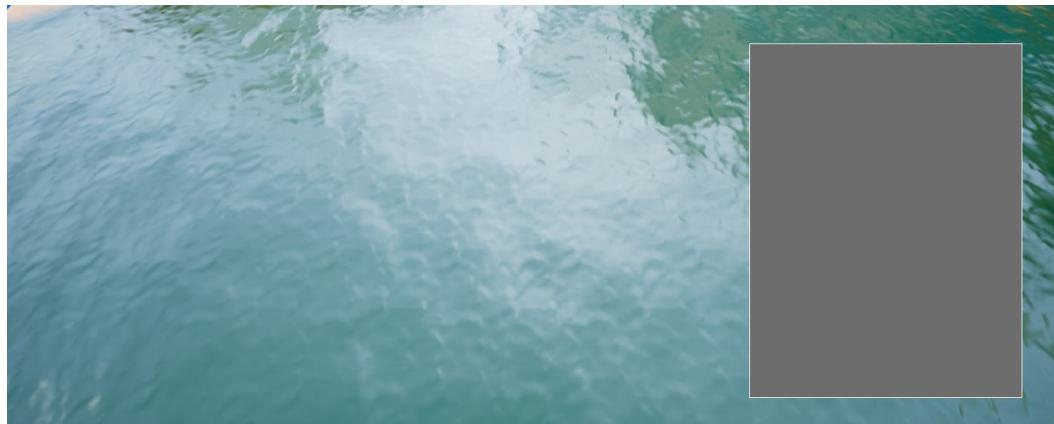


Рисунок 58. Панель инвентаря в процессе работы программы

Для добавления ячеек необходимо создать структуру, хранящую данные о возможных ячейках, такие как Name (имя), Stackable (возможность группировки), Quantity (число), Thumbnail (иконка) и Mesh (сетка объекта). Для подобных целей в среде UE5 используется объект Structure.

Далее был создан ещё один объект WidgetBlueprint отображающий пустую ячейку в виде белого квадрата, изображение которого может быть изменено при добавлении элемента исходя из хранимого в нём объекта Thumbnail.

Затем был написан Blueprint скрипт формирования всех необходимых параметров отображения элементов инвентаря (рис. 59):



Рисунок 59. Blueprint скрипт загрузки элементов инвентаря

Далее была реализована система переключения отображения инвентаря и изменения максимального числа элементов (рис. 60):

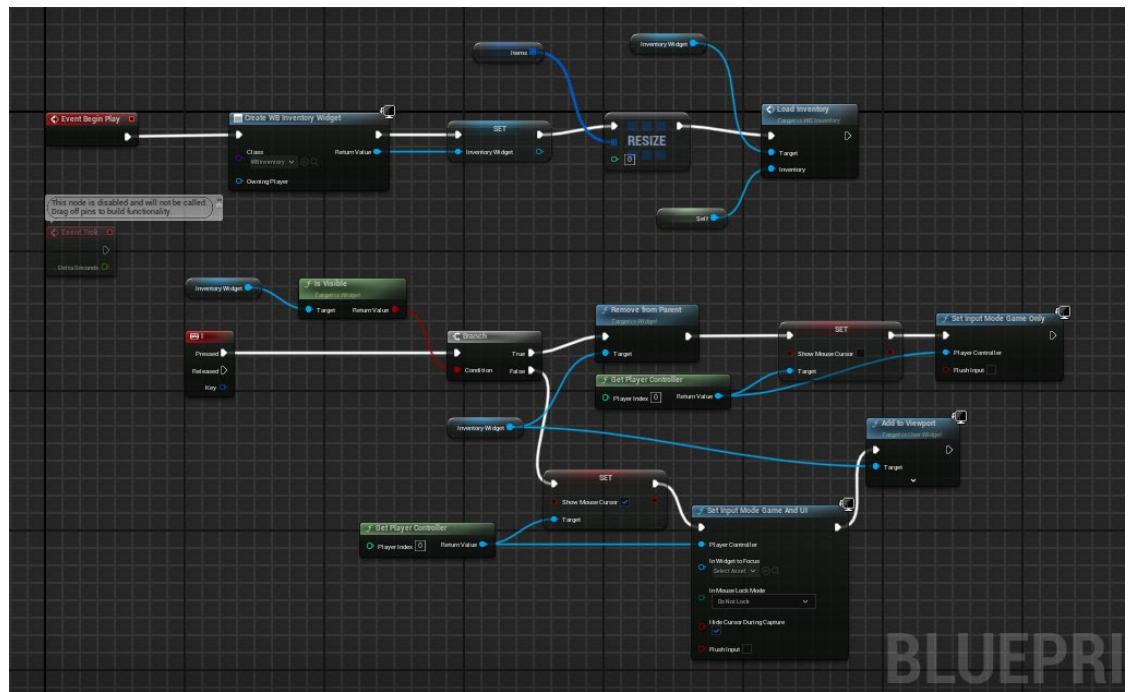


Рисунок 60. Blueprint скрипт системы отображения

Добавление 85 пустых ячеек инвентаря выглядит следующим образом (рис. 81):



Рисунок 61. Панель инвентаря с 85 пустыми ячейками

Для реализации возможности подбора, хранения и возможности удаления предметов из инвентаря был создан ещё один Blueprint скрипт.

Анализ доступ к подбираемому предмету осуществляется через добавление невидимой сферической области вокруг области пользователя по нажатию клавиши взаимодействия. В случае если подбираемый предмет оказывается в области системой проверяется возможность добавления данного элемента в группу. В случае возможности группировки объекта внутри инвентаря осуществляется поиск схожих элементов и если их группа не превышает стандартного максимального числа в 64 элемента, то этот элемент будет добавлен в ту же ячейку инвентаря. В противном случае он будет добавлен в первую пустую ячейку.

Удаление предметов производится по нажатию мыши на иконку предмета в инвентаре, при этом система вызывает данный объект перед пользователем с использованием соответствующего параметра сетки, указанного в структуре данных.

Ниже представлен Blueprint скрипт данного алгоритма (рис. 62):

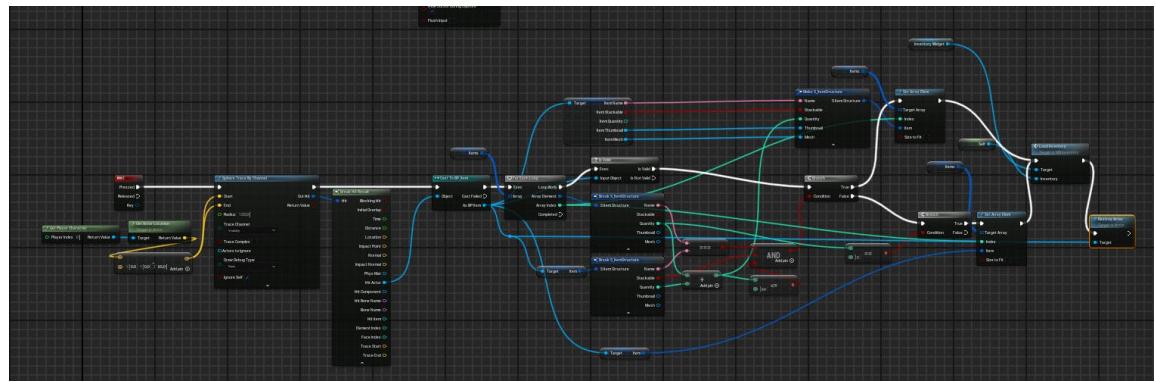


Рисунок 62. Blueprint скрипт подбора и выброса предмета из инвентаря

Для добавления статусной панели пользователя был разработан ещё один WidgetBlueprint (рис. 63) состоящий из двух сфер, отражающих параметры персонажа, такие как здоровье (слева) и энергия (справа).

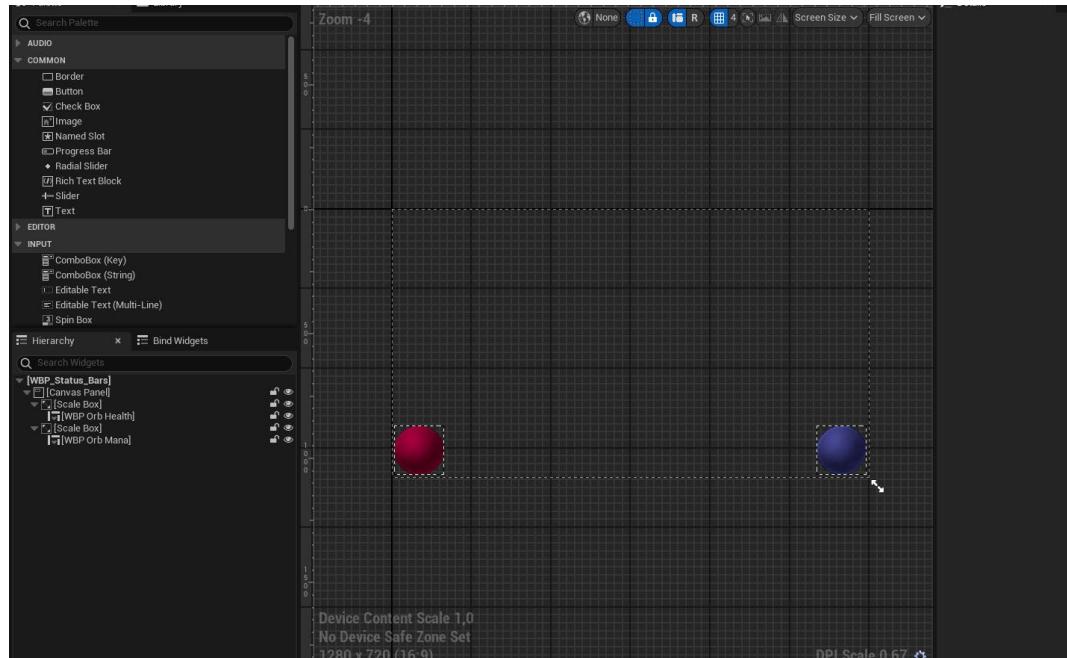


Рисунок 63. WidgetBlueprint статусной панели пользователя

Ниже представлено отображение статусной панели в процессе работы программы (рис. 64):

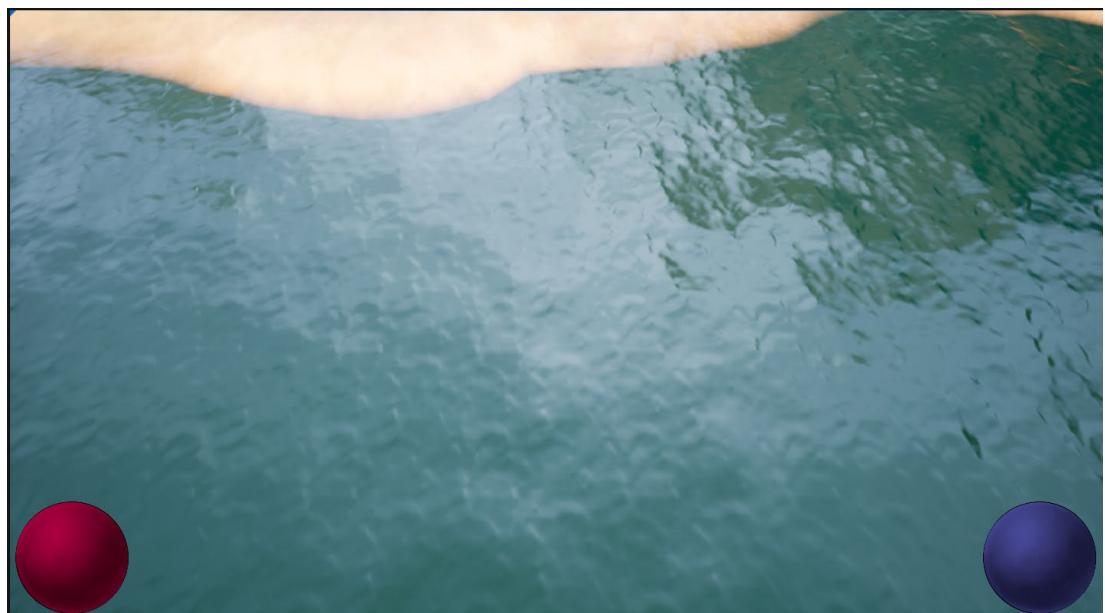


Рисунок 64. Статусная панель пользователя в процессе работы программы

ЗАКЛЮЧЕНИЕ

В результате выполнения работы были рассмотрены методы создания ландшафтов в современных программных продуктах и способы создания систем с элементами присущими проектам типа RPG в Unreal Engine 5.

В процессе были решены практические задачи изучения методов и создания ландшафта, добавления и оптимизации деревьев, создания и экспорта 3D-моделей, а также добавления элементов характерных проектам RPG.

Был поэтапно описан процесс получения необходимого ландшафта в программе World Creator 3, который затем был преобразован в основание трёхмерной сцены в среде движка Unreal Engine 5. Проанализированы ключевые моменты создания 3d-моделей в программной среде Blender с последующим изучением методов экспорта и интеграции объектов в иные системы. Рассмотрены и применены алгоритмы оптимизации производительности множества использованных объектов сцены, в частности наиболее ресурсоёмких объектов таких как растительность. Исследован функционал системы создания 2d-объектов UMG с последующим использованием для создания необходимых характерных элементов функционала для проектов типа RPG.

В результате работы был создан проект 3d-сцены в Unreal Engine 5 с возможностью взаимодействия между пользователем и созданной средой с установленными ограничениями характерными для RPG проекта.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Basic Navigation / [Электронный ресурс] // Unreal Engine Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/basic-navigation-in-unreal-engine?application_version=5.1 (дата обращения: 16.09.2023).
2. Modifying the Navigation Mesh / [Электронный ресурс] // Unreal Engine Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/overview-of-how-to-modify-the-navigation-mesh-in-unreal-engine?application_version=5.1 (дата обращения: 16.09.2024).
3. Landscape Material Layer Blending // Unreal Engine Documentation URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/landscape-material-layer-blending-in-unreal-engine?application_version=5.1 (дата обращения: 16.09.2024).
4. Landscape Quick Start Guide / [Электронный ресурс] // Unreal Engine Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/landscape-quick-start-guide-in-unreal-engine?application_version=5.0 (дата обращения: 16.09.2024).
5. Creating Landscapes / [Электронный ресурс] // Unreal Engine Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/creating-landscapes-in-unreal-engine?application_version=5.2 (дата обращения: 16.09.2024).
6. Landscape Edit Layers / [Электронный ресурс] // Unreal Engine Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/landscape-edit-layers-in-unreal-engine?application_version=5.2 (дата обращения: 16.09.2024).
7. Landscape Collision Guide / [Электронный ресурс] // Unreal Engine Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/landscape-collision-guide-in-unreal-engine?application_version=5.2 (дата обращения: 16.09.2024).

8. Chapter 2. Terrain Rendering Using GPU-Based Geometry Clipmaps / [Электронный ресурс] // Nvidia Developer : [сайт]. — URL: <https://developer.nvidia.com/gpugems/gpugems2/part-i-geometric-complexity/chapter-2-terrain-rendering-using-gpu-based-geometry> (дата обращения: 10.02.2025).

9. Hair Rendering / [Электронный ресурс] // Unreal Engine Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/hair-rendering-in-unreal-engine?application_version=5.2 (дата обращения: 15.02.2025).

10. Generating Groom Textures / [Электронный ресурс] // Unreal Engine Documentation : [сайт]. — URL: https://dev.epicgames.com/documentation/en-us/unreal-engine/generating-groom-textures-in-unreal-engine?application_version=5.2 (дата обращения: 19.03.2025).

11. UMG UI Designer Quick Start Guide / [Электронный ресурс] // Unreal Engine Documentation : [сайт]. — URL: <https://dev.epicgames.com/documentation/en-us/unreal-engine/umg-ui-designer-quick-start-guide-in-unreal-engine> (дата обращения: 13.05.2025)

12. Using Cameras / [Электронный ресурс] // Unreal Engine Documentation : [сайт]. — URL: <https://dev.epicgames.com/documentation/en-us/unreal-engine/using-cameras-in-unreal-engine> (дата обращения: 14.05.2025).

ПРИЛОЖЕНИЕ А. Графическая часть выпускной квалификационной работы

В графическую часть выпускной квалификационной работы входят 8 чертежей.

- Создание ландшафта в WorldMachine.
- Импорт созданного ландшафта в Unreal Engine.
- Процедурная генерация деревьев и травы с помощью PCG.
- Разделение леса на зоны.
- Настройка анимаций персонажа.
- Основные деревья поведения персонажа и противника.
- Поддеревья поведения персонажа (охота и рубка леса).
- Поддеревья поведения персонажа (приготовление и поглощение пищи, сон, добавление древесины в костер).