Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет інформаційних радіотехнологій та технічного захисту інформації
(повна назва)

Кафедра медіаінженерії та інформаційних радіоелектронних систем
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Комплексна тема. Оптимізація 3D-моделі для ігрового рушія Unity 3D
(тема)

Виконав:
студент 2 курсу, групи СТМм-22-1
Максим ГАЄВИЙ
(прізвище, ініціали)

Спеціальність 171 Електроніка
(код і повна назва спеціальності)
Тип програми освітньо-професійна
Освітня програма Системи, технології і комп'ютерні засоби мультимедіа
(повна назва освітньої програми)

Керівник ас. Олена ОЛЕЙНІКОВА
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Володимир КАРТАШОВ
(підпис) (прізвище, ініціали)

2023 р.

Харківський національний університет радіоелектроніки

Факультет Інформаційних радіотехнологій та технічного захисту інформації

Кафедра Медіаінженерії та інформаційних радіоелектронних систем

Рівень вищої освіти другий (магістерський)

Спеціальність 171 Електроніка
(код і повна назва)

Тип програми освітньо-професійна

Освітня програма "Системи, технології і комп'ютерні засоби мультимедіа"
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«____»_____ 20 ___ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Гаєвому Максиму Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Комплексна тема. Оптимізація 3D-моделі для ігрового рушія Unity 3D
затверджена наказом університету від 20 листопада 20 23 р. № 1371СТ
2. Термін подання студентом роботи до екзаменаційної комісії 8 січня 2024 р.
3. Вихідні дані до роботи
1. Розглянути існуючі безкоштовні ігрові рушії
2. Розглянути основні етапи процесу створення та оптимізації 3D моделі
3. Створити ігрову сцену та імпортувати оптимізовану 3D-модель до ігрового рушія Unity 3D. Порівняти оптимізовану та неоптимізовану 3D-моделі в різних умовах сцени
4. Перелік питань, що потрібно опрацювати в роботі
ВСТУП
1. ПОРІВНЯННЯ ІГРОВИХ РУШІЇВ ДЛЯ СТВОРЕННЯ ВІДЕОІГОР. ОЗНАЙОМЛЕННЯ З МОЖЛИВОСТЯМИ ІГРОВОГО РУШІЯ UNITY 3D
2. СТВОРЕННЯ ТА ОПТИМІЗАЦІЯ 3D-МОДЕЛІ ДЛЯ ІГРОВОГО РУШІЯ UNITY 3D
3. СТВОРЕННЯ СЦЕНИ ТА ПОРІВНЯННЯ ОПТИМІЗОВАНОЇ МОДЕЛІ З ВИСОКОПОЛІГОНАЛЬНОЮ МОДЕЛЛЮ В ІГРОВОМУ РУШІЇ UNITY 3D
ВИСНОВКИ
ПЕРЕЛІК ПОСИЛАНЬ
ДОДАТКИ
5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій:
1. Референсні зображення для створення 3D-моделі мортири; 2. Створена High-poly модель мортири; 3. Створена Low-poly модель мортири; 4. Створена UV-розгортка Low-poly моделі; 5. Затекстурована 3D-модель мортири; 6. Імпортована оптимізована модель в ігровий рушій Unity.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Порівняння ігрових рушіїв для створення відеоігор. Ознайомлення з можливостями ігрового рушія Unity 3D	21.11.23 – 27.11.2023	
2	Створення та оптимізація 3D-моделі для ігрового рушія Unity 3D	28.11.23 – 02.12.2023	
3	Створення сцени та порівняння оптимізованої моделі з високополігональною моделлю в ігровому рушії Unity 3D	04.12.2023 – 09.12.2023	
4	Графічна частина проекту	11.12.2023 – 20.12.2023	
5	Перевірка керівником проекту	24.12.2023 – 26.12.2023	
6	Перевірка нормконтролем	27.12.2023 – 28.12.2023	
7	Перевірка зав.кафедрою, рецензування	3.01.2024 – 10.01.2024	

Дата видачі завдання _____20.11.2023_____

Студент _____ _____ Максим ГАЄВИЙ
(підпис)

Керівник роботи _____ ___ас. Олейнікова О.І._____
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до атестаційної роботи: 75 сторінок, 68 рисунків, 3 таблиці, 15 джерел.

HIGH POLY, LOW POLY, 3D, МОДЕЛЬ, РОЗГОРТКА, ТЕКСТУРУВАННЯ, ОПТИМІЗАЦІЯ, 3D МОДЕЛЮВАННЯ, ІГРОВІ РУШІЇ, ТРИАНГУЛЯЦІЯ

Об’єкт дослідження – оптимізація 3D-моделі для ігрового рушія Unity 3D.

Мета роботи – засвоїти основні правила створення 3D-моделі, оптимізацію, типи програм для моделінгу, імпортувати 3D-модель в ігровий рушій Unity 3D.

В атестаційній роботі було виконано моделювання 3D-мортири та її оптимізацію за допомогою програмного забезпечення Blender 3D.

У першому розділі описано основні безкоштовні ігрові рушії, їх переваги та недоліки в порівнянні з Unity 3D.

У другому розділі описано основні етапи та правила, які використовуються під час 3D моделювання, текстурування та оптимізації 3D моделі.

У третьому розділі описано процес імпорту 3D моделі в ігровий рушій Unity 3D та показано еталоні тести оптимізованої та неоптимізованої моделі в різних умовах використання.

ABSTRACT

Explanatory note to the certification work: 75 pages, 68 figures, 3 tables, 15 sources.

HIGH POLY, LOW POLY, 3D, MODEL, SNAP, TEXTURE, OPTIMIZATION, 3D MODELING, GAME ENGINES, TRIANGULATION

The object of research is optimization of the 3D model for the Unity 3D game engine.

The purpose of the work is to learn the basic rules of creating a 3D model, optimization, types of modeling programs, to import a 3D model into the Unity 3D game engine.

In the certification work, a 3D mortar was modeled and optimized using the Blender 3D software.

The first section describes the main free game engines, their advantages and disadvantages compared to Unity 3D.

The second chapter describes the main steps and rules used during 3D modeling, texturing and optimization of a 3D model.

The third section describes the process of importing a 3D model into the Unity 3D game engine and shows benchmark tests of an optimized and non-optimized model under different conditions of use.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

Low-poly – модель з низькополігональною сіткою;

High-poly – модель з високополігональною сітку;

Triangulate – Перетворення полігонів в трикутники та оптимізація полігональної сітки;

SP – Substance Painter – програмне забезпечення для текстурування;

UV-mapping – UV-розгортка;

FPS – Frame Per Second;

HDRP – High Definition Render Pipeline;

LOD – Level of detail;

AWS – Amazon Web Service;

SSAO - Screen Space Ambient Occlusion;

SSGI - Screen Space Global Illumination;

URP – Universal Render Pipeline

ВСТУП

У сучасному світі відеоігор, де реалізм і занурення стають ключовими факторами, ефективне використання ресурсів і оптимізація є важливими завданнями для розробників. Ця робота спрямована на дослідження та розробку методів оптимізації 3D-моделей з метою забезпечення високої продуктивності ігрових рушіїв та збереження візуальної якості графіки.

Індустрія відеоігор в останні роки стрімко розвивається, а разом з нею зростають і вимоги до оптимізації графічних ресурсів. В умовах обмеженого обчислювального потенціалу консолей і персональних комп'ютерів розробники повинні шукати нові підходи та інноваційні методи оптимізації, щоб забезпечити гравцям максимально плавний і вражаючий ігровий процес.

Оптимізація не тільки покращує продуктивність, але й дозволяє розробникам максимально використовувати можливості графічних движків для досягнення вражаючих візуальних ефектів.

Оскільки ігрова індустрія разом з технічним прогресом стрімко розвивається, актуальним питанням залишається як створення нових тривимірних графічних моделей, так і подальше удосконалення якості існуючих тримірних моделей.

В атестаційній роботі охоплюється сучасні етапи оптимізації 3D-моделей з подальшим імпортом в ігровий рушій та проведенням графічних тестів на продуктивність.

Саме тому створення і оптимізація тривимірних моделей для ігрової сфери є актуальною темою сьогодення.

1 ПОРІВНЯННЯ ІГРОВИХ РУШІЇВ ДЛЯ СТВОРЕННЯ ВІДЕОІГОР. ОЗНАЙОМЛЕННЯ З МОЖЛИВОСТЯМИ ІГРОВОГО РУШІЯ UNITY 3D

1.1 Порівняння ігрових рушіїв для створення відеоігор

Для створення відеоігор використовуються ігрові рушії, в них проходить основна частина роботи над ігровим проектом. Існує багато ігрових рушіїв, деякі компанії обирають вже створені рушії, а деякі створюють новий рушій для свого проекту. Найпопулярніші ігрові рушії:

- Unity;
- Unreal Engine;
- CryEngine;
- Godot Engine;
- Lumberyard;
- Source Engine

1.1.1 Unity 3D

Unity 3D — багатоплатформовий інструмент, заснований у 2005 році, для розроблення відеоігор і застосунків, на якому вони працюють. Створені за допомогою Unity програми працюють на настільних комп'ютерних системах, мобільних пристроях та гральних консолях у дво- та тривимірній графіці, та на пристроях віртуальної чи доповненої реальності. Застосунки, створені за допомогою Unity, підтримують DirectX та OpenGL.

Сильні сторони Unity:

- Кросплатформеність:

Unity дозволяє розробляти ігри для різних платформ, таких як ПК, консолі, мобільні пристрої (iOS, Android), веб і багато інших, як показано на (рис. 1.1). Це

робить його ідеальним вибором для розробників, які хочуть охопити широку аудиторію. [1]

Рисунок 1.1 — Приклад гри на рушії Unity [1]

– Велика спільнота та ресурси:

Unity має активну спільноту розробників, що робить легше знаходження відповідей на питання, обмін досвідом та вивчення нових технік. Онлайн-ресурси, такі як Unity Asset Store та Unity Learn, надають доступ до безлічі активів, розширень та навчальних матеріалів.

– Швидкість розробки:

Unity має інтуїтивний інтерфейс та зручний редактор, що сприяє швидкій розробці прототипів та експериментам.

– Мова програмування:

Unity використовує мову програмування C#, яка є ефективною та легкою для вивчення. Це робить його привабливим для студентів та новачків.

– Unity Asset Store:

Магазин активів Unity надає велику кількість готових ресурсів, моделей, текстур та інших матеріалів, які розробники можуть використовувати для полегшення роботи та зекономлення часу.

Слабкі сторони Unity:

– Графічні можливості:

Порівняно з іншими рушіями, такими як Unreal Engine, Unity може мати менш вражаючі графічні можливості для створення фотореалістичних ігор.

– Високі вимоги до ресурсів:

Деякі складні 3D-проекти в Unity можуть вимагати значних ресурсів, особливо щодо обсягу пам'яті та потужності графічної карти.

– Система фізики:

Деякі розробники вказують на те, що система фізики в Unity не є такою потужною, як у деяких конкурентів.

1.1.2 Unreal Engine

Unreal Engine - це потужний ігровий рушій, розроблений компанією Epic Games. Він використовується для створення високоякісних ігор, віртуальної реальності, архітектурних візуалізацій та інших інтерактивних додатків. Unreal Engine відомий своєю потужною графікою та широким спектром функціональностей.

Сильні сторони Unreal Engine:

– Потужна графіка:

Unreal Engine славиться своєю імпресивною графікою та вражаючими візуальними ефектами, як показано на (рис 1.2). Він надає розробникам широкий функціонал для створення фотореалістичних ігрових світів.

– Blueprints та C++:

Unreal Engine пропонує два підходи до програмування - використання Blueprints (візуального програмування) та мови програмування C++. Це надає розробникам гнучкість у виборі способу програмування. [3]

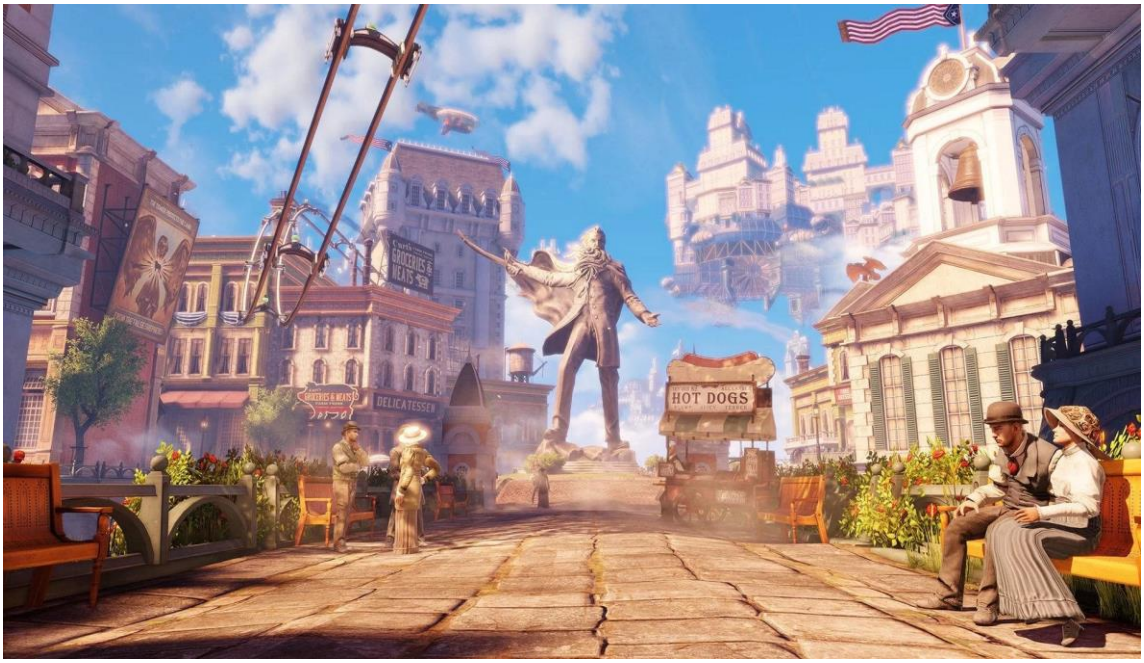

Рисунок 1.2 — Приклад гри на рушії Unreal Engine [2]

– Великий обсяг готових ресурсів:

Unreal Engine також має свій власний магазин ресурсів - Unreal Marketplace, де розробники можуть знаходити готові активи, текстури та інші матеріали для використання у своїх проектах.

– Вбудовані інструменти для VR:

Unreal Engine активно використовується для створення віртуальної реальності (VR) і має вбудовані інструменти, які полегшують розробку VR-проектів.

– Можливості для масштабування:

Рушій може бути використаний для розробки як невеликих інді-ігор, так і великих AAA-проектів. Він добре масштабується для потреб різних розмірів та типів проектів.

Слабкі сторони Unreal Engine:

– Вивчення кривої:

В порівнянні з іншими рушіями, вивчення Unreal Engine може вимагати більше часу, особливо для новачків. Процес вивчення може бути складним через обширну функціональність та глибокий рівень деталей. [2]

– Високі вимоги до апаратного забезпечення:

Створення графіки високої якості може вимагати значних обчислювальних ресурсів та сучасної графічної карти.

– Обмежена кросплатформеність:

Хоча Unreal Engine підтримує різні платформи, процес кросплатформенної розробки може виявитися менш простим порівняно з Unity. [4]

1.1.3 CryEngine

CryEngine - це високопродуктивний ігровий рушій, розроблений компанією Crytek. Він використовується для створення графічно продвинутих та реалістичних ігор, а також для віртуальної реальності та інших інтерактивних візуалізацій.

Сильні сторони CryEngine:

– Графічні можливості:

Однією з основних переваг CryEngine є його потужність у графіці. Він володіє вражаючими візуальними ефектами та можливостями для створення реалістичних світів, як показано на (рис 1.3).

Рисунок 1.3 — Приклад гри на рушії CryEngine [2]

– Система освітлення та реалістичність:

CryEngine відомий своєю реалістичною системою освітлення, динамічними ефектами та фотореалістичним рендерингом.

– Свобода розробки:

Рушій надає розробникам велику свободу та гнучкість у створенні власних ігор та великих географічно розгалужених світів.

– Віртуальна реальність:

CryEngine підтримує віртуальну реальність, що робить його привабливим вибором для розробки VR-проектів.

– Інструменти для архітектурної візуалізації:

Рушій також широко використовується для створення візуалізацій архітектурних проектів, дозволяючи створювати деталізовані та вражаючі моделі.

Слабкі сторони CryEngine:

– Високі вимоги до апаратного забезпечення:

Створення реалістичних ігор за допомогою CryEngine може вимагати потужного обладнання та графічних карт.

– Специфічна крива вивчення:

Навчання використанню CryEngine може бути більш витратним з точки зору часу порівняно з іншими рушіями, особливо для новачків.

– Спільнота та документація:

Спільнота розробників та доступна документація можуть бути менш обширними, порівняно з іншими популярними ігровими рушіями.

– Не така популярна, як Unity або Unreal Engine:

Хоча CryEngine є потужним рушієм, він не так широко використовується в ігровій індустрії, як, наприклад, Unity або Unreal Engine. [6]

1.1.4 Godot Engine

Godot Engine - це відкритий ігровий рушій, який використовується для розробки як 2D, так і 3D ігор, а також інших інтерактивних додатків. Приклад гри можна побачити на (рис. 1.4). Його особливість полягає в тому, що він є вільним програмним забезпеченням з відкритим кодом, що робить його доступним та редагованим групою розробників.

Рисунок 1.4 — Приклад гри на рушії Godot Engine [4]

Сильні сторони Godot Engine:

– Відкритий код та безкоштовність:

Godot Engine є повністю відкритим та безкоштовним з вихідним кодом, що робить його доступним для всіх розробників. Це сприяє розширенню та адаптації за допомогою спільноти.

– Кросплатформеність:

Рушій підтримує розробку для різних платформ, таких як Windows, macOS, Linux, Android, iOS та інші, що дозволяє створювати ігри для широкої аудиторії.

– Широкий набір функцій:

Незважаючи на свою безкоштовність, Godot Engine має вражаючий набір функцій, який включає в себе системи фізики, систему частинок, систему анімації, систему аудіо, вбудовані інструменти для створення гри в реальному часі та інші.

– GDScript:

Godot Engine використовує свою мову програмування - GDScript, яка є легкою для вивчення та використання, особливо для новачків. Він також підтримує інші мови програмування, такі як C#.

– Спільнота та навчальні ресурси:

Існує активна спільнота розробників Godot, яка надає підтримку, обмін досвідом та навчальні матеріали, такі як відеоуроки та блоги.

Слабкі сторони Godot Engine:

– Менший екосистема ресурсів:

Особливо порівняно з іншими популярними рушіями, Godot може мати менше готових ресурсів та активів, доступних в онлайн-магазинах.

– Менша популярність в індустрії:

У порівнянні з Unity або Unreal Engine, Godot менше використовується в ігровій індустрії, особливо для великих AAA-проектів.

– Менше продвинуті графічні можливості:

В порівнянні з Unreal Engine чи CryEngine, Godot може мати менш продвинуті графічні можливості для створення вражаючих візуальних ефектів. [7]

1.1.5 Amazon Lumberyard

Amazon Lumberyard - це ігровий рушій, розроблений компанією Amazon. Lumberyard пропонує інтеграцію з обліковим записом Amazon Web Services (AWS) та іншими сервісами Amazon. Цей рушій спрямований на створення багатокористувацьких ігор, як показано на (рис 1.5), а також використання хмарних технологій для поліпшення розширюваності та ефективності розробки.

Рисунок 1.5 — Приклад гри на рушії Lumberyard [2]

Сильні сторони Lumberyard:

– Інтеграція з Amazon Web Services (AWS):

Lumberyard має глибоку інтеграцію з AWS, що дозволяє розробникам легко використовувати хмарні сервіси для таких завдань, як обробка аудіо, інтеграція з штучним інтелектом та інші хмарні обчислення.

– Багатокористувацькі можливості:

Lumberyard спеціалізується на створенні багатокористувацьких ігор, і має вбудовані функції для розробки масштабних графічно вражаючих ігор, що підтримують велику кількість одночасних гравців. [8]

– Безкоштовність та відкритість коду:

Lumberyard є безкоштовним для використання, і його вихідний код доступний для розробників, що робить його привабливим для тих, хто шукає доступні та редаговані ігрові рушії.

– Розширюваність:

Рушій підтримує можливість розширення за допомогою різноманітних плагінів та модулів, що дозволяє розробникам додавати нові функціональності та підтримувати власні розробки.

Слабкі сторони Lumberyard:

– Обмеженій обсяг документації та спільнота:

Однією з викликів для розробників Lumberyard є менша кількість документації та обмежена активність спільноти порівняно з іншими популярними ігровими рушіями.

– Високі вимоги до апаратного забезпечення:

Створення графічно продвинутих ігор за допомогою Lumberyard може вимагати потужного апаратного забезпечення.

– Менша популярність в ігровій індустрії:

Lumberyard менше використовується в порівнянні з іншими рушіями, такими як Unity чи Unreal Engine, що може вплинути на доступність ресурсів та активність спільноти.

– Менша екосистема готових активів:

В порівнянні з іншими рушіями Lumberyard може мати менше готових активів та ресурсів для використання.

1.1.6 Source Engine

Source Engine - це ігровий рушій, розроблений компанією Valve Corporation. Він вперше з'явився в 2004 році та використовується для створення багатьох ігор, серед яких "Half-Life 2", "Portal", "Left 4 Dead" та інші. [9]

Сильні сторони Source Engine:

– Фізичний движок та геймплей:

Source Engine відомий своїм добре налаштованим фізичним движком, який взаємодіє з об'єктами в ігровому світі натуральним чином (рис 1.6). Він також підтримує реалістичні фізичні ефекти та вражаючий геймплей.

Рисунок 1.6 — Приклад гри на рушії Source Engine [4]

– Гнучкість та доступність:

Source Engine дуже гнучкий та легко використовується для створення різних жанрів ігор. Легка навігація по редактору та швидкий прототипування роблять його привабливим для розробників.

– Мультиплеєрна спрямованість:

Source Engine часто використовується для розробки мультиплеєрних ігор, таких як "Counter-Strike: Global Offensive" та "Team Fortress 2". Він має вбудовані засоби для створення різних мультиплеєрних геймплей-механік. [10]

– Моддінг та спільнота:

Source Engine активно підтримує моддінг і велику спільноту розробників. Гравці можуть створювати та встановлювати модифікації, що розширюють геймплей та вигляд ігор.

– Оптимізація:

Двигун добре оптимізований для роботи на різних конфігураціях комп'ютерів, що дозволяє запускати гру на різних платформах.

Слабкі сторони Source Engine:

– Графічні можливості:

У порівнянні з деякими іншими сучасними ігровими рушіями, Source Engine може мати менш вражаючі графічні можливості та менше деталізовані ефекти. [11]

– Вік:

Source Engine випущений у 2004 році, і хоча він пройшов кілька оновлень, він все ще базується на технологіях, розроблених у ті часи, що може вплинути на його конкурентоспроможність у порівнянні з більш новими рушіями.

– Обмежені можливості для розробки 3D-графіки:

У порівнянні з деякими сучасними рушіями, Source Engine може мати обмежені можливості для розробки вражаючих 3D-графічних ефектів.

1.2 Технічні особливості та переваги Unity

Unity підтримує кілька мов програмування, але основна із них - C#, визначається своєю чистотою та високою продуктивністю. Це дозволяє розробникам вибирати мову, яка найкраще відповідає їхнім потребам та вподобанням. [12]

Рушій забезпечує можливість розробляти ігри для різних платформ - ПК, консолей, мобільних пристроїв, віртуальної реальності, а навіть веб-браузерів. Це робить його ідеальним інструментом для розробки додатків на різних пристроях.

HDRP - це передова технологія 3D-рендерінгу в режимі реального часу, розроблена для забезпечення високоякісної графіки та безкомпромісної

продуктивності GPU. Завдяки гібридному підходу HDRP до візуалізації, який підтримує растеризацію, трасування променів і трасування шляху, ви можете виразити своє творче бачення на широкому діапазоні платформ, як показано на (рис 1.7). Крім того, HDRP пропонує розширені шейдери з підповерхневим розсіюванням, напівпрозорістю, райдужністю, анізотропією та мозаїкою в реальному часі, які можна налаштувати за допомогою Shader Graph. HDRP дає змогу відтворювати широкий спектр реалістичних непрозорих і прозорих поверхонь, від комбінації складних діелектричних і металевих поверхонь до більш важких обчислювальних матеріалів, таких як волосся, тканина, очі та багатошарові матеріали.

Імерсивне освітлення

Творці будь-якого рівня можуть чудово освітлювати свої сцени завдяки передовій системі освітлення HDR, що відповідає промисловому стандарту, із фізично правильним ослабленням світла, реальними одиницями (люкс, люмен, EV) для всього потоку освітлення та експозиції, а також розширеними типами світла та тінню. параметри.

HDRP пропонує низку методів, починаючи від традиційної растеризації (наприклад, відображення та планарні зонди), запікання світла, маршування променів і трасування променів (SSAO, SSGI, SSR, контакт або тіні з трасуванням променів), повнокадрове трасування траєкторії та численні анти - параметри псевдонімів, які можна легко перемикати або змішувати для масштабування продуктивності та точності на різних платформах.

Рисунок 1.7 - Імерсивне освітлення в Unity за допомогою HDRP [1]

Вражаюча атмосфера

Створюйте прекрасне небо та вражаючі хмари. Unity підтримує кілька типів неба, від простіших статичних градієнтів і небес HDRI до фізичних процедурних неб, які можуть автоматично імітувати різний час доби в реальному часі.

Щоб збільшити сприйняття глибини вашої сцени та створити більш щільну атмосферу, ви можете використовувати об’ємний туман і локальний об’ємний туман, які реалістично реагують на світло й тіні.

Об’ємні хмари та хмарні шари можуть допомогти вам збільшити динамічність вашого неба, щоб створити приголомшливі погодні умови, від вражаючих сходів сонця до штормового полудня, з мінімальним впливом на продуктивність.

Розширені кінематографічні ефекти

Створіть кінематографічний вигляд, який імітує артефакти реальних об’єктивів і плівок, як показано на (рис 1.8). Ефекти постобробки, як-от глибина різкості, віньєтування, розмиття в русі, розмиття та зернистість плівки, імітують фізичні параметри камери. Ви також можете отримати доступ до професійних, але зручних інструментів градації кольорів.

Удосконалена система відблисків на об’єктиві дозволяє створювати процедурні відблиски з автоматичною оклюзією. Він поставляється з багатьма

готовими до використання світловими сигналами, які допоможуть вам швидко розпочати роботу. [13]

Рисунок 1.7 - Створення кіноматографічних ефектів в Unity за допомогою HDRP [1]

Для трансляції та віртуального виробництва Graphics Compositor дозволяє комбінувати кілька вхідних даних рендерингу. Це означає, наприклад, що ви можете інтегрувати відео реального актора на зеленому екрані у віртуальний фон або навпаки.

Насичене та детальне середовище

Створюються світи реалістичної рослинності, як показано на (рис. 1.8), яка реагує на вітер, за допомогою деталей ландшафту, інструментів для малювання дерев та інтеграції SpeedTree. Досягніть високого рівня деталізації за допомогою мозаїки в реальному часі та ідеально закріпіть світ за допомогою напівпрозорості, карт тіней, тіней із трасуванням променів, контактних тіней, мікротіней, запеченого освітлення, а також оклюзії навколишнього середовища з маршем або трасуванням променів і глобального освітлення.

Використовуються наклейки, сітки та проектори, щоб швидко нанести на геометрію деталі, як-от тріщини, бруд або намальовані знаки. Місцево змінюйте

матеріали, розміщуючи калюжі, протікання, різьблення або покриття, щоб розбити повторювані візерунки. [2]

Рисунок 1.8 - Середовище створене в Unity за допомогою HDRP [1]

Універсальний конвеєр візуалізації (URP) — це зібраний конвеєр візуалізації, створений Unity. URP забезпечує робочі процеси, зручні для художників, які дозволяють швидко та легко створювати оптимізовану графіку, як показано на (рис. 1.9) на різних платформах, від мобільних до консолей високого класу та ПК.

Рисунок 1.9 - Середовище створене в Unity за допомогою URP [1]

Вбудований фізичний механізм Unity

Вбудований фізичний механізм Unity базується на PhysX 3.4 від NVIDIA, який є широко використовуваним і надійним фізичним проміжним програмним забезпеченням. Фізичний механізм Unity пропонує простий та інтуїтивно зрозумілий інтерфейс для створення твердих тіл, коллайдерів, з'єднань і тригерів і керування ними. Ви також можете використовувати API сценаріїв Unity для доступу та зміни налаштувань фізики та поведінки. Фізичний механізм Unity добре інтегрований із системами анімації, аудіо та інтерфейсу користувача та підтримує як 2D, так і 3D фізику, як показано на (рис. 1.10). [1]

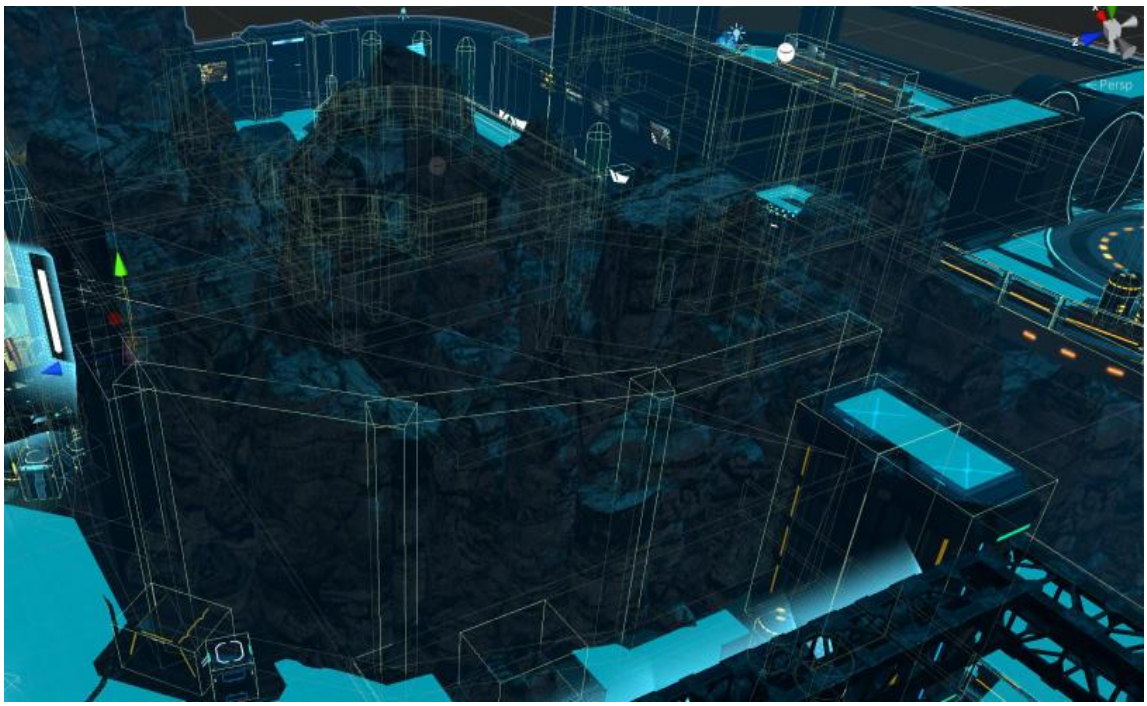

Рисунок 1.10 - Використання PhysX в Unity [1]

Unity Analytics надає наскрізне рішення для обробки даних і аналізу, щоб ви могли створити більш захоплюючий ігровий процес.

Аналітика дозволяє студіям зрозуміти продуктивність гри та поведінку гравців. Швидко збирайте статистичні дані за допомогою готових інформаційних панелей і візуалізацій на основі надійних даних у реальному часі. Потім використовуйте ці дані, щоб покращити свій проект і зробити його більш привабливим і привабливим для вашої аудиторії.

Висновки по розділу: отже, ігровий рушій Unity 3D має перевагу над деякими іншими безкоштовними рушіями завдяки своїм особливостям рендеру в реальному часі, обробці візуальної частини, можливістю аналізу даних на основі гри або поведінці гравців. Недоліком є те, щоб опанувати всі можливості рушія потрібно знати мови програмування, для написання скриптів та шейдерів до нього.

2 СТВОРЕННЯ ТА ОПТИМІЗАЦІЯ 3D-МОДЕЛІ ДЛЯ ІГРОВОГО РУШІЯ UNITY 3D

2.1 Огляд кроків для створення 3D-моделі

Для створення 3D-моделі було використано програмне забезпечення Blender-3d, в ньому було виконано основні кроки, такі як:

- Створення High-poly моделі;
- Створення Low-poly моделі;
- Створення UV-розгортки;
- Оптимізація моделі під ігровий рушій.

Також було використано програмний пакет Substance Painter для текстурування 3D-моделі.

2.2 Процес оптимізації 3D-моделі

Оптимізація 3D-моделей - це процес удосконалення та оптимізації геометричної та топологічної структури тривимірних об'єктів з метою покращення їх продуктивності, ефективності обробки, використання ресурсів і відтворення, зменшення обсягу пам'яті та оптимізації швидкодії графічного рендерингу в контексті програмного забезпечення, ігор або візуальних застосунків. [16]

Метою оптимізації є досягнення більш ефективного використання ресурсів обчислювального обладнання та покращення візуального відтворення об'єктів в тривимірному просторі, зменшення вимог до обчислювальних та графічних потужностей, а також забезпечення оптимальної продуктивності програм або ігор.

2.3 Пошук референсних зображень для створення блокінгу

Першим кроком створення 3D-моделі є підбір референсних зображень, для того щоб розуміти як повинна вона виглядати у реальному житті.

Поняття «референс» (ще використовують скорочення «реф») походить від англійського слова reference, яке перекладається як згадка, відсилка до джерела, довідкова інформація. Довідковою інформацією зазвичай виступають картинки або відео. Їх 3D спеціалісти використовують як зразки, на які треба рівнятися. Наприклад: – моделери шукають фотореференси того об'єкта, який вони будуть моделювати (фотографії архітектури, тварин, креслення автомобілів і т.д.), як показано на (рис. 2.1);

Було вирішено створювати середньовічну мортиру, усі референси були взяті з інтернету.

Рисунок 2.1 – Референсні зображення мортири [2]

2.4 Створення High-poly моделі

Для створення надточних моделей застосовується такий вид моделювання як високополігональні моделювання (High-poly) – створення точних копій об'єкта. Сучасні технології 3D дозволяють створити практично будь-який об'єкт, незалежно від рівня складності.

Високополігональне моделювання — це процес створення тривимірних моделей з великою кількістю багатокутників або вершин. Цей підхід використовується в галузі комп’ютерної графіки, також відомої як 3D-моделювання, для створення деталізованих об’єктів з високим ступенем реалістичності та деталізації.

Основні особливості високополігонального моделювання:

Однією з головних переваг високополігонального моделювання є можливість додавати велику кількість деталей і текстур, щоб надати об’єктам реалістичний вигляд, як показано на (рис. 2.2). Це особливо важливо для створення персонажів, архітектурних елементів і об’єктів у відомій інформації.

Рисунок 2.2 – Приклад високополігональної моделі [3]

Деякі художники використовують метод скульптинга для створення високополігональних моделей, як показано на (рис. 2.3). Це дозволяє їм деталізувати об’єкти, додаючи текстурні елементи, складки та спотворення для створення природного вигляду.

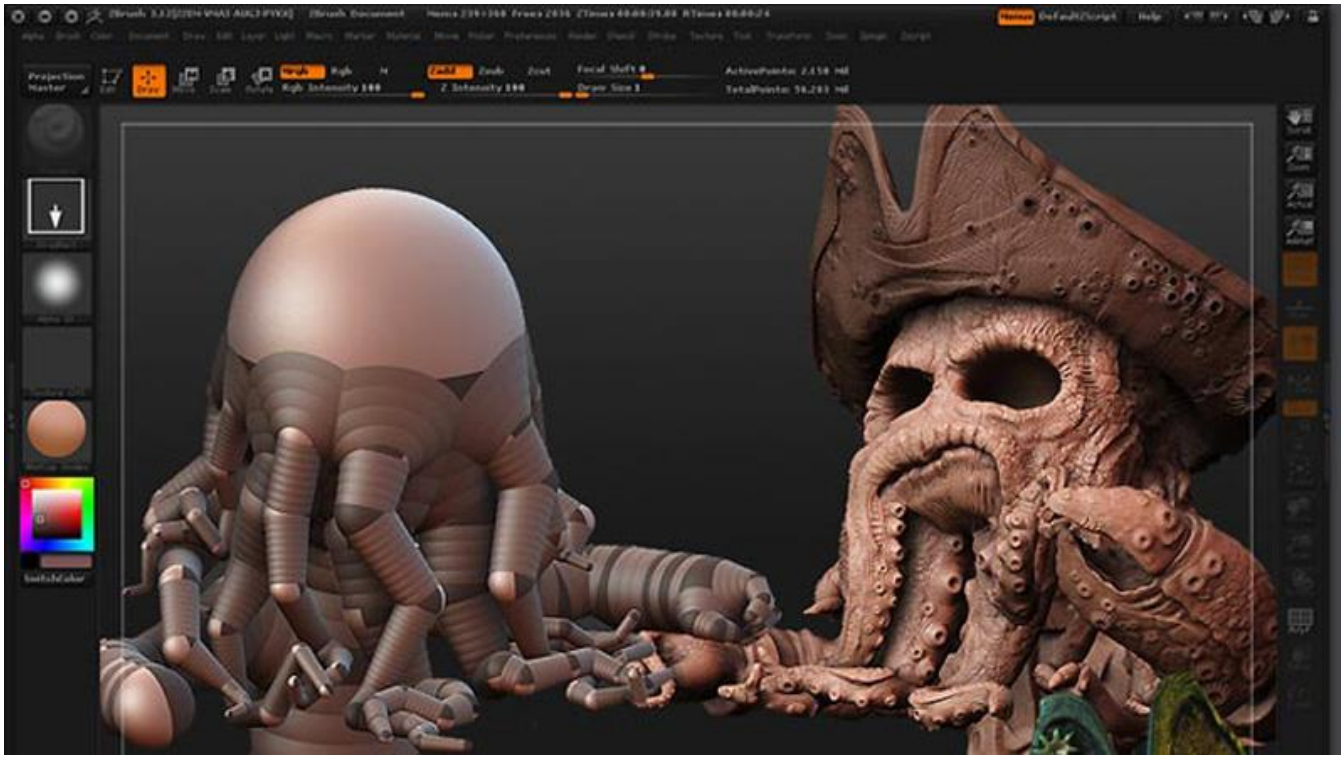

Рисунок 2.3 – Приклад методу скульптингу [3]

Високополігональні моделі часто використовуються для створення вражаючих візуалізацій, як показано на (рис. 2.4), оскільки вони можуть захопити кожну деталь, що додає реалістичності зображенню.

Рисунок 2.4 – Приклад створення візуалізації [5]

Високополігональні моделі дозволяють реалістично імітувати взаємодію об'єктів зі світлом, як показано на (рис. 2.5). Вони забезпечують точніше відтворення тіней, відблисків світла та взаємодію матеріалів, завдяки чому сцена виглядає більш природно.

Рисунок 2.5 – Приклад роботи зі світлом за допомогою використання високополігональних моделей [5]

Однак для створення та рендеринг високополігональних моделей може потребувати значних обчислювальних ресурсів. Для їх створення та обробки може знадобитися більше часу, а для використання може знадобитися потужне обчислювальне обладнання.

High-poly моделі відрізняються від Low-Poly моделей кількістю полігонів та полігональною сіткою.

High-poly модель була створена в програмному пакеті Blender3D.

На (рис 2.6, 2.7, 2.8) можна побачити полігональну сітку High-Poly моделі з різних ракурсів.

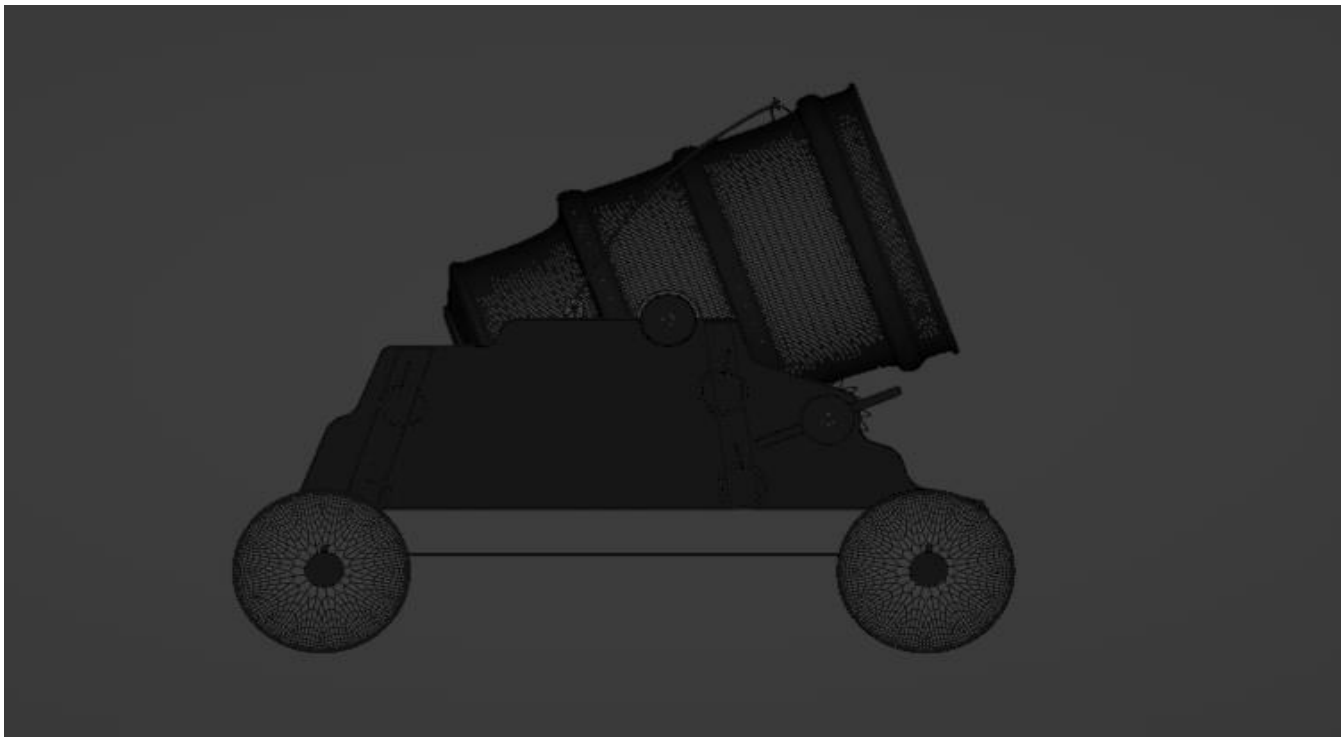

Рисунок 2.6 – Полігональна сітка High-Poly моделі на виді збоку

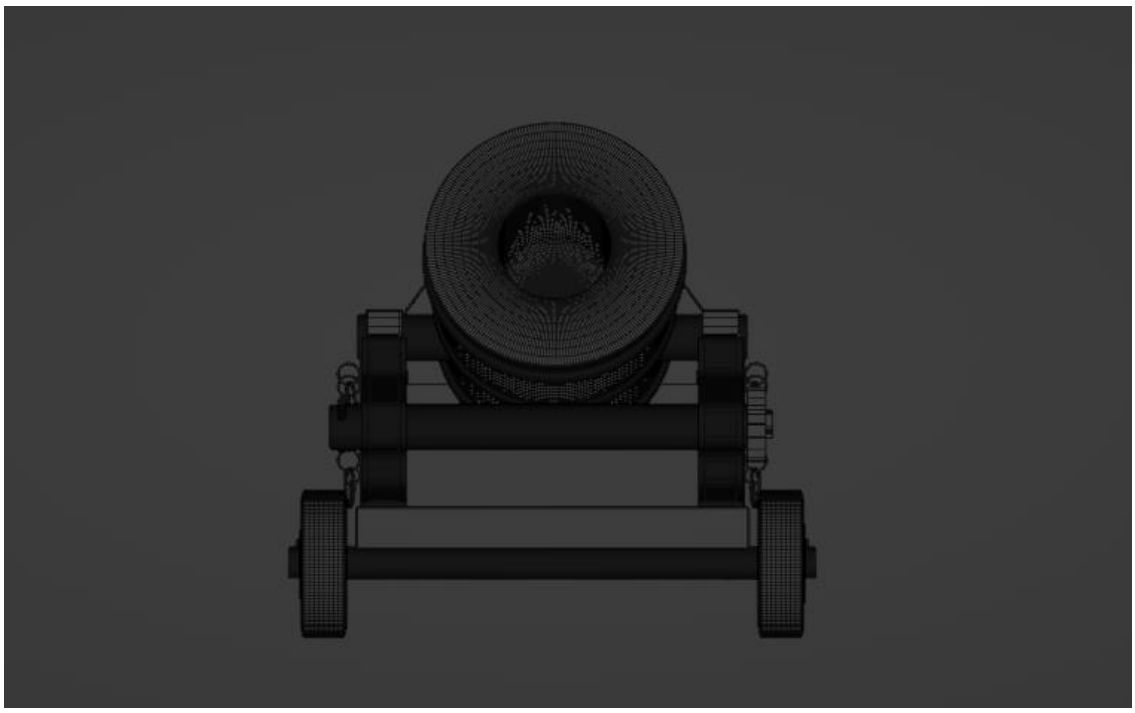

Рисунок 2.7 – Полігональна сітка High-Poly моделі на виді спереду

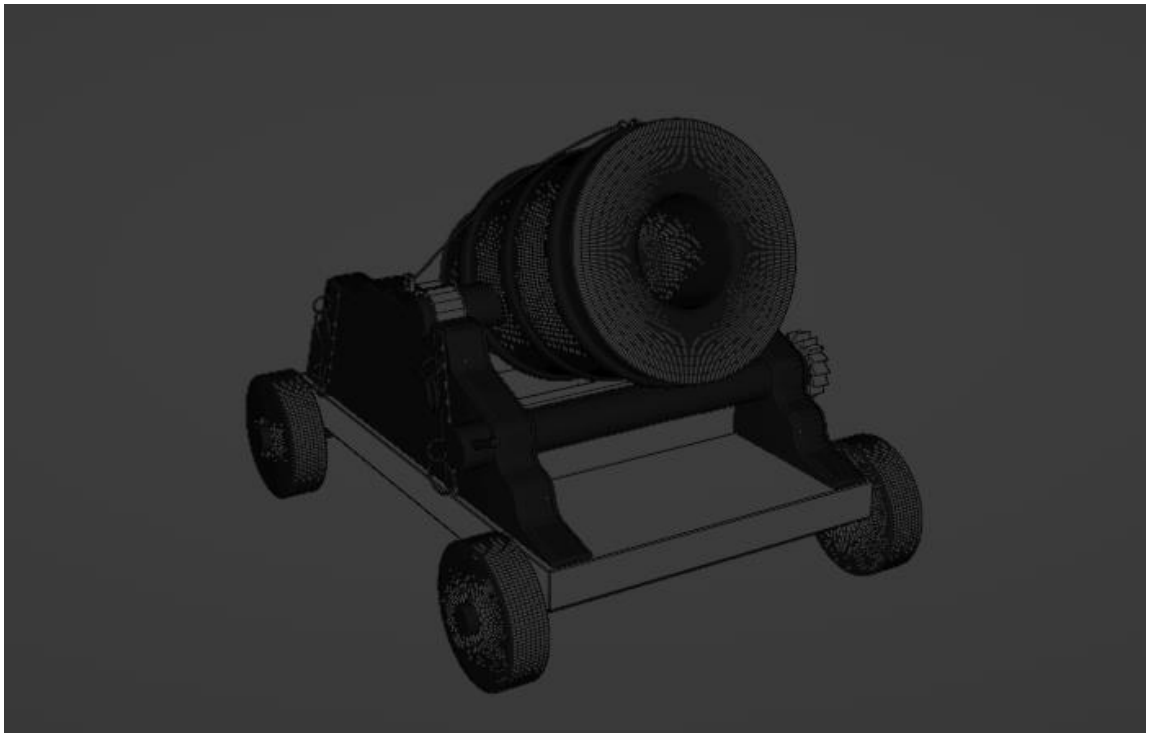

Рисунок 2.8 – Полігональна сітка High-Poly моделі в ізометрії

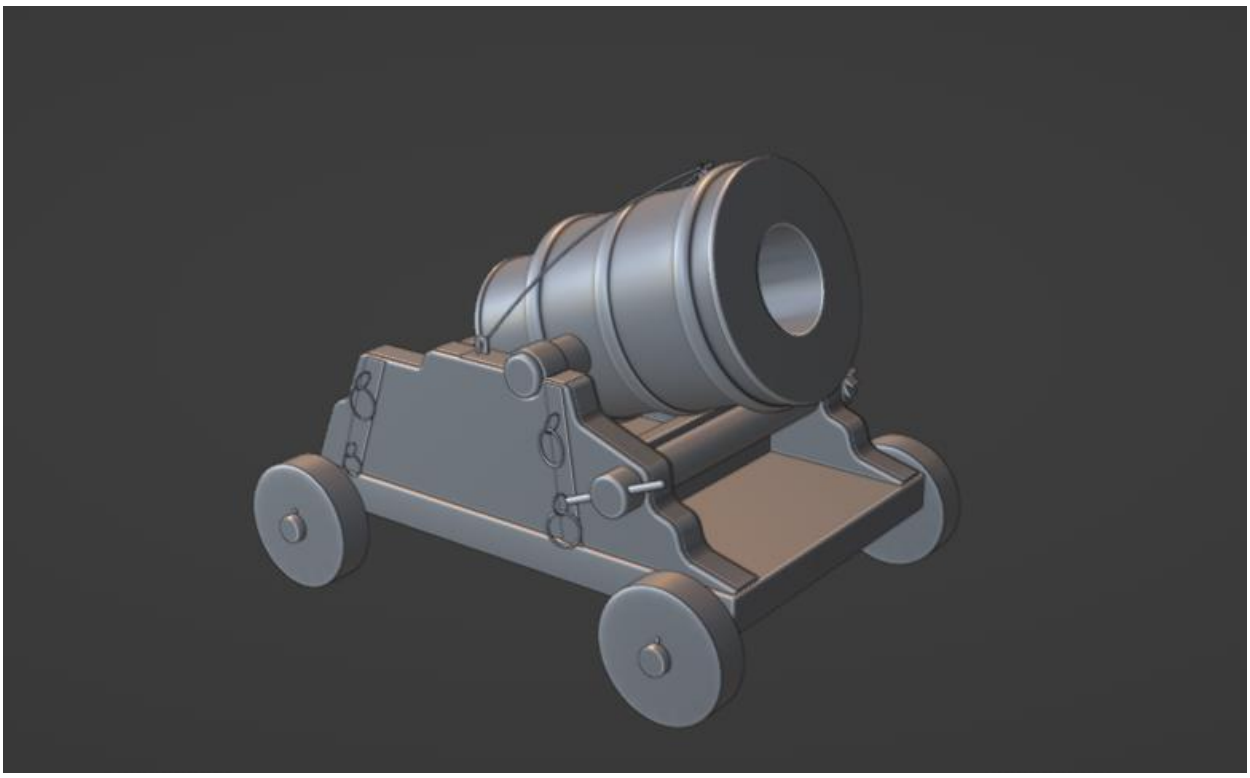

Рисунок 2.9 – Вид готової High-poly моделі

2.5 Створення Low-poly моделі

Низькополігональна модель (Low-poly) — це тривимірний об’єкт або, який містить невелику кількість багатокутників або точок у площині більш детальних 3D-моделей. Головна особливість low-poly — стратегічна зміна геометричної композиції для оптимізації продуктивності та ефективності обробки графіки. Цей підхід широко використовується в індустрії ігор, комп’ютерної графіки та візуалізації для збільшення швидкості візуалізації, зменшення ресурсів обробки та створення оптимальних візуальних ефектів. Зображення на різних платформах, включаючи мобільні пристрої.

Низькополігональне моделювання — це метод створення тривимірних моделей з обмеженою кількістю багатокутників або вершин. Цей підхід використовується в галузі комп’ютерної графіки для створення об’єктів із малою кількістю полігонів, що дозволяє оптимізувати ресурси та покращити продуктивність у реальному часі. Ключові особливості низькополігонального моделювання:

Низькополігональні моделі використовують менше полігонів, як показано на (рис. 2.10), що робить їх ідеальними для використання в іграх та інтерактивних програмах. Зменшення кількості полігонів допомагає оптимізувати ресурси, що особливо важливо у випадку обмежених обчислювальних можливостей, таких як мобільні пристрої або застарілі комп’ютери.

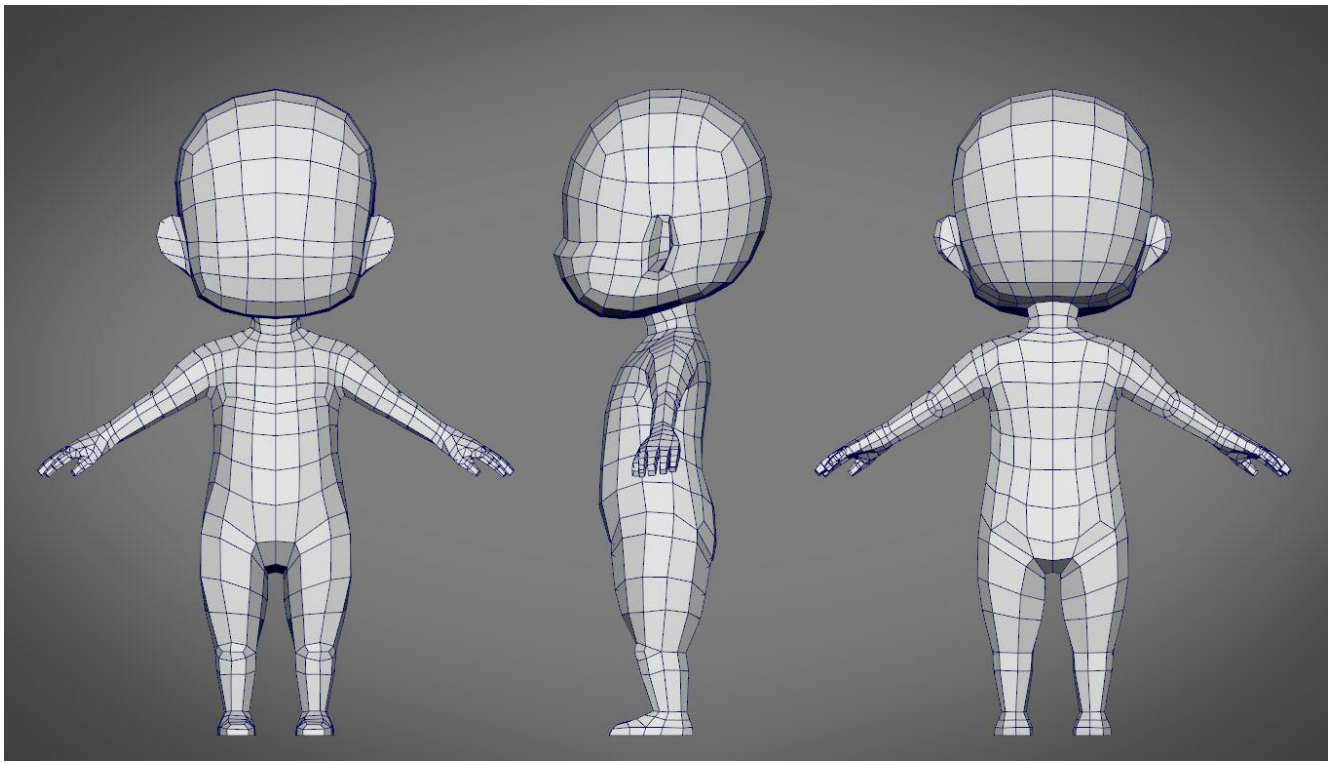

Рисунок 2.10 – Приклад полігональної сітки низькополігональної моделі [5]

Низькополігональні моделі часто використовуються для створення стилізованих і абстрактних об’єктів, як показано на (рис 2.11). Це дозволяє художникам виразно використовувати геометричні форми та кольори для створення унікального візуального ефекту.

Рисунок 2.11 – Приклад створення стилізованої сцени з низькополігональних об’єктів [6]

Оскільки низькополігональні моделі мають менше багатокутників, для їх рендерингу потрібно менше часу. Це особливо важливо в інтерактивних програмах, де велика кількість кадрів за секунду важлива для плавного відтворення.

Створення низькополігональних моделей може бути менш трудомістким процесом порівняно з високополігональним моделюванням. Особливо підходить для розробників таких проектів, де необхідно швидко створювати і текстурувати різні об'єкти.

Низькополігональні моделі часто використовуються для створення об’єктів у ретро-стилі, які нагадують графіку старих ігор чи мультфільмів, як показано на (рис. 2.12).

Рисунок 2.12 – Приклад створення ретро сцени з низькополігональних об’єктів [6]

Low-poly моделювання використовується в ігровій індустрії, архітектурній візуалізації, а також в індустрії анімації, як показано на (рис. 2.13). Він дозволяє досягти бажаного візуального ефекту і при цьому знизити навантаження на обчислювальні ресурси.

Рисунок 2.13 – Приклад створення UV-розгортки

для низькополігональної моделі [5]

Процец створення Low-poly моделі базується на оптимізації High-poly моделі, тобто із готової багатополігональної моделі зменшуємо її кількість полігонів. Результати можна побачити на (рис. 2.14, 2.15, 2.16, 2.17).

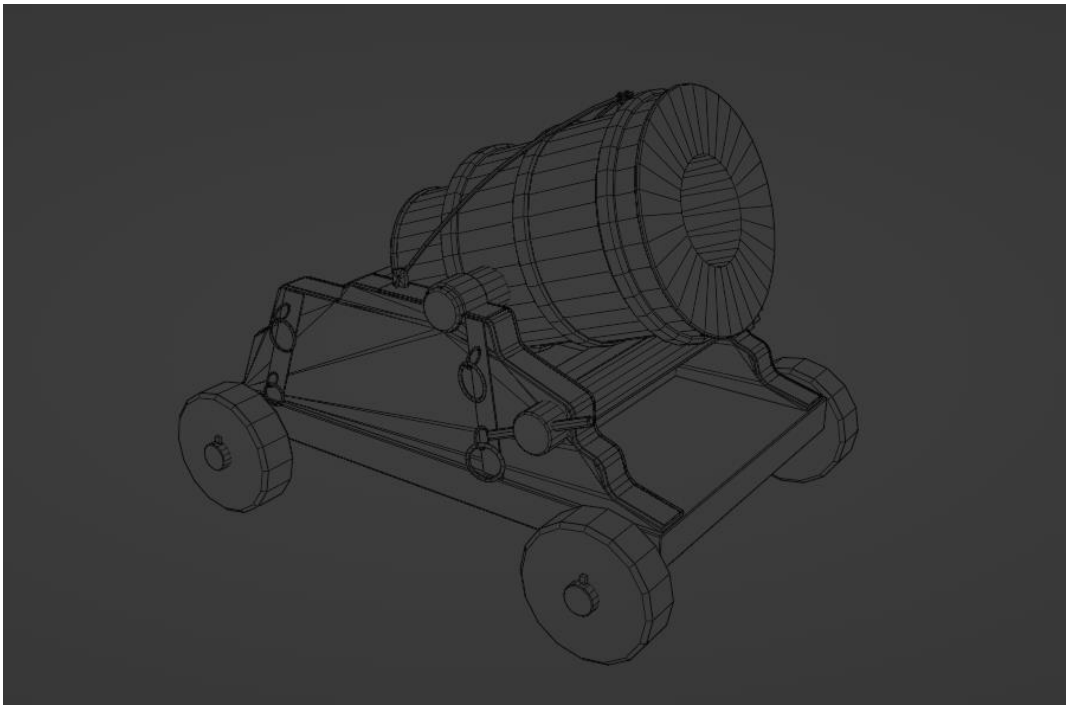

Рисунок 2.14 – Полігональна сітка Low-Poly моделі в ізометрії

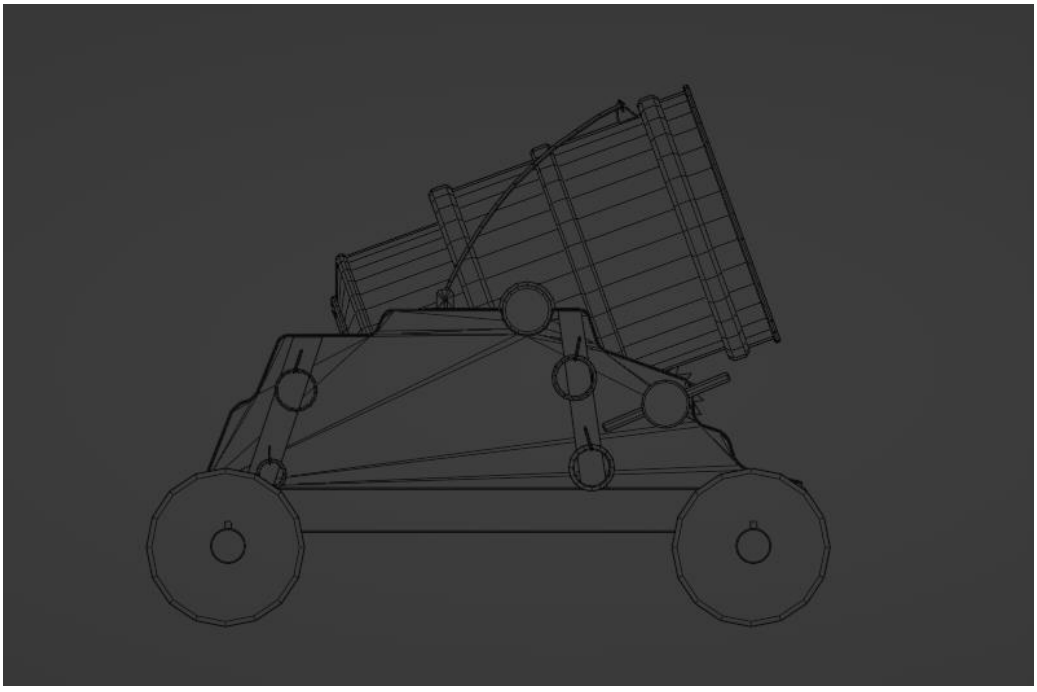

Рисунок 2.15 – Полігональна сітка Low-Poly моделі на виді збоку

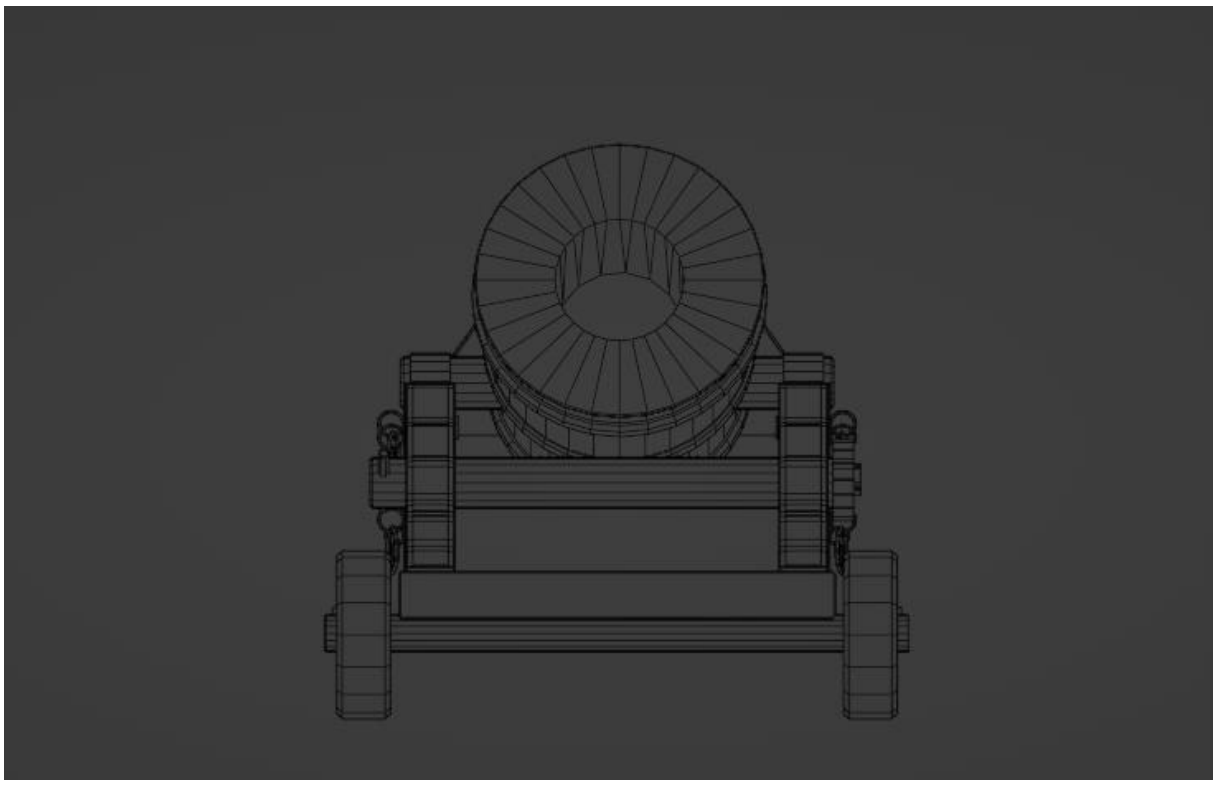

Рисунок 2.16 – Полігональна сітка Low-Poly моделі на виді збоку

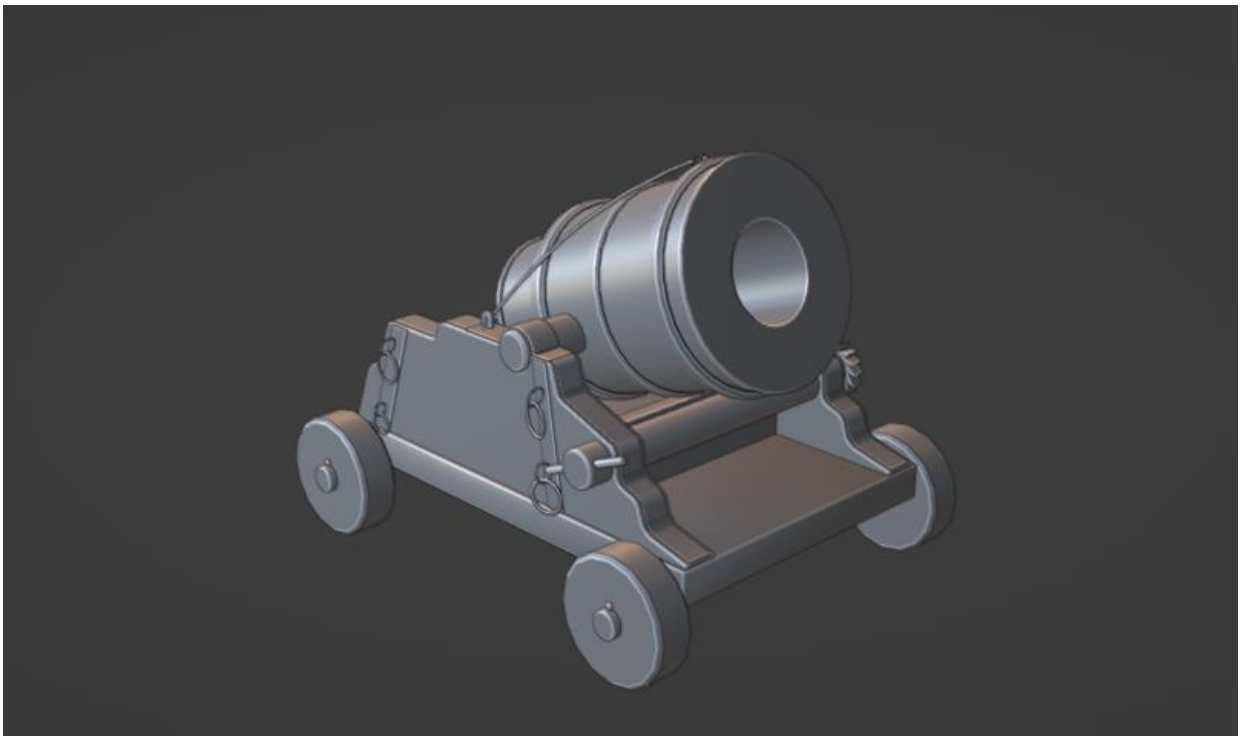

Рисунок 2.17 – Полігональна сітка Low-Poly моделі на виді збоку

2.6 Створення UV-розгортки для Low-Poly моделі

UV-розгортка (або UV-розгортання) — це процес візуалізації текстури, який передбачає розміщення поверхні тривимірного об’єкта у двовимірному просторі, як показано на (рис. 2.18). А 3D-моделювання та розгортання UV необхідні для належного холодного текстурування елементів керування об’єктами.

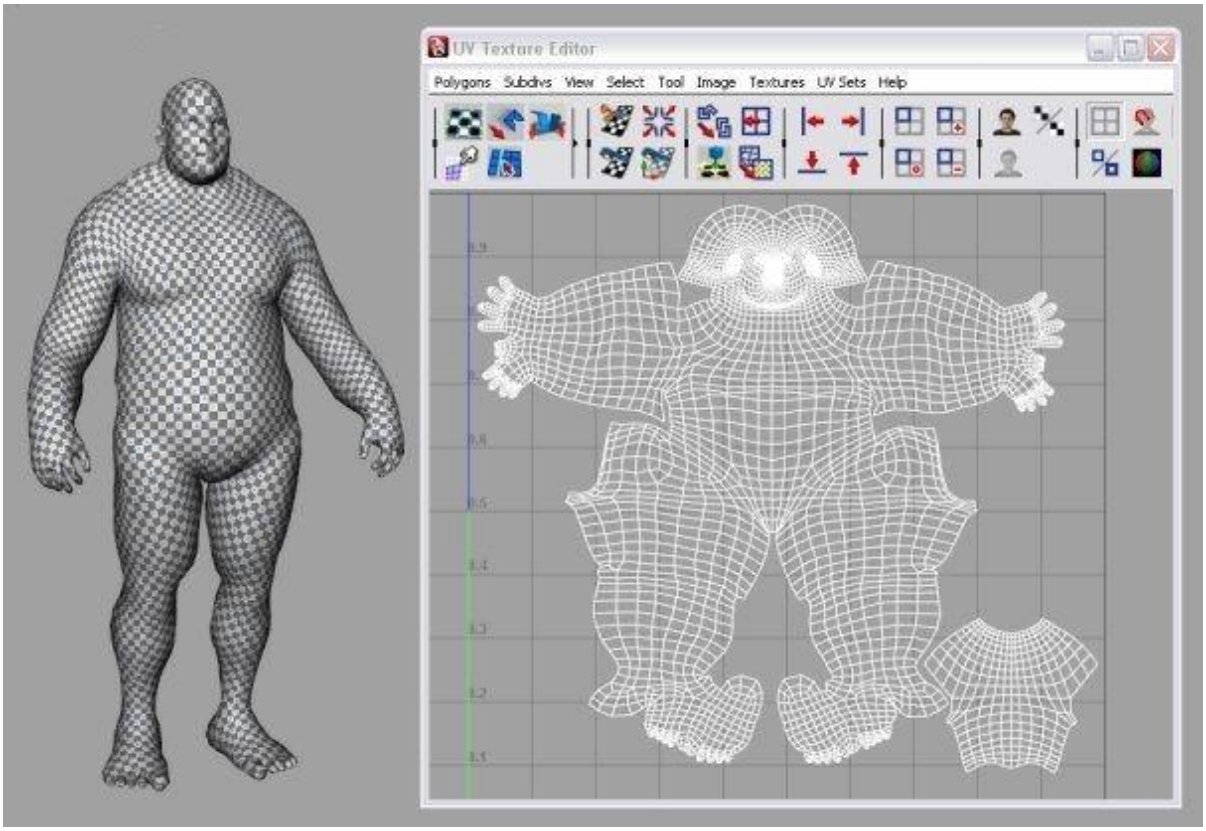

Рисунок 2.18 – Приклад створення UV-розгортки [5]

Кожна точка на поверхні 3D-моделі має відповідний плоский піксель на карті текстури, і UV-Explorer визначає, як взаємодіють два простори. Термін «UV» походить від осей просторових координат текстури, які позначають горизонтальну (U) і вертикальну (V) координати.

Мета UV-розгортки – максимально зберегти форму і деталі об’єкта на текстурній карті для забезпечення якісного і правильного відображення текстур на 3D-моделі під час рендеринга, як показано на (рис. 2.19). Успішна UV-розгортка допомагає художникам створювати реалістичні та естетично привабливі текстури

для використання в іграх, анімації, візуалізації та інших галузях, де важливий візуальний аспект 3D-моделювання.

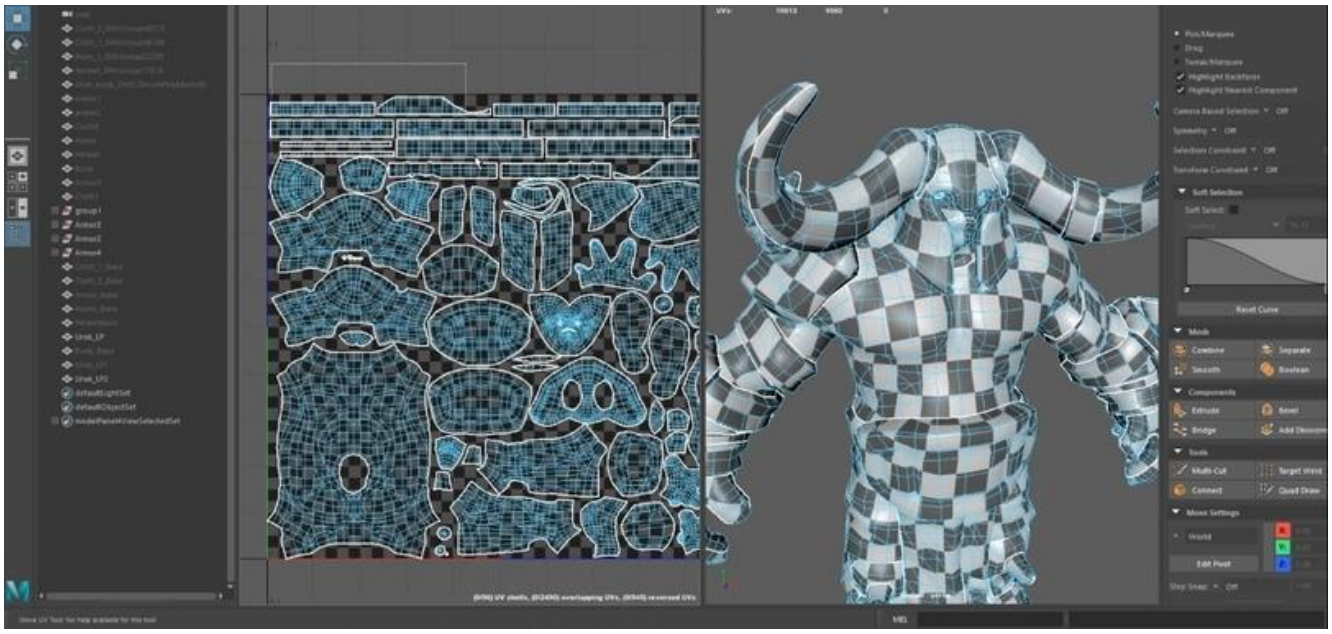

Рисунок 2.19 – Приклад створення UV-розгортки [5]

Процес створення UV-розгортки залежить від правильного створення Low-poly моделі. Для того, щоб розгорнути 3D-модель – необхідно проставити розрізи (seam) на об’єкті, як показано на (рис. 2.20). Після того як розгорнули модель, її 2D вид, як показано на (рис. 2.21) можна переглянути і за потреби переробити. Для того щоб оптимізувати і збільшити якість текстури на об’єкті використовують overlap або накладання одної частини моделі на іншу. Таким чином можна на однакові частини нанести одну й ту саму текстуру. Приклад функції overlap відображений на (рис. 2.21).

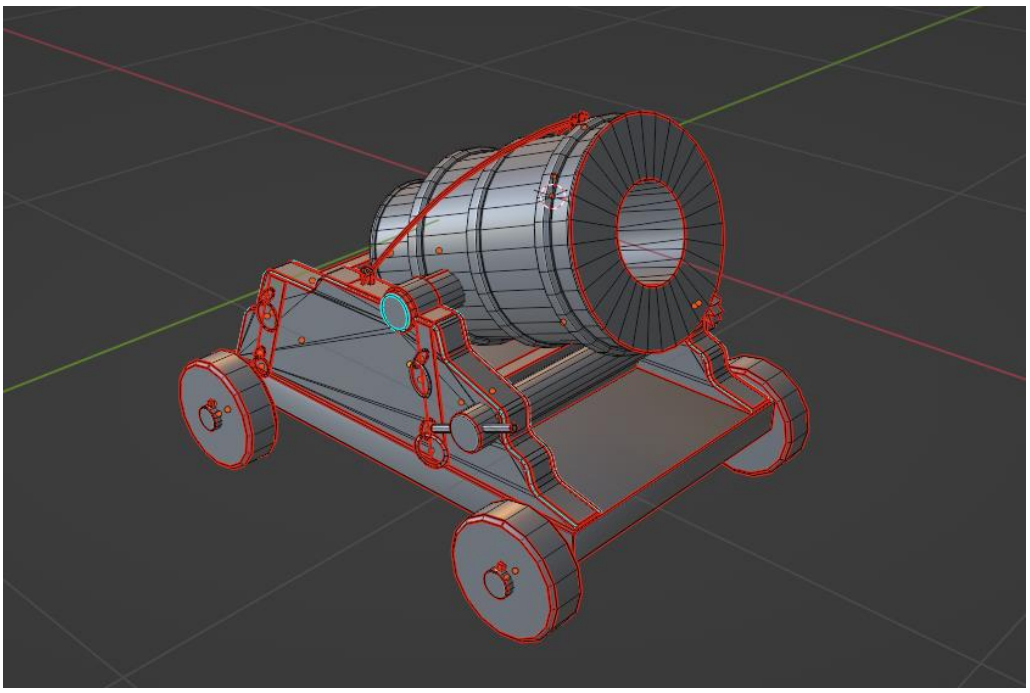

Рисунок 2.20 – Процес створення UV-розгортки

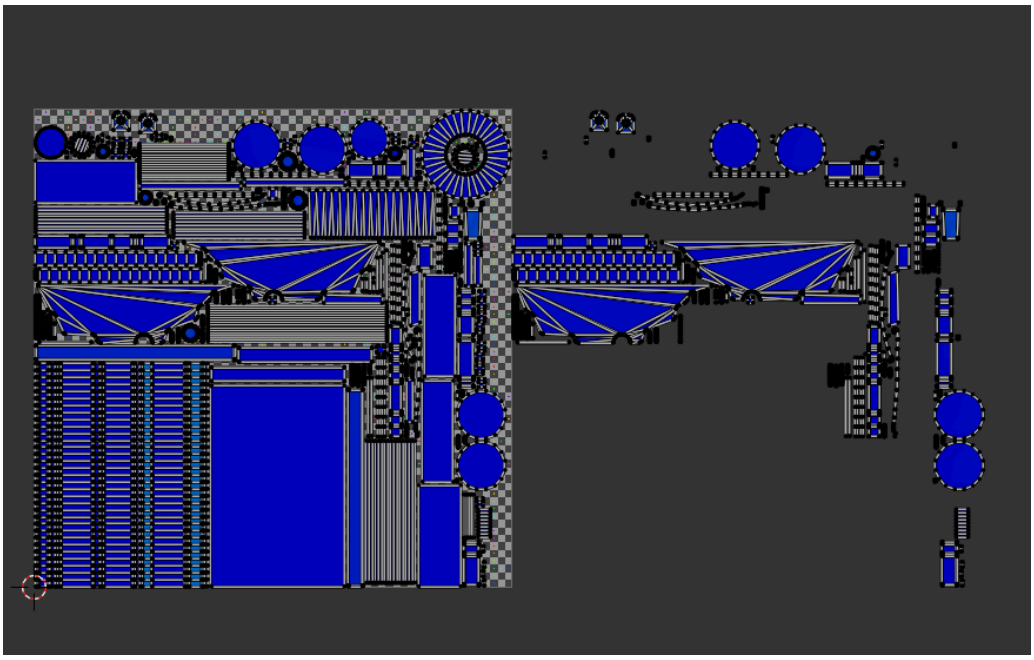

Рисунок 2.21 – Використання Overlap для збільшення якості текстури

2.7 Текстурування та запікання карт

Запікання карт виконується для перенесення деяких деталей із високополігональної моделі до її ретопології

Для того, щоб запікти карти High-poly з моделі на Low-poly модель бажано для початку їх розділити на частини, щоб не було помилок при запіканні, як показано на (рис. 2.22, 2.23).

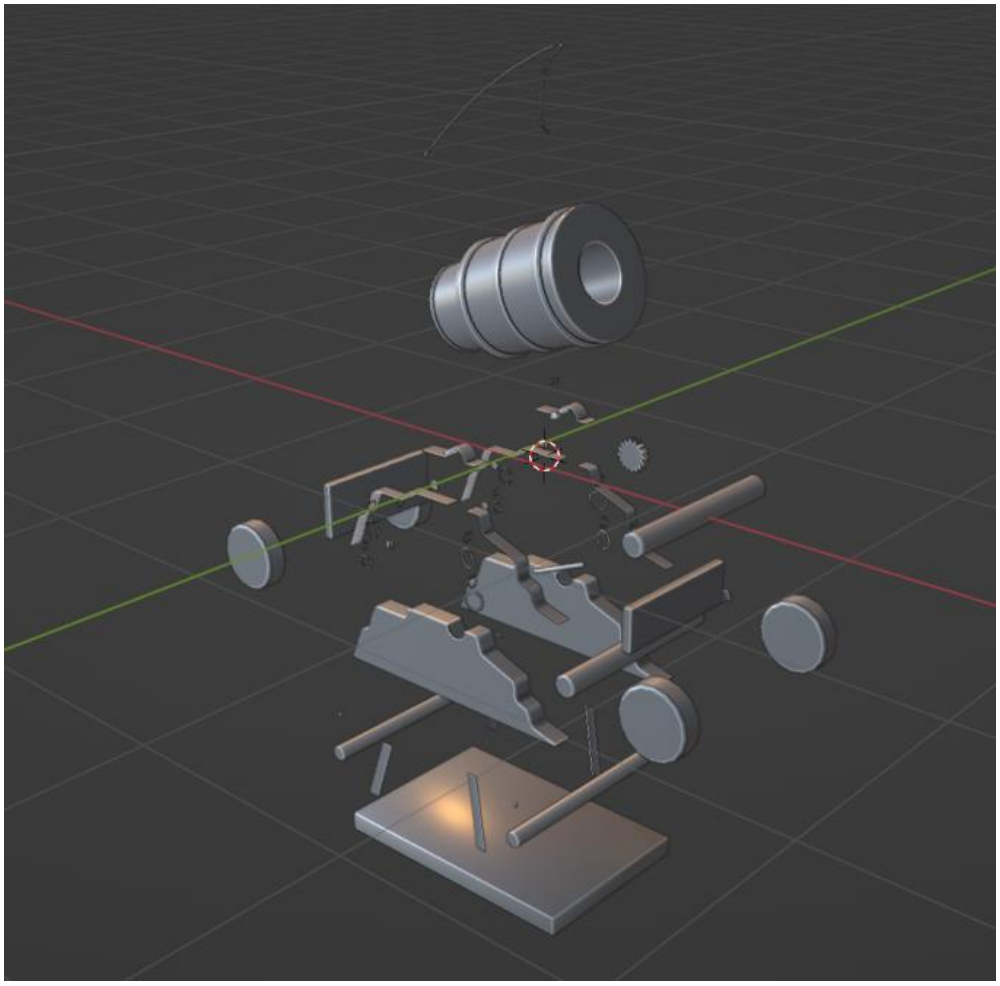

Рисунок 2.22 – Розділення моделі на частини для запікання карт

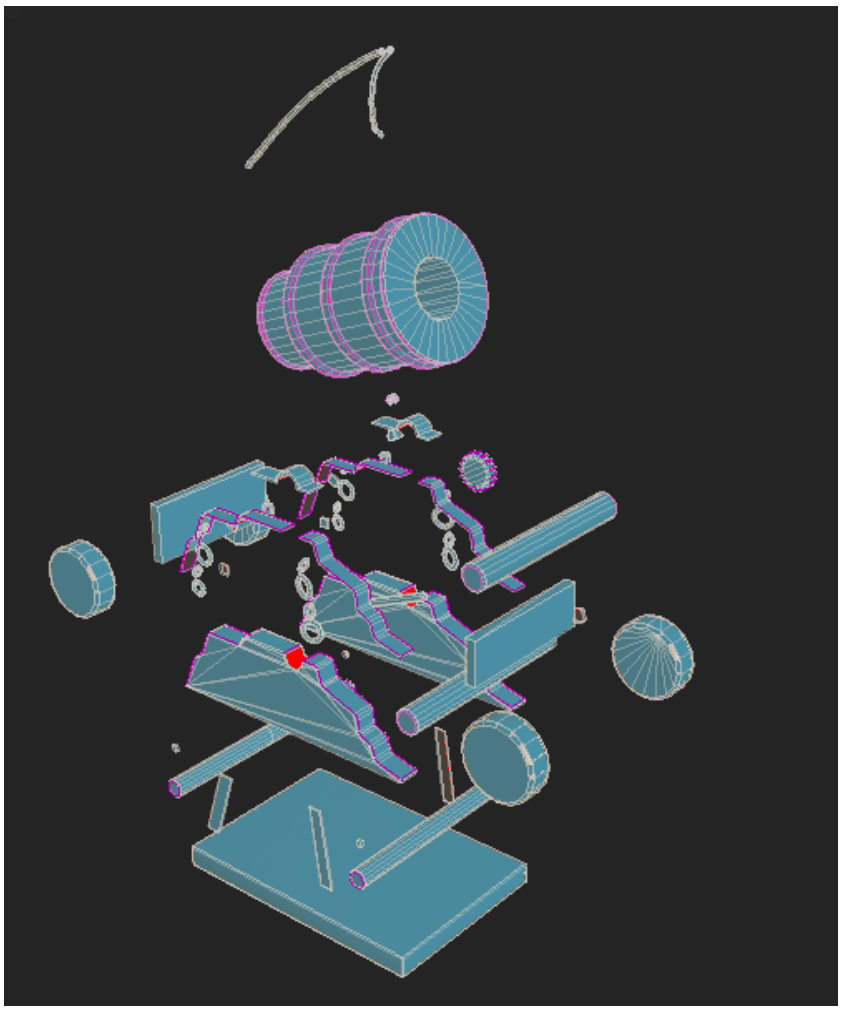

Рисунок 2.23 – Процес запікання текстурних карт

Процес оптимізації для текстурування 3D-моделі включає ряд заходів, спрямованих на підвищення продуктивності та ефективності використання текстур у графічних програмах, іграх або рендерингу. Основні аспекти оптимізації текстури в контексті 3D-моделювання включають в себе зменшення розміру текстур, наприклад для мобільних платформ, для того щоб зекономити ресурси та прискорити завантаження текстур.

Також використовують міпмапи (Mipmaps), як показано на (рис. 2.24), що дозволяє забезпечити оптимальне відображення текстур на різних відстанях та кутах огляду, зменшуючи артефакти та покращуючи продуктивність.

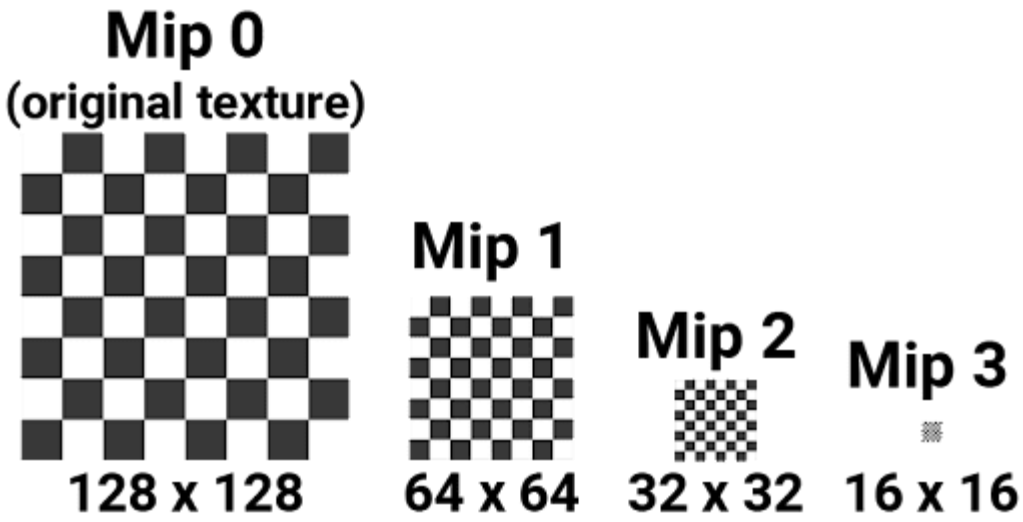

Рисунок 2.24 – Приклад використання mipmap [5]

Для оптимізації використовують текстурну атласізацію, як показано на (рис. 2.25), це процес об’єднання декількох текстур в одному атласі, що дозволяє зменшити кількість оброблюваних текстур.

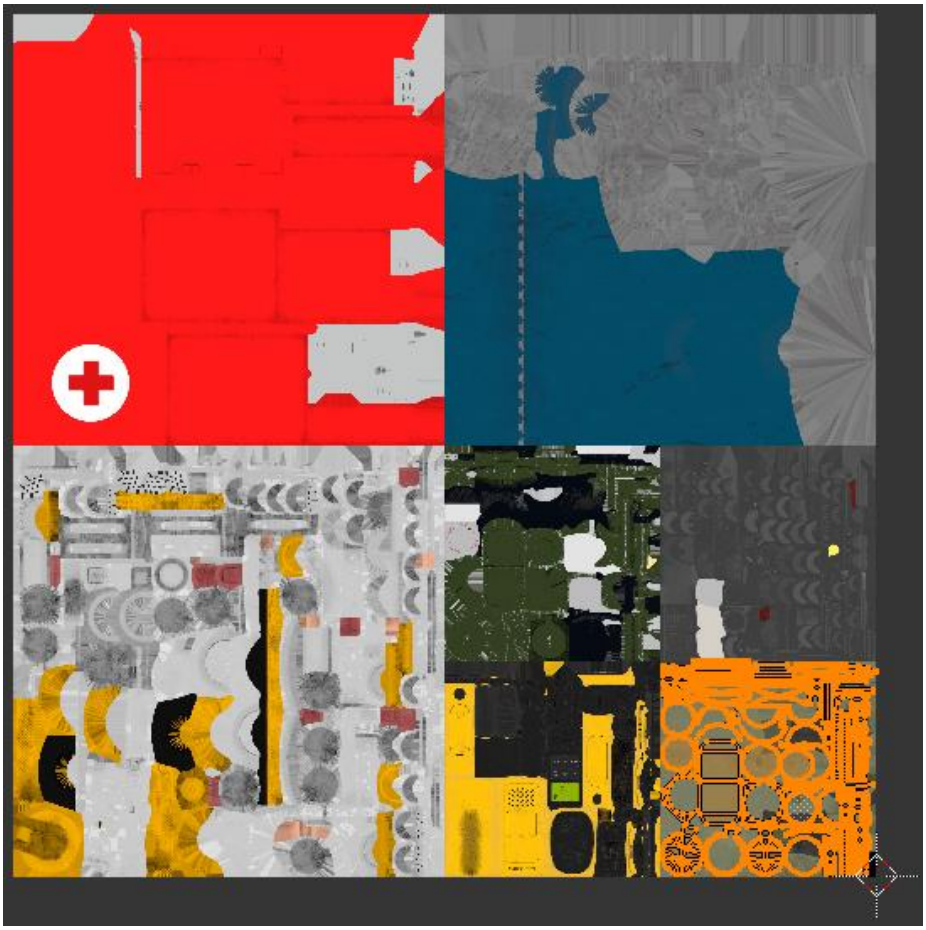

Рисунок 2.25 – Приклад використання текстурних атласів [6]

Дуже часто розробники ігор для оптимізації використовують лоди (LOD – Level of Detail) – це застосування різних рівнів деталізації текстур або моделей на різних відстанях, як показано на (рис. 2.26), що довзволяє підтримувати прийнятну якість графіки при зменшенні навантаження на обчислювальні ресурси.

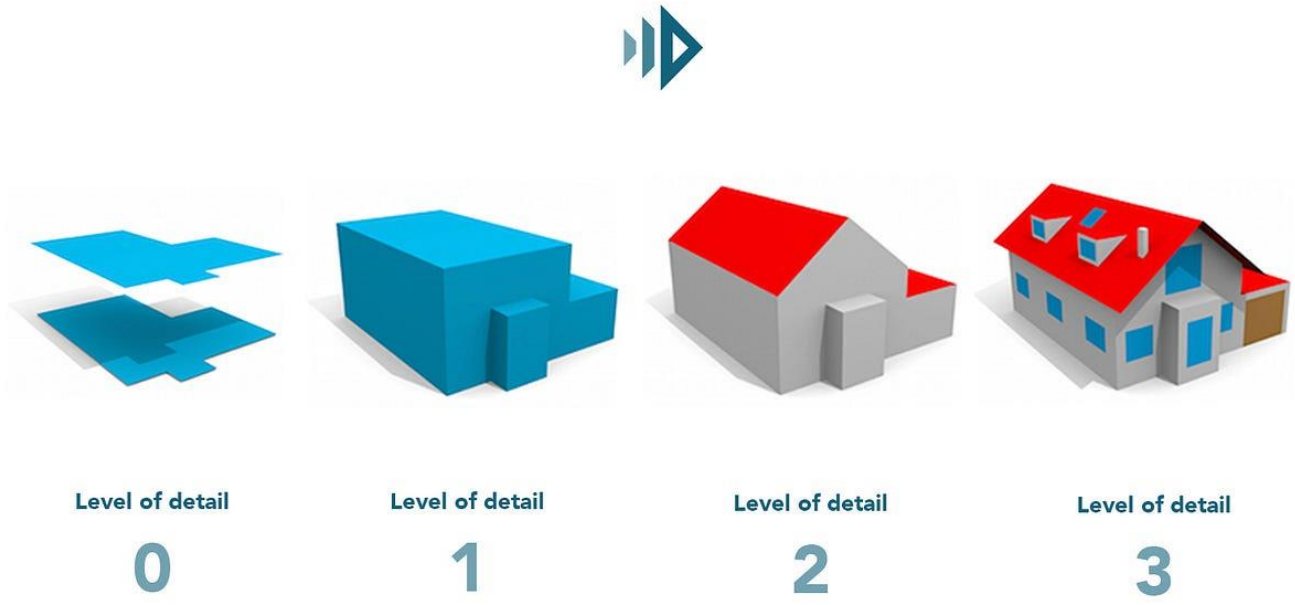

Рисунок 2.26 – Приклад використання LOD системи [5]

Також корисним способом є обрізка текстур, що допомагає уникнути використання непотрібних областей і зменшує обсяг необхідної відеопам’яті.

Оптимізація текстур у 3D-моделі дозволяє досягти балансу між високою якістю графіки та ефективністю використання обчислювальних ресурсів.

Текстурування зазвичай проводять у програмному пакеті Adobe Substance 3D Painter. Але можна використовувати, наприклад, Adobe Photoshop, а також є функції текстурування у програмних пакетах для моделювання.

Експортувати готові текстури в даному випадку треба в форматі Unity Universal Render Pipeline.

Результат створення текстур можна побачити на (рис 2.27, 2.28, 2.29).

Рисунок 2.27 – Результат створення текстур для оптимізованої 3D-моделі (вид збоку)

Рисунок 2.28 – Результат створення текстур для оптимізованої 3D-моделі (вид спереду)

Рисунок 2.29 – Результат створення текстур для оптимізованої 3D-моделі (вид в ізометрії)

2.8 Триангуляція та експорт 3D-моделі для подальшої роботи в ігровому рушії

Триангуляція 3D моделей є важливим етапом при розробці ігор. Триангуляція — це процес розбиття поверхні об'єкта на трикутники, які є базовими елементами для візуалізації в тривимірному просторі. Такий підхід дозволяє оптимізувати обчислення та полегшити обробку графіки у ігрових рушіях.

Для того, щоб триангулювати 3D-модель, в програмному пакеті Blender 3D, було використано модифікатор triangulate, завдяки якому всі полігони (чотирикутники) були перетворені на трикутники.

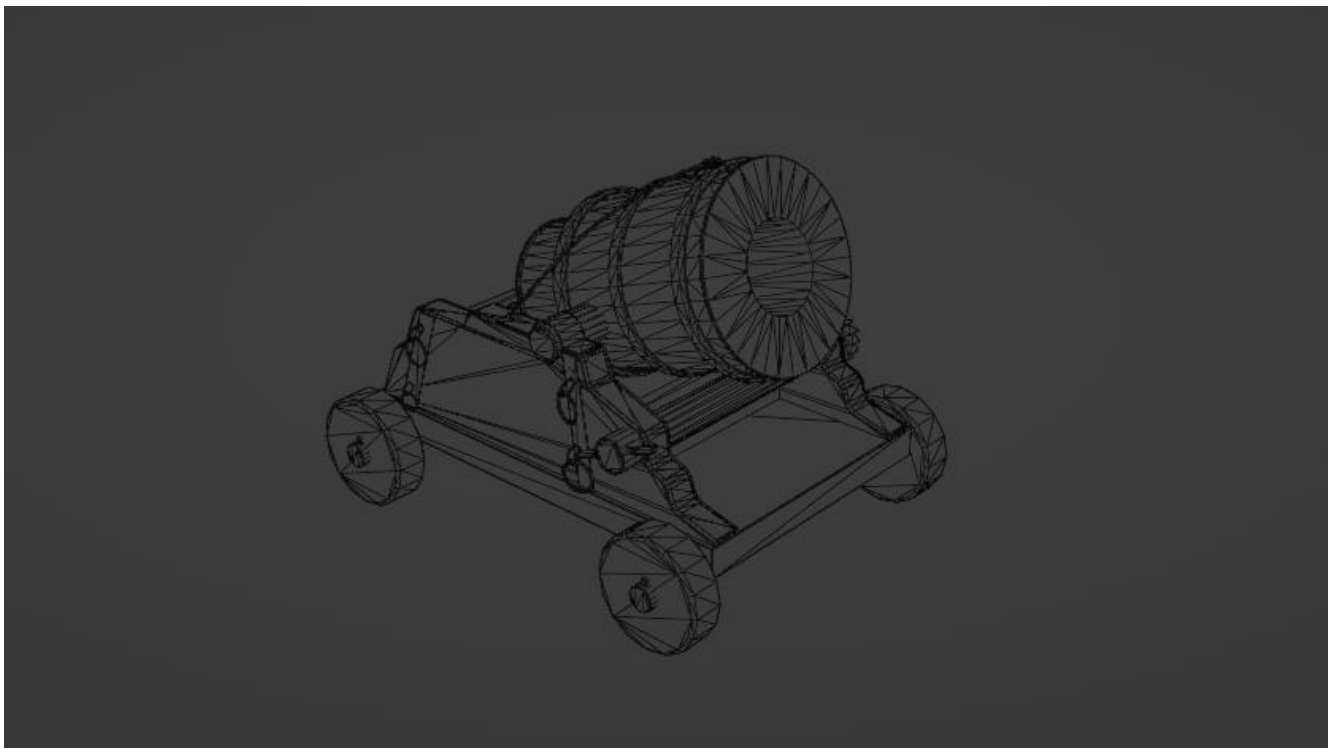

Рисунок 2.30 – Тріангульована 3D-модель

Для того, щоб експортувати модель для імпортування в ігровий рушій необхідно використати вбудований експорт в програмному пакеті Blender 3D. Експортуємо модель в форматі Autodesk Maya FBX.

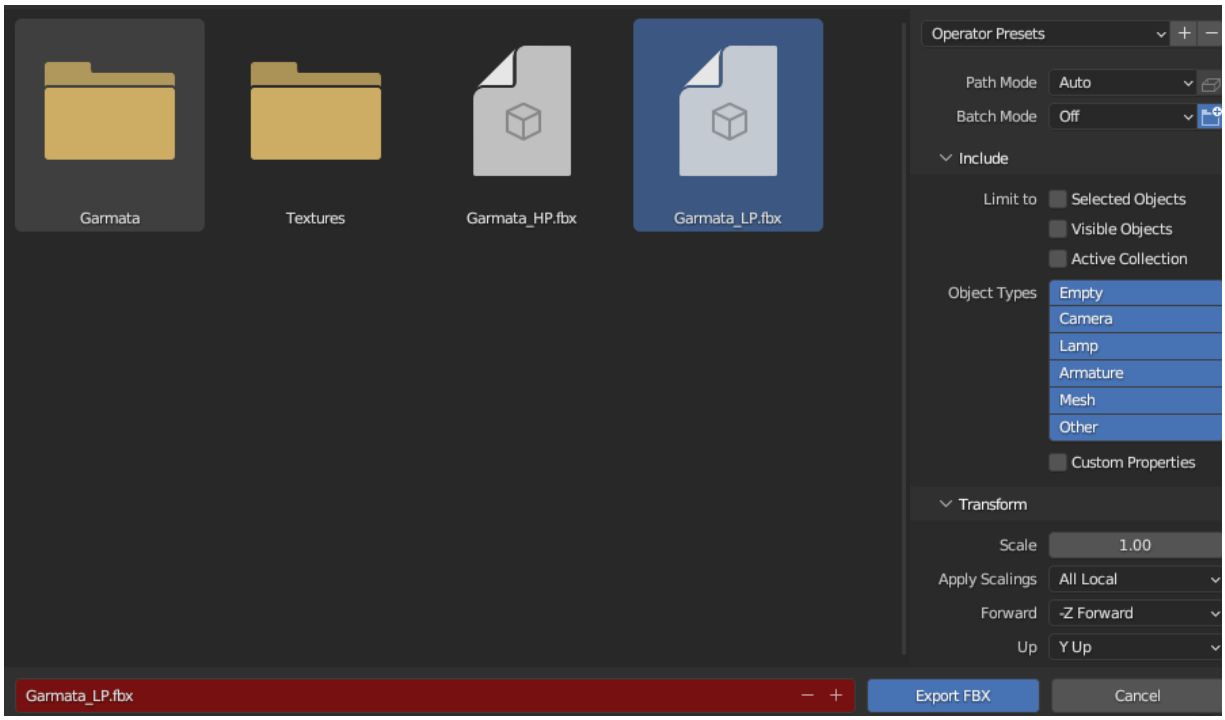

Рисунок 2.31 – Вікно експорту в програмному пакеті Blender 3D

2.9 Використання векторів та матриць в 3D-моделюванні

Вектор – це математичний об'єкт, що характеризується величиною і напрямом. Наприклад, у геометрії та у природничих науках вектор є спрямований відрізок прямий у евклідовому просторі (або на площині).

Вектори використовуються для представлення нормалей поверхонь. Це важливо для обчислення освітлення та створення реалістичних відображень. Також вектори використовуються для представлення координат точок у простор, як можна побачити на (рис. 2.32), кожна точка має свій вектор нормалі. Вектори можуть вказувати напрямок від одної точки до іншої.

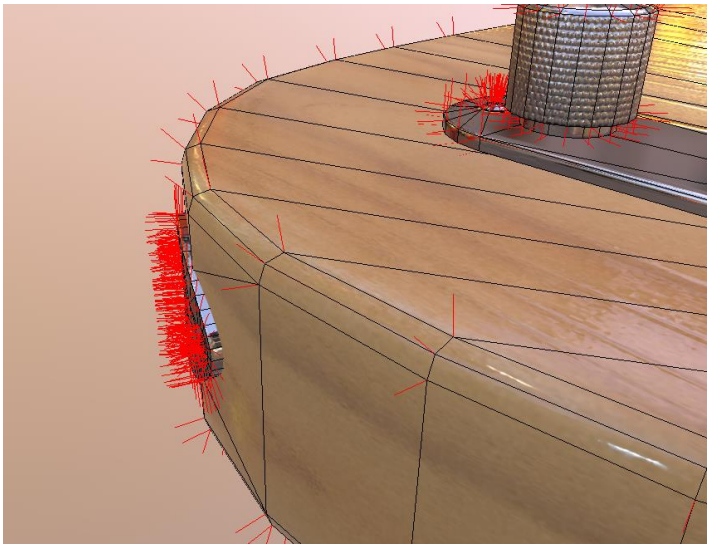

Рисунок 2.32 – Відображення векторів нормалей на 3D-моделі

Вектори можуть вказувати напрямок та положення камери, як можна побачити на (рис 2.33).

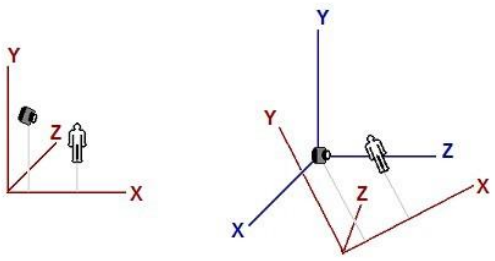

Рисунок 2.33 – Використання векторів при розміщенні камери

Використання матриць та векторів у 3D-моделюванні дозволяє точно та ефективно виконувати різні операції, що важливо для створення реалістичних та динамічних тривимірних сцен. [7]

Матриці використовуються для представлення трансляції (зміщення), обертання та масштабування об'єктів у 3D-просторі.

Для трансляції матриця виглядає наступним чином:

$$T = \begin{bmatrix} 1 & 0 & 0 & \Delta x \\ 0 & 1 & 0 & \Delta y \\ 0 & 0 & 1 & \Delta z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.1)$$

Для обертання навколо осі X:

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.2)$$

Для обертання навколо осі Y:

$$R_y = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.3)$$

Для обертання навколо осі Z:

$$R_z = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.4)$$

Де θ - кут обертання.

Для масштабування матриця виглядає наступним чином:

$$S = \begin{bmatrix} \mathrm{s_x} & 0 & 0 & 0 \\ 0 & \mathrm{s_y} & 0 & 0 \\ 0 & 0 & \mathrm{s_z} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.5)$$

Де s_x, s_y, s_z – коефіцієнти масштабування по відповідних осях

Висновки до розділу: отже, для того, щоб оптимізувати 3D-модель для використання в ігровому рушії необхідно виконати наступні етапи:

– створення високополігональної моделі;

- створення низькополігональної моделі і оптимізація;
- створення правильної UV-розгортки;
- створення текстур;
- тріангуляція та експорт моделі;

3 СТВОРЕННЯ СЦЕНИ ТА ПОРІВНЯННЯ ОПТИМІЗОВАНОЇ МОДЕЛІ З ВИСОКОПОЛІГОНАЛЬНОЮ МОДЕЛЛЮ В ІГРОВОМУ РУШІЇ UNITY 3D

3.1 Створення сцени для використання 3D-моделі в ігровому рушії Unity 3D

Для того, щоб завантажити 3D-модель в Unity 3D для початку потрібно створити або завантажити готову ігрову сцену. Вона дозволить побачити як навантажуються ресурси комп’ютеру.

Сцену для цієї роботи було завантажено з офіціального сайту ассетів для Unity 3D. До неї лише потрібно було додати штучне світло – воно потребує менше ресурсів комп’ютеру та зменшує навантаження.

В цій сцені використовуються системи LOD, для того щоб зменшити навантаження на комп’ютер, шляхом застосування різних рівнів деталізації для текстур на різних відстанях (рис. 3.1).

Рисунок 3.1 - Завантажена сцена для Unity 3D

Імпортуємо готову оптимізовану модель в форматі Autodesk Maya FBX. Та додаємо її до сцени.

Ця сцена в Unity використовує формат URP (Universal Render Pipeline), тому матеріал використовуємо спеціально під нього. Кожен матеріал має свій шейдер.

Шейдер в Unity 3D — це програмний код, написаний на мові шейдерів (зазвичай HLSL або Cg), який визначає, як поверхня чи об’єкт взаємодіє зі світлом і тінями, як він має виглядати та які спеціальні ефекти він має відображати. Шейдери в Unity 3D використовуються для керування графічними аспектами об’єктів у грі та дозволяють реалізувати різні візуальні ефекти.

Шейдери в Unity 3D використовуються для досягнення різноманітних візуальних ефектів, таких як відображення тіней, текстур, блиску, водяних ефектів та інших аспектів графіки у реальному часі. Шейдери дозволяють розробникам створювати унікальні та захоплюючі візуальні враження для своїх ігор та додатків.

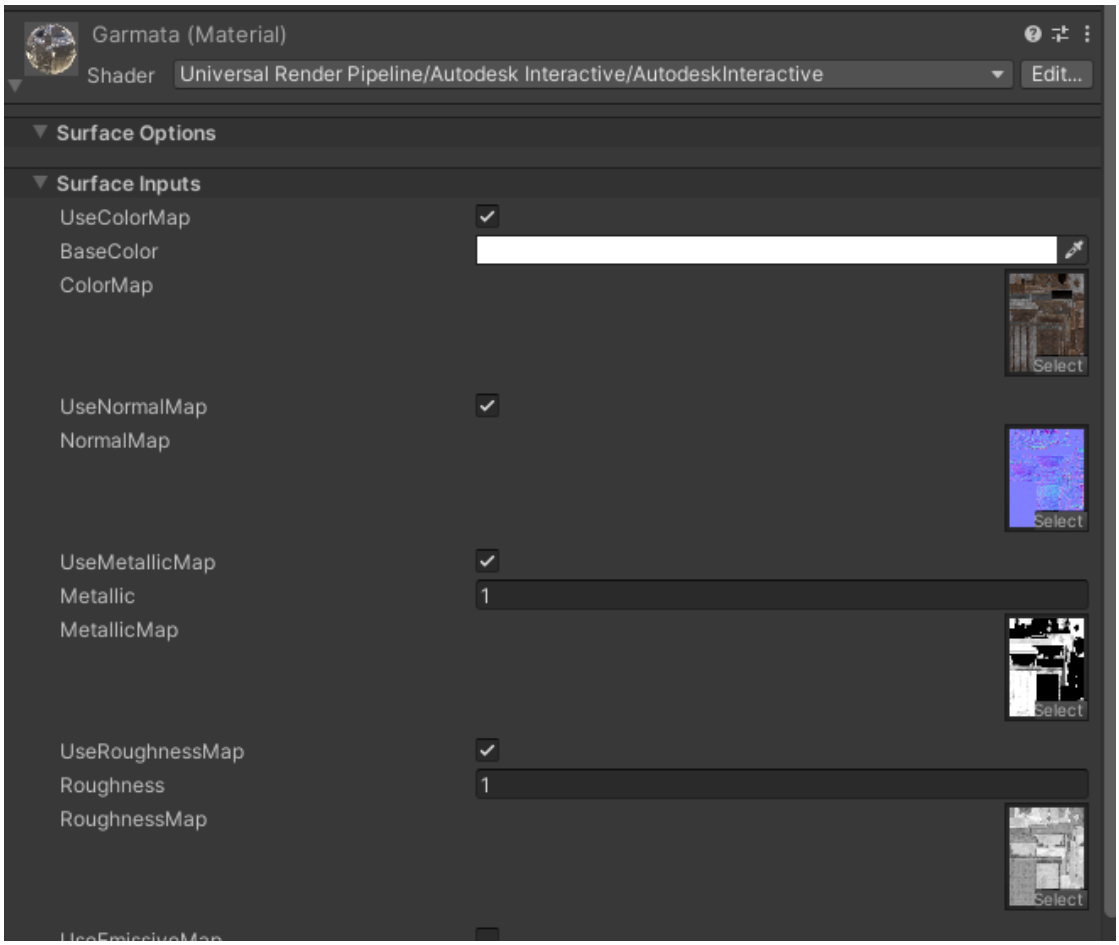

Рисунок 3.2 – Вікно налаштування шейдеру для матеріалу

Рисунок 3.3 – Імпортована модель в ігровому рушії Unity 3D

3.2 Порівняння високополігональної моделі з оптимізованою моделлю на комп’ютерах з різними технічними характеристиками

Для порівняння було використано два комп’ютера, які відрізняються між собою обсягами відеопам’яті та трохи різною частотою роботи центрального процесора.

Спочатку було проведено порівняння на основі однієї моделі в сцені.

Результати можна побачити в таблиці 1.

Таблция 1 - Порівняння оптимізованої і високополігональної моделі на комп’ютерах з різними технічними характеристиками

Характеристики комп’ютера	№1: Ryzen 5 5600X 4 ГГц GTX 1660 Super 6 Гб ОЗУ 32 Гб DDR4, 3600 1 ТБ SSD	№2: Ryzen 5 5600x, 4.4ГГц RTX 3080Ti 12гб ОЗУ 32Gb DDR4, 3600 1ТБ SSD
Оптимізована модель	69,9 FPS (14,3 ms) – 80,8 FPS (12.4 ms)	129,7 FPS (7,7 ms) – 180,3 FPS (5.5 ms)
Високополігональна модель	49,4 FPS (20,3 ms) – 55,5 FPS (18.0 ms)	137,6 FPS (7,3 ms) – 123,0 FPS (8.1 ms)

Рисунок 3.4 – Еталоний тест продуктивності оптимізованої моделі на комп’ютері №2

Рисунок 3.5 – Еталоний тест продуктивності оптимізованої моделі на комп’ютері №2

Рисунок 3.6 – Еталоний тест продуктивності високополігональної моделі на комп’ютері №2

Рисунок 3.7 – Еталоний тест продуктивності високополігональної моделі на комп’ютері №2

Рисунок 3.8 – Еталоний тест продуктивності високополігональної моделі на комп’ютері №1

Рисунок 3.8 – Еталоний тест продуктивності високополігональної моделі на комп’ютері №1

Рисунок 3.9 – Еталоний тест продуктивності оптимізованої моделі на комп’ютері №1

Рисунок 3.10 – Еталоний тест продуктивності оптимізованої моделі на комп’ютері №1

3.3 Порівняння шести оптимізованих і високополігональних моделей на комп’ютерах з різними технічними характеристиками

Для того, щоб більше навантажити систему, було додано по шість високополігональних і оптимізованих моделей в сцену. Через це можна отримати інші результати, бо сцена буде потребувати більше відеопам’яті комп’ютера.

Результати можна побачити в таблиці 2.

Таблция 2 - Порівняння шести оптимізованих і високополігональних моделей на комп’ютерах з різними технічними характеристиками

	№1: Ryzen 5 5600X 4 ГГц GTX 1660 Super 6 Гб ОЗУ 32 Гб DDR4, 3600 1 ТБ SSD	№2: Ryzen 5 5600x, 4.4ГГц RTX 3080Ti 12гб ОЗУ 32Gb DDR4, 3600 1ТБ SSD
Характеристики комп’ютера		
Оптимізована модель	68,2 FPS (14,7 ms) – 85,7 FPS (11.7 ms)	120,0 FPS (8,3 ms) – 167,2 FPS (6.0 ms)
Високополігональна модель	12,2 FPS (81,9 ms) – 19,5 FPS (51.3 ms)	63,6 FPS (15,7 ms) – 66,8 FPS (15.0 ms)

Рисунок 3.11 – Еталоний тест продуктивності шістьох оптимізованих моделей на комп’ютері №1

Рисунок 3.12 – Еталоний тест продуктивності шістьох оптимізованих моделей на комп’ютері №1

Рисунок 3.13 – Еталоний тест продуктивності шістьох високополігональних моделей на комп’ютері №1

Рисунок 3.14 – Еталоний тест продуктивності шістьох високополігональних моделей на комп’ютері №1

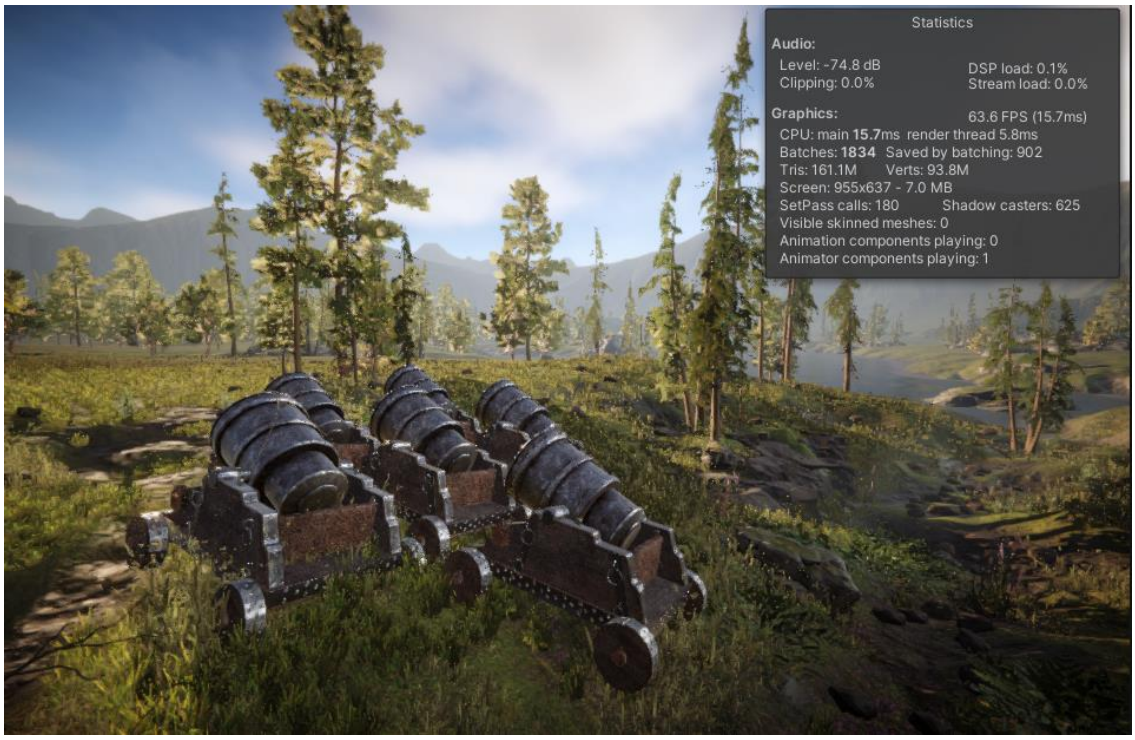

Рисунок 3.15 – Еталоний тест продуктивності шістьох високополігональних моделей на комп’ютері №2

Рисунок 3.16 – Еталоний тест продуктивності шістьох високополігональних моделей на комп’ютері №2

Рисунок 3.17 – Еталоний тест продуктивності шістьох оптимізованих моделей на комп’ютері №2

Рисунок 3.18 – Еталоний тест продуктивності шістьох оптимізованих моделей на комп’ютері №2

3.4 Порівняння оптимізованої і високополігональної моделі з вимкненою сценої на комп’ютерах з різними технічними характеристиками

Для того, щоб подивитися як оптимізована модель, було вимкнено сцену на якій вона знаходилась, щоб інші чинники не навантажували відеопам’ять комп’ютера.

Результати можна побачити в таблиці 3.

Таблция 3 - Порівняння оптимізованої і високополігональної моделі з вимкненою сценою на комп’ютерах з різними технічними характеристиками

	№1: Ryzen 5 5600X 4 ГГц GTX 1660 Super 6 Гб ОЗУ 32 Гб DDR4, 3600 1 ТБ SSD	№2: Ryzen 5 5600x, 4.4ГГц RTX 3080Ti 12гб ОЗУ 32Gb DDR4, 3600 1ТБ SSD
Характеристики комп’ютера		
Оптимізована модель	574,2 FPS (1,7 ms) – 626,1 FPS (1.6 ms)	533,4 FPS (1.9) – 600,7 FPS (1,7 ms)
Високополігональна модель	137,5 FPS (7,3 ms) – 137,7 FPS (7.3 ms)	481,2 FPS (2,1 ms) – 486,8 FPS (2.1 ms)

Рисунок 3.19 – Еталоний тест продуктивності оптимізованої моделі з вимкненою сценою на комп’ютері №1

Рисунок 3.20– Еталоний тест продуктивності оптимізованої моделі з вимкненою сценою на комп’ютері №1

Рисунок 3.21 – Еталоний тест продуктивності високополігональної моделі з вимкненою сценою на комп’ютері №1

Рисунок 3.22 – Еталоний тест продуктивності високополігональної моделі з вимкненою сценою на комп’ютері №1

Рисунок 3.23 – Еталоний тест продуктивності оптимізованої моделі з вимкненою сценою на комп’ютері №2

Рисунок 3.24 – Еталоний тест продуктивності оптимізованої моделі з вимкненою сценою на комп’ютері №2

Рисунок 3.25 – Еталоний тест продуктивності високополігональної моделі з вимкненою сценою на комп’ютері №2

Рисунок 3.26 – Еталоний тест продуктивності високополігональної моделі з вимкненою сценою на комп’ютері №2

Висновки по розділу: отже, можна побачити, що найбільше кадрів можна отримати, якщо оптимізувати модель та якщо вона буде лише одна в сцені. Велику роль відіграє обсяг відеопам’яті на комп’ютерах, де буде відтворюватись гра. Чим більше відеопам’яті – тим більшу кількість кадрів та меншу затримку буде отримувати користувач. Також можна побачити, що краще оптимізовувати 3D-моделі, бо якщо їх кількість велика, то разом вони не так сильно навантажують систему, як неоптимізовані моделі.

ВИСНОВКИ

Темою атестаційної роботи було оптимізація 3D-моделі для ігрового рушія. Unity 3D.

У роботі проаналізовано основні ігрові рушії, найбільш популярні безкоштовні ігрові рушії для створення ігор. Для виконання роботи обране програмне забезпечення Blender 3d та Unity 3D, оскільки вони є безкоштовними та добре підходять для всіх встановлених задач, які потрібно було виконати.

Для практичного завдання, було обрано модель мортири. У ході роботи використані різні методи моделювання, такі як, high-poly моделювання, low-poly моделювання, створення UV-розгортки, текстурування моделі в програмному пакеті Substance 3D painter. Під час роботи було створено сцену в ігровому рушії Unity 3D, куди було завантажено оптимізовану модель. Фінальним етапом усієї роботи був проведений тест на оптимізацю моделі, було порівняно на комп’ютерах з різними технічними характеристиками оптимізовану і неоптимізовану моделі, що вдало реалізує концепцію даної атестаційної роботи.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. UNITY.COM [Електронний ресурс]. – Режим доступу до ресурсу: https://unity.com/ (дата звернення 20.11.2023)

2. ULAB [Електронний ресурс]. – Режим доступу до ресурсу: https://ulab.sumdu.edu.ua/uk/yak-optimizuvati-vashi-3d-dlya-ar-vr (дата звернення 20.11.2023)

3. WIKIPEDIA [Електронний ресурс]. – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%A2%D1%80%D1%96%D0%B0%D0%BD%D0%B3%D1%83%D0%BB%D1%8F%D1%86%D1%96%D1%8F_(%D0%BA%D0%BE%D0%BC%D0%BF%27%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B8%D0%B9_%D0%B7%D1%96%D1%80) (дата звернення 20.11.2023)

4. ULAB [Електронний ресурс]. – Режим доступу до ресурсу: https://ulab.sumdu.edu.ua/uk/10-najkrashhih-igrovih-rushiiv (дата звернення 20.11.2023)

5. Game-ace.com [Електронний ресурс]. - Режим доступу до ресурсу: https://game-ace.com/blog/high-poly-models/ (дата звернення 20.11.2023)

6. DOU [Електронний ресурс]. - Режим доступу до ресурсу: https://gamedev.dou.ua/blogs/how-and-why-to-create-a-game-engine/ (дата звернення 20.11.2023)

7. V. M. Kartashov, V. N. Oleynikov, S. A. Sheyko, I. V. Koryttsev, S. I. Babkin, O. V. Zubkov, "Peculiarities of small unmanned aerial vehicles detection and recognition," Telecommunications and Radio Engineering, 2019, V. 78, Iss. 9, pp. 771–781. DOI: 10.1615/TelecomRadEng.v78.i9.30.

8. V. N. Oleynikov, O. V. Zubkov, V. M. Kartashov, I. V. Korytsev, S. I. Babkin, S.A. Sheiko, "Investigation of detection and recognition efficiency of small unmanned aerial vehicles on their acoustic radiation," Telecommunications and Radio Engineering, 2019, V. 78, Iss. 9, pp. 759–770. DOI: 10.1615/TelecomRadEng.v78.i9.20.

9. Kartashov, V.M., Sidorov, G.I., Sheiko, S.A., Kolendovska, M.M., Sergienko O.Yu. Principles of Construction and Assessment of technical Characteristics of multi-Frequency atmospheric Sodar in the Humidity Measurement Mode / Telecommunications and Radio Engineering.- New York. - 2020.- Vol. 79, №4.- P.323-333. (стаття). DOI: 10.1615/TelecomRadEng.v79.i4.50.

10. Kartashov, V.M., Oleynikov V.N, Zubkov, O.V., Korytsev I.V., Babkin, S. I., Sheiko, S.A.,. Kolendovskaya, M.M. Spatial-temporal Processing of acoustic Signals of Unmanned Aerial Vehicles; Telecommunications and Radio Engineering, 2020. Vol. 79, Iss, 9, pp.769-780.

11. V.M. Semenets, V.M. Kartashov, V.I. Leonidov. Features of Acoustic Noise of Small Unmanned Aerial Vehicles // Telecommunications and Radio Engineering.- New York. - 2020.- Vol. 79, №11.- P. 985-995. DOI: 10.1615/TelecomRadEng.v79.i11.80 (стаття).

12. Oleynikov V.N., Kartashov, V.M., Babkin, S. I., Zubkov, O.V., Korytsev I.V., Sheiko, S.A.,. Seleznov I.S. Structure and Parameter Unmanned Aerial Vehicles Sound Fields/ Telecommunications and Radio Engineering.- New York. - 2020.- Vol. 79, №17.- P.1539-1550. DOI: 10.1615/TelecomRadEng.v79.i17.50 (стаття).

13. В.А. Тихонов, В.М. Карташов, В.М. Олейніков, В.И. Леонидов, Л.П. Тимошенко, И.С. Селезнеов, Н.В. Рыбников. Обнаружение-распознавание беспилотных летательных аппаратов с использованием составной модели авторегрессии их акустического излучения// Вісник НТУУ «КПІ». Радіотехніка. Радіоапаратобудування. - 2020.- Вип. №81. – С.38-46. (Web of Science)

14. Карташов В.М., Корытцев И.В., Олейников В.Н., Зубков О.В., Шейко С.А., Бабкин С.И., Левский Н.А., Селезнев И.С. Алгоритмы пеленгации беспилотных летательных аппаратов по их акустическому излучению// Радиотехника. (Харьков). — 2019. — Вып. 196. — С. 22-31.

15. Карташов В.М., Харченко О.И., Чумаков В.И. Использование эффекта стохастического резонанса для анализа спектров акустического излучения малых беспилотных летательных аппаратов // Радиотехника. (Харьков). — 2019. — Вып. 197. — С. 100-106.

16. Карташов В.М., Сидоров Г.І., Толстих Є.Г., Шаповалов С.В. Акустичний вимірювач швидкості вітру в атмосферному прикордонному шарі// Радиотехника. (Харьков). — 2019. — Вып. 199. – С. 54