Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра комп’ютерних наук і математики

Допущено до захисту
Завідувач кафедри комп’ютерних наук
доктор технічних наук, професор
Андрій БОНДАРЧУК
«01 » червня 2026 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр
спеціальність 122 Комп’ютерні науки
освітня програма 122.00.01 Інформатика

Тема: МОДЕЛЮВАННЯ ТА ОПТИМІЗАЦІЯ ТРИВИМІРНИХ ОБ’ЄКТІВ ДЛЯ ІНТЕРАКТИВНИХ ПЛАТФОРМ

Виконав
Студент групи ІНб-1-22-4.0д
Нагорний Олексій Сергійович

_____
(підпис)

Науковий керівник
к.п.н., доцент,
Бойко Марія Анатоліївна

_____
(підпис)

Київ – 2026

Київський столичний університет імені Бориса Грінченка
Факультет інформаційних технологій та математики
Кафедра комп’ютерних наук і математики

«Затверджую»
Завідувач кафедри комп’ютерних наук
доктор технічних наук, професор
Андрій БОНДАРЧУК
« 31 » жовтня 2025 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА
«Моделювання та оптимізація тривимірних об’єктів для інтерактивних платформ»

Виконавець – студент групи ІНб-1-22-4.0д,
спеціальності 122 Комп’ютерні наук,
освітньої програми 122.00.01 Інформатика
Нагорний Олексій Сергійович

1. Вихідні дані: Наукові публікації та навчально-методичні матеріали з питань тривимірного моделювання, оптимізації 3D-об’єктів та їх використання в інтерактивних платформах. Матеріали щодо застосування програмних засобів Blender та ZBrush для створення тривимірних моделей.
2. Основні завдання: Здійснити аналіз сучасних підходів до моделювання та оптимізації тривимірних об’єктів для інтерактивних платформ. Створити тривимірні моделі в різних стилістиках. Виконати рендеринг моделей без оптимізації та після застосування оптимізації. Провести порівняльний аналіз отриманих результатів. Здійснити інтеграцію створених моделей в інтерактивну платформу та оцінити вплив оптимізації на продуктивність. Сформулювати висновки за результатами дослідження та оформити кваліфікаційну роботу відповідно до вимог кафедри.
3. Пояснювальна записка: Обсяг близько 70 сторінок формату А4 комп’ютерного набору з дотриманням вимог стандартів і методичних рекомендацій кафедри.
4. Графічні матеріали: Рендери тривимірних моделей до та після оптимізації, порівняльні ілюстрації результатів.
5. Додатки: Рендер зображення моделей, додаткові ілюстративні матеріали, скріншоти з інтерактивної платформи.
6. Строк подання роботи на кафедру: «01» червня 2026 р.

Науковий керівник
к.п.н., доцент
_____ Бойко М.А.

Виконавець:
_____ Нагорний О.С.

КАЛЕНДАРНИЙ ПЛАН

№	Етапи виконання роботи	Термін виконання	Примітка
1	Затвердження теми кваліфікаційної роботи бакалавра	31.10.25	Виконано
2	Аналіз проблеми, вивчення аналогів, формування цілей і завдань дослідження	16.10.25 – 20.11.25	Виконано
3	Аналіз методів моделювання та оптимізації тривимірних об'єктів для інтерактивних платформ	21.11.25 – 10.12.25	Виконано
4	Дослідження програмних засобів створення та оптимізації тривимірних моделей	11.12.25 – 20.01.26	Виконано
5	Розробка тривимірних моделей та виконання ретопології для подальшої оптимізації	21.01.26 – 10.02.26	Виконано
6	Оптимізація тривимірних моделей та підготовка моделей до використання в ігровому рушії	11.02.26 – 28.02.26	Виконано
7	Впровадження моделей у середовище Unity та дослідження показників продуктивності	01.03.26 – 15.04.26	Виконано
8	Оформлення пояснювальної записки, підготовка графічних матеріалів та додатків	01.05.26 – 15.05.26	Виконано
9	Захист кваліфікаційної роботи	16.06.26	

«Затверджую»
Науковий керівник
к.п.н., доцент, Бойко М.А.

Індивідуальний план склав
Нагорний О.С.

РЕФЕРАТ

Кваліфікаційна робота: 55 с., 29 рис., 15 табл., 39 посилань.

Актуальність. Технічна документація провідних ігрових рушіїв Unity та Unreal Engine фіксує рекомендовані методи оптимізації 3D-активів, однак не надає кількісних даних щодо їхнього реального впливу на продуктивність рушія за конкретних умов сцени. Наукові публікації здебільшого розглядають окремі аспекти оптимізації ізольовано, без наскрізного порівняння повного виробничого циклу на різнотипних об'єктах. Це зумовлює потребу в експериментальному дослідженні, яке б кількісно оцінило вплив послідовного застосування методів оптимізації на показники продуктивності рушія реального часу.

Об'єкт дослідження: процес підготовки тривимірних моделей до використання в інтерактивних програмних середовищах.

Предмет дослідження: методи моделювання та оптимізації 3D-об’єктів і їхній вплив на продуктивність рендерингу в Unity 6 URP.

Мета роботи: експериментально дослідити вплив оптимізації 3D-об’єктів на продуктивність рендерингу в інтерактивному середовищі Unity 6 URP.

Завдання:

1. Проаналізувати підходи до моделювання та оптимізації 3D-об’єктів.
2. Визначити технічні вимоги інтерактивних платформ до 3D-активів.
3. Створити тестові 3D-моделі у Blender та ZBrush.
4. Застосувати методи оптимізації до створених моделей.
5. Порівняти показники продуктивності моделей до та після оптимізації.

Основні результати. Тестування виконувалось на конфігурації Intel Core i5-12500 / NVIDIA GeForce RTX 3050 Laptop GPU (4 ГБ VRAM) / 16 ГБ RAM у двох ідентичних сценах Unity 6 URP, кожна з яких містила всі чотири моделі одночасно. Полігонаж скоротився на 91–97%. У зведеній сцені FPS зріс із 55,5 до 252,0. GPU Frametime знизився з 14,9 до 2,5 мс (скорочення на 83%),

розрахунковий обсяг VRAM зменшився на 914,7 МБ (скорочення на 64%). Встановлено, що величина приросту продуктивності визначається домінуючим типом навантаження: для об'єктів із великою кількістю підоб'єктів вирішальним є скорочення Draw Calls; для органічних моделей ретопологія із запіканням карт нормалей.

Практичне значення: відпрацьований pipeline може бути безпосередньо застосований при розробці контенту для рушіїв Unity та Unreal Engine, VR/AR-застосунків і навчальних симуляторів із широким спектром цільових платформ.

Ключові слова: тривимірне моделювання, оптимізація 3D-об'єктів, Unity URP, полігональне моделювання, цифровий скульптинг, запікання текстур, ретопологія, продуктивність рендерингу.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AR (augmented reality) – доповнена реальність; технологія суміщення цифрових об'єктів із зображенням реального середовища в реальному часі.

CPU (central processing unit) – центральний процесор; основний обчислювальний компонент комп'ютерної системи.

FPS (frames per second) — кадрів за секунду; показник частоти відтворення кадрів, що характеризує продуктивність рендерингу в реальному часі.

GPU (graphics processing unit) – графічний процесор; спеціалізований обчислювальний пристрій для паралельної обробки графічних даних.

LOD (level of detail) – рівень деталізації; система автоматичного перемикання між версіями моделі з різним полігонажем залежно від відстані до камери.

PBR (physically based rendering) – фізично коректний рендеринг; модель візуалізації матеріалів, що відтворює реальні фізичні властивості поверхонь на основі законів відбиття та поглинання світла.

URP (Universal Render Pipeline) – універсальний конвеєр рендерингу Unity; масштабований рендер-пайплайн із підтримкою PBR, орієнтований на широкий діапазон цільових платформ.

UV – двовимірна система координат розгортки тривимірної поверхні, що використовується для коректного накладання текстурних карт на геометрію об'єкта.

ЗМІСТ

ВСТУП

Тривимірна графіка є невід'ємним компонентом сучасних інтерактивних програмних систем: ігрових рушіїв, тренажерів, навігаційних додатків з підтримкою доповненої та віртуальної реальності. Попит на реалістичну візуалізацію в реальному часі зростає разом із розширенням цільових платформ: від потужних настільних систем до мобільних пристроїв із суттєво обмеженими обчислювальними ресурсами.

Високий рівень деталізації 3D-сцен потребує значних обчислювальних ресурсів. Обмеженість GPU, CPU та відеопам'яті є одним із ключових чинників, що визначають практичну реалізовуваність складних сцен у рушіях реального часу. Особливо критичним це стає у VR/AR-застосунках, де падіння частоти кадрів нижче порогового значення спричиняє фізичний дискомфорт користувача внаслідок розузгодження між рухами голови та оновленням зображення. Відповідно, кадровий бюджет у таких застосунках є жорсткою технічною вимогою, а не рекомендацією.

Сучасні засоби тривимірного моделювання, зокрема Blender і ZBrush, дозволяють створювати об'єкти з мільйонами полігонів і надзвичайно високим рівнем деталізації. Без попередньої оптимізації модель суттєво знижує продуктивність сцени в ігровому рушії. Це породжує практичну задачу підготовки 3D-активів: необхідно зберегти прийнятний рівень візуальної якості при суттєвому скороченні полігонального та текстурного навантаження.

Проблема оптимізації 3D-активів для рушіїв реального часу досліджується у низці наукових та технічних робіт. Розроблено методи ретопології, LOD-систем (Level of Detail), запікання карт нормалей та оклюзії, а також алгоритми автоматичного спрощення геометрії. Водночас комплексного кількісного аналізу впливу повного циклу підготовки активів на конкретні метрики продуктивності (FPS, GPU Frametime, споживання

відеопам'яті) у відкритих публікаціях представлено недостатньо. Це зумовлює актуальність і практичну значущість даного дослідження.

Мета роботи – експериментально дослідити вплив оптимізації 3D-об’єктів на продуктивність рендерингу в інтерактивному середовищі Unity 6 URP.

Завдання:

1. Проаналізувати підходи до моделювання та оптимізації 3D-об’єктів.
2. Визначити технічні вимоги інтерактивних платформ до 3D-активів.
3. Створити тестові 3D-моделі у Blender та ZBrush.
4. Застосувати методи оптимізації до створених моделей.
5. Порівняти показники продуктивності моделей до та після оптимізації.

Об’єкт дослідження: процес підготовки тривимірних моделей до використання в інтерактивних програмних середовищах.

Предмет дослідження - методи моделювання та оптимізації 3D-об’єктів і їхній вплив на продуктивність рендерингу в Unity 6 URP.

Методи дослідження. У теоретичній частині застосовано методи аналізу наукових публікацій та технічної документації з тривимірного моделювання, оптимізації 3D-активів і ігрових рушіїв. Практична частина реалізована шляхом порівняльного експерименту: чотири тривимірні моделі різних типів створено та оптимізовано в середовищах Blender 4.3 і ZBrush 2022, після чого протестовано в рушії Unity 6 URP за однакових умов сцени у двох станах, до та після оптимізації.

Практична значущість дослідження визначається апробацією методів створення та оптимізації 3D-моделей для інтерактивних середовищ. Роботу виконано на конкретних прикладах із застосуванням Blender, ZBrush та Unity, що охопило повний технологічний ланцюжок: від полігонального моделювання та ретопології до UV-розкладки, спрощення геометрії й замірів продуктивності.

Експериментальна частина підтвердила можливість суттєвого скорочення полігонального навантаження та кількості draw calls при

збереженні прийнятного рівня візуальної деталізації. Вимірювання FPS до та після оптимізації зафіксували реальний приріст швидкодії сцени й зниження споживання апаратних ресурсів.

Одержані результати мають пряме прикладне значення: для мобільних ігор із жорстким бюджетом кадру, для VR-додатків, де падіння частоти кадрів спричиняє дискомфорт користувача, а також для навчальних симуляторів із широким спектром цільових платформ. Описана методика може слугувати відтворюваним робочим шаблоном, а для студентів суміжних напрямів наочною демонстрацією наскрізного процесу роботи з 3D-активом.

РОЗДІЛ 1

АНАЛІТИЧНИЙ ОГЛЯД МЕТОДІВ МОДЕЛЮВАННЯ ТА ОПТИМІЗАЦІЇ

1.1 Роль 3D-об’єктів в інтерактивних платформах та ігрових рушіях

Полігональна сітка складається з вершин, ребер і граней. Вершина це не просто точка у просторі: до неї прив'язані вектор нормалі та UV-координати, без яких текстура не лягає на поверхню коректно. Рушій звертається до цих даних при побудові кожного кадру. Перевернута нормаль або зламані UV дають цілком конкретний результат: пляму від освітлення не там, де треба, або текстуру, що розтягується у непередбачуваний бік [1].

Студійний рендерер і ігровий рушій працюють в абсолютно різних умовах. Офлайн-рендерер не має часових обмежень на обробку одного кадру. Рендер в реальному часі має жорстке обмеження: при 60 кадрах на секунду на весь кадр відведено рівно 16 мілісекунд. Складна модель забирає частину цього бюджету, яка більше нікуди не піде.

Blender Cycles і Unity не взаємозамінні середовища. Модель, що чудово виглядає при офлайн-рендері, в рушії може може суттєво знизити частоту кадрів у рушії. Оптимізація тут не означає зіпсувати модель, вона означає перемістити деталі із сітки в текстурні карти так, щоб на екрані різниця була мінімальною [2].

Геометрія впливає і на фізичні розрахунки. Для зіткнень будується окремий спрощений колайдер, повна сітка для цього надто дорога. Форма оригінальної моделі задає орієнтир для його побудови. Погана топологія дає грубий колайдер або взагалі некоректний.

Структуру полігональної сітки тривимірного об'єкта подано на рисунку 1.1.

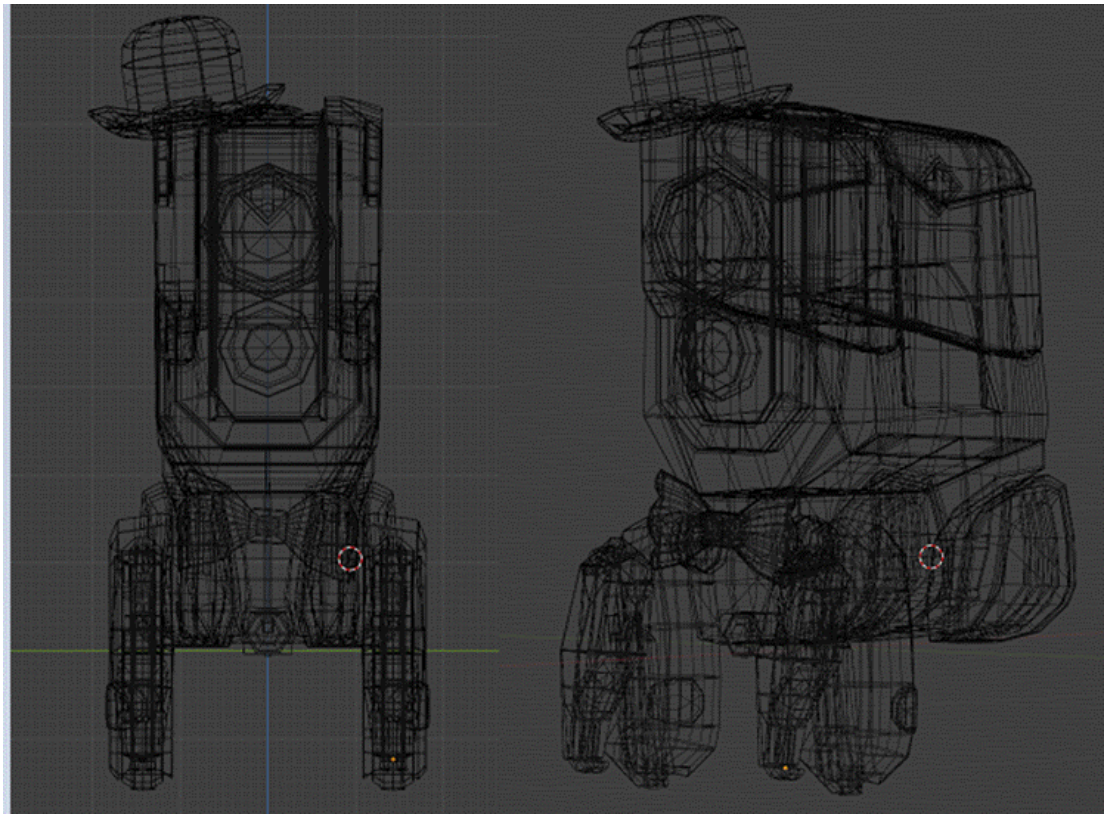

Рисунок 1.1. Структура полігональної сітки тривимірного об'єкта

У рушії тривимірний об'єкт є одночасно картинкою на екрані, статтею витрат у GPU-бюджеті та фізичним тілом у симуляції. Урахування цих функцій на початковому етапі моделювання дозволяє забезпечити передбачуваність результату в рушії.

1.2 Методи створення 3D-моделей

Для моделювання механічних об'єктів і твердих поверхонь застосовується полігональний метод, тоді як персонажі й органічні форми переважно створюються методом цифрового скульптингу.

Цифровий скульптинг побудований на іншій логіці. Тут не маніпулюють окремими полігонами, тут ліплять форму кистями, подібно до роботи з глиною. ZBrush у режимі Dynamesh сам перерозподіляє геометрію при потребі, художник формує об'єм моделі шляхом нарощування або зменшення геометричної маси у режимі реального часу. Для органічних форм, де кожна деталь поверхні є унікальною, де важлива природність рельєфу шкіри або

складок тканини, скульптинг дає результат, недосяжний полігональним методом.

Метод цифрового скульптингу має суттєве обмеження: модель після скульпт-сесії не придатна для використання у рушії в тому вигляді, в якому вона виходить. Мільйони хаотично розподілених полігонів не мають топологічного порядку. Тому за скульптингом обов'язково йде ретопологія, побудова нової чистої сітки поверх готової форми [3].

Процедурне моделювання відрізняється від обох попередніх методів за своєю суттю. Художник не будує форму руками, а задає правила і параметри, за якими програма генерує геометрію автоматично. Змінив параметр і отримав інший варіант. Це незамінно для масового контенту: тисячі будівель відкритого світу, різноманітна рослинність, скельні породи. Houdini є основним інструментом для цього підходу у великих студіях.

Фотограмметрія стала робочим інструментом не так давно, коли GPU навчились обробляти великі хмари точок за прийнятний час. Суть методу: об'єкт знімають із десятків позицій, потім програма зіставляє спільні точки на фото і відновлює тривимірну форму. Рівень деталізації поверхні, отриманий цим методом, є практично недосяжним при ручному моделюванні, оскільки результатом є цифрова копія реального об'єкта [4]. Старі цеглини, поламаний асфальт, вивітрений камінь, подібні речі зараз сканують, а не моделюють.

На практиці методи майже ніколи не використовуються окремо. Пайплайн більшості проєктів поєднує декілька підходів у межах одного завдання. Тіло персонажа ліпиться у ZBrush, одяг моделюється полігональним методом у Blender, бо тканини зі швами краще будуються через петлеві вирізання. Зброя і спорядження йдуть через полігональне моделювання через їх механічну природу. Ефективне поєднання методів є обов'язковою компетентністю у виробничому пайплайні сучасних проєктів.

Порівняльну характеристику методів моделювання за ключовими критеріями подано у таблиці 1.1.

Таблиця 1.1

Порівняльна характеристика методів моделювання

Метод	**Тип об'єктів**	**Переваги**	**Обмеження**	**Придатність для рушія**	**Потреба в оптиміза ції**
Полігональне моделювання	Предмети, механіка	Точна форма, контроль сітки	Трудомістки й для органіки	Висока, за умови коректної топології	Помірна
Цифровий скульптинг	Персонажі, органіка	Висока деталізація	Потребує ретопології	Потрібна ретопологія	Висока
Процедурне моделювання	Середовищ а	Швидке створення варіацій	Потребує знання математични х основ	Складність налаштування	Помірна
Фотограммет рія	Реальні об'єкти	Реалістичність	Залежить від якості зйомки	Потрібне очищення моделі	Висока

Результати застосування різних методів моделювання на прикладі одного об'єкта наведено на рисунку 1.2.

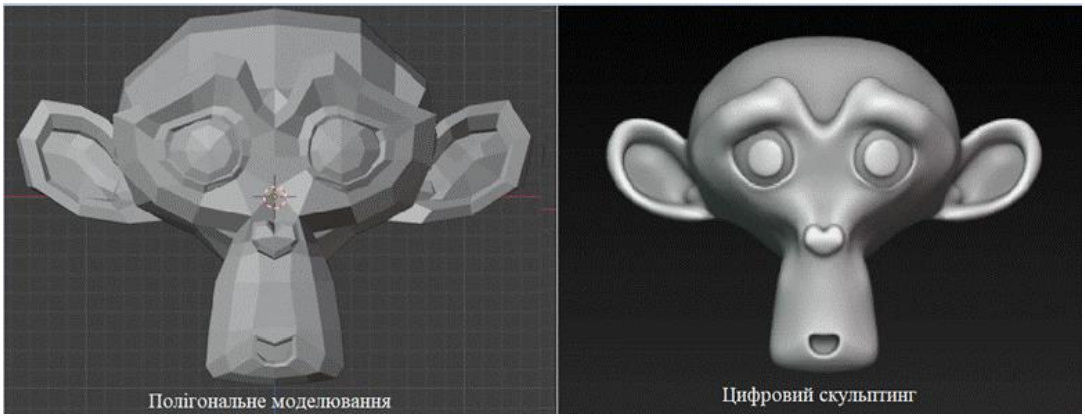

Рисунок 1.2. Результати різних підходів до моделювання

1.3 Програмні засоби для 3D-моделювання та їх оцінка

Ринок програм для тривимірного моделювання неоднорідний: кожен продукт прийшов до своєї аудиторії через різні завдання і різну історію. Maya

і Blender здаються конкурентами лише на перший погляд, насправді вони існують у різних контекстах. Знати різницю між ними потрібно не для вибору найкращого, а щоб не брати не той інструмент для конкретної роботи [5].

Blender поширюється безкоштовно під відкритою ліцензією, це одразу визначає його аудиторію. До 2019 року програма мала репутацію незручного інструменту. Після релізу 2.8 все змінилось: студії почали брати Blender у робочі пайплайни, а для незалежних авторів і навчальних закладів він став стандартом [6]. Версія 4.3 продовжує цей курс.

З практичної точки зору Blender 4.3 добре вирішує задачі полігонального моделювання завдяки системі модифікаторів. Mirror Modifier відображає половину об'єкта симетрично, що скорочує час роботи з симетричними формами вдвічі. Subdivision Surface Modifier збільшує щільність сітки за алгоритмом Catmull-Clark, даючи плавні поверхні без ручного додавання геометрії [7]. Стек модифікаторів є неруйнівним: вихідна геометрія не змінюється, і на будь-якому етапі можна повернутись і скорегувати параметри.

Edit Mode дає прямий доступ до вершин, ребер і граней. Loop Cut, Inset Faces, Merge by Distance, Bridge Edge Loops, це стандартний набір операцій, без якого не обходиться жодна модель. Face Orientation Overlay візуально показує напрямок нормалей прямо у вьюпорті.

ZBrush від компанії Maxon є окремою категорією серед 3D-програм. У його основі лежить технологія Pixol: кожна точка зберігає не тільки координати XYZ, а й дані про матеріал, колір і глибину. Саме через це програма тягне моделі на десятки мільйонів полігонів без відчутного гальмування [8]. ZBrush 2022 включає вдосконалений ZRemesher для автоматичної ретопології та оновлений набір кистей.

Dynamesh є однією з ключових особливостей ZBrush. Режим автоматично перераховує геометрію при злитті, що дозволяє вільно додавати і видаляти масу без деформації сітки. Кисті Dam Standard для різьблення тонких деталей, Clay Buildup для нарощування об'єму, Standard для загальної форми і Smooth для вирівнювання поверхонь утворюють базовий робочий

набір [9]. Разом вони дозволяють будувати органічні форми, недосяжні полігональним методом за розумний час.

При роботі з ZBrush 2022 на практиці виявляються проблеми. Інтерфейс програми є нестандартним і помітно відрізняється від будь-якого іншого 3D-додатка: функції знаходяться там, де їх не очікуєш, навігація у вьюпорті підпорядковується іншій логіці. Адаптація займає час після досвіду в інших програмах. Для запобігання втраті даних при тривалих сесіях рекомендується регулярне збереження проміжних результатів. Збереження кожні 10-15 хвилин є необхідністю.

Autodesk Maya є стандартом великих анімаційних студій і провідних ігрових виробників. Її перевага виявляється найбільш виразно в анімаційному пайплайні: інструменти для побудови рига і прив'язки шкіри у Маya є найбільш продуманим серед усього доступного [10]. Autodesk 3ds Max традиційно присутній в архітектурній візуалізації і серед розробників активів для Unreal Engine завдяки широкій екосистемі плагінів.

Houdini від SideFX займає нішу процедурного підходу і VFX. Кожна операція у Houdini є вузлом у мережі, що дає надзвичайну відтворюваність і гнучкість результатів. Але ця ж особливість вимагає глибшого розуміння математичних основ порівняно з іншими програмами. Великі студії використовують Houdini переважно для ефектів руйнувань, симуляцій рідин і процедурного генерування середовищ [11].

Adobe Substance 3D Painter у цьому проєкті відповідає за PBR-текстурування. Малювати можна прямо по тривимірній моделі з підтримкою фізично коректних матеріалів, що значно зручніше, ніж редагувати пласкі текстурні карти. При виборі пресету експорту для Unity URP програма сама формує потрібний набір файлів, і їх можна одразу підключати до шейдера Lit без жодних додаткових дій [12]. Smart Materials додатково прискорюють роботу: знос, подряпини і бруд генеруються автоматично на основі кривизни і оклюзії моделі.

Порівняльну характеристику програмних засобів за основними критеріями подано у таблиці 1.2.

Таблиця 1.2

Порівняльна характеристика програмних засобів за основними критеріями

Програма	**Функціональність**	**Підтримка оптимізації**	**Сфера застосування**	**Ціна**
Blender	Моделювання, UV, рендер	Є вбудовані інструменти	Інді, освіта	Безкоштовно
ZBrush	Скульптинг, деталізація	Retopo та decimation	Персонажі, органіка	Комерційна
Autodesk Maya	Моделювання, риг, анімація	Обмежена	Анімаційні студії, AAA	Комерційна
Houdini	Процедурне моделювання, VFX	Процедурна LOD-генерація	VFX, великі студії	Комерційна
Substance 3D Painter	PBR-текстурування	Експорт під URP	GameDev, текстурування	Комерційна

Інтерфейс Blender 4.3 та ZBrush 2022 у процесі роботи подано на рисунку 1.3.

Рисунок 1.3. Інтерфейси Blender 4.3 та ZBrush 2022 під час роботи

1.4 Поняття та необхідність оптимізації тривимірних об'єктів

Надлишкова геометрія і погано організовані текстури це не естетична проблема, а технічна. У рушії реального часу вони безпосередньо знижують частоту кадрів. Офлайн-рендерер не має часових обмежень, він рахує кадр стільки, скільки треба. На рушій при 60 fps на весь кадр відведено 16,6

мілісекунди [13]. Вийти за цей ліміт через важку модель означає одразу отримати просадку fps. Оптимізація тому є не опціональним кроком, а умовою придатності моделі до роботи в інтерактивному середовищі.

За 16,6 мілісекунди рушій встигає визначити, які об'єкти у полі зору, передати їх на GPU, порахувати освітлення, накласти матеріали і виконати пост-обробку. Якщо одна тільки геометрія з'їдає 10 мілісекунд, то решті сцени не лишається нічого.

Сучасний GPU теоретично тягне сотні мільйонів трикутників за секунду. Але є інше обмеження: кожен окремий матеріал на кожному об'єкті це окремий draw call, тобто окрема команда CPU до GPU. Вона коштує часу незалежно від того, наскільки проста геометрія [14].

N-гони наочно демонструють, як технічна неохайність стає видимим дефектом. Полігон із п'ятьма або більше вершинами перед рендерингом тріангулюється рушієм. Алгоритм може розбити його по-різному. Наслідок, темні зірки або неправильні тіні прямо на поверхні, які не прибирає ні зміна матеріалу, ні корекція освітлення [15]. Єдине рішення, розбити N-гони вручну у програмі моделювання ще до імпорту.

Нормалі є проблемою, яка часто дивує при першому переносі скульпта або після булевих операцій. Нормаль це вектор, перпендикулярний до поверхні і спрямований назовні. На ньому тримається весь розрахунок освітлення. Якщо вона перевернута, рушій освітлює полігон так, ніби дивиться на нього зсередини: він темний, або зовсім не видно при стандартному однобічному рендерингу. Після тривалого редагування або об'єднання об'єктів інвертовані нормалі практично гарантовані в якійсь кількості.

Обмеження відеопам'яті GPU є не менш критичним чинником продуктивності, ніж частота кадрів, хоча й рідше розглядається окремо. Одна текстура розміром 4096x4096 без стиснення займає близько 64 МБ на канал. Повний PBR-набір, albedo, metallic, roughness, normal, occlusion, це вже п'ять таких файлів для одного об'єкта. При десятках об'єктів у сцені загальний обсяг легко перевищує ліміт відеопам'яті [16]. На відміну від системної оперативної

пам'яті, перевищення обсягу VRAM призводить до критичного зниження продуктивності рендерингу.

Головна ідея оптимізації, яку варто прийняти від початку: мета не спростити модель, а перерозподілити деталі між геометричним і текстурним рівнями. Дрібний рельєф, подряпини, болти, шви, це ефективніше зберігати у карті нормалей, ніж у геометрії. При правильному підході оптимізована модель у рушії виглядає майже так само, як оригінальний скульпт, але важить менше.

1.5 Методи оптимізації 3D-моделей для інтерактивних платформ

Оптимізація являє собою структуровану послідовність операцій, порядок виконання яких є принципово важливим: порушення послідовності або пропуск окремих кроків може призвести до некоректного відображення моделі в рушії. Застосування Shade Smooth до моделі з N-гонами не усуває наявні артефакти, а навпаки, посилює їхню помітність. Тому порядок виконання кроків має значення.

Ретопологія застосовується тоді, коли вихідна топологія принципово не придатна для рушія. Це завжди стосується скульптингу: мільйони нерівномірних полігонів без чіткої структури. Ретопологія вручну означає побудову нової чотирикутникової сітки поверх готової форми. Для персонажів з анімацією петлі нової сітки проводяться так, щоб деформація суглоба виглядала природно: кільця навколо ліктів і колін, петлі навколо очей і рота. ZRemesher у ZBrush робить це автоматично, на складних зонах обличчя і суглобів дає прийнятний результат, тому критичні ділянки дороблюються вручну [17].

Видалення зайвих петель застосовується тоді, коли загальна топологія прийнятна, але є надлишкова геометрія. У процесі роботи в Blender петлеві вирізання і модифікатори нерідко додають більше ребер, ніж потрібно. Операція Dissolve Edges прибирає вибрані ребра без зміни форми об'єкта. На

складних моделях це може давати зниження полігонажу на 20-30% без жодних видимих змін [18].

Усунення N-гонів є обов'язковим кроком, без якого модель не можна вважати готовою. Blender підсвічує їх у режимі Overlay і дозволяє виправити вручну або через команду Triangulate Faces [19]. Орієнтир: сітка з чотирикутників у зонах деформації і трикутників у кутових та перехідних місцях, де анімації немає. Трикутники у зонах деформації є допустимими, але не бажаними, бо при вигині можуть давати небажані складки.

Виправлення нормалей не змінює форму, але усуває візуальні артефакти. Команда Recalculate Outside через Shift+N автоматично направляє нормалі назовні. Режим Face Orientation у Blender показує стан: сині грані мають правильний напрямок, червоні перевернуті. На моделях після тривалого редагування і операцій злиття кілька інвертованих нормалей залишаються після автоматичного виправлення і потребують ручного втручання [20].

Інструмент Shade Smooth не змінює геометрію об’єкта (кількість полігонів), проте суттєво трансформує візуальне сприйняття поверхні. За відсутності згладжування кожна грань відображається відповідно до її власної нормалі, що робить межі між гранями видимими. Застосування Shade Smooth активує алгоритм інтерполяції нормалей між вершинами сусідніх граней, завдяки чому поверхня візуально виглядає плавною. Функція Auto Smooth дозволяє встановити граничне значення кута, нижче якого відбувається згладжування, тоді як гострі ребра залишаються чіткими. Для моделювання механічних об’єктів, що поєднують циліндричні площини та гострі краї, такий підхід є оптимальним.

Запікання текстурних карт є методом переносу деталей із high-poly на low-poly через текстуру. Рушій пускає промені від поверхні low-poly до high-poly і записує знайдені дані у пікселі. Карта нормалей фіксує відхилення нормалей від геометричних, завдяки чому рушій при освітленні імітує рельєф, якого фізично немає. Карта Ambient Occlusion зберігає затінення у западинах і кутах. Разом вони дозволяють low-poly моделі виглядати порівняно з

оригінальним скульптом значно деталізованіше, ніж вказує реальна кількість полігонів [21].

UV-розгортка визначає якість текстурування напряму. Острівці UV не повинні перекриватись і мають займати простір якомога щільніше. Шви краще проводити по місцях, які рідко видно при стандартних ракурсах: під руками, по задній поверхні, у прихованих складках. Нерівномірна щільність UV дає помітну різницю деталізації між різними частинами одного об'єкта, і ніяка хороша текстура це не виправить.

Level of Detail, або LOD, є системним інструментом, вбудованим у рушій. Для одного об'єкта із зменшуваним полігонажем. Рушій перемикається між ними автоматично залежно від відстані до камери: на близькій відстані LOD0 із повною деталізацією, на великій LOD3 із мінімальною. Об'єкт за 50 метрів від камери однаково не буде читатися у деталях. LOD є стандартним елементом у будь-якому серйозному ігровому проєкті.

У межах даного дослідження система LOD не розглядається, оскільки метою є оцінка базових методів оптимізації без динамічного перемикання рівнів деталізації.

1.6 Вимоги інтерактивних платформ до тривимірних моделей

Перехід моделі із середовища моделювання в ігровий рушій це зміна технічного контексту. У Blender є одні конвенції, в Unity інші. Частина проблем, що виникають при імпорті, не пов'язана з якістю самої моделі: неправильний масштаб, перевернута орієнтація, некоректні нормалі після конвертації. Знати ці технічні деталі наперед означає заощадити час і уникнути неприємних сюрпризів після кількох годин роботи.

Серед підтримуваних форматів імпорту Unity саме FBX найкраще передає геометрію, UV, матеріали та анімації в одному файлі. Програми по-різному трактують вертикальну вісь, Z у Blender і Y у Unity. Експортер зазвичай конвертує це сам, але не завжди так, як очікуєш, і після імпорту

модель інколи стоїть боком. Тому після кожного імпорту перевірка орієнтації є першим кроком.

Масштаб є класичною пасткою при першому переносі між Blender і Unity. Об'єкт може виглядати коректним у вьюпорті Blender, але мати значення масштабу 0.01 або 100 у властивостях трансформації. В Unity це проявиться як велетенська або мікроскопічна модель. Вирішення просте: перед фінальним експортом треба застосувати масштаб через Ctrl+A, Apply Scale. Ця операція записує поточний масштаб у геометрію і скидає значення трансформації до одиниці [22].

Universal Render Pipeline, скорочено URP, є сучасним пайплайном рендерингу Unity і стандартним вибором для нових проєктів. Він замінив застарілий Built-in Pipeline і оптимізований для широкого діапазону платформ, від мобільних до ПК. URP використовує фізично коректну модель освітлення PBR, де матеріали описуються через реальні фізичні властивості поверхні.

PBR у Unity URP реалізований через шейдер Lit із стандартним набором текстурних карт. Albedo визначає базовий колір без освітлення. Metallic і Smoothness описують металічність і глянець. Normal Map додає дрібний рельєф через маніпуляцію нормалями. Occlusion підсилює глибину у западинах. Adobe Substance 3D Painter при виборі пресету для Unity URP формує цей набір автоматично, і його можна підключати до шейдера без додаткових перетворень.

Кількість матеріалів на об'єкті це кількість draw calls. Персонаж із вісьмома підмешами і вісьмома матеріалами дає вісім draw calls тільки на себе. Текстурний атлас вирішує це: усі поверхні збираються в одному файлі через UV-острівці, весь об'єкт обходиться одним матеріалом.

Розміри текстур мають бути степенями двійки: 256, 512, 1024, 2048 або 4096. Це не формальна вимога, так влаштовані алгоритми GPU-компресії і генерації мінімап. Unity стискає текстури під платформу при імпорті: BC7 або DXT5 для ПК, ASTC для мобільних. Нестандартний розмір не стискається оптимально і ламає частину алгоритмів.

Нормалі при імпорті Unity не виправляє. Увімкнення двобічного рендерингу через матеріал є обхідним рішенням, але змушує GPU рендерити кожен полігон двічі. Правильний підхід: виправити нормалі у Blender до експорту, а не компенсувати проблему вже в рушії.

Перед фінальним FBX-експортом з Blender варто пройтись по контрольному списку. Apply Transformations, щоб масштаб і поворот були записані у геометрію. Параметри Forward та Up для правильної орієнтації осей. Mesh Data для передачі UV-координат. Face Orientation Overlay покаже стан нормалей. Mesh Analysis підсвітить N-гони і проблемні місця геометрії. Статистика у правому куті вьюпорту дає підсумковий полігонаж (рис.1.4).

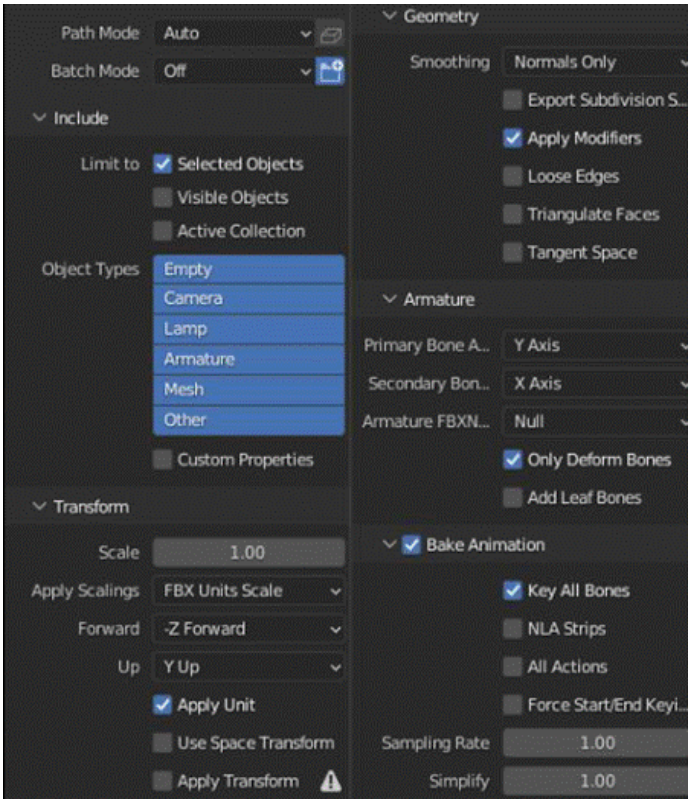

Рисунок 1.4. Параметри FBX-експорту Blender для коректного імпорту в Unity URP

Попри широкий арсенал інструментів оптимізації, у практиці розробки залишається відкритим питання балансу між якістю і продуктивністю. Наявні рекомендації щодо полігонажу, текстур і draw calls сформульовані узагальнено, вони не враховують специфіки конкретного типу об'єктів і умов

сцени. Практичне дослідження дозволить виміряти реальний вплив оптимізації у контрольованих умовах через порівняльне тестування оптимізованих і неоптимізованих моделей у Unity URP.

Висновки до розділу 1

У рушії реального часу тривимірний об'єкт виконує три функції одночасно. А саме: художню, обчислювальну і фізичну. Від якості побудови залежить продуктивність рендерингу, коректність освітлення і придатність активу до анімації та текстурування.

З розглянутих методів полігонального, скульптингу, процедурного і фотограмметрії для цілей дослідження обрано два: полігональне моделювання у Blender 4.3 для предметних об'єктів і скульптинг у ZBrush 2022 для персонажа. Вибір зумовлений відповідністю методів типам об'єктів і наявністю інструментів експорту під Unity URP.

Blender 4.3 закриває повний цикл роботи з предметними моделями. Від моделювання до UV і запікання. ZBrush 2022 незамінний для органічних форм завдяки Dynamesh і ZRemesher, але потребує ретопології перед експортом. Substance 3D Painter забезпечує PBR-текстурування з прямим виведенням у формат Unity URP.

Ретопологія, видалення надлишкових петель, усунення N-гонів, виправлення нормалей, запікання карт, оптимізація UV і система LOD утворюють послідовний виробничий процес. Дотримання цього порядку є умовою коректного відображення активу у рушії.

Практична частина будується на прямому вимірюванні: чотири моделі, два варіанти кожної, одне середовище Unity URP і конкретні цифри замість загальних порад.

РОЗДІЛ 2

ДОСЛІДЖЕННЯ ПРОЦЕСУ СТВОРЕННЯ ТА ОПТИМІЗАЦІЇ ТРИВИМІРНИХ ОБ'ЄКТІВ

2.1 Методика дослідження та технічні вимоги до тривимірних моделей для інтерактивних платформ

Як цільове середовище дослідження було вибрано Unity Universal Render Pipeline (URP). Вибір URP обґрунтований підтримкою фізично коректної моделі освітлення (PBR) та сумісністю з широким спектром цільових платформ – від мобільних пристроїв до настільних ПК, що відповідає вимогам більшості сучасних ігрових проєктів. Художнім референсом для стилістики моделей обрано гру Deep Rock Galactic, що характеризується чіткими силуетами, акцентованими формами та мінімальним поверхневим рельєфом підхід, що поєднує художні та технічні переваги. Менше деталей у геометрії означає менший тиск на рушій. Референси зібрано зі скріншотів гри та відкритих джерел і зведено в єдину дошку у PureRef (додаток А).

Весь цикл робіт виконувався на одному пристрої. Процесор - Intel Core i5-12500 (2,50 ГГц, 12-те покоління), відеокарта NVIDIA GeForce RTX 3050 Laptop GPU з 4 ГБ відеопам'яті, що становило практичне обмеження при роботі з текстурними наборами великого обсягу. Для моделювання використовувались Blender 4.3 і ZBrush 2022, для текстурування Adobe Substance 3D Painter 2025, а Unity 2022 LTS із підключеним URP слугував цільовим середовищем тестування.

До вибірки увійшли чотири об'єкти: органічний персонаж Дворф, механічний персонаж-робот Боско, кайло зі змішаною геометрією та простий твердотільний кухоль. Разом вони перекривають основні категорії ігрових активів, саме це і дозволяє побачити, як оптимізація веде себе на різній геометрії.

Перш ніж переходити до самого моделювання, варто пояснити, за якими показниками взагалі порівнювалися версії моделей. Без цього результати, наведені у підрозділах 2.4–2.5 та розділі 3, були б просто набором цифр без контексту.

Koulaxidis та Xinogalos у дослідженні оптимізації мобільних ігор у Unity виокремлюють три групи метрик: частоту кадрів як показник завантаженості конвеєра рендерингу, кількість draw calls як показник завантаженості центрального процесора та кількість полігонів і трикутників як показник завантаженості геометричного конвеєра GPU [23]. Саме ці три групи і лягли в основу системи вимірювань у даній роботі.

Кількість полігонів і трикутників найпряміший показник геометричного навантаження. Кожен трикутник у полі зору камери проходить через трансформацію вершин і растеризацію. Що більше трикутників, то більше часу займає цей етап у загальному бюджеті кадру. Вимірювання у другому розділі виконуються через статистику Blender у режимі EEVEE, у третьому через профайлер Unity.

Частота кадрів (FPS): інтегральна характеристика продуктивності. При цільових 60 кадрах за секунду рушій має встигати будувати кожен кадр за 16,67 мілісекунди, при 30 кадрах за 33,33 мілісекунди [24]. Якщо геометрія або draw calls з'їдають надто велику частку цього бюджету, частота кадрів падає. Саме тому FPS вимірюється на обох версіях кожної моделі під час тестування у рушії.

Кожен підоб'єкт з окремим матеріалом формує окремий draw call зі своїми даними: геометрією, шейдером, текстурами, станом матеріалу. Перетворення цих даних у апаратні команди коштує часу CPU, і надмірна кількість draw calls може стати вузьким місцем навіть при невеликому полігонажі [25]. Вимірювання виконується через вбудований профайлер Unity, розділ Rendering, мітка Draw Calls.

Час рендерингу одного кадру в Blender EEVEE використовується як проміжний показник ефективності геометрії. Умови ізольовані: нейтральний

фон, фіксований набір джерел освітлення, однаковий кут зйомки для кожної моделі. Такий підхід дозволяє зафіксувати внесок саме геометричної складності, не змішуючи його з ігровою логікою, фізикою чи пост-обробкою, що неминуче присутні у рушії.

Розмір текстур і використання відеопам'яті GPU (VRAM) потрапили до списку критеріїв через просте обмеження тестового пристрою: 4 ГБ VRAM. Одна карта формату 4096×4096 без стиснення займає близько 64 МБ на канал. Повний PBR-набір із п'яти карт може перевищити 300 МБ у нестиснутому вигляді тільки для одного об'єкта. До цього набору входять п'ять карт. Albedo фіксує базовий колір поверхні без тіней, без бліків, чистий колір матеріалу. Normal Map не змінює геометрію, але змушує рушій освітлювати поверхню так, ніби на ній є рельєф через маніпуляцію нормаллю кожного пікселя. Metallic вказує, які зони поводяться як метал і відповідно відбивають світло. Roughness регулює шорсткість від матової до майже дзеркальної поверхні. Occlusion підсилює затінення там, де світло природно не дістає: у западинах, швах, кутових зонах. Перевищення доступної VRAM змушує систему передавати дані між відеопам'яттю та оперативною, що різко знижує продуктивність [26].

До початку моделювання для кожного об'єкта були визначені кількісні межі: граничний полігонаж, склад текстурних карт, їх розміри та стратегія UV-розгортки. Ці межі задавали орієнтир на всіх наступних етапах і забезпечували порівнянність результатів.

Для кайла та кухля встановлено діапазон 500-3000 граней у фінальній версії. Для Дворфа та Боска 5000-15000 трикутників залежно від ролі об'єкта. Ці діапазони спираються на рекомендації документації Unity щодо оптимізації мобільних ігор, відповідно до яких прийнятна кількість полігонів на меш для мобільних платформ становить від 300 до 1500.

GPU теоретично здатний обробляти мільярди вершин за секунду. Але практична межа визначається не цією цифрою, а сукупним кадровим бюджетом: геометрія конкурує за мілісекунди з освітленням, тінями, фізикою

та пост-обробкою. Об'єкт другого плану, що витрачає стільки ж часу, скільки головний персонаж, це погано розподілений бюджет. Тому для простих предметів закладено до 3000 граней, а для персонажів до 15000 трикутників. Дослідження оптимізації ігрових активів методом квадратичних метрик похибки показує, що навіть скорочення полігонажу на 8–33% дає приріст FPS від 9% до 41% залежність нелінійна, і кожен зайвий полігон обходиться дорожче, ніж здається.

Кожен об'єкт отримав PBR-набір, сумісний із шейдером Lit у Unity URP: Albedo, Normal Map, Metallic/Roughness та Ambient Occlusion. Для об'єктів із поверхнями, що самосвітяться, додано Emission-карту.

Розміри: Дворф і Боско 2048x2048 пікселів, кайло й допоміжні частини 1024x1024, кухоль 2048x2048. Усі значення є степенями двійки. Ця вимога не формальна: саме такі розміри коректно стискаються форматами BC7 і DXT5 на ПК та ASTC на мобільних пристроях, які Unity застосовує при імпорті автоматично. Нестандартний розмір не стискається оптимально і порушує генерацію мінімап. Розмір 2048x2048 обрано для персонажів через необхідність достатньої щільності текселів при типових ігрових ракурсах; 1024×1024 є достатнім для невеликих предметів другого плану.

На оптимізованих версіях усіх моделей виконувалась ручна розгортка з прихованими швами. На неоптимізованих навмисно залишено Smart UV Project, щоб зафіксувати типові артефакти автоматичних алгоритмів. Нерівномірна щільність UV безпосередньо позначається на якості відображення текстурних карт і є вимірюваним показником підготовки моделі.

Сформована специфікація стала орієнтиром на всіх наступних етапах роботи і забезпечила контрольовані умови для порівняльного аналізу, результати якого наведено у підрозділах 2.4–2.5 та розділі 3.

2.2 Створення тривимірних моделей у програмах Blender та ZBrush

2.2.1 Створення моделей у Blender 4.3

Кухоль, кайло та робот Боско змодельовано методом полігонального моделювання у Blender 4.3. Кожен об'єкт складається з набору окремих мешів, а не єдиної суцільної геометрії. Це дозволяє редагувати кожну частину незалежно, спрощує UV-розгортку і дає змогу призначати різні матеріали окремим елементам.

Кухоль побудовано на основі циліндричного примітива. Вибір обумовлений геометричною природою об'єкта та можливістю задати кількість граней безпосередньо при створенні примітива. В режимі Edit Mode через екструзії сформовано стінки, дно та перехідні зони корпусу. Ручку виконано як окремий меш із подальшим замиканням до корпуса через Bridge Edge Loops. Декоративна емблема у вигляді рельєфного виступу сформована за допомогою петлевих вирізань із збереженням форми при перегляді через модифікатор Subdivision Surface. Кухоль складається з трьох підоб'єктів: корпус, ручка та декоративна емблема. Wireframe-вигляд кухля та розбивку на складові частини наведено на рисунку 2.1.

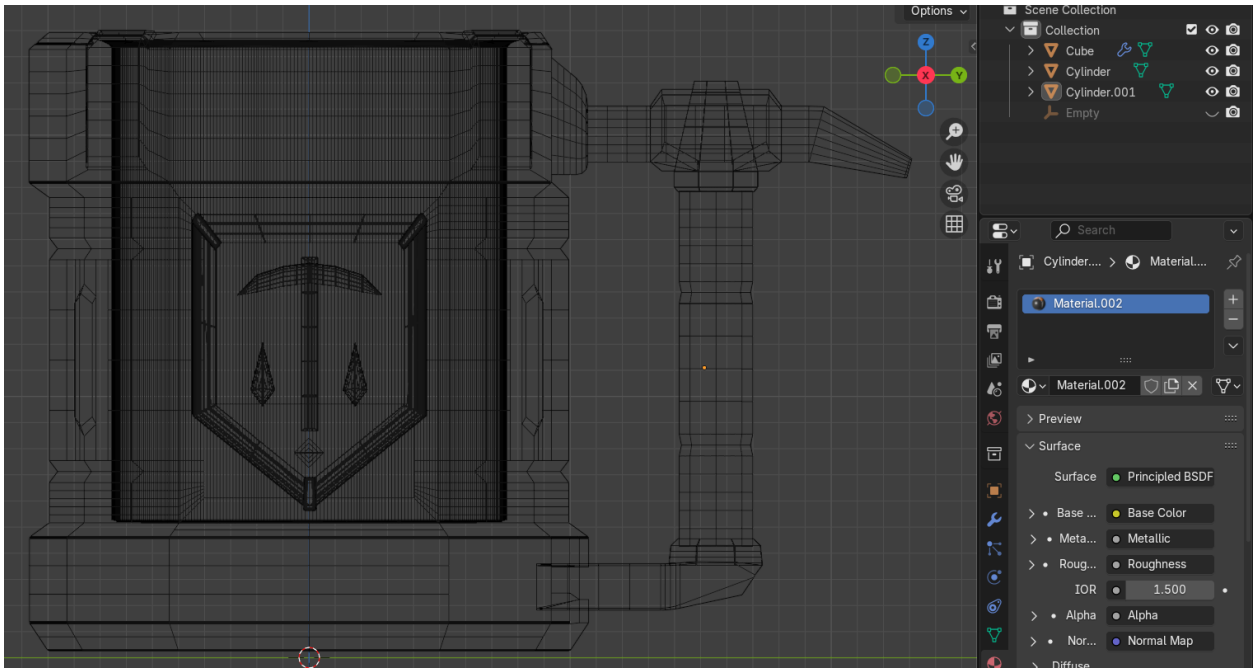

Рисунок 2.1. Wireframe-вигляд кухля та окремих його складових частин

Кайло є геометрично найскладнішим із трьох об'єктів, виконаних у Blender. Основу сформовано з прямокутних примітивів із подальшим масштабуванням та злиттям. Циліндричні елементи кріплення введено до складу моделі як окремі примітиви. Руків'я являє собою витягнутий циліндр із деталями на обох кінцях. Бокові леза сформовано через екструзію граней із застосуванням Bevel для загостреного краю. Модель налічує сімнадцять підоб'єктів, що відображає механічну складність виробу. Wireframe-вигляд кайла та розбивку на складові частини наведено на рисунку 2.2.

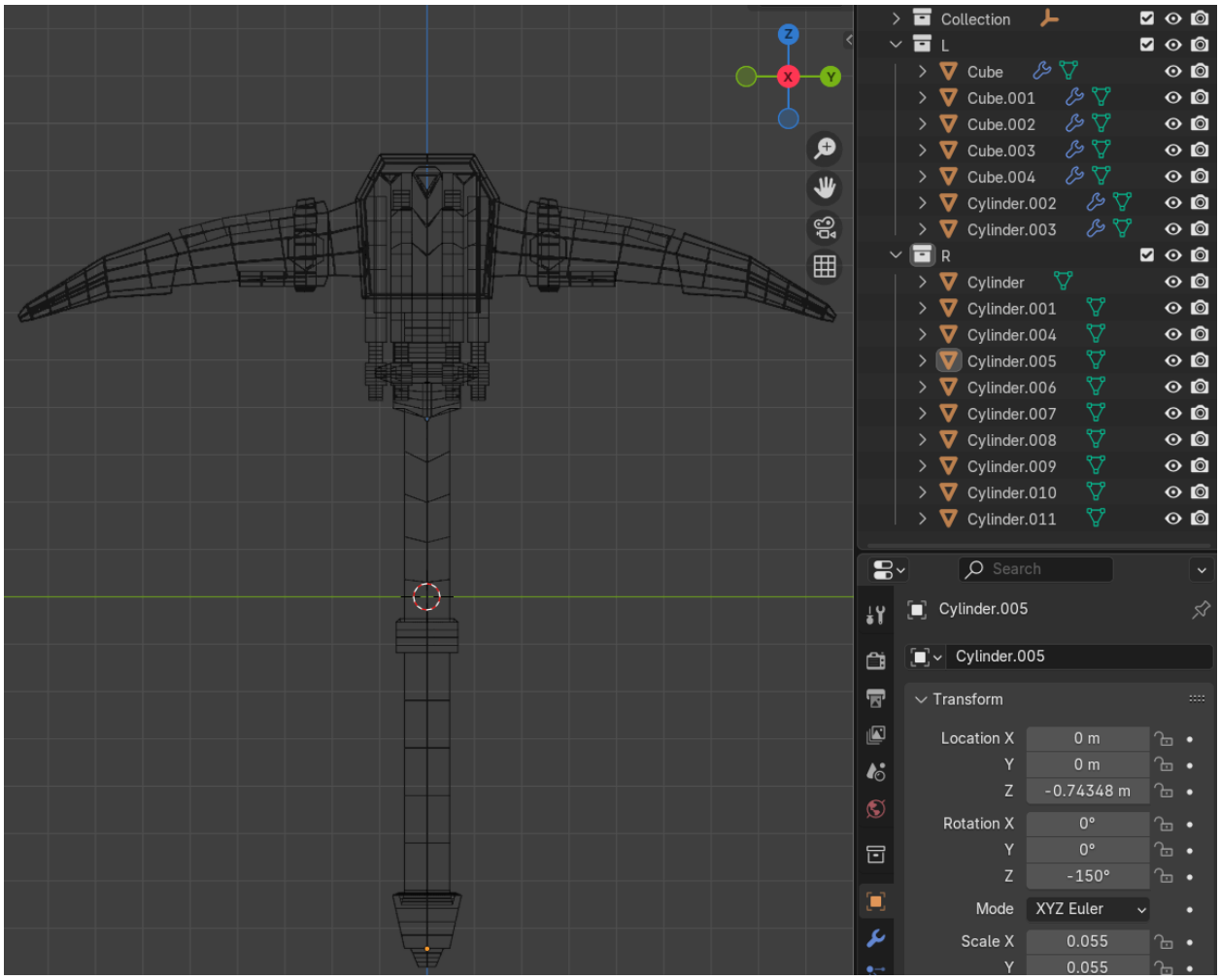

Рисунок 2.2. Wireframe-вигляд кайла та окремих його складових частин

Робот Боско побудований на основі кубічного примітива, що формує основний корпус. Кришка, бічні панелі, ліхтарне вікно та кінцівки виконані як окремі меші. Панельний рельєф на корпусі сформовано через Inset Faces із зміщенням вставленої грані всередину, що забезпечує чіткі кутові ребра без

збільшення загальної кількості полігонів. Капелюх і краватка-метелик додані на фінальному етапі після завершення основної геометрії. Загальна кількість підоб'єктів понад дев'ятнадцяти. Wireframe-вигляд Боска та розбивку на підоб'єкти наведено на рисунку 2.3.

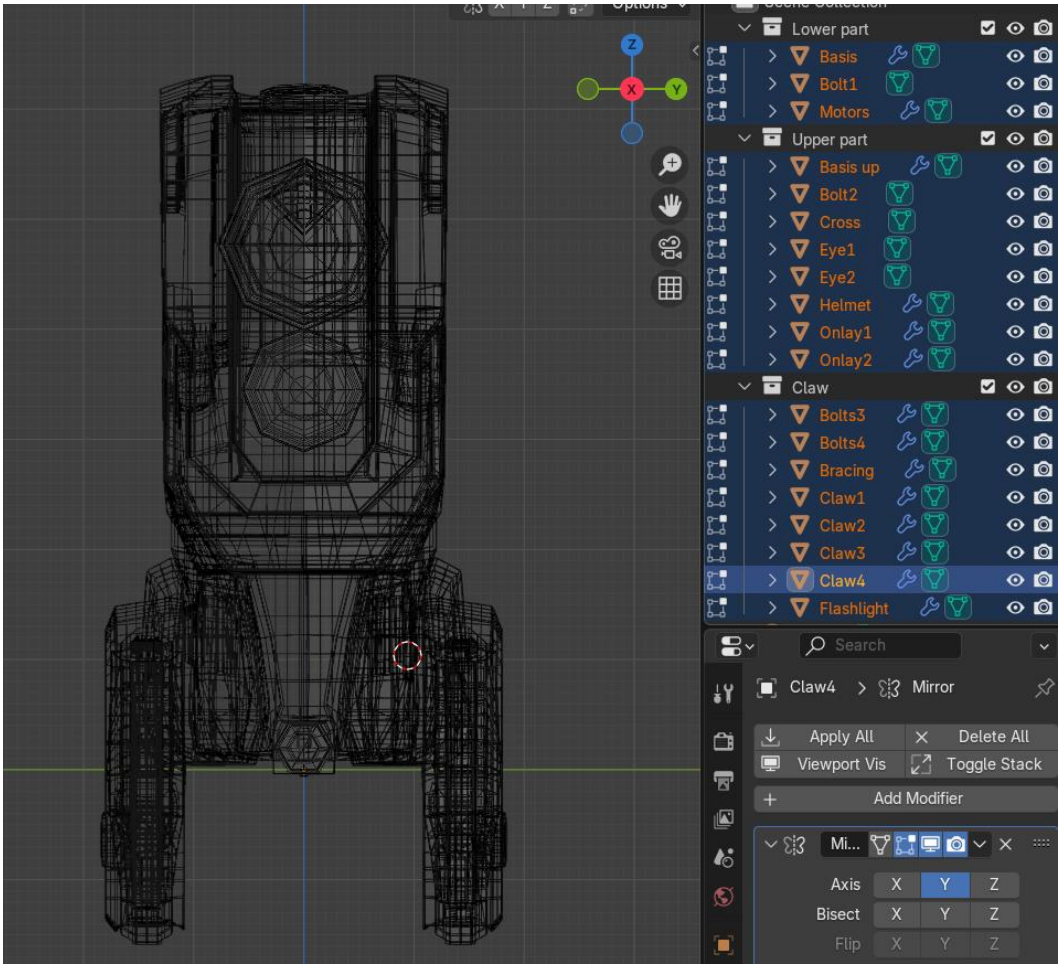

Рисунок 2.3. Wireframe-вигляд робота Боско та розбивка на підоб'єкти

При роботі над усіма трьома моделями модифікатори застосовувались у неруйнівному режимі. Mirror Modifier забезпечував симетрію там, де вона потрібна. Subdivision Surface підключався тимчасово для контрольного перегляду форм. Перед переходом до UV-роботи всі модифікатори зафіксовано через Apply.

2.2.2 Створення моделі у ZBrush

Модель Дворфа, на відміну від трьох попередніх об'єктів, вимагала застосування методу цифрового скульптингу. Органічний персонаж з

анатомічними формами та деталями спорядження не може бути ефективно відтворений методом полігонального моделювання через складність ручного відтворення природного рельєфу поверхні.

Підготовчий етап включав розробку паперового ескізу: проєкції спереду та збоку з розбивкою на ключові анатомічні елементи та пропорції. У виробничому середовищі такий підхід позначається терміном «оберт персонажа» (англ. character turnaround) і дозволяє остаточно вирішити питання пропорцій до початку роботи у тривимірному середовищі [27]. Попереднє блокування (від англ. blocking - попередня закладка основних форм) силуету виконувалось у растровому редакторі для швидкого погодження загальних пропорцій, після чого робота переходила до ZBrush.

У ZBrush скульптинг розпочинався у режимі Dynamesh, що автоматично перерозподіляє геометрію при операціях злиття та додавання маси. Базові частини тіла формувались як окремі субтули: тулово, плечі, руки, ноги, кисті та голова. Загальна кількість субтулів перевищує десять. На піку деталізації кількість полігонів high-poly версії становила 3 640 000, що є типовим показником для персонажа такого рівня у ZBrush [28]. Скульпт Дворфа, вигляд спереду, наведено на рисунку 2.4.

Рисунок 2.4. ZBrush-скульпт Дворфа, вигляд спереду

Деталізація поверхонь виконувалась стандартним набором кистей: Dam Standard для різьблення тонких ліній та швів, Clay Buildup для нарощування

об'єму, Standard для загального формування, Smooth для вирівнювання переходів між формами. Борода та бакенбарди побудовані як окремі субтули через ZSphere з подальшою деталізацією кистями IMM Hair.

Розгортка для неоптимізованих версій застосовувалась Smart UV Project. При нанесенні контрольної текстури (checker map) нерівномірність масштабу острівців проявляється у вигляді клітинок різного розміру, особливо помітного спотворення на циліндричних поверхнях та кутових зонах. UV-листи неоптимізованих версій (додаток Б).

Кількість підоб'єктів та полігональні характеристики моделей до оптимізації наведено в таблиці 2.1.

Таблиця 2.1

Кількість підоб'єктів та полігонаж моделей до оптимізації

Модель	**Кількість підоб'єктів**	**Полігони до оптимізації**	**Трикутники до оптимізації**	**Розмір файлу (МБ)**
Кухоль	3	46 330	93 180	4
Кайло	17	49 700	99 190	6
Боско	19	140 480	280 710	9
Дворф	34	3 640 000	7 284 300	192

2.3 Оптимізація тривимірних моделей

Оптимізація кухля, кайла і Боска проводилась у Blender 4.3. Для Дворфа перший етап виконано у ZBrush за допомогою ZRemesher, фінальне доопрацювання у Blender 4.3. Послідовність кроків однакова для всіх об'єктів.

Видалення надлишкових петльових ребер. Операція Dissolve Edges в Edit Mode видаляла надлишкові петлі без зміни силуету. Скорочення полігонажу після цього кроку: кайло 22%, кухоль 18%, Боско 15%. Видимих змін форми не зафіксовано.

Усунення N-гонів (полігонів із п'ятьма і більше вершинами). Кожен виявлений через Mesh Analysis N-гон розбивався вручну через Loop Cut або

Knife. Автоматична тріангуляція не застосовувалась, оскільки алгоритм не контролює напрямок діагональних ребер і може спричиняти небажані тіньові артефакти. У Дворфа після ZRemesher N-гони відсутні. N-гони у геометрії кайла, підсвічені режимом Mesh Analysis, наведено на рисунку 2.5.

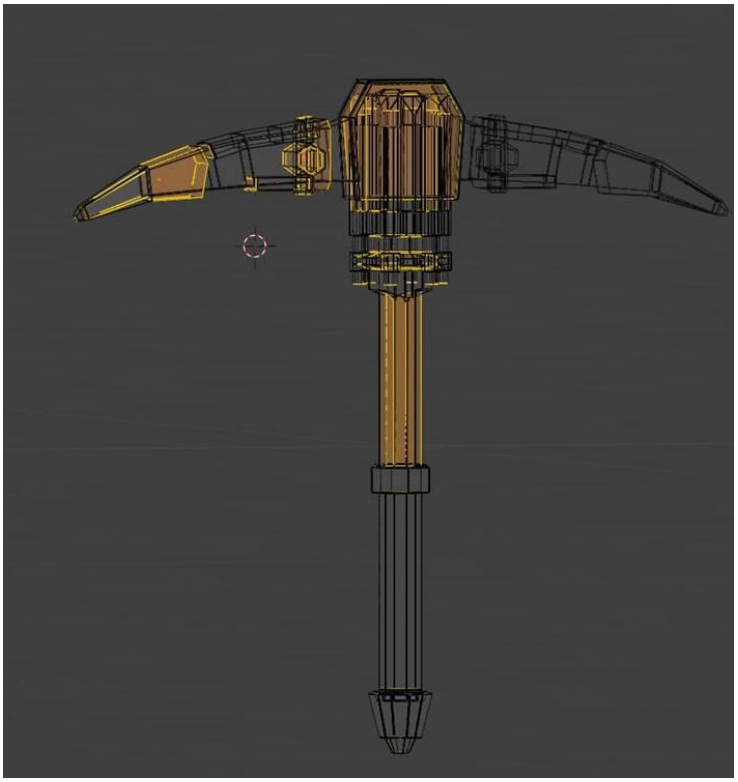

Рисунок 2.5. N-гони у геометрії кайла, підсвічені режимом Mesh Analysis

Виправлення нормалей через Recalculate Outside (Shift+N). Режим Face Orientation Overlay після операції підтверджував правильний напрямок нормалей по всій поверхні. На Боско та кайлі кілька граней залишились інвертованими після ручних операцій злиття і виправлялись через Flip Normals.

Застосування Shade Smooth з Auto Smooth із граничним кутом 60 градусів. Це забезпечує інтерполяцію нормалей між суміжними гранями на циліндричних ділянках зі збереженням чіткості кутових ребер. Для Боска Shade Smooth застосовувався вибірково: лише до заокруглених частин, плоскі панелі залишались зі Shade Flat. Порівняння кухля зі Shade Flat та Shade Smooth наведено на рисунку 2.6.

Рисунок 2.6 Порівняння кухля зі Shade Flat та Shade Smooth з Auto Smooth

UV-розгортка для всіх чотирьох оптимізованих моделей. Шви розмічались через Mark Seam у місцях з мінімальною помітністю при стандартних ракурсах: знизу об'єкта, вздовж внутрішніх кромок, по задніх поверхнях. UV-листи оптимізованих версій (додаток В).

Після завершення розгортки моделі експортовано у форматі FBX із попереднім Apply Transformations. Текстурування виконувалось у Adobe Substance 3D Painter 2025. Для кожної моделі підготовлено набір карт: Albedo, Normal Map, Metallic/Roughness та Ambient Occlusion. Для ліхтарного вікна Боска та деталей спорядження Дворфа додатково підготовлено Emission-карту. Фінальний вигляд оптимізованих моделей у Blender наведено на рисунку 2.7.

Рисунок 2.7 Фінальний вигляд оптимізованих моделей у Blender

Технічні показники після оптимізації наведено в таблиці 2.2.

Таблиця 2.2

Кількість підоб'єктів та полігонаж моделей після оптимізації

Модель	**Кількість підоб'єктів**	**Полігони до оптимізації**	**Трикутники до оптимізації**	**Розмір файлу (МБ)**
Кухоль	1	4140	6600	1
Кайло	1	3400	5230	1
Боско	9	8720	17320	3
Дворф	1	120000	243450	10

2.4 Рендеринг тривимірних моделей до та після оптимізації

Рендеринг усіх чотирьох моделей виконувався у Blender EEVEE в однакових умовах: нейтральний темний фон, три точкових джерела світла для забезпечення читабельного об'єму з вираженим тіньовим боком і підсвіткою силуету. Кожна модель рендерилась окремо в ізольованій сцені, що унеможливлює взаємний вплив об'єктів на показники часу рендерингу. Вимірювання виконувались двічі для кожного варіанта, фіксувалось середнє значення.

До початку рендерингу кожна модель перевірялась засобами Mesh Analysis у Blender. Перевірка виявила N-гони у геометрії кайла та Боска. При рендерингу неоптимізованих версій ці полігони проявлялись як нечіткі тіньові плями на поверхні, особливо виражені при бічному куті падіння світла. Кухоль та Дворф N-гонів не містили, тому зазначений артефакт для них не зафіксовано.

Технічні показники до оптимізації наведено в таблиці 2.3.

Таблиця 2.3

Технічні показники моделей до оптимізації

Модель	Полігони	Трикутники	Час рендерингу (с)
Кухоль	46 330	93 180	00.00.98
Кайло	49 700	99 190	00.01.43
Боско	140 480	280 710	00.15.83
Дворф	3 640 000	7 284 300	00.03.77

Значний час рендерингу Боска (15,83 с) при відносно меншому полігонажі порівняно з Дворфом обумовлений кількістю підоб'єктів: дев'ятнадцять незалежних мешів генерують відповідну кількість draw calls, кожен з яких потребує окремої команди CPU до GPU. Дев'ятнадцять окремих мешів із незалежними матеріалами генерують відповідну кількість draw calls. Час рендерингу Дворфа (3,77 с) при полігонажі 3640000 є вищим за очікуваний для одного підоб'єкта, що обумовлено граничним завантаженням геометричного конвеєра GPU при обробці мільйонних масивів вершин.

Технічні показники після оптимізації наведено в таблиці 2.4.

Таблиця 2.4

Технічні показники моделей після оптимізації

Модель	Полігони	Трикутники	Час рендерингу (с)
Кухоль	4140	6600	00.00.73
Кайло	3400	5230	00.00.44
Боско	8720	17320	00.05.53
Дворф	120000	243450	00.01.48

2.5 Порівняльний аналіз результатів до та після оптимізації

Аналіз охоплює чотири аспекти: скорочення полігонажу, зміну часу рендерингу, якість UV-розгортки та усунення геометричних артефактів. Зведені кількісні показники наведено в таблиці 2.5.

Таблиця 2.5

Зведені порівняльні показники до та після оптимізації

Модель	**Полігони до**	**Полігони після**	**Скорочення (%)**	**Час до (с)**	**Час після (с)**	**Скорочення часу (%)**
Кухоль	46330	4140	91,1	00.00.98	00.00.73	25,5
Кайло	49700	3400	91,7	00.01.43	00.00.44	69,2
Боско	140480	8720	93,7	00.15.83	00.05.53	65,1
Дворф	3640000	120000	96,7	00.03.77	00.01.48	60,7

За результатами оптимізації скорочення кількості полігонів становить від 91,1% для кухля до 96,7% для Дворфа. Найбільше абсолютне скорочення зафіксовано для Дворфа: з 3 640 000 до 120 000 полігонів, що обумовлено специфікою методу цифрового скульптингу, який генерує значно більший полігонаж, ніж полігональне моделювання. Для предметних об'єктів (кухоль, кайло) скорочення є порівнянним: 91,1% та 91,7%. Попри різну початкову кількість підоб'єктів і складність геометрії.

Між скороченням полігонажу та скороченням часу рендерингу відсутня лінійна залежність. Кухоль при скороченні полігонажу на 91,1% показав скорочення часу рендерингу лише на 25,5%, тоді як кайло при схожому скороченні полігонажу (91,7%) дало 69,2% зменшення часу. Ця розбіжність пояснюється тим, що шейдинг та обробка текстурних карт займають фіксовану частку часу кадру незалежно від кількості трикутників [29]. Найбільше абсолютне скорочення часу зафіксовано для Боска: з 15,83 с до 5,53 с (65,1%), що безпосередньо пов'язано зі зменшенням кількості підоб'єктів з дев'ятнадцяти до дев'яти і відповідним скороченням кількості draw calls.

Автоматична розгортка Smart UV Project, застосована на неоптимізованих версіях усіх чотирьох моделей, продемонструвала характерні артефакти нерівномірної щільності текселів. При накладанні контрольної текстури (checker map) нерівномірність масштабу UV-острівців проявляється у вигляді клітинок різного розміру, особливо вираженого на циліндричних поверхнях та в кутових зонах. Для Дворфа на неоптимізованій

версії застосовувався (UV Master) інструмент ZBrush, що дав перетягнуту розгортку без рівномірного розподілу щільності. Ручна UV-розгортка на оптимізованих версіях забезпечила рівномірну щільність текселів по всіх поверхнях кожного об'єкта. Порівняльний рендер неоптимізованих й оптимізованих моделей (додаток Г).

Показники геометрії та розгортки узагальнено в таблиці 2.6.

Таблиця 2.6

Показники геометрії та розгортки

Показник	**Кухоль**	**Кайло**	**Боско**	**Дворф**
N-гони до оптимізації	Відсутні	Наявні	Наявні	Відсутні
N-гони після оптимізації	Відсутні	Відсутні	Відсутні	Відсутні
Розгортка до оптимізації	Smart UV (деформована)	Smart UV (деформована)	Smart UV (деформована)	UV Master (перетянута)
Розгортка після оптимізації	Ручна (рівна)	Ручна (рівна)	Ручна (рівна)	Ручна (рівна)
Тіньові артефакти до	Відсутні	Наявні	Наявні	Відсутні
Тіньові артефакти після	Відсутні	Відсутні	Відсутні	Відсутні

За умови коректно виконаної оптимізації, ручної UV-розгортки та запікання PBR-карт оптимізовані моделі при перегляді зі стандартних ігрових ракурсів візуально не поступаються вихідним версіям. Карта нормалей відтворює дрібний поверхневий рельєф, що у вихідних версіях забезпечувала геометрія. Карта Ambient Occlusion підсилює глибину у западинах і кутових зонах. Сукупний ефект цих карт дозволяє low-poly моделі зберігати сприйняту деталізацію оригіналу при суттєво нижчому полігонажі.

Отримані результати підтверджують, що послідовне застосування методів оптимізації: видалення надлишкових петльових ребер, усунення N-гонів, виправлення нормалей, ручна UV-розгортка та запікання текстурних карт забезпечує скорочення полігонажу на 91–97% та часу рендерингу на 25–69% без втрати візуальної якості при перегляді з типових ігрових ракурсів [30].

Висновки до розділу 2

У другому розділі сформульовано технічні специфікації для чотирьох об'єктів різного типу: органічного персонажа, механічного персонажа, предмета зі змішаною геометрією та простого твердотільного об'єкта що дозволило зафіксувати контрольовані умови для подальшого порівняльного аналізу.

Вибір методу моделювання визначався типом об'єкта. Полігональне моделювання у Blender 4.3 забезпечило точний контроль топології для кухля, кайла та Боска. Скульптинг у ZBrush 2022 дозволив відтворити органічні анатомічні форми Дворфа з рівнем деталізації, недосяжним полігональним методом, однак high-poly версія з 3640000 полігонів підтвердила принципову непридатність скульпт-результату для рушія без подальшої ретопології.

Оптимізація за єдиною послідовністю кроків: видалення надлишкових петльових ребер, усунення N-гонів, виправлення нормалей, ручна UV-розгортка та запікання текстурних карт дала скорочення полігонажу від 91,1% для кухля до 96,7% для Дворфа. Скорочення часу рендерингу склало від 25,5% до 69,2% і не корелює лінійно зі скороченням полігонажу. Для Боска визначальним чинником виявилось зменшення кількості підоб'єктів з дев'ятнадцяти до дев'яти, що скоротило кількість draw calls і дало найбільше абсолютне скорочення часу рендерингу серед усіх об'єктів. Для кухля, навпаки, фіксовані витрати на шейдинг і обробку текстурних карт обмежили скорочення часу 25,5% попри 91,1% скорочення полігонажу.

Автоматична UV-розгортка Smart UV Project на неоптимізованих версіях продемонструвала нерівномірну щільність текселів на циліндричних

поверхнях і в кутових зонах. Ручна розгортка з прихованими швами усунула цей недолік і забезпечила коректне відтворення PBR-текстур. Усунення N-гонів через ручне розбиття повністю ліквідувало тіньові артефакти на кайлі та Боску. Заміна геометричного рельєфу картами нормалей і Ambient Occlusion дозволила зберегти сприйняту деталізацію при суттєво нижчому полігонажі.

РОЗДІЛ 3

ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ТЕСТУВАННЯ ТРИВИМІРНИХ ОБ'ЄКТІВ В ІНТЕРАКТИВНОМУ СЕРЕДОВИЩІ

3.1 Методика тестування в Unity 6 URP

Практичне тестування виконувалось у середовищі Unity 6 (гілка 6000.x, що базується на ядрі Unity 2022 LTS) з рендер-пайплайном Universal Render Pipeline. Вибір середовища обумовлений практичними потребами роботи. URP дає фізично коректне освітлення за PBR, однаково стабільно веде себе на мобільному залізі й на настільних ПК, а текстури з Substance 3D Painter 2025 за пресетом Unity URP підключаються одразу без додаткових кроків.

Тестовий стенд мав однакову конфігурацію на всіх етапах роботи. Процесор Intel Core i5-12500 (2,50 ГГц, 12-те покоління), відеокарта NVIDIA GeForce RTX 3050 Laptop GPU з 4 ГБ відеопам'яті, оперативна пам'ять 16 ГБ. Зміна апаратної конфігурації в процесі дослідження унеможливила б коректне порівняння результатів вимірювань.

Під кожен із двох наборів моделей готувалась окрема сцена, але налаштування в обох були абсолютно однаковими. Єдине джерело направленого світла інтенсивність 1,0, кут нахилу 50 градусів. Фон суцільний темний колір. Стандартні ефекти постобробки URP увімкнені, їхні параметри між сценами не змінювались. Вікно Game View зафіксоване на роздільній здатності 1920x1080, камера стоїть на незмінній відстані від геометричного центру сцени. Усі чотири моделі завжди присутні в сцені одночасно без дублювань і замінників.

Bloom і Tonemapping є типовими компонентами постобробки у проєктах на Unity URP та вносять однаковий фіксований внесок у кадровий бюджет обох тестових сцен, що унеможливлює їхній вплив на порівняльні результати. Фізика та ігрова логіка навмисно вимкнені щоб у вимірюваннях відображався виключно вплив геометрії й текстур.

Продуктивність вимірювалась двома інструментами паралельно. Stats у вікні Game View давав агреговану картину в реальному часі: FPS, час кадру, кількість Draw Calls, кількість трикутників і вершин. Показники знімались у режимі Play Mode. Unity Profiler давав покадровий аналіз навантаження. В розділі Rendering записувався час виконання GPU та CPU, в розділі Memory відстежувались загальна виділена пам'ять (Total Allocated Memory) та розрахунковий обсяг графічної пам'яті (Estimated Graphics Memory).

Кожне вимірювання проводилось за однаковою схемою. Після запуску сцени в Play Mode вичікувалось п'ять секунд, поки рушій добудовував шейдери та підвантажував ресурси у відеопам'ять. Далі йшли дві незалежні сесії тривалістю не менше 30 секунд кожна. У кожній сесії записувались мінімальне, максимальне та середнє значення, а до підсумкових таблиць потрапляло середнє з двох сесій.

3.2 Імпорт та підготовка моделей в Unity 6

Перенести модель з Blender 4.3 до Unity 6 без артефактів завдання не тривіальне. Причина в різних системах координат: Blender використовує праворуку з вертикальною віссю Z, Unity ліворуку з вертикальною віссю Y. Якщо параметри конвертації не задати явно під час експорту, об'єкт в рушії опиниться розвернутим у непередбачуваний бік.

Для всіх чотирьох моделей застосовувались однакові параметри FBX-експорту (рис. 3.1). Forward задає напрямок осі глибини після конвертації. Up відповідає вертикальній осі Unity. Apply Transform записує поточні значення трансформації безпосередньо в геометрію, без цього масштаб в Unity часто виявляється некоректним. Apply Unit перетворює метричні одиниці Blender у формат, який рушій читає правильно. Apply Modifiers фіналізує стек модифікаторів і передає готову геометрію. Limit to обмежує експорт лише вибраними об'єктами, виключаючи допоміжну геометрію сцени. Smoothing передає згладжування на рівні граней, що дозволяє коректно перенести Shade Smooth / Shade Flat.

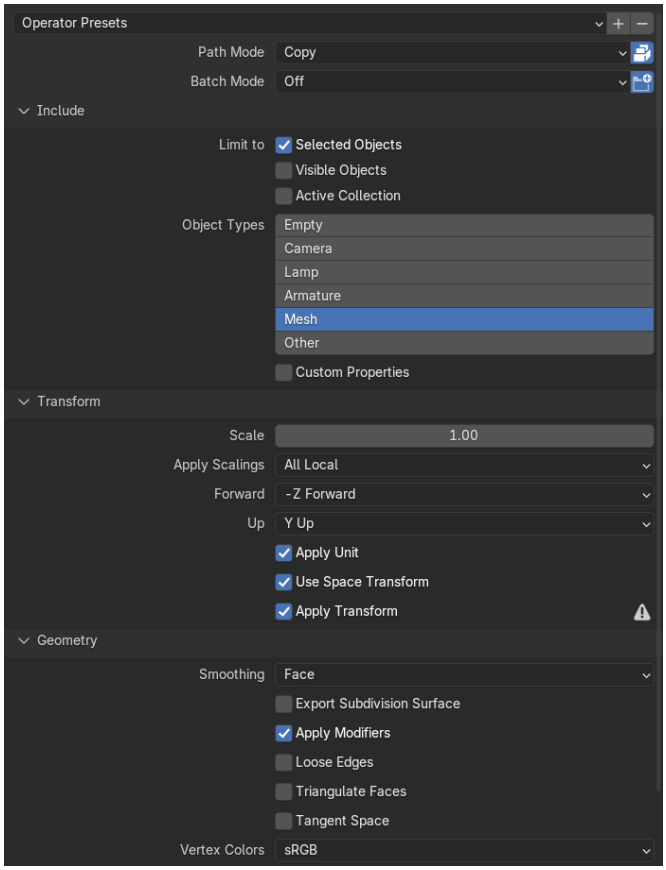

Рис. 3.1 Параметри FBX-експорту в Blender

FBX-файли імпортувались безпосередньо до директорії Assets проєкту Unity. Unity 6 автоматично розпізнає геометрію, UV-координати та назви матеріальних слотів, визначених у Blender. Для підключення текстур використовувалась функція Extract Textures в Inspector імпортованого FBX, вона витягує матеріальні слоти й створює окремі матеріальні активи в Assets.

Після імпорту кожної моделі перевірялись три речі. Орієнтація: чи правильно визначено напрямки «вгору» та «вперед». Завдяки коректним налаштуванням експорту жодна модель ручного коригування не вимагала. Масштаб: моделі порівнювались між собою в сцені, а Дворф як антропоморфний персонаж слугував масштабним орієнтиром. Apply Transform гарантував, що значення Scale у компоненті Transform кожного об'єкта дорівнює (1, 1, 1). По-третє, нормалі: некоректно орієнтована нормаль видає себе ділянкою, яка залишається темною за будь-якого кута освітлення, або інвертованим відблиском. На робота Боско після першого імпорту саме такі ділянки виявились на бічних панелях корпусу, наслідок злиття мешів на

фінальному етапі моделювання. Проблему усунули поверненням до Blender, перевіркою через Face Orientation Overlay і точковим застосуванням Flip Normals до проблемних граней з повторним експортом. Цей випадок добре ілюструє, чому нормалі варто перевіряти саме в рушії: конвертація між правою та лівою системами координат може інвертувати нормаль грані, орієнтованої вздовж перетвореної осі, і це не завжди помітно в самому Blender.

3.3 Аналіз продуктивності сцени з усіма моделями

Перше вимірювання була сцена з неоптимізованими версіями всіх чотирьох моделей, друга та сама сцена, але з оптимізованими. Параметри між вимірюваннями не змінювались. Зведені результати наведено в таблиці 3.1. Сцени з усіма оптимізованими та неоптимізованими моделями в Unity (додаток Д).

Таблиця 3.1

Порівняльні показники продуктивності сцени з усіма моделями

Показник	**Неоптимізована сцена**	**Оптимізована сцена**	**Зміна**
Продуктивність			
FPS (середнє)	55,5	252,0	+354%
FPS (мін.)	55,4	250,1	-
FPS (макс.)	55,6	253,8	-
Global Frametime (мс)	18,0	3,95	-78%
GPU Frametime (мс)	14,9	2,5	-83%
CPU Main Thread (%)	91,6	60,1	-34%
Render Thread (%)	8,5	40,0	+370%
Пам'ять			
Total Allocated Memory (МБ)	4015,5	3838,5	-177 МБ
Estimated Graphics Memory (МБ)	1425,1	510,4	-914, 7 МБ
Total Texture Memory (МБ)	306,5	275,7	-30,8 МБ

FPS зріс із 55,5 до 252,0 кадрів на секунду у 4,5 рази. Принциповим є те, що значення 55,5 кадру/с вже нижче цільового порогу 60 FPS. Такий результат означає, що тільки чотири об'єкти систематично перевищують бюджет кадру 16,67 мс. У реальному проєкті, де сцена містить десятки або сотні елементів, такий старт неминуче обернувся б помітними просадками.

GPU Frametime впав із 14,9 до 2,5 мс скорочення на 83%. Цей показник точніше за FPS відображає завантаженість графічного конвеєра: він не залежить від V-Sync і подібних обмежень синхронізації. 14,9 мс означає, що GPU витрачав майже весь кадровий бюджет лише на обробку геометрії та текстур чотирьох об'єктів.

Заслуговує на увагу перерозподіл навантаження між потоками виконання. У неоптимізованій сцені CPU Main Thread був зайнятий на 91,6% процесор витрачав частку часу кадру на формування команд для GPU. В оптимізованій картина змінилась: CPU Main Thread 60,1%, Render Thread 40,0%. Такий баланс суттєво здоровіший: Render Thread бере на себе більшу частку паралельної роботи, а головний потік CPU вивільняється для ігрової логіки.

Estimated Graphics Memory скоротилась із 1 425,1 до 510,4 МБ на 914,7 МБ, тобто на 64,4%. Для пристрою з 4 ГБ VRAM це критично: неоптимізована сцена вже з чотирма моделями займала понад третину всієї відеопам'яті. Якщо масштабувати цю тенденцію на сотні об'єктів реального проєкту VRAM рано чи пізно переповниться, і GPU почне передавати дані через системну оперативну пам'ять, що різко б'є по продуктивності незалежно від обсягу RAM.

3.4 Аналіз продуктивності окремих моделей

Ізольоване тестування кожної моделі дозволяє кількісно оцінити індивідуальний внесок об'єкта в загальний бюджет кадру та виявити взаємозв'язок між типом геометрії та характером навантаження на рушій. Порівняння показників кухля наведено в таблиці 3.2.

Таблиця 3.2

Порівняльні показники кухля

Показник	Неоптимізована	Оптимізована	Зміна
FPS (середнє)	230,6	313,2	+35,8%
Global Frametime (мс)	4,4	3,2	-27,3%
GPU Frametime (мс)	2,9	2,1	-27,6%
GPU Main Thread (%)	58,9	50,3	-14,6%
Render Thread (%)	41,1	49,7	+20,9%
Total Allocated Memory (МБ)	5418,0	3829,6	-1588,4 МБ
Estimated Graphis Memory (МБ)	1428,2	491,7	-65,6%

Кухоль є геометрично найпростішим об'єктом вибірки. Приріст FPS склав 35,8%, найменший результат серед чотирьох моделей. Це узгоджується з результатами розділу 2: навіть при скороченні полігонажу на 91,1% (з 46 330 до 4 140 граней) час рендерингу в EEVEE знизився лише на 25,5%. Фіксовані витрати на шейдинг та обробку текстурних карт займають значну частку часу кадру незалежно від кількості трикутників [31], і для простого об'єкта з невеликою початковою геометрією ця частка є відносно вищою.

Скорочення Estimated Graphics Memory на 65,6% пояснюється переходом від трьох підоб'єктів із трьома окремими наборами текстур до єдиного меша з одним матеріалом після об'єднання.

Кайло є характерним прикладом об'єкта зі змішаною геометрією: 17 підоб'єктів до оптимізації, зведені до єдиного меша після. Приріст FPS склав 24,2%, що менше ніж у кухля, попри більше початкове число підоб'єктів. Ключова причина: GPU Frametime знизився лише з 2,8 до 2,7 мс. Це мінімальна зміна, що вказує на те, що у неоптимізованій версії кайла основним вузьким місцем був не геометричний конвеєр GPU, а саме кількість полігонів від 17 підоб'єктів, яка навантажувала CPU Main Thread. Зведення до одного меша усунуло більшість зайвих Draw Calls, і приріст продуктивності

відобразився насамперед у показнику Global Frametime. Детально розглянемо показники кайла, які знаходяться в таблиці 3.3.

Таблиця 3.3

Порівняльні показники кайла

Показник	**Неоптимізована**	**Оптимізована**	**Зміна**
FPS (середнє)	220,7	274,1	+24,2%
Global Frametime (мс)	4,55	3,65	-19,8%
GPU Frametime (мс)	2,8	2,7	-3,6%
GPU Main Thread (%)	56,3	58,4	+3,7%
Render Thread (%)	43,7	41,7	-4,6%
Total Allocated Memory (МБ)	5440,4	3888,1	-1552,3 МБ
Estimated Graphis Memory (МБ)	1428,4	507,3	-64,5%

Для кайла CPU Main Thread виріс з 56,3% до 58,4%, а Render Thread впав з 43,7% до 41,7% тобто динаміка протилежна порівняно з іншими моделями.

Порівняльні показники Боска наведено в таблиці 3.4.

Таблиця 3.4

Порівняльні показники Боска

Показник	**Неоптимізована**	**Оптимізована**	**Зміна**
FPS (середнє)	90,5	260,4	+187,7%
Global Frametime (мс)	11,5	3,85	-65,2%
GPU Frametime (мс)	9,55	2,5	-73,8%
GPU Main Thread (%)	88,4	57,3	-35,2%
Render Thread (%)	11,6	42,7	+268%
Total Allocated Memory (МБ)	5394,9	3838,7	-1556,2 МБ
Estimated Graphis Memory (МБ)	1427,7	507,6	-64,4%

Боско демонструє найбільший приріст продуктивності серед усіх чотирьох об'єктів: FPS зріс з 90,5 до 260,4 приріст 187,7%. Значення 90,5 FPS у неоптимізованій версії є найгіршим серед предметних моделей, хоча полігонаж Боска (140 480 до оптимізації) порівнянний з іншими об'єктами. Причиною є структура: 19 підоб'єктів із окремими матеріалами генерують 19 Draw Calls лише від одного персонажа. Кожен Draw Call це окрема команда CPU до GPU із власними витратами на налаштування буферів, передачу стану шейдера та синхронізацію конвеєра; ці витрати не залежать від складності самої геометрії [32]. Це підтверджується показником CPU Main Thread 88,4% у неоптимізованій версії. Процесор майже повністю зайнятий диспетчеризацією команд рендерингу.

Після оптимізації CPU Main Thread знизився до 57,3%, в Render Thread зрфз до 42,7% паралельна частина конвеєра отримала значно більше ресурсів. GPU Frametime знизився з 9,55 до 2,5 мс (-73,8%).

Для підтвердження виявленої закономірності розглянемо показники органічної моделі Дворфа (таблиця 3.5).

Таблиця 3.5

Порівняльні показники Дворфа

Показник	**Неоптимізована**	**Оптимізована**	**Зміна**
FPS (середнє)	93,2	255,3	+173,9%
Global Frametime (мс)	10,8	3,95	-63,4%
GPU Frametime (мс)	8,4	2,65	-68,5%
GPU Main Thread (%)	80,2	58,7	-26,9%
Render Thread (%)	19,8	41,3	+108,6%
Total Allocated Memory (МБ)	5415,6	3817,8	-1597,8 МБ
Estimated Graphis Memory (МБ)	1428,8	507,35	-64,5%

Дворф єдина органічна модель вибірки демонструє другий за величиною приріст FPS: з 93,2 до 255,3 (+173,9%). Характер навантаження від неоптимізованої версії Дворфа суттєво відрізняється від Боска. Незважаючи на зіставний показник FPS, причини різні: у Боска слабким місцем була кількість Draw Calls, у Дворфа обсяг геометрії. High-poly скульпт із 3640000 полігонами (7284300 трикутників) перевантажував геометричний конвеєр GPU: значення GPU Frametime 8,4 мс при одному об'єкті є критично високим. Після ретопології до 120 000 полігонів (243 450 трикутників) GPU Frametime знизився до 2,65 мс, скорочення на 68,5%.

Показник CPU Main Thread 80,2% у неоптимізованій версії пояснюється не Draw Calls, а витратами на трансформацію вершин: передача та обробка 7,2 мільйона трикутників потребує значних ресурсів саме CPU-сторони конвеєра на етапі підготовки буферів.

Зведемо показники й на основі показників сформулюємо закономірність. Зведені показники продуктивності представленні в таблиці 3.6.

Таблиця 3.6

Зведені показники приросту продуктивності по моделях

Модель	**FPS до**	**FPS після**	**Приріст FPS (%)**	**GPU Frametime до (мс)**	**GPU Frametime після (мс)**	**Est. GPU Memory до (МБ)**	**Est. GPU Memory після (МБ)**	**Зниження GPU Memory (%)**
Кухоль	230,6	313,2	+35,8	2,9	2,1	1 428,2	491,7	-65,6
Кайло	220,7	274,1	+24,2	2,8	2,7	1 428,4	507,3	-64,5
Боско	90,5	260,4	+187,7	9,55	2,5	1 427,7	507,6	-64,4
Дворф	93,2	255,3	+173,9	8,4	2,65	1 428,8	507,35	-64,5
Сцена (всі 4)	55,5	252,0	+354	14,9	2,5	1 425,1	510,4	-64,2

Дані таблиці 3.6 дозволяють сформулювати ключову закономірність: приріст продуктивності від оптимізації є нелінійним і залежить від

домінуючого джерела навантаження в конкретній моделі. Об'єкти з великою кількістю підоб'єктів отримують найбільший приріст від об'єднання мешів і скорочення Draw Calls. Органічні моделі з надвисоким полігонажем отримують найбільший приріст від ретопології. Прості об'єкти з помірним полігонажем мають менший потенціал для приросту, оскільки фіксовані витрати на шейдинг і текстурування займають вищу відносну частку їх кадрового бюджету.

Скорочення Estimated Graphics Memory є рівномірним по всіх моделях. Близько 64–66% що є прямим наслідком переходу від текстурних наборів неоптимізованих версій до уніфікованих наборів оптимізованих версій (2048x2048 або 1024x1024 залежно від типу об'єкта) зі стисненням файлів BC7/DXT5, що автоматично застосовується Unity при імпорті.

3.5 Аналіз якості відображення

Порівняння якості відображення проводилось при ідентичних умовах освітлення для обох версій кожної моделі. Знімки виконувались з фіксованої позиції камери для забезпечення коректного порівняння. Знімки моделей в Unity (додаток Е).

При спостереженні з ігрових ракурсів відстань від камери 2–5 метрів у масштабі сцени, кут огляду 60 градусів візуальна різниця між оптимізованими та неоптимізованими версіями мінімальна для всіх чотирьох об'єктів. Карта нормалей відтворює дрібний поверхневий рельєф, що у вихідних версіях забезпечувала безпосередньо геометрія. Карта Ambient Occlusion підсилює глибину у западинах, кутових зонах та місцях з'єднання підоб'єктів, компенсуючи відсутність геометричного затінення від дрібних деталей.

Дворф є найбільш показовим прикладом ефективності підходу. High-poly скульпт із 3,64 млн полігонів у Unity 6 взагалі не може бути використаний у виробничому проєкті в такому вигляді: значення GPU Frametime 8,4 мс для одного об'єкта залишає менше 8 мс на весь решта рушія при цільових 60 FPS. Оптимізована версія після ZRemesher-ретопології та запікання карти нормалей

зберегла всі характерні риси персонажа: силует, пропорції, фактуру бороди та деталі спорядження. Це підтверджує ефективність поєднання ZRemesher та запікання нормалей як основного методу для переносу органічних персонажів у рушій [33].

Боско підтвердив ефективність вибіркового застосування Shade Smooth: плоскі панелі корпусу зберегли чіткі грані та прямі межі, тоді як заокруглені деталі: кришка, кінцівки, ліхтарне вікно виглядають плавно без додаткової геометрії.

Предметні моделі такі як кайло та кухоль після оптимізації не виявили жодних помітних артефактів при ігрових ракурсах. Усунення N-гонів у кайлі ліквідувало тіньові плями, що були присутні в неоптимізованій версії.

При наближенні камери до поверхні на відстань менше 0,5 м у масштабі сцени ілюзія рельєфу від карти нормалей починає руйнуватись: силует оптимізованої моделі залишається плавним, тоді як у high-poly версії він зберігає геометричну деталізацію. Це є фізичним обмеженням методу запікання нормалей і не усувається підвищенням роздільної здатності карти [34].

Порівняння відображення моделей наведено в таблиці 3.7.

Таблиця 3.7

Якісне порівняння відображення моделей в Unity 6

Модель	**Візуальна якість (неопт.)**	**Візуальна якість (опт.)**	**Артефакти до**	**Артефакти після**	**Придатність для виробництва**
Кухоль	Висока	Висока	Відсутні	Відсутні	Обидві версії
Кайло	Середня	Висока	Тіньові плями (N-гони)	Відсутні	Лише оптимізована
Боско	Висока	Висока	Артефакти рендерингу	Відсутні	Обидві версії
Дворф	Неприйнятна (GPU Frametime 8,4 мс)	Висока	Критичне навантаження	Відсутні	Лише оптимізована

Висновки до розділу 3

Проведено практичну інтеграцію чотирьох тривимірних моделей до Unity 6 URP та виконано структуроване порівняльне тестування двох варіантів сцен за єдиним протоколом вимірювань.

Імпорт моделей у форматі FBX виконувався з фіксованим набором параметрів: –Z Forward, Y Up, Apply Transform, Apply Modifiers. Підключення текстур здійснювалось через функцію Extract Textures з подальшим ручним призначенням карт до шейдера Lit. Виявлені після імпорту проблеми з нормалями на Боску усунуто через повторну перевірку у Blender та повторний експорт.

Сцена з оптимізованими моделями показала середній FPS 252,0 проти 55,5 для неоптимізованого варіанту приріст 354%. GPU Frametime знизився з 14,9 до 2,5 мс (-83%). Estimated Graphics Memory скоротилась з 1425,1 до 510,4 МБ (-64,4%). Навантаження на CPU Main Thread знизилось з 91,6% до 60,1%, натомість Render Thread pic з 8,5% до 40,0%, що свідчить про покращення паралелізму конвеєра рендерингу.

Аналіз ізольованих вимірювань встановив, що характер і величина приросту продуктивності залежать від домінуючого джерела навантаження: для об'єктів із великою кількістю підоб'єктів на прикладі Боско: +187,7% FPS визначальним є скорочення Draw Calls. Для органічних моделей із надвисоким полігонажем на приклдаі Дворфа: +173,9% FPS ретопологія та запікання нормалей. Для простих предметних моделей (кухоль, кайло +24–36% FPS) ефект є меншим через домінування фіксованих витрат на шейдинг.

Неоптимізована версія Дворфа з GPU Frametime 8,4 мс є принципово непридатною для виробничого використання: у складній сцені цей показник унеможливлює досягнення цільових 60 FPS. Для решти моделей неоптимізовані версії технічно працездатні в ізольованому тесті, однак не

залишають достатнього кадрового бюджету для решти елементів реального ігрового проєкту.

Візуальна якість оптимізованих моделей при ігрових ракурсах відповідає неоптимізованим версіям: карти нормалей та Ambient Occlusion ефективно компенсують відсутність геометричного рельєфу. Єдиним виявленим обмеженням є деградація ілюзії рельєфу при екстремальному наближенні камери що є фізичним обмеженням методу, а не артефактом реалізації.

ВИСНОВКИ

Робота була спрямована на вивчення як саме створюються й оптимізуються тривимірні об'єкти для інтерактивних платформ, і що це дає з точки зору продуктивності. Вдалося підтвердити ефективність обраних методів, виміряти їхній вплив у конкретних цифрах. Завдання, сформульовані на початку роботи, реалізовано повністю.

Перший розділ заклав теоретичне підґрунтя дослідження. З'ясовано, що кожен тривимірний об'єкт у середовищі ігрового рушія несе одразу три навантаження: художнє, обчислювальне та фізичне. Детально розглянуто наявні методи побудови геометрії від класичного полігонального моделювання до фотограмметрії і для практичної частини свідомо обрано два з них. Blender 4.3 використовувався для предметних моделей, ZBrush 2022 для органічного персонажа. Окремо описано вимоги Unity URP до імпортованих активів та визначено кроки виробничого процесу оптимізації.

Другий розділ охопив безпосередню роботу з моделями. Для дослідження відібрано чотири об'єкти, що представляють різні категорії ігрових активів: кухоль, кайло, робот Боско та персонаж Дворф. Вихідний полігонаж коливався від кількох десятків тисяч граней для предметів до понад трьох мільйонів у скульпті Дворфа. Після послідовного проходження всіх етапів оптимізації. Від прибирання зайвих петельових ребер і виправлення N-гонів до запікання текстурних карт і ручної UV-розгортки. Геометрія скоротилась на 91–97% залежно від об'єкта. Зниження часу рендерингу при цьому виявилось нелінійним: де головним навантаженням була кількість підоб'єктів, ефект був помітнішим, ніж там, де основну частку бюджету кадру займали шейдинг і текстурування.

Третій розділ присвячено практичній інтеграції та тестуванню в Unity 6 URP. Порівняння двох ідентичних сцен показало разючу різницю: середній показник FPS зріс із 55,5 до 252,0 кадрів за секунду. Час обробки кадру на GPU скоротився більш ніж утричі, а розрахунковий обсяг відеопам'яті зменшився

приблизно на 915 МБ. Показово, що навантаження на головний потік процесора впало з 91,6% до 60,1%, звільнивши ресурси для ігрової логіки. Скульпт Дворфа у вихідному вигляді виявився непридатним для реального проєкту через критичне навантаження на GPU, тоді як після ретопології та запікання нормалей він повністю вписався у виробничі вимоги. Усі чотири оптимізовані моделі зберегли візуальну якість при стандартних ігрових ракурсах.

Головний практичний висновок роботи полягає в наступному: оптимізація не руйнує деталізацію об'єкта, а переносить її з геометрії у текстурні карти. Грамотно виконана ретопологія з наступним запіканням нормалей дає результат, який гравець не відрізняє від оригінального скульпту, проте рушій обробляє у рази швидше. Отримані матеріали можуть бути використані у виробництві контенту для ігор та AR/VR середовищ, а також як навчальний приклад із повним циклом роботи над тривимірним активом.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Akenine-Möller T., Haines E., Hoffman N. Real-Time Rendering. 4th ed. Boca Raton : CRC Press, 2018. 1198 с.
2. Gregory J. Game Engine Architecture. 3rd ed. Boca Raton:CRC Press,2018. 1240 с.
3. Ingrassia T. 3D Game Textures: Create Professional Game Art Using Photoshop. 3rd ed. Burlington : Focal Press, 2012. 384 с.
4. Guédon J.-P. Photogrammetry and 3D Reconstruction. Paris : Eyrolles, 2021. 280 с.
5. Vaughan W. Digital Modeling. Berkeley : New Riders, 2012. 408 с.
6. Blender Foundation. Blender 2.80 Release Notes. URL: https://www.blender.org/download/releases/2-80/ (дата звернення: 10.03.2025).
7. Blender Foundation. Subdivision Surface Modifier. Blender 4.3 Manual. URL: https://docs.blender.org/manual/en/4.3/modeling/modifiers/generate/subdivision_surface.html (дата звернення: 10.03.2026).
8. Pixologic. ZBrush Features: Pixol Technology. URL: https://pixologic.com/zbrush/features/ (дата звернення: 12.03.2026).
9. Keller M. ZBrush Character Creation: Advanced Digital Sculpting. 2nd ed. Indianapolis : Sybex, 2012. 456 с.
10. Palamar T. Mastering Autodesk Maya 2016. Indianapolis : Sybex, 2015. 912 с.
11. Reese M. Houdini on the Spot: Time-Saving Tips and Shortcuts from the Pros. Burlington : Focal Press, 2006. 272 с.
12. Adobe Inc. Substance 3D Painter User Guide. URL: https://helpx.adobe.com/substance-3d-painter/home.html (дата звернення: 15.03.2026).
13. Sherrod A., Jones W. Beginning DirectX 11 Game Programming. Boston : Course Technology, 2012. 432 с.
14. Pranckevičius A. Optimizing the Rendering Pipeline of Animations Running on the GPU. GPU Pro 360 Guide to Shadows / ed. by W. Engel. Boca Raton : CRC Press, 2018. 113 с.
15. Blender Foundation. N-Gons and Mesh Analysis. Blender 4.3 Manual. URL:

https://docs.blender.org/manual/en/4.3/modeling/meshes/mesh_analysis.html (дата звернення: 18.03.2026).

16. Catuhe D. Programming 2D Games. Boca Raton : CRC Press, 2013. 472 с.
17. Maxon. Retopology. ZBrush Documentation. URL: https://help.maxon.net/zbr/en-us/Content/html/user-guide/3d-modeling/topology/Retopo/01-retopo.html (дата звернення: 20.03.2026).
18. Boutsi A.-M., Ioannidis C., Verykokou S. Multi-Resolution 3D Rendering for High-Performance Web AR. *Sensors*. 2023. DOI: https://doi.org/10.3390/s23156893 (дата звернення: 20.03.2026).
19. Blender Foundation. Triangulate Faces. Blender 4.3 Manual. URL: https://docs.blender.org/manual/en/4.3/modeling/meshes/editing/face/triangulate_faces.html (дата звернення: 20.03.2026).
20. Blender Foundation. Recalculate Normals. Blender 4.3 Manual. URL: https://docs.blender.org/manual/en/4.3/modeling/meshes/editing/mesh/normals.html (дата звернення: 21.03.2026).
21. Ahearn L. 3D Game Textures: Create Professional Game Art Using Photoshop. 4th ed. Burlington : Focal Press, 2019. 416 с.
22. Blender Foundation. Apply Object Transformations. Blender 4.3 Manual. URL: https://docs.blender.org/manual/en/4.3/scene_layout/object/editing/apply.html (дата звернення: 22.03.2026).
23. Koulaxidis G., Xinogalos S. Improving Mobile Game Performance with Basic Optimization Techniques in Unity. *Modelling*. 2022. DOI: https://doi.org/10.3390/modelling3020014 (дата звернення: 01.04.2026).
24. Wang S., Zhang J. Research and Implementation of Real-time Render Optimization Algorithm Based on GPU. *Journal of Physics: Conference Series*. 2021. DOI: https://doi.org/10.1088/1742-6596/2136/1/012059 (дата звернення: 01.04.2026).
25. Unity Technologies. Draw Call Optimization. Unity 2022 LTS Documentation. URL: https://docs.unity3d.com/Manual/optimizing-draw-calls.html (дата звернення: 02.04.2026).
26. Unity Technologies. Texture Compression and GPU Memory. Unity 2022 LTS

Documentation. URL: https://docs.unity3d.com/Manual/texture-compression-formats.html (дата звернення: 02.04.2026).

27. Bancroft C. Character Mentor: Learn by Example to Use Expressions, Poses, and Staging to Tell a Story. Burlington : Focal Press, 2012. 240 с.
28. Pixologic. Working with Dynamesh. ZBrush 2022 Manual. URL: https://docs.pixologic.com/user-guide/3d-modeling/modeling-basics/dynamesh/ (дата звернення: 08.04.2026).
29. Pranckevičius A. Rendering Performance Fundamentals. GPU Pro 7 / ed. by W. Engel. Boca Raton : CRC Press, 2016. 231 с.
30. Unity Technologies. Art Asset Best Practice Guide. Unity 2022 LTS Documentation. URL: https://docs.unity3d.com/Manual/HOWTO-ArtAssetBestPracticeGuide.html (дата звернення: 10.04.2026).
31. Gibbons D. Unity Shaders and Effects Cookbook. Birmingham : Packt Publishing, 2013. 386 с.
32. Unity Technologies. Unity Profiler: GPU Usage. Unity 2022 LTS Documentation. URL: https://docs.unity3d.com/Manual/ProfilerGPU.html (дата звернення: 15.04.2026).
33. Cignoni P., Rocchini C., Scopigno R. Metro: Measuring Error on Simplified Surfaces. *Computer Graphics Forum*. 1998. DOI: https://doi.org/10.1111/1467-8659.00236 (дата звернення: 15.04.2026).
34. Стрельніков, В. І., & Бондарчук, А. П. (2025). Комплексний підхід до інтелектуального управління, моделювання та виявлення мережних аномалій на основі ентропійних та нейромережевих підходів. Телекомунікаційні та інформаційні технології, (2), 100-107. https://orcid.org/0000-0001-5124-5102
35. Lagarde S., de Rousiers C. Moving Frostbite to Physically Based Rendering. SIGGRAPH 2014 Course Notes. URL: https://seblagarde.wordpress.com/2015/07/14/siggraph-2014-moving-frostbite-to-physically-based-rendering/ (дата звернення: 18.04.2026).
36. Бондарчук, А. П., Олейніков, І. А., & Бажан, Т. О. (2024). Застосування методів машинного навчання до управління 3D принтером. Телекомунікаційні та

інформаційні технології, (1), 4-15. https://doi.org/10.31673/2412-4338.2024.010415

37. Співак, С., Бондарчук, А., & Черевик, О. (2025). AI-система для професійної орієнтації у сфері 3D-графіки. Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 3(31), 698-709. https://doi.org/10.28925/2663-4023.2025.31.1065

38. Співак, С. М., Білоус, В. В., Горбатовський, Д. В., & Бондарчук, А. П. (2025). Adapting education to the 3D graphics market using AI. Телекомунікаційні та інформаційні технології, 89(4), 215-221. https://doi.org/10.31673/2412-4338.2025.048925

39. Машкіна, І. В., Абрамов, В. О., Носенко, Т. І., Іваніченко, Є. В., & Яскевич, В. О. (2024). Навчально-методичний посібник до виконання і захисту кваліфікаційної роботи бакалавра спеціальностей 122 Комп'ютерні науки для здобувачів вищої освіти денної форми навчання.

ДОДАТОК А

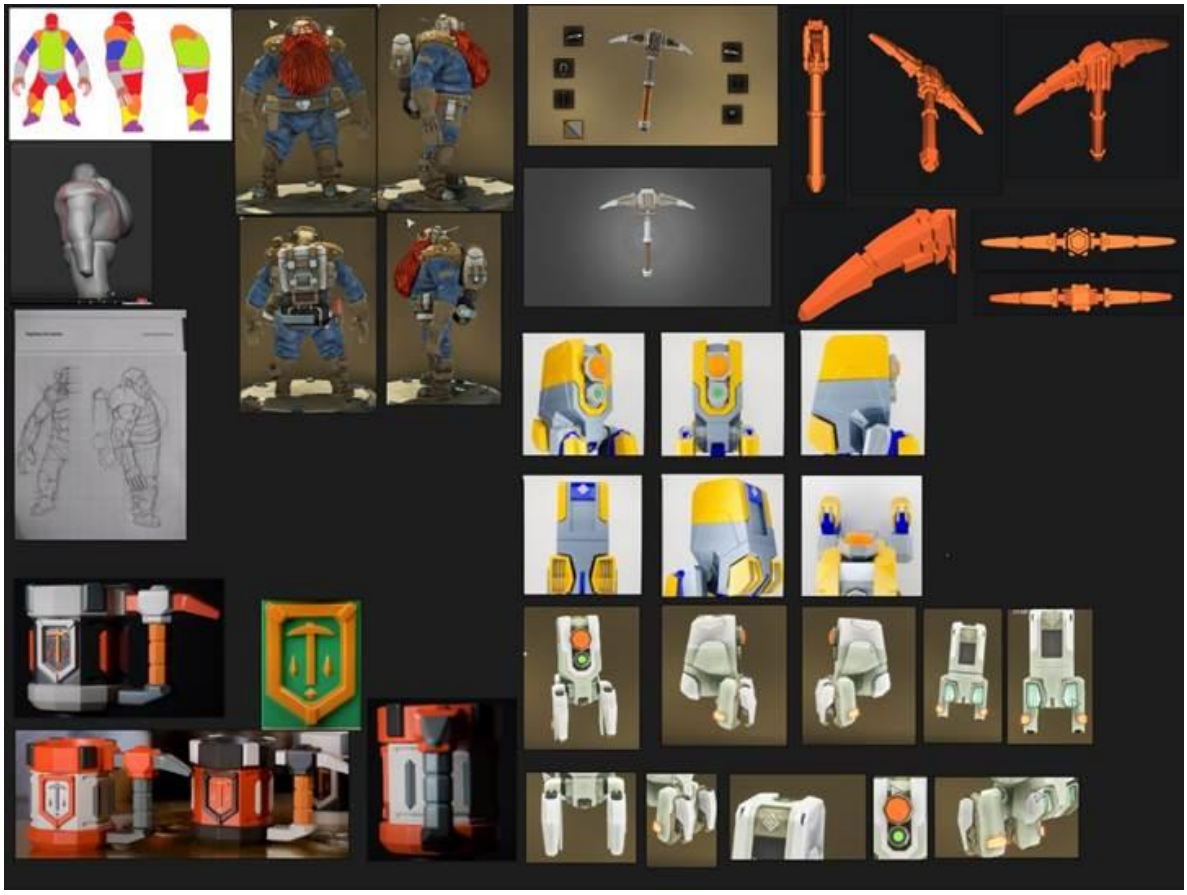

Рисунок А.1. Дошка референсів у PureRef для чотирьох об'єктів дослідження

ДОДАТОК Б

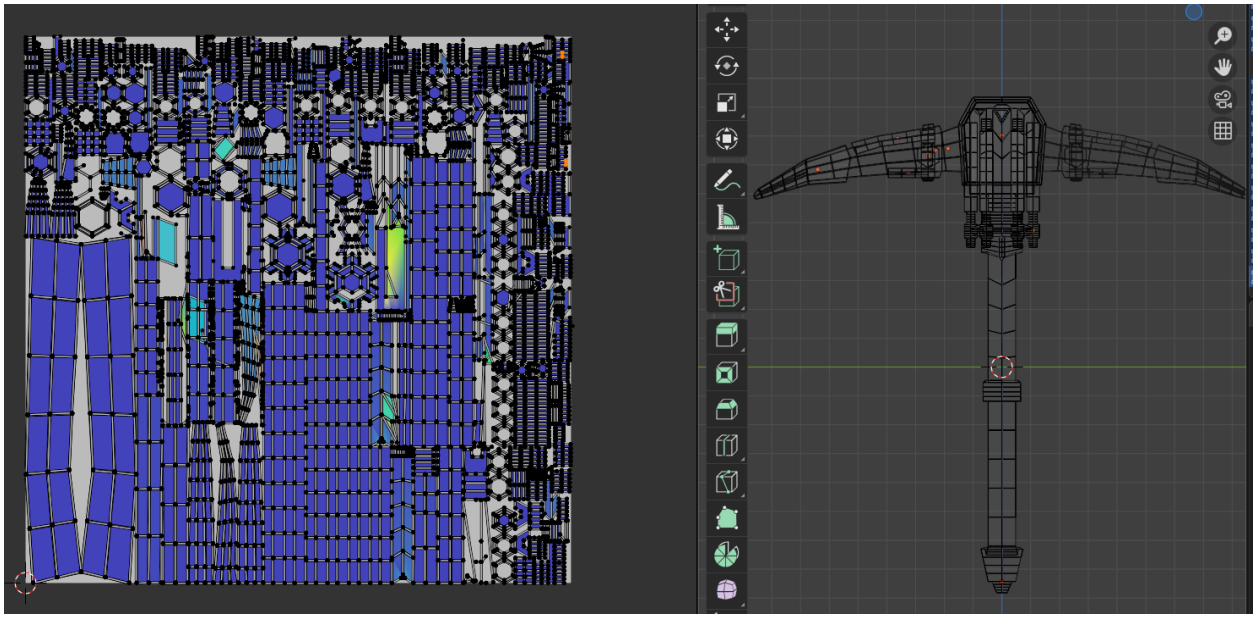

Рисунок Б.1. UV лист неоптимізованого кайла

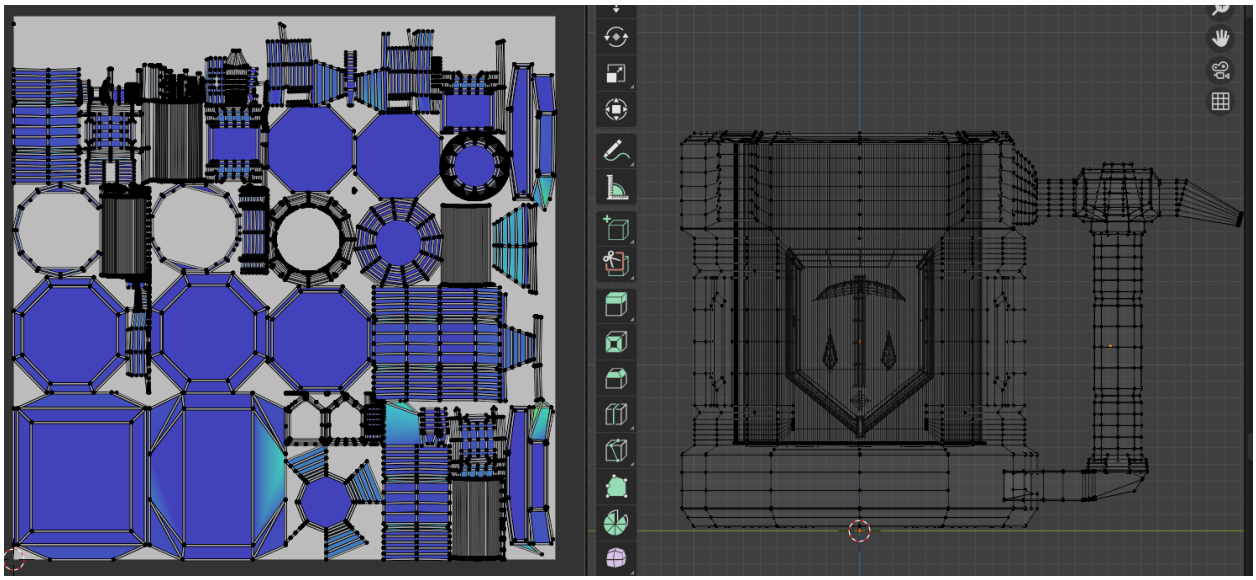

Рисунок Б.2. UV лист неоптимізованого кухля

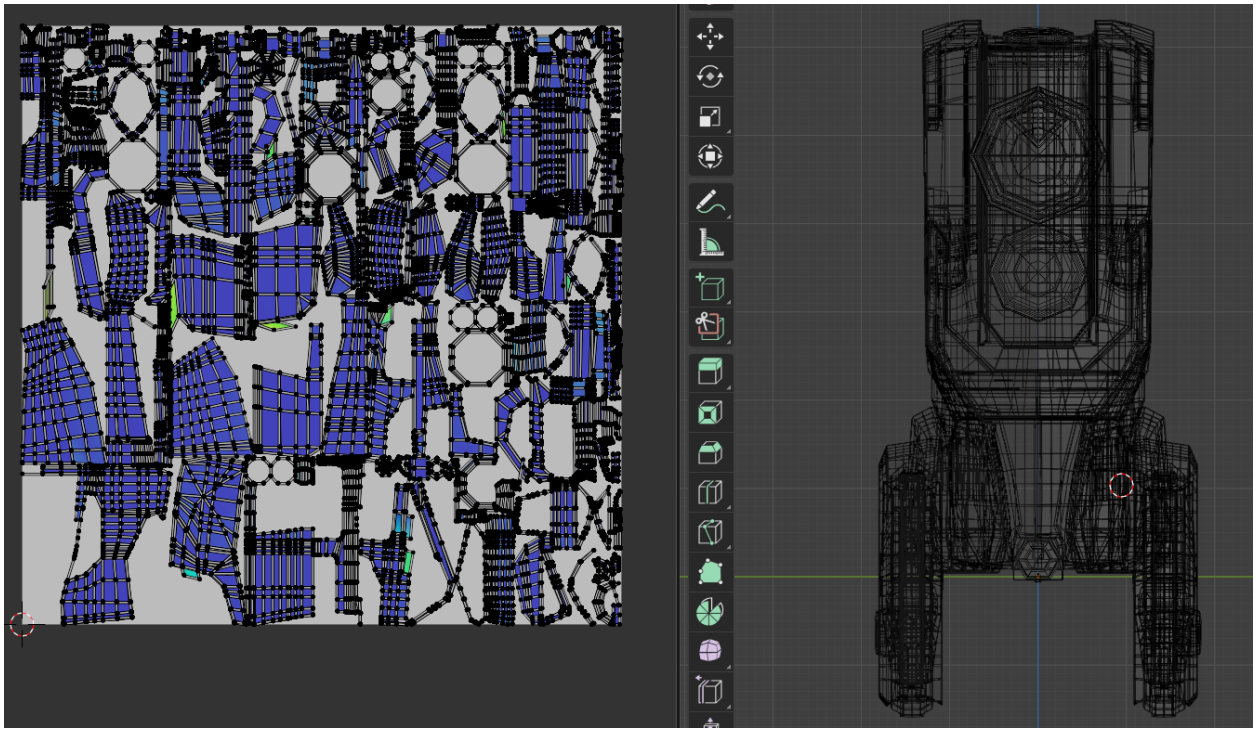

Рисунок Б.3. UV лист неоптимізованого Боско

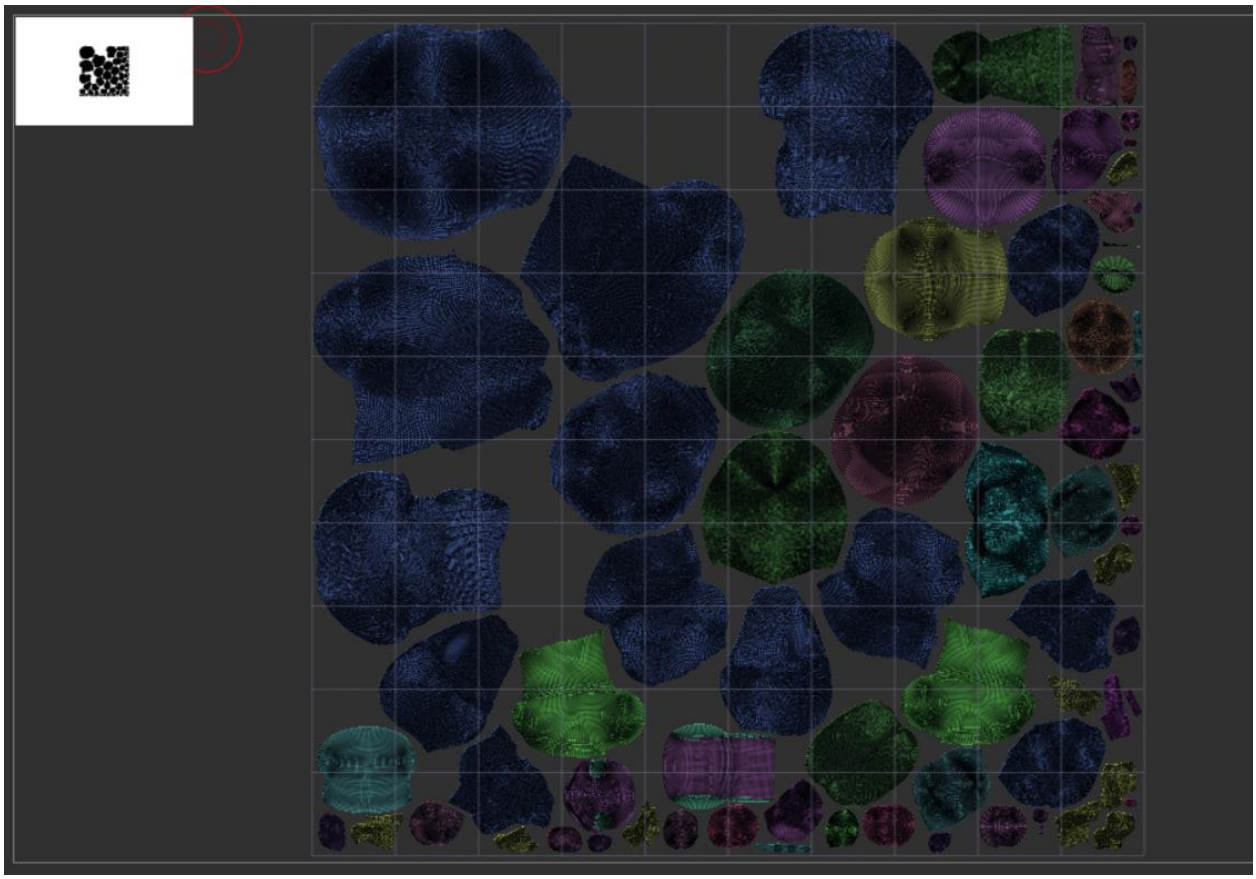

Рисунок Б.4. UV лист неоптимізованого Дворфа

ДОДАТОК В

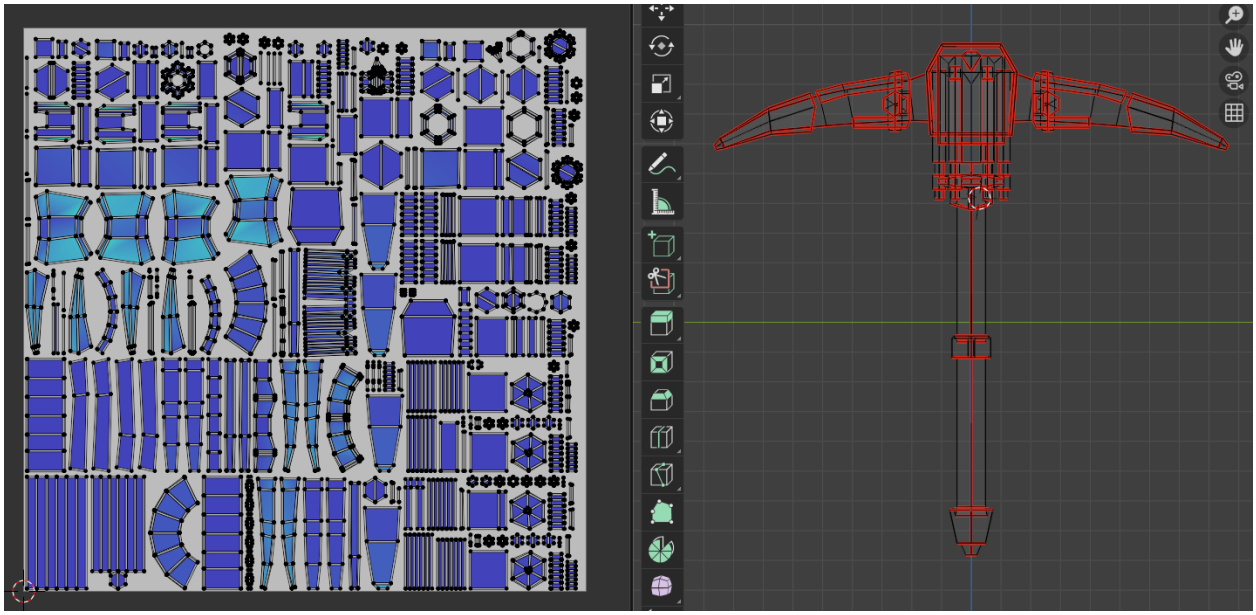

Рисунок В.1. UV лист оптимізованого кайла з mark seam

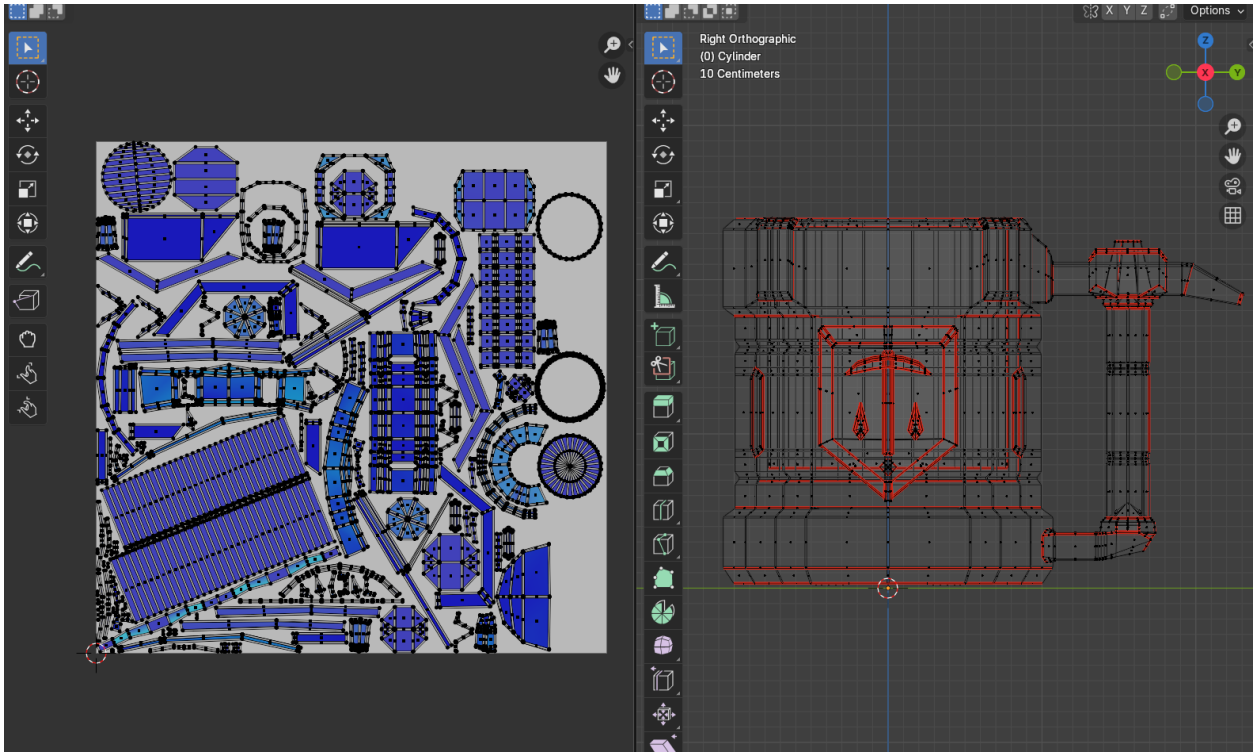

Рисунок В.2. UV лист оптимізованого кухля з mark seam

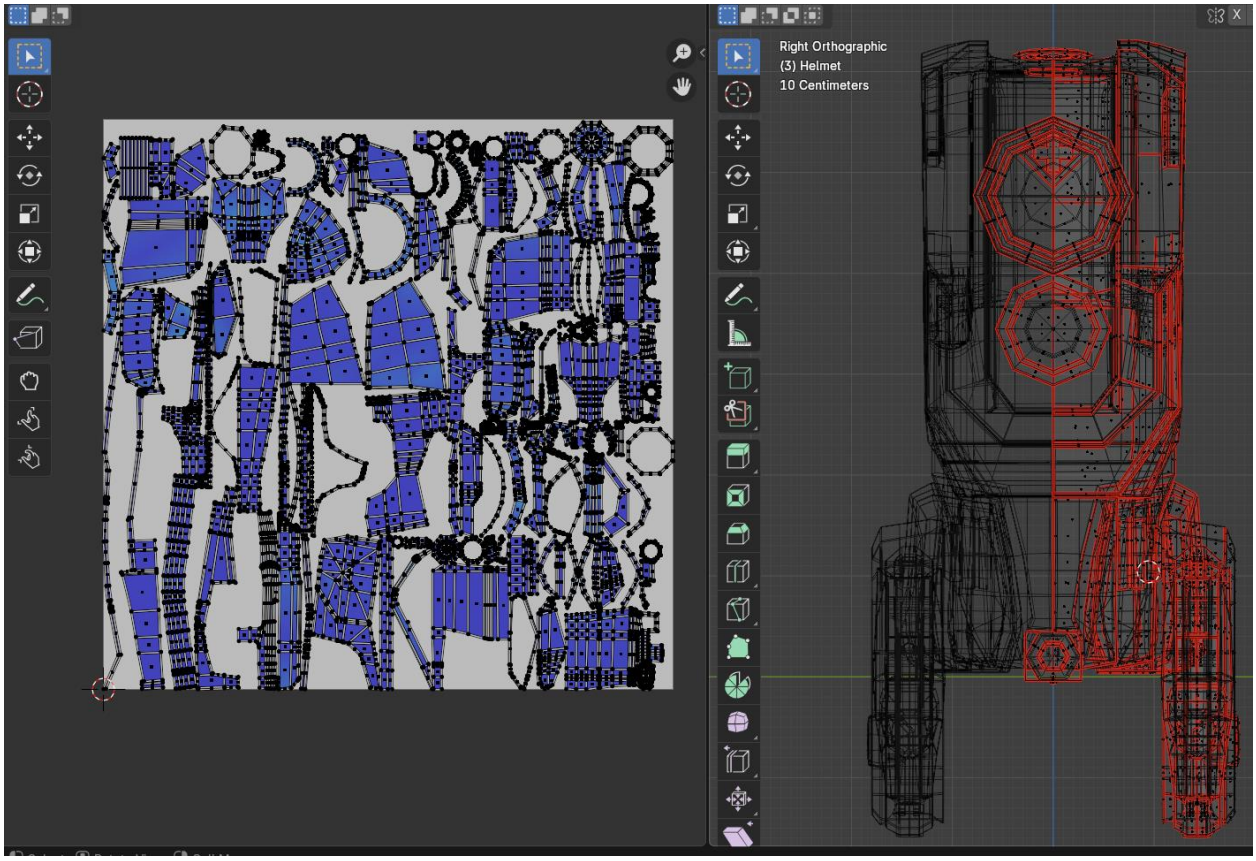

Рисунок В.3. UV лист оптимізованого Боско з mark seam

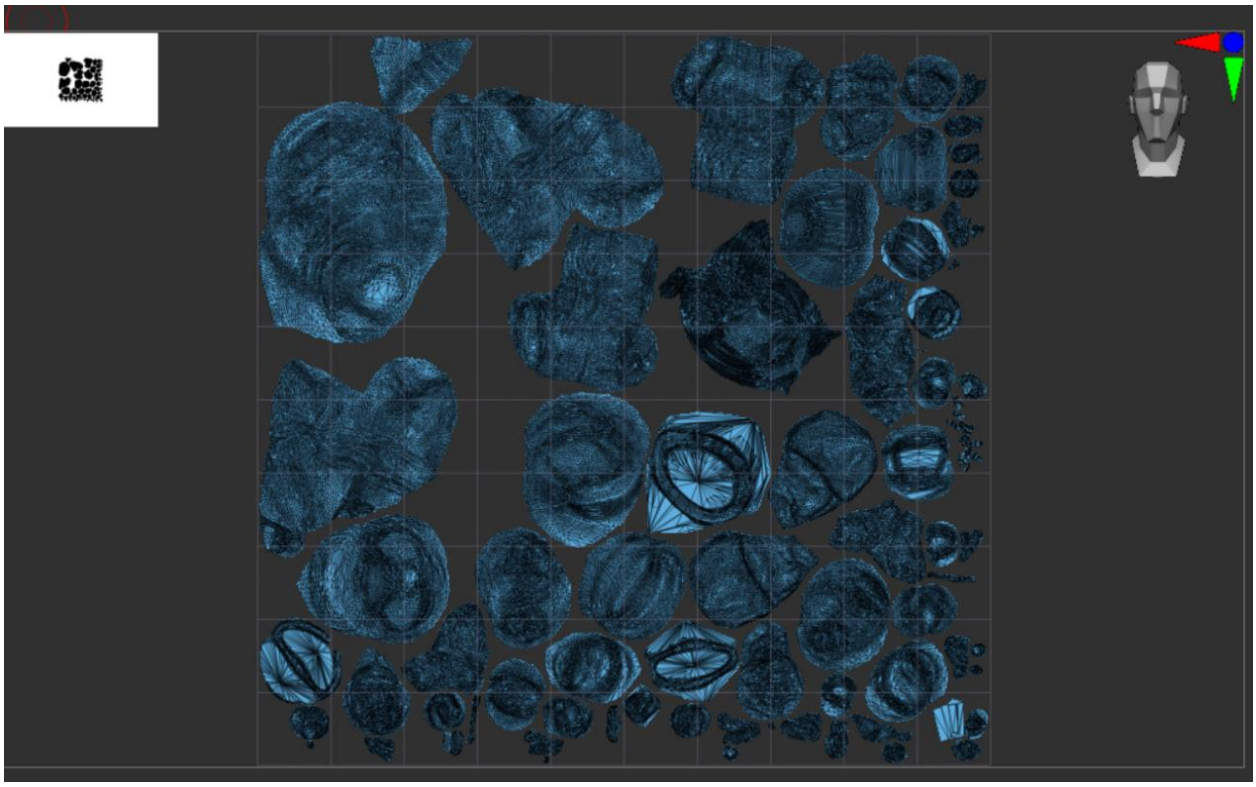

Рисунок В.4. UV лист оптимізованого Дворфа

ДОДАТОК Г

Рисунок Г.1. Порівняльний рендер неоптимізованого кухля (ліворуч) та оптимізованого кухля (праворуч)

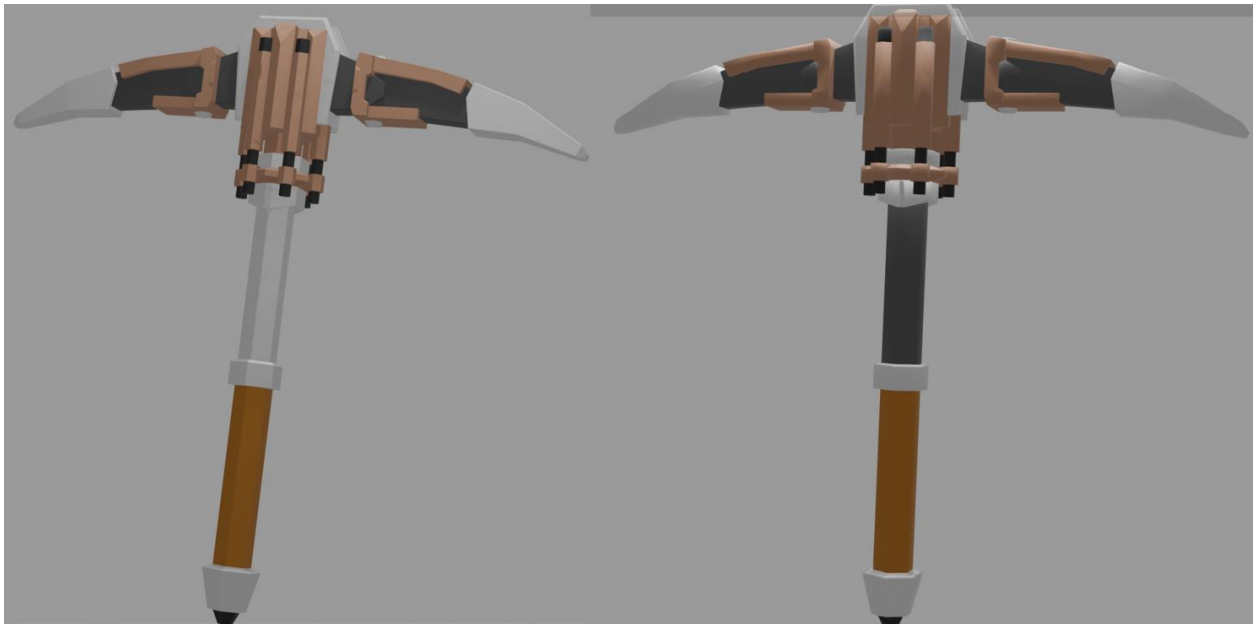

Рисунок Г.2. Порівняльний рендер неоптимізованого кайла (ліворуч) та оптимізованого кайла (праворуч)

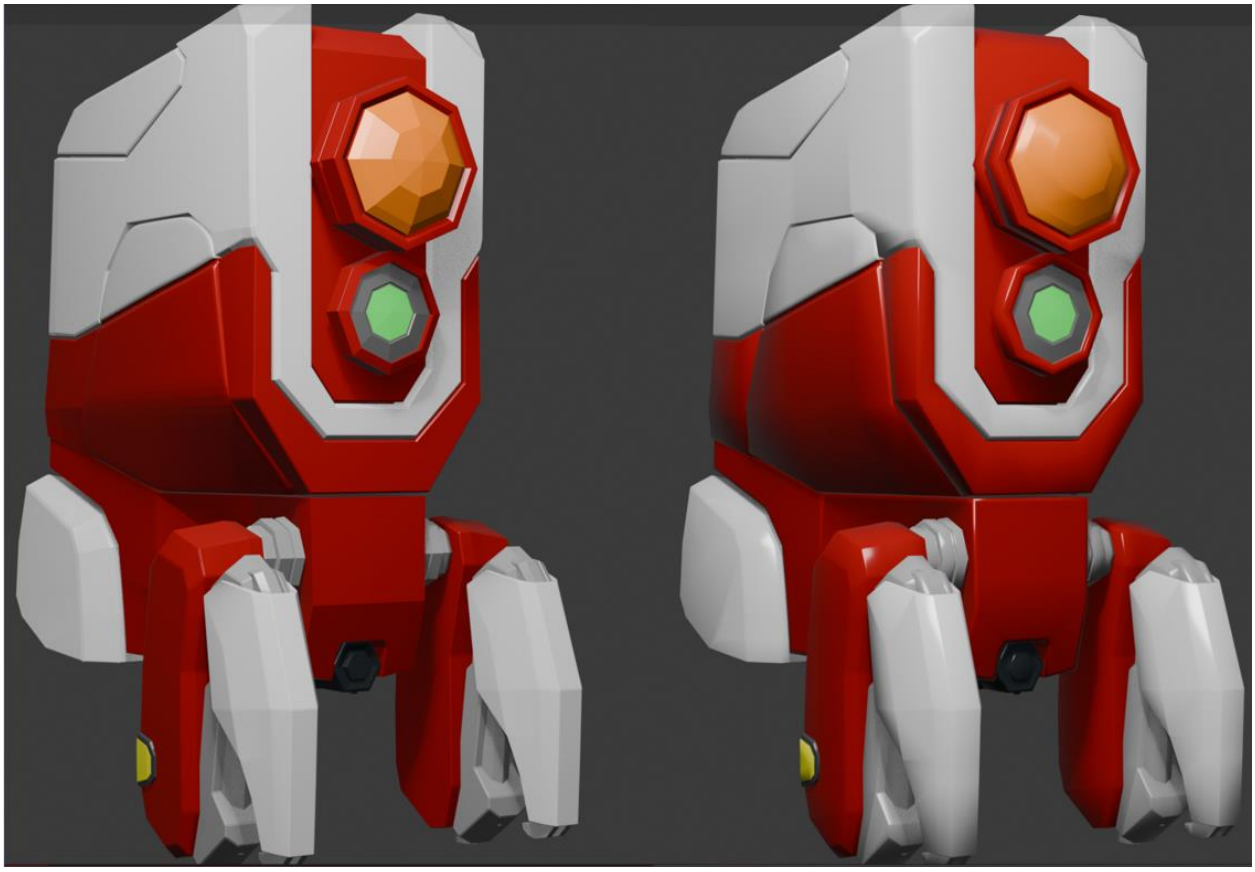

Рисунок Г.3. Порівняльний рендер неоптимізованого Боско (ліворуч) та оптимізованого Боско (праворуч)

ДОДАТОК Д

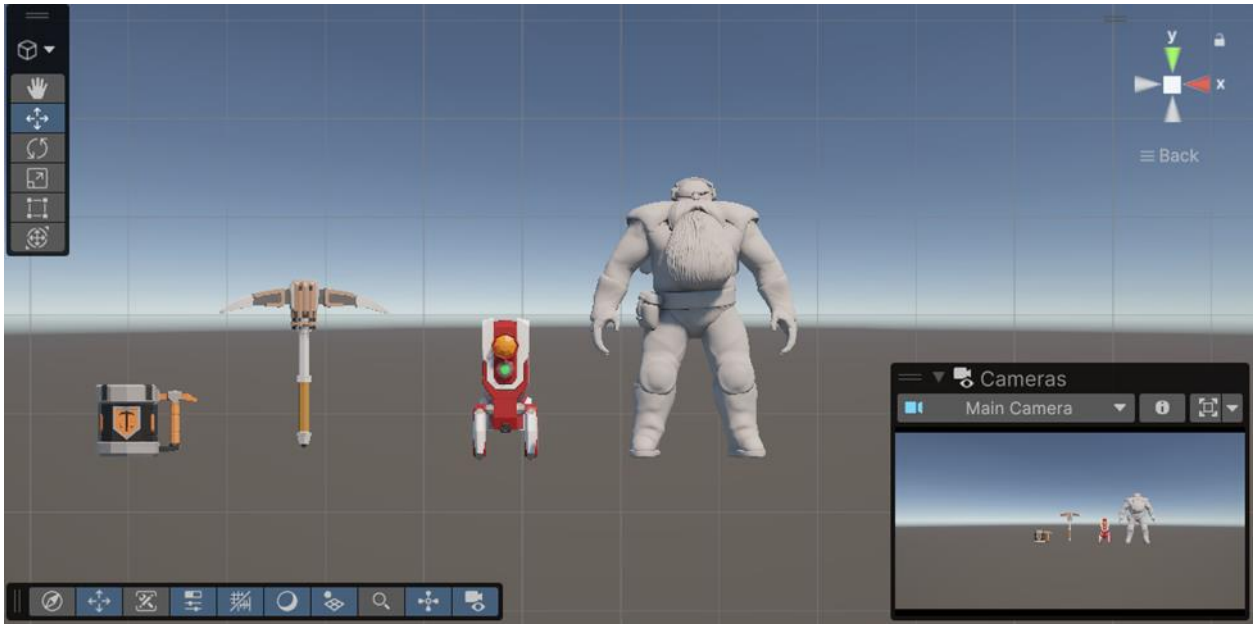

Рисунок Д.1. Неоптимізована сцена з усіма моделями

Рисунок Д.2. Оптимізована сцена з усіма моделями

ДОДАТОК Е

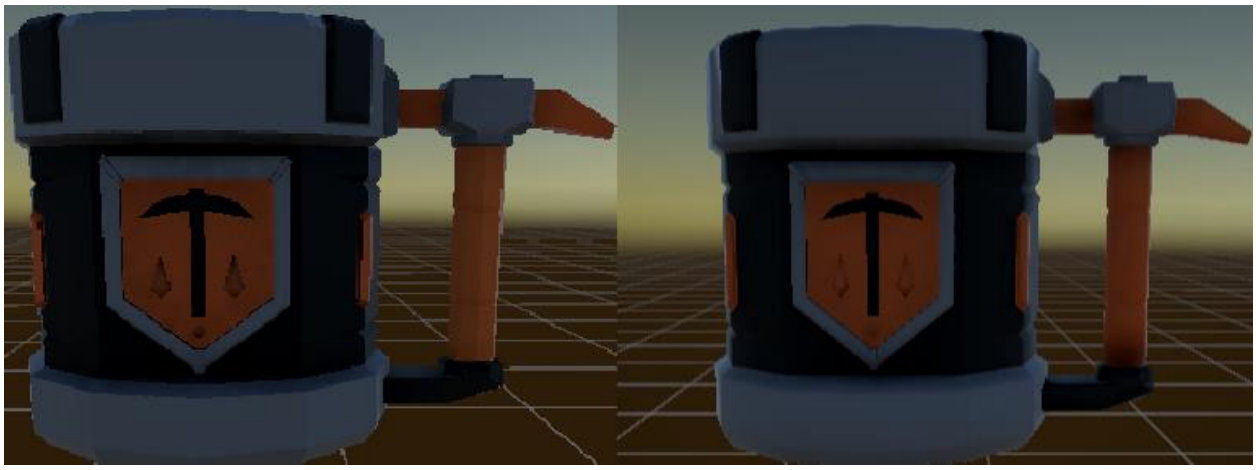

Рисунок Е.1. Порівняння під час запуску в Unity неоптимізованого кухля (ліворуч) та оптимізованого кухля (праворуч)

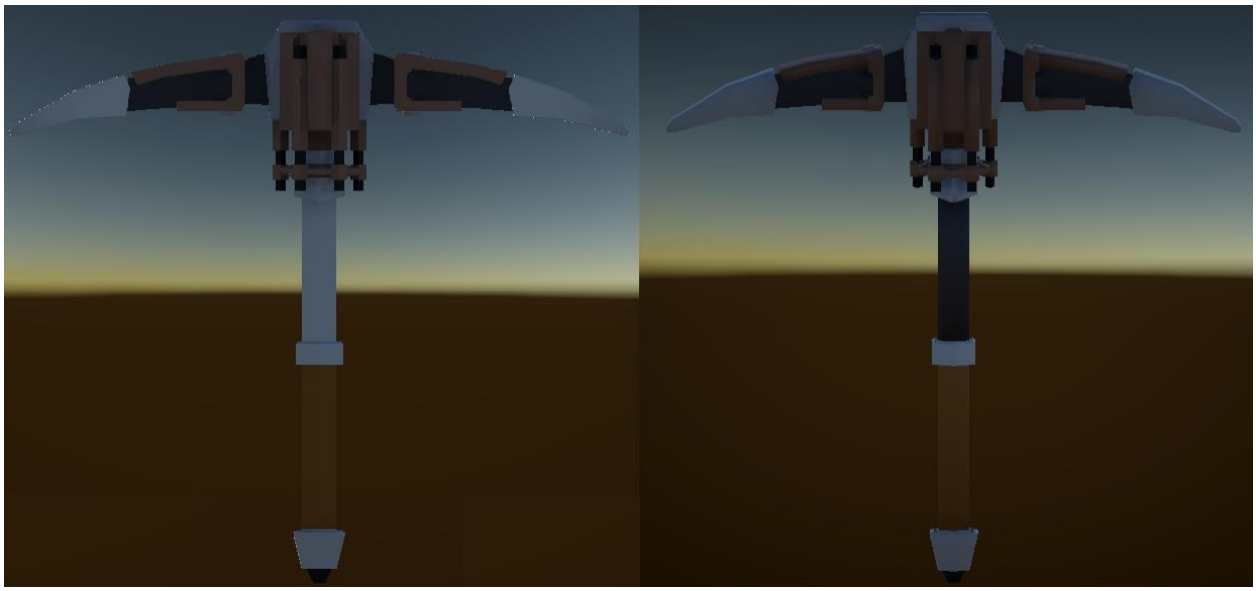

Рисунок Е.2. Порівняння під час запуску в Unity неоптимізованого кайла (ліворуч) та оптимізованого кайла (праворуч)

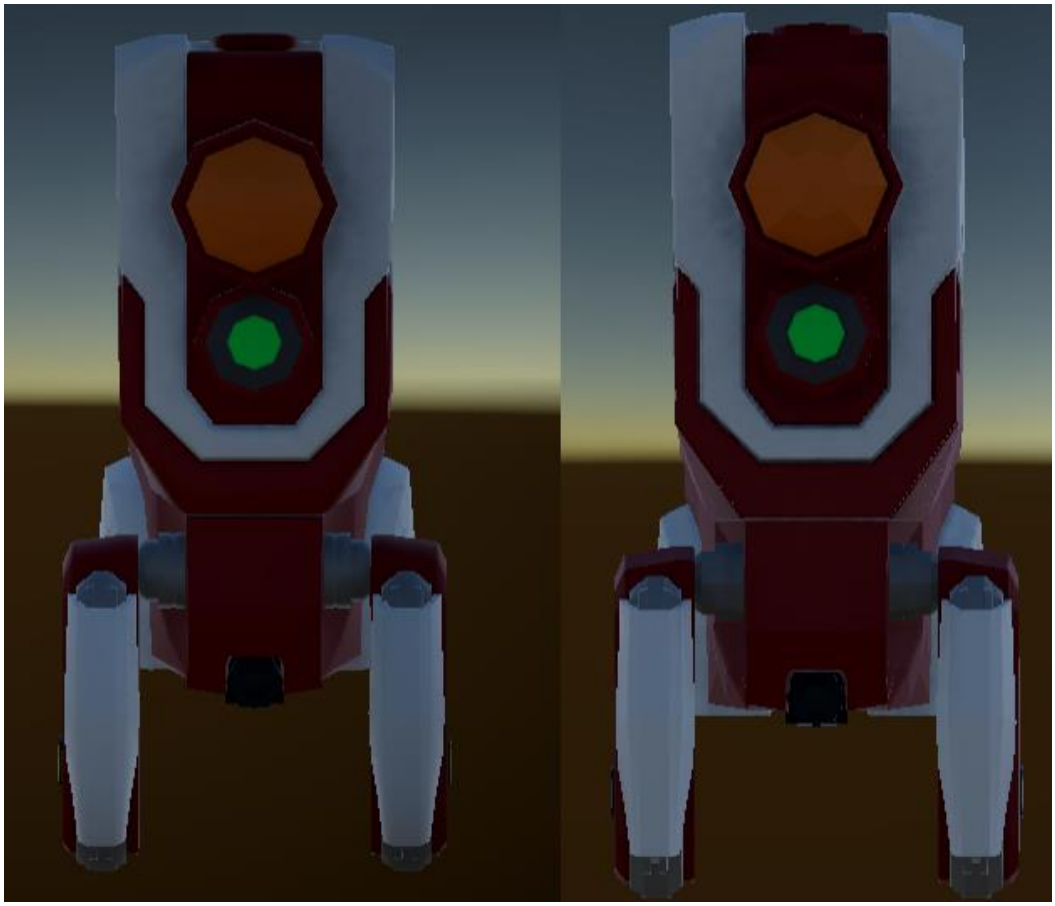

Рисунок Е.3. Порівняння під час запуску в Unity неоптимізованого Боско (ліворуч) та оптимізованого Боско (праворуч)

Рисунок Е.4. Порівняння під час запуску в Unity неоптимізованого Дворфа (ліворуч) та оптимізованого Дворфа (праворуч)